UNIVERSITAT POLITÈCNICA DE CATALUNYA

Programa de Doctorat:

AUTOMATITZACIÓ AVANÇADA I ROBÒTICA

Tesi Doctoral

SEQUENCING IN MIXED MODEL NON-PERMUTATION FLOWSHOP PRODUCTION LINES USING CONSTRAINED BUFFERS

Gerrit Färber

Directora: Anna Maria Coves Moreno

Institut d'Organització i Control de Sistemes Industrials

Desembre del 2006

Acknowledgements

First of all, I would like to express my gratitude to my supervisor Dr. Anna Coves for providing me the opportunity, the necessary dialog, the motivating support and her scientific guidance to accomplish this thesis and to Dr. Hao Niu who triggered my initial interest in the thesis' subject.

I would like to acknowledge the financial support granted from the *Universitat Politècnica de Catalunya* and the project *DELIMER* with partial support from the Ministry of Science and Technology of Spain, and the Funding for Regional Research DPI2004-03472.

Furthermore, I direct my thanks to the *Institut d'Organització i Control de Sistemes Industrials* where I felt cordially welcomed. The high motivation of its members makes the IOC a truly special place to work and study. Special thanks to Dr. Raúl Suárez for his work as a tutor and for his clarifications at all times. Also to Dr. Luis Basañez, Dr. Rafael Pastor and Dr. Enric Fossas which in some way or the other facilitated my work.

Thanks also to Dr. Cecilio Mar and Dr. Laia Ferrer for facilitating the first contact with the *Kent Business School* and special thanks to Dr. Said Salhi for the very inspiring discussions and for receiving me at the *Centre of Heuristic Optimisation*. Also thanks to Dr. Gábor Nagy as coexaminer.

Many thanks also to the staff of administration Carme Martínez, Marta Fuentes and Noemí Zapata, the library staff, Pilar Bombín, and the technical staff Vicenç Roig and, back then, Albert Riego. Without their support this work would not have been finished by now.

Over the years I have been fortunate to meet many dear people who have supported me with their expertise, encouragement, and, most importantly, their friendship. In particular, I wish to thank Ericka, Emmanuel and Wendy, Patrick and Raymundo. I also would like to express my deep gratitude to Deborah for her love, understanding, support and encouraging words. Not to forget Adolfo, Alberto, Arif, Israel, Jordi, Joseph, Leopold, Liliana, Martino, Norberto, Orlando, Shane, Ricardo and Virginia. Special thanks also to Dr. Ramón Barberán, Gerda Schneider and Dr. Klaus Weinhold for encouraging me on this long way.

Needless to say that this section would not be complete without expressing my deep gratitude to Ariane, Thomas and Maren for their support and for enduring this long process with me.

I would like to dedicate this thesis to my parents,
Ute and Eduard, who always provided me with
hope, confidence, and the necessary motivation.

Abstract

In the classical production line, only products with the same options were processed at once. Products of different models, providing distinct options, were either processed on a different line or major equipment modifications were necessary. For today's production lines approaches, considering more flexibility, are required which results more and more in the necessity of manufacturing a variety of different models on the same line, motivated by offering a larger variety of products to the client. Furthermore, with the Just-In-Time philosophy, the stock and with that the expenses derived from it, especially for finished products, are considerably reduced and lead to the case in which a production with batches is no longer favorable.

Taking into account this panorama, the simultaneous production of distinct products or models in the same line, without batches, lead to an increased importance and at the same time the logistic complexity enlarges. The decision-making in sequencing and scheduling become essential.

Various designs of production lines exist which permit resequencing of jobs within the production line: using large buffers (Automatic-Storage-and-Retrieval-System) which decouple one part of the line from the rest of the line; buffers which are located offline; hybrid or flexible lines; and more seldom, the interchange of job attributes instead of physically changing the position of a job within the sequence. Resequencing of jobs within the line is even more relevant with the existence of an additional cost or time, occurring when at a station the succeeding job is of another model, known as setup cost and setup time.

The present thesis considers a flowshop with the possibility to resequence jobs between consecutive stations. The buffers are located offline either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size.

Following the extensive *State-of-the-Art*, which led to the problem under study, a *Novel Classification of Non-permutation Flowshops* is proposed. This classification was indispensable, due to the lack of an adequate classification for flowshop production lines that would consider the diversity of arrangements which permit resequencing of jobs within the production line. Furthermore, distinct formulations are presented: an exact approach, utilizing *Constrained Logic Programming* (CLP), various hybrid approaches, based on *CLP*, and a heuristic approach, utilizing a *Genetic Algorithm* (GA).

During the course of this work, the realized studies of performance demonstrate the effectiveness of resequencing. The results of the simulation experiments reveal the benefits that come with a centralized buffer location, compared to the intermediate buffer location.

The considered problem is relevant to various flowshop applications such as chemical productions dealing with client orders of different volumes and different sized resequencing tanks. Also in productions where split-lots are used for engineering purpose, such as the semiconductor industry. Even in the production of prefabricated houses with, e.g., large and small walls passing through consecutive stations where electrical circuits, sewerage, doors, windows and isolation are applied.

Resumen

En una línea de producción clásica, solamente se producían productos con las mismas opciones. Para la fabricación de variaciones del mismo producto básico se utilizaba una línea diferente o eran necesarias modificaciones importantes de la maquinaria. En los últimos años se ha visto acrecentada la necesidad de considerar métodos que permitan más flexibilidad ofreciendo una mayor variedad de productos al cliente. En general estos métodos consisten en producir diferentes tipos de productos en una misma línea de producción. Además, con la filosofía de Just-In-Time, los stocks y sus costes derivados, especialmente el stock de productos acabados, se reducen considerablemente y consecuentemente una producción con lotes ya no es favorable.

Con este panorama la producción de distintos productos o modelos en la misma línea de forma simultánea, sin lotes, adquiere un gran auge y con ello la complejidad de gestión de la línea aumenta. La toma de decisiones en las fases de *secuenciación* y *programación* se convierte en esencial.

Existen varios diseños de líneas que pueden permitir la resecuenciación, como son: utilizar grandes almacenes (Automatic-Storage-and-Retrieval-System), desacoplar una parte del proceso del resto de la línea; disponer de almacenes con plazas limitadas fuera de la línea; existencia de líneas híbridas o flexibles; posibilitar la división y unión de líneas; o cambiar los atributos de las piezas en vez de cambiar la posición en la secuencia. La resecuenciación de piezas dentro de la línea llega a ser más efectiva cuando se presenta un tiempo o coste adicional, conocido como setup-time y setup-cost, necesario en muchos casos, cuando en una estación, la siguiente pieza es de otro modelo.

Esta tesis considera el caso de una línea de flujo con la posibilidad de resecuenciar piezas entre estaciones consecutivas. Los almacenes están ubicados fuera de la línea y en un primer

paso accesible desde una sola estación (caso del almacén intermedio). A continuación se utilizará un solo almacén, centralizado, accesible desde varias estaciones. En ambos casos se considera que una pieza, debido a su tamaño, quizás no pueda ocupar ciertas plazas del almacén ya sea intermedio o centralizado.

Como resultado del estudio y análisis del *Estado del Arte*, que permitió delimitar el caso a estudiar, se propone una *Novedosa Clasificación de líneas de flujo no permutación*. Esta clasificación era indispensable, debido a que en la literatura actual no se ha clasificado con profundidad este tipo de producción, hasta hoy la clasificaciones existentes no consideran las múltiples opciones que se presentan al incluir la posibilidad de resecuenciar piezas en la línea. La presente tesis presenta distintas formulaciones: un método exacto, utilizando un modelo de programación por restricciones (CLP), varios métodos híbridos, basados en CLP, y un método heurístico, utilizando un Algoritmo Genético (GA).

Durante el curso de este trabajo, los estudios que se han realizado muestran la efectividad de resecuenciar. Los resultados de los experimentos simulados muestran los beneficios que sumergen con un almacén centralizado, comparado con los almacenes intermedios.

El problema considerado es relevante para una variedad de aplicaciones de líneas de flujo como es el caso de la industria química, donde los pedidos de los clientes tienen diferentes volúmenes y en la misma línea existen tanques de diferentes volúmenes para resecuenciar. También, en líneas en las cuales se utilizan lotes divididos (split-lot) con el fin de investigar variaciones en los procesos, así como en la industria de semiconductores, o en la producción de casas prefabricadas, donde fabrican paredes grandes y pequeñas que pasan por estaciones consecutivas y en las que se instalan circuitos eléctricos, tuberías, puertas, ventanas y aislamientos.

Contents

Acknowledgements **I**

Abstract **III**

Resumen **V**

Contents **VII**

1 Introduction and Scope of the Thesis **1**

1.1 Justification 4

1.2 Objectives 5

1.3 Problem Definition 6

1.4 Nomenclature 8

2 State of the art **13**

2.1 Sequencing in flowshops 13

2.1.1 Objective Functions 13

2.1.2 Definition of the ”basic” flowshop 17

2.1.3 Diversity of flowshops 18

2.1.4 Characteristics for solving flowshop problems 20

2.1.5 Setup cost/time in flowshops 23

2.2 Resequencing in flowshops 25

2.2.1 Objectives of resequencing 26

2.2.2 Methods for resequencing 26

2.2.3 Flexible, hybrid or compound flowshop (FFS) 30

2.2.4 Merging and splitting (FS-PL) 31

2.2.5 Re-entrant flowshops 31

2.2.6 Change of job attributes (no physical change) 31

2.2.7 Undesired resequencing 32

2.2.8 Related work on resequencing 32
2.2.9 Summary 32
2.3 Static and dynamic demand 38
2.3.1 Static demand 38
2.3.2 Dynamic demand 39
2.3.3 Transition from pure static to dynamic demand 40
2.3.4 Solution techniques 41
2.4 Optimization methods 42
2.4.1 Elements of optimization 43
2.4.2 Complexity of problems 45
2.4.3 Exact methods 45
2.4.4 Approximate methods 47
2.4.5 Evaluation of the performance of a heuristic 50
2.4.6 Literature providing input data 50
2.5 Summary 51
2.6 Conclusions 53

3 Novel Classification of Non-permutation Flowshops 55
3.1 General notation 56
3.2 Station environment (α) 56
3.2.1 Station arrangement 56
3.2.2 Operating properties 59
3.3 Job processing environment (β) 60
3.3.1 Demand 60
3.3.2 Time and cost restrictions 61
3.3.3 Processing restrictions 62
3.4 Objectives (γ) 63
3.4.1 Purpose of the optimization 63
3.4.2 Objective function 64
3.5 Resuming table 69
3.6 Conclusions 70

4 Exact Approach: CLP 71
4.1 Index definition 72
4.2 Parameter definition 72
4.3 Variable definition 74
4.4 Unconstrained resequencing 75

4.4.1 Non-permutation flowshop . . . 75
4.4.2 Unnecessary job changes . . . 76
4.4.3 Setup cost . . . 77
4.4.4 Setup time . . . 78
4.5 Constrained resequencing . . . 78
4.5.1 Permutation flowshop . . . 78
4.5.2 Intermediate resequencing buffer . . . 79
4.5.3 Centralized resequencing buffer . . . 83
4.6 Computational enhancement . . . 85
4.6.1 Imposing start time of jobs . . . 85
4.6.2 Reduction of variable size . . . 87
4.7 Final formulation . . . 88
4.7.1 Redefinitions . . . 88
4.7.2 Intermediate resequencing buffer . . . 89
4.7.3 Centralized resequencing buffer . . . 92
4.8 Conclusions . . . 94

5 Experimentation CLP 95
5.1 Basic arrangement . . . 95
5.2 Alternative arrangements (Hybrid Approaches) . . . 96
5.2.1 Analysis on semi dynamic demand . . . 97
5.2.2 Cascading . . . 98
5.2.3 Multistart cascading . . . 99
5.3 Mnemonics . . . 105
5.4 Adjustment of parameters and selection of arrangement . . . 105
5.5 Limited flexibility . . . 108
5.6 Experimentation . . . 110
5.6.1 Intermediate versus centralized buffer location . . . 110
5.6.2 Difference in physical size of jobs . . . 112
5.6.3 Semi dynamic demand . . . 114
5.7 Conclusions . . . 115

6 Heuristic Approach: Genetic Algorithm 117
6.1 General outline and terminology . . . 118
6.2 Data structure . . . 120
6.2.1 Random number generation . . . 120
6.2.2 Parameter and variable definition . . . 121

6.2.3 Reduction of chromosome size . . . 121
6.2.4 Population information . . . 123
6.2.5 Initial population . . . 124
6.3 Schedule . . . 124
6.3.1 Parameter and variable definition . . . 124
6.3.2 Schedule calculation . . . 125
6.3.3 Buffer occupation . . . 126
6.4 Fitness function . . . 130
6.5 Genetic operators . . . 131
6.5.1 Validation . . . 131
6.5.2 Inheritance . . . 133
6.5.3 Crossover . . . 133
6.5.4 Mutation . . . 138
6.6 Condition for termination of Genetic Algorithm . . . 141
6.7 Cascade of Genetic Algorithm . . . 142
6.8 Preliminary analysis of parameters . . . 143
6.9 Analysis and adjustment of parameters . . . 149
6.10 Time performance analysis . . . 152
6.11 Conclusions . . . 158

7 Experimentation Genetic Algorithm 159

7.1 General frame . . . 160
7.2 Setup considerations . . . 160
7.3 Intermediate versus centralized buffer location . . . 162
7.4 Number of buffer places . . . 164
7.5 Difference in physical size of jobs . . . 165
7.6 Distribution of data . . . 167
7.6.1 Linear distribution . . . 167
7.6.2 Spreading . . . 169
7.6.3 Normal distribution . . . 171
7.7 Semi dynamic demand . . . 172
7.8 Paced production line . . . 175
7.9 Benchmark comparison . . . 176
7.9.1 Watson, Barbulescu, Whitley, Howe . . . 176
7.9.2 Brucker, Heitmann, Hurink . . . 182
7.10 Conclusions . . . 188

8 Performance Comparison: Exact versus Heuristic Approach 191
8.1 General frame . . . 191
8.2 Difference in physical size of jobs . . . 191
8.3 Semi dynamic demand . . . 193
8.4 Applicability of CLP versus Genetic Algorithm . . . 194
8.5 Conclusions . . . 195

9 Conclusions, contributions and future work 197
9.1 Main results . . . 197
9.2 Contributions . . . 200
9.3 Future work . . . 202

A Appendix 203
A.1 Nomenclature of parameters for flowshops (alphabetically sorted) . . . 203
A.2 Experimentation CLP . . . 208
A.3 Heuristic Approach: Genetic Algorithm . . . 215
A.3.1 Preliminary analysis of parameters . . . 215
A.3.2 Performance analysis . . . 216
A.4 Experimentation Genetic Algorithm . . . 219
A.4.1 Setup considerations . . . 219
A.4.2 Intermediate versus centralized buffer location . . . 221
A.4.3 Number of buffer places . . . 221
A.4.4 Difference in physical size of jobs . . . 222
A.4.5 Distribution of data . . . 223
A.4.6 Semi dynamic demand . . . 225
A.5 Paced flowshop production line . . . 226
A.5.1 Benchmark comparison for Watson, Barbulescu, Whitley, Howe . . . 227
A.5.2 Benchmark comparison for Brucker, Heitmann, Hurink . . . 235

List of Figures 239

List of Tables 243

References 256

Index of first Authors 257

1 Introduction and Scope of the Thesis

The production line structured as *flowshop* requires all jobs (products) to visit the workstations in the same sequence. The current market demands greater flexibility and variety of products together with the reduction of life cycles. This leads to the use of multi model or *mixed model* production lines. In the multi model production the products form lots of the same model, whereas in the mixed model production the job sequence may be arbitrary.

Each station of the production line performs different tasks. The design of the production line considers the conceptual architecture of the production line and roughly assigns these tasks to the stations, subject to technological precedence relations and is a on long term decision. Then, the balancing problem firmly assigns the tasks to the stations as ashort term decision. The balancing problem usually implying the minimization of the station number and the determination of a cycle time, obtained by calculating, e.g., an average of the task times, necessary to assemble the various models. The balancing procedure, in many cases, results in the prevention of the occurrence of bottlenecks so that the final production line will not experience stoppage, and unnecessary inventory will not accumulate. Studies on the production line balancing problems are numerous and may be object to various criteria like cost-oriented or profit-oriented approaches, as described in the survey of Becker and Scholl, 2006, and in Scholl and Becker, 2006, Scholl, 1995. The survey of Erel and Sarin, 1998, explains different measures for balancing problem with respect to its complexity; furthermore, it gives an extensive classification with related solution procedures, such as heuristics as well as optimum seeking algorithms. Further comprehensive studies are found in Ghosh and Gagnon, 1989.

Once the design of the line is obtained and the line is balanced, it is necessary to achieve a reasonable, if not optimum, order for the jobs to be processed consecutively, known as *sequencing* or *scheduling*. Most of the existing literature mentions the optimization of line balancing and job ordering in a consecutive order and therefore focus on one of the two. Kim and Kim, 2000, present a genetic algorithm, optimizing the two at the same time. It has to be mentioned that in a productive production it actually makes sense having the two problems separated. It would be very inefficient if a minor change in the demand would result in a new order and also in the re-assignation of the tasks to the stations.

The present thesis is located in the field of solution techniques for the sequencing problem of mixed-model flowshop production lines. Such type of production line is found in an increasing number in real production industry, resulting from the increased necessity of customer orientated product spectrums. This implies that orders are no longer accumulated to lots of the same models and then stored until a customer orders the product of this exact model, but are rather produced in production lines allowing the possibility of producing various models at the same time and on short notice. The great majority of published research done in this field limits the solutions to permutation sequences where the order of jobs is determined before the jobs enter the production line and maintain unchanged until the end of the line.

Job/Station	I_1	I_2	I_3	I_4
J_1	7	1	1	7
J_2	1	7	7	1

Table 1.1: Processing time of two jobs, to be processed on a four station flowshop.

In the case of more than three stations and with the objective to minimize the makespan, a unique permutation is no longer optimal. As a simple example for the possible savings which may already occur in the case without any setup considerations, table 1.1 shows the processing time of two jobs which are to be processed at four stations. Figure 1.1 shows the optimal makespan, being $c_{max} = 18$ for the non-permutation sequence. The permutation sequence would result in a makespan of $c_{max} = 23$.

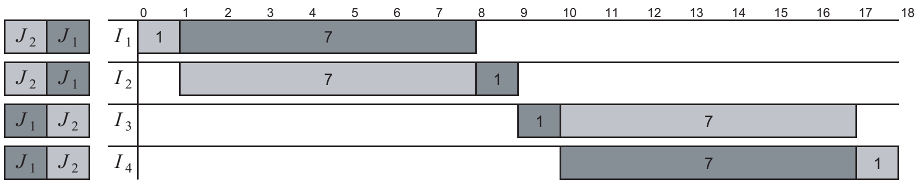


Figure 1.1: The non-permutation sequence leads to a makespan of $c_{max} = 18$, which is achieved by changing the job sequence between station 2 and station 3.

In Potts et al., 1991, and Liao et al., 2006, a study of the benefit of using non-permutation flowshops is presented. Furthermore, there exist various designs of production lines which permit resequencing of jobs: using large buffers (Automatic-Storage-and-Retrieval-System) which decouple one part of the line from the rest of the line; buffers which are located offline; hybrid or flexible lines; and more seldom, the interchange of job attributes instead of physically changing the position of a job within the sequence. Resequencing of jobs within the line is even more relevant with the existence of an additional cost or time, occurring when at a station the succeeding job is of another model, known as setup cost and setup time.

The problem, which is studied in this thesis, considers the possibility of resequencing jobs between selected stations which results in considerable improvements, even more evident when setup cost/time exists. The problem is *NP-hard* and as highlighted by Lahmar et al., 2003, only few resequencing possibilities are necessary in order to achieve the greatest benefit.

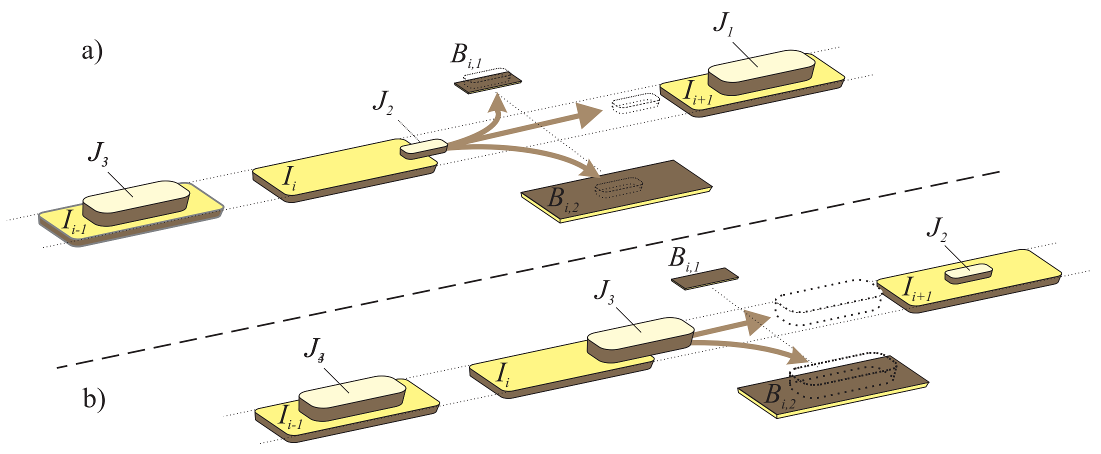


Figure 1.2: Scheme of the considered flowshop. The jobs J_j pass consecutively through the stations I_i. The buffer B_i permits to temporally store a job with the objective of reinserting it at a later position in the sequence. a) Job J_2 can pass through any of the two buffer places $B_{i,1}$ or $B_{i,2}$ of buffer B_i. b) Job J_3 can pass only through buffer place $B_{i,2}$, due to its physical size.

The buffers, which are used in this work, in order to accomplish resequencing, are located off the production line, either accessible from a single station (intermediate case) or from various stations (centralized case), and are furthermore constrained in terms of the number of buffer places and the fact that a job may not be able to be stored in a certain buffer place, due to its extended physical size, see figure 1.2.

1.1 Justification

Mixed model production lines consider more than one model being processed on the same production line in an arbitrary sequence. Nevertheless, the majority of publications in this area are limited to solutions which determine the job sequence before the jobs enter the line and maintain it without interchanging jobs until the end of the production line, which is known as permutation flowshop.

Besides the fact that solution methods in the literature are not numerous, it seems that production lines in the industry are not yet considering the advantages that come with the possibilities of resequencing jobs in a mixed model production. This may either be caused by additional hardware to be installed, like buffers, but also due to extra efforts in terms of logistics complexity.

The conclusion of the literature revision led to the proposal of the present thesis. First of all, and due to the lack of a classification for flowshop production lines that would consider the diversity of arrangements which permit resequencing of jobs within the production line, a novel classification for non-permutation flowshop production lines is introduced.

Furthermore, a new case of non-permutation flowshop is studied in which the possibility to resequence jobs exists between consecutive stations, using buffers which are located off the line, either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size, see figure 1.2.

The considered problem is relevant to various real life flowshop applications which can be found in the industry, such as chemical productions dealing with client orders of different volumes and different sized resequencing tanks. Also in productions where split-lots are used for engineering purpose, such as the semiconductor industry. Even in the production of prefabricated houses with, e.g., large and small walls passing through consecutive stations where electrical circuits, sewerage, doors, windows and isolation are applied.

1.2 Objectives

The doctoral thesis consider both, the determination of a theoretical model for solving a case in the area of flowshop production lines that is not yet considered in the literature, and the extended study of performance, also considering its feasibility with respect to problem size.

Particularly, the following contributions to sequencing in mixed model non-permutation flowshop production lines are treated:

1. *State of the art* of sequencing in mixed model flowshop production lines with special focus on setup cost and setup time and resequencing in static and dynamic context.
2. A *Novel Classification of Non-permutation Flowshops*, based on the notation used by Pinedo, 1995, that also establishes criteria which adequately categorizes flowshops that provide the possibility of resequencing, including a wide scope of resequencing facilities and objectives.
3. *Formulation of a special case of non-permutation flowshop* with characteristics, not yet studied in its totality. An *exact approach* will be formulated, using Constrained Logic Programming (CLP) which will also be used for a hybrid approach, for moderately sized problems. Thereafter, and in order to also solve larger problems, a *heuristic approach*, using a Genetic Algorithm (GA), will be formulated.
4. *Analysis* and *validation* of the developed solution methods will be given in order to demonstrate the effectiveness of resequencing, the intermediate case will be compared with the centralized case, the benefit of constrained resequencing buffers will be discussed, the limitations of the models will be determined with respect to problem size and recommendations will be given for future investigations.

1.3 Problem Definition

The problem under study is a non-permutation flowshop in which there exists the possibility to resequence jobs between consecutive stations, using buffers which are located off the line, either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size, see figure 1.2.

The explicit definition of the problem under study contemplates the following considerations:

1. **Line considerations:**
   In the basic case a set of N jobs have to be sequenced on M stations, arranged in series. Each job has M sets of operations and requires its first set of operation on station 1, its second set on station 2, and so on. A station can process only one job at a time. The set of sequences Π $(\Pi_1, \Pi_2, ..., \Pi_i, ..., \Pi_M)$ describes the order in which the jobs are processed on the M stations; non-permutation sequences are allowed. Processing times are known and constant.

2. **Buffers:**
   The following two buffer types are considered:
   - Finite intermediate buffer between the stations, operated in FIFO mode.
   - In order to permit resequencing there exist a free-access buffer with different properties for the individual buffer places. This leads to a constraint such that not all jobs from the model mix can be stored in all of the places of the buffer. In a concrete case, the sizes of the places of a buffer are different and as a consequence, large jobs can not be freely assigned to the buffer places.

3. **Number and location of constrained buffers:**
   The work of Lahmar et al., 2003, concludes that the biggest amount of cost saving is achieved by implementing only few resequencing buffers in the line. As a consequence a small number of constrained buffers is implemented.
   As for the location of the constrained buffers two arrangements are used:
   - Intermediate buffer accessible from a single station.
   - Centralized buffer accessible from various stations.

4. **Setup cost and time considerations:**
   The problem of sequencing jobs of different models implies that a station may need modification of its arrangements, which may result in an additional cost or time. The problem under study contemplates sequence dependent setup cost and time.

5. **Demand considerations:**
   Two cases are considered:
   - In the static case all required information is known before the first job is processed, furthermore, all jobs are accessible at the first station at any time.
   - In the semi dynamic case, the demand is defined by a series of incoming jobs, which directly enter the first station. Resequencing takes place within the line, using the constrained resequencing buffer. This case is treated briefly as an extension of the static case and is meant to point out the vast benefit for this type of production line.

6. **Objective function consideration:**
   The objective of the optimization is the minimization of the makespan. For the case in which setup considerations are contemplated, the objective is the minimization of the weighted sum of the makespan and the setup cost. In the latter case, the setup time is not concerned with a weight but is indirectly included in the calculation of the makespan.

7. **Resolution methods:**
   Due to the problem complexity an exact approach is formulated, using Constrained Logic Programming (CLP), which can handle moderately sized problems. Thereafter, various hybrid approaches are introduced, based on the CLP, which slightly augment the size of the considered problems. Finally, and in order to also solve larger problems, a heuristic approach, using a Genetic Algorithm (GA), is formulated.

Additionally to the development of the two approaches, performance studies follow which demonstrate the effectiveness of resequencing, compare the intermediate with the centralized case, discuss the benefit of constrained resequencing buffers, and determine the limitations of the models with respect to the problem size.

1.4 Nomenclature

The nomenclature presented here is according to the nomenclature found in the literature, however, the terms used in the literature are not always coherent or unique. In order to prevent misunderstanding and improper use, the nomenclature used here is given. This nomenclature is arranged in such order to give the best understanding. In the appendix A.1 the same nomenclature is presented in alphabetical order for the purpose of a better access during the reading of this thesis.

Pinedo, 1995, arranges these parameters into a triplet $\alpha|\beta|\gamma$ that helps classifying problems that appear in flowshop production lines. The triplet determines the specific problem with α describing the station environment, β providing details of the processing characteristics and constraints and γ containing the objective to be minimized. The first and the second field usually have one entry and the third various or none. The most relevant parameters for flowshop sequencing are presented at next:

Task t: Non-divisible activity which is performed in either station. The balancing problem solves the problem of assigning tasks to stations and therefore often is called the assignment problem. In the Sequencing problem this task assignation is already performed and only stations are considered.

Job j: A part, subassembly or assembly, processed by a station is called job. In a mixed model production line the jobs belong to different models which include different processing times at the stations, depending on the model type. The number of jobs to be processed is N.

Station i: One or more tasks may be assigned to station i. In the basic flowshop problem M stations are aligned in series and all jobs j have to pass the stations in the same order. A station may be open or closed, depending on whether or not the operator working in it is allowed to cross its boundary. In the case of a paced production line, the time to realize the assigned tasks at a station, is defined by the launch interval.

Operation i: Processing of a job in a station is called operation. This operation can include various performed tasks at one and the same station and is determined by the processing time P_{ij}.

Processing time P_{ij}: Also called assembly time, is the time that job j maintains at station i while being processed. Due to the nature of a flowshop, job j that is not processed at station i has to pass through this station with a processing time equal to zero.

Preemptive/Nonpreemptive: Preemptive operation means that processing times may be interrupted and resumed at a later time, even on another station. Furthermore an operation may be interrupted several times. If preemption is not allowed, the operation is called nonpreemptive.

Setup time ST_{efi}: Setup time is concerned if an additional time appears to change the setup of station i, in order to be able to process job $j+1$ which is of model f after job j which is of model e. If the setup time is independent of the model, it can be simply added to the processing time.

Setup cost SC_{efi}: In a similar way, setup cost is concerned if an additional cost appears to change the setup of station i, in order to be able to process job $j+1$ which is of model f after job j which is of model e. If the setup cost is independent of the model, it can be simply added to the processing cost.

Start time s_j: The time job j enters the system is called start time.

Completion time c_j: The time job j exits the system is called completion time and is the completion time on the last station on which it requires processing.

Demand D: The demand describes the total volume of N jobs to be processed. In the scheduling problem the individual jobs can furthermore be specified by start-date and due-date. These values describe the earliest possible point of time to start working on a particular job and when the finished products have to be delivered to the customer. In order to fulfill the due-dates a penalty may be applied for delivering too early or too late. In the sequencing problem it is frequent to release customer orders to production once a certain number of jobs has been accumulated, and then sequence and produce these orders together as a lot, see for example Burns and Daganzo, 1987. The demand in this case is a **static demand** and only depends on the volume of jobs and the objective usually is to minimize the processing time to complete the entire order, called makespan. Whereas a **dynamic demand** implies that the customer orders arrive continuously or at least are not completely determinable beforehand.

Model μ: In the mixed model flowshop several variations, called models, of the same basic product are manufactured. The difference from one model to another may be due to an option that is not applied to all models or likewise in a variation of an option. Therefore the mixed-model sequencing problem consists of the determination of the consecutive order of the models. μ_i determines the model of job i. **Minimal-Part-Set** (MPS): The MPS is denoted by the vector $d(d_1, d_2, ..., d_k)$ which represents a product mix, such that $d_\mu = D_\mu/h$. D_μ being the number of units of model type μ which needs to be assembled during an entire planning horizon and h being the greatest common divisor of $D_1, D_2, ..., D_\mu$. Obviously, h times repetition of the MPS sequence meets the total demand. With the MPS the number of possible sequences is reduced to $D!/(d_1! \cdot d_2! \cdot ... \cdot d_k!)$, Korkmazel and Meral, 2001. The MPS is considered to be a good choice, however, Klundert and Grigoriev, 2001, show that many times reducing the sequence to the one of the MPS does not result in the optimal sequence.

Launch-interval λ: The time between two consecutive jobs entering the production line is called launch-interval. Usually it is a constant value, also called cycle time. A constant launch-interval results in a fixed production rate (production quantity per unit of time).

Job sequence Π_i: The job sequence defines the order of jobs at station i. A job sequence that is the same for all stations is called a **permutation** sequence. In flowshops the station sequence, the order in which the individual jobs visit the stations, is the same for all jobs.

Machine breakdown/maintenance: Machine breakdown/maintenance describes the state of a station which does not permit processing of any job due to failure or failure prevention. In real production systems the breakdowns occur in a stochastic way and can be simulated using the values Mean-Time-Between-Failure (MTBF) and Mean-Time-To-Repair (MTTR).

Precedence: The precedence gives a dependency of jobs in respect to the processing. A job j is said to be predecessor of job k if job j has to be processed before job k. An immediate predecessor then is a job that has to be processed immediately before another job.

Rework operations: The detection of defective jobs may cause either rework operations or removal of the job from the line. The occurrence of defective jobs in the production is of probabilistic nature.

Paced/unpaced production line: In a paced production line the mechanical material handling equipment like conveyor belts couple the stations in an inflexible manner. The jobs are either steadily moved from station to station at constant speed or they are intermediately transferred after processing. The available amount of time for the operation is the same in both cases. In the unpaced line, in contrast, the stations are decoupled by buffers. In a specific case this buffer stores jobs that can not be passed to the downstream station which is still occupied with processing the previous job.

Deterministic-stochastic models: The deterministic model is characterized by the fact that the elements, e.g. processing time, do not involve variation and that the consequences of any given decision can be predicted in a precise manner. The stochastic model is characterized by its explicit recognition of variation and uncertainty, which could exist in one or more of the elements with known probabilistic behavior. This may result, for example, in a variation of the performance of the operator.

Buffer: Buffers were originally introduced between two consecutive stations to decouple them in order to avoid blocking and starving. Buffers are often located before and after bottleneck stations. The reason is that this already critical part of the production usually is the limiting section. In automobile productions buffers of enormous dimensions can be found, which in principle decouple the main successive production sections. This buffer is, furthermore, used to reorder the jobs, available in the buffer, on a large scale.

2 State of the art

2.1 Sequencing in flowshops

The sequencing problem in flowshops appear when variations of the same basic product are produced simultaneously in the same production line. These variations imply that the processing times on the individual stations differ, dependent on the model to be processed. This type of problem is called the mixed model flowshop and is defined by various parameters which reflect the complexity of the possible layouts and the different operation modes of the production line. This chapter gives an overview the diversity of flowshops presented in the literature, common objective functions and finally characteristics of sequencing problems of flowshops.

2.1.1 Objective Functions

Within the sequencing problem of mixed-model production lines a variety of objective functions are to be found, the most common being time and cost orientated objectives. As a basic principle of optimization, the considered solutions are part of a set of feasible solutions and with the use of additional objectives lead to the optimal solution. For example, the method proposed by Dar-El and Cucuy, 1977, minimizes the overall line length and first determines the feasible solutions that result in a non-idle-time schedule.

2.1.1.1 Time orientated objectives

Makespan $c_{\max}$: One of the most common objective functions in sequencing is to minimize the maximum completion time necessary to process the entire demand, called makespan or total production time. The makespan optimization generally ensures high utilization of the production resources, early satisfaction of the customer demand and the reduction of in-process inventory by minimizing the total production run.

Makespan:

$$\max\{c_j | j = 1, ..., N\}$$

Maximum flow time $F_{\max}$: The minimization of the flow time leads to stable and even utilization of resources, rapid turn-around of jobs and the minimization of in-process inventory. French, 1982, mentions that in the case where all release dates are zero, $c_{\max}$ and F_{max} are identical. The weighted flowtime includes a weight related to the station.

Maximum flow time:

$$\max\{c_j - s_j | j = 1, ..., N\}$$

Weighted flow time:

$$\sum_{j=1}^{N} \omega_j (c_j - s_j)$$

Mean flow time $\overline{F}$: Similar to the maximum flow time, the mean flow time leads to stable and even utilization of resources, but tries to average the flow time of all jobs and therefore leads to more balanced flow of jobs.

Mean flow time:

$$\sum_{j=1}^{N} (c_j - s_j)/N$$

Weighted mean flow time:

$$\sum_{j=1}^{N} \omega_j (c_j - s_j)/N$$

Setup time: In a mixed model production, setup time ST_{efi} may occur when at station i a job $j+1$ of model type f follows job j of model type e. Minimizing total setup time, furthermore, tends to decrease the total flowtime. The mean setup time leads to a balanced distribution of setup times for all jobs and therefore results in a more stable and uniform workflow.

Setup time:

$$\sum_{j=1}^{N} ST_{efi}$$

Mean setup time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} ST_{efi}/M$$

Idle time: Idle time I_{ij} at station i occurs when an operator is kept waiting for job j. This may be caused by a job that has not yet arrived, or because an auxiliary operator is still occupied with the job. When setup time occurs, that is separable from the processing time, the operator can benefit from this idle time in order to perform the necessary changes for the next job to be processed. French, 1982, highlights that the mean and the maximum for idle time are taken over the stations rather than over the jobs.

Idle time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}$$

Mean idle time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}/M$$

Utility time: Utility time U_{ij} at station i occurs when an operator has to continue with job $j+1$ before finishing with job j. In this case an auxiliary operator finishes the job; the time the auxiliary operator requires is called utility time. As well as the idle time, here the mean is taken over the stations. The minimization of idle and utility time is, for example, applied by Sarker and Pan, 2001, varying the station length and using individual weights for the calculation of idle and utility time.

Utility time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} U_{ij}$$

Mean utility time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} U_{ij}/M$$

Earliness and tardiness of jobs (E, T): Earliness and tardiness is regarded in the scheduling problem when due times appear. The earliness and tardiness can be considered in different ways, e.g. the maximum value or the mean value. In general it is common practice to consider the minimization of the tardiness as the more important objective of the two, in order to satisfy the client. On the other hand, when jobs are finished early, they have to be stored until shipping which obviously implies an additional cost for storage.

Production rate (PR): The objective of maximizing the production rate is relevant in a production system which uses, e.g., a conveyer belt with fixed distances or time intervals for the jobs which are launched into the production line. Besides a constant production rate, cases can be found where a variable production rate is used for further optimization, as e.g. Bau, .

Deviation: In general for all of the above mentioned time oriented objectives it is possible to use the deviation, or the squared deviation, over stations or over jobs, in order to equalize the deviation and to avoid solutions that provide extreme values for single stations or jobs, see e.g. Bukchin, 1998.

Regular/non-regular objective: Baker, 1974, explains that an objective function is called regular objective if the function increases only if at least one of the completion times of the jobs increases. Surveys on non-regular objective functions are presented by Baker and Scudder, 1990, and Raghavachari, 1988.

2.1.1.2 Cost orientated objectives

Line length: Kim et al., 1996, study the problem of minimizing the overall length of the production line. The production line in study contains hybrid stations, being a mixture of open and closed stations.

Line length:

$$\sum_{i=1}^{M} L_i$$

Setup cost: The occurrence of setup cost may lead to the objective of minimizing the total setup cost to keep the production costs small. Setup cost SC_{efi} may occur when at station i a job $j+1$ of model type f follows job j of model type e.

Setup cost:

$$\sum_{i=1}^{M} SC_{efi}$$

Constant part usage (CPU): A well known objective for Just-In-Time productions is the constant part usage. The main idea behind this objective is to simplify the material handling and as a result to reduce the need for extra manpower which would be used to meet the peaks in part usage.

2.1.1.3 Combined objectives

In the literature, the use of individual objective functions, as mentioned above, as well as combinations can be found. As an example for combined objectives in sequencing problems Allahverdi, 2003, uses the bicriteria of makespan and mean flowtime, Ishibuchi et al., 2003, uses the bicreteria of makespan and the maximum tardiness, whereas Bard et al., 1992, uses makespan and line length as bicriteria for their algorithm.

2.1.2 Definition of the "basic" flowshop

The sequencing problem in the basic mixed model flowshop consists in a set of N jobs $(J_1, J_2, ..., J_J, ..., J_N)$ which have to be sequenced on M stations $(I_1, I_2, ..., I_i, ..., I_M)$, arranged in series. Each job has M sets of operations and requires its first set of operations on station 1, its second on station 2, and so on. The **set of sequences** Π $(\Pi_1, \Pi_2, ..., \Pi_i, ...\Pi_M)$ describes the order in which the jobs are processed on the M stations; in a **permutation flowshop** the sequence Π of jobs in station i is the same for all stations, i.e., $\Pi_1 = \Pi_2 = ... = \Pi_i = ... = \Pi_M$.

Furthermore, the **processing time** P_{ij} of job j on station i is known and constant. The time job j enters the system is called **start time** s_j, similarly, the time job j exits the system is called **completion time** c_j. If **setup time** is concerned, an additional time ST_{efi} may occur, necessary to change the setup of station i, in order to be able to process job $j+1$, which is of model f, after job j, which is of model e. The **setup cost** SC_{efi} is defined respectively. An extensive survey on setup considerations is presented by Allahverdi et al., 1999. Due to additional complexity most algorithms do not consider setup time nor setup cost, assuming that setup time and setup cost is sequence independent and can simply be added to the processing time/cost. Another simplification is used by algorithms for batch processing which pool jobs of the same model and process them in lots.

2.1.3 Diversity of flowshops

Next to the basic mixed model flowshop production line a variety of the same exist, resulting from the manifold problems which can be found in real life production systems and their specific products. In chemical processing, for example, it is common practice that once a job is started it can not be interrupted and therefore leads to a no-wait flowshop.

There exist various classifications and surveys that classify the multitude of flowshop problems. As mentioned before, Pinedo, 1995, classifies sequencing and scheduling problems using the triplet $\alpha|\beta|\gamma$. The classification of Vieira et al., 2003, is focused on rescheduling problems, the classification of Herrmann et al., 1993, is dedicated to static scheduling problems, and the work of Lageweg et al., 1978, is reduced to permutation flowshop problems. Plans, 1999 and Plans and Corominas, 2000, utilize five groups to classify assembly line problems, Niu, 2005, includes special issues like the *Management of the Assembly line* in her classification with four groups, and Becker and Scholl, 2006, survey problems and methods, focused mainly on the generalized assembly line balancing problem. The most common variations are as follows:

Non-permutation flowshop (FS): One of the pioneers who mention the flowshop problem is Johnson, 1954. In the non-permutation flowshop (basic flowshop) M stations are arranged in series, according to the technological sequence of the operations. A set of N jobs has to be processed on these stations. Each of the N jobs has the same ordering of stations for its processing sequence. Each job can be processed on one, and only one station at a time and each job is processed only once on each station. Furthermore each station can process only one job at a time. Jobs may bypass other jobs between two stations. The problem consists in finding a job sequence for each station. The case with $M = 1$ is called the **single station** case.

Buffers, installed between the stations are generally used for queuing reason in order to decouple the individual stations from each other and prevent blocking of a station; depending on the type of buffer, they offer the possibility of changing the sequence downstream of the buffer.

Permutation flowshop: Here the solutions are restricted to job sequences $\Pi_1, ..., \Pi_M$ with $\Pi_1 = \Pi_2 = ... = \Pi_M$, that is, the sequence on the first station is maintained for all stations in the flowshop. A set of permutation sequences is denoted dominant if no better sequence can be found than the best permutation sequence. This for example occurs in the no-wait flowshop.

Zero-buffer and no-wait flowshop: These two variations of the basic flowshop do not allow the jobs to form queues between the stations. The first case is with buffers of zero capacity. In this case a job j finishing at station i cannot advance to station $i + 1$ if there is still a job being processed. Station blocking of station i is the result. The second case, described by Aldowaisan and Allahverdi, 1998, is more restrictive. Once a job begins its processing on station 1, that job must continue without delay to be processed on each of the M stations. Here only sequences are feasible which do not result in blocking of any station.

No-idle flowshop: This constraint implies that each station, once started with processing, has to process all operations assigned to it without interruption. As mentioned by Cepek et al., 2002, a real life situation can be found, for example, if machines represent expensive pieces of equipment which have to be rented only for the duration between the start of its first operation and the completion of its last operation.

Flexible/hybrid/compound flowshop (FFS): Another variation of flowshops mentioned in the literature is the one in which parallel stations exist, see e.g. Li, 1997, Gendreau et al., 2001, Azizoglu et al., 2001, Riane et al., 1998, or Sun et al., 2003. This type of flowshop is named, by the majority of authors, flexible, hybrid or compound flowshop. The parallel station reduces cycle times needed for an operation at a station. Since in the mixed-model case the processing time of a station is dependent on the model, it gives the possibility of one job overtaking its predecessor. For this type of flowshop basically two setups exist: identical parallel stations and non-identical parallel stations.

2.1.4 Characteristics for solving flowshop problems

The flowshop is a widely studied problem. Within these studies not only the basic arrangement is taken into account but furthermore simplified setups, the most common being the permutation flowshop or the reduction to single station. Also variations were studied, amongst these the hybrid or flexible flowshop, providing parallel stations. In this section some properties of flowshops, derived from certain setups, are presented.

The basic flowshop permits the sequence of the jobs to change after each station which then leads to a total number of sequences being $(N!)^M$. For the simplified case of the permutation flowshop, in which the jobs have a unique sequence for all stations, the number of sequences is reduced to $N!$.

In the single station flowshop only permutation sequences are possible and the same makespan is achieved for all sequences. As shown by Baker, 1974, the mean flow time is minimized by the shortest-processing-time rule (SPT) and the weighted flow time is minimized by shortest-weighted-processing-time rule (SWPT). Furthermore, in the flowshop a common practice is to first determine the bottleneck station and consider it as being the only station.

The single station flowshop considering sequence dependent setup time corresponds to what is usually called the travelling salesperson problem (TSP). Allahverdi et al., 1999, explains that each city correspond to a job and the distance between cities corresponds to the time required to change from one job to another.

Johnson, 1954, considers a two and three stations flowshop problem with makespan objective ($F_2||c_{max}$ and $F_3||c_{max}$). His studies lead to the conclusion that in the two and three stations case the optimum permutation sequence is dominant. Also an exact solution for the two stations case is presented, namely SPT(1)-LPT(2). For the three stations case this method only finds the optimal solution if station two is dominant. That is, all processing times on the second station are larger then the largest on the two other stations. Gupta and Reddi, 1978, propose improved dominance conditions that are not limited to the dominance of the second station. Moreover, various heuristics exist that take advantage of the Johnson's rule in order to solve larger problems, see for example the CDS heuristic from Campbell et al., 1970, or Riane et al., 1998. Péridy et al., 1999, points out that the dominance of the permutation sequence also holds if on the first station precedence relations of jobs exist.

Baker, 1974, explains certain properties of the basic flowshop. With respect to any regular objective function, it is sufficient to consider only schedules in which the same job sequence occurs on the first two stations. Furthermore, with respect to the makespan objective, it is sufficient to consider only schedules in which the same job sequence occurs on stations $(M-1)$ and M, so that $(N!)^{M-2}$ schedules constitute a dominant set for $M > 2$. Thus, for a three-station flowshop with makespan minimization, the optimal sequence is a permutation sequence. For the case $M > 3$, Potts et al., 1991, determined the ratio, for $M = 2 \cdot N$, between the best permutation sequence and the optimum sequence being greater than $1/2 \cdot \lceil \sqrt{M} + 1/2 \rceil$. For a four station flowshop the difference is already 50%.

Pinedo, 1995, discusses and proofs that inverting the station sequence and the job sequence $j_1, ...j_N$ of a permutation flowshop with unlimited intermediate storage to $j_N, ...j_1$ results in the same makespan. He presents a mixed integer program (MIP) for the basic flowshop of M stations with makespan objective and permutation sequences, known as $F_M|prmu|c_{max}$, and proves that the basic flowshop with makespan objective and more than two stations is strongly NP-hard, see also Garey et al., 1976. Additionally, two special cases are discussed: station-dominance with increasing processing times for successive stations, and the proportional-flowshop where all processing times on the individual stations are the same for all jobs.

Buffers were initially introduced in permutation flowshops to decouple stations in order to prevent station blocking and starvation, see Pinedo, 1995. Blocking appears in the case in which the downstream station experiences problems, and therefore may not be able to process the assigned job. On the other hand, locating a buffer right after the station being down due to problems prevents its downstream station from starvation. The reasons for a station to be down are various, for example preventive maintenance, unavailable subassembled parts or repair of machine, etc. Moreover, bottleneck stations are generally provided with a buffer to avoid serious problems that may result in a line stoppage. The decoupling buffer is operated in first-in-first-out (FIFO) strategy. The flowshop with limited intermediate storage buffers is the more complex case in which a job may not be discarded from a station and therefore results in blocking which is the station is kept idle. In his studies of permutation sequences Pinedo, 1995, discusses a flowshop with limited buffers by reducing the storage buffers to zero. Reason for this is that a buffer can be presented by a station with zero processing time.

A directed graph is used for the calculation of the makespan of a permutation sequence. In the case of infinite buffers the nodes represent the processing time, see for example Ríos-Mercado and Bard, 1999, they also include setup time being the arcs between the nodes. When buffers are finite or not present, Pinedo, 1995, uses a different directed graph, the processing time here is presented by the arcs between the nodes. In both cases the makespan is calculated by the maximum weighted path. Lomnicki, 1965, presents a branch and bound solution for the three station case. Even though permutation sequences are dominant here, the presented graph-theoretical interpretation of Roy, 1962, allows the calculation of the makespan of a sequence which is not a permutation sequence.

Most solution techniques for sequencing in flowshop focuses on permutation sequences. Exact approaches for makespan minimization are presented by Ignall and Schrage, 1965, Lomnicki, 1965, Szwarc, 1971, Lageweg et al., 1978, Potts, 1980, Companys, 1992, Carlier and Rebai, 1996.

In a recent review of Framinan et al., 2002, heuristic methods for sequencing problems with focus on makespan objective are presented. Framinan and Leisten, 2003, furthermore present a comparison of heuristics for flowtime minimization in permutation flowshops. The most promising heuristics for permutation flowshops seem to be the Nawaz-Enscore-Ham (NEH) heuristic by Nawaz et al., 1983. Allahverdi, 2003, studies makespan and mean flowtime as multicriteria objective, using a linear combination. He proposes three new heuristics, AAH1, AAH2 and AAH3 and compares them with a series of existing heuristics, for both, the 2- and the M-station case.

In what follows is focused on two special types of flowshops, considering setup cost/time and afterwards non-permutation flowshops that permit to change the sequence between stations in order to allow further optimization.

2.1.5 Setup cost/time in flowshops

The mixed model production line produces a variety of products in the same line. These products may differ only in some optional components that are applied or not applied at a station. In this case the processing time at a station differs from one product to the next but does not require a special setup. In the case in which the operator needs to change the setup of the station in order to process the next job, the change of the setup may result in an additional production cost or an additional time, necessary to realize the change in setup. Pinedo, 1995, presents a proof of the NP-hardness of the single station case with setup consideration.

2.1.5.1 Appearance of setup

Burns and Daganzo, 1987, discuss a flowshop with setup costs and distinguish between three different types of setup cost/time:

- **Wasted material** resulting in an additional cost due to, for example, discard of the paint in the paint-shop of an automobile production. This setup cost has only impact on the objective function that minimizes the production costs.
- **Station downtime** and **labor** required to change the setup. This occurs for example when the mounting or a tool needs to be changed. In this case the schedule of the jobs is influenced directly, this means a sequence without setup time is not possible because some job change requires additional time for preparation.
- **Product quality implications** which affect the performance of the station. For example the paint quality may temporarily decline when a change of color occurs.

Apart from the differentiation of the setup cost/time, a distinction is done in respect to the stations. Bolat, 1994, classifies the stations into two types: Some stations are assigned to do exactly the same operation on every job, and some allow operations that are not performed on every job or performed on every job but with some options. In a mixed model production line usually both types of stations are found.

2.1.5.2 Sequencing problems considering setup

A partly efficient but widespread technique to avoid setup is to form batches, groups of jobs belonging to the same model. In this way the number of instances in which setup occurs is the number of batches that are to be processed. This simple method permits neither an advanced optimization of the sequence, nor additional constraints, such as an option can only be applied on every second job. Nevertheless until now it is a widely applied method in the industry.

Another approach considers that setup time exists, but the average of the setup time is added to the processing time. The adapted processing times are then used to solve the sequencing problem; however, the determined sequence needs to be revised for feasibility. The method is sensitive to inhomogeneous distribution of the setup time.

In order to further improve the solution, the setup cost/time is considered completely separated. Allahverdi et al., 1999, highlight that when setup cost is directly proportional to setup time, a sequence that is optimal with respect to setup cost is also optimal with respect to setup time. Their survey on setup considerations furthermore considers sequencing problems in flowshops regarding the following characteristics:

- **Batch setup**: Here jobs are grouped into batches and a major setup is incurred when switching between jobs belonging to different batches, whereas a minor setup is incurred for switching between jobs within the batches.
- **Sequence dependent setup**: Including setup cost/time that depends on the succeeding job gives the possibility to further improve the sequence. In the symmetric case the appearance of setup cost/time is the same for a change from model e to f and for a change from model f to e.
- **Setup time separable from processing time**: The case in which setup time is separable from processing time leads to the possibility of further reducing the total processing time. This results from the fact that once on a station i a job j is finished, the setup can already be changed way before job $j + 1$ arrives.

Apart from the single use of these characteristics various authors exist that incorporate other line considerations. Burns and Daganzo, 1987, discuss the paced flowshop line with setup costs. In addition, constraints are used to determine the minimum distance between two jobs of the same model, if not fulfilled, the station capacity is exceeded. The proposed method uses grouping and spacing to sequence the jobs. The work does not tempt to achieve the optimum solution but establishes analytical principles that aid in developing effective production line job sequencing methods.

The work of Kim and Kim, 2000, proposes two methods for solving the sequencing problem including setup times necessary for a model-change and uses closed stations. The first method proposed is a branch and bound using a lower bound that calculates the lowest possible unfinished work of the remaining sequence. The authors remark that the method finds the optimum solution for mid-scale problems. The second method is a heuristic method called MST (Minimum-Setup-Time) and favors jobs that prevent idle and unfinished work.

With a genetic algorithm Kim et al., 1996, find near optimal solutions solving the problem of minimizing the overall length of the production line. The production line in study contains hybrid stations which is a mixture of open and closed stations. They highlight the importance to achieve a proper balance between exploitation and exploration of the genetic algorithm. These characteristics describe the good choice of the size of populations and the technique of forming new populations. The results are then compared with a conventional, slowly proceeding branch and bound algorithm provided by standard software packages.

Bolat, 1994, presents next to a branch and bound algorithm a heuristic technique, applied to an automobile production line which takes into account setup costs, caused by discarded paint and solvent when a change in color occurs. Bolat highlight the importance of considering both setup and utility costs, mainly to evaluate the economic benefit of investments in setup cost reduction. The used heuristic, TRIM, basically selects jobs that result in less utility-work without considering setup costs.

2.2 Resequencing in flowshops

As mentioned in section 2.1, in the basic flowshop with three stations and makespan objective, the permutation sequence is dominant. Pinedo, 1995, demonstrates that the problem of three or more stations with makespan objective is strongly NP-hard and that for a production line consisting of more than three stations furthermore a permutation sequence is not dominant and clearly leads to additional complexity.

2.2.1 Objectives of resequencing

In order to permit a job to overtake another job within a production line, arranged as a flowshop, normally the line has to undergo essential changes, many times combined with investment. These changes may result in hardware to be installed, like buffers, but also in additional efforts in terms of logistics implementations. Clearly, these additional efforts are only reasonable if the resequencing pays off the necessary investment. This cost calculation obviously depends on the particular production line and therefore has to be taken into account for the individual cases.

The major objective of resequencing a pre-arranged set of jobs in flowshops is further minimization of production costs, for example resulting in a higher utilization of the production resources. This is desirable even more when setup cost/time is involved or the processing times of the individual jobs diverge among one another.

Apart from the deliberate resequencing of jobs in order to improve the productivity, an undesired resequencing, of a permutation sequence, may occur. Here the objective is to regenerate the original sequence.

2.2.2 Methods for resequencing

Various possibilities of resequencing jobs in a flowshop exist, some of which may already be included in the line's setup. Others may require additional installations:

- Offline or online **buffers**; the latter not operated in FIFO mode,
- Flexible/hybrid/compound flowshops containing **parallel stations**,
- **Splitting** and **merging** of parallel lines,
- **Re-entrant** of jobs in the production line and
- Change of **job attributes** (no physical change).

References on methods for resequencing are summarized in table 2.1 to table 2.4 on page 34 to 37.

2.2.2.1 Buffers

The buffer types, used to temporarily store jobs within a flowshop, basically have three objectives: decoupling, resequencing and end-product storage. Here only buffers are concerned that permit resequencing which results in the use of non-permutation sequences. The usefulness of this type of buffers is obvious already in the simple case, in which a line is divided into two parts; a permutation sequence that is optimum for the first part usually is not optimum for the second part.

2.2.2.1.1 Infinite buffers The case of infinite buffers is basically a theoretical case in which no limitation exists with respect to the number of jobs that may be buffered between two stations. Roy, 1962, presents a graph-theoretical interpretation which allows the calculation of the makespan of a sequence in the flowshop that is not a permutation sequence. Surveys on heuristics treating the case of infinite buffers are presented by Liesegang and Schirmer, 1975, and Park et al., 1984.

2.2.2.1.2 Large ASRS buffers Large buffers are used, for example, in the automobile industry. Production lines in the automobile industry are divided into three parts: body-, paint- and assembly-shop. Two buffers are located between each part of the production line. The reason for using large buffers in this case is to resequence the jobs in a large scale. As a result, batches are formed and each shop is optimized separately. Only in the case in which the optimal sequence for one shop is the same as for the following, no resequencing is performed.

Solution techniques using large buffers, called **Automatic Storage and Retrieval System** (ASRS), due to Lee and Schaefer, 1997, were introduced in the automobile industry in the 1950s. Choi and Shin, 1997, call their ASRS-buffer a painted-body-storage (PBS) which uses a dynamic sequencing method for the resequencing. Their buffer has various rows and the jobs arriving at the buffer then get loaded to the row that already has jobs loaded being similar.

Lee and Schaefer, 1997, present an approach to statically and dynamically load an ASRS buffer. Two modes are used: in the single cycle mode a job is stored or retrieved within one cycle and in the double cycle mode a single job is stored and another retrieved in the same cycle. The objective is to minimize the total travel time required by processing the entire storage and retrieval.

Inman, 2003, studies how to undo undesired scramble of the original sequence for an automotive production. The scrambling is caused by parallel inspection stations with random inspection times and repair loops. The ASRS size here depends on the most negative sequence displacement, the goal is to obtain a 90% service level.

Ding and Sun, 2004, utilize a buffer, placed between two successive parts of the automobile production, to resequence a fixed number of vehicles. The algorithm determines a loading sequence for h incoming vehicles to a buffer with h storing places and then releases them downstream. The resequencing can also be used to overcome unintentional sequence alternation caused by rework or equipment breakdowns. An exact and a heuristic solution are presented. The method can only be used with a large buffer because the number of vehicles to be resequenced at once is depending on the buffer size. Furthermore, the method overlooks the fact that it might be beneficial to incorporate the possibility of determining an improved sequence already at the beginning of the line that would result in a better overall solution.

2.2.2.1.3 Small buffers The buffer, initially used for decoupling, basically forms a serial chain of the incoming jobs. Two operation modes exist: The **free-access operation** permits to remove any stored job from the buffer, in contrast to the **FIFO operation** where only jobs can be removed from the buffer in first-in-first-out sequence. The buffer location can be **online**, located between two stations, or **offline**, removing the job from the line. An online buffer of size larger than one can be used for resequencing, if not operated in FIFO mode. The **offline** buffer may be operated in either mode; jobs remaining in the line can bypass the buffered jobs and resequencing takes effect. As compared to the large ASRS buffers, the use of essentially smaller sized buffers conduces to a completely different implementation. It has to be clarified how removing or adding a job affects the line.

As explained by Pinedo, 1995, mathematically a buffer can be realized as a station with zero processing time. Approaches which consider a limited number of buffer places are studied by Dutta and Cunningham, 1975, Reddi, 1976, Papadimitriou and Kanellakis, 1980, and Leisten, 1990.

Wilhelm and Shin, 1985 study a Flexible Flowshop (FFS) and take into account the use of a common buffer. The performance measures were makespan, system utilization, utilization of individual machines, flow-time, maximum spaces required in the common storage, and maximum number of vehicles required.

Koshla, 1995, studies a two station flexible flowshop. The first station consists of various parallel machines which have access to a common buffer, limited by the number of buffer places.

Holthaus and Rajendran, 2002, study the performance of dispatching rules which originate from job shop production lines and apply them to a dynamic flowshop with a limited number of intermediate buffer places. They conclude that the performance of the used dispatching rules are sensitive to the presence/absence of missing operations of jobs, which appear in *Flowline-based Manufacturing system* (FBMS).

Lahmar et al., 2003, and Lahmar and Benjaafar, 2003, study the problem of resequencing a pre-arranged set of jobs with the objective of minimizing changeover costs. The example under study is the paint shop of an automobile production and a setup cost appears every time a color change occurs and paint is flushed. For example, the cost for a metallic paint is higher than for an ordinary paint. The study considers pull-off tables, originally designed for rework, which allows to pull-off one car at a time. A color assignment matrix is used to define the possible colors with which a car may be painted. The problem with 1 buffer is solved optimally and with L buffers the problem is decomposed to L problems with 1 buffer, not guaranteeing optimality. The conclusions show that the biggest amount of cost saving is achieved by only implementing few pull-off tables in the line.

Brucker et al., 2003, present a Tabu search for a flowshop with a buffer positioned between all consecutive stations. The number of buffer places is the same for all resequencing buffers and the variable parameter is the number of buffer places which is 0, 1, 2, or infinite.

Witt and Voss, 2005, use intermediate buffers which permit the use of non-permutation sequences. Additionally, it is considered that different jobs have different physical sizes and occupy only a portion of the intermediate resequencing buffer which is used for decoupling and resequencing.

Jain and Grossmann, 2000 present a MILP model for a flexible flowshop with continuous processing. The Continuous processing is referred to the fact that a job (here liquids) is not completed at a station before it leaves the station and starts to occupy an intermediate buffer place while the job is still processed on the previous station. Furthermore, the job can already be processed on the subsequent station. The intermediate buffer places are restricted by their physical size and it is noted that jobs that are too large for a buffer place can be split and stored in two buffer places. A flexible flowshop with two stations is considered, each having various parallel stations. Next to the exact model, a heuristic approach is introduced.

Ha et al., 2000 present a MILP model to minimize the makespan for a multi product batch processing which uses intermediate storage tanks. The products can furthermore be restricted in terms of assignation to certain tanks. Results are presented which discuss a semi dynamic demand, being the comparison of the optimal permutation sequence with the case in which intermediate constrained buffers are used (including product restrictions) using the same incoming sequence. Furthermore, there exists handling time and setup time.

The introduction of buffers to a production line also has negative effects. Amongst these are the increase of work-in-process, the additional cost for installation and maintenance and the increase of necessary area. All of these factors are cost or time relevant issues and should not be overlooked in the design of the production line.

2.2.3 Flexible, hybrid or compound flowshop (FFS)

Flexible flowshop problems overcome one of the limitations of the basic model of flowshops by allowing parallel stations. As described earlier, the use of parallel stations in a mixed-model production line gives the possibility of one job overtaking its predecessor. The parallel station may also serve as a buffer with zero processing time, and let various jobs pass by the buffered job. This however is not a desirable case because the actual parallel station may block a job from being processed and therefore provides only limited resequencing possibilities in terms of cost savings.

Sawik, 2000, presents a mixed integer programming formulation for a flexible flowshop with one or more identical parallel stations. Also blocking due to infinite online buffers, alternative routing and reentrant of products is included. Additional constraints are introduced by Riane et al., 1998, which include jobs that are assigned to a certain parallel station and therefore the flow can not freely be determined. Next to the mixed integer programming formulation two heuristic procedures are proposed, based on dynamic programming and on branch and bound.

Pinedo, 1995, explains the heuristic method used at IBM for a flexible flowshop with limited buffers and bypass, called flexible-flow-line-loading (FFLL). The loading of job j is determined by minimizing the overload that is cumulated until job j. Further heuristics are described by e.g. the forward-and-backward heuristic of Li, 1997, and the divide-and-merge heuristic of Gendreau et al., 2001, and Sun et al., 2003.

2.2.4 Merging and splitting (FS-PL)

Engstrőm et al., 1996, report the introduction of parallel segments of stations to the automobile manufacturer Volvo that permit to resequence jobs where the line split and merge. The splitting of a production line is somewhat more challenging due to the fact that two parallel lines may not perform the same options and constraints exist that may additionally influence the sorting.

2.2.5 Re-entrant flowshops

Within the studies of Sawik, 2000, also the re-entrant flowshop is considered that manufacture double sided printed-circuit-boards (PCBs). The PCBs run twice through the same line, first to assemble the bottom side and then to assemble the top side. Instead of assembling all PCBs on one side first, the optimum sequence may consist in interleaving the first PCBs already having finished the first run on the bottom side.

2.2.6 Change of job attributes (no physical change)

Instead of physically changing the job order Rachakonda and Nagane, 2000, mention the solution implemented in the automobile production of the Ford assembly plant in Wixcon, USA. Here a dynamic resequencing system is used, involving the swapping of cars by changing their attributes. Consequently it is not necessary to physically change the job position within the sequence.

2.2.7 Undesired resequencing

Undesired resequencing occurs in many real life arrangements in which jobs require repair, exist parallel inspection stations with unequal processing time or a problem occurs in the part delivery, necessary to process the sequence in the correct order.

Inman, 2003, and Korkmaz et al., , study how to undo undesired scramble of the original sequence for an automotive production. The scrambling is caused by parallel inspection stations with random inspection times and repair loops. Inman uses a large buffer, ASRS-automated storage and retrieval system. Choi and Shin, 1997, call their ASRS-buffer a painted-body-storage (PBS) which uses a dynamic sequencing method for the resequencing. The considered buffer has various rows and arriving jobs then get loaded to the row that already has jobs loaded being similar.

2.2.8 Related work on resequencing

Péridy et al., 1999, uses a selection of elimination rules to reduce the search tree of the branch and bound algorithm proposed by Potts, 1980, for the permutation flowshop and by Carlier and Rebai, 1996, for the basic flowshop. The idea is to calculate improved bounds by not including certain sequences that result infeasible due to the elimination rules. The simplest elimination rule is for example the rule by Johnson. Pugazhendhi et al., 2002, and Pugazhendhi and Thiagarajan, 2004, study non-permutation sequences in flowline-based manufacturing systems (FBMS). The line is similar to the basic flowshop with some or all jobs having missing operations in some stations, leading to a property which is not proper of the flowshop, i.e. the station precedence is not the same for all job. Hence, an optimum sequence can be obtained that is not feasible if the processing time would be infinitesimal small instead of zero.

2.2.9 Summary

The summary of existing solution methods for flowshop production lines which consider resequencing of jobs within the production line are presented in table 2.1 to 2.4. The summary considers exact approaches as well as heuristic approaches; their objectives, as well as the realization of the distinct resequencing possibilities, are manifold. The objectives range from minimizing the makespan or the accumulated setup cost to the objective of sequence restauration. The distinct realizations on the other hand contemplate implementations ranging from the installation of buffers off the production line, parallel stations, merging and splitting of parallel line parts and production lines where products re-enter the line, to the case in which job attributes are swapped without changing the physical position of the job in the sequence.

The classifications and classifying surveys by Pinedo, 1995, Vieira et al., 2003, Herrmann et al., 1993, and Lageweg et al., 1978, Plans and Corominas, 2000, Niu, 2005, Becker and Scholl, 2006, mentioned in section 2.1.3, do not consider exhaustively the possibilities of resequencing jobs within the production line.

Table 2.1: Resequencing methods I

Type	Reference	Shop	Static Dynamic	Method	Objective	Sequence-Type	Observation
Infinite buffers	Roy, 1962	FS	S	-	-	Non-perm	Graph-theoretical interpretation for makespan calculation in a FS.
	Péridy et al., 1999	FS	S	E	Makespan	Perm/ Non-perm	Elimination rules for lower bounds.
	Pugazhendhi et al., 2002	FBMS	S	H	Makespan	Non-perm	Station precedence not same as for flowshop.
	Pugazhendhi and Thiagarajan, 2004	FBMS	S	H	Makespan Flowtime	Non-perm	Station precedence not same as for flowshop. Also considering sequence-dependent setup times.
AS/RS buffer	Lee and Schaefer, 1997	FS	S/D	E/H	Traveltime of jobs in buffer	Non-perm	AS/RS buffer operates in single- or double-cycle mode for load and unload.
	Choi and Shin, 1997	FS	D	H	Deviation from ideal sequence	Non-perm	Measures deviation of leaving jobs from the desired and sequences due to spacing constraints between jobs.
	Ding and Sun, 2004	FS	S	E/H	Sequence restauration	Non-perm	Buffer with k storing places to resequence k jobs.
	Inman, 2003	FFS	S/D	E/H	Sequence restauration	Non-perm	AS/RS buffer size depends on most negative sequence dspacement. Unwanted sequence change occurs due to repair loops and parallel stations
	Korkmaz et al.,	FS	D	H	Sequence restauration	Non-perm	Unwanted sequence change occurs due to part shortage or inspection stations.

Shop:

FS	Flowshop
FBMS	Flowline-based Manufacturing system (not all jobs have to pass through all stations)
FFS	Flexible Flowshop with parallel stations
FS-PL	Flowshop with parallel lines

Method:

E	Exact method
H	Heuristic and metaheuristic method

Table 2.2: Resequencing methods II

Type	Reference	Shop	Static Dynamic	Method	Objective	Sequence-Type	Observation
Small buffers	Dutta and Cunningham, 1975	FS	S	E/H	Makespan	Non-perm	Limited number of intermediate buffer places.
	Reddi, 1976	FS	S	E	Makespan	Non-perm	Limited number of intermediate buffer places.
	Papadimitriou and Kanellakis, 1980	FS	S	H	Makespan	Non-perm	Limited number of intermediate buffer places.
	Leisten, 1990	FS	S	H	Makespan	Non-perm	Limited number of intermediate buffer places.
	Wilhelm and Shin, 1985	FFS	S	H	Makespan, etc.	Non-Perm	Flexible Flowshop, considers a common buffer.
	Holthaus and Rajendran, 2002	FS/FBMS	D	H	Flowtime, etc.	Non-Perm	Comparison of dispatching rules for a flowshop with limited number of intermediate buffer places.
	Lahmar et al., 2003	FS	S	E/H	Setup cost	Non-perm	Resequencing with feature assignment. Decomposition of problem with N buffers to N problems with one buffer.
	Lahmar and Benjaafar, 2003	FS	S	E/H	Setup cost	Non-perm	Study of resequencing limited by the number of jobs a certain job can move forward or backward.

Shop:

FS	Flowshop
FBMS	Flowline-based Manufacturing system (not all jobs have to pass through all stations)
FFS	Flexible Flowshop with parallel stations
FS-PL	Flowshop with parallel lines

Method:

E	Exact method
H	Heuristic and metaheuristic method

Table 2.3: Resequencing methods III

Type	Reference	Shop	Static Dynamic	Method	Objective	Sequence-Type	Observation
Small buffers	Brucker et al., 2003	FS	S	H	Makespan	Non-perm	Intermediate buffers are positioned between all consecutive stations. The number of buffer places is the same for all resequencing buffers.
	Witt and Voss, 2005	FS	S	H	Makespan	Non-perm	Jobs have different physical sizes and occupy only a portion of the intermediate buffer, used for decoupling and resequencing.
	Jain and Grossmann, 2000	FFS	S	E/H	Makespan	Non-perm	Jobs have different physical sizes as well as the intermediate storage tanks for continuous processing.
	Ha et al., 2000	FS	S	E	Makespan	Semi Dynamic	Comparison of optimal permutation sequence with the case in which intermediate constrained buffers are used (including product restrictions) using the same incoming sequence. Handling time and setup time exist.

Shop:

FS	Flowshop
FBMS	Flowline-based Manufacturing system (not all jobs have to pass through all stations)
FFS	Flexible Flowshop with parallel stations
FS-PL	Flowshop with parallel lines

Method:

E	Exact method
H	Heuristic and metaheuristic method

Table 2.4: Resequencing methods IV

Type	Reference	Shop	Static Dynamic	Method	Objective	Sequence-Type	Observation
Hybrid or flexible flowshop	Sawik, 2000	FFS	S	E	Makespan	Non-perm	Limited intermediate buffers. Station blocking and alternative processing routes are possible.
	Riane et al., 1998	FFS	S	E/H	Makespan	Perm/Non-perm	Exist jobs assigned to a certain parallel station.
	Pinedo, 1995	FFS	S	H	Makespan	Non-perm	Exist unlimited intermediate buffers.
	Li, 1997	FFS	S	H	Makespan	Non-perm	Consider major and minor setups, part families and batch splitting. Only single station case.
	Gendreau et al., 2001	FFS	S	H	Makespan	Non-perm	Divide-and-merge heuristic. Considers setup time. Only single station case.
	Sun et al., 2003	FFS	S	H	Makespan	Non-perm	Exist jobs assigned to a certain parallel station.
Mergin and splitting	Engstrőm et al., 1996	FS-PL				Non-perm	Various parallel stations lead to the possibility of resequencing.
Re-entrant flowshop	Sawik, 2000	FFS	S	E	Makespan	Non-perm	Limited intermediate buffer. Also re-entrant of jobs is considered.
Change of job attributes	Rachakonda and Nagane, 2000	FS	S	H	Setup cost	Non-perm	Swapping of cars in the sequence, without changing their physical location, but changing their other attributes.

Shop:		**Method:**	
FS	Flowshop	E	Exact method
FBMS	Flowline-based Manufacturing system (not all jobs have to pass through all stations)	H	Heuristic and metaheuristic method
FFS	Flexible Flowshop with parallel stations		
FS-PL	Flowshop with parallel lines		

2.3 Static and dynamic demand

The production planning for mixed model flowshops can be various and usually depends on the planning horizon and in some cases on the possibility of decoupling the customer orders from the production planning. In the latter case the incoming customer orders may only give a guideline for what needs to be produced in the plant. For example, it might be advisable to produce a second part of a single-part order. This results from the fact that a negligible additional cost occurs, and in case of a quality problem with one part, a second one is available as reserve. On the other hand it might be advisable for a production not to start the production of this single-part order until a reasonable number of the same parts have accumulated.

Even though many sequencing procedures take into account a fixed, static demand, it would also be desirable to include urgent customer orders when the production is already up and running. This then leads to the need of dynamic planning.

Engel et al., 1997, distinguish between the two cases of build-to-plan and build-to-order production. The two terms describe the dynamics of the input sequence. In the first case the demand is well known in advance and only a few model types are produced repeatedly. In the latter case each product is determined by an individual selection of options corresponding to a customer order.

In what follows, more detailed explanations of the characteristics of static and dynamic production planning are given. These individual cases lead to intermediate cases that may be found in the transition from the static to the dynamic case.

2.3.1 Static demand

In a sequencing problem a demand is called static if the information required to determine a feasible but not necessarily optimal sequence is known before the first job is processed. In this case the sequencing and the execution of the sequence are considered consecutively. Díaz et al., 2003, mentions that this type of production planning is often referred to as offline sequencing (scheduling). The static production planning accumulates customer orders to lots; the lot then is sequenced and released to the production. Burns and Daganzo, 1987, highlight that an increased lotsize allows more efficient sequences to be identified, whereas longer lead times for customer orders are necessary.

Minimal-part-set (MPS): One of the most common representations is the use of the MPS which is the least common multiple of the individual models for the entire demand. Repetition of the MPS leads to the required demand. See for example Sarker and Pan, 1998, Bard et al., 1992, or Hyun et al., 1998.

The MPS is considered to be a good choice, however, Klundert and Grigoriev, 2001, show that many times it is not the best choice. They use a travellings salesperson problem (TSP) where the cities are to be visited various times in order to resolve the problem and conclude that better sequences are possible which obviously result in extended computational effort. Seasonal production is a classical example of a static production planning which refers to a production demand that is known for a predetermined period of time, i.e. the season.

2.3.2 Dynamic demand

A sequencing problem is called dynamic if the sequence is constructed while jobs are already entering the flowshop, the sequence is determined online. This is typically the case if the total demand or production relevant parameters are not known in advance, but become available during the execution. The sequencing is then only possible for jobs we have already knowledge of, i.e. it can only be done on the basis of a limited planning horizon; while new jobs arrive, the current sequence has to be updated appropriately. This uncertainty of timing in the production leads to the difficulty of controlling efficiently the material supply of the line and respond to the customer orders in a predictable manner. In order to apply an adequate measure of performance, the objective functions, presented in part 2.1.1, may be modified. Rather than the absolute value, the objective function should determine the mean value over the jobs.

Smed et al., 1999, mention various reasons that lead to dynamic demand planning: machine breakdown, component shortage and maintenance delay, urgent prototype series surpassing normal production, and the production plan itself which can be subject to sudden alterations during the production period.

Vieira et al., 2003, in their framework of resequencing (rescheduling), mainly for the dynamic case, furthermore present common performance measures. A separation is done into measures of schedule efficiency, schedule stability, and cost.

2.3.3 Transition from pure static to dynamic demand

The above mentioned separation into static and dynamic demand is not always realistic and leads to the use of models that share properties of both of the two individual cases. In fig.2.1 three different cases demonstrate the transition from the static to the dynamic demand. In the first case (1) the demand is known beforehand and permits an offline optimization, (2) is the intermediate case in which the demand is known with a limited time horizon. (3) then is the dynamic case in which the orders arrive without advise.

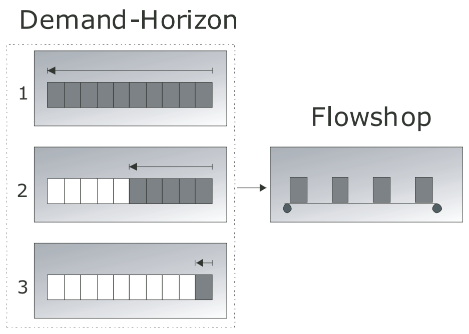


Figure 2.1: Illustration of the demand horizon. 1) entire demand is known beforehand and permits an offline optimization 2) demand is known with a limited horizon and 3) demand is not known. The last case is the most dynamic one and does not permit sequencing before the jobs enter the first station of the flowshop and therefore has to be calculated online.

These variations lead to different solution techniques that focus on arranging the job order before the jobs enter the line (sequencing) or involve the option of reordering the jobs in the line (resequencing). In what follows, the distinct configurations of possible demands are listed:

- **Static case (permutation sequence)**: Determination of the optimal permutation sequence.
- **Static case (non-permutation sequence)**: Determination of the overall optimal sequence, also including resequencing within the line.
- **Nearly-Dynamic case**: Determination of introduction of jobs to the plant with dynamically changing demand. A predetermined number of jobs ready to enter into the plant is given which can be ordered before entering into the line. In this case not only sequencing before the line (permutation) but also resequencing in the line (non-permutation) might be considered.

- **Semi-Dynamic case**: The focus here is the determination of a better sequence for a given incoming sequence by using e.g. buffers, except before the first station. In the case in which the demand is completely known, the resequencing is calculated offline.
- **Dynamic case**: Here the jobs enter the first station without the possibility of sequencing beforehand. This predetermined sequence then is resequenced within the line in order to improve the production.

Figure 2.2 summarizes these configurations by considering three different cases for sequencing of the jobs and two cases for the demand. *Sequencing* is referred to permutation flowshop in which the order of jobs on the first station is maintained for all stations in the flowshop. *Sequencing & Resequencing* is the case of non-permutation flowshop, the order of the jobs in the first station as well as for each station with access to a resequencing buffer can be different. *Resequencing* is the case in which the order of jobs can not be varied before entering the first station but may be altered at any other station with access to a resequencing buffer.

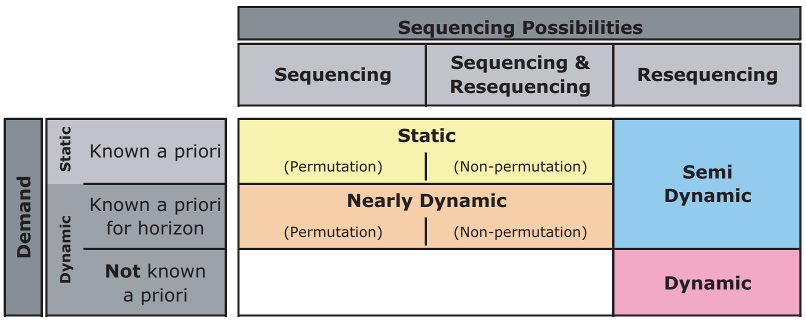


Figure 2.2: Sectioning of static and dynamic demand as a function of sequencing possibilities and the demand horizon.

2.3.4 Solution techniques

Solution techniques are various and basically are separated depending on if a static or dynamic case is studied and, furthermore, depend on the size of the problem. In the static case with moderate size, optimum seeking algorithms like branch and bound or dynamic programming are applied; see for example Dar-El and Cucuy, 1977, Sarker and Pan, 2001, or Stafford and Tseng, 2002. Larger sized static problems are generally solved with heuristic procedures, which, for the sake of computation do not ensure an optimal sequence, see for example Bard et al., 1992, some of which are derivations of exact solutions, see Ríos-Mercado and Bard, 1999. In the more complex case of dynamic process planning procedures are used, called dispatching rules, that give priorities in the selection of jobs in a queue formed in the buffers before the stations. These dispatching rules also find application in scheduling problems and in jobshop applications where the routing of the individual jobs is different for each job.

2.4 Optimization methods

Various attempts with distinct approaches exist to solve the problems occurring in flowshops and can be found in literature concerning flowshop problems as well as in literature pertaining to combinatorial optimization. The simplest attempt would be an enumeration of all possible solutions, obviously leading to a total of $(N!)^M$ solutions. This would result in over 24 billion possibilities for a problem as simple as 5 jobs and 5 stations. All attempts have in common to start with taking apart the entire problem in order to isolate the static part of the problem, namely the line itself. The optimization routine then applies the input data for the line, together with the data for demand, resulting in a measure of efficiency, obtained by some objective function.

The separation of the numerous optimization methods with all its modifications usually concerns two major divisions. Primarily, there exist exact, optimum seeking methods, which by nature are limited in problem size. On the other hand, the non-exact methods, namely heuristics and metaheuristics. Within the heuristic methods stand out detailed algorithms as well as heuristics that are based on rules on how to proceed with the search for good solutions. A variety of heuristic approaches, which are applied to resequencing of jobs in flowshop production lines are listed in tables 2.1 to 2.4. Obviously, gaining a reduction in computational time results in a possible reduction of quality. This reduction, however, can many times lead to near optimum solutions which for real life problems are indispensable.

2.4.1 Elements of optimization

The assembly line optimization consists of several parts which can firstly be treated separately in order to establish a consistent and structured model. Fig.2.3 shows the main parts: **demand**, **optimization variables**, **line parameters**, **line efficiency**, the **optimization procedure**, and finally, the **result**.

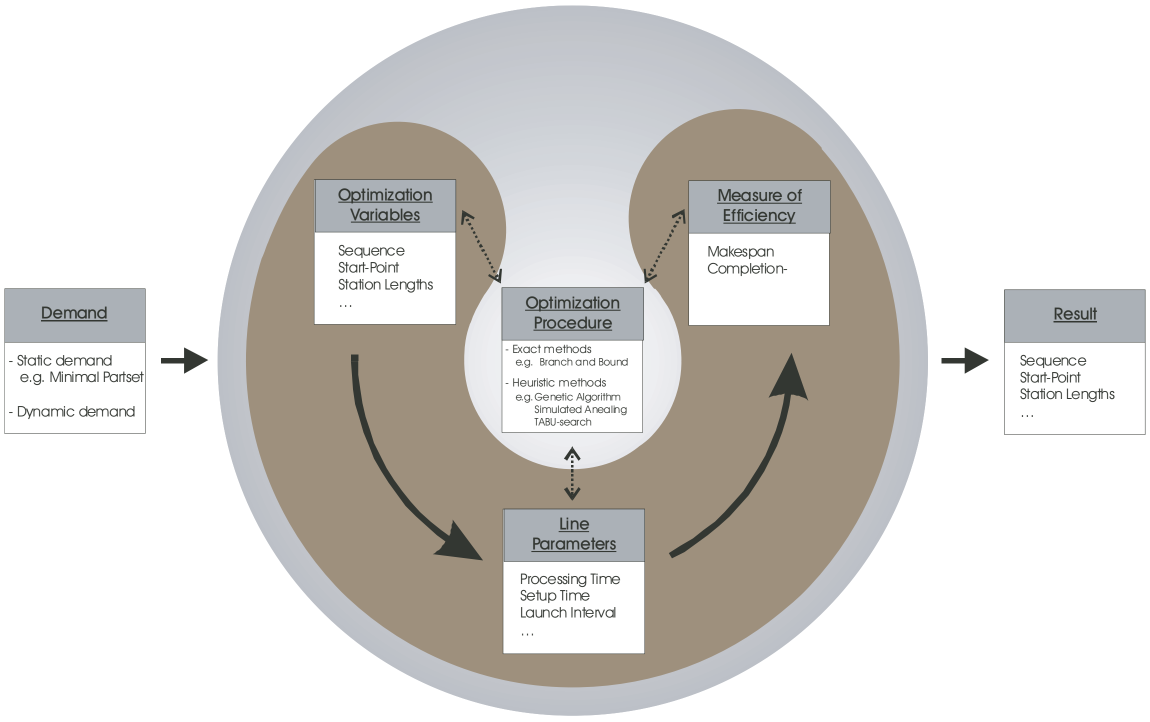


Figure 2.3: Structure of the sequencing problems in assembly lines.

As described in the previous parts of this document, two lines may have very distinct aspects and objectives to optimize. Nevertheless, before starting with the simulation and optimization, it is necessary to model the line and define all the surrounding parameters that concretize and limit the problem in question.

- **Demand**: The demand is synonym to customer order. No matter if a static or a dynamic case is studied, the demand describes the amount of products that need to be processed. The customer orders may be accumulated to lots and then be processed all at once, which is the most static case and many times the Minimal-Part-Set (MPS) is used to work only with the least-common-multiple of the different models to be produced. More dynamic cases imply that customer orders arrive while the production has already started.

- **Line parameters**: The line parameters describe the pool of static parameters that are defined beforehand and are fixed for the entire optimization procedure. For example, the sequencing problem in flowshop optimization in general uses a constant launch interval that is obtained by the design and balancing procedure, performed beforehand. Other line parameters may be processing time, setup cost/time, buffer location, precedence relations, etc.

- **Optimization variables**: The optimization variables defines the pool of variable parameters that are to be optimized by the optimization procedure. It is desirable to keep the number and ranges of optimization parameters relatively small in order to limit the considered problem to a tractable one. Many times an initial set of parameters that describe a feasible solution is found by a fast heuristic and serves as an initial bound. Depending on the objective of optimization and the possibilities that are provided by a line-arrangement, the optimization parameters can be various, such as the job sequence, the station length, etc.

- **Measure of efficiency**: The line efficiency is a measure of performance and is calculated with the objective function to be optimized. This measure of efficiency in the general case is based on a single objective, but also combined objectives are possible that, e.g., minimize makespan as well as the idle time of operators.

- **Optimization procedure**: Once the line is defined with all its input and output parameters, an optimization procedure is needed to optimize the line efficiency by, e.g., varying the line parameters, always taking into account that the demand has to be satisfied. There exist basically two groups of methods: the exact methods that obtain the optimal solution, if existent, and approximation methods which for computational merit not necessarily obtain the optimal solution.

- **Results**: The result of the optimization procedure usually contains the values for the optimization variables and the measure of performance, together with additional information as for example the job sequence.

2.4.2 Complexity of problems

For the classification of combinatorial problems, such as sequencing in flowshops, a concept is used that represents the computational effort related with the problem in question and is used by various authors, as e.g. Dudek et al., 1992, Brucker, 1998, Shakhlevich et al., 2000. A problem is assigned to the P-problem (polynomial time) class if an algorithm exists which solves the problem in polynomial time. Thereafter, a problem is assigned to the NP-problem (nondeterministic polynomial time) class if it is possible to solve it within polynomial time on a hypothetical non-deterministic Turing machine, allowing parallel executions. The class of P-problems is a subset of the class of NP-problems, but there also exist problems which are not NP, known to be NP-hard. Extended explanations on computational complexity can be found in Garey and Johnson, 1979.

In the classification of Pinedo, 1995, several sequencing problems in flowshop production lines are specified. Polynomial time solvable problems $F_2|block|c_{\max}$ and $F_2|nwt|c_{\max}$ are two station cases with blocking and no-wait respectively. Also a special case for M stations, $F_M|P_{ij} = P_j|\sum c_j$ with equal processing times for job j at all stations is part of these problems. Within the strongly NP-hard problems $F_2||\sum c_j$ is the two station case with total flowtime optimization and $F_3||c_{\max}$ and $F_3|nwt|c_{\max}$ are three station cases with and without no-wait sequences and the makespan objective.

2.4.3 Exact methods

Exact methods always lead to the optimal solution. Clearly, a limit on problem size for exact methods exist that is way inferior compared to heuristic methods. In order to reduce the complexity for the necessary computation it is advisable to isolate the problem as tight as possible, i.e. when dominant rules can be used such as permutation sequences are sufficient for the two station case, it is not necessary to consider all possible sequences at all stations.

In order to find the minimum or maximum of an objective function the most obvious approach would be of analytical nature, using its derivation (**Calculus** and **Lagrange multipliers**). It is important that the function and its first derivation is continuous. As highlighted by Nicholson, 1971, these methods can be used only on small problems containing few continuous variables with simple objective functions and no constraints, therefore clearly is not applicable to flowshop sequencing problems.

- **Complete enumeration**: The simplest way to solve the sequencing problem in flowshops is the complete enumeration. The solution for all sequences on all stations is calculated. The method is not very efficient and as described before has a complexity of $(N!)^M$.
- **Linear programming**: With the help of linear programming large problems with continuous variables can be solved, provided that the objective function is linear and constraints are in the form of linear equalities and inequalities. The simplex method, first published by Dantzig, 1948, solves problems of this type. Nicholson, 1971, and Luenberger, 1984, give detailed introductions on linear programming.
- **Integer and mixed integer programming**: Integer and mixed integer programming find its application on problems which consider that all or some of the variables are constrained to integer values. The most widely used method for solving integer programs is branch and bound or derivations. Lower bounds, in the case of minimization, can be provided by the linear-programming relaxation of the problem.
- **Branch and bound**: A widely used and very promising strategy in many problems seem to be the branch and bound approach. As explained by Bellman et al., 1982, it is an implicit enumeration or tree search method which can find an optimal solution by systematically examining the subsets of feasible solutions.

  The branch and bound method was developed by Little et al., 1963, for the use on the travelling salesperson problem and modified by Ignall and Schrage, 1965, and Lomnicki, 1965, for application to flowshop sequencing. It is also called backtrack programming and uses generally one of the three seeking strategies: depth first which first explores a feasible sequence, breadth-first which completely explores the branches of the same level before descending in the tree, and frontier or branch-from-lowest-bound which explores the solution space in terms of the most promising branch. French, 1982, highlights that the first method results in more computational expense but at the same time requires less storage, compared to the latter.

  Ignall and Schrage, 1965, as well as Lomnicki, 1965, independently developed a branch and bound method for the three station case which is presented by French, 1982.

A considerable analysis of tree search methods, related with branch and bound is presented by Pastor, 1999, also including a resolution method called branch and win which is described in further detail by Pastor and Corominas, 2004.

- **Dynamic programming**: This method originates from Bellman, 1957, and can be used in multi-stage decision processes, in particular types of problem structures. Just as the branch and bound method it is limited to comparatively small problems. The method tries to reduce the dimensionality of a problem and one of the challenges is to convert the actual problem to a multi-stage decision process in order to solve it recursively. Bautista et al., 1996, furthermore present an algorithm based on bounded dynamic programming (BDP) for the sequencing problem in Just-In-Time (JIT) environments, introduced by Monden, 1983, and known as the Monden problem. Dynamic programming is used here with the additional use of bounds, obtained by solving the problem with a heuristic algorithm.

2.4.4 Approximate methods

These methods do not guarantee an optimal solution. The idea is to reach solutions that, for the sake of computation or storage, are near optimal solutions. In order to exhaustively study their performance it is indispensable to compare the obtained solution with the optimal solution obtained by an exact method. Challenging problems are found when non-linear objective functions are given. Nicholson, 1971, highlights that here the difficulty lies in the fact that a non-linear objective function is used that may have local optimum and therefore usually finds solutions that describe local optimum and do not result in the optimal solution.

2.4.4.1 Heuristics

There exist a wide range of heuristic methods and some of them are based on exact methods. The simplest being the **randomly generated solutions**. Lee and Shaw, 2000, mention that heuristic methods either belong to **improvement heuristics**, also called **neighborhood search**, or to **constructive heuristics**. Improvement heuristics attempt to continuously improve the solution by modifying the sequence, whereas constructive heuristics build a sequence of jobs and once a decision is taken, it cannot be reversed. The first usually generates better sequences at the expense of larger computation. The most simple neighborhood search starts with a feasible initial solution that is used as initial seed. In the next step solutions are evaluated that are part of the neighborhood of the seed. If none of the solutions is better than the seed with respect to the performance measure it is the final solution. Otherwise the solution that was proved to be better is used as the new seed. This procedure obviously is not capable of avoiding falling in a local optimum.

Silver, 2004, additionally annotates the separation to **inductive methods**, which first solve a mathematically related problem and extends it to the actual problem, **reduction of solution space**, here the possible solutions are drastically cut back, and **approximation methods**, here a solution for the approximated mathematical model is to be calculated.

A special case of heuristic technique is the priority dispatching rules which are used for a production with buffers between the stations that provide the possibility of selecting any of the stored jobs. Priority dispatching stands for the procedure of selecting a job from a queue with the highest priority. The priority may be calculated depending on various measures. The rules can be as simple as ”first come first serve” (FCFS), ”shortest processing time” (SPT), ”longest total remaining processing on other machines” (LTROM) but also may be a combination of simple rules. Dispatching rules are also called greedy-procedures since it makes the selection in the most favorable way, without regard to the possibilities that might arise later. Collections of existing rules can be found in Pinedo, 1995, or Blackstone et al., 1982. These dispatching rules also find application in jobshop arrangements where the routing of the individual jobs is different for each job.

2.4.4.2 Metaheuristics

The survey of Silver, 2004, on heuristic methods in operational research, describes metaheuristics as high level heuristic procedure which are designed to guide towards achieving reasonable solutions. Metaheuristics are particularly concerned with not getting trapped at a local optimum. Metaheuristics have one or more adjustable parameters which permit flexibility, but for any application requires careful calibration.

- **Simulated annealing**: This method goes through a number of iterations, comparing the current best sequence with a neighbor, see e.g. Kirkpatrick et al., 1983. Furthermore, a probability indicates if a worse sequence is chosen next. The major difference with the general neighboring search is that worse solutions are allowed in order to find an optimum that is not the next local optimum.

- **Tabu search**: There exist a Tabu-list that prevents returning to a local optimum. The procedure stops when a predetermined number of moves is performed without having improved the seed, see e.g. Glover, 1990. An essential parameter of the technique is the list size which may be represented as the short-term memory; if chosen wrong, the search may result in cycling or may be constrained unduly.

- **Genetic algorithm**: Rather than using a single initial solution this technique uses an increased number of initial solutions, see e.g. Kim et al., 1996, or Hyun et al., 1998. These solutions, called individuals, form a generation. The succeeding generation is generated through reproduction and mutation of individuals that were part of the previous generation. Individuals are also referred to as chromosomes and are characterized by their fitness, which is measured by the associated objective function. Mutation then is achieved by crossover of chromosomes. Hyun et al., 1998, highlights the two characteristics of the genetic algorithm which are exploitation and exploration. Exploitation is the ability to find good solutions quickly, whereas exploration describes the behavior of maintaining a set of diverse individuals.

- **Beam search**: Beam search is a partial branch and bound with the basic idea to eliminate the branches of the tree that are likely to not include the optimal solution, see for example Morton and Pentico, 1993. The used parameter is the beam width that is the number of nodes that are retained at each level.

- **Ant colony**: The basic concept is to mimic the pheromone trail used by real ants as a medium for communication and feedback among ants and was first proposed by Dorigo et al., 1996. The algorithm is a population based, cooperative search procedure which iteratively construct solutions, guided by (artificial) pheromone trails. Solution components are defined which the ants use to iteratively construct solutions, the ants mark the utilized components with pheromone. The pheromone then indicate the intensity of ant-trails with respect to solution components,and such intensities are determined on the basis of the influence or contribution of each solution component with respect to the objective function. Rajendran and Ziegler, 2004, apply the concept on permutation flowshop.

- **GRASP**: The GRASP (greedy randomized adaptive search procedure) is an iterative process, in which each iteration consists of two phases. A construction phase, in which a feasible solution is produced, and a local search phase, in which a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list, but not necessarily the top candidate. The list of best candidates is called the restricted candidate list (RCL). A recent survey on GRASP is presented by Festa and Resede, 2004.

2.4.5 Evaluation of the performance of a heuristic

Dannenbring, 1977, uses a set of six different evaluation measures for testing of heuristic procedures: (1) Relative error, (2) Consistency, (3) Error Potential Ratio (4) Percentage of solution equaling the optimum, the estimate of the optimum or the best heuristic solution, (5) Improvement Potential and (6) Sampling quality.

Silver, 2004, furthermore explains that two main measures of performance exist to evaluate the performance of a heuristic method. First, comparing the obtained value of the objective function to the achievable by the optimal solution or some other benchmark procedure, already mentioned in Dannenbring, 1977. Then, second, the computational requirements in terms of computational effort and memory consumption for realistic sized problems. In particular, for what percentage of the problem instances does the heuristic obtain the optimal solution. If the results are found to be sensitive, then it is helpful to specify under which conditions the heuristic should or should not be used.

In case no optimal solution is available the heuristic solution can, for a minimization, be compared with the lower bound. Silver, 2004, points out that for a small gap the difference to the optimal solution is small. For a large gap, however, a poor lower bound was used, or heuristic solution is far away from the optimal solution.

2.4.6 Literature providing input data

For the case of larger problem sizes it is more difficult to find adequate input data. This is contingent on the great variety of existent problems and the fact that many times only results are presented rather than input data. Rajendran and Chaudhuri, 1992, use a test-bed of 670 problems considering 5, 10, 15, 20 and 25 stations as well as 5, 6, 7, 8, 9 and 10 jobs, also used by Framinan et al., 2002.

Taillard, 1993, presents a widely used test-bed. A detailed explanation on the generation of input data for various production lines, including the flowshop problem is given. A random number generator is used, based on Bratley et al., 1983, that produces evenly spread values and problems with 5, 10 and 20 stations and from 20 to 500 jobs are proposed. The advantage of this generator is the repeatability in the generation of the test-bed.

In the work of Watson et al., 2002, different data sets for permutation flowshop instances are presented, without setup considerations. The provided sets are based Taillard, 1993, and next to the sample data, also the so far best known solutions, together with the lower bound, are given for permutation sequences.

The work of Brucker et al., 2003, presents a Tabu search for a flowshop with the possibility of resequencing within the production line. They position a buffer between all consecutive stations. The number of buffer places is the same for all resequencing buffers. The variable parameter is the number of buffer places which is 0, 1, 2, or infinite. The presented results consider nine instances of 20, 50 and 100 Jobs on a 5-, 10-, and 20-station flowshop. Next to the results of the Tabu search, optimal solutions from other authors are listed, and if not available, the so far best lower bounds and the best knwon solutions are presented.

2.5 Summary

The problem of sequencing in **mixed model flowshops** is known to be NP-hard already for the simple case of three stations. Johnson, 1954, one of the pioneers, proposed a solution that solves the case of two stations with optimality, known as the Johnson-rule. Here permutation sequences are dominant, i.e. the job sequence is the same for all stations. This simplification indeed is only valid until the three station case with makespan objective. If more than three stations exist, Potts et al., 1991, and Liao et al., 2006, show that improvements are possible, using resequencing capacities. This is even more evident when setup cost/time, necessary when changing from one product model to the next, appears.

In the present work a literature overview was given on solution techniques considering basic as well as more advanced and consequently more complex arrangements of mixed model flowshops. We first analyzed the occurrence of **setup time/cost**; existing solution techniques are mainly focused on permutation sequences. Thereafter we discussed objectives resulting in the introduction of a variety of methods allowing **resequencing** of jobs within the production line. The possibility of resequencing within the line ranges from 1) offline or online buffers, 2) parallel stations, namely flexible, hybrid or compound flowshops, 3) merging and splitting of parallel lines, 4) re-entrant flowshops, to 5) change of job attributes without physically interchanging the position. In continuation the differences in the consideration of **static and dynamic demand** was studied. Also intermediate setups are possible, depending on the horizon and including the possibility of resequencing, four problem cases were highlighted: static, semi dynamic, nearly dynamic and dynamic case. Finally a general overview was given on existent **solution methods**, including exact and approximation methods. The approximation methods are furthermore divided in two cases, known as heuristics and metaheuristics, depending on if the exact solution is known or not, different performance measures are applied. Furthermore, different benchmark studies which can be used for performance comparison, including the well known test-bed from Taillard, 1993.

Reviewing the literature regarding sequencing considerations in mixed model flowshops, it seems that extended work is done on the basic flowshop, mainly focusing on permutation sequences. Furthermore a remarkable amount of publications exist, solving flowshops with setup time/cost; in contrast to arrangements taking into account resequencing capabilities within the line, resulting in non-permutation sequences. Besides the fact that solution methods in the literature are not numerous, it seems that production lines in the industry are not yet considering the advantages that come with the possibilities of resequencing jobs in a mixed model production. This may either be caused by additional hardware to be installed, like buffers, but also due to extra efforts in terms of logistics complexity. Clearly, these additional efforts are reasonable if the resequencing pays off the necessary investment.

2.6 Conclusions

In this chapter the literature revision was presented with focus on flowshop production lines. The introduction of producing more than one model in the same production line is a relevant problem of today's production lines and is motivated by various reasons, e.g., offering a larger variety of products to the client, but also considerably reducing the stock for finished products with respect to a production with batches, and so are the expenses derived from it.

It was shown that the majority of publications in this area is limited to permutation sequences, whereas considering a flowshop with more than three stations and with the objective function to minimize the makespan, a unique permutation is no longer optimal. Resequencing of jobs in the line is even more relevant with the existence of an additional cost or time, occurring when at a station the succeeding job is of another model, known as setup cost and setup time.

Tables 2.1 to 2.4 give a concise overview of the existing literature which is considering resequencing. Resequencing can be found in various industrial sectors, including large volume productions as automotive or the chemical production. Exact approaches are limited to small problem sized and the use of heuristic approaches is indispensable in order to be able to successfully solve larger problems.

Based on the review of existing designs of production lines which permit resequencing of jobs and considering the conclusions of Lahmar et al., 2003, a very cost effective method of resequencing jobs within the production line is the introduction of a small number of buffer places, allowing to remove certain jobs between successive stations in order to reinsert them to the line at a later point of time. It is highlighted that the existing classifications and classifying surveys do not consider exhaustively the possibilities of resequencing jobs within the production line.

Figure 2.2 gives a summary of the possible configurations which occur when sequencing and resequencing of jobs is considered, taking into account different settings for the demand, being either static or dynamic. An overview on how optimization method may be structured is presented in figure 2.3.

The conclusion of the literature revision led to the proposal of the present thesis, contemplating a novel classification for non-permutation flowshop production lines. Followed by, a special case of non-permutation flowshop in which the possibility to resequence jobs between consecutive stations exists, using buffers which are located off the line, either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size, see figure 1.2. Due to the problem complexity an exact model with several alternative arrangements is formulated which can handle moderately sized problems. Thereafter, in order to also solve larger problems, a heuristic approach is formulated.

3 Novel Classification of Non-permutation Flowshops

As commented in section 2.6, the characteristics of the problem which are studied in this thesis and defined in section 1.3, are not considered exhaustively in previous classifications and classifying surveys (Pinedo, 1995, Vieira et al., 2003, Herrmann et al., 1993, and Lageweg et al., 1978, Plans and Corominas, 2000, Niu, 2005, Becker and Scholl, 2006). Consequently, the *Novel Classification of Non-permutation Flowshops* is proposed in this chapter.

In the basic flowshop, as introduced by Johnson, 1954, M stations are arranged in series, according to the technological sequence of the operations. A set of N jobs has to be processed on these stations. Each of the N jobs has the same ordering of stations for its processing sequence. Each job can be processed on one, and only one station at a time and each job is processed only once on each station and each station can process only one job at a time. Jobs may bypass other jobs only between stations.

The *Novel Classification of Non-permutation Flowshops* is based on the classification scheme provided by Pinedo, 1995, and adequately classifies flowshop problems with particular consideration of options and restrictions, which appear when resequencing of jobs within the production line takes place.

The notation used by Pinedo, 1995, contains the triplet $\alpha|\beta|\gamma$ and helps classifying sequencing and scheduling problems. The same triplet is used here with various additions, mainly with respect to resequencing possibilities. The triplet determines a specific problem as: α describes the station environment; β provides details on characteristics and constraints for the processing of the jobs; γ contains information on the objectives of the optimization.

3.1 General notation

The determination of flowshop problem requires the definition of a finite number of stations, being i the index and M the number of stations, and a finite number of jobs, being j the index and N the number of jobs. The processing of a job is described by:

Start time (s_j): The point of time job j starts being processed on the first station is called start time.

Processing time (P_{ij}): The processing time, also called assembly time, is the time that job j maintains at station i while being processed. Due to the nature of the flowshop, job j that is not processed at station i has to pass this station with a processing time being zero.

Completion time (c_j): The point of time job j completes processing on the last station and exits the system is called completion time.

3.2 Station environment (α)

The station environment provides relevant information on the characteristics which are related to the stations, more specific, with respect to the layout of the production line. The considered categories include the way in which the stations are arranged (station arrangement), and the way in which the stations are operated (operating properties).

3.2.1 Station arrangement

The way in which the stations are arranged significantly influences the layout of the production line and furthermore determines if resequencing of jobs is possible or not.

Flowshop (F_M): In the basic flowshop M stations are arranged in series. All jobs have the same station routing, being the primary difference to the Openshop and the Jobshop. In the basic flowshop, the *Non-permutation flowshop*, the job sequences Π_i can vary from one station to the next.

> *Single station (F_1)*: The most simple case of a production line is the one which provides only a single station. Here the jobs require to be operated on only one station. The single station case clearly can be considered a permutation flowshop.

Permutation flowshop (*perm*): The solutions are restricted to job sequences $\Pi_1, ..., \Pi_M$ with $\Pi_1 = \Pi_2 = ... = \Pi_M$, that is, the sequence on the first station is maintained for all stations in the flowshop. A set of permutation sequences is denoted dominant if no better sequence can be found than the best permutation sequence, occuring for example in the no-wait flowshop.

Flexible flowshop (*FFS*): In this special case of flowshop parallel stations exist which perform operations in parallel. The main reason for installing parallel stations is the reduction of the cycle time at a station. Furthermore, due to the fact that the processing time of a station is dependent on the model type of the job, there exist the possibility of one job overtaking its predecessor without taking it off the line. The flexible flowshop is also called *hybrid* or *compound flowshop* and, depending on the degree of similarity of the parallel stations, three different types can be considered:

> *Identical parallel station*: The parallel stations are identical and, if not defined differently in the β field, the jobs may be processed by any of the parallel stations which result in a reduction of the cycle time at a station.
>
> *Parallel stations with different speed* (*Qm*): The parallel stations have different processing times, caused by, e.g., varying operator skills or a difference in the available tools. For this reason it may be favorable for a certain job to pass through a determined parallel station.
>
> *Unrelated parallel stations* (*Rm*): The case in which the parallel stations are unrelated occurs when, e.g., tools or operator skills are provided only at certain parallel stations.

Flowline-based manufacturing system (*FBMS*): The line is similar to the basic flowshop, apart from the fact that some jobs have missing operations at some stations and can bypass the particular station. This leads to a characteristic which is not proper of the flowshop, i.e. the station precedence is not the same for all jobs. Hence, an optimum sequence can be obtained that would not be feasible if the processing time would be infinitesimally small instead of zero.

Intermediate buffer (*IntBuf*): In order for an intermediate buffer to permit resequencing, it can not be operated in *FIFO* (first-in-first-out) mode.

Offline buffer (*OffBuf*): An offline buffer is located off the production line in order to let pass other jobs. An additional handling time HT_{OffBuf} may occur which is necessary to transfer a job to and fro the offline buffer. Further distinction is done as follows:

> *First In First Out* (*FIFO*): For the case of more than one offline buffer place, this buffer can be operated in FIFO-mode. On the one hand this opens the way for simplification of the mechanics and logistics of the offline buffer, but on the other hand restricts the solutions considerably.
>
> *Automated Storage and Retrieval System* (*ASRS*): A multitude of buffer places are provided between two main parts of the production line. The reason for using large buffers in this case is to resequence the jobs in a large scale. As a result, batches are formed and each part of the line is optimized separately. Only in the case in which the optimal sequence for one part is the same as for the following, no resequencing is performed.
>
> *Intermediate/centralized location* (*int/centr*): The access to the offline buffer can be limited to only one station (intermediate case) or to various stations (centralized case). A production line, arranged in U-shape, is especially suitable for the use with a centralized offline buffer.
>
> *Physical size limitation* (*phsize*): The physical size of the individual buffer places is limited, which leads to the restriction that not every job can pass through a certain buffer place. In the case of a chemical production, a buffer place represents a tank and the physical size limitation is the provided volume. Instead of two large tanks, one large and one medium sized tank may be sufficient, which in sum on the other hand results in less investment and a reduced area occupation.

Splitting and merging of parallel lines (*split/merge*): The introduction of parallel segments of stations permits to resequence jobs where the line splits or merges. The splitting of a production line is somewhat more challenging due to the fact that two parallel lines may not perform the same options, and constraints exist that may additionally influence the sorting.

3.2.2 Operating properties

The operating properties describe the way in which a station is operated and give details on its restrictions, as for example to prohibit blocking of stations.

Paced/unpaced line (*PL*/*UPL*): In a paced production line the mechanical material handling equipment, like conveyor belts, couple the stations in an inflexible manner. The jobs are either steadily moved from station to station at constant speed or they are immediately transferred after processing. The available amount of time for the operation is the same in both cases. In the unpaced line, in contrast, the stations are decoupled by buffers. In a specific case these buffers store jobs that can not be passed to the downstream station which is still occupied with processing the previous job.

Blocking (*block*): Blocking can occur in a flowshop when between two succeeding stations only a limited number of buffer places is provided. When all buffer places are occupied, the upstream station can not be unloaded and is blocked from further processing. The flag *block* is used to indicate that only schedules are feasible that do not result in blocking.

Zero-buffer (*ZeroBuffer*): This variation of the basic flowshop does not allow the jobs to form queues between the stations. A job j leaving station i cannot advance to station $i+1$ if there is still a job being processed. Station blocking of station i is the result.

No-idle (*noidle*): This constraint implies that each station, once started with processing, has to process all operations assigned to it without interruption. A real life situation can be found, for example, if machines represent expensive pieces of equipment which have to be rented for the duration between the start of its first operation and the completion of its last operation.

Station breakdown (*breakdown*): Station breakdown describes the state of a station which does not permit processing of any job due to failure. In real production systems the breakdowns occur in a stochastic way and can be simulated using the values Mean-Time-Between-Failure (MTBF) and Mean-Time-To-Repair (MTTR).

Station maintenance (*maintenance*): Station maintenance describes the state of a station which does not permit processing of any job due to prevention. In contrast to the station breakdown, the maintenance occurs in a deterministic way and usually with regular cycles, demanded by the tool manufacturer.

3.3 Job processing environment (β)

The Job processing environment provides relevant details on characteristics and constraints for the processing of the jobs. The considered categories include the demand, time and cost constraints and processing restrictions.

3.3.1 Demand

The production planning for mixed model flowshops can be various and usually depends on the planning horizon and in some cases on the possibility of decoupling the customer orders from the production planning.

Single model (D_{single}): The simplest case is the single model production where only one type of product is produced.

Multi model (D_{multi}): In the multi model production the products form lots of the same model, which are then produced in batches.

Mixed model (D_{mixed}): In the mixed model production the job sequence is not determined by batches and therefore allows an arbitrary order.

> *Minimal part set (D_{MPS})*: The most common representations for the mixed model case is the use of the minimal part set which is the least common multiple of the individual models for the entire demand.

Launch interval fixed/variable (D_{Lifix}/D_{Livar}): In a paced line, the time between two consecutive jobs entering the production line is called launch-interval or cycle time. The fixed launch interval results in a constant production rate (production quantity per unit of time). The variable launch interval gives more flexibility, resulting in better solutions.

Static/dynamic demand (D_{stat}/D_{dyn}): The static demand refers to the fact that the entire demand, necessary to produce in a time window, is produced in an accumulated lot, known beforehand. Whereas, the dynamic demand implies that the customer orders arrive continuously or at least are not completely determinable beforehand.

Priority (w_j): The priority of job j is determined by its weight w_j which defines its importance with respect to the other jobs.

3.3.2 Time and cost restrictions

Restrictions which are directly related to the processing of jobs can depend on an individual job, as in the case of the release time, or can furthermore depend on the previous job, as in the case of the setup time:

Release time (R_j): The earliest time at which job j can start its processing.

Due time (D_j): The due time is either the latest possible time job j may leave the production line, or it is the time at which the job should be finished.

Setup cost (SC_{efi}): A setup cost is concerned if an additional cost appears to change the setup of station i, in order to be able to process job $j + 1$ which is of model f after job j which is of model e. If the setup cost is independent of the model, it can be simply added to the processing cost.

Setup time (ST_{efi}): In a similar way a setup time is concerned if an additional time appears to change the setup of station i, in order to be able to process job $j + 1$ which is of model f after job j which is of model e. If the setup time is independent of the model, it can be simply added to the processing time.

Handling time (HT): The way in which the products are passed from one station to the next can be classified by its degree of automation. In contrast, this option is referred to the existence of a handling time HT which occurs when passing a job from one station to the next or between a station and a buffer.

Deterministic/Stochastic (*det/stoch*): The processing times, also including, e.g., setup times, are generally regarded to be deterministic. The more automated the production is, the more likely it is that these deterministic values are met. In a realistic production, depending on human operators, tool accuracy, the punctuality of suppliers, and where machine breakdowns occur, it may be desirable to include stochastic uncertainty to processing times. Another case of varying processing times is the learning process of the operator:

Learning of operator (*learning*): A human operator, who is new to the processes of a certain station may not perform his tasks with the same velocity as after some time when he starts to experience routine.

3.3.3 Processing restrictions

Preemption (*prmp*): Preemption is referred to the case in which it is not necessary to finish the processing of a certain job at once. It is allowed to interrupt its processing in order to process another job and to continue with the interrupted job at a later point of time. When preemption is allowed, bypassing of jobs can occur.

Precedence (*prec*): The production of jobs may be constrained by precedence which is that a Job may not start processing before a certain job started or even completed. Two more specific cases of job precedences are

> *Strict precedence ($prec_{strict}$)*: Between two jobs of the same model or two jobs that require the same option at a station, a minimum number of different jobs or a minimum time is necessary.
>
> *Quality implication ($prec_{QI}$)*: The quality implications may affect the performance of a station. For example the paint quality may temporarily decline when a change of color occurs.

No-wait (*nwt*): This case is more restrictive than the *Zero-buffer* case and comprehensively similar to the *No-idle* case, however, instead of ensuring a station not to stop processing, here it has to be ensured that the jobs are not left without being processed. Once a job begins its processing on station 1, that job must continue without delay to be processed on each of the M stations. Only sequences are feasible which do not result in blocking of any station.

Re-entrant (*reent*): The re-entrant, or recirculation flowshop, considers that a job runs more than once through the same line, e.g., first to assemble the bottom side and then to assemble the top side of a printed-circuit-board.

Change of Job attributes (*jobatt*): Instead of physically changing the job order, swapping of jobs appears by changing their attributes. Rather than a resequencing facility, the change of job attributes is a method which requires logistical implement in the production flow of the plant.

Station eligibility (SE_j): When parallel stations exist (*FFS*), a station may not be able to process a certain job. SE_j determines the set of stations which can process job j.

3.4 Objectives (γ)

The environment of the objectives of the optimization firstly provides information on the purpose of the optimization, and then on the objective function which is used in the optimization itself.

3.4.1 Purpose of the optimization

The optimization of a flowshop production line is basically divided into two phases, the design phase and the operation phase. In the first phase the tasks are assigned to the stations, subject to technological precedence relations, and is called the line balancing problem. Once the line is balanced and the design of the line is obtained, it is necessary to achieve a reasonable, if not optimum, order for the jobs to be processed consecutively, being the operating phase with sequencing and scheduling of the jobs.

Balancing (*Balancing*): The process of balancing the load results in the design of the production line, usually implying the minimization of the station number and the determination of a cycle time, obtained by calculating, e.g., an average of the processing times, necessary to assemble the various models. The balancing procedure in many cases results in the prevention of the occurrence of bottlenecks so that the final production line will not experience stoppage and unnecessary inventory will not accumulate. Studies on the production line balancing problem are numerous and may be subject to various criteria like cost-oriented or profit-oriented approaches.

Sequencing (*Sequencing*): The sequencing problem determines an appropriate order for the jobs to be processed within, e.g., the shortest possible time called makespan. As a result of the sequencing problem, a schedule is obtained which specifies the starting time for each job at each station.

Scheduling (*Scheduling*): Even though, within the sequencing problem the term *schedule* is used, here a distinction is made between the two terms sequencing and scheduling, based on Beaty, 1992. In addition to determining an appropriate order for the jobs, the scheduling problem contemplates prioritizing the order of the jobs due to resource usage and e.g. due-date-limits of the jobs.

Unexpected Incidents (*Incidents*): A problem which is classified *Incidents*, form part of the sequencing/scheduling phase but determines a type of problem which reacts in a more dynamic way to a production which is already up and running, and most of all, is confronted with unexpected job or station related incidents:

> *Station incidents* ($Rescheduling_{Stat}$): Station breakdown, shortage of material, operator absenteeism, maintenance, etc.
>
> *Job incidents* ($Rescheduling_{Job}$): Job cancellation, urgent job arrival, due time change, delay in arrival, job priority change, rework or quality problems, over- or underestimation of processing times, etc.

These unexpected incidents lead to rescheduling, even if the rescheduling results in the confirmation that the current production is not influenced, but may also trigger the following steps: Overtime, in-process subcontracting, process change or re-routing, station substitution, limitation of manpower, setup times, etc.

Even though the outcome of unexpected incidents may result in the fact that the jobs are sequenced in a different order than before, here, the term *Resequencing* is used in a different context, indicating that the job sequence from one station to the next may be different and jobs have to be resequenced.

3.4.2 Objective function

Within the mixed-model flowshop a variety of objective functions are to be found, the most common being time and cost orientated objectives. As a basic principle of optimization, the considered solutions are part of a set of feasible solutions. In what follows, the most common objectives are listed, followed by objectives which are specific for resequencing.

Time orientated objectives:

> *Makespan ($c_{\max}$)*: One of the most common objective functions in sequencing is to minimize the maximum completion time necessary to process the entire demand, called makespan or total production time. The makespan optimization generally ensures high utilization of the production resources, early satisfaction

of the customer demand and the reduction of in-process inventory by minimizing the total production run.

Makespan:

$$\max\{c_j | j = 1, ..., i\}$$

Maximum flow time ($F_{\max}$): The minimization of the maximum flow time leads to stable and even utilization of production resources, rapid turn-around of jobs and the minimization of in-process inventory. In the case where all release dates are zero, $c_{\max}$ and F_{max} are identical. The weighted maximum flowtime includes a weight related to the jobs.

Maximum flow time:

$$\max\{c_j - s_j | j = 1, ..., N\}$$

Weighted maximum flow time:

$$\max\{\omega_j(c_j - s_j) | j = 1, ..., N\}$$

Mean flow time ($\overline{F}$): The mean flow time leads to similar results as the maximum flow time, but tries to average the flow time of all jobs and therefore leads to a more balanced flow of jobs. The weighted mean flowtime includes a weight related to the jobs.

Mean flow time:

$$\frac{1}{N}\sum_{j=1}^{N}(c_j - s_j)$$

Weighted mean flow time:

$$\frac{1}{N}\sum_{j=1}^{N}\omega_j(c_j - s_j)$$

Setup time (ST): In a mixed model production, setup time ST_{efi} may occur when at station i a job $j+1$ of model type f follows job j of model type e. Minimizing total setup time, furthermore, tends to decrease the total flowtime.

Setup time:

$$\sum_{j=1}^{N} ST_{efi}$$

Idle time (I): Idle time I_{ij} at station i occurs when an operator is kept waiting for job j. This may be caused by a job that has not yet arrived, or because an auxiliary operator is still occupied with the job. When setup time occurs, that is separable from the processing time, the operator can benefit from this idle time in order to perform the necessary changes for the next job to be processed.

Idle time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}$$

Mean idle time:

$$\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}$$

Utility time (U): Utility time U_{ij} at station i occurs when an operator has to continue with job $j+1$ before finishing with job j. In this case an auxiliary operator finishes the job; the time the auxiliary operator requires is called utility time. As well as the idle time, here the mean is taken over the stations.

Utility time:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} U_{ij}$$

Mean utility time:

$$\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N} U_{ij}$$

Earliness and tardiness of jobs (E, T): Earliness and tardiness is regarded in the scheduling problem when due times appear. The earliness and tardiness can be considered in different ways, e.g. the maximum value or the mean value. In general it is common practice to consider the minimization of the tardiness as the more important objective of the two, in order to satisfy the client. On the other hand, when jobs are finished early, they have to be stored until shipping which obviously implies an additional cost for storage.

Production rate (PR): The objective of minimizing the production rate is relevant in a production system which uses, e.g., a conveyer belt with fixed distances or time intervals for the jobs which are launched into the production line. Besides a constant production rate, examples can be found which use a variable production rate for further optimization.

Deviation: In general for all of the above mentioned time oriented objectives it is possible to use the deviation, or the squared deviation, over stations or over jobs, in order to equalize the deviation and to avoid solutions that provide extreme values for single stations or jobs.

Cost orientated objectives:

Line length (Length): The length of the production line is related to the production costs. This can be the initial investment necessary to construct the line, as well as the cost that are necessary to keep the production up and running.

Line length:

$$\sum_{i=1}^{M} L_i$$

Setup cost (SC): The occurrence of setup cost may lead to the objective of minimizing the total setup cost to keep the production costs small. Setup cost SC_{efi} may occur when at station i a job $j+1$ of model type f follows job j of model type e.

Setup cost:

$$\sum_{i=1}^{M} SC_{efi}$$

Constant part usage (CPU): A well known objective for Just-In-Time productions is the constant part usage. The main idea behind this objective is to simplify the material handling and as a result to reduce the need for extra manpower which would be used to meet the peaks in part usage.

Combined objectives: In the literature, the use of individual objective functions, as mentioned above, as well as combinations can be found.

Resequencing: The major objective of resequencing in flowshops is further minimization of production costs, for example resulting in a higher utilization of the production resources. This is desirable even more when setup cost/time is involved or the processing times of the individual jobs diverge among one another.

Minimize number of jobs to be resequenced ($Resequ_{Jobs}$): Resequencing a job results in an additional effort. Therefore, if two sequences, resulting in a different number of jobs to be resequenced, provide the same value of the objective function, the one with fewer resequencing is to be preferred.

Minimize load-unload time ($Resequ_{LoadTime}$): When considering a handling time HT, which occurs for transferring a job to or from a buffer place, an objective may be to minimize the accumulated time caused by this handling. However, the calculation of the makespan already indirectly considers the handling time.

Minimize travel time of jobs in ASRS-buffers ($Resequ_{TravelTime}$): When an ASRS-buffer (Automated Storage and Retrieval System) is used, large travel distances occur, which result in a notable travel time. Here, the objective is to minimize the total accumulated travel time.

Undo undesired resequencing ($Resequ_{Undo}$): Apart from the aim of further optimizing the makespan or reducing setup cost or time, there exist objectives such as to undo undesired resequencing which is caused by unexpected incidents, such as rework, parallel inspection stations, unequal processing times on parallel stations or problems in part delivery.

3.5 Resuming table

Station environment (α)

- **Station arrangement**
  - Flowshop (F_M)
    - *Simple Station case (F_1)*
  - Permutation flowshop (*perm*)
  - Flexible flowshop (*FFS*)
    - *Identical Paralel Station*
    - *Parallel Stations with different speed (Qm)*
    - *Unrelated parallel stations (Rm)*
  - ▨ Flowline-based manufacturing system (*FBMS*)
  - ▨ Intermediate buffer (*Int Buf*)
  - ▨ Offline buffer (*OffBuf*)
    - ▨ *First-In-First-Out (FIFO)*
    - ▨ *Automated Storage and Retrieval System (ASRS)*
    - ▨ *Intermediate/centralized location (int/cent)*
    - ▨ *Physical size limitation (phsize)*
  - ▨ Splitting and merging of parallel lines (*merge/split*)
- **Operation properties**
  - Paced/unpaced line (*PL/UpL*)
  - Blocking (*block*)
  - Zero-buffer (*zerobuffer*)
  - No-idle (*noidle*)
  - Station breadown (*breakdown*)
  - Station Maintenance (*maintenance*)

Job processing environment (β)

- **Demand**
  - Single Model (D_{single})
  - Multi Model (D_{multi})
  - Mixed Model (D_{mixed})
    - *Minimal-Part-Set (MPS)*
  - Launch intervall fixed/ variable ($D_{lifixed}/D_{livar}$)
  - Static/Dynamic Demand (D_{stat}/D_{dyn})
  - Priority (ω_j)
- **Time and cost restrictions**
  - Release time (R_j)
  - Due time (D_j)
  - Setup cost (*SC*)
  - Setup time (*ST*)
  - ▨ Handling time (*HT*)
  - Deterministic/Stochastic (*det/stoch*)
  - Learning of operator (learning)
- **Processing restrictions**
  - Preemption (*prmp*)
  - Precedence (*prec*)
    - *Strict precedence ($prec_{strict}$)*
    - ▨ *Quality implication ($prec_{QI}$)*
  - No-Wait (*nwt*)
  - ▨ Re/entrant (*reent*)
  - ▨ Change of Job-attributes (*jobatt*)
  - Station eligibility (SE_j)

Objectives (γ)

- **Purpose of optimization**
  - ▨ Balancing (*Balancing*)
  - ▨ Sequencing (*Sequencing*)
  - ▨ Scheduling (*Scheduling*)
  - ▨ Unexpected Incidents (*Incidents*)
    - ▨ *Station incidents ($Rescheduling_{Stat}$)*
    - ▨ *Job incidents ($Rescheduling_{Job}$)*
- **Objective function**
  - Time orientated objectives
    - *Makespan (c_{max})*
    - *Maximum flow time (F_{max})*
    - *Mean flow time ($\bar{F}$)*
    - *Setup time (ST)*
    - *Idle time (I)*
    - *Utility time (U)*
    - *Earliness, Tardiness (E, T)*
    - ▨ *Production rate (PR)*
  - Cost orientated objectives
    - ▨ *Line length (Length)*
    - ▨ *Setup cost (SC)*
    - ▨ *Constant part usage (CPU)*
  - ▨ Resequencing
    - ▨ *Minimize resequenced jobs ($Resequ_{Jobs}$)*
    - ▨ *Minimize load-unload time ($Resequ_{LoadTime}$)*
    - ▨ *Minimize travel time ($Resequ_{TravTime}$)*
    - ▨ *Undo undesired resequencing ($Resequ_{Undo}$)*

▨ Cells with grey background are not included in the classification of Pinedo 1995.

3.6 Conclusions

As shown in the literature review, there exist various classifications and classifying surveys, as for example Pinedo, 1995, Vieira et al., 2003, Herrmann et al., 1993, and Lageweg et al., 1978, Plans and Corominas, 2000, Niu, 2005, Becker and Scholl, 2006, none of which considers exhaustively the possibilities of resequencing jobs within the production line.

The presented *Novel Classification on Non-permutation Flowshops* is based on the classification by Pinedo, 1995, and furthermore includes various extensions in order to meet the requirements for resequencing jobs within the production line.

The classification is intended to adequately classify non-permutation flowshop problems and uses the triplet $\alpha|\beta|\gamma$, determining a specific problem as: α describing the station environment; β providing details on characteristics and constraints for the processing of the jobs; γ containing information on the objectives of the optimization.

As a matter of fact, the versatile facilities and methods for resequencing jobs within the production line were contemplated, such as: using large buffers (Automatic-Storage-and-Retrieval-System) which decouple one part of the line from the rest of the line; buffers which are located offline; hybrid or flexible lines; and more seldom,the interchange of job attributes instead of physically changing the position of a job within the sequence. Also a wide range of additional characteristics, as for example, the possible handling times, which occur when offline buffers are used, are taken into account. Furthermore rescheduling is included as an important purpose of the optimization, and various objectives which are related to resequencing.

The elaborated instrument can be used for properly categorizing the diversity of flowshop problems, in order to simplify their comparison and to improve the possibilities to finding new configurations that are not yet investigated and may lead to further optimization. The special case of non-permutation flowshop, studied in continuation, is classified as follows:

Intermediate case

$(F_m, \mathit{OffBuf}_{int,phsize} \mid D_{mixed,Lifix,static}, SC_{det}, ST_{det} \mid Sequencing, C_{max}, ST, SC)$

Centralized case

$(F_m, \mathit{OffBuf}_{cent,phsize} \mid D_{mixed,Lifix,static}, SC_{det}, ST_{det} \mid Sequencing, C_{max}, ST, SC)$

4 Exact Approach: CLP Constrained Logic Programming

The problem under study is a special case of a non-permutation flowshop in which there exists the possibility to resequence jobs between consecutive stations, using buffers which are located off the line, either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size, see figure 1.2.

The exact approach solves optimally the problem of sequencing jobs in the flowshop with respect to the objective function. As a result of its complexity and, as can be seen in the experimentation in chapter 5, clearly the number of jobs, stations and buffer places in the exact approach is limited. In order to also solve larger problems, a hybrid CLP is proposed and in continuation, in chapter 6, a heuristic approach, using a Genetic Algorithm, is formulated, followed by the experimentation in chapter 7 and the performance comparison of the two approaches in chapter 8.

In order to present the complete formulation, containing the desired resequencing capabilities, the necessary constraints are introduced starting with the most basic formulation, the non-permutation flowshop, permitting unrestricted exchange of jobs between stations. Thereafter, a constraint is introduced in order to avoid unnecessary job changes, mainly for the case in which zero processing time exists, then constraints for setup cost and setup time are added. The introduction of finite buffers lead to restricted resequencing of jobs within the line. The formulation for the most restricted case, the permutation flowshop is introduced, not permitting any resequencing. The formulation then is enlarged by introducing intermediate resequencing buffers, located between two consecutive stations, with the possibility of taking off jobs from the production line and introducing them at a later point of time. The formulation then is extended to a centralized buffer which is accessible by more than one station. Finally, computational enhancements are introduced which is to impose the start time of the jobs and to reduce the variable size.

4.1 Index definition

The following indices are used throughout this document:

j, k	Index for jobs
i, h	Index for station
d	Index for buffer places

4.2 Parameter definition

The following parameters are used throughout this document:

Job related parameters:

M	Number of stations	$i, h = 1, ..., M$
N	Number of jobs	$j, k = 1, ..., N$
$P_{i,j}$	Processing time of job j at station i	
μ_j	Model type of job j	
$SC_{i,e,f}$	Setup cost occurring at station i to change from model type e to model type f	
$ST_{i,e,f}$	Setup time occurring at station i to change from model type e to model type f	
ϕ_j	Physical size of job j	$\phi_j = 1, 2, ...$

Let $P_{i,j}$ be the processing time of job j at station i, whereas $P_{i,[j]}$ is the processing time of the job at position j at station i. This convention is also applicable for other parameters and variables.

Parameters used with **intermediate** resequencing buffers:

D	Maximum Number of buffer places	$d = 1, ..., D$
B_i	A buffer, permitting resequencing, is located after station i if $B_i = 1$.	$B_i \in \{0, 1\}$ $i = 1, ..., M - 1$
$\Phi_{i,d}$	A buffer with a maximum of D buffer-places, permitting resequencing, is located after stations i. The argument of $\Phi_{i,d}$ determines the permitted job size.	$\Phi_{i,d} = 0, 1, 2, ...$ $i = 1, ..., M - 1$

Parameters used with the **centralized** resequencing buffer:

B'_i	Indicates if a job, after leaving station i, may be passed to the centralized resequencing buffer in order to be stored temporarily.	$B_i \in \{0, 1\}$ $i = 1, ..., M - 1$
Φ'_d	The resequencing buffer with D buffer-places is located as centralized buffer, accessible from various or all stations. The argument of Φ'_d determines the permitted job-size.	$\Phi'_d = 0, 1, ...$

Weight for objective function

α	Weight for makespan	$\alpha = [0..1]$
β	Weight for change of job-position	$\beta = [0..1]$
γ	Weight for setup cost	$\gamma = [0..1]$
δ	Weight for setup time	$\delta = [0..1]$

4.3 Variable definition

The following variables are defined:

Job related variables:

$s_{i,j}$	Start time of job j at station i	
$c_{i,j}$	Completion time of job j at station i	
Π_i	Job sequence at station i	$\Pi_i = \{\Pi_{i,1}, ..., \Pi_{i,N}\}$
$\Pi_{i,j}$	Job j at station i.	
$\Pi'_{i,j}$	Position of Job j at station i in the sequence $\Pi_{i,j}$.	
$\lambda_{i,[j],[k]}$	Indicates if the job at position k succeeds the job at position j at station i as well as at station $i+1$.	$\lambda_{i,[j],[k]} \in \{0,1\}$ $i = 1, ..., M-1$
$\Lambda_{i,[j]}$	Indicates if the job at position j requires to be taken off the line after station i for the purpose of resequencing.	$\Lambda_{i,[j]} \in \{0,1\}$ $i = 1, ..., M-1$

Variables used with **intermediate** resequencing buffers:

$\Delta_{i,[j],[k]}$	Indicates if at the point of time the job at position j is taken off the line after station i, the job at position k has been stored temporarily in the resequencing buffer.	$\Delta_{i,[j],[k]} \in \{0,1\}$ $i = 1, ..., M-1$
$\psi_{i,d,[j]}$	Indicates if the job at position j, after leaving station i, is assigned to the buffer place d of the respective buffer.	$\psi_{i,d,[j]} \in \{0,1\}$ $i = 1, ..., M-1$
$\Psi_{i,[j]}$	The argument of $\Psi_{i,[j]}$ indicates the buffer place to which the job at position j is assigned when leaving station i.	$\Psi_{i,[j]} \in \{1, ..., D\}$ $i = 1, ..., M-1$

Variables used with the **centralized** resequencing buffer:

$\Delta'_{(i-1)\cdot N+[j], (h-1)\cdot N+[k]}$	Indicates if at the point of time the job at position j is taken off the line after station i, the job at position k, after station h, has been temporarily stored in the resequencing buffer .	$i, h = 1, ..., M-1$ $\Delta'_{(i-1)\cdot N+[j], (h-1)\cdot N+[k]} \in \{0,1\}$

$\psi'_{i,d,[j]}$	Indicates if the job at position j, after leaving station i, is assigned to the buffer-place d of the centralized buffer.	$\psi'_{i,d,[j]} \in \{0,1\}$ $i = 1, ..., M-1$
$\Psi'_{i,[j]}$	The argument of $\Psi'_{i,[j]}$ indicates the buffer-place of the centralized buffer to which the job at position j is assigned after leaving station i.	$\Psi'_{i,[j]} \in \{1, ..., D\}$ $i = 1, ..., M-1$

4.4 Unconstrained resequencing

Flowshops with unconstrained resequencing have buffers with infinite places located after each station. The determination of the job sequences Π_i for all stations is independent of limited buffer places or the absence of a buffer between two consecutive stations.

The flowshop formulation derived in this section is the (basic) flowshop with unconstrained resequencing, called **non-permutation flowshop**. Furthermore setup time and setup cost is considered, the latter does not directly influence the schedule.

As a simple example for the possible savings which may already occur in the case without any setup considerations, table 1.1 shows the processing time of two jobs which are to be processed at four stations. Figure 1.1 shows the optimal makespan, being $c_{max} = 18$ for the non-permutation sequence. The permutation sequence would result in a makespan of $c_{max} = 23$.

4.4.1 Non-permutation flowshop

The basic flowshop, called the Non-permutation flowshop, permits arbitrary sequencing of jobs for all stations. In other words, the job sequence used for the first station does not have to be the same for the second station.

The problem can be formulated as follows:

$$\text{Minimize } (s_{M,[N]} + P_{M,[N]}) \tag{4.1}$$

subject to

$$s_{i,[j]} + P_{i,[j]} \leq s_{i+1,[j]} \qquad i = 1, ..., M-1; \forall j \tag{4.2}$$

$$\begin{aligned}((s_{i,[j]} + P_{i,[j]} \leq s_{i,[k]}) \wedge (\Pi^{'}_{i,j} < \Pi^{'}_{i,k}))\vee \\ \vee((s_{i,[k]} + P_{i,[k]} \leq s_{i,[j]}) \wedge (\Pi^{'}_{i,k} < \Pi^{'}_{i,j})) \qquad \forall i; \forall j,k|_{j\neq k}\end{aligned} \tag{4.3}$$

$$\Pi_{i,[k]} = j \;\;\rightarrow\;\; \Pi^{'}_{i,j} = k \qquad \forall i,j,k \tag{4.4}$$

The objective function (4.1) minimizes the makespan, i.e., the time the last job completes processing on the last station. Constraint (4.2) specifies the station precedence, i.e., after finishing processing at station i the job has to move to station $i+1$ (flowshop). Constraint (4.3) ensures that only one job can be processed at a station. Apart from that it determines $\Pi^{'}_{i,j}$, indicating the position of job j at station i. If job j at station i completes before job k starts, job j appears at an earlier position than job k, and vice versa. Furthermore, constraint (4.4) transforms $\Pi_{i,k}$ to $\Pi^{'}_{i,j}$, indicating that the job j at station i is sequenced at position k.

4.4.2 Unnecessary job changes

One of the inherent characteristics of the flowshop is that all jobs have to pass through all stations, even though a job j may have zero processing time at a station (i.e., $P_{i,j} = 0$). It may be desired to avoid unnecessary job changes, resulting in the same makespan. In the case of zero processing time it is very likely for the found solution to contain unnecessary job changes and in order to avoid them, the objective function (4.1) is extended to the objective function (4.5), using the weights α for the makespan and β for the sum of the number of job changes.

$$\text{Minimize } (\alpha \cdot \; (s_{M,[N]} + P_{M,[N]}) + \beta \cdot \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} \Lambda_{i,[j]}) \tag{4.5}$$

Additionally, in order to determine whether a job requires to be taken off the production line, with the aim of reinserting it at a later point of time, two constraints are used, (4.6) and (4.7).

> $\lambda_{i,[j],[k]}$ indicates if the job at position k succeeds the job at position j at station i as well as at station $i+1$. In this case $\lambda_{i,[j],[k]}$ is forced to 1.
>
> $$\begin{aligned} \Pi'_{i+1,\Pi_{i,j}} < \Pi'_{i+1,\Pi_{i,k}} &\rightarrow \lambda_{i,[j],[k]} = 1 \qquad i = 1, ..., M-1; \forall j,k|_{j<k} \qquad (4.6)\\ \Pi'_{i+1,\Pi_{i,j}} \geq \Pi'_{i+1,\Pi_{i,k}} &\rightarrow \lambda_{i,[j],[k]} = 0 \qquad i = 1, ..., M-1; \forall j,k|_{j<k}\\ &\quad\;\; \lambda_{i,[j],[k]} = 0 \qquad i = 1, ..., M-1; \forall j,k|_{j\geq k} \end{aligned}$$
>
> Then, $\Lambda_{i,[j]}$ determines if the number of jobs succeeding the job at position j at station i and at station $i+1$ is not the same as the number of jobs to succeed the job at position j at station i. In this case $\Lambda_{i,[j]}$ is 1, indicating that job j occupies a buffer place of the resequencing buffer after leaving station i.
>
> $$\begin{aligned} \sum_{k=1}^{N} \lambda_{i,[j],[k]} \neq N-j &\rightarrow \Lambda_{i,[j]} = 1 \qquad i = 1, ..., M-1; \forall j \qquad (4.7)\\ \sum_{k=1}^{N} \lambda_{i,[j],[k]} = N-j &\rightarrow \Lambda_{i,[j]} = 0 \end{aligned}$$

4.4.3 Setup cost

The problem of sequencing jobs of different models implies that a station may need modification of its arrangements, which may result in additional costs and should be regarded in the objective function. The minimization of the production costs may lead to a different schedule with reduced production costs but possibly resulting in a longer makespan.

$$\text{Minimize } (\alpha \cdot (s_{M,[N]} + P_{M,[N]}) + \gamma \cdot \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} SC_{i,\mu_{i,[j]},\mu_{i,[j+1]}}) \qquad (4.8)$$

In order to incorporate the setup cost, the objective function (4.8) includes the additional weight γ for the sum of the setup cost $SC_{i,e,f}$, occurring when a job of model type e precedes a job of model type f at station i.

4.4.4 Setup time

Similar to the consideration of the setup cost, there exists an additional time, when a model type change takes place at a station. In contrast to the case where setup cost is regarded, the occurrence of setup time directly influences the schedule.

$$\text{Minimize } (\alpha \cdot \ (s_{M,[N]} + P_{M,[N]}) + \delta \cdot \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} ST_{i,\mu_{i,[j]},\mu_{i,[j+1]}}) \tag{4.9}$$

$$\begin{aligned} &((s_{i,[j]} + P_{i,[j]} + ST_{i,\mu_{i,[j]},\mu_{i,[k]}} \leq s_{i,[k]}) \wedge (\Pi^{'}_{i,j} < \Pi^{'}_{i,k}))\vee \\ &\vee((s_{i,[k]} + P_{i,[k]} + ST_{i,\mu_{i,[k]},\mu_{i,[j]}} \leq s_{i,[j]}) \wedge (\Pi^{'}_{i,k} < \Pi^{'}_{i,j})) \qquad \forall i; \forall j,k|_{j\neq k} \end{aligned} \tag{4.10}$$

In order to incorporate the setup time, the objective function (4.9) is defined as the weighted sum of the makespan and the setup time. The constraint (4.3) is substituted by constraint (4.10) also including the setup time $ST_{i,e,f}$, existing when a job of model type e precedes a job of model type f at station i.

4.5 Constrained resequencing

Constrained resequencing is referred to the fact that jobs may not be taken off the line in any arbitrary manner.

4.5.1 Permutation flowshop

The most restricted case is the permutation flowshop. The job sequences Π_i for all stations i is the same, in other words, consists of the same permutation.

The problem can be formulated as follows:

$$\text{Minimize } (s_{M,[N]} + P_{M,[N]})$$

subject to

$$s_{i,[j]} + P_{i,[j]} \leq s_{i+1,[j]} \qquad i = 1, ..., M-1; \forall j$$

$$\begin{aligned}((s_{i,[j]} + P_{i,[j]} \leq s_{i,[k]}) \wedge (\Pi'_{i,j} < \Pi'_{i,k}))\vee \\ \vee((s_{i,[k]} + P_{i,[k]} \leq s_{i,[j]}) \wedge (\Pi'_{i,k} < \Pi'_{i,j})) \qquad \forall i; \forall j,k|_{j\neq k}\end{aligned}$$

$$\sum_{i=1}^{M} |s_{i,[j]} - s_{i,[k]}| \leq |\sum_{i=1}^{M}(s_{i,[j]} - s_{i,[k]})| \qquad \forall j,k|_{j>k} \tag{4.11}$$

The additional constraint (4.11) restricts the possible solutions to permutation sequences. $s_{i,[j]} - s_{i,[k]}$ is the difference of start time for two jobs at position j and k at station i. This difference results positive if the job at position k precedes the job at position j. If the absolute value of the sum over all stations is equal to the sum of the absolute values (see table 4.1), the jobs at position j and k are processed in the same sequence in all stations, resulting in a permutation sequence.

Case	1	2	3	4
Π_i	$j\ k$	$j\ k$	$k\ j$	$k\ j$
Π_{i+1}	$j\ k$	$k\ j$	$j\ k$	$k\ j$
$s_{i,[j]} - s_{i,[k]}$	+	+	-	-
$s_{i+1,[j]} - s_{i+1,[k]}$	+	-	+	-
$s_{i,[j]} - s_{i,[k]}$	7	7	-7	-7
$s_{i,[j]} - s_{i,[k]}$	1	-1	1	-1
$\sum_i^{i+1} \|s_{i,[j]} - s_{i,[k]}\|$	8	8	8	8
$\|\sum_i^{i+1}(s_{i,[j]} - s_{i,[k]})\|$	8	6	6	8

Table 4.1: Calculation of start time difference for two consecutive jobs j and k.

4.5.2 Intermediate resequencing buffer

Buffers are used which permit to remove jobs from the line temporarily in order to introduce them at a later point of time. The resequencing buffer is located between two consecutive stations i and $i+1$ and does only permit to remove jobs between these two stations. In the case in which no resequencing buffer is located between two station, resequencing is not possible.

4.5.2.1 Basic formulation

Resequencing is restricted to determined buffer locations. The number of buffer-places at each buffer is infinite, hence, more than one job can be taken off the line temporarily for resequencing.

$$\sum_{i}^{i+1} |s_{i,[j]} - s_{i,[k]}| \leq |\sum_{i}^{i+1}(s_{i,[j]} - s_{i,[k]})| \qquad \forall j,k|_{j>k} \qquad i = 1,...,M-1|_{B_i=0} \tag{4.12}$$

The formulation is similar to the one presented for the permutation flowshop. In contrast to constraint (4.11), constraint (4.12) considers two consecutive stations i and $i+1$. The binary vector B_i determines if resequencing is permitted between station i and the consecutive station $i+1$. In the case in which resequencing is not permitted, B_i is equal to 0, no resequencing buffer is located after station i.

4.5.2.2 Constrained resequencing buffer

The size of the resequencing buffer places is constrained which results in the fact that only jobs may be temporarily stored in a buffer place which meets the respective job size. The resequencing buffer has infinite buffer places and all buffer places at a station have the same physical size.

$$\begin{aligned}(\sum_{i}^{i+1} |s_{i,[j]} - s_{i,[k]}| \leq |\sum_{i}^{i+1}(s_{i,[j]} - s_{i,[k]})|)\vee \\ \vee((s_{i+1,[j]} - s_{i+1,[k]}) \leq 0) \qquad \forall j,k|_{j\neq k}; \\ i = 1,...,M-1|_{\phi_{\Pi_{i,[j]}}>\Phi_i}\end{aligned} \tag{4.13}$$

Constraint (4.12) is extended to constraint (4.13), permitting resequencing between two consecutive stations if the buffer permits to store the respective job size. Vector $\phi_{\Pi_{i,[j]}}$ indicates the size of the job at position j at station i and vector Φ_i indicates the biggest permitted job size which may be stored in the buffer located between station i and station $i+1$.

4.5.2.3 Intermediate (finite constrained) resequencing buffer

The resequencing buffers, located between two consecutive stations, contain a finite number of buffer places. Furthermore the individual buffer places have different physical sizes. The overall space necessary for the individual resequencing buffers is therefore reduced considerably.

This formulation is more complex than the previous ones due to the fact that it has to be determined whether a job passes to the next station without resequencing or if it has to be taken off the line and assigned to one of the available buffer places, considering at the same time that the buffer places may not be able to store a particular job due to job size restrictions (see figure 1.2).

The objective function is the makespan and maintains unchanged as in (4.1). Also constraints (4.2) to (4.4), the station precedence, and the job precedence are unchanged.

$$\text{Minimize } \ (s_{M,[N]} + P_{M,[N]})$$

subject to

$$s_{i,[j]} + P_{i,[j]} \leq s_{i+1,[j]} \qquad i = 1, ..., M-1; \forall j$$

$$\begin{aligned} &((s_{i,[j]} + P_{i,[j]} \leq s_{i,[k]}) \wedge (\Pi'_{i,[j]} < \Pi'_{i,[k]}))\vee \\ &\qquad \vee((s_{i,[k]} + P_{i,[k]} \leq s_{i,[j]}) \wedge (\Pi'_{i,[k]} < \Pi'_{i,[j]})) \qquad \forall i; \forall j,k|_{j\neq k} \end{aligned}$$

$$\Pi_{i,[k]} = j \ \rightarrow \ \Pi'_{i,j} = k \qquad \forall i, j, k$$

In order to determine whether a job requires to be taken off the production line, with the aim of reinserting it at a later point of time, the two constraints (4.6) and (4.7) were already introduced.

$\lambda_{i,[j],[k]}$ indicates if the job at position k succeeds the job at position j at station i as well as at station $i+1$. In this case $\lambda_{i,[j],[k]}$ is forced to 1.

$$\begin{aligned} \Pi'_{i+1,\Pi_{i,j}} < \Pi'_{i+1,\Pi_{i,k}} \ &\rightarrow \ \lambda_{i,[j],[k]} = 1 \qquad i = 1, ..., M-1; \forall j,k|_{j<k} \\ \Pi'_{i+1,\Pi_{i,j}} \geq \Pi'_{i+1,\Pi_{i,k}} \ &\rightarrow \ \lambda_{i,[j],[k]} = 0 \qquad i = 1, ..., M-1; \forall j,k|_{j<k} \\ &\quad\ \ \lambda_{i,[j],[k]} = 0 \qquad i = 1, ..., M-1; \forall j,k|_{j\geq k} \end{aligned}$$

Then, $\Lambda_{i,[j]}$ determines if the number of jobs succeeding the job at position j at station i and at station $i+1$ is not the same as the number of jobs which succeed the job at position j at station i. In this case $\Lambda_{i,[j]}$ is 1, indicating that job j occupies a buffer place of the resequencing buffer after station i.

$$\sum_{k=1}^{N} \lambda_{i,[j],[k]} \neq N - j \quad \rightarrow \quad \Lambda_{i,[j]} = 1 \qquad i = 1, ..., M-1; \forall j$$
$$\sum_{k=1}^{N} \lambda_{i,[j],[k]} = N - j \quad \rightarrow \quad \Lambda_{i,[j]} = 0$$

Once the jobs have been determined which are to be taken off the line for the purpose of resequencing, it is necessary to ensure that the available resequencing buffer place can only be loaded with one job at a time. Furthermore, the assigned job can not exceed the size of the buffer place. Constraints (4.14) to (4.18) are used for this purpose.

$\Delta_{i,[j],[k]}$ is 1 in the case in which the job at position k after station i has been temporarily stored in the respective resequencing buffer at the point of time the job at position j is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in a buffer.

$$\begin{aligned} &(\Lambda_{i,[j]} = 1) \wedge (\Lambda_{i,[k]} = 1) \wedge \\ &\wedge (c_{i,[j]} \geq c_{i,[k]}) \wedge (c_{i,[j]} < s_{i+1,[k]}) \quad \rightarrow \quad \Delta_{i,[j],[k]} = 1 \\ &\qquad i = 1, ..., M-1; \forall j, k|_{j \neq k} \end{aligned} \tag{4.14}$$

$$\begin{aligned} &(\Lambda_{i,[j]} = 0) \vee (\Lambda_{i,[k]} = 0) \vee \\ &\vee (c_{i,[j]} < c_{i,[k]}) \vee (c_{i,[j]} \leq s_{i+1,[k]}) \quad \rightarrow \quad \Delta_{i,[j],[k]} = 0 \\ &\qquad i = 1, ..., M-1; \forall j, k|_{j \neq k} \end{aligned}$$

$$\Delta_{i,[j],[k]} = 0 \qquad i = 1, ..., M-1; \forall j, k|_{j = k}$$

$\psi_{i,d,[j]}$ assigns the job at position j after leaving station i to the buffer-place d at the respective resequencing buffer if the size of the job ($\phi_{\Pi_{i,[j]}}$) does not exceed the size of the buffer place ($\Phi_{i,d}$).

$$0 \leq \psi_{i,d,[j]} \cdot (\Phi_{i,d} - \phi_{\Pi_{i,[j]}}) \qquad i = 1, ..., M-1; \forall d, j \tag{4.15}$$

In order to ensure that two jobs are not assigned to the same buffer place simultaneously, $\psi_{i,d,[j]}$ can not be set if the job at position k is already loaded to the buffer place d after station i, indicated by $\psi_{i,d,[k]}$ and $\Delta_{i,[j],[k]}$.

$$\Delta_{i,[j],[k]} = 1 \qquad \rightarrow \quad (\psi_{i,d,[j]} \neq \psi_{i,d,[k]}) \vee (\psi_{i,d,[j]} = 0) \qquad (4.16)$$
$$i = 1, ..., M-1; \forall d; \forall j,k|_{j \neq k}$$

Constraint (4.17) ensures that all jobs, which require resequencing, are assigned to a buffer-place at the respective buffer.

$$\sum_{d=1}^{D} \psi_{i,d,[j]} = \Lambda_{i,[j]} \qquad i = 1, ..., M-1; \forall d, j \qquad (4.17)$$

Finally, resuming the assignation of jobs to the resequencing buffers, the argument of $\Psi_{i,[j]}$ indicates the buffer place to which the job at position j is assigned after leaving station station i.

$$\Psi_{i,[j]} = \sum_{d=1}^{D} (d \cdot \psi_{i,d,[j]}) \qquad i = 1, ..., M-1; \forall d, j \qquad (4.18)$$

4.5.3 Centralized resequencing buffer

The centralized resequencing buffer is accessible from various stations. The benefit of the centralization lies in the reduction of buffer space. Clearly, no two jobs may occupy the same buffer space at the same time.

The objective function as well as the constraints for station precedence, job precedence and the determination whether a job requires to be taken off the line are the same as for the case of intermediate resequencing buffers, i.e., (4.1) to (4.4), (4.6) and (4.7). Due to the fact that the centralized buffer is accessible from various stations, the constraints for the buffer place assignation, (4.14) to (4.18), have to be altered and substituted with the constraints (4.19) to (4.23)

Δ' is 1 in the case in which the job at position k after leaving station h has been temporarily stored in the respective resequencing buffer at the point of time the job at position j after leaving station i is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in the buffer.

$$\begin{aligned}
&(\Lambda_{i,[j]} = 1) \wedge (\Lambda_{h,[k]} = 1) \wedge \\
&\wedge (c_{i,[j]} \geq c_{h,[k]}) \wedge (c_{i,[j]} < s_{h+1,[k]}) \quad \rightarrow \quad \Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]} = 1 \\
&\qquad i,h = 1,...,M-1|_{B'_i=1}; \forall j,k|_{j \neq k}
\end{aligned} \tag{4.19}$$

$$\begin{aligned}
&(\Lambda_{i,[j]} = 0) \vee (\Lambda_{h,[k]} = 0) \vee \\
&\vee (c_{i,[j]} < c_{h,[k]}) \vee (c_{i,[j]} \geq s_{h+1,[k]}) \quad \rightarrow \quad \Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]} = 0 \\
&\qquad i,h = 1,...,M-1|_{B'_i=1}; \forall j,k|_{j \neq k}
\end{aligned}$$

$$\begin{aligned}
\Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]} = 0 \qquad i,h = 1,...,M-1|_{B'_i=0}; \forall j,k|_{j \neq k} \\
\Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]} = 0 \qquad i,h = 1,...,M-1|_{B'_i=1}; \forall j,k|_{j = k}
\end{aligned}$$

$\psi'_{i,d,[j]}$ assigns the job at position j leaving station i to the buffer-place d of the resequencing buffer if the size of the job ($\phi_{\Pi_{i,[j]}}$) does not exceed the size of the buffer place (Φ'_d).

$$0 \leq \psi'_{i,d,[j]} \cdot (\Phi'_d - \phi_{\Pi_{i,[j]}}) \qquad i = 1,...,M-1; \forall d, j \tag{4.20}$$

$\psi'_{i,d,[j]}$ indicates if job j, leaving station i, is to be loaded to the buffer place d. In order to ensure that two jobs are not assigned to the same buffer place simultaneously, $\psi'_{i,d,[j]}$ can not be set if the job at position k, leaving station h, is already loaded to the buffer place d, indicated by $\psi'_{i,d,[k]}$ and $\Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]}$.

$$\begin{aligned}
\Delta'_{(i-1)\cdot N+[j],(h-1)\cdot N+[k]} = 1 \quad \rightarrow \quad (\psi'_{i,d,[j]} \neq \psi'_{h,d,[k]}) \vee (\psi'_{i,d,[j]} = 0) \\
i,h = 1,...,M-1; \forall d; \forall j,k|_{j \neq k}
\end{aligned} \tag{4.21}$$

Constraint (4.22) ensures that all jobs, which require resequencing, are assigned to a buffer-place.

$$\sum_{d=1}^{D} \psi'_{i,d,[j]} = \Lambda_{i,[j]} \qquad i = 1, ..., M-1; \forall d, j \tag{4.22}$$

Finally, resuming the assignation of jobs to the centralized resequencing buffer, the argument of $\Psi'_{i,[j]}$ indicates the buffer place to which the job at position j is assigned after leaving station station i.

$$\Psi'_{i,[j]} = \sum_{d=1}^{D} (d \cdot \psi'_{i,d,[j]}) \qquad i = 1, ..., M-1; \forall d, j \tag{4.23}$$

4.6 Computational enhancement

The formulation of the exact approach, presented in the previous sections, solves the problem of resequencing in flowshops to optimality. Due to the complexity of the problem and the resulting limitations of the size of the problem, with respect to the number of stations and jobs, enhancements regarding the computational effort is indispensable.

4.6.1 Imposing start time of jobs

As shown in figure 4.1, station i can nor start processing the job at position j ($s_{i,[j]}$) before the station has finished processing the job at the previous position $j-1$ ($c_{i,[j-1]}$), neither before the job at position j has completed on the previous station $i-1$ ($c_{i-1,[j]}$).

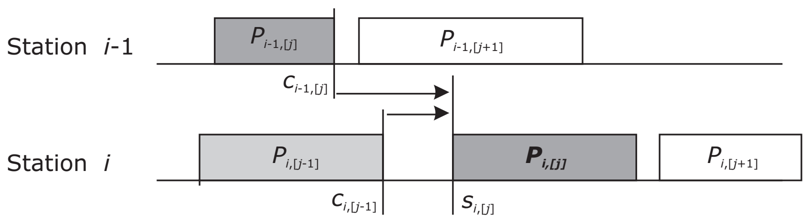


Figure 4.1: Constraints of the schedule calculation at station i.

Constraint (4.3) considers that these two restrictions have to be fulfilled, without enforcing one or the other. Considering resequencing and the corresponding constraints for the offline-buffers, a schedule may be possible which includes an additional idle time for a certain job because the respective buffer-place may not yet be freed.

A computational enhancement can be achieved by imposing the job precedence to exactly fulfill one of the two restrictions, without the possibility of exceeding it.

$$\begin{aligned}
s_{1,[1]} &= 0 && && (4.24)\\
s_{1,[j+1]} &= s_{1,[j]} + P_{1,[j]} && j = 1, ..., N-1 && (4.25)\\
s_{i+1,[1]} &= s_{i,[1]} + P_{i,[1]} && i = 1, ..., M-1 && (4.26)\\
s_{i+1,[j+1]} &= \max\{s_{i,[j+1]} + P_{i,[j+1]}, && i = 1, ..., M-1; && (4.27)\\
&\qquad s_{i+1,[j]} + P_{i+1,[j]}\} && j = 1, ..., N-1 &&
\end{aligned}$$

In order to be an optimal solution, job $J_{1,[1]}$ has to start at time 0 (4.24). Furthermore, a job at the first station is only restricted by the completion time of the prior job (4.25) and a job at the first position of the sequence, at all stations, is restricted only by its completion time at the prior station (4.26). For all other cases, constraint (4.27) is applicable.

In the case in which setup time exists, an additional time $ST_{i,\mu_{i,[j]},\mu_{i,[k]}}$ is considered when at station i a job of model type e precedes a job of model type f. In order to consider this setup time constrains 4.25 and 4.27 need to be altered to 4.29 and 4.31.

$$\begin{aligned}
s_{1,[1]} &= 0 && && (4.28)\\
s_{1,[j+1]} &= s_{1,[j]} + P_{1,[j]} + ST_{1,\mu_{1,[j]},\mu_{1,[j+1]}} && j = 1, ..., N-1 && (4.29)\\
s_{i+1,[1]} &= s_{i,[1]} + P_{i,[1]} && i = 1, ..., M-1 && (4.30)\\
s_{i+1,[j+1]} &= \max\{s_{i,[j+1]} + P_{i,[j+1]}, && i = 1, ..., M-1; && (4.31)\\
&\qquad s_{i+1,[j]} + P_{i+1,[j]} + ST_{i+1,\mu_{i+1,[j]},\mu_{i+1,[j+1]}}\} && j = 1, ..., N-1 &&
\end{aligned}$$

The introduction of constraints (4.24) to (4.27) and (4.28) to (4.31) results in an essential reduction of computational time and is furthermore utilized in the schedule calculation in the heuristic approach, which will be presented in chapter 6.

4.6.2 Reduction of variable size

One of the primary variables in the above described formulation is the variable Π_i, indicating the job sequence at station i. In the case in which station i does not have access to a resequencing buffer, the sequence Π_{i+1} is equal to the sequence Π_i. Hence, due to the fact that not all stations have access to a resequencing buffer, the job sequence for various consecutive stations is the same.

Instead of imposing that the sequences of two consecutive stations are identical, the variable can be reduced in size. It is only necessary to consider the stations which have access to a resequencing buffer. The index for the resulting job sequences is r and the total number of stations with access to a resequencing buffer is L, resulting in a maximum number of unique job sequences of $L + 1$.

This reduction furthermore influences considerably the size of other variables. It is no longer necessary to contemplate stations which do not have access to a resequencing buffer for variables which carry information with respect to the resequencing buffers.

The following additional parameters are introduced:

L	Number of resequencing possibilities	
r	Index of resequencing possibilities	$r = 1, ..., L$
Γ_i	Indicates which sequence is used at station i. In the case in which no resequencing buffer is provided, $\Gamma_i = 1$ for all i.	$\Gamma_i \in \{1, ..., L + 1\}$
Ω_r	Indicates which station has access to a resequencing buffer.	$\Omega_r \in \{1, ..., M\}$

The number of resequencing buffers, L, is determined as:

$$L = \sum_{i=1}^{M-1} B_i \tag{4.32}$$

Parameter Γ_i indicates which sequence is used at station i and is determined as:

$$\Gamma_i = 1 + \sum_{h=1}^{i-1} B_h \qquad \forall i \tag{4.33}$$

The parameter which indicates which station has access to a resequencing buffer, Ω_r, is determined as:

$$\Omega_r = i \mid_{\Gamma_i = r \ \wedge \ (\Gamma_{i+1} - \Gamma_i) = 1} \qquad \forall r; i = 2, ..., M \tag{4.34}$$

The proposal of only including few resequencing buffers, $(L << M)$, results in a substantial reduction, e.g., of constraint 4.4 to the following:

$$\Pi_{r,[k]} = j \ \rightarrow \ \Pi'_{r,j} = k \qquad r = 1, ..., L; \forall j, k \tag{4.35}$$

4.7 Final formulation

The *Final Formulation* includes all necessary constraints for the Constrained Logic Programming (CLP) to solve the problem of sequencing jobs in a flowshop with the possibility to resequence between consecutive stations. The buffers are located off the line, either accessible from a single station (intermediate case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size (see figure 1.2). Also included are the computational enhancements like imposing the start time of jobs (section 4.6.1) and the reduction of variable size (section 4.6.2).

First, the additional indices and parameter, and the redefinition of the variables for the *Final Formulation* are presented. Thereafter, the intermediate case is presented and then the centralized case.

4.7.1 Redefinitions

As described in the previous section, two indices, a parameter and various variables undergo a redefinition in order to reduce the variable size. This is caused by the fact that these variables depend on the number of resequencing possibilities, rather than on the number of stations.

Additional indices and parameters:

$$\mathbf{i}, \mathbf{h} \qquad \Rightarrow \qquad \mathbf{r}, \mathbf{t}$$

$$\mathbf{M} \qquad \Rightarrow \qquad \mathbf{L}$$

Job related variables:

$$\begin{aligned}
\Pi_{i,j} \quad &\Rightarrow \quad \Pi_{r,j} \\
\Pi'_{i,j} \quad &\Rightarrow \quad \Pi'_{r,j} \\
\lambda_{\mathbf{i},[j],[k]} \quad &\Rightarrow \quad \lambda_{\mathbf{r},[j],[k]} \\
\Lambda_{\mathbf{i},[j]} \quad &\Rightarrow \quad \Lambda_{\mathbf{r},[j]}
\end{aligned}$$

Variables used with **intermediate** resequencing buffers:

$$\begin{aligned}
\Delta_{\mathbf{i},[j],[k]} \quad &\Rightarrow \quad \Delta_{\mathbf{r},[j],[k]} \\
\psi_{\mathbf{i},d,[j]} \quad &\Rightarrow \quad \psi_{\mathbf{r},d,[j]} \\
\Psi_{\mathbf{i},[j]} \quad &\Rightarrow \quad \Psi_{\mathbf{r},[j]}
\end{aligned}$$

Variables used with **centralized** resequencing buffers:

$$\begin{aligned}
\Delta'_{(\mathbf{i}-1)\cdot N+[j],(\mathbf{h}-1)\cdot N+[k]} \quad &\Rightarrow \quad \Delta'_{(\mathbf{r}-1)\cdot N+[j],(\mathbf{t}-1)\cdot N+[k]} \\
\psi'_{\mathbf{i},d,[j]} \quad &\Rightarrow \quad \psi'_{\mathbf{r},d,[j]} \\
\Psi'_{\mathbf{i},[j]} \quad &\Rightarrow \quad \Psi'_{\mathbf{r},[j]}
\end{aligned}$$

4.7.2 Intermediate resequencing buffer

The resequencing buffers, located between two consecutive stations, contain a finite number of buffer places. These buffer places are differently sized and therefore may not be able to store a particular job due to job size restrictions.

$$\text{Minimize } \left(s_{M,[N]} + P_{M,[N]}\right) \tag{4.36}$$

subject to

$$\begin{aligned}
s_{1,[1]} &= 0 && && (4.37) \\
s_{i+1,[1]} &= s_{i,[1]} + P_{i,[1]} && i = 1, ..., M-1 && (4.38) \\
s_{1,[j+1]} &= s_{1,[j]} + P_{1,[j]} && j = 1, ..., N-1 && (4.39) \\
s_{i+1,[j+1]} &= \max\{s_{i,[j+1]} + P_{i,[j+1]}, && i = 1, ..., M-1; && (4.40) \\
&\qquad s_{i+1,[j]} + P_{i+1,[j]}\} && j = 1, ..., N-1 &&
\end{aligned}$$

$$((s_{i,[j]} + P_{i,[j]} \leq s_{i,[k]}) \wedge (\Pi^{'}_{\Gamma_i,[j]} < \Pi^{'}_{\Gamma_i,[k]}))\vee \tag{4.41}$$
$$\vee((s_{i,[k]} + P_{i,[k]} \leq s_{i,[j]}) \wedge (\Pi^{'}_{\Gamma_i,[k]} < \Pi^{'}_{\Gamma_i,[j]})) \qquad \forall i; \forall j,k|_{j \neq k}$$

$$\Pi_{r,[k]} = j \;\; \rightarrow \;\; \Pi^{'}_{r,j} = k \qquad r = 1...L+1; \forall j,k \tag{4.42}$$

In order to determine whether a job requires to be taken off the production line, with the aim of reinserting it at a later point of time, the two constraints (4.43) and (4.44) are used.

$\lambda_{r,[j],[k]}$ indicates if the job at position k succeeds the job at position j in the sequence Π_r as well as in sequence Π_{r+1}. In this case $\lambda_{r,[j],[k]}$ is forced to 1.

$$\begin{aligned} \Pi^{'}_{r+1,\Pi_{r,j}} < \Pi^{'}_{r+1,\Pi_{r,k}} \;\; &\rightarrow \;\; \lambda_{r,[j],[k]} = 1 \qquad \forall r; \forall j,k|_{j<k} \\ \Pi^{'}_{r+1,\Pi_{r,j}} \geq \Pi^{'}_{r+1,\Pi_{r,k}} \;\; &\rightarrow \;\; \lambda_{r,[j],[k]} = 0 \qquad \forall r; \forall j,k|_{j<k} \\ &\quad\;\;\; \lambda_{r,[j],[k]} = 0 \qquad \forall r; \forall j,k|_{j \geq k} \end{aligned} \tag{4.43}$$

Then, $\Lambda_{r,[j]}$ determines if the number of jobs succeeding the job at position j in sequence Π_r and in sequence Π_{r+1} is not the same as the number of jobs which succeed the job at position j in sequence Π_r. In this case $\Lambda_{r,[j]}$ is 1, indicating that job j occupies a buffer place of the resequencing buffer r.

$$\begin{aligned} \sum_{k=1}^{N} \lambda_{r,[j],[k]} \neq N - j \;\; &\rightarrow \;\; \Lambda_{r,[j]} = 1 \qquad \forall r, j \\ \sum_{k=1}^{N} \lambda_{r,[j],[k]} = N - j \;\; &\rightarrow \;\; \Lambda_{r,[j]} = 0 \end{aligned} \tag{4.44}$$

Once the jobs have been determined which are to be taken off the line for the purpose of resequencing, it is necessary to ensure that the available resequencing buffer place can only be loaded with one job at a time. Furthermore, the assigned job can not exceed the size of the buffer place. Constraints (4.45) to (4.49) are used for this purpose.

$\Delta_{r,[j],[k]}$ is 1 in the case in which the job at position k after sequence Π_r has been temporarily stored in the respective resequencing buffer at the point of time the job at position j is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in a buffer.

$$\begin{aligned} &(\Lambda_{r,[j]} = 1) \wedge (\Lambda_{r,[k]} = 1) \wedge \\ \wedge(c_{\Omega_r,[j]} \geq c_{\Omega_r,[k]}) \wedge &(c_{\Omega_r,[j]} < s_{\Omega_r+1,[k]}) \quad \rightarrow \quad \Delta_{r,[j],[k]} = 1 \\ & \qquad\qquad \forall r; \forall j,k|_{j\neq k} \end{aligned} \tag{4.45}$$

$$\begin{aligned} &(\Lambda_{r,[j]} = 0) \vee (\Lambda_{r,[k]} = 0) \vee \\ \vee(c_{\Omega_r,[j]} < c_{\Omega_r,[k]}) \vee &(c_{\Omega_r,[j]} \geq s_{\Omega_r+1,[k]}) \quad \rightarrow \quad \Delta_{r,[j],[k]} = 0 \\ & \qquad\qquad \forall r; \forall j,k|_{j\neq k} \end{aligned}$$

$$\Delta_{r,[j],[k]} = 0 \qquad \forall r; \forall j,k|_{j=k}$$

$\psi_{r,d,[j]}$ assigns the job at position j after sequence Π_r to the buffer-place d at the respective resequencing buffer if the size of the job ($\phi_{\Pi_{i,[j]}}$) does not exceed the size of the buffer place ($\Phi_{r,d}$).

$$0 \leq \psi_{r,d,[j]} \cdot (\Phi_{r,d} - \phi_{\Pi_{i,[j]}}) \qquad \forall r,d,j \tag{4.46}$$

In order to ensure that two jobs are not assigned to the same buffer place simultaneously, $\psi_{r,d,[j]}$ can not be set if the job at position k is already loaded to the buffer place d after sequence Π_r, indicated by $\psi_{r,d,[k]}$ and $\Delta_{r,[j],[k]}$.

$$\begin{aligned} \Delta_{r,[j],[k]} = 1 \quad \rightarrow \quad & (\psi_{r,d,[j]} \neq \psi_{r,d,[k]}) \vee (\psi_{r,d,[j]} = 0) \\ & \qquad\qquad \forall r,d; \forall j,k|_{j\neq k} \end{aligned} \tag{4.47}$$

Constraint (4.48) ensures that all jobs, which require resequencing, are assigned to a buffer-place at the respective buffer.

$$\sum_{d=1}^{D} \psi_{r,d,[j]} = \Lambda_{r,[j]} \qquad \forall r,d,j \tag{4.48}$$

Finally, resuming the assignation of jobs to the resequencing buffers, the argument of $\Psi_{r,[j]}$ indicates the buffer place to which the job at position j is assigned after Sequence Π_r.

$$\Psi_{r,[j]} = \sum_{d=1}^{D}(d \cdot \psi_{r,d,[j]}) \qquad \forall r, d, j \tag{4.49}$$

4.7.3 Centralized resequencing buffer

The centralized resequencing buffer is accessible from various stations. The benefit of the centralization lies in the reduction of buffer space. Clearly, no two jobs may occupy the same buffer space at the same time.

The objective function as well as the constraints for station precedence, job precedence and the determination whether a job requires to be taken off the line are the same as for the case of intermediate resequencing buffers, i.e., (4.37) to (4.44). Due to the fact that the centralized buffer is accessible from various stations, the constraints for the buffer place assignation, (4.45) to (4.49), have to be altered and substituted with the constraints (4.50) to (4.54)

Δ' is 1 in the case in which the job at position k at the t-th resequencing possibility has been temporarily stored in the respective resequencing buffer at the point of time the job at position j at the r-th resequencing possibility is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in the buffer.

$$\begin{aligned}
&(\Lambda_{r,[j]} = 1) \wedge (\Lambda_{t,[k]} = 1) \wedge \\
&\wedge (c_{\Omega_r,[j]} \geq c_{\Omega_t,[k]}) \wedge (c_{\Omega_r,[j]} < s_{\Omega_t+1,[k]}) \quad \rightarrow \quad \Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 1 \\
&\qquad\qquad \forall r, t; \forall j, k|_{j \neq k}
\end{aligned} \tag{4.50}$$

$$\begin{aligned}
&(\Lambda_{r,[j]} = 0) \vee (\Lambda_{t,[k]} = 0) \vee \\
&\vee (c_{\Omega_r,[j]} < c_{\Omega_t,[k]}) \vee (c_{\Omega_r,[j]} \geq s_{\Omega_t+1,[k]}) \quad \rightarrow \quad \Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 0 \\
&\qquad\qquad \forall r, t; \forall j, k|_{j \neq k}
\end{aligned}$$

$$\Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 0 \qquad \forall r, t; \forall j, k|_{j = k}$$

$\psi'_{r,d,[j]}$ assigns the job at position j at the r-th resequencing possibility to the buffer-place d of the resequencing buffer if the size of the job ($\phi_{\Pi_{i,[j]}}$) does not exceed the size of the buffer place (Φ'_d).

$$0 \leq \psi'_{r,d,[j]} \cdot (\Phi'_d - \phi_{\Pi_{i,[j]}}) \qquad \forall r, d, j \tag{4.51}$$

$\psi'_{r,d,[j]}$ indicates if job j, at the r-th resequencing possibility, is to be loaded to the buffer place d. In order to ensure that two jobs are not assigned to the same buffer place simultaneously, $\psi'_{r,d,[j]}$ can not be set if the job at position k at the t-th resequencing possibility is already loaded to the buffer place d, indicated by $\psi'_{r,d,[k]}$ and $\Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]}$.

$$\begin{aligned}\Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 1 \qquad &\rightarrow \quad (\psi'_{r,d,[j]} \neq \psi'_{t,d,[k]}) \vee (\psi'_{r,d,[j]} = 0) \\ &\forall r, t, d; \forall j, k|_{j \neq k}\end{aligned} \tag{4.52}$$

Constraint (4.53) ensures that all jobs, which require resequencing, are assigned to a buffer-place.

$$\sum_{d=1}^{D} \psi'_{r,d,[j]} = \Lambda_{r,[j]} \qquad \forall r, d, j \tag{4.53}$$

Finally, resuming the assignation of jobs to the centralized resequencing buffer, the argument of $\Psi'_{r,[j]}$ indicates the buffer place to which the job at position j is assigned at the r-th resequencing possibility.

$$\Psi'_{r,[j]} = \sum_{d=1}^{D} (d \cdot \psi'_{r,d,[j]}) \qquad \forall r, d, j \tag{4.54}$$

In the case in which setup cost exists, the objective function (4.36) has to be changed to (4.8). In the case in which setup time exists, the objective function, has to include the part for the setup time which can be found in equation (4.9). Furthermore, the equations (4.37) to (4.40) have to be altered to equations (4.28) to (4.31) and equation (4.41) to equation (4.10).

4.8 Conclusions

This chapter presented the formulation of Constrained Logic Programming (CLP), by introducing the most simplest case, the non-permutation flowshop. In order to present the final formulation, the basic arrangement was gradually adapted in order to solve the problem under study, for both, the intermediate and the centralized case. The final formulation furthermore includes computational enhancements which is to impose the start time of the jobs and to reduce the variable size. The performance-study of the CLP formulation is accomplished in the next chapter.

5 Experimentation CLP

The experimentation attempts to show the viability of resequencing with constrained buffers. First, the basic arrangement of the CLP model is presented which solves the problem to optimality. Then, an analysis on semi dynamic demand is performed, necessary to justify the following alternative arrangements. The succeeding experimentations take into account alternatives to the pure non-permutation model, based on the assumption that a good permutation sequence tends to result in a good non-permutation sequence, when in the latter case the permutation sequence is used as a fixed job sequence for the first station. All of the experiments were run on a 3.00 GHz Pentium 4 with 512 MB of Random Access Memory (RAM).

5.1 Basic arrangement

Figure 5.1 shows the schematic for applying the non-permutation model directly to the problem under study. Unless the time for the execution of this arrangement is restricted, it solves the problem to optimality. However, due to the complexity of the problem, the execution time is limited to t_{max}, and if the optimal solution is not found within this time, the best solution found until that instance is used for comparison.

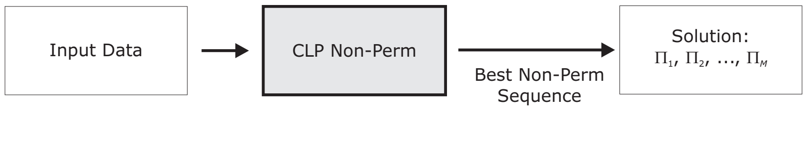


Figure 5.1: The Basic arrangement of the non-permutation model solves the problem to optimality, considering non-permutation sequences.

The input data contain all information about the station environment, like the number of stations, buffer locations, etc. and about the job processing environment, as e.g. processing times, setup costs and setup times.

In order to show the limitation of the basic arrangement, an example of a 10-station flowshop is introduced. Two intermediate resequencing buffers are used, accessible after station 2 and station 5, each with one resequencing buffer place. The range of the production time is [1..50], the setup cost is generated in the range [2..8] and the setup time in [1..5]. The objective function is the weighted sum of the makespan with a factor of 1.0, and the setup cost with factor of 0.3. Note that the setup time is indirectly included in the calculation of the makespan and its weight (δ) is set to zero.

	Station									
Job	1	2	3	4	5	6	7	8	9	10
1	19	7	5	4	26	26	9	31	25	15
2	50	44	14	34	36	34	22	23	33	47
3	47	28	28	7	2	25	41	9	36	42
4	32	40	17	44	49	42	21	36	17	15

Table 5.1: Processing time for 4 jobs on a 10 station flowshop.

Table 5.1 shows the processing times and the setup costs and setup times occurring at each station can be found in the appendix in table A.1 and A.2. The value of the objective function is 475 (makespan 436, setup cost 130, setup time 96). The optimal sequence is (2, 3, 4, 1) for the first part, (2, 3, 1, 4) for the second part and (2, 3, 1, 4) for the last part of the line, resulting job 4 being temporarily removed from the line at the first resequencing possibility. In order to solve the problem and prove optimality, a total time of 4314 sec was necessary for the basic arrangement.

5.2 Alternative arrangements (Hybrid Approaches)

On account of the elevated time spent on the calculation of a 10 station flowshop with two resequencing possibilities and only 4 jobs, the necessity rises to seek for alternatives to the basic arrangement. In this section, first an analysis on semi dynamic demand is performed, described in figure 2.2. Based on this, several alternative arrangements are proposed to approach the more complex case with a static demand. These alternative arrangements are hybrids of the exact approach with heuristic concepts and therefore do not ensure optimality.

5.2.1 Analysis on semi dynamic demand

This analysis studies the correlation between a good permutation sequence and the corresponding solution with resequencing after the first station. In this case, a good permutation sequence is promising and can give an excellent starting point for good non-permutation sequences. For the analysis a single instance is considered and solved with 50 randomly generated permutation sequences (see figure 5.2).

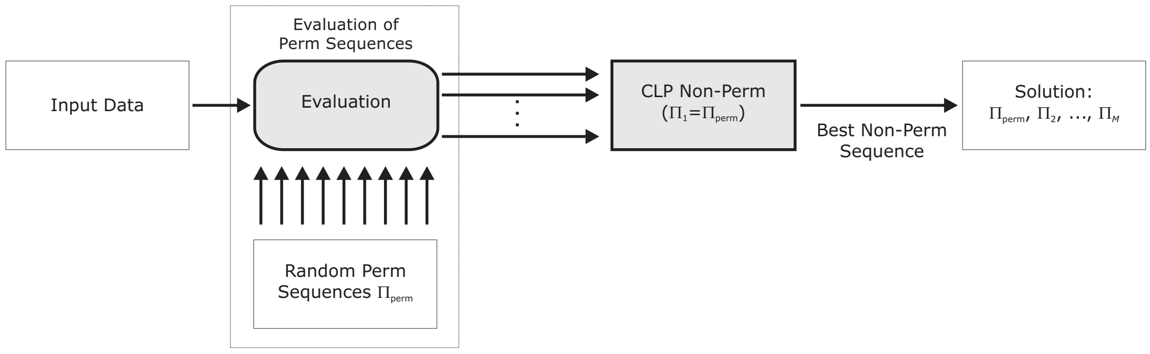

Input Data
Evaluation of Perm Sequences
Evaluation
Random Perm Sequences Π_{perm}
CLP Non-Perm (Π_1=Π_{perm})
Best Non-Perm Sequence
Solution: Π_{perm}, Π_2, ..., Π_M

Figure 5.2: Randomly generated permutation sequences are evaluated with the fully constrained permutation model. The sequences are then passed to the non-permutation model, pretending a semi dynamic demand with fixed job sequence for the first station ($\Pi_1 = \Pi_{perm}$).

The considered flowshop consists of 5 stations with access to a resequencing buffer with two buffer places after the third station. The range of the production time is [1..20], the setup cost is generated in the range [2..8] and the setup time in [1..5]. The objective function is the weighted sum of the makespan with a factor of 1.0, and the setup cost with a factor of 0.3. Note that the setup time is indirectly included in the calculation of the makespan and its weight (δ) is set to zero.

	Perm	Non-Perm						Job Changes					
Nr.		5s	10s	30s	60s	120s	300s	5s	10s	30s	60s	120s	300s
20	259,5	245,9	245,9	245,9	245,9	245,9	245,9	3	3	3	3	3	3
21	260,3	**260,3**	**260,3**	**260,3**	**260,3**	**260,3**	**260,3**	1	1	1	1	1	1
22	261,2	249,9	249,9	249,9	249,9	249,9	249,9	3	3	3	3	3	3
23	263,2	256,0	256,0	252,2	252,2	252,2	252,2	2	2	4	4	4	4
24	264,5	**264,8**	**264,8**	263,6	262,4	262,4	262,4	1	1	1	2	2	2
25	264,9	256,6	256,6	249,6	249,6	247,7	247,7	3	3	3	3	3	3
26	265,4	264,8	264,8	263,0	263,0	263,0	262,3	2	2	4	4	4	4

Table 5.2: Representative sample data for seven randomly generated permutation sequences. The grey cells show an improvement with respect to the same case in the previous column. Cells with bold text show that the permutation case is not improved.

Table 5.2 shows the solutions of a small group of seven representative data sets. The second column is the result for the permutation case, the third to the eighth column show the result for the semi dynamic case with the same permutation sequences applied for the first station, considering 5, 10, .., 300 seconds for the maximal execution time. The last 6 columns show the respective number of job changes, which tend to be larger for better solutions.

The complete data set of 50 instances can be found in the appendix in table A.3 and figure A.1, ordered by the results of the permutation case. Better permutation sequences tend to result in better non-permutation sequences but the best found permutation sequence does not necessarily result in the best non-permutation sequence. Furthermore, due to the time restriction, the non-permutation model does not necessarily give an improved solution compared to the permutation model (e.g. case 21 and 24).

Based on the analysis on semi dynamic demand, the experimental study considers two alternative arrangements, *Cascading* and *Multistart Cascading*, which do not guarantee optimality. In order to be able to compare the results, the same maximal time limitation, t_{max}, is used for the three cases.

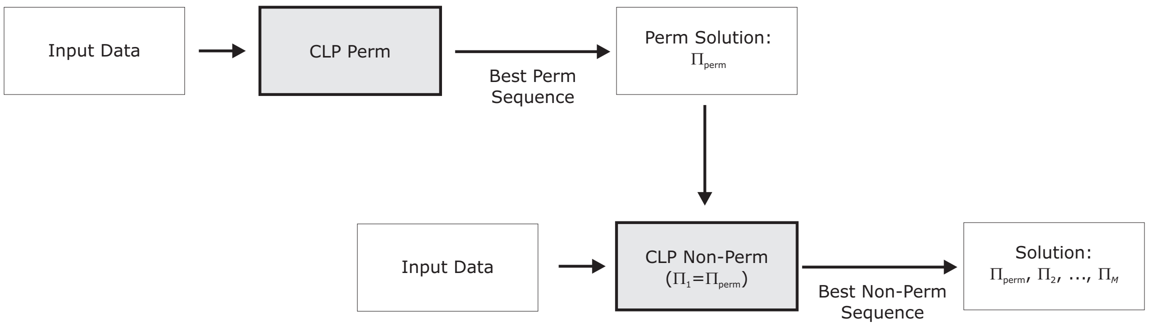

Input Data
CLP Perm
Best Perm Sequence
Perm Solution: Π_{perm}
Input Data
CLP Non-Perm ($\Pi_1=\Pi_{perm}$)
Best Non-Perm Sequence
Solution: $\Pi_{perm}, \Pi_2, ..., \Pi_M$

Figure 5.3: The first cascade considers only permutation sequences. The second cascade uses the permutation solution of the first cascade as the solution for the first station and solves the problem as if it would be a semi dynamic case with fixed job sequence for the first station.

5.2.2 Cascading

Figure 5.3 shows the schematic for the arrangement of applying the permutation model in a first cascade and then the non-permutation model. The intention of the first cascade is to determine the optimum permutation sequence. Then, in the second step and with the use of the non-permutation model, the additional constraint is introduced which uses a semi dynamic demand with fixed job sequence for the first station to the one previously determined by the first cascade. The sum of the maximal execution time for the two cascades is set to t_{max}.

5.2.3 Multistart cascading

As indicated by its name, the multistart arrangement consists of solving the problem several times, where the execution time is limited to a fraction of t_{max}. Every time a different sequence is fed to the first cascade (evaluation with the completely constrained permutation model) only its corresponding value of the objective function is determined. This operation requires a very small amount of time, compared to the second cascade which then determines the improvement, taking into account resequencing after the first station. The sequence of the first station is fixed to the one determined by the first cascade. As a consequence, and in contrast to pure cascading (section 5.2.2), the multistart arrangement is able to consider permutation sequences that are not optimal within the set of permutation sequences, but which may result in a relatively better final solution after applying the second cascade. The multistart arrangement is applied in two conceptional different ways:

- *Without Feedback:* The permutation sequences are generated in a random order and therefore are not subject to any guidance.
- *With Feedback:* The result of the second cascade is fed back to the first cascade. This scheme provides information at the time the permutation sequences are generated.

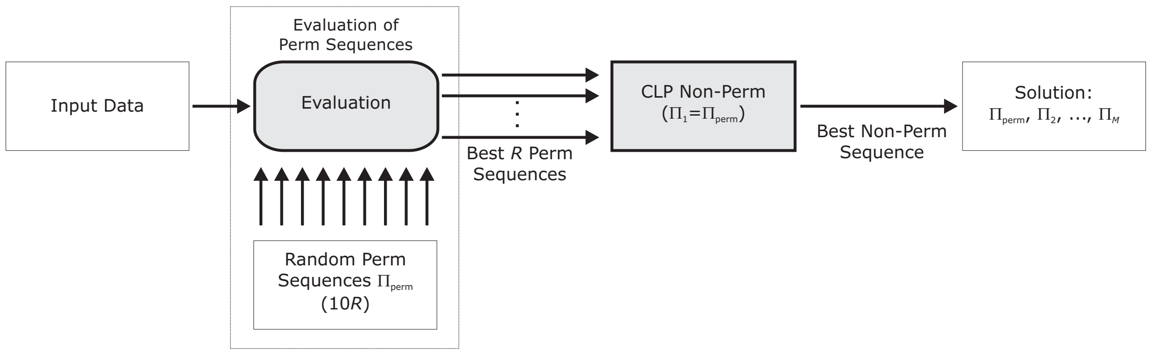

Input Data
Evaluation of Perm Sequences
Evaluation
Random Perm Sequences Π_{perm} (10R)
Best R Perm Sequences
CLP Non-Perm (Π_1=Π_{perm})
Best Non-Perm Sequence
Solution: Π_{perm}, Π_2, ..., Π_M

Figure 5.4: The CLP works in a multistart cascading mode without feedback, a total of 10 times R randomly generated permutation sequences are used.

Without Feedback

In this case, a certain number of randomly generated permutation sequences is evaluated with the completely constrained permutation model, summarized, and the most promising R sequences are passed to the non-permutation model pretending a semi dynamic demand with fixed job sequence for the first station ($\Pi_1 = \Pi_{perm}$). The arrangement is shown in figure 5.4 with a total of 10 times R randomly generated permutation sequences.

With Feedback

In this case, the execution is performed in an iterative way, while the number of randomly generated permutation sequences is constantly reduced (from R to 1), whereas the information obtained by the non-permutation model is fed back in an increasing way (from 0 to $R-1$) in order to preserve valid information. Three different arrangements are studied.

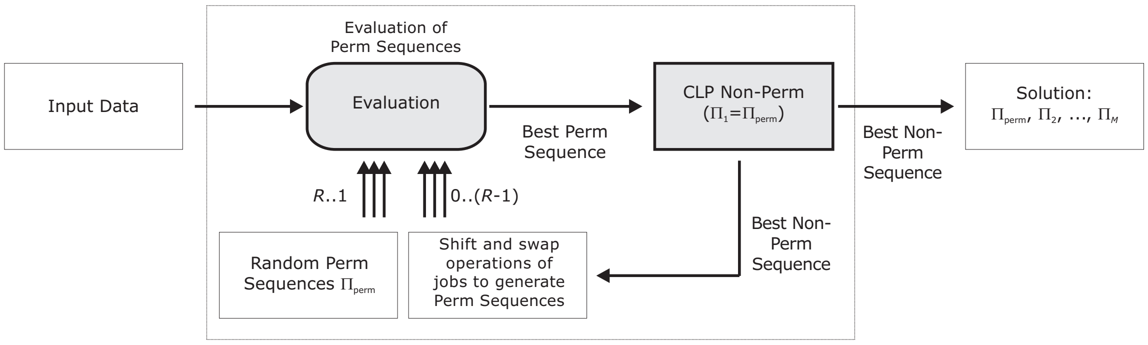

Input Data
Evaluation of Perm Sequences
Evaluation
Best Perm Sequence
CLP Non-Perm (Π_1=Π_{perm})
Best Non-Perm Sequence
Solution: Π_{perm}, Π_2, ..., Π_M
R..1
0..(R-1)
Best Non-Perm Sequence
Random Perm Sequences Π_{perm}
Shift and swap operations of jobs to generate Perm Sequences

Figure 5.5: The CLP works in a multistart cascading mode with feedback of the sequence which obtained the best solution in the second cascade.

Shift and swap operations of jobs: Figure 5.5 shows the schematic of the first arrangement. At the beginning, the permutation sequences used by the first cascade contain only randomly generated sequences. With each iteration, the number of sequences which are fed back from the second cascade by applying shift and swap operations of jobs, is increased.

Probability for position of jobs: Figure 5.6 shows the modification of the previous arrangement. In order to further enhance the optimization, an additional module is inserted in the feedback loop which determines the probability $p_{j,n}^a$ indicating that job j is at position n in the permutation sequence at iteration a. The probability for each job is updated after each iteration a and is then used in an increasing way for the generation of the permutation sequences of the first cascade. The maximum number of iterations is A.

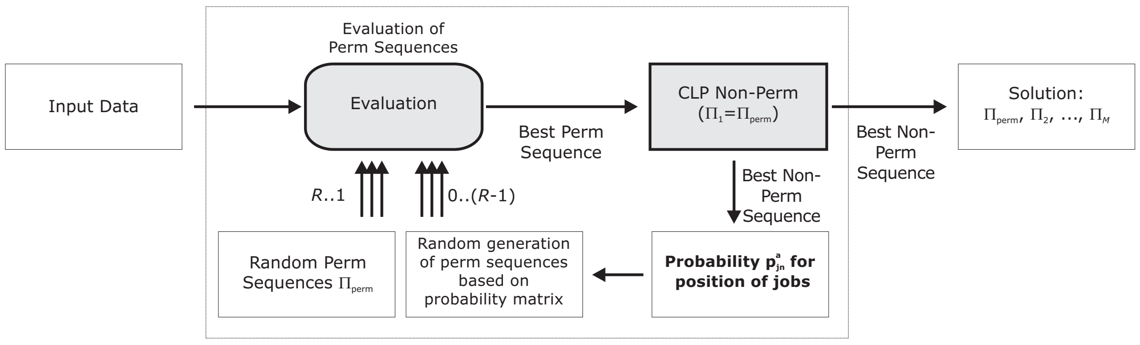

Evaluation of Perm Sequences
Input Data
Evaluation
Best Perm Sequence
CLP Non-Perm ($\Pi_1=\Pi_{perm}$)
Best Non-Perm Sequence
Solution: Π_{perm}, Π_2, ..., Π_M
R..1
0..(R-1)
Best Non-Perm Sequence
Random Perm Sequences Π_{perm}
Random generation of perm sequences based on probability matrix
Probability p^a_{jn} for position of jobs

Figure 5.6: The arrangement uses the probability $p_{j,n}^a$ which indicates that job j is at position n in order to feed back information to the first cascade. The parameter a indicates the iteration.

The pseudocode for the determination of a permutation sequence with the use of the probability $p_{j,n}^a$ can be formulated as follows:

- Calculation of the initial probability $p_{j,n}^0$ for all jobs (this is performed only once before the first iteration).
- Calculation of the iterative probability $p_{j,n}^a$.
    1. Update of W_n, indicating if a job is already assigned to position n, at the first station.
    2. Calculation of the cumulative probability $q_n{}^a$ for all jobs which are not already assigned, indicated by W_n.
    3. Normalization of the cumulative probability $q_n{}^a$.
    4. Selection of a job.
- Repeat steps 1-4 until all jobs are assigned.

Calculation of the initial probability $p^0_{j,n}$ for all jobs: Initially, before the first iteration when $a = 0$, the probability for each job is the same and is determined as:

$$p^0_{j,n} \quad = \quad \frac{1}{N} \qquad\qquad \forall j, n \tag{5.1}$$

Calculation of the iterative probability $p^a_{j,n}$: For each iteration the probability is reduced by the fraction $1/(N \cdot A)$ when job j is not at position n at the first station (5.2) and is augmented by $(N - 1)$-times the same fraction otherwise (5.3). The probability for each iteration a is calculated as:

$$p^a_{j,n} \quad = \quad p^{a-1}_{j,n} + \frac{-1}{N \cdot A} \qquad \text{if} \quad \Pi'_{1,j} \neq n \qquad a = 1..A; \forall j, n \tag{5.2}$$

$$p^a_{j,n} \quad = \quad p^{a-1}_{j,n} + \frac{N-1}{N \cdot A} \qquad \text{if} \quad \Pi'_{1,j} = n \qquad a = 1..A; \forall j, n \tag{5.3}$$

In this way the sum of probabilities $\sum_{n=1}^{N} p^a_{j,n}$ for a single job j for each iteration is always 1. Also, if a job never appears at a certain position its probability for that position after A iterations is 0, on the other hand, if a job always appears at the same position its probability for that position after A iterations is 1.

Update of W_n, indicating if a job is already assigned to position n, at the first station: As part of the selection process, initially the variable W_n is 0 for all positions n. Every time a job gets assigned to a position n, the binary variable W_n is set to 1, otherwise its value is maintained 0.

$$W_n \quad = \quad 1 \qquad \text{if} \quad \Pi'_{1,j} = n \qquad \forall j, n \tag{5.4}$$

Calculation of the cumulative probability $q^a_{j,n}$: The cumulative probability is the sum of the probability for a single job j from the first to the n-th position, if no job is assigned to position n, indicated by W_n. The cumulative probability is calculated as:

$$q_{j,n}^{a} = \sum_{n'=1}^{n} p_{j,n'}^{a}|_{W_n=0} \qquad \forall a, j, n \tag{5.5}$$

Normalization of the cumulative probability $q_{j,n}^{a}$: In the case in which at least one job is already assigned to the sequence, the calculation for the last position $q_{j,N}^{a}$ does not reach 1. The normalization ensures that the cumulative probability is spread to a value of 1 and is calculated as:

$$q_{j,n}^{a} = \frac{q_{j,n}^{a}}{q_{j,N}^{a}} \qquad \forall a, j, n \tag{5.6}$$

Selection of a job: In order to select a job for a position, based on its probability, a random number is generated and compared with the cumulative probability in the following way:

- For position n a random (float) number r is generated in the range [0..1].
- If $r < q_{1,n}^{a}$ then the first job is selected; otherwise the j-th job ($2 \leq j \leq N$) is selected such that $q_{j-1,n}^{a} < r \leq q_{j,n}^{a}$.

Probability for succeeding jobs: When setup time and setup cost are involved, it is important to determine the succeeding job for a certain job, rather than the position of a job itself. Figure 5.7 shows the modification of the previous arrangement. The variable $p_{j,k}^{'a}$ determines the probability that job k succeeds job j for the sequence of the first station.

In this case the determined sequences can be very different in terms of job position but will be similar in terms of succeeding jobs, caused by the fact that the first job is randomly selected and the succeeding jobs are determined considering their probability of being successor to the previously selected job.

Compared to the previous arrangement, using the probability for the position of the jobs, this arrangement uses the same proceeding as before in order to determine a permutation sequence, except for the calculation of the probability $p_{j,k}^{'a}$ (5.2 and 5.3), and the selection of jobs:

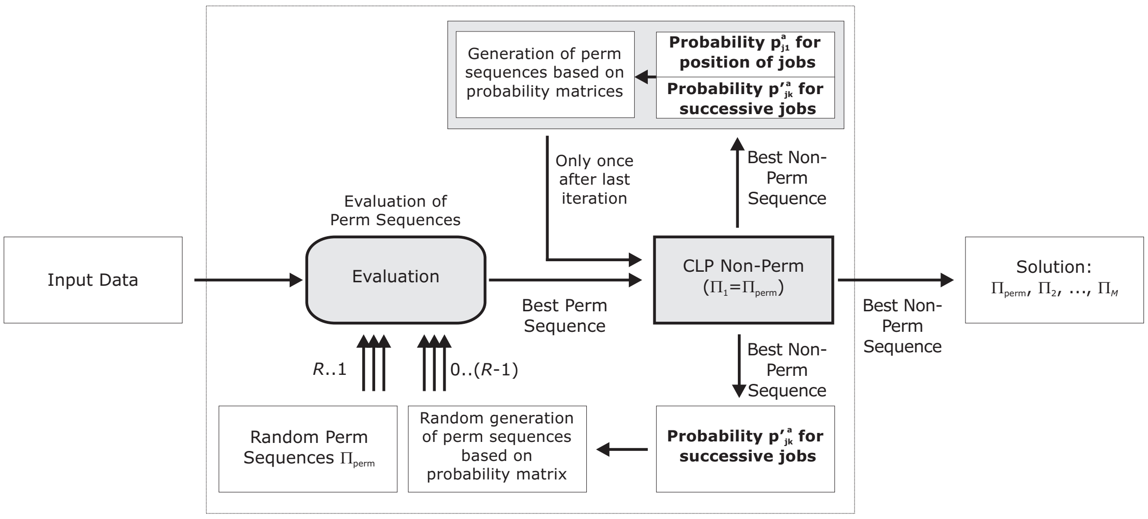

Generation of perm sequences based on probability matrices
Probability p_{j1}^{a} for position of jobs
Probability p'^{a}_{jk} for successive jobs
Only once after last iteration
Best Non-Perm Sequence
Evaluation of Perm Sequences
Input Data
Evaluation
CLP Non-Perm ($\Pi_1=\Pi_{perm}$)
Solution: Π_{perm}, Π_2, ..., Π_M
Best Perm Sequence
Best Non-Perm Sequence
Best Non-Perm Sequence
R..1
0..(R-1)
Random Perm Sequences Π_{perm}
Random generation of perm sequences based on probability matrix
Probability p'^{a}_{jk} for successive jobs

Figure 5.7: The arrangement uses the probability $p'^{a}_{j,k}$ which indicates that job k succeeds job j in order to feed back information to the first cascade. After the last iteration an additional sequence is determined using also the probability for the first job.

Calculation of the iterative probability $p'^{a}_{j,n}$: For each iteration the probability is reduced by the fraction $1/(N \cdot A)$ when the succeeding job $j+1$ is not job k (5.7) and is augmented by $(N-1)$-times the same fraction otherwise (5.9). The probability for each iteration a is calculated as:

$$p'^{a}_{j,k} = p'^{a-1}_{j,k} + \frac{-1}{N \cdot A} \quad \text{if} \quad \Pi_{1,j+1} \neq k \qquad a = 1..A; \forall k \quad j = 1..N-1 \tag{5.7}$$

$$p'^{a}_{j,k} = p'^{a-1}_{j,k} + \frac{N-1}{N \cdot A} \quad \text{if} \quad \Pi_{1,j+1} = k \qquad a = 1..A; \forall k \quad j = 1..N-1 \tag{5.8}$$

Selection of a job: The first job of a sequence is selected randomly. Thereafter, in order to select a job for a position, based on its probability a random number is generated and compared with the cumulative probability $q'^{a}_{j,k}$ in the following way:

- For position k a random (float) number r is generated in the range [0..1].
- If $r < q'^{a}_{1,k}$ then the first job is selected; otherwise the j-th job ($2 \leq j \leq N$) is selected such that $q'^{a}_{j-1,k} < r \leq q'^{a}_{j,k}$.

Finally, after the last iteration, an additional sequence is sent directly to the second cascade. The first job is determined with the use of the first column of the probability matrix for the position of the first job, $p_{j,1}^{a}$. Then, the succeeding jobs are determined as before using the probability matrix for succeeding jobs $p'^{a}_{j,k}$.

5.3 Mnemonics

In what follows, the different arrangements are used in a comparative way in the experimentation. Table 5.3 shows the mnemonic abbreviations and the relevant parameters used.

Mnemonic	Time	Description of arrangement
A_P	t_P	Permutation arrangement
A_{NP}	t_{NP}	Basic non-permutation arrangement
A_C	t_{CP}, t_{CNP}	Cascading
A_M	t_M	Multistart cascading without feedback
A_{MSS}	t_{MSS}	Multistart cascading with feedback and shift and swap operations
A_{MJP}	t_{MJP}	Multistart cascading with feedback and probability for position of jobs
A_{MSJ}	t_{MSJ}	Multistart cascading with feedback and probability for succeeding jobs

Table 5.3: Mnemonic abbreviations and the relevant parameters for the execution time.

5.4 Adjustment of parameters and selection of arrangement

The main parameter of the optimization is the maximum execution time t_{max}. The time necessary for the evaluation of a permutation sequence is in the range of tens of a second and therefore negligible. In order to be able to compare the performance of the different arrangements, the time t_{max} is set as follows:

$$t_{max} = 600\,\text{sec}$$

In the case of the first two arrangements, A_P and A_{NP}, the execution times are equal to the maximal execution time:

$$t_P = t_{NP} = t_{max} = 600\,\text{sec}$$

Then, for the arrangement A_C, different fractions for the execution times for each cascade were analyzed. Better results were achieved for the cases in which t_{CP} and t_{CNP} are set to $\frac{1}{3}$ and $\frac{2}{3}$ of t_{max}, respectively. The fraction $\frac{1}{3}$ is used for the permutation case and the fraction $\frac{2}{3}$ for the non-permutation case, justified by the fact that the second cascade is the far more complex case without overlooking that the final result is linked to a good permutation sequence.

$$\begin{aligned} t_{CP} &= 1/3 \cdot t_{max} &= 200\,\text{sec} \\ t_{CNP} &= 2/3 \cdot t_{max} &= 400\,\text{sec} \end{aligned}$$

For the arrangements which use iterative proceedings, the maximal execution time is related to the number of iterations. For the arrangements A_{MSS}, A_{MJP} and A_{MSJ} a flowshop which consists of 10 stations is studied by applying different numbers of iterations, connected to different fractions of time for each iteration. After station 3, 5 and 8 a single intermediate buffer place is located. The range of the production time is [1...100], for the setup cost [2...8] and for the setup time [1...5]. The number of jobs is 6 and the objective function is the weighted sum of the makespan (factor 1.0) and the setup cost (factor 0.3), where the setup time is indirectly included in the calculation of the makespan and its weight (δ) is set to zero. The results obtained by the arrangements A_P, A_{NP} and A_C are 979.7, 979.7 and 974.9, respectively.

Iterations	R	time/It	t_{max}	A_{MSS}	A_{MJP}	A_{MSJ}	Average
20	10	30	600	977,9	982,6	**979,8**	980,1
20	5	30	600	966,5	982,2	989,2	979,3
10	10	60	600	972,8	978,4	985,9	979,0
10	5	60	600	**966,3**	**977,4**	982,7	975,4
5	10	120	600	978,4	992,4	1001,4	990,7

Table 5.4: Different sets of parameters used for the adjustment of the arrangements A_{MSS}, A_{MJP} and A_{MSJ}.

The different sets of parameters together with the obtained results are shown in table 5.4. The results show the average value from five repetitions, each with a different seed for the generation of the random permutation sequences. The best solution for each arrangement is highlighted using a bold font. Then, the last column shows the average of the three arrangements under study. The cell with the best average value is highlighted with a grey background (975.4). The detailed results for the five repetitions are shown in the appendix in table A.4.

Perm Sequ	R	time/It	t_{max}	A_M
100	20	30	600	975,4
100	10	60	600	**973,1**
100	5	120	600	988,4
50	20	30	600	985,2
50	10	60	600	982,9
50	5	120	600	993,3

Table 5.5: Different sets of parameters used for the adjustment of the arrangement A_M.

Finally, the parameters for the arrangement A_M are studied, considering similar numbers in terms of number of sequences for the first and for the second cascade, as in the previous arrangements. The number of randomly generated permutation sequences is studied for 50 and for 100. The number of the best permutation sequences which are subsequently passed to the second cascade is 20, 10 and 5. Accordingly, the execution time for the second cascade is set to 30, 60 and 120 seconds, respectively. Table 5.5 shows the average value from five repetitions, each with a different seed for the generation of the random permutation sequences. The detailed results for the five repetitions are shown in the appendix in table A.5. Table 5.6 shows the definite values for the parameters.

Arrangement	Perm Sequ	Iterations	R	Time (sec)	t_{max} (sec)
A_P	-	-	-	$t_P = 600$	600
A_{NP}	-	-	-	$t_{NP} = 600$	600
A_C	-	-	-	$t_{CP} = 200$	600
				$t_{CNP} = 400$	
A_M	100	-	10	$t_M = 60$	600
A_{MSS}	-	10	5	$t_{MSS} = 60$	600
A_{MJP}	-	10	5	$t_{MJP} = 60$	600
A_{MSJ}	-	10	5	$t_{MSJ} = 60$	600

Table 5.6: Final set of parameters for the different arrangements.

In order to select the arrangement which performs best on the proposed problem, the same 10 station flowshop as before is used with different numbers of jobs in the range of 4 to 8.

Jobs	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
4	481,2	481,2	480,0	478,2	477,1	477,5	477,5	504,3	**476,9**
5	470,5	470,5	470,5	459,0	**454,8**	467,8	468,4	489,6	464,2
6	564,6	564,6	563,4	556,7	**552,9**	557,7	557,1	623,0	557,1
7	568,7	568,7	**567,8**	578,5	572,5	575,9	585,7	650,8	585,7
8	589,6	622,1	**589,6**	639,2	623,3	638,0	642,6	679,5	642,6

Table 5.7: Comparison of the proposed arrangements for the intermediate case.

The comparison of the previously proposed arrangements for the intermediate and the centralized case are shown in table 5.7 and 5.8, respectively. The results show the average value from five repetitions. The solutions which perform at least as good as the best found permutation sequence are highlighted with a grey background and the best solution of each line is highlighted using a bold font. The detailed results for the five repetitions are shown in the appendix in tables A.6 and A.7.

The column $A_{MSJ}(I)$ shows the result of the arrangement A_{MSS}, not taking into account the last tested sequence, using the probability for the first job position which is represented by $A_{MSJ}(II)$. Then, column A_{MSJ} shows the final result.

Jobs	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
4	481,2	481,2	481,2	478,2	**477,4**	478,0	478,0	504,5	477,7
5	470,5	470,5	470,5	459,0	**454,8**	467,8	468,4	489,6	464,2
6	564,6	564,6	564,0	556,7	**553,9**	558,6	554,5	614,1	554,5
7	568,7	568,7	**567,5**	572,6	567,6	574,6	577,3	604,6	577,3
8	589,6	589,6	**589,6**	625,8	618,4	622,0	625,0	666,4	625,0

Table 5.8: Comparison of arrangements for the centralized case.

It can be seen that in general all alternative arrangements achieve similar results or even outperform the best found permutation sequence (A_P) for less than 7 jobs. In particular, the arrangement A_{MSS} performs better and is therefore selected.

5.5 Limited flexibility

In the previous section the arrangement A_{MSS} showed improvements compared to the permutation case up to 7 jobs. It can furthermore be observed that, against expectation, in some cases the centralized case results in better solutions than the intermediate case. This can be explained by the fact that the centralized case is more restrictive than the intermediate case, which leads to the proposal of introducing limited flexibility as an additional restriction.

Limited flexibility is referred to the fact that the number of positions a job can move upwards or downwards is limited by the parameter W_{up} and W_{down}, respectively, and can be formulated with constraint (5.9).

$$(\Pi'_{r,j} - \Pi'_{r+1,j} \leq W_{up}) \quad \wedge \quad (\Pi'_{r+1,j} - \Pi'_{r,j} \leq W_{down}) \qquad \forall r, j \tag{5.9}$$

Four different cases are studied, including the case in which no limitation is applied.

1. $W_{up} = N - 1$ and $W_{down} = N - 1$ (No limitation)
2. $W_{up} \approx \frac{N}{4}$ and $W_{down} \approx \frac{N}{4}$
3. $W_{up} = 1$ and $W_{down} \approx \frac{N}{2}$
4. $W_{up} \approx \frac{N}{2}$ and $W_{down} = 1$

The first case does not give any restriction on the flexibility. The second case restricts the jobs towards both directions in the same way. The third and the fourth case restrict the flexibility of the jobs mainly towards one direction.

In what follows, the same flowshop as in the previous section is used and applied to the case of 7, 8, 9 and 10 jobs. The limitation on flexibility is set to the same values for the four cases. Due to the additional constraint the time for approaching promising solutions is reduced and therefore the time for each iteration, t_{MSS}, is reduced to 30 seconds and the number of iterations is consequently increased to 20.

		No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
Jobs	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
7	568,7	568,7	567,8	572,5	568,7	566,6	566,4	568,7	566,6	**565,8**	568,7	566,6	569,1
8	589,6	622,1	589,6	623,3	589,6	589,6	599,0	621,6	589,6	**589,1**	589,6	591,7	599,0
9	684,3	694,5	732,2	695,9	694,5	692,6	694,3	711,1	684,3	685,5	694,5	**684,0**	694,1
10	736,1	755,8	755,1	746,1	752,3	813,4	747,4	755,8	746,1	746,5	752,3	748,1	747,2

Table 5.9: Limited flexibility for the intermediate case.

Table 5.9 and 5.10 show the average value from five repetitions for the intermediate and the centralized case, respectively. The solutions which perform at least as good as the best found permutation sequence is highlighted with a grey background and the best solution of each line is highlighted using a bold font. The detailed results for the five repetitions are shown in the appendix in tables A.8 and A.9.

		No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
Jobs	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
7	568,7	568,7	567,5	567,6	568,7	567,5	**566,9**	572,7	567,5	567,0	568,7	567,5	567,1
8	589,6	589,6	589,6	618,4	589,6	589,6	588,5	621,6	589,6	589,4	621,6	589,6	**588,0**
9	684,3	694,5	682,5	691,5	694,5	683,4	688,9	711,1	**680,7**	684,8	694,5	682,5	689,7
10	736,1	752,3	735,5	743,6	752,3	735,5	743,7	755,8	735,5	736,9	742,4	**735,2**	742,6

Table 5.10: Limited flexibility for the centralized case.

It can be observed that the introduction of the limited flexibility considerably improves the solutions. In the intermediate case, the solutions for 7 and 8 jobs are especially improved when the flexibility of jobs is limited downwards, whereas in the centralized case an improvement is achieved for all cases which include limitations on flexibility.

5.6 Experimentation

Based on the previous study of the basic and the alternative arrangements, the experimentation is focused on the following aspects.

- Intermediate versus centralized buffer location
- Difference in physical size of jobs
- Semi dynamic demand

A flowshop which consists of 10 stations is used. After station 3, 5 and 8 there is access to resequencing buffer places. In the case of the intermediate buffer, a single buffer place is provided at each resequencing possibility and in the case of the centralized buffer, the same three stations have access to three resequencing buffer places simultaneously. The range of the production time is [1...100], for the setup cost [2...8] and for the setup time [1...5]. The number of jobs is varied in the range of 4 to 10 and the objective function is the weighted sum of the makespan (factor 1.0) and the setup cost (factor 0.3), where the setup time is indirectly included in the calculation of the makespan and its weight (δ) is set to zero.

5.6.1 Intermediate versus centralized buffer location

The arrangement A_P gives the best found permutation solution. The arrangements which are then used for comparison are A_{NP}, A_C and A_{MSS} and the four cases of limited flexibility, previously introduced in section 5.5, are considered. The maximum execution time, t_{max}, in any case is set to 600 sec. In the case of the alternative arrangement A_{MSS}, the time for each iteration, t_{MSS}, is set to 30 seconds and the number of iterations is consequently 20.

Jobs	A_P	Interm	Centr
4	481,2	**477,7**	**477,7**
5	470,5	**451,8**	**451,8**
6	564,6	**537,1**	537,7
7	568,7	**565,8**	566,9
8	589,6	589,1	**588,0**
9	684,3	684,0	**680,7**
10	736,1	746,1	**735,2**

Table 5.11: Comparison of the intermediate versus the centralized buffer location. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

The results for the permutation case and the summary for the intermediate and the centralized case are given in table 5.11. In any case, the solutions from the permutation case are improved. For the case of 4 and 5 jobs the intermediate case gives as good results as the centralized case. For the case of 6 and 7 jobs, the intermediate case performs better and afterwards, for the case of 8, 9 and 10 jobs, the centralized case performs better.

		No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
Jobs	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
4	481,2	481,2	480,0	**477,7**	481,2	480,0	**477,7**	481,2	480,0	**477,7**	481,2	480,0	**477,7**
5	470,5	470,5	470,5	**451,8**	470,5	470,5	**451,8**	470,5	470,5	**451,8**	470,5	470,5	**451,8**
6	564,6	564,6	563,4	**537,1**	564,6	563,4	**537,1**	564,6	563,4	**537,1**	564,6	563,4	**537,1**
7	568,7	568,7	567,8	572,5	568,7	566,6	566,4	568,7	566,6	**565,8**	568,7	566,6	569,1
8	589,6	622,1	589,6	623,3	589,6	589,6	599,0	621,6	589,6	**589,1**	589,6	591,7	599,0
9	684,3	694,5	732,2	695,9	694,5	692,6	694,3	711,1	684,3	685,5	694,5	**684,0**	694,1
10	736,1	755,8	755,1	746,1	752,3	813,4	747,4	755,8	746,1	746,5	752,3	748,1	747,2

Table 5.12: Detailed information for the intermediate buffer location. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

The detailed results for the intermediate case are presented in table 5.12. For the cases up to 8 jobs the alternative arrangement A_{MSS} produces the most promising results, especially when limited flexibility is included. The arrangement A_C performs second best.

		No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
Jobs	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
4	481,2	481,2	480,0	**477,7**	481,2	480,0	**477,7**	481,2	480,0	**477,7**	481,2	480,0	**477,7**
5	470,5	470,5	470,5	**451,8**	470,5	470,5	**451,8**	470,5	470,5	**451,8**	470,5	470,5	**451,8**
6	564,6	564,6	564,0	**537,7**	564,6	564,0	**537,7**	564,6	564,0	**537,7**	564,6	564,0	**537,7**
7	568,7	568,7	567,5	567,6	568,7	567,5	**566,9**	572,7	567,5	567,0	568,7	567,5	567,1
8	589,6	589,6	589,6	618,4	589,6	589,6	588,5	621,6	589,6	589,4	621,6	589,6	**588,0**
9	684,3	694,5	682,5	691,5	694,5	683,4	688,9	711,1	**680,7**	684,8	694,5	682,5	689,7
10	736,1	752,3	735,5	743,6	752,3	735,5	743,7	755,8	735,5	736,9	742,4	**735,2**	742,6

Table 5.13: Detailed information for the centralized buffer location. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

In the centralized case, presented in table 5.13, the results for 8 to 10 jobs are improved compared to the intermediate case. For the cases of 9 and 10 jobs the alternative arrangement A_C gives the best improvement.

5.6.2 Difference in physical size of jobs

Introducing limitations on the physical size of the buffer places on the one hand restricts possible solutions, but on the other hand minimizes the necessary buffer area. As a concrete example, three differently sized buffer places (large, medium, small) are available and the ratio of jobs is $\frac{3}{10}$, $\frac{3}{10}$ and $\frac{4}{10}$ for large, medium and small, respectively. The allocation of the buffer places to the buffers considers five scenarios for the intermediate case ("I111", "I231", "I132", "I222", "I333") and three scenarios for the centralized case ("C1", "C2", "C3"). "I132" represents 1 small, 1 large and 1 medium buffer place, located as intermediate resequencing buffer places after stations 3, 5 and 8, respectively. "C2" represents 1 medium buffer place, located as a centralized buffer place, accessible from stations 3, 5 and 8. "I333" and "C3" are the two cases which provide the largest flexibility in terms of physical size restrictions.

The maximum execution time, t_{max}, is set to 600 seconds for all cases. For the alternative arrangement A_{MSS}, the time for each iteration, t_{MSS}, is set to 30 seconds and the number of iterations is consequently 20. Furthermore, only the following two cases, including limited flexibility are considered:

$$W_{up} = 5 \quad \text{and} \quad W_{down} = 1$$

$$W_{up} = 1 \quad \text{and} \quad W_{down} = 5$$

The case for the arrangement A_{NP} which does not result in improvements of the permutation sequence is shown in table 5.14, and the case for the arrangement A_C is given in table 5.15. In general, the best solutions are achieved for the cases "I333" and "C3".

		I111		I231		I132		I222		I333		C1		C2		C3	
Jobs	A_P	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5
4	481,2	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**	**481,2**
5	470,5	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**
6	564,6	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**	**564,6**
7	568,7	572,7	**568,7**	572,7	**568,7**	572,7	**568,7**	572,7	**568,7**	**568,7**	**568,7**	572,7	**568,7**	572,7	**568,7**	572,7	**568,7**
8	589,6	621,3	**589,6**	621,6	**589,6**	621,6	**589,6**	621,6	**589,6**	621,6	**589,6**	621,6	**589,6**	621,6	**589,6**	621,6	621,6
9	684,3	701,4	**684,3**	701,4	**684,3**	701,4	**684,3**	701,4	**684,3**	711,1	694,5	701,4	**684,3**	701,4	**684,3**	711,1	694,5
10	736,1	755,8	**742,4**	755,8	**742,4**	755,8	**742,4**	755,8	**742,4**	755,8	752,3	755,8	**742,4**	755,8	**742,4**	755,8	**742,4**

Table 5.14: Difference in physical size of jobs for the arrangement A_{NP}. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

In the case of the alternative arrangement A_{MSS}, the variation of the solutions is more significant. In general, the best solutions are among the cases "I333" and "C3". However, considering the reduction of the occupied buffer area or the reduced tank size, an inferior solution may be justified.

		I111		I231		I132		I222		I333		C1		C2		C3	
Jobs	A_P	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5
4	481,2	481,2	481,2	**480,0**	**480,0**	**480,0**	**480,0**	481,2	481,2	**480,0**	**480,0**	**480,0**	**480,0**	**480,0**	**480,0**	**480,0**	**480,0**
5	470,5	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**	**470,5**
6	564,6	564,6	564,6	564,3	564,3	564,3	564,3	564,3	564,3	**563,4**	**563,4**	564,0	564,0	564,0	564,0	564,0	564,0
7	568,7	568,7	568,7	568,7	568,7	567,8	567,8	567,8	567,8	**566,6**	**566,6**	567,5	567,5	567,5	567,5	567,5	567,5
8	589,6	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	591,7	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**	**589,6**
9	684,3	684,3	682,8	684,3	681,0	684,3	681,0	684,3	681,0	684,3	684,0	684,3	681,6	684,3	681,6	**680,7**	682,5
10	736,1	735,5	735,5	735,5	736,7	735,5	736,7	735,5	736,7	746,1	748,1	735,5	**735,2**	735,5	**735,2**	735,5	**735,2**

Table 5.15: Difference in physical size of jobs for the arrangement A_C. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

The comparison of the different arrangements is shown in table 5.17 for the intermediate case (a) and the centralized case (b). Considering the best obtained solution for all cases of different physical size of the buffer places, the arrangement A_{MSS} performs better for cases up to 8 jobs and does not improve the best permutation sequence from arrangement A_P for the case of 9 and 10 jobs. Whereas the arrangement A_C seems to perform good for all cases and in any case does reach at least as good results as the best permutation solution.

		I111		I231		I132		I222		I333		C1		C2		C3	
Jobs	A_P	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5	5/1	1/5
4	481,2	478,9	478,9	**477,7**	**477,7**	**477,7**	477,7	478,9	478,9	**477,7**	**477,7**	478,9	478,9	**477,7**	**477,7**	**477,7**	**477,7**
5	470,5	470,5	470,5	464,2	464,2	467,6	467,6	464,2	464,2	**451,8**	**451,8**	470,5	470,5	464,2	464,2	**451,8**	**451,8**
6	564,6	539,2	539,2	**537,1**	**537,1**	539,2	539,2	537,7	537,7	**537,1**	**537,1**	539,2	539,2	537,7	537,7	537,7	537,7
7	568,7	568,7	568,7	568,7	568,7	568,4	568,4	567,6	567,6	**565,8**	569,1	568,7	568,7	567,9	567,9	567,0	567,1
8	589,6	593,3	591,3	595,2	595,2	592,9	592,4	592,9	592,4	589,1	599,0	593,3	591,3	592,9	591,1	589,4	**588,0**
9	684,3	690,6	688,1	691,5	691,5	686,9	687,3	689,7	690,3	685,5	694,1	691,3	688,7	690,9	688,6	**684,8**	689,7
10	736,1	744,0	741,9	743,8	743,8	740,9	746,3	743,0	744,6	746,5	747,2	744,0	741,7	743,0	741,4	**736,9**	742,6

Table 5.16: Difference in physical size of jobs for the arrangement A_{MSS}. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

Considering the average value of the obtained solutions, it is more evident that the arrangement A_{MSS} gives good results up to 6 jobs and the arrangement A_C gives promising results beyond this point.

Finally, table 5.18 shows the comparison of the intermediate versus the centralized case. In general, the result is similar to the one obtained in section 5.6.1. However, in some cases the solution is improved. This behavior was already recognized in section 5.5.

	a)	Minimum			Average			b)	Minimum			Average		
Jobs	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
4	481,2	481,2	480,0	**477,7**	481,2	480,5	**478,2**	481,2	481,2	480,0	**477,7**	481,2	480,0	**478,1**
5	470,5	470,5	470,5	**451,8**	470,5	470,5	**463,7**	470,5	470,5	470,5	**451,8**	470,5	470,5	**462,2**
6	564,6	564,6	563,4	**537,1**	564,6	564,2	**538,1**	564,6	564,6	564,0	**537,7**	564,6	564,0	**538,2**
7	568,7	568,7	566,6	**565,8**	570,3	**567,9**	568,2	568,7	568,7	567,5	**567,0**	570,7	**567,5**	567,9
8	589,6	589,6	589,6	**589,1**	605,6	**589,8**	593,4	589,6	589,6	589,6	**588,0**	610,9	**589,6**	591,0
9	684,3	684,3	**681,0**	685,5	694,8	**683,1**	689,6	684,3	684,3	**680,7**	684,8	696,2	**682,5**	689,0
10	736,1	742,4	**735,5**	740,9	750,1	**738,2**	744,2	736,1	742,4	**735,2**	736,9	749,1	**735,4**	741,6

Table 5.17: Comparison of the different arrangements when a difference in physical size of jobs appears. a) Intermediate case. b) Centralized case. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

Jobs	AP	Interm	Centr
4	481,2	**477,7**	**477,7**
5	470,5	**451,8**	**451,8**
6	564,6	**537,1**	537,7
7	568,7	**565,8**	567,0
8	589,6	589,1	**588,0**
9	684,3	681,0	**680,7**
10	736,1	735,5	**735,2**

Table 5.18: Comparison of the intermediate and the centralized case when a difference in physical size of jobs appears. Fields with grey background indicate that the obtained solution is equal or better than the permutation solution A_P, the best solution is highlighted in bold text.

5.6.3 Semi dynamic demand

The semi dynamic demand is referred to the case in which a fixed (but arbitrary) sequence is used for the first station. Due to the fact that the jobs can not be sequenced before entering the production line, the job sequence is supposed to be of worse quality with respect to the objective function. On the other hand, that is why the resequencing without prior sequencing is expected to result in a higher benefit.

Jobs	Perm	I111	I231	I132	I222	I333	C1	C2	C3
4	483,1	480,9	**480,7**	**480,7**	480,9	**480,7**	480,9	**480,7**	**480,7**
5	552,0	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**
6	647,5	627,0	**620,1**	**620,1**	**620,1**	**620,1**	628,5	622,2	622,2
7	636,5	627,7	627,7	625,0	625,0	**609,5**	628,0	627,7	616,1
8	673,6	646,6	646,6	644,8	644,8	644,8	646,6	644,8	**632,3**
9	744,3	719,4	719,4	716,1	716,1	716,1	719,4	716,1	**712,1**
10	813,7	786,5	763,2	**762,5**	785,9	791,2	786,5	786,5	764,0

Table 5.19: Semi dynamic demand, considering at the same time physical size limitations for the resequencing buffer places.

The same flowshop as in section 5.6.2 is considered, except for the fact that a semi dynamic demand is used. In order to solve this problem, the second cascade of the arrangement A_C, see figure 5.3, is used with an arbitrary job sequence. Table 5.19 shows the obtained results.

In general the presented solutions show improvements up to 11.1% with an average of 4.3%. The case with less restrictions, here the restrictions on the physical size of the resequencing buffer places, results in better solutions. However, in the case of 10 jobs, the cases ”I231” and ”I132” obtain better solutions. This behavior was already recognized in section 5.5.

5.7 Conclusions

In this chapter the proposed formulation for the Constrained Logic Programming (CLP), see chapter 4, was applied to the problem of resequencing jobs in a flowshop production line, using buffers which are located off the production line, either accessible from a single station (intermediate case) or from various stations (centralized case). These buffers are furthermore constrained in terms of the number of buffer places and the fact that a job may not be able to be stored in a certain buffer place, due to its extended physical size.

First, the basic arrangement was presented which uses the presented formulation in an unmodified way. In addition, and based on an analysis on semi dynamic demand, several alternative arrangements were presented, including *Cascading* of the permutation and the non-permutation case, and the *Multistart Cascading* with and without feedback. In the case of *Multistart cascading with feedback*, furthermore three different cases are distinguished: the ’Shift and swap operations of jobs’, the ’Probability for position of jobs’ and the ’Probability for succeeding jobs’.

Subsequently, the parameters of the individual arrangements were adjusted and the most promising two alternative arrangements, *Cascading* and *Multistart cascading with feedback and shift and swap operations*, were selected.

Due to the fact that, in the so far realized experimentation of the CLP, the centralized case performed better for larger problems, an important characteristic of the CLP was studied. The introduction of additional constraints, as for example limiting the number of positions a job can move upwards or downwards, namely *Limited Flexibility*, results in an improvement of the solutions for problems with more than 7 jobs.

Based on these studies, the experimentation was accomplished, using a flowshop with 10 stations and three possibilities to resequence jobs within the production line, including setup cost and setup time. The experimentation is divided into three parts. First, the comparison of the intermediate versus the centralized buffer location showed that the intermediate case performed better for problems with less than 8 jobs. Then, including buffers, constrained by the physical size of jobs to be stored, on the one hand limits the solutions, but on the other hand minimizes the necessary buffer area and in some case even leads to better solutions, caused by the fact that the problem is additionally constrained. Finally, the study of the semi dynamic demand showed an increased benefit with an average of 4.3% for the studied flowshop. More than 800 experiments were conducted and the case in which offline buffers are used in order to resequence jobs within the production line, in general outperformed the solutions which consider only permutation sequences.

It has to be mentioned that despite the fact that the replacement of the intermediate resequencing buffers by a centralized resequencing buffer results in benefits with respect to the objective function or in a reduced buffer area, additional effort maybe necessary in terms of logistics, in order to keep track of the stored jobs, or in terms of additional hardware which may be necessary to install in order to enable the transfer of the jobs to the centralized resequencing buffer. In any case the additional effort has to be properly weighed for the specific case.

In the next chapter a Genetic Algorithm is designed, based on the well performing alternative arrangement A_{MSS} (see figure 5.5), using shift and swap operations for jobs, known as *Mutation*. In addition, the two genetic operators for inheritance and crossover are included. The purpose of introducing this heuristic method is to be able to solve larger instances of the problem under study. Apart from the performance study of the GA, accomplished in chapter 7, a comparison of the CLP and the GA is accomplished in chapter 8.

6 Heuristic Approach: Genetic Algorithm

The exact approach, presented in chapter 4 solves to optimality the problem of resequencing jobs within the flowshop, considering constrained buffers. Due to the fact that the problem is NP-hard and due to the problem size limitations of the exact approach, a more versatile heuristic approach is required.

The immanent characteristic of many heuristic methods is its inability to solve the problem to optimality but for the sake of computation allow increased problem sizes. Among the multitude of heuristic approaches the Genetic Algorithm was chosen, based on promising results achieved by a plurality of researchers, in general combinatorial optimization and furthermore in the special field of flowshop sequencing and scheduling, as for example Reeves, 1995, Rubin and Ragatz, 1995, Leu et al., 1996, Michaelewicz, 1996, Hyun et al., 1998, Sevaux and Dauzère-Péres, 2003, Niu, 2005, and Ruiz et al., 2006.

The heuristic approach used here is a variation of the Genetic Algorithm explained by Michaelewicz, 1996. In general, Genetic algorithms are used in maximizing or minimizing an objective function within a set of constraints, as for example in the case of a flowshop, and are especially applicable when the relationships are non-linear or discontinuous.

For the implementation of the heuristic approach the programming language Ansi-C was used. In contrast to programming in Constrained Logical Programming (CLP), used for the exact approach, the programming of the heuristic approach in C is sequential, which results in differences in the implementation of the calculation of the schedule for the sequenced jobs as well as in the determination of the buffer occupation.

6.1 General outline and terminology

The concept of the Genetic Algorithm was first formulated by Holland, 1973, and Holland, 1975, and can be seen as the application of the principles of evolutionary biology, also known as the survival of the fittest, to computer science.

Genetic algorithms are typically implemented as a computer simulation in which a population of chromosomes, each representing a solution of the optimization problem, evolves towards better solutions. The evolution starts from an initial population which may be determined randomly. In each generation, the fitness of the whole population is evaluated, multiple individuals are stochastically selected from the current population, based on their fitness, and modified to form a new population, which becomes current in the next iteration of the algorithm. The modifications are biologically-derived techniques, commonly achieved by inheritance, mutation and crossover (or recombination).

The pseudocode of the Genetic Algorithm can be formulated as follows:

```
Initialize population
   1. Calculate schedule of sequenced jobs, defined by
      the chromosomes of the current population
   2. Evaluate fitness values of individuals of the current population
   3. Select parents for reproduction
   4. Perform modifications to selected parents to generate the next population.
Repeat steps 1-4 until some stopping rule is reached.
```

The terminology used throughout this chapter with relation to the Genetic Algorithm is as follows:

- **Gene:** Smallest fraction of a possible solution, here a gene is represented by a single job. Considering the sequence of jobs at one station, the individual jobs may not be repeated, except in the case of using the minimal-part-set which is the definition of repetitive models.

- **Chromosome v_s:** Representation of a solution of the optimization problem, formed by a set of genes. These chromosomes are evaluated by the fitness function, which rates them according to how good their solution is. Chromosomes which produce better solutions are more likely to breed. Depending on the problem formulation, it is possible to have chromosomes which do not represent a feasible solution but may contain some promising character.

- **Generation:** In biology, a generation can be defined as the chromosomes born at about the same time. Passing chromosome-information from one generation to the next, the chromosomes evolve in terms of fitness. The concept of the Genetic Algorithm is to breed a generation with a high level of fitness and generally terminates after the reproduction of a predetermined **number of generations** G or after the **maximum time** $T_{\max}$ has elapsed.

- **Population:** The Set of chromosomes which represent a generation is called population. The number of chromosomes, belonging to one generation is the **population size** R. The first generation of chromosomes is the **initial population**.

- **Fitness function** $eval(v_s)$**:** The fitness function is the objective function which quantifies the optimality of a chromosome v_s, so that a particular chromosome may be ranked against all other chromosomes.

- **Offspring/reproduction:** In biology, offspring is the product of the process of generating a new organism produced by one or more parents. In Genetic Algorithm this offspring is accomplished by different reproduction-mechanisms:

  - **Inheritance** is the procedure of maintaining promising parents as an exact copy at the moment the next generation is formed.
  - **Mutation** is the procedure of applying arbitrary changes to individual genes of a single chromosome. The purpose of mutation in Genetic Algorithm is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most Genetic Algorithm avoid only taking the fittest of the population in generating the next but rather a random selection with a weighting towards those that are more fit.

– **Crossover** is the procedure of arbitrarily considering two or more chromosomes for interbreed. A crossover point on the parent chromosomes is selected and all data beyond this point in the chromosome is swapped between the two parents.

6.2 Data structure

The use of the Genetic Algorithm for the problem of resequencing jobs in a flowshop, provided with limited intermediate or centralized offline-buffers requires various adaptations. An overview on the data structure of the Genetic Algorithm is given in continuation.

6.2.1 Random number generation

The Genetic Algorithm is a metaheuristic solution method which is strongly based on the generation of random numbers. Starting with the random creation of the initial generation, the random selection of parents, even though also depending on their fitness, as well as the determination of the genes which are to be mutated right up to the determination of the crossover point.

A random number generator is a computational or physical device designed to generate a sequence of numbers that does not have any easily discernable pattern. However, the generation of a random number, realized on a computer is only producing a pseudo-random number. A pseudo-random number is not truly random, it only approximates some of the properties of random numbers.

Most such algorithms attempt to produce samples that are uniformly distributed. Furthermore a seed may be used for its initialization with which, every time it is started, the same sequence of random numbers is generated. The generator also may use a random event, such as the current time in milliseconds, in order to give different sequences every time it is used. The proposed Genetic Algorithm uses a defined seed in order to be able to repeat the obtained results.

6.2.2 Parameter and variable definition

In what follows, the necessary parameters and variables of the Genetic Algorithm are presented. The introduction of parameters and variables, necessary for the calculation of the schedule will be given in section 6.3.

s	Index of chromosomes		
g	Index of generations		
r	Index of resequencing possibilities		
υ_s	Chromosome s		
R	Population size	$s = 1, ..., R$	
G	Number of generations	$g = 1, ..., G$	
L	Number of resequencing possibilities	$r = 1, ..., L$	
N	Number of jobs		
MBS	Number of best solutions to maintain		
p_{b}	Probability to eliminate best solutions	p_{b}	$\in [0..1]$
$p_{\text{c-I}}$	Probability of crossover-I	$p_{\text{c-I}}$	$\in [0..1]$
$p_{\text{c-II}}$	Probability of crossover-II	$p_{\text{c-II}}$	$\in [0..1]$
g_{deg}	Parameter defining the degression of $p_{\text{c-I}}$ and $p_{\text{c-II}}$		
$p_{\text{m-I(f)}}$	Probability of mutation I (forward)	$p_{\text{m-I(f)}}$	$\in [0..1]$
$p_{\text{m-I(b)}}$	Probability of mutation I (backward)	$p_{\text{m-I(b)}}$	$\in [0..1]$
$p_{\text{m-II}}$	Probability of mutation II	$p_{\text{m-II}}$	$\in [0..1]$
pos	Random position of a gene in the chromosome υ		
FP	Penalty for non-feasible solution		

6.2.3 Reduction of chromosome size

The computational effort necessary to solve the Genetic Algorithm is directly related to the size of the chromosomes. Therefore it is preferable to work with the minimum necessary size.

The chromosome is basically formed by queuing the sequences $\Pi_1 ... \Pi_M$, M fractions of length N. Due to the fact that a change in the sequence can only occur in the case a station

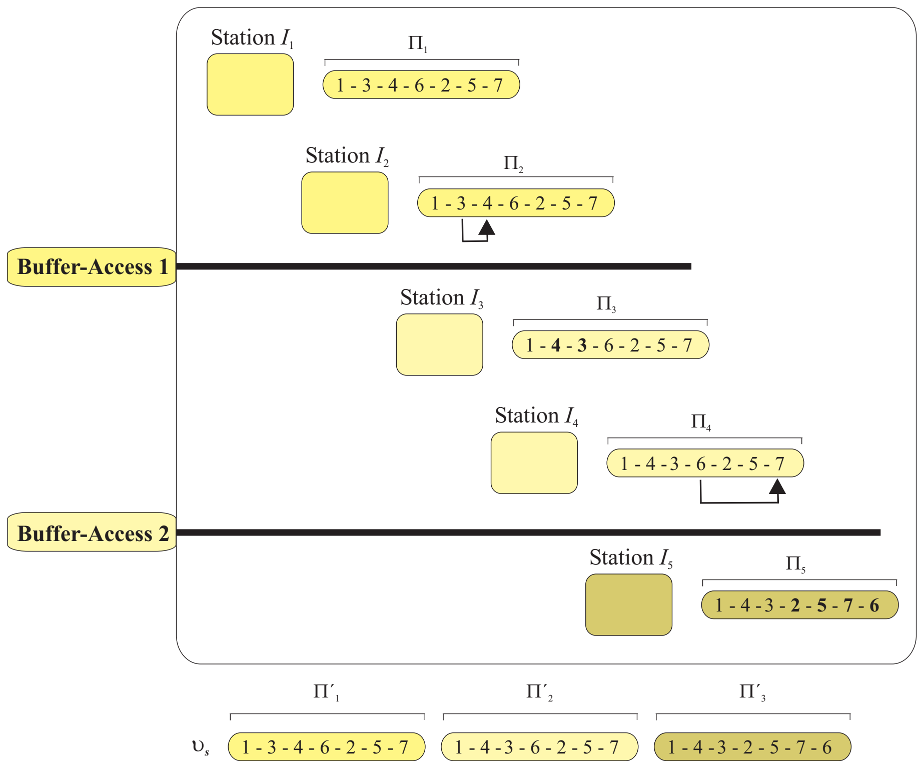


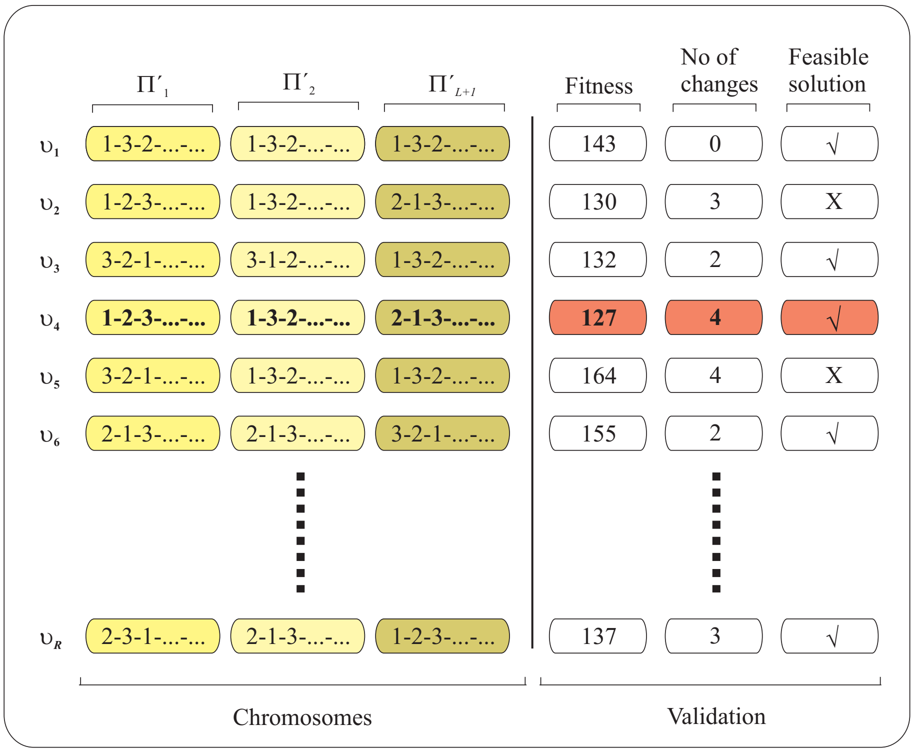
Figure 6.1: Reduction of chromosome υ_s from $\Pi_1 ... \Pi_5$ to $\Pi_1^{'} ... \Pi_3^{'}$ for the Genetic Algorithm due to the fact that resequencing is not possible for all stations.

is provided with access to a resequencing buffer, the sequences for several consecutive stations is the same. The sequences of stations which are subsequent to a station with access to resequencing buffers, until the next station with access to resequencing buffers, are not considered in the chromosome. This results in the reduction of the necessary chromosomes from $\Pi_1, ..., \Pi_i, ..., \Pi_M$ to $\Pi_1^{'}, ..., \Pi_r^{'}, ..., \Pi_{L+1}^{'}$. L is the number of stations with access to a resequencing buffer, plus 1 for the sequence at station 1.

Figure 6.1 shows an example of a flowshop with seven jobs to be processed on five stations. Station 2 and station 4 have access to a resequencing buffer. This results in the same sequence for the first two stations ($\Pi_1^{'} = \Pi_1 = \Pi_2$) and the next two stations ($\Pi_2^{'} = \Pi_3 = \Pi_4$). The final station processes the sequence $\Pi_3^{'} = \Pi_5$. The size of the resulting chromosome υ_s is reduced to a fraction of $2/5$.

6.2.4 Population information

The population of one generation is defined by the chromosomes υ_s, each containing a set of sequences $\Pi_1^{'}...\Pi_{L+1}^{'}$, and the population size R, the number of chromosomes.



	Π'_1	Π'_2	Π'_{L+1}	Fitness	No of changes	Feasible solution
υ_1	1-3-2-...-...	1-3-2-...-...	1-3-2-...-...	143	0	√
υ_2	1-2-3-...-...	1-3-2-...-...	2-1-3-...-...	130	3	X
υ_3	3-2-1-...-...	3-1-2-...-...	1-3-2-...-...	132	2	√
υ_4	**1-2-3-...-...**	**1-3-2-...-...**	**2-1-3-...-...**	**127**	**4**	√
υ_5	3-2-1-...-...	1-3-2-...-...	1-3-2-...-...	164	4	X
υ_6	2-1-3-...-...	2-1-3-...-...	3-2-1-...-...	155	2	√
υ_R	2-3-1-...-...	2-1-3-...-...	1-2-3-...-...	137	3	√

Figure 6.2: Data structure used for one generation of the Genetic Algorithm, divided into the chromosome information and the validation. The validation part contains information about fitness, number of job changes and the feasibility of the solution.

The set of data, used for one generation of the Genetic Algorithm is shown in figure 6.2. The first part considers R chromosomes. Each chromosome is divided into $L+1$ fractions of N genes: L representing the number of stations with access to a resequencing buffer, 1 is added for the sequence at the first station, and N being the number of jobs. The second part, the validation, states the fitness (qualitative measure of the chromosome), the number of job changes necessary to process the jobs sequenced according the chromosome, and finally an indicator specifying if the chromosome represents a feasible solution or not.

6.2.5 Initial population

The initial population forms the population of the first generation. The initial population used in this Genetic Algorithm is composed by randomly generated permutation sequences, resulting in chromosomes with $\Pi_1^{'} = \Pi_2^{'} = ... = \Pi_{L+1}^{'}$.

6.3 Schedule

For the schedule calculation, the relevant information on how jobs are sequenced is provided by the reduced chromosome $\upsilon_s = \{\Pi_1^{'}..\Pi_{L+1}^{'}\}$. As a result of the calculation of the schedule, the start time $s_{i,j}$ and the completion time $c_{i,j}$ of all jobs j at all stations i is calculated, also considering the setup time $ST_{i,e,f}$, if existent.

The schedule calculation in addition determines the buffer occupation, taking into account the physical size limitations of jobs which are to be stored temporarily in a specific buffer place. The resequencing buffer can be intermediate or centralized.

The first step is the **expansion of the reduced chromosome**. The reduction of the size of the chromosome was performed in order to reduce the computational effort for the Genetic Algorithm, see section 6.2.3. For the calculation of the schedule of the sequenced jobs the extended sequence representation is required, indicating the particular sequence for each station.

6.3.1 Parameter and variable definition

The necessary parameters and variables for the calculation of the schedule of the sequenced jobs are given in continuation.

i	Index for station	
M	Number of stations	$i = 1, ..., M$
j, k	Index for jobs	
N	Number of jobs	$j, k = 1, ..., N$
Π_i	Job sequence at station i	$\Pi_i = \{\Pi_{i,1}, ..., \Pi_{i,N}\}$
$P_{i,j}$	Processing time of job j at station i	
$s_{i,j}$	Start time of job j at station i	
$c_{i,j}$	Completion time of job j at station i	

Let $P_{i,j}$ be the processing time of job j at station i, whereas $P_{i,[j]}$ is the processing time of the job at position j at station i. This convention is also applicable for variables like $S_{i,[j]}$ and $C_{i,[j]}$

Variables used with setup considerations:

μ_j	Model type of job j
$SC_{i,e,f}$	Setup cost occurring at station i to change from model type e to model type f
$ST_{i,e,f}$	Setup time occurring at station i to change from model type e to model type f

6.3.2 Schedule calculation

The **schedule of the sequenced jobs** is calculated, based on the equations already introduced for the exact approach (see chapter 4).

$$\begin{aligned}
s_{1,[1]} &= 0 && && (6.1)\\
s_{i,[1]} &= c_{i-1,[1]} && i \geq 2 && (6.2)\\
s_{1,[j]} &= c_{1,[j-1]} && j \geq 2 && (6.3)\\
s_{i,[j]} &= \max\{c_{i,[j-1]}, c_{i-1,[j]}\} && i \geq 2; j \geq 2 && (6.4)\\
c_{i,[j]} &= s_{i,[j]} + P_{i,[j]} && && (6.5)
\end{aligned}$$

In case in which setup time exists, an additional time $ST_{i,e,f}$ is considered when at station i a job of model type e precedes a job of model type f at station i. In order to consider this setup time constrains (6.3) and (6.4) need to be altered to (6.6) and (6.7).

$$\begin{aligned}
s_{1,[1]} &= 0 && &&\\
s_{i,[1]} &= c_{i-1,[1]} && i \geq 2 &&\\
s_{1,[j]} &= c_{1,[j-1]} + ST_{1,\mu_{1,[j-1]},\mu_{1,[j]}} && j \geq 2 && (6.6)\\
s_{i,[j]} &= \max\{c_{i,[j-1]} + ST_{i,\mu_{i,[j-1]},\mu_{i,[j]}}, c_{i-1,[j]}\} && i \geq 2; j \geq 2 && (6.7)\\
c_{i,[j]} &= s_{i,[j]} + P_{i,[j]} && &&
\end{aligned}$$

In the case in which **zero processing time** exists together with setup time or setup cost, the calculation of the schedule is more complex. This is caused by the fact that if a job at position j has zero processing time at station i, the resulting setup time or setup cost is depending on the last job without zero processing time, e.g. at position $j-1$, and the next job without zero processing time, e.g. at position $j+1$. In order for the following formulation to work properly, it must be assured that if a job has zero processing time at a station, then the setup time, which appears at this station in relation with this particular job, has to be zero as well.

$$
\begin{aligned}
c_{i,0} &= -1 && i = 1, ..., M && (6.8)\\
s_{1,[1]} &= 0 && &&\\
s_{i,[1]} &= c_{i-1,[1]} && i \geq 2 &&\\
s_{1,[j]} &= max\{c_{1,[k]} + ST_{1,\mu_{1,[k]},\mu_{1,[j]}}, 0\} && j \geq 2 && (6.9)\\
& && k = \max_{x=[0,1,...,i-1]}\{x\} \mid c_{1,[x]} \neq 0 &&\\
s_{i,[j]} &= \max\{c_{i,[k]} + ST_{i,\mu_{i,[k]},\mu_{i,[j]}}, c_{i-1,[j]}\} && i \geq 2; j \geq 2 && (6.10)\\
& && k = \max_{x=[0,1,...,i-1]}\{x\} \mid c_{i,[x]} \neq 0 &&\\
c_{i,[j]} &= s_{i,[j]} + P_{i,[j]} && &&
\end{aligned}
$$

Considering the setup time, when calculating the schedule of the jobs, the array for the completion time, $c_{i,j}$ has to be extended by a fictitious vector, $c_{i,0}$, set to -1 for all stations, constraint (6.8). Then, constraint (6.9) considers station 1 and calculates the sum of the completion time of job k and the setup time, occurring when changing from model type e to model type f. Job k is the first job succeeding job j without zero processing time. In the case that all succeeding jobs have zero processing time, k is zero and with the max-function of constraint (6.9) the start time, $s_{1,[j]}$, of job j is set to zero. Constraint 6.10 considers the other stations.

6.3.3 Buffer occupation

Once the schedule for the sequenced jobs is calculated, it is necessary to determine the resulting buffer occupation of the resequencing buffer. The number of buffer places is finite and the physical size of the individual buffer places is limited by its physical size, not allowing to store a job which exceeds the physical size.

The procedure is similar to the one already presented in the exact approach, chapter 4. However, due to the fact that the heuristic approach is programmed in a sequential way in the programming language C, the formulation for the calculation of the Buffer occupation is slightly different.

First the formulation of the general part of the determination of the buffer occupation is described. Then, the parts which are unique for the case of **intermediate resequencing buffers**, only accessible from a single stations, and the case of a **centralized resequencing buffer**, accessible from various stations, are described separately.

General proceeding

The buffers are defined by $\Phi_{r,d}$, for the intermediate case, and Φ'_d, for the centralized case. The definition may be accomplished in an arbitrary manner, i.e. the assignation of the size of the buffer places does not have to be in an ascending way. That is why the first step in the determination of the buffer occupation is the sorting of $\Phi_{r,d}$ and Φ'_d, in an ascending manner.

Then follows the transformation of $\Pi_{r,[k]}$ to $\Pi'_{r,j}$, indicating that the job j after the r-th resequencing possibility is sequenced at position k.

$$\Pi_{r,[k]} = j \quad \rightarrow \quad \Pi'_{r,j} = k \qquad r = 1...L+1; \forall j,k \tag{6.11}$$

In order to determine whether a job requires to be taken off the production line, with the aim of reinserting it at a later point of time, the two constraints (6.12) and (6.13) were already introduced for the exact approach.

$\lambda_{r,[j],[k]}$ indicates if the job at position k succeeds the job at position j in the sequence Π_r as well as in sequence Π_{r+1}. In this case $\lambda_{r,[j],[k]}$ is forced to 1.

$$\begin{aligned}
\Pi'_{r+1,\Pi_{r,j}} < \Pi'_{r+1,\Pi_{r,k}} \quad &\rightarrow \quad \lambda_{r,[j],[k]} = 1 \qquad \forall r; \forall j,k|_{j<k} \\
\Pi'_{r+1,\Pi_{r,j}} \geq \Pi'_{r+1,\Pi_{r,k}} \quad &\rightarrow \quad \lambda_{r,[j],[k]} = 0 \qquad \forall r; \forall j,k|_{j<k} \\
&\qquad \lambda_{r,[j],[k]} = 0 \qquad \forall r; \forall j,k|_{j\geq k}
\end{aligned} \tag{6.12}$$

Then, $\Lambda_{r,[j]}$ determines if the number of jobs succeeding the job at position j in sequence Π_r and in sequence Π_{r+1}, indicated by $\lambda_{r,[j],[k]}$, is the same as the number of jobs which succeed the job at position j in sequence Π_r. In this case $\Lambda_{r,[j]}$ is set to 1, indicating that job j occupies a buffer place of the resequencing buffer r.

$$\begin{aligned} \sum_{k=1}^{N} \lambda_{r,[j],[k]} \neq N - j \quad &\rightarrow \quad \Lambda_{r,[j]} = 1 \qquad \forall r, j \\ \sum_{k=1}^{N} \lambda_{r,[j],[k]} = N - j \quad &\rightarrow \quad \Lambda_{r,[j]} = 0 \end{aligned} \tag{6.13}$$

Intermediate resequencing buffer

Once the jobs have been determined which are to be taken off the line for the purpose of resequencing, it is necessary to ensure that the available resequencing buffer place can only be loaded with one job at a time. Furthermore, the assigned job can not exceed the size of the buffer place.

$\Delta_{r,[j],[k]}$ is set in the case in which the job at position k after sequence Π_r has been temporarily stored in the respective resequencing buffer at the point of time the job at position j is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in a buffer.

$$\begin{aligned} &(\Lambda_{r,[j]} = 1) \wedge (\Lambda_{r,[k]} = 1) \wedge \\ &\wedge (c_{\Omega_r,[j]} \geq c_{\Omega_r,[k]}) \wedge (c_{\Omega_r,[j]} < s_{\Omega_r+1,[k]}) \quad \rightarrow \quad \Delta_{r,[j],[k]} = 1 \\ &\qquad\qquad \forall r; \forall j, k|_{j \neq k} \\ \\ &(\Lambda_{r,[j]} = 0) \vee (\Lambda_{r,[k]} = 0) \vee \\ &\vee (c_{\Omega_r,[j]} < c_{\Omega_r,[k]}) \vee (c_{\Omega_r,[j]} \geq s_{\Omega_r+1,[k]}) \quad \rightarrow \quad \Delta_{r,[j],[k]} = 0 \\ &\qquad\qquad \forall r; \forall j, k|_{j \neq k} \\ \\ &\Delta_{r,[j],[k]} = 0 \qquad \forall r; \forall j, k|_{j = k} \end{aligned} \tag{6.14}$$

Each job, requiring to be taken off the line for the purpose of resequencing, is assigned to a free buffer place which fulfills the physical size restriction of the job, starting with the smallest free buffer place. The argument of $\Psi_{r,[j]}$ is set to the number of the respective resequencing buffer place at r-th resequencing possibility to which the job at position j is assigned.

Centralized resequencing buffer

Similar to the case of the intermediate resequencing buffer, it is necessary to ensure that the available resequencing buffer place can only be loaded with one job at a time and the assigned job can not exceed the size of the buffer place.

Δ' is set in the case in which the job at position k at the t-th resequencing possibility has been temporarily stored in the respective resequencing buffer at the point of time the job at position j at the r-th resequencing possibility is to be taken off the line. This restriction serves to detect how many jobs are already temporarily stored in the buffer.

$$\begin{aligned} &(\Lambda_{r,[j]} = 1) \wedge (\Lambda_{t,[k]} = 1) \wedge \\ &\wedge (c_{\Omega_r,[j]} \geq c_{\Omega_t,[k]}) \wedge (c_{\Omega_r,[j]} < s_{\Omega_t+1,[k]}) \quad \rightarrow \quad \Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 1 \\ &\qquad \forall r,t; \forall j,k|_{j \neq k} \end{aligned} \tag{6.15}$$

$$\begin{aligned} &(\Lambda_{r,[j]} = 0) \vee (\Lambda_{t,[k]} = 0) \vee \\ &\vee (c_{\Omega_r,[j]} < c_{\Omega_t,[k]}) \vee (c_{\Omega_r,[j]} \geq s_{\Omega_t+1,[k]}) \quad \rightarrow \quad \Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 0 \\ &\qquad \forall r,t; \forall j,k|_{j \neq k} \end{aligned}$$

$$\Delta'_{(r-1)\cdot N+[j],(t-1)\cdot N+[k]} = 0 \qquad \forall r,t; \forall j,k|_{j=k}$$

Each job, requiring to be taken off the line for the purpose of resequencing, is assigned to a free buffer place which fulfills the physical size restriction of the job, starting with the smallest free buffer place. The argument of $\Psi'_{r,[j]}$ is set to the number of the respective resequencing buffer place to which the job at position j is assigned at the r-th resequencing possibility.

6.4 Fitness function

The evaluation function $eval(\upsilon_s)$ describes the qualitative measure of the chromosome υ_s to be minimized and includes five components:

$$\alpha \cdot \ (s_{M,[N]} + P_{M,[N]}) \qquad \text{Makespan} \tag{6.16}$$

$$\gamma \cdot \ \sum_{i=1}^{M} \sum_{j=1}^{N-1} SC_{i,\mu_{i,[j]},\mu_{i,[j+1]}} \qquad \text{Setup cost} \tag{6.17}$$

$$\delta \cdot \ \sum_{i=1}^{M} \sum_{j=1}^{N-1} ST_{i,\mu_{i,[j]},\mu_{i,[j+1]}} \qquad \text{Setup time} \tag{6.18}$$

$$\left\{ \begin{array}{ll} FP & \text{if not feasible} \\ 0 & \text{if feasible} \end{array} \right\} \qquad \text{Penalty for Non-Feasibility} \tag{6.19}$$

The components (6.16) to (6.18) were already introduced as part of the objective function for the determination of the exact solution in Chapter 4. In contrast to the Constrained Logic Programming, it is not necessary to impose a penalty for a high number of resequencing because in the case of equivalent solutions, with respect to the fitness function, the one which requires less resequencing is selected. In the case in which jobs have zero processing time and also setup cost or setup time are appear, the equations (6.17) and (6.18) only consider setup cost and setup time for succeeding jobs that do not have zero processing time, similar to equation (6.9) and (6.10).

The component (6.19) is necessary to impose a penalty for solutions which are not feasible, generated by the genetic operators. This is caused by the fact that position changes of jobs may be forced which may not be possible due to the physical size of a job exceeding the physical size of the respective buffer place, or the limited number of buffer places. In this case an additional cost is applied in form of a penalty, FP. Despite the fact that a solution is not feasible, it may generate valuable and feasible solutions in one of the succeeding generations. Therefore it is not erased from the population, but imposed with a penalty.

The "fittest" chromosome is the one which, amongst the feasible solutions, has the smallest calculated by the fitness function.

6.5 Genetic operators

The genetic operators specify in which way the subsequent population is generated by reproduction of the actual population, taking into account that "fitter" solutions are more promising and therefore are more likely to reproduce. Even a non-feasible solution is able to reproduce, caused by the fact that it may generate valuable and feasible solutions in one of the proceeding generations.

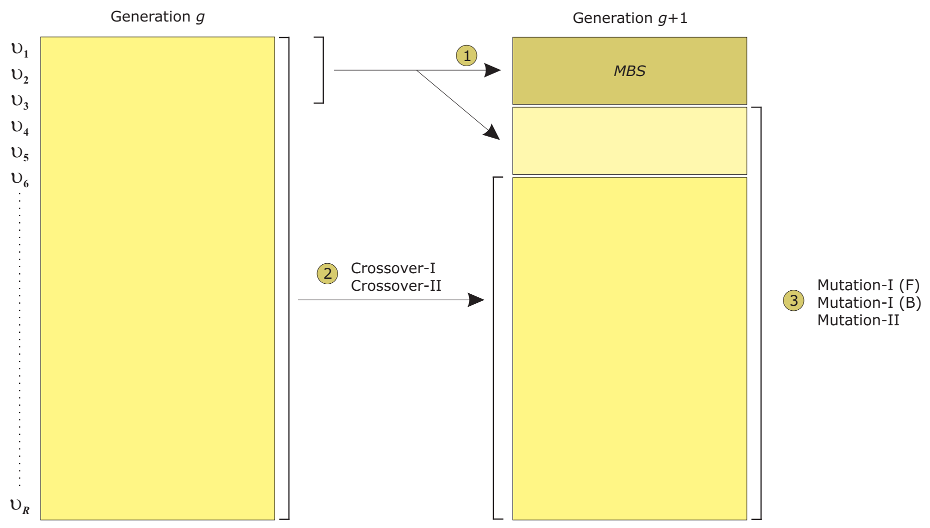


Figure 6.3: Application of the genetic operators to the population of generation g to form the next generation $g + 1$. The sequence in which the genetic operators are applied is (1) Inheritance, (2) Crossover and finally (3) Mutation.

Figure 6.3 gives an overview on how the genetic operators are applied to the population of generation g in order to form the next generation $g + 1$.

6.5.1 Validation

Before applying the genetic operators, the actual population requires to be validated and the chromosomes for reproduction need to be selected.

Total fitness of population F: The total fitness of the population F is the sum of the individual fitness values of the individual chromosomes and is calculated as:

$$F = \sum_{s=1}^{R} eval(\upsilon_s) \tag{6.20}$$

Probability of individual chromosome p_s: The probability of the individual chromosomes p_s is a measure of how likely a chromosome is to be selected for reproduction. The sum over all chromosomes is equal to 1. The probability of the individual chromosomes is calculated as:

$$p_s = eval(\upsilon_s)/F \tag{6.21}$$

Sorting solutions by fitness: After the evaluation of the chromosomes with respect to their fitness, they are sorted in increasing order by their fitness. Then, the fittest chromosome is the uppermost. The reason for sorting them is due to the far easier handling of the population when e.g. declaring taboo to overwrite a certain amount of best solutions, defined by MBS.

Cumulative Probability The cumulative probability is the sum of the probability of the individual chromosomes p_s, from the first chromosome until the one at position s in the list of chromosomes. The cumulative probability is calculated as:

$$q_s = \sum_{j=1}^{s} p_j \tag{6.22}$$

Storage of best solution: Apart from the initial population, the actual population does not necessarily contain a copy of the so far best feasible solution. If the actual population is the initial population, the best feasible solution is to be stored. Otherwise, the currently best solution gets overwritten if the actual population provides an enhanced (”fitter”) and feasible solution.

Deletion of duplicate solutions: In order to avoid occupation of chromosomes with duplicate samples, duplicate chromosomes are deleted. This is achieved after the chromosomes have been sorted by their fitness. The chromosomes which result in the same fitness are then evaluated regarding their genes’ sequence. If two chromosomes are identical, one is deleted from the actual population and replaced by a new permutation chromosome, as in the initial population. An additional chromosome is not yet evaluated and therefore is assigned with the weakest fitness obtained so far, hence is not likely to reproduce. The added chromosome describes a permutation sequence, hence, is a feasible solution.

6.5.2 Inheritance

Inheritance is the first of the genetic operators and describes the procedure of creating an exact copy of a complete chromosome, see figure 6.3. The parameter MBS defines a number of best solutions which will not be overwritten. Apart from this, there exists the parameter p_{b} which is the percentage of chromosomes, within the number of best solutions (MBS) that will be deleted. The removed chromosome is replaced by a permutation chromosome which gets assigned the weakest fitness obtained in the current population and the list of chromosomes is sorted again. Consequently the deleted chromosome will not be used any further for reproduction.

The parameter MBS ensures that promising chromosomes will not be extinct, on the other hand and in order to not remain in a local minima, the parameter p_{b} removes chromosomes with a certain percentage.

Inheritance also takes place within the remaining population due to the fact that not all chromosomes are overwritten by the succeeding genetic operators like crossover and mutation.

6.5.3 Crossover

The second genetic operator is the crossover with the parameters $p_{\text{c-I}}$ and $p_{\text{c-II}}$, indicating the percentage of chromosomes which will be used for crossover, see figure 6.3.

Selection of chromosomes

The selection process is based on spinning a roulette wheel $(R - 2 \cdot MBS)$-times. Reason for this reduced number is that if p_{c} is set to 1.0, a total of 100% crossover has to be performed and only chromosomes can be overwritten which are not part of the "best solutions", determined by MBS or are part of the second copy of the best solutions which only get applied the genetic operator mutation.

Each time a single chromosome is selected in the following way:

- For each chromosome in the population a random (float) number r is generated in the range [0..1].

- If $r < q_1$ then the first chromosome (υ_1) is select; otherwise the s-th chromosome υ_s $(2 \leq s \leq (R - 2 \cdot MBS))$ is selected such that $q_{s-1} < r \leq q_s$.

Clearly, some chromosomes are selected more than once. This results from the fact that more promising chromosomes are used for reproduction more often, the average stay even, and the worst are not reproduced and most probably will die off within one of the next generations.

The parameter p_{C} defines the overall probability of crossover. The expected number of chromosomes which undergo the crossover is $p_{\text{C}} \cdot (R - 2 \cdot MBS)$. From the previously selected chromosomes some have to be discarded in order to meet the total number of chromosomes for the crossover:

- A random (float) number r is generated in the range [0..1].
- If $r < p_{\text{C}}$ the given chromosome is selected for crossover.

In the next step the selected chromosomes are mated depending on the type of crossover and the proportion with which the operator crossover-I and the operator crossover-II is performed, defined by $p_{\text{C-I}}$ and $p_{\text{C-II}}$, respectively. The sum of $p_{\text{C-I}}$ and $p_{\text{C-II}}$ is smaller or equal to 1.0:

$$(p_{\text{C-I}} + p_{\text{C-II}}) \leq 1.0 \tag{6.23}$$

Crossover-I

The genetic operator crossover-I performs for a pair of two chromosomes a one-point interchange of genes. The proportion with which the operator crossover-I is performed is defined by $p_{\text{C-I}}$. The crossover point *pos* is a random integer number in the range $\{1..(N \cdot L) - 1\}$ and determines the position in the chromosome where the crossover is performed.

In the case in which $pos = r \cdot N$, the crossover is a simple procedure. Figure 6.4 shows an example of a pair of chromosomes υ_1 and υ_2, representing seven jobs which are to be processed on a flowshop with two resequencing possibilities. They are crossed over at $pos = 7$ and the genes appearing after *pos* are crossed over completely.

The case in which $pos \neq r \cdot N$, the crossover procedure is more complicated. In figure 6.5 the point of crossover falls within $\Pi_2^{'}$ and the jobs which are already sequenced can not be

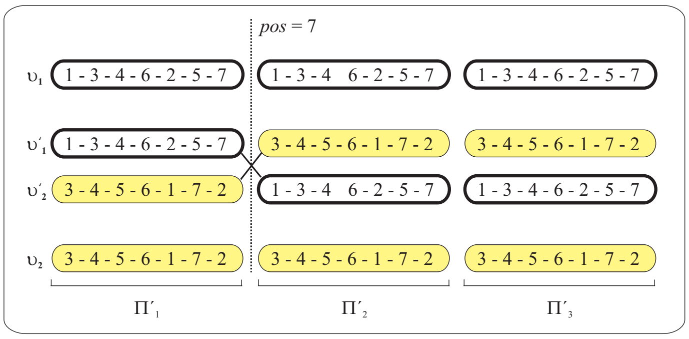


Figure 6.4: Genetic operator **Crossover-I:** for the simple case in which $pos = r \cdot N$: Chromosomes υ_1 and υ_2 undergo the crossover operation at position $pos = 7$, between two fractions of the sequences. After the crossover point, the chromosomes are completely crossed over. Chromosomes υ_1' and υ_2' are the resulting chromosomes.

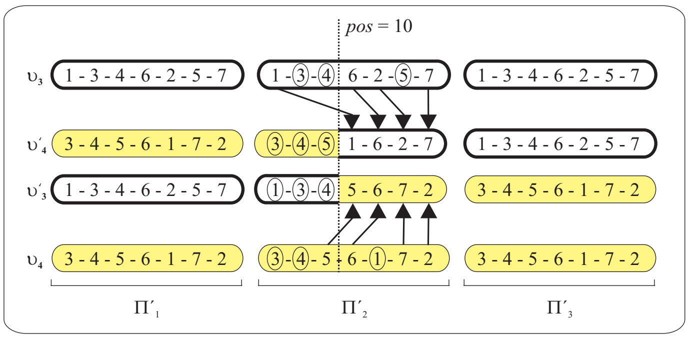


Figure 6.5: Genetic operator **Crossover-I** for the case in which $pos \neq r \cdot N$: Chromosomes υ_3 and υ_4 undergo the crossover operation at position $pos = 10$. Due to the fact that in υ_3' the jobs 1, 3, and 4 are already sequenced by Π_2', the entire sequence Π_2' of υ_4 has to be considered. Chromosomes υ_3' and υ_4' are the resulting chromosomes.

repeated. Chromosome υ_3' already has the jobs $(1, 3, 4)$ included in Π_2'. Starting from the first job of Π_2' in chromosome υ_4 and skipping the already sequenced jobs, the remaining jobs are $(5, 6, 7, 2)$. Π_2' of the new chromosome υ_3' results in $(1, 3, 4, 5, 6, 7, 2)$. Respectively for chromosome υ_4' which has already sequenced the jobs $(3, 4, 5)$. The sequence is completed with the jobs $(1, 6, 2, 7)$.

Crossover-II

The second crossover operator considers the interchange of a partial sequences of genes from one chromosome to another. The proportion with which the operator crossover-II is performed, is defined by $p_{\text{c-II}}$. The two crossover points, pos_1 and pos_2, are two distinct random integer numbers in the range $\{1..(N \cdot L)-1\}$, and pos_1 being the smaller of the two. These are used to extract a partial sequence from a chromosome and to replace it with one of another chromosome, considering that no two jobs are repeated within a sequence Π_i'.

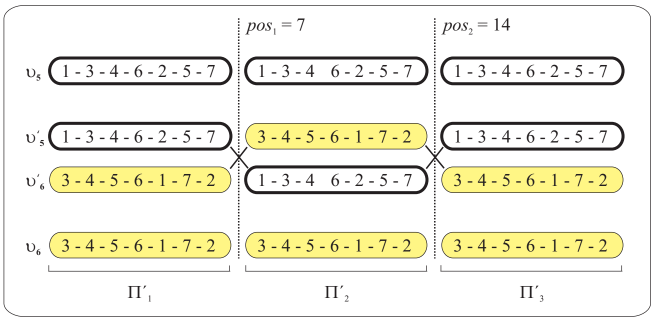


Figure 6.6: Genetic operator **Crossover-II** for the simple case in which $pos_1, pos_2 = r \cdot N$: Chromosomes υ_5 and υ_6 are subject to crossover, using the fraction of genes within the two crossover points $pos_1 = 7$ and $pos_1 = 14$. Chromosomes υ_5' and υ_6' are the resulting chromosomes.

In the case in which $pos_1, pos_2 = r \cdot N$, the crossover is a relatively simple procedure. Figure 6.6 shows a pair of chromosomes υ_5 and υ_6 which are crossed over at $pos_1 = 7$ and $pos_2 = 14$, the sequence of genes appearing between pos_1 and pos_2 is crossed over completely.

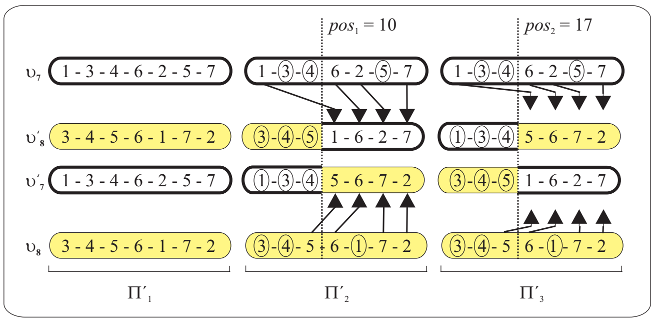


Figure 6.7: Genetic operator **Crossover-II** for the case in which $pos_1, pos_2 \neq r \cdot N$: Chromosomes υ_7 and υ_8 undergo the crossover operation between $pos_1 = 10$ and $pos_1 = 17$. Due to the fact that in υ_7' the jobs 1, 3, and 4 are already sequenced by Π_2', the entire sequence Π_2' of υ_8 has to be considered. Chromosomes υ_7' and υ_8' are the resulting chromosomes.

The case in which $pos_1, pos_2 \neq r \cdot N$, the crossover procedure is more complicated. In figure 6.7 the points of crossover fall within Π_2' and Π_3'. The jobs which are already sequenced can not be repeated. Considering first the sequence Π_2', the chromosome υ_7' already has the jobs $(1, 3, 4)$ included. Starting from the first job of Π_2' in chromosome υ_8 and skipping the already sequenced jobs, the remaining jobs are $(5, 6, 7, 2)$. Continuing with Π_3', the chromosome υ_7' already has the jobs $(3, 4, 5)$ included. Starting from the first job of Π_3' in chromosome υ_7 and skipping the already sequenced jobs, the remaining jobs are $(1, 6, 2, 7)$. Π_2' of the new chromosome υ_7' results in $(1, 3, 4, 5, 6, 7, 2)$ and Π_3' results in $(3, 4, 5, 1, 6, 2, 7)$. Respectively for chromosome υ_8' which results in $(3, 4, 5, 1, 6, 2, 7)$ and $(1, 3, 4, 5, 6, 7, 2)$.

Degression

The genetic operators Crossover-I and Crossover-II result in considerable changes in the generation of the chromosomes. This behavior is desirable during the first generations but may be suppressed once the Genetic Algorithm has further evolved. In order to achieve a degression of $p_{\text{c-I}}$ and $p_{\text{c-II}}$, the parameter g_{deg} is introduced.

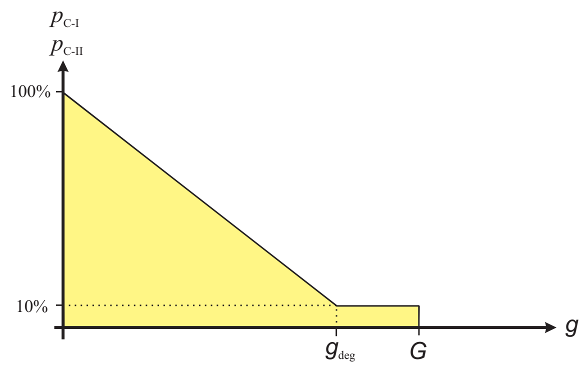


Figure 6.8: Linear degression of $p_{\text{c-I}}$ and $p_{\text{c-II}}$ for $g_{\text{deg}} \geq G$. The parameter g_{deg} defines the point at which $p_{\text{c-I}}$ and $p_{\text{c-II}}$ have decreased to 10% of their original values.

This parameter defines the number of generations at which $p_{\text{c-I}}$ and $p_{\text{c-II}}$ have linearly decreased to 10% of their original values and are maintained at this percentage until G is reached (see figure 6.8).

Figure 6.9 shows the case in which g_{deg} is smaller than G. The degression slope of the probabilities $p_{\text{c-I}}$ and $p_{\text{c-II}}$ is flatter and does not decrease to 10% of the original value before G is reached.

Overwrite-position

The new chromosomes, generated by the genetic operators Crossover-I and Crossover-II, can overwrite any of the chromosomes which are not part of the best solutions or the second copy of best solutions which are reserved for the genetic operators of mutation.

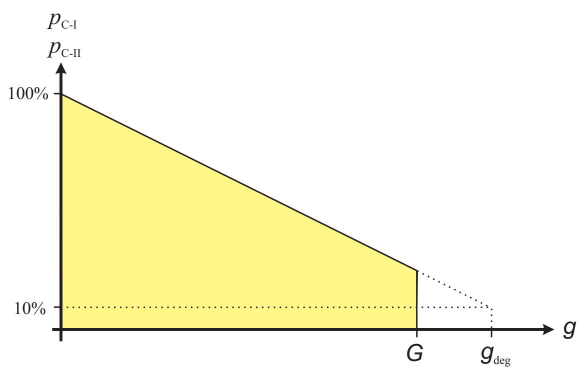


Figure 6.9: Linear degression of $p_{\text{C-I}}$ and $p_{\text{C-II}}$ for $g_{\text{deg}} < G$. The degression slope is flatter and does not decrease to 10% before the Genetic Algorithm terminates.

Two possible strategies are implemented in the Genetic Algorithm:

- **Last position**
  The chromosomes of the new population are overwriting the chromosomes of the previous population in increasing order of their fitness, starting with the weakest.
- **Random position**
  The chromosomes of the new population are randomly overwriting the chromosomes of the previous population. Clearly one position is not overwritten two times in order to ensure the correct number of new chromosomes.

In both cases neither the best solutions nor the second copy of the best solutions, which are reserved for mutation only, are overwritten.

6.5.4 Mutation

The genetic operator mutation is applied to single chromosomes and is performed on a gene-by-gene basis. Every gene in all chromosomes in the whole population, except the ones which are part of the best solutions, defined by MBS, has an equal chance to undergo mutation, see figure 6.3. Furthermore, there exists an exact copy of the best solutions which only gets applied the genetic operators for mutation and are untouched by the crossover.

Two random integer numbers pos_1 and pos_2 in the range $\{1..(N \cdot L) - 1\}$ within the same sequence Π'_r define the genes which undergo the mutation. Three possible mutations are considered:

- Mutation-I (forward) removes a job from the current position pos_1 and inserts it at another position pos_2, with $(pos_1 < pos_2)$.
- Mutation-I (backward) is similar to mutation-I (forward), but $(pos_1 > pos_2)$.
- Mutation-II interchanges two jobs which are found at position pos_1 and pos_2.

Selection of chromosomes

The way in which a chromosome is selected for mutation is basically the same as in the case of crossover. A roulette wheel is spined $(R - MBS)$-times.

Each time a single chromosome is selected in the following way:

- For each chromosome in the population a random (float) number r is generated in the range [0..1].
- If $r < q_1$ then the first chromosome(v_1) is select; otherwise the s-th chromosome v_s $(2 \leq s \leq (R - MBS))$ is selected such that $q_{s-1} < r \leq q_s$.

The parameter p_{m} defines the probability of mutation. The expected number of chromosomes which undergo the crossover is $p_{\mathrm{m}} \cdot (R - MBS)$.

- A random (float) number r is generated in the range [0..1].
- If $r < p_{\mathrm{m}}$ the given chromosome is selected for mutation.

In the next step the selected chromosomes are subject to one of the genetic operators mutation-I (forward), mutation-I (backward) or mutation-II. The proportion with which one of the operators is performed is defined by $p_{\text{m-I(f)}}$, $p_{\text{m-I(b)}}$ and $p_{\text{m-II}}$, respectively.

Mutation-I (forward)

The genetic operator mutation-I specifies the operation of relocating a single job at position pos_1 to another position pos_2 within the same sequence $\Pi_r^{'}$. The downstream-relocation of a job, with $(pos_1 < pos_2)$, is achieved with the genetic operator forward mutation-I, see Figure 6.10.

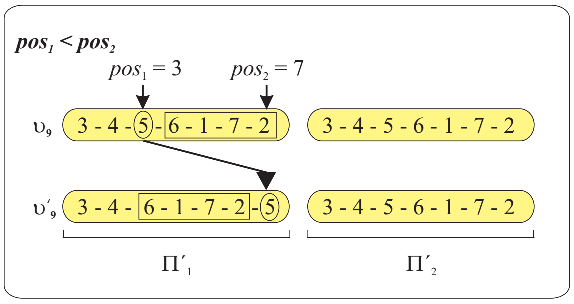


Figure 6.10: Genetic operator **Mutation-I (forward)**: Forward mutation-I is the case in which a job at position pos_1 is taken off the line, stored temporarily in the respective resequencing buffer and reinserted to the line at position pos_2.

The equivalent action is to take off the job, determined by pos_1, store it temporarily in the respective resequencing buffer, let bypass the succeeding jobs until the position determined by pos_2 and reinsert the removed job at pos_2. The proportion with which the forward mutation-I is performed is defined by $p_{\text{m-I(f)}}$.

Mutation-I (backward)

As before, the job at position pos_1 gets relocated to position pos_2 in the same sequence Π_r', but $(pos_1 > pos_2)$, see figure 6.11.

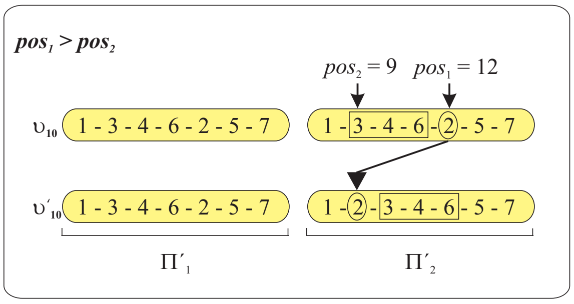


Figure 6.11: Genetic Operator **Mutation-I (backward)**: Backward mutation-I is far more complicated since all jobs between pos_2 and pos_1 have to be stored temporarily in the respective resequencing buffer.

The realization in the line is significantly more complicated, caused by the fact that a resequencing buffer can only remove a job from the line and reinsert it at a later position in the sequence. The way in which a job can be relocated upstream is to take off all jobs which are located between position pos_2 and pos_1, letting pass the job at position pos_2 and reinserting the temporarily stored jobs to the line in the same order as they had been removed. The only exception is when $(pos_2 - pos_1 = 1)$, here the upstream relocation of a job from pos_2 to pos_1 is equivalent to the downstream relocation of a job from pos_1 to pos_2. The proportion with which the backward mutation-I is performed is defined by $p_{\text{m-I(b)}}$.

Mutation-II

The genetic operator mutation-II is an interchange operation of two jobs at position pos_2 and pos_1 within the same sequence Π'_r. The proportion with which the operator mutation-II is performed, is defined by $p_{\text{m-II}}$.

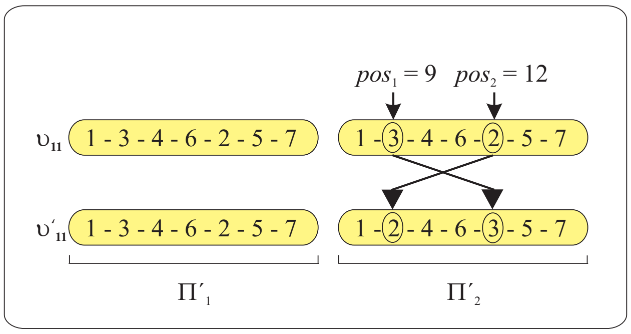


Figure 6.12: Genetic Operator **Mutation-II**: Two jobs at position pos_1 and pos_2 in the sequence of a single chromosome are interchanged.

Figure 6.12 shows an example of the genetic operator mutation-II. The two jobs at position $pos_1 = 9$ and $pos_2 = 12$ of the same sequence Π'_2 in the same chromosome interchange their position.

The degree of difficulty for the realization of this operation in the line is similar to the difficulty of the backward mutation-I, also for the simple case in which $(pos_2 - pos_2 = 1)$. This is due to the fact that all jobs between pos_1 and pos_2 have to be taken off the line temporarily in order to achieve the desired resequencing.

6.6 Condition for termination of Genetic Algorithm

Apart from the number of generations (G), which terminates the Genetic Algorithm when the maximum number of generations is reached, the algorithm can use a second condition which may result in an early termination, the Convergence-control. In the case in which the algorithm has not improved the so far best solution for the last 300 generations, it is assumed that the the algorithm has converged and further search will be time-consuming. The Algorithm is interrupted.

- Maximum number of generations, G, reached
- Last 300 generations have not improved the so far best solution (Convergence-control)

6.7 Cascade of Genetic Algorithm

In the case in which the considered flowshop permits resequencing between stations, the size of the chromosomes increases by a factor, linear to the number of stations with access to resequencing buffers (see section 6.2.3). In order to further enhance the Genetic Algorithm, the proceeding is divided in two steps:

1. **Permutation sequences (Step-1):**
   - Resequencing does not take place.
   - Chromosome-size is only depending on the number of jobs, N.
   - Last generation forms the initial generation of step 2. The worst solution is replaced by the so far best solution.
2. **Non-permutation sequences (Step-2):**
   - Resequencing takes place between determined stations.
   - Chromosome-size is depending on the number of jobs to be processed, N, and the number of stations with access to resequencing buffers, L.

In a first step the Genetic Algorithm ignores the possibility of resequencing jobs within the production line and therefore considers only permutation sequences as possible solutions. The chromosome is reduced to the number of jobs N. The last generation leads to a preliminary best generation, the worst solution of this generation is replaced by the so far best solution.

In a second step the Genetic Algorithm takes into account the resequencing possibilities provided by stations with access to resequencing buffers. The chromosome-size is depending on the number of jobs to be processed, N, and the number of stations with access to resequencing buffers, L. The initial solutions which form the initial generation consist of permutation sequences. The permutation sequences $\Pi_1^{'}$ (for υ_1 to υ_R) are copied to each of the sequences $\Pi_1^{'}$ to $\Pi_{L+1}^{'}$ (for υ_1 to υ_R).

6.8 Preliminary analysis of parameters

Following to the extended analysis and adjustment of the parameters which are used by the Genetic Algorithm, a preliminary analysis is performed. This analysis intents to obtain a better understanding of the behavior of the parameters, as for example to estimate how likely it is for the following adjustment of the parameters to remain in local minima.

Name	Description	Values
N	Number of Jobs	10
M	Number of Stations	5
L	Number of Resequencing possibilities	2
R	Population size	100
G	Number of Generations	1000
MBS	Number of best solutions to maintain	5
p_{b}	Probability to eliminate best solutions	0.1
g_{deg}	Degression of pC-I and pC-II	N
FP	Penalty for non-feasible solution	10
Seed	Seed for generation of random number	47, 57, 67, 77

Table 6.1: Fixed parameters used for the preliminary analysis of the Genetic Algorithm.

The parameters which are not considered in this preliminary analysis and therefore are set to preset values, are listed in table 6.1. Then, three analysis are performed, each varying two of the five considered parameters in the range of 0.0 to 1.0. The considered parameters are the two parameters for crossover and the three parameters for mutation. Each experiment is repeated with a set of four seeds, then the average of the objective function is used to plot the results. It has to be taken into account that only two parameters are varied at a time and the intention is not to determine the values of the individual parameters. Thereafter, an analysis is performed which highlights the variability of the solutions of the Genetic Algorithm.

Crossover-I versus Crossover-II

The first analysis varies the two parameters crossover-I and crossover-II, see table 6.2. Due to the fact that the sum of the two may not be bigger than 1.0, only values are considered which do not exceed this limit. The three remaining parameters for mutation are set to low values.

Name	Description	Values
$p_{\text{c-I}}$	Crossover-I	$0.0, 0.1, ..., 1.0$[1]
$p_{\text{c-II}}$	Crossover-II	$0.0, 0.1, ..., 1.0$[1]
$p_{\text{m-I(f)}}$	Mutation-I (forward)	0.1
$p_{\text{m-I(b)}}$	Mutation-I (backward)	0.3
$p_{\text{m-II}}$	Mutation-II	0.1

Table 6.2: Range of variable parameters $p_{\text{c-I}}$ and $p_{\text{c-II}}$ used for the preliminary analysis of the Genetic Algorithm.

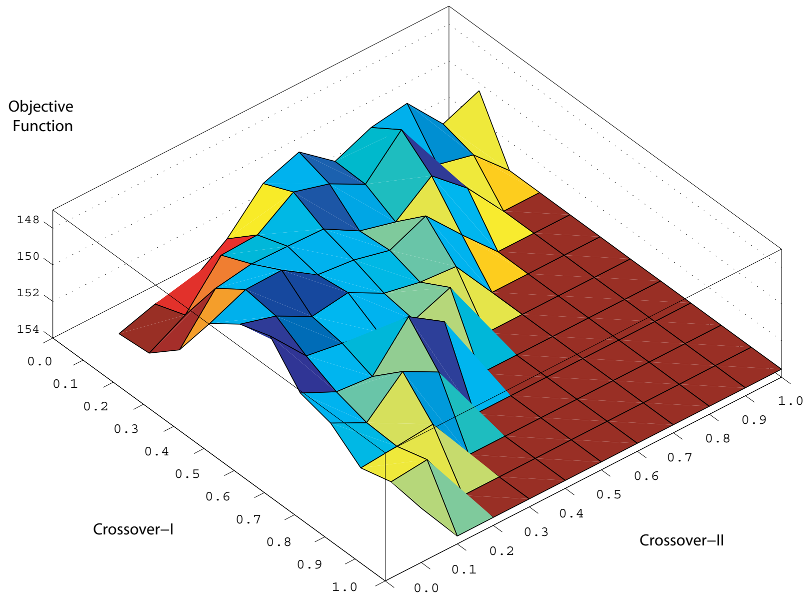


Figure 6.13: Study of the influence of $p_{\text{c-I}}$, the probability for crossover-I and $p_{\text{c-II}}$, the probability for crossover-II, on the objective function.

Figure 6.13 shows the influence of the two parameters for crossover on the resulting value of the objective function. In order to plot the mesh of solution points, the combinations of the two parameters which result in the sum being larger than 1.0, are complemented with the maximum value of all considered points. This results in a plane surface for the half of the plotted mesh without any relevance for the analysis. Details on the values are listed in the appendix in table A.10.

[1]$p_{\text{c-I}} + p_{\text{c-II}}$ can not exceed 1.0, therefore any combination which exceeds 1.0 is not considered.

The plotted mesh shows that a fairly continuous surface is formed with a wide valley. The interpretation of this valley leads to the conclusion that neither setting the considered parameters at the same time to small values, nor setting them such that their sum is close to 1.0 is advisable.

In the following analysis the parameters for crossover-I and crossover-II are set to 0.3 and 0.2, respectively. This combination is a combination which results to be within the valley with its sum being relatively small in order to not have any dominating effect.

Mutation-I (f) versus Mutation-I (b)

The second analysis varies the two parameters mutation-I (f) and mutation-I (b), see table 6.3. The two parameters do not face any limitation. The parameter for mutation-II is set to a low value.

Name	Description	Values
$p_{\text{c-I}}$	Crossover-I	0.3
$p_{\text{c-II}}$	Crossover-II	0.2
$p_{\text{m-I(f)}}$	Mutation-I (forward)	$0.0, 0.1, ..., 1.0$
$p_{\text{m-I(b)}}$	Mutation-I (backward)	$0.0, 0.1, ..., 1.0$
$p_{\text{m-II}}$	Mutation-II	0.1

Table 6.3: Range of variable parameters $p_{\text{m-I(f)}}$ and $p_{\text{m-I(b)}}$ used for the preliminary analysis of the Genetic Algorithm.

Figure 6.14 shows the influence of the two parameters for mutation-I on the resulting value of the objective function. The plotted mesh shows that a continuous surface is formed which is improving when both parameters for mutation-I are decreased. Except for the case in which both of them become too small. Details on the values are listed in the appendix in table A.11.

In the following analysis the parameter for mutation-I (b) is set to 0.1 in order to not have any dominating effect.

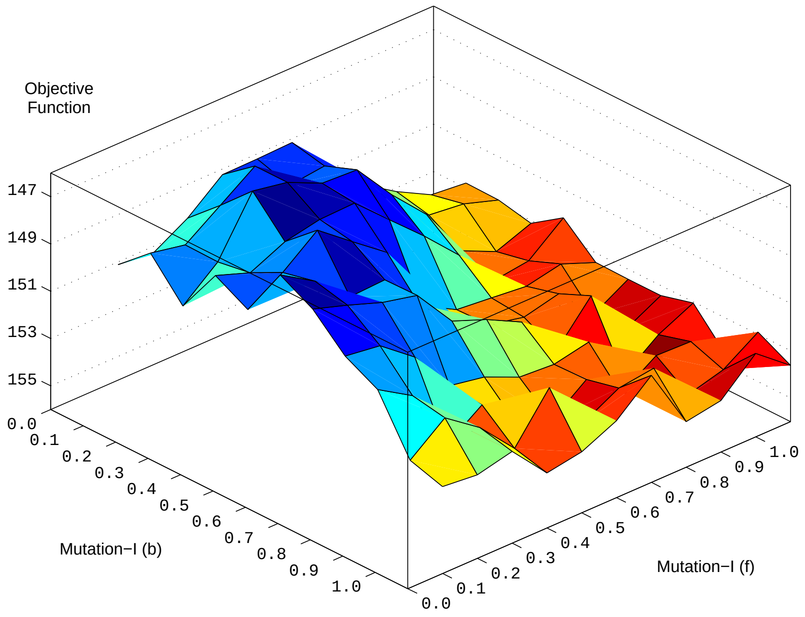


Figure 6.14: Study of the influence of $p_{\text{m-I(f)}}$, the probability for mutation-I (f) and $p_{\text{m-I(b)}}$, the probability for mutation-I (b), on the objective function.

Mutation-I (f) versus Mutation-II

The third analysis varies the two parameters mutation-I (f) and mutation-II, see table 6.4. The two parameters do not face any limitation.

Name	Description	Values
$p_{\text{c-I}}$	Crossover-I	0.3
$p_{\text{c-II}}$	Crossover-II	0.2
$p_{\text{m-I(f)}}$	Mutation-I (forward)	$0.0, 0.1, ..., 1.0$
$p_{\text{m-I(b)}}$	Mutation-I (backward)	0.1
$p_{\text{m-II}}$	Mutation-II	$0.0, 0.1, ..., 1.0$

Table 6.4: Range of variable parameters $p_{\text{m-II}}$ and $p_{\text{m-I(f)}}$ used for the preliminary analysis of the Genetic Algorithm.

Figure 6.15 shows the influence of the two parameters mutation-I (f) and mutation-II on the resulting value of the objective function. Similar to the previous case, the plotted mesh shows that a continuous surface is formed which is improving, when both parameters are decreased. Details on the values are listed in the appendix in table A.12.

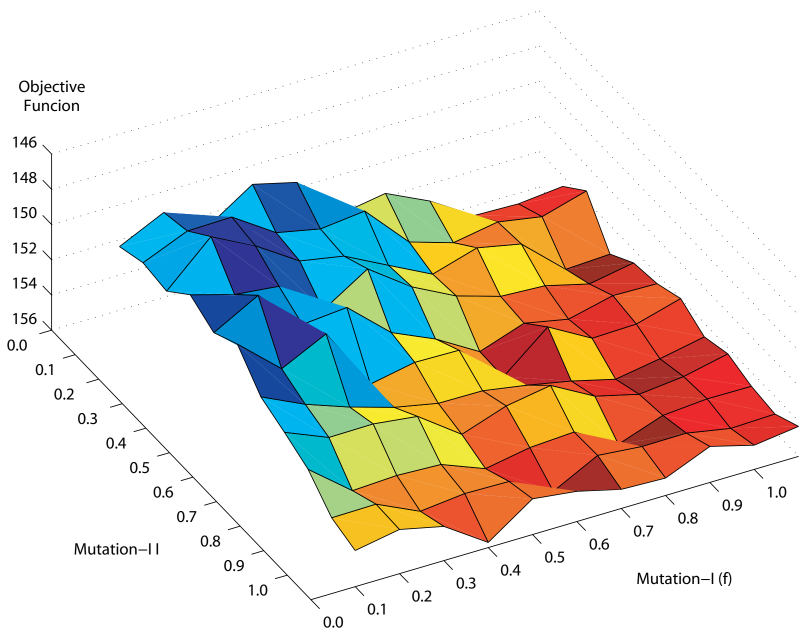


Figure 6.15: Study of the influence of $p_{\text{m-II}}$, the probability for mutation-II and $p_{\text{m-I(f)}}$, the probability for mutation-I (f), on the objective function.

Variability of solutions

The Genetic Algorithm is based on random numbers, giving the algorithm its strength. However, this also leads to the disadvantage that the algorithm on the other hand is not very predictable and in order to determine promising parameters, useful for a multitude of problems, the analysis of the parameters is to be repeated with various different seeds.

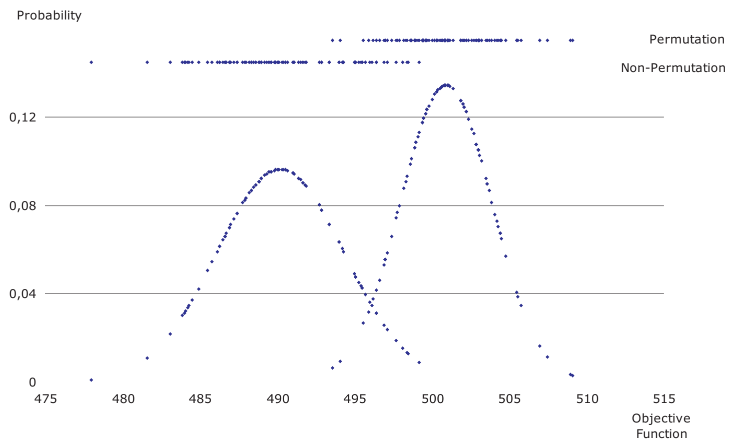


Figure 6.16: Variability of the solutions of the Genetic Algorithm. The same problem is solved 100 times, each time a different seed is used, for the permutation case as well as for the non-permutation case.

Figure 6.16 shows the variability of the Genetic Algorithm with respect to the obtained solutions. The same problem is solved by the Genetic Algorithm 100 times, each time a different seed is used, for the permutation case as well as for the non-permutation case. The solutions, permitting non-permutation sequences, in general result in better solutions with a larger deviation. The average value of the objective function of the particular example, used in figure 6.16, is 500.93 with a standard deviation of 2.96 for the permutation case and 490.21 with a standard deviation of 4.13 for the non-permutation case.

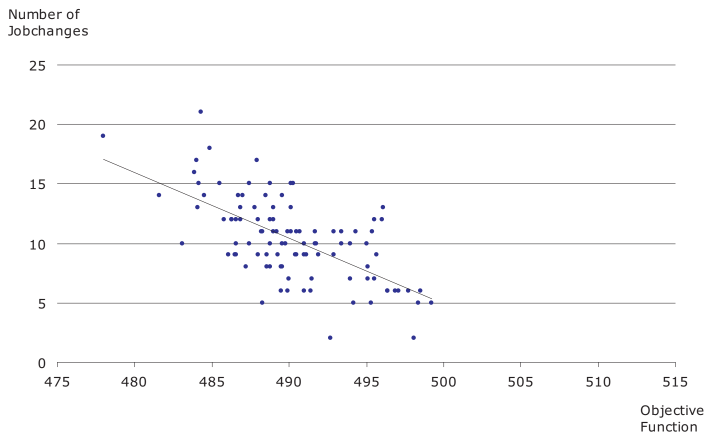


Figure 6.17: Dependency of the number of job changes on the objective function for the non-permutation case. Better solutions tend to have a larger number of jobs which have to be taken off the line in order to be resequenced.

Analyzing the obtained data with respect to the number of job changes with more detail, figure 6.17 shows that in order to obtain better solutions, the number of jobs, which have to be taken off the line for resequencing, tends to be higher.

Conclusions

Concluding the preliminary analysis of the parameters, it can be resumed that the variation of the five considered parameters for crossover and mutation result in a continuous solution space with smooth transitions. It can be outlined that it is expected that the adjustment of these five parameters will lead to values that can be found in the lower or in the mid part of their range. Furthermore, due to the variability of the results, it is necessary to repeat the individual experiments with several seeds.

6.9 Analysis and adjustment of parameters

Previous to the use of the Genetic Algorithm, the parameters which result in a multitude of possible combinations, have to be analyzed and adjusted. In what follows, an analysis proceeding is proposed in order to adjust the parameters which lead to rapid convergence and encouraging solutions. Thereafter, in chapter 7, these parameter adjustments are used for the experimental phase. The four phases of the analysis and adjustment of the parameters are accomplished as follows:

Rough adjustment

On account of the numerous parameters of the presented Genetic Algorithm, it is necessary to roughly adjust the parameters by using a predetermined number of discrete values for each parameter, see table 6.5. The total number of possible sets of parameters is 3456 and only one seed is used.

Name	Description	Range	Values
R	Number of parents	Small, Medium	70, 100
G	Number of generations	Small, Medium	1000, 10000
MBS	Number of best solutions to maintain	Medium	5
p_{b}	Probability eliminate best solutions	Small, Medium	0.1, 0.4
$p_{\text{c-I}}$	Probability of Crossover-I	Small, Medium, Large	0.1, 0.3, 0.6[2]
$p_{\text{c-II}}$	Probability of Crossover-II	Small, Medium, Large	0.1, 0.3, 0.6[2]
g_{deg}	Degression in % of G	Not applied, Medium	N, 100%
$p_{\text{m-I(f)}}$	Probability of Mutation-I (forward)	Small, Medium, Large	0.1, 0.3, 0.5
$p_{\text{m-I(b)}}$	Probability of Mutation-I (backward)	Small, Medium, Large	0.1, 0.3, 0.5
$p_{\text{m-II}}$	Probability of Mutation-II	Small, Medium, Large	0.1, 0.3, 0.5
FP	Penalty for non-feasible solution	Small, Medium	10, 50
OWP	Overwrite-position for crossover	Last, Random	Last, Rand

Table 6.5: Discrete values of the parameters for the rough adjustment of the Genetic Algorithm.

In order to adjust the parameters of the Genetic Algorithm in a robust manner, the above mentioned set of parameters is applied to a series of differently sized problems, first for instances which consider permutation sequences and then for instances which consider non-

[2] $p_{\text{c-I}} + p_{\text{c-II}}$ can not exceed 1.0, therefore the combination $p_{\text{c-I}} = 0.6$ and $p_{\text{c-II}} = 0.6$ is not considered.

permutation sequences. The size of the problems is determined by N, the number of jobs to be sequenced, M, the number of stations and in the non-permutation case by L, the number of resequencing possibilities. Table 6.6 and 6.7 show the considered variations in problem size which lead to 6 instances with differently sized problems for the permutation case and 8 instances for the non-permutation case.

Name	Description	Range	Values
N	Number of jobs	Small, Medium	10, 20
M-L	Number of stations - number of stations with access to resequencing buffers	Small - Null,	5-0
		Medium - Null	10-0
		Large - Null	20-0

Table 6.6: Variable parameters of the flowshop used for the analysis and adjustment of the Genetic Algorithm (Permutation case).

The sets of parameters are then summarized and the 300 most promising ones out of the 3456, which show good performance are used for further analysis. Two different sets of most promising parameters are determined, one for the permutation case , based on 6 instances, and one for the non-permutation case, based on 8 instances.

Name	Description	Range	Values
N	Number of jobs	Small, Medium	10, 20
M-L	Number of stations - number of stations with access to resequencing buffers	Small - Null,	5-2
		Small - Small	10-2
		Medium - Null	20-2
		Medium - Small	20-5

Table 6.7: Variable parameters of the floswhop used for the analysis and adjustment of the Genetic Algorithm (Non-permutation case).

Repeatability

The use of only one seed in the rough adjustment requires to determine amongst the set of 300 promising parameter which parameter set achieves good results for a multitude of seeds (one for the permutation case and one for the non-permutation case). The fact that a parameter set achieves good results for different seeds indicates that the same parameter set also performs well for different flowshops. The 300 promising sets of parameters are taken and verified with 16 different seeds on the same differently sized problems as mentioned before (6 instances for permutation case and 8 instances for the non-permutation case).

Clustering of parameter sets

Once the two sets of 300 promising parameters are examined with respect to repeatability, two sets are determined for further adjustment. The Matlab toolbox from Balasko et al., 2005, is used to group the parameter sets into clusters and use the one which performs best for the permutation case and one for the non-permutation case. Due to the fact that up to now predetermined values for the parameters were used, a fine adjustment succeeds.

Name	Description	Non-Perm	Perm
R	Number of parents	100	100
G	Number of generations	1000	1000
MBS	Number of best solutions to maintain	5	5
p_{b}	Probability eliminate best solutions	0.1	0.4
$p_{\text{c-I}}$	Probability of Crossover-I	0.3	0.5
$p_{\text{c-II}}$	Probability of Crossover-II	0.6	0.35
g_{deg}	Degression in % of G	N	N
$p_{\text{m-I(f)}}$	Probability of Mutation-I (forward)	0.25	0.1
$p_{\text{m-I(b)}}$	Probability of Mutation-I (backward)	0.25	0.1
$p_{\text{m-II}}$	Probability of Mutation-II	0.25	0.1
FP	Penalty for non-feasible solution	10	10
OWP	Overwrite-position for crossover	Rand	Last

Table 6.8: Adjustment of variable parameters of the Genetic Algorithm obtained by the extended analysis.

Fine adjustment

The two sets of parameters, obtained in the previous section, are part of a set of parameters whose discrete values were determined as a result of the reasonable but fixed scattering of their values within a certain range for each parameter. In this section a fine adjustment is performed which allows to further improve the parameter sets by using a small neighborhood search. The parameters for crossover and mutation are subject to an adjustment of two times 0.05 for the previously determined sets of parameters and are revised with 16 seeds. This is accomplished for both parameter sets, one on the 6 instances for the permutation case and the other one on the 8 instances for the non-permutation case, respectively, already used for the repeatability. As a result, two different sets of parameters are obtained and are listed in table 6.8.

6.10 Time performance analysis

The performance of the Genetic Algorithm, with respect to computational time, is depending on the chromosome size, the number of parents (R) and the number of generations (G). The chromosome size in turn is depending on the number of jobs (N) to be sequenced, the number of stations (M) and the number of resequencing possibilities (L).

Two cases are examined. First, the algorithm stops when the maximum number of generations, is reached (Abort-No). Second, as described in section 6.6, the algorithm stops in the case in which the algorithm has not improved the so far best solution for the last 300 generations (Abort-300). The first case is not depending on if the algorithm converges and gives a initial solution for the second case.

The performance study, with respect to computational time, does not give any measure of quality of the obtained solutions. The interest lies in understanding the behavior of the Genetic Algorithm with respect to time consumption.

The graphical representations in this section show results which were obtained by experimentation with discrete data points. For a better visual presentation the individual data points of a series of performance tests are depicted as a connected curve. As a matter of fact, these connected curves may give an indication on the behavior between two successive data points but can not be interpreted as actual data points.

Number of stations (M)

Increasing the number of stations results in an increased computational time, caused by the fact that the calculation of the schedule is longer when the jobs are to be sequenced on more stations.

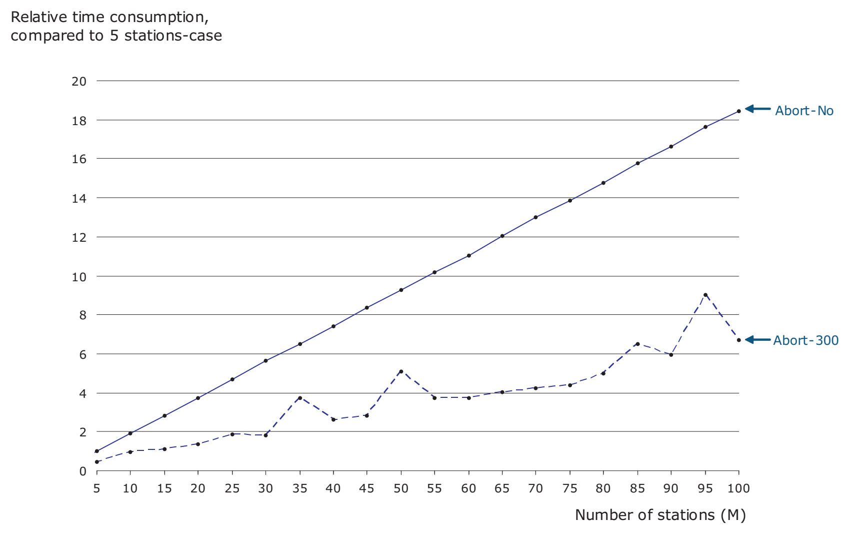


Figure 6.18: Linear dependency of the time consumption, compared to a five station case, over the number of stations (M) to be considered. The time with convergence-control (Abort-300) in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Figure 6.18 shows the **linear** dependency of the computational time, compared to a five-stations case, over the number of stations to be considered. The time with convergence-control (Abort-300), see section 6.6, in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Number of resequencing possibilities (L)

Increasing the number of resequencing possibilities does not result in a remarkable increase of computational time. The increased computational time is caused by the fact that the execution of the genetic operators, e.g. crossover-I, is raised when the reduced chromosome size augmented. The effort for the schedule calculation is basically the same.

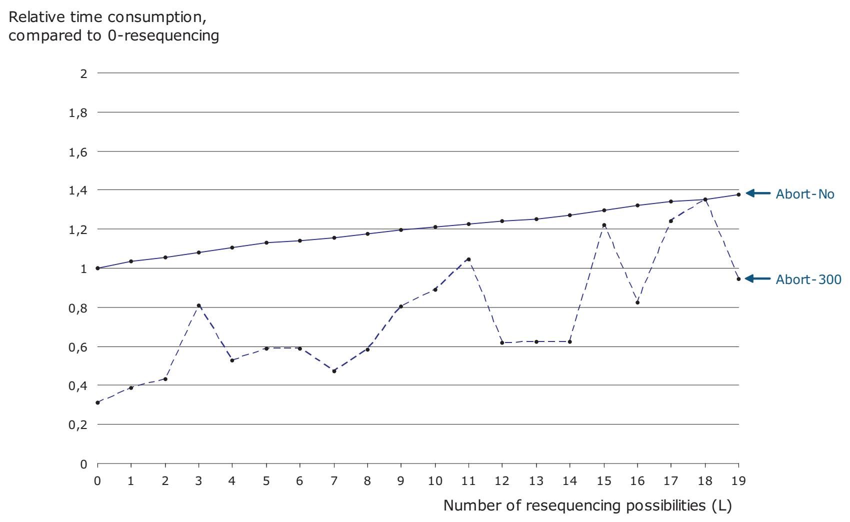


Figure 6.19: Dependency of the time consumption, with respect to the case with zero resequencing possibilities, over the number of resequencing possibilities (L) to be considered. The time with convergence-control (Abort-300) in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Figure 6.19 shows the dependency of the computational time, with respect to the case with zero resequencing possibilities, over the number of resequencing possibilities to be considered. The time with convergence-control (Abort-300), see section 6.6, in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Number of jobs (N)

Increasing the number of jobs results in an increased computational time, caused by the fact that the calculation of the schedule is longer when additional jobs are to be sequenced.

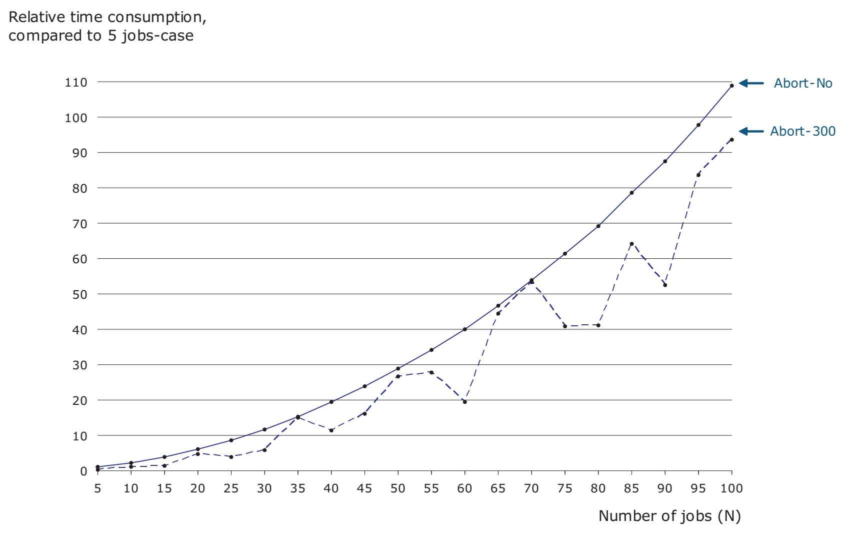


Figure 6.20: Non-linear dependency of the time consumption, compared to a five job case, over the number of jobs (N) to be sequenced. The time with convergence-control (Abort-300) in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Figure 6.20 shows the **non-linear** dependency of the computational time, compared to a five job case, over the number of jobs to be sequenced. The time with convergence-control (Abort-300), see section 6.6, in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Number of parents (R)

Increasing the number of parents results in an increased computational time, caused by the increased number of schedules to be calculated in each generation.

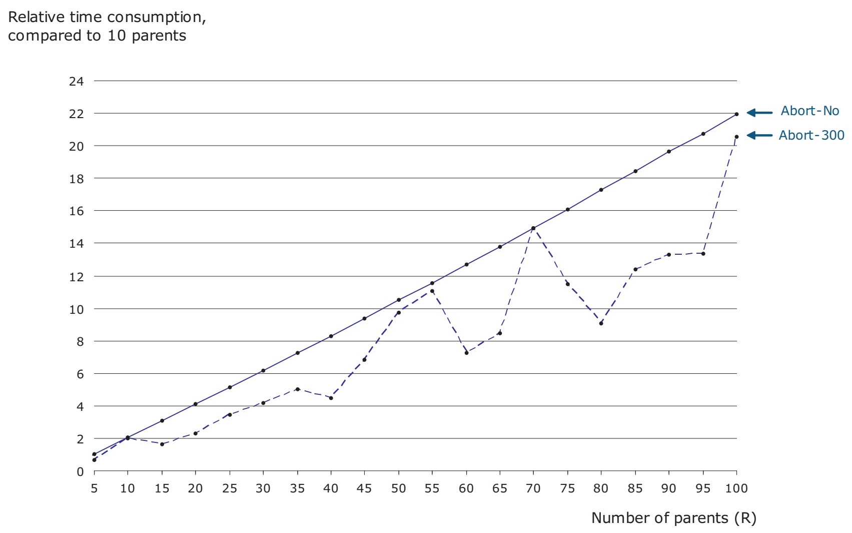


Figure 6.21: Performance of the Genetic Algorithm, depending on the number of parents (R) to be sequenced, compared to the case with five parentes. The time with convergence-control (Abort-300) in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Figure 6.21 shows the **linear** dependency of the computational time, compared to the case with five parents, over the number of considered parents. The time with convergence-control (Abort-300), see section 6.6, in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Number of generations (G)

Increasing the number of generations results in an increased computational time, caused by the fact that more schedules have to be calculated when more generations are considered, similar to the case in which the number of parents is augmented.

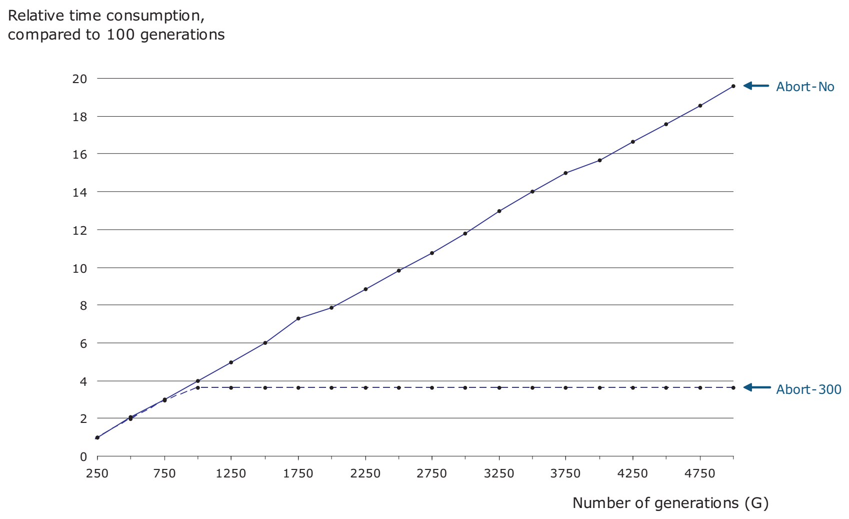


Figure 6.22: Performance of the Genetic Algorithm, depending on the number of generations (G) to be sequenced, compared to the case with 250 generations. The time with convergence-control (Abort-300) in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No).

Figure 6.22 shows the **linear** dependency of the computational time, compared to the case with 250 generations, over the number of considered generations. The time with convergence-control (Abort-300), see section 6.6, in any case is equal or inferior, compared to the case where the algorithm calculates all generations (Abort-No). The straight, horizontal line in the case of the convergence-control is caused by the fact that even if the number of generations grows, the algorithm stops the first time a block of 300 consecutive generations are calculated without having improved the so far best solution.

6.11 Conclusions

The Genetic Algorithm, presented in this chapter, uses several genetic operators, including the most common ones for inheritance, crossover and mutation. The Algorithm furthermore uses two cascades, first considering permutation sequences and then non-permutation sequences.

The preliminary analysis (section 6.8) with approximately 2200 runs on the one hand highlights that the variation of the parameters for crossover and mutation results in a continuous solution space with smooth transitions. On the other hand the random character of the GA is shown by analyzing the variability of the solutions. In particular, in the non-permutation case the solutions have a larger standard deviation, compared to the permutation case.

Then, the values of the genetic operators were determined (section 6.9) following the four steps *Rough adjustment*, *Repeatability*, *Clustering* and *Fine adjustment*. A total of 3456 parameter sets were considered and applied to 14 differently sized problems, including cases with and without access to resequencing buffers.

Finally, a time performance analysis (section 6.10) was accomplished which shows the dependency of the GA on the variables as the number of parents, the number of generations or variables which are related to the size of the chromosomes. The number of jobs appeared to have the most severe influence due to its non linear dependency.

7 Experimentation Genetic Algorithm

The experimentation, with the consideration of concrete values, on the one hand aims to show the proper functionality of the proposed method of resolution, and on the other hand analyzes the advantage of resequencing with constrained buffers from different points of view. Taking into account the complexity which is inherent in the determination of the adequate distribution of constrained resequencing buffers, the primer objective is to demonstrate the effectiveness and the performance improvement of a production line. The quantitative analysis with respect to the benefit of resequencing is depending on the concrete data. All of the experiments were run on a 3.00 GHz Pentium 4 with 512 MB of Random Access Memory (RAM). The realized experimentation studies the performance of the presented Genetic Algorithm, considering the following aspects:

- Influence of setup considerations
- Intermediate versus centralized buffer location
- Number of buffer places
- Difference in physical size of jobs
- Distribution of data
  1. Linear distribution
  2. Spreading
  3. Normal distribution
- Semi dynamic case: No sequencing before the first station
- Paced flowshop production line
- Benchmark comparison
  1. Watson et al., 2002
  2. Brucker et al., 2003

The graphical representations in this chapter show results which were obtained by experimentation with discrete data points. For a better visual presentation the individual data points of a series of performance tests are depicted as a connected curve. As a matter of fact, these connected curves may give an indication on the behavior between two successive data points but can not be interpreted as actual data points.

7.1 General frame

For the experimental study of the Genetic Algorithm, in order to study its performance, a flowshop which consists of 5 stations is considered. The range of the production time is [0...20] such that for some jobs zero-processing time at some stations exists, for the setup cost [2...8] and for the setup time [1...5]. This particular composition is not followed in section 7.2, which discusses setup considerations, section 7.6, which discuses the distribution of data, and section 7.9, which performs benchmark comparisons.

The objective function is the weighted sum of the makespan (factor of 1.0) and the setup cost (factor of 0.3), where the setup time is not concerned with a weight but is indirectly included in the calculation of the makespan. The experiments are repeated eight times with eight different seeds and the **average values** are considered, except in section 7.9 which performs the benchmark comparison, where the **best out of the eight** experiments is compared.

7.2 Setup considerations

In the case in which the operator needs to change the setup of the station, in order to process the next job, the change of the setup may result in an additional production cost or an additional time, necessary to realize the change. When setup is considered, the improvement due to resequencing is expected to be more obvious.

As a concrete example, a flowshop with 30 stations is considered with access to resequencing buffer places after station 6 and after station 20. Two cases are considered: the intermediate and the centralized case. In the intermediate case (I22), each resequencing buffer consists

Case	Intermediate		Centralized	
Setup	*SC*	*ST*	*SC*	*ST*
1	No	No	No	No
2	No	Yes	No	Yes
3	Yes	No	Yes	No
4	Yes	Yes	Yes	Yes

Table 7.1: Four scenarios for the setup considerations, including the setup cost (SC) and setup time (ST), for the intermediate and the centralized case.

of 2 buffer places. In the centralized case (C3), the same two stations have access to the centralized resequencing buffer with 3 resequencing buffer places. Table 7.1 shows the four possible scenarios for the intermediate and the centralized case.

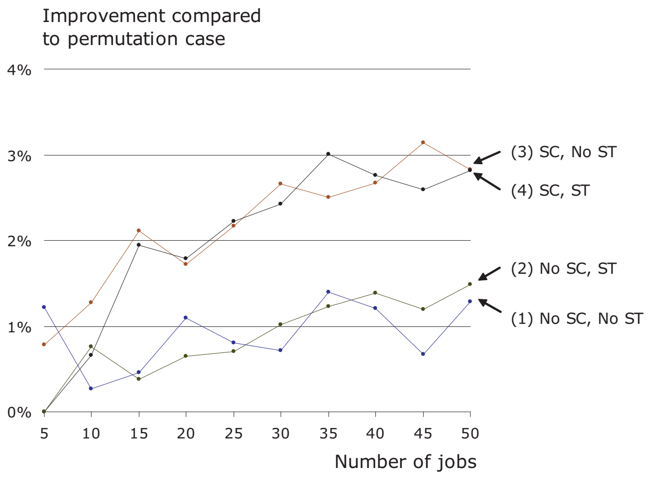


Figure 7.1: Consideration of setup cost (SC) and setup time (ST) for the intermediate case (I22), with station 6 and station 20 having access to a resequencing buffer, each provided with two buffer places.

Figure 7.1 and figure 7.2 show the improvement of the objective function, for the intermediate case and the centralized case, respectively, with respect to the case without resequencing buffers (permutation flowshop). The number of jobs to be sequenced is increased from 5 jobs to 50 jobs with increments of 5 jobs.

Apparently, within the four scenarios, the two with setup cost obtain better results than the two without setup cost. This observation is not a general observation and depends on γ, the weight of the setup cost in the objective function, and also on the setup costs itself, the setup times and the processing times. The important conclusion which can be taken from the study of the setup considerations is the fact that the appearance of setup, specially the appearance of the two at a time, results in a further improvement.

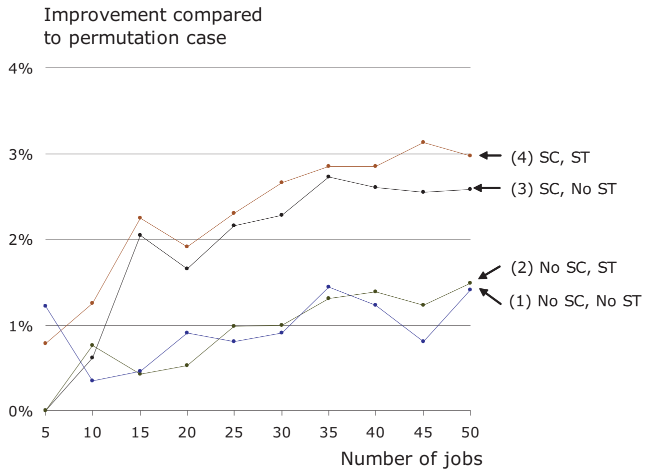


Figure 7.2: Consideration of setup cost (SC) and setup time (ST) for the centralized case (C3), with station 6 and station 20 having access to a single resequencing buffer, provided with three buffer places.

7.3 Intermediate versus centralized buffer location

Replacing the intermediate resequencing buffer places with centralized resequencing buffer places has two benefits. On the one hand, for the case of the same number of buffer places, the objective function of the final solution is expected to be at most as good. This is caused by the fact that in some instances of time, all buffer places of a certain intermediate resequencing buffer may be occupied and do not allow an additional job to be removed from the line, buffer places from other intermediate resequencing buffers are not accessible. Whereas, in the case of a centralized buffer, blocking only appears when all buffer places are occupied. On the other hand, the number of buffer places may be reduced in order to obtain values of the objective function, similar to the case of the intermediate resequencing buffer. Depending on the number of buffer places which are reduced, this reduction in area is relevant.

In what follows, the 5 station flowshop, explained in section 7.1, is considered. Furthermore, after station 2 and after station 3 access to resequencing buffers exist and three cases are compared: one centralized resequencing buffer with four buffer places (C4); two intermediate resequencing buffers, each providing two buffer places (I22); one centralized resequencing buffer with three buffer places (C3).

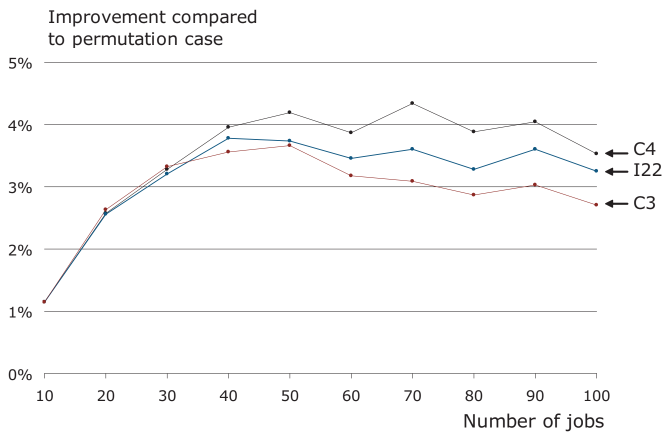


Figure 7.3: Comparison of three cases: one centralized resequencing buffer with four buffer places (C4); two intermediate resequencing buffers, each providing two buffer places (I22); one centralized resequencing buffer with three buffer places (C3).

Then, figure 7.3 shows the improvement of the objective function with respect to the case without resequencing buffers (permutation flowshop). The number of jobs to be sequenced is increased from 10 jobs to 100 jobs with increments of 10 jobs. It can be seen that the case C4 improves the solutions, compared to case I22, while the same number of buffer places is maintained. Whereas a reduction in the number of buffer places by 1, as in case C3, obviously results in solutions which are inferior, compared to case C4. The reduction in necessary buffer area however may payoff the negative effect on the objective function.

The corresponding number of jobs to be taken off the line in order to be resequenced is shown in figure 7.4. Case C4 has the highest number of jobs to be taken off the line in order to be resequenced, followed by case I22 and finally case C3.

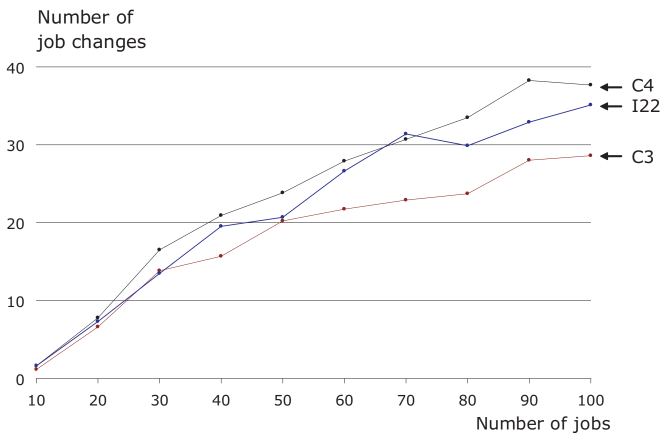


Figure 7.4: Number of jobchanges for the three cases: one centralized resequencing buffer with four buffer places (C4); two intermediate resequencing buffers, each providing two buffer places (I22); one centralized resequencing buffer with three buffer places (C3).

7.4 Number of buffer places

The increase in the number of buffer places makes the limitations less strict and as already seen in the previous case, solutions are expected to improve.

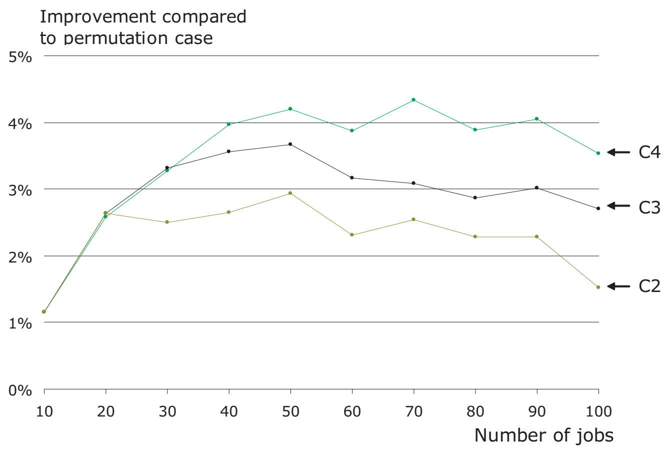


Figure 7.5: Variation of the number of buffer places for the centralized case. Station 2 and station 3 have access to a centralized buffer with two buffer places (C2), three buffer places (C3) and four buffer places (C4).

Figure 7.5 shows the improvement with respect the permutation case, without resequencing, for the centralized case. Jobs leaving the second or the third station have access to the centralized buffer, provided with 2, 3 or 4 buffer places.

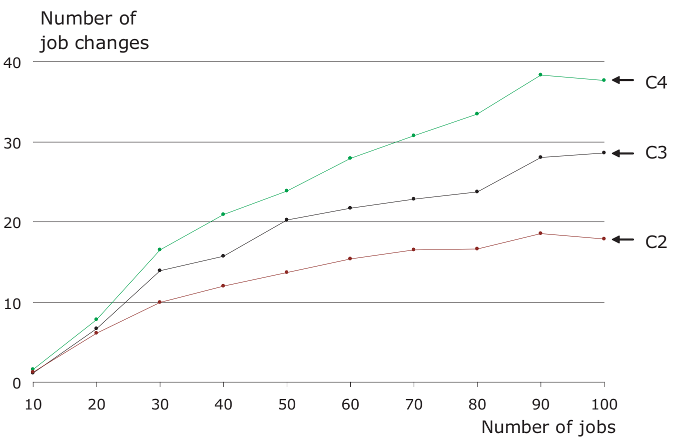


Figure 7.6: Number of job changes for the centralized case. Station 2 and station 3 have access to a centralized buffer with two buffer places (C2), three buffer places (C3) and four buffer places (C4).

Figure 7.6 shows the corresponding number of jobs which are to be taken off the line in order to resequence. Providing more buffer places results in better solutions together with an elevated number of job changes.

7.5 Difference in physical size of jobs

Introducing limitations on the physical size of the buffer places on the one hand restricts possible solutions, but on the other hand minimizes the necessary buffer area. This limitation arises, for example, in a chemical production. The arrangement of two tanks which are located off the line, accessible after a certain station, equals an intermediate resequencing buffer with two buffer places. With tank capacities of 50 and 100 liters, a client order of 80 liters can be stored only in the larger of the two tanks which is capable of storing this volume. Whereas, a client order of 50 liters can be stored in either of the tanks. A close look at the local conditions may amortize an increase in the objective function compared to a reduction of investment with respect to tank size and gained area.

As a concrete example, three differently sized buffer places (large, medium, small) are available and the ratio of jobs is $\frac{3}{10}$ large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small. As before, the second and the third station have access to the resequencing buffers. Table 7.2 shows the allocation of the buffer places to the buffers, considering four scenarios for the intermediate case and the same four scenarios for the centralized case. "300" represents 3 large, 0 medium and 0 small buffer places. In the intermediate case the first buffer is provided with 1 and the second buffer with 2 large buffer places. In the centralized case the same two stations have access to a single centralized buffer, containing the three buffer places.

Case	Intermediate			Centralized		
Size	l	m	s	l	m	s
(300)	1/2	0/0	0/0	3	0	0
(111)	0/1	1/0	0/1	1	1	1
(102)	0/1	0/0	1/1	1	0	2
(012)	0/0	0/1	1/1	0	1	2

Table 7.2: Allocation of the buffer places to the buffers. In the intermediate case the allocation is done to two different buffers.

Figure 7.7 shows the influence of the limitation of the physical size for the intermediate case. The variation of the size of the buffer places towards smaller buffer places on the one hand decreases the benefit achieved by the possibility of resequencing jobs. On the other hand, it may amortize when taking into account the reduction of investment with respect to tank size and gained area.

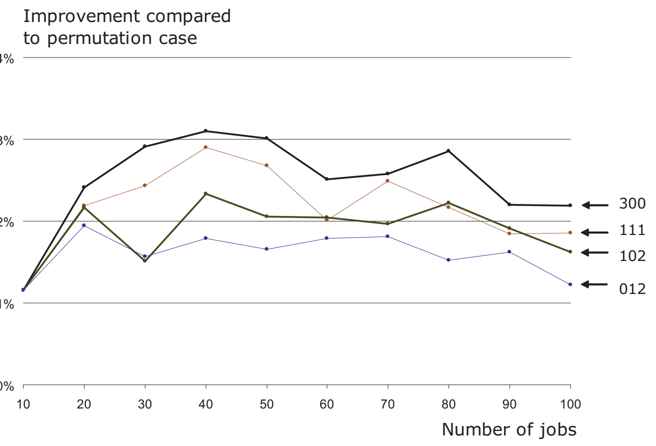


Figure 7.7: Influence of the variation of the physical size of the buffer places for the intermediate case. "102" represents 1 large, 0 medium and 2 small buffer places. The buffer places are divided into two buffers, each with access from a designated station. The ratio of jobs is $\frac{3}{10}$ large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small.

Figure 7.8 shows the influence of the limitation of the physical size for the centralized case. The performance is similar to the previous case. The variation of the size of the buffer places towards smaller buffer places decreases the benefit, achieved by the possibility of resequencing jobs, but may amortize when taking into account the reduction of investment with respect to tank size and gained area.

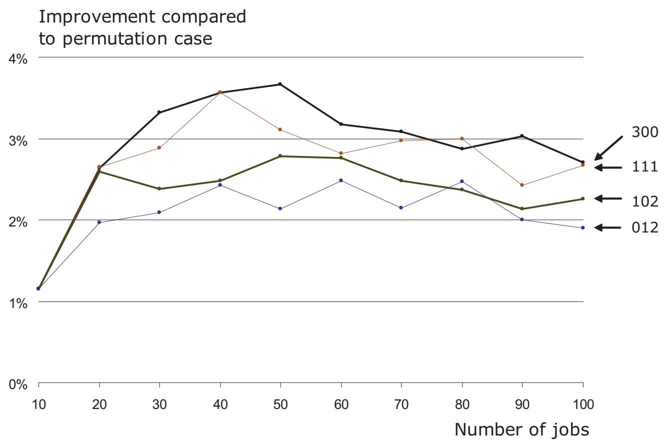


Figure 7.8: Influence of the variation of the physical size of the buffer places for the centralized case. "102" represents 1 large, 0 medium and 2 small buffer places. The same two stations have simultaneously access to the buffer, containing all three buffer places. The ratio of jobs is $\frac{3}{10}$ large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small.

As already seen in section 7.3, the comparison of the intermediate versus the centralized buffer location, the centralized case in general results in better solutions.

7.6 Distribution of data

The input data, if not homogenously distributed, can be subject to various distributions. The influence of the range of the data is studied and two different types of distributions are applied.

7.6.1 Linear distribution

In the simple case in which no setup cost or setup time is occurring and the processing times are the same for all jobs at all stations, that is $P_{i,j} = P$, no effort is necessary to sequence or resequence the jobs. In contrast, when the processing times differ, rise the necessity of sequencing and resequencing jobs. Furthermore, it is expected that specially the benefit of resequencing jobs within the line is more meaningful.

In what follows, a flowshop with 20 stations with setup cost and setup time, as described in section 7.1, is considered. The processing times range from x_{min} to x_{max}, the mean value

Description	Range		Mean
Case	x_{min}	x_{max}	μ
1	16	24	20
2	12	28	20
3	8	32	20

Table 7.3: Range of the processing time for the three contemplated cases with the same mean value.

being μ. In the case of the intermediate resequencing buffers (I22), after station 7 and station 14 access to a resequencing buffer exist with 2 buffer places each. In the centralized case (C3), a single resequencing buffer with three buffer places is accessible after the same two stations. Table 7.3 gives the upper and lower values of the three contemplated cases, the mean value for the three cases is the same.

Figure 7.9 and 7.10 shows the result for the number of jobs ranging from 10 jobs to 90 jobs with increments of 20 jobs for the intermediate case and the centralize case. As expected, the increase of the range of the processing times leads to an increase of the benefit of resequencing. The results obtained for the centralized case with three buffer places are similar to the intermediate case with four buffer places.

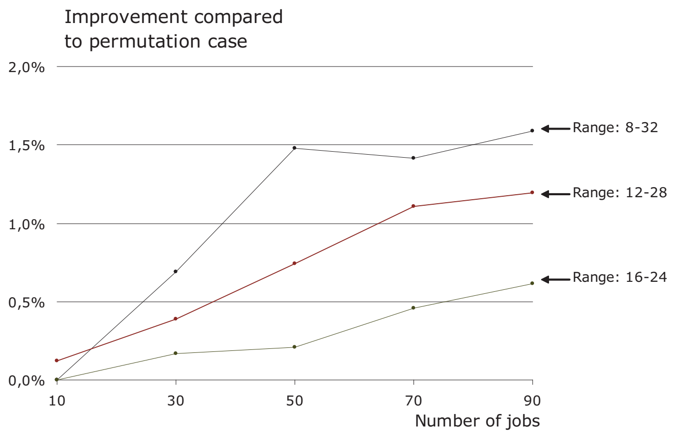


Figure 7.9: Result of the range analysis for the intermediate case (I22). The increase of the range of the production times $P_{i,j}$ leads to an increase of the benefit of resequencing.

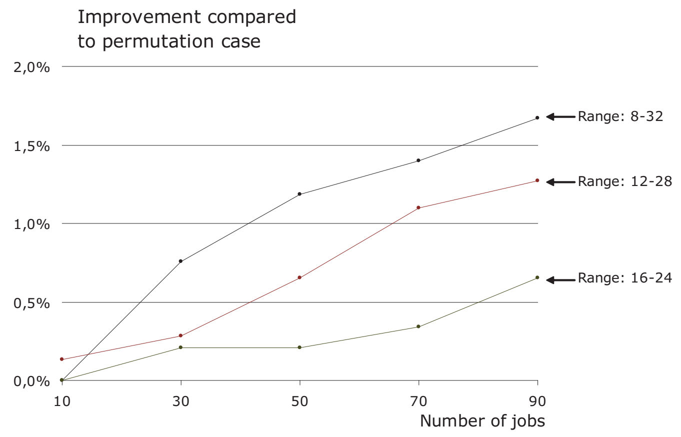


Figure 7.10: Result of the range analysis for the centralized case (C3). The increase of the range of the production times $P_{i,j}$ leads to a similar increase of the benefit of resequencing as in the intermediate case with four buffer places.

7.6.2 Spreading

In order to spread out the homogenous distribution, a sinusoidal function is applied. Function $\Psi(x)$ is applied to the random value x, which is part of a homogenous distribution with the lower bound x_{min} and the upper bound x_{max} (see equation 7.1).

$$\Psi(x) \quad = \quad \kappa_{min} + (\kappa_{max} - \kappa_{min}) \cdot (1 - cos(\frac{x - x_{min}}{x_{max} - x_{min}} \cdot \pi)) \tag{7.1}$$

The function value of $\Psi(x)$ in turn is limited to the lower bound κ_{min} and the upper bound κ_{max}, respectively. Setting the two bounds of the input and the two bounds of the output to the same value, that is

$$\begin{aligned} \kappa_{min} &= x_{min} \\ \kappa_{max} &= x_{max}, \end{aligned}$$

the homogeneous distribution, maintaining its range. As can be seen in figure 7.14, the homogenous distribution is D^0, applying the function $\Psi(x)$ one time results in D^1, two time results in D^2, and so on. μ is the mean value which is the same for all cases.

In order to study the effect of the distribution of the generated data, three different sets of data are generated for a flowshop with five stations, each defined by processing times, setup costs and setup times. The first set, Dispersion 0, contains the homogenously distributed

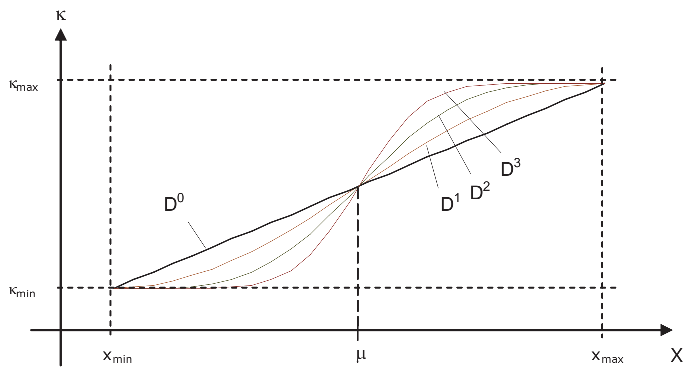


Figure 7.11: Applying the function $\Psi(x)$ one time to the homogenous distribution D^0 leads to the more spread out data D^1. Then D^2 and D^3 results from applying the function two and three time, respectively. The mean value μ does not change.

data. The second set, Dispersion 1, has the function $\Psi(x)$ applied one time and the third set, Dispersion 3, three times.

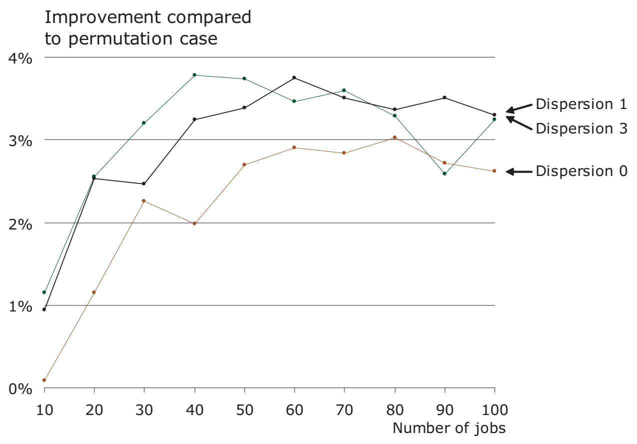


Figure 7.12: Analysis of spreading out the input data for the intermediate case (I22). Spreading out the data results in a higher effectiveness of resequencing.

The flowshop has access to resequencing buffers after station 2 and station 3. In the intermediate case (I22), each buffer is provided with two buffer places and in the centralized case (C3), the two stations have access to the same centralized buffer with three buffer places.

Figure 7.12 and figure 7.13 show that spreading out the values (D^1 and D^3) results in a higher efficiency of the resequencing. The centralized case with three buffer places appears as effective as the intermediate case with four buffer places.

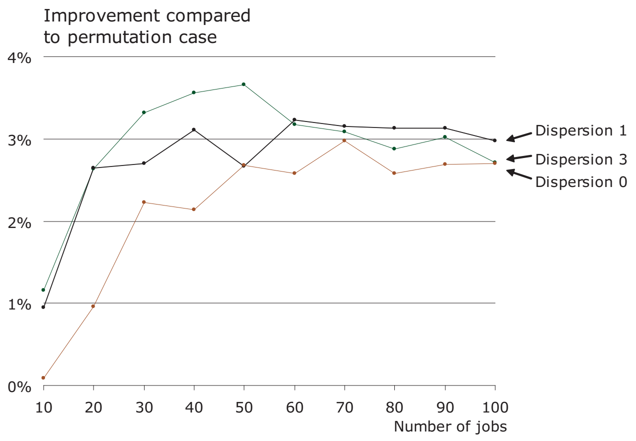


Figure 7.13: Analysis of spreading out the input data for the centralized case (C3). The centralized case with three buffer places appears to be as effective as the intermediate case with four buffer places.

7.6.3 Normal distribution

The normal distribution, also called the gaussian distribution, can be defined by the mean value μ and the standard deviation σ. The standard deviation is a measure of the degree of dispersion of the data from the mean value and 68.27% of the area under the curve is within the standard deviation of the mean.

$$\varphi(x) \quad = \quad \frac{1}{\sqrt{2\pi}\cdot\sigma}\cdot e^{-0.5(\frac{x-\mu}{\sigma})^2} \tag{7.2}$$

Equation 7.2 gives the probability density function $\varphi(x)$ of the normal distribution of x with the mean value μ and the standard deviation σ which is displayed in figure 7.14.

As already shown in section 7.6.1, the range of the input data directly influences the effectiveness of resequencing. As a concrete example of the normal distribution, the 5 station flowshop with the same mean value μ for the processing times is used as in the case of spreading in section 7.6.2.

The distribution of the data follows the probability density function $\phi(x)$ with a standard deviation of 10%, 20%, 30% and 40% of the mean value of the processing times. Each of the four experiments is performed for the intermediate case (I22) where station 2 and station 3 have access to a resequencing buffer, each providing two buffer places, and then for the centralized case (C3) with three resequencing buffer places, accessible from station 2 and station 3 simultaneously.

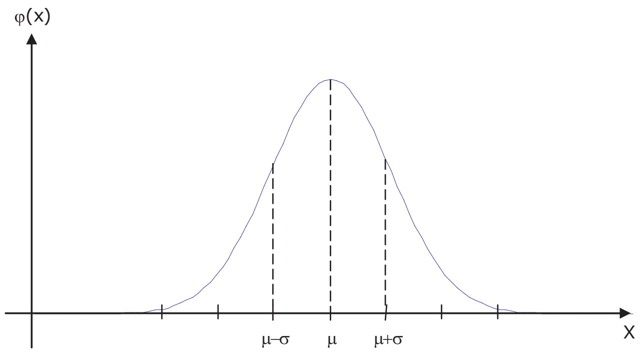


Figure 7.14: Probability density function $\varphi(x)$ with the mean value μ and the standard deviation σ. The percentage of 68,27% of the values lie between $(\mu - \sigma)$ and $(\mu + \sigma)$.

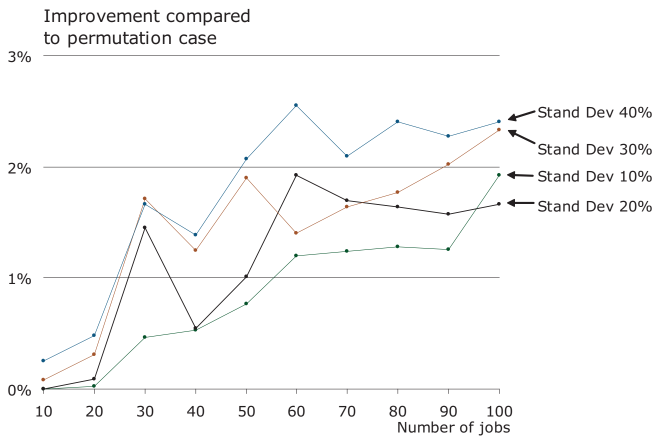


Figure 7.15: Analysis of an increasing standard deviation for the intermediate case (I22). The standard deviation of 20% is calculated as 20% of the mean value.

Figure 7.15 and 7.16 show the influence of the standard deviation for the processing times. The number of jobs to be sequenced ranges from 10 jobs to 100 jobs with increments of 10 jobs. An increase of the standard deviation results in a larger benefit when jobs are resequenced within the flowshop.

7.7 Semi dynamic demand

In the case in which a production line is separated into two parts, and the first station of the second part is not provided with a large ASRS buffer (Automated Storage and Retrieval System), no mayor resequencing of jobs is possible before the jobs enter the second part of the production line.

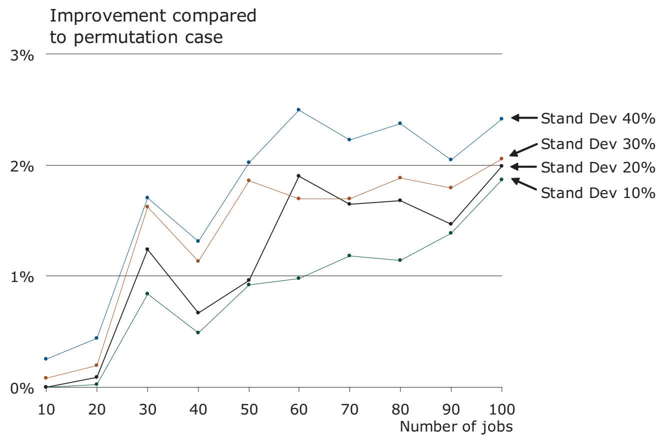


Figure 7.16: Analysis of an increasing standard deviation for the centralized case (C3).

Compared to the previous experimentation, this case is the more dynamic case. Considering, for the above mentioned case, the second part of the production line as a separate production line, only resequencing takes place, without the possibility of sequencing the jobs before they enter the line.

The case in which before the first station a resequencing buffer with only a few buffer places is available, no major sequencing of jobs can takes place either. This particular case can be solved with the Genetic Algorithm by introducing a dummy station before the first station which has zero processing times for all jobs. This, however, only enables limited resequencing possibilities, compared to an ASRS buffer.

Due to the fact that the jobs can not be sequenced before entering the production line, the job sequence is supposed to be of worse quality with respect to the objective function. On the other hand, that is why the resequencing without prior sequencing is expected to result in a higher benefit.

As a concrete example, the 5 station flowshop is considered with access to resequencing buffers after station 2 and after station 3. The sequence of jobs before the first station is generated randomly and the option of sequencing the jobs before the first station is disabled in the Genetic Algorithm.

For the intermediate case, three scenarios are regarded: in I22, both buffers are provided with two buffer places each; in I20, only station 2 has access to a resequencing buffer with two places; and in I02, only station 3 has access to a resequencing buffer with two places.

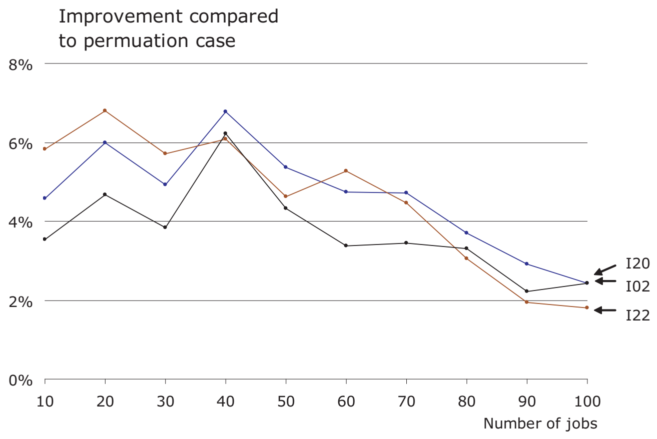


Figure 7.17: Semi dynamic demand without sequencing before the first station for the intermediate case. In I22, both buffers are provided with two buffer places each; in I20, only station 2 has access to a resequencing buffer with two places; and in I02, only station 3 has access to a resequencing buffer with two places.

For the centralized case, station 2 and station 3 have access to a centralized buffer with two buffer places (C2), three buffer places (C3) and four buffer places (C4).

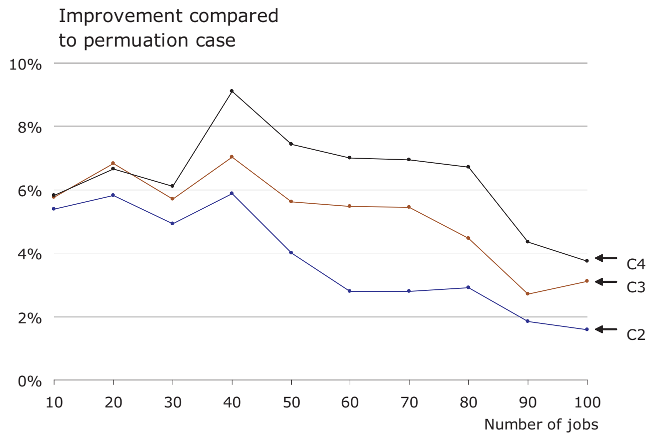


Figure 7.18: Semi dynamic demand without sequencing before the first station for the centralized case. Station 2 and station 3 have access to a centralized buffer with two buffer places (C2), three buffer places (C3) and four buffer places (C4).

Without sequencing the jobs before entering the production line, the job sequence is of worse quality with respect to the objective function and therefore gives a higher margin for the possible benefit of resequencing jobs within the line. Figure 7.17 and 7.18 show the improvement which is achieved by the Genetic Algorithm, when resequencing of a randomly generated sequence within the production line (semi dynamic case) is permitted, compared to the case without resequencing.

7.8 Paced production line

One of the main characteristic of a paced production line is the fact that the jobs are transported by some material handling system, as e.g. a conveyor belt, and usually the launch interval, i.e. the time between two successive jobs, is constant. As a concrete example, Lahmar et al., 2003, and Lahmar and Benjaafar, 2003, study the problem of resequencing a set of pre-arranged jobs for the paint shop of an automobile production. Neither the processing time of the jobs nor the appearance of setup time is considered. The objective is the minimization of the changeover cost which occur every time a color change occurs and paint is to be flushed, e.g., the cost for a metallic paint is higher than for a normal paint.

In order for the proposed Genetic Algorithm to consider this type of paced production line, both, the production times $P_{i,j}$ and the setup times $ST_{i,e,f}$ are set to uniform values. Also α and δ, the weights for the makespan and the setup time, respectively, are set to zero. In the work of Lahmar et al., 2003, and Lahmar and Benjaafar, 2003, the setup cost is not sequence dependent. However, due to the fact that their results do not provide sufficient information in order to reconstruct their results and the fact that the setup cost in their case is not sequence dependent, the results shown here are based on the data which were introduced in section 7.1, only tanking into account the setup cost.

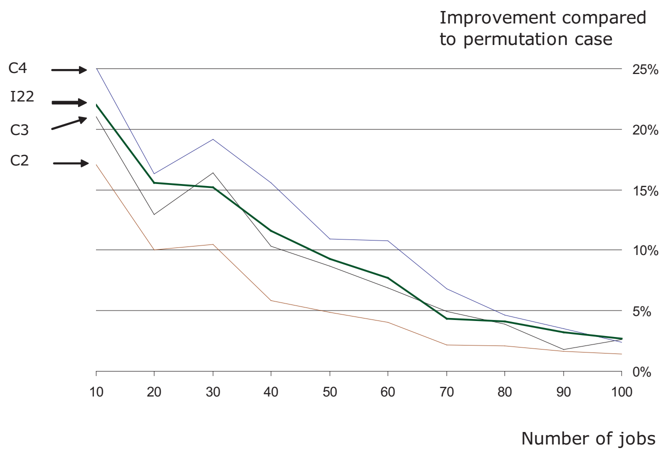


Figure 7.19: Paced flowshop production line only taking into account the setup cost. Station 2 and station 3 have access to resequencing buffers.

Figure 7.19 shows the considerable improvement for a 5 station flowshop, considering that the flowshop is a paced production line and the only interest lies in the reduction of the setup cost. Resequencing is possible after station 2 and after station 3. One intermediate case and three centralized cases are studied: two intermediate resequencing buffers provide two buffer places each (I22), one centralized resequencing buffer with two, three and four buffer places (C2, C3, C4).

Comparing the centralized cases, it can be seen that the case C4 is superior to case C3 which in turn is superior to the case C2. The more buffer places are available, the higher the improvement of the solutions compared to permutation solutions. Furthermore, the case C4 is superior to the case I22, caused by the fact that the case C4 provides more flexibility. Whereas the case C3 achieve similar results as the case I22, but with the reduction of one buffer place.

7.9 Benchmark comparison

The previous experimentation was performed using data sets which can not be found in the literature, caused by the fact that the problem, as formulated here, can not be found in the literature yet.

As highlighted in section 2.4.6 there exists a test-bed from Taillard, 1993, used by several authors for benchmark comparison, mainly using permutation sequences. These data do not take into account setup considerations.

In what follows, an attempt is made to compare results from other authors which presented results on permutation sequences or which were treating problems which are similar to the one studied in this work, based on the test-bed provided by Taillard, 1993.

7.9.1 Watson, Barbulescu, Whitley, Howe

In the work of Watson et al., 2002, different data sets for permutation flowshop instances are presented, without setup considerations. In one of the sets, the operation durations are independently and uniformly sampled from the interval [1,99], following Taillard, 1993, containing four different problem sizes: 20, 50, 100 and 200 jobs, each for a 20 station flowshop. For each problem size 100 samples are presented. Next to the sample data, also the so far best known solutions together with the lower bound are given, taking into account only permutation sequences.

In what follows, the Genetic Algorithm is applied to some of the data sets presented by Watson et al., 2002, permitting resequencing within the production line. During the study of the first instance it was discovered that the variation of the results was in the range of +/-0.5% and that in turn meaningful results are obtained by considering only 20 out of 100 samples. In the case of 200 jobs and 20 stations only 5 sample data were studied.

In a first step the data is used exactly as provided by Watson et al., 2002, without setup considerations. The results show the comparison of the Genetic Algorithm with the so far best known permutation sequence and the improvement of the non-permutation sequence compared to the permutation sequence, both determined by the Genetic Algorithm. Then, in a second step, next to the data which only consist of production times, a setup time in the range [2,8] and a setup cost in the range of [1,5] is additionally generated.

The Genetic Algorithm considers that for the first third of the jobs the physical size is 1, the second third has a physical size of 2, and the last third 3. Due to the fact that the number of jobs is not dividable by 3, the rest of the jobs are sized 1 and 2. And as a consequence, this study also includes that the buffer places of the resequencing buffer first are set to 3, which is that all jobs may pass through every buffer place ($I3$, $C33$). Then in a second step, the physical size of the buffer places are limited to a physical size of 1, 2 and 3, respectively ($I3(lim)$, $C33(lim)$).

The solutions shown for the Genetic Algorithm are the best out of eight runs each. For each of the eight runs a different seed is used.

The graphical representations in this section show results which were obtained by experimentation with discrete data points. The connections of the data points do not result neither in a physical nor a meaningful representation and are only justified by giving a better visual presentation of the individual data points.

Instance 1: 20 stations, 20 jobs

Figure 7.20 shows the comparison of the found solutions with respect to the best kown solutions for permutation sequences. When the Genetic Algorithm is limited to permutation

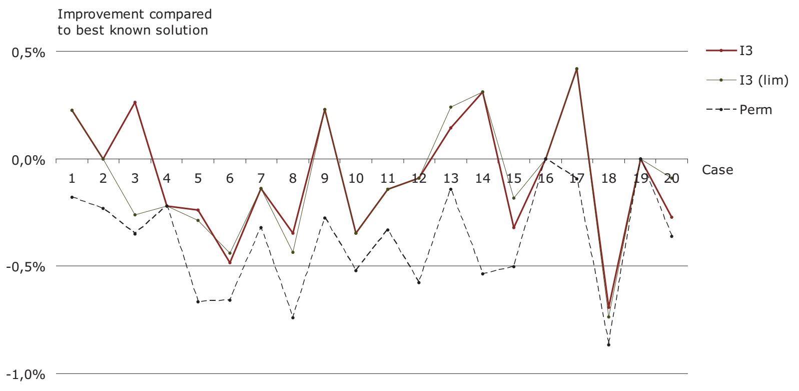


Figure 7.20: Comparison of solutions with the best known solutions for permutation sequences for 20 jobs and 20 stations.

sequences only ($Perm$), the algorithm reaches the best known solutions only in two of the twenty cases. Whereas, when resequencing is permitted, $I3$, the solutions are improved and in eight cases reach at least the best known solutions for permutation sequences. In the case in which the physical size of the buffer places are limited, $I3(lim)$, the solutions in general are similar to the previous ones and in all but three cases are better than the permutation sequences obtained by the Genetic Algorithm.

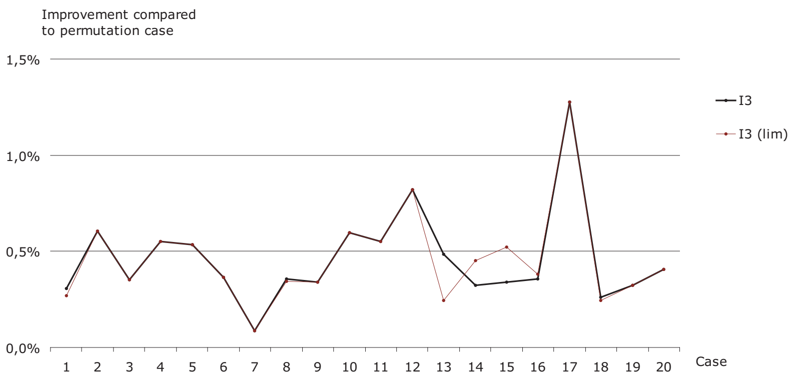


Figure 7.21: Comparison of solutions permutation case for 20 jobs and 20 stations, with setup considerations.

Figure 7.21 shows the comparison of the solutions found by the Genetic Algorithm, taking into account setup considerations, with respect to the best found permutation sequence of the Genetic Algorithm. An average improvement of about 0.5% is achieved.

The explicit data can be found in the appendix in tables A.32 and A.33. Furthermore, the comparison for the average out of eight runs can be found in figures A.2 and A.3 and the respective data in tables A.34 and A.35.

Instance 2: 50 jobs, 20 stations

Figure 7.22 shows that in the case of 50 jobs, the average mismatch of the permutation sequence found by the Genetic Algorithm is about 3%. Whereas, when resequencing is permitted, $I3$, the solutions are improved by about 1%. In the case in which the physical size of the buffer places are limited, $I3(lim)$, the solutions in general are nearly as promising as the the previous ones.

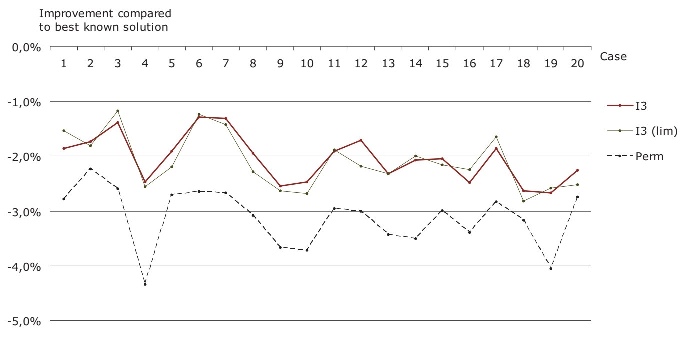


Figure 7.22: Comparison of solutions with the best known solutions for permutation sequences for 50 jobs and 20 stations.

Figure 7.23 shows that when setup considerations are taken into account, the average improvement with respect to permutation sequences found by the Genetic Algorithm is about 1.5%. The limitation of the physical size of the buffer places give almost as good results.

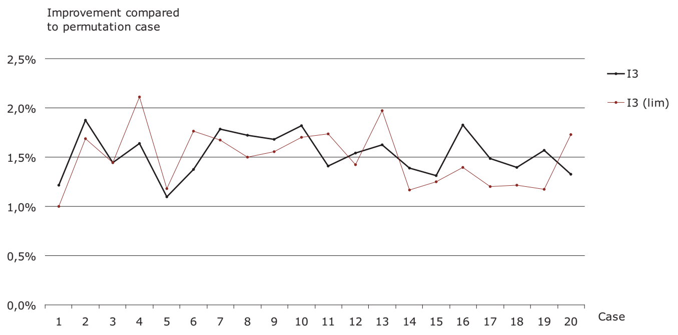


Figure 7.23: Comparison of solutions permutation case for 50 jobs and 20 stations, with setup considerations.

The explicit data can be found in the appendix in tables A.36 and A.37. Furthermore, the comparison for the average out of eight runs can be found in figures A.4 and A.5 and the respective data in tables A.38 and A.39.

Instance 3: 100 jobs, 20 stations

Figure 7.24 shows that in the case of 100 jobs, the average mismatch of the permutation sequence found by the Genetic Algorithm is just below 3%. Whereas, when resequencing is permitted, $C33$, the solutions are improved by about 0.5%. In the case in which the physical size of the buffer places are limited, $C33(lim)$, the solutions in general are nearly as promising as without this limitation.

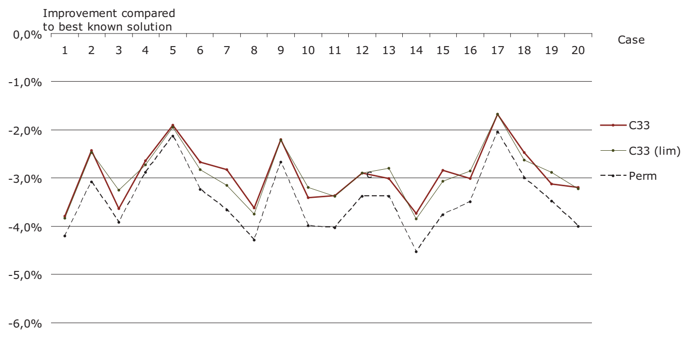


Figure 7.24: Comparison of solutions with the best known solutions for permutation sequences for 100 jobs and 20 stations.

Figure 7.25 shows that when setup considerations are taken into account, the average improvement with respect to permutation sequences found by the Genetic Algorithm is about 2%. The limitation of the physical size of the buffer places clearly limits the found solutions by nearly 0.3%.

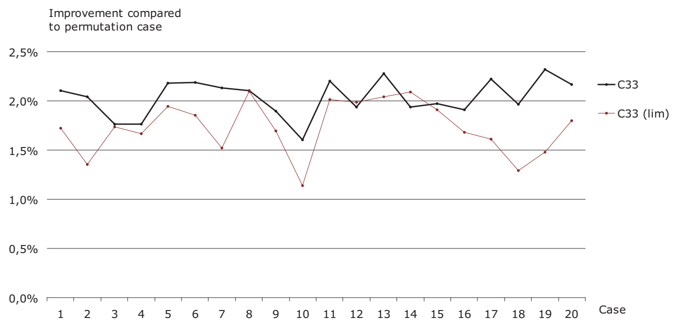


Figure 7.25: Comparison of solutions permutation case for 100 jobs and 20 stations, with setup considerations.

The explicit data can be found in the appendix in tables A.40 and A.41. Furthermore, the comparison for the average out of eight runs can be found in figures A.6 and A.7 and the respective data in tables A.42 and A.43.

Instance 4: 200 jobs, 20 stations

Figure 7.26 shows that in the case of 200 jobs, the average mismatch of the permutation sequence found by the Genetic Algorithm is just below 2.7%. Due to the elevated number of jobs, the possibility of resequencing within the line, $C33$ and $C33(lim)$ in any case results in an improvement of the achieved sequence, but only of about 0.2%.

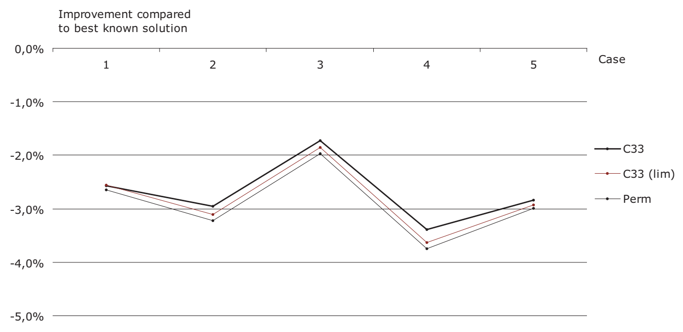


Figure 7.26: Comparison of solutions with the best known solutions for permutation sequences for 200 jobs and 20 stations.

Figure 7.27 shows that when setup considerations are taken into account, the average improvement with respect to permutation sequences found by the Genetic Algorithm varies between 1 and 2%. The limitation of the physical size of the buffer places still gives in two of five cases similar results to the case without limitation.

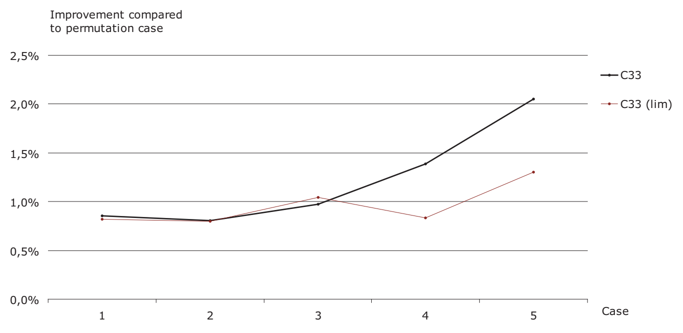


Figure 7.27: Comparison of solutions permutation case for 200 jobs and 20 stations, with setup considerations.

The explicit data can be found in the appendix in tables A.44 and A.45. Furthermore, the comparison for the average out of eight runs can be found in figures A.8 and A.9 and the respective data in tables A.46 and A.47.

7.9.2 Brucker, Heitmann, Hurink

The work of Brucker et al., 2003, presents a Tabu search for a flowshop with the possibility of resequencing within the production line. They position a buffer between all consecutive station with the possibility to resequence. The variable parameter is the number of buffer places which is 0, 1, 2, or infinite. The number of buffer places is the same for all resequencing buffers. The case of 0 buffer places does not allow resequencing and is equivalent to the permutation flowshop.

The presented results consider nine instances of 20, 50 and 100 Jobs on a 5-, 10-, and 20 station flowshop. Next to the results of the Tabu search, optimal solutions from other authors are listed, and if not available, the so far best lower bounds and the best known solutions for the permutation case and for the case in which all types of sequences are allowed, being the non-permutation case without any restrictions.

Case	Station with access to resequencing buffer		
No of stations	$I3$	$C3$	$I22$
5	2	1, 3	1,3
10	5	3, 6	3,6
20	10	6, 14	6,14

Table 7.4: Three cases are considered: $I3$, $C3$ and $I22$. Depending on the number of stations, 5, 10 or 20, the stations with access to the resequencing buffer are different.

For the comparison of the solutions presented by Brucker et al., 2003, together with the optimal solutions and the so far best known solutions with the results obtained by the Genetic Algorithm, five of the nine instances were tested. Three arrangements are considered for each instance and table 7.4 shows where the resequencing buffers are located.

For the Genetic Algorithm it is considered that for the first third of the jobs the physical size is 1, the second third has a physical size of 2, and the last third 3. Due to the fact that the number of jobs is not dividable by 3, the rest of the jobs are sized 1 and 2.

Case	Without limitation	With limitation
$I3$	3,3,3	1,2,3
$C3$	3,3,3	1,2,3
$I22$	3,3 / 3,3	2,3 / 1,2

Table 7.5: Allocation of the buffer places to the buffers.

For each of the three cases it is considered that in a first step the physical size of all jobs is at least as small as the one of the smallest buffer place. Then, in the second step, the physical size of the jobs and the buffer places are set to 1, 2 and 3. Table 7.5 shows the physical size of the buffer places for the three cases $I3$, $C3$ and $I22$.

In what follows, the solutions shown for the Genetic Algorithm are the best out of eight runs each. For each of the eight runs a different seed is used.

Instance 1: ($N = 20$**,** $M = 5$)

The comparison for the permutation case is shown in table 7.6. In eight out of ten cases the Genetic Algorithm obtains the optimal solution. The comparison for the average values are shown in the appendix in table A.48. The solutions, obtained by the Tabu search, differ by 6% to 25%.

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta010	1108	-	1302	1108
ta009	1230	-	1433	1230
ta008	1206	-	1436	1206
ta007	1234	-	1436	1239
ta006	1195	-	1434	1195
ta005	1235	-	1370	1235
ta004	1293	-	1471	1300
ta003	1081	-	1353	1081
ta002	1359	-	1451	1359
ta001	1278	-	1437	1278

Table 7.6: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker and solutions obtained by the Genetic Algorithm for 20 jobs and 5 stations. Fields with grey background indicate that the obtained solution is equal to the optimal solution.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta010	1108	-	1103	-	1134	1115	**1103**	**1103**	**1103**	1108	1108	1108	1108
ta009	1230	-	1210	-	1289	1249	**1210**	1230	1230	1230	1230	1230	1230
ta008	1206	-	1199	-	1227	1215	**1199**	**1199**	**1199**	1206	1206	1206	1206
ta007	1234	-	1234	-	1266	1251	1234	1239	1239	1239	1239	1239	1239
ta006	1195	-	1193	-	1268	1217	**1193**	**1193**	**1193**	1195	1195	1195	1195
ta005	1235	-	1231	-	1262	1250	**1231**	1235	1235	1235	1235	1235	1235
ta004	1293	-	1292	-	1329	1329	**1292**	1293	1293	1297	1297	1297	1297
ta003	1081	-	1073	-	1132	1098	**1073**	**1080**	**1080**	1081	1081	1081	1081
ta002	1359	-	1358	-	1365	1365	**1358**	1359	1359	1359	1359	1359	1359
ta001	1278	-	1278	-	1287	1287	1278	1278	1278	1278	1278	1278	1278

Table 7.7: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker for 1, 2 or n buffer places which are available at each station and the results obtained by the Genetic Algorithm for $I3$, $C3$ and $I22$, also for the case of physical size limitations for the buffer places (lim). The instances contain 20 jobs and 5 stations. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal permutation solution or the so far best known solutions for permutation solutions.

Table 7.7 shows the comparison for the non-permutation case. The same comparison, but using the average values can be found in the appendix in table A.49. It can be seen that the Genetic Algorithm obtains for the majority of the ten instances the optimal permutation solution, and in some cases even better solutions. In general the use of the arrangement $I3$ gives the most promising results. The solutions, obtained by the Tabu search of Brucker et al., 2003, show good results mainly for the case of n buffer places.

Instance 2: ($N = 20$**,** $M = 10$)

The comparison for 20 jobs and 10 stations for the permutation case are shown in table 7.8. The results show that in one instance the Genetic Algorithm reaches the optimal solution and in the other nine cases is in a range of 1.1%, whereas the Tabu search varies between 11% and 18% from the optimal solution. The comparison for the average values are shown in the appendix in table A.50.

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta020	1591	-	1806	1608
ta019	1593	-	1772	1594
ta018	1538	-	1788	1558
ta017	1484	-	1673	1484
ta016	1397	-	1632	1401
ta015	1419	-	1678	1427
ta014	1377	-	1620	1383
ta013	1496	-	1755	1508
ta012	1659	-	1875	1678
ta011	1582	-	1758	1590

Table 7.8: Comparison of optimal solutions or the best kown solutions with solutions obtained by Brucker and solutions obtained by the Genetic Algorithm for 20 jobs and 10 stations.

In the case of non-permutation sequences, table 7.9 shows that the Genetic Algorithm in many cases achieves results as good or even better than the optimal permutation sequence. For the instance "ta015", the case of $I3$ shows even a better solution than the so far best known solutions, instead of 1413 the value 1409 was found. In general, the three different arrangements, $I3$, $C3$, and $I22$, result in similar solutions, and the fact of limiting the physical size of the buffer sizes tends to result in slightly worse values. The comparison for the average values can be found in the appendix in table A.51. The solutions, obtained by the Tabu search, show better results for the case of n buffer places but remain relatively high.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta020	1591	-	1559	1525	1632	1642	1642	1603	1603	1606	1603	1603	1603
ta019	1593	-	1586	1558	1672	1628	1626	1594	1594	1594	1594	1594	1594
ta018	1538	-	1527	1446	1582	1585	1580	1538	1538	1554	1554	1553	1554
ta017	1484	-	1428	1400	1559	1521	1505	**1476**	**1479**	**1468**	**1468**	**1468**	**1479**
ta016	1397	-	1369	1347	1424	1413	1419	1400	1397	1397	1400	1397	1397
ta015	1419	-	1413	1374	1501	1476	1463	**1409**	**1409**	1419	1419	1419	1419
ta014	1377	-	1368	1356	1433	1413	1402	1382	1382	1383	**1374**	**1374**	1383
ta013	1496	-	1486	1450	1565	1544	1540	1508	1508	1501	1501	1501	1504
ta012	1659	-	1644	1603	1763	1737	1737	1667	1664	1665	1665	1665	1665
ta011	1582	-	1560	1549	1681	1659	1659	1583	1583	1586	1586	1586	1586

Table 7.9: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker for 1, 2 or n buffer places and the results obtained by the Genetic Algorithm for $I3$, $C3$ and $I22$, also for the case of physical size limitations for the buffer places (lim). The instances contain 20 jobs and 10 stations.

Instance 3: $(N = 20, M = 20)$

The comparison for 20 jobs and 20 stations for the permutation case are shown in table 7.10. The results show that the solutions of the Genetic Algorithm are in a range of 0.6%, whereas the Tabu search varies between 7% and 9% from the optimal solution. The comparison for the average values are shown in the appendix in table A.52.

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta030	2178	-	2361	2192
ta029	2237	-	2411	2242
ta028	2200	-	2367	2212
ta027	2273	-	2436	2280
ta026	2226	-	2414	2230
ta025	2291	-	2507	2302
ta024	2223	-	2389	2229
ta023	2326	-	2527	2336
ta022	2099	-	2292	2111
ta021	2297	-	2512	2303

Table 7.10: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker and solutions obtained by the Genetic Algorithm for 20 jobs and 20 stations.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta030	2178	-	2178	1992	2257	2257	2257	2178	**2173**	2179	2179	2179	2179
ta029	2237	-	2227	1941	2310	2298	2300	2242	2242	2242	2242	2242	2242
ta028	2200	-	2183	1961	2247	2249	2249	2212	2204	2212	2212	2212	2212
ta027	2273	-	2267	2010	2383	2383	2347	**2272**	2277	**2267**	**2265**	**2272**	2273
ta026	2226	-	2221	1971	2285	2270	2349	2228	2228	2228	2228	2228	2228
ta025	2291	-	2291	2086	2365	2369	2369	2298	2298	2291	2297	2294	2291
ta024	2223	-	2223	2001	2242	2242	2242	**2199**	**2199**	**2220**	**2220**	**2220**	**2220**
ta023	2326	-	2313	2006	2396	2401	2404	2336	2333	2329	2329	2329	2329
ta022	2099	-	2092	1847	2134	2134	2134	2104	2105	2103	2107	2105	2107
ta021	2297	-	2293	2021	2428	2404	2428	2300	2297	2300	2300	2300	2300

Table 7.11: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker for 1, 2 or n buffer places and the results obtained by the Genetic Algorithm for $I3$, $C3$ and $I22$, also for the case of physical size limitations for the buffer places (lim). The instances contain 20 jobs and 20 stations.

In the case of non-permutation sequences, table 7.11 shows that the Genetic Algorithm in many cases achieves results as good or even better than the optimal permutation sequence. For the three instances "ta024", "ta027" and "ta030" even better solutions than the so far best known solutions are found. In many cases the fact of having two stations with access to a resequencing buffer ($C3$ and $I22$) shows better results and limiting the physical size of the buffer sizes also tends to be slightly worse. The comparison for the average values can be found in the appendix in table A.53. The solutions, obtained by the Tabu search, in general show better results for the case of 2 buffer places but remain relatively high.

Instance 4: ($N = 50$**,** $M = 5$)

The comparison for the permutation case is shown in table 7.12. In eight out of ten cases the Genetic Algorithm obtains the optimal solution. The comparison for the average values are shown in the appendix in table A.54. The solutions, obtained by the Tabu search, differ by 16% to 24%.

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta040	2782	-	3350	2782
ta039	2552	-	3045	2561
ta038	2683	-	3187	2683
ta037	2725	-	3166	2725
ta036	2829	-	3364	2829
ta035	2863	-	3350	2863
ta034	2751	-	3334	2751
ta033	2621	-	3265	2621
ta032	2834	-	3385	2838
ta031	2724	-	3238	2724

Table 7.12: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker and solutions obtained by the Genetic Algorithm for 50 jobs and 5 stations.

In the case of non-permutation sequences, table 7.13 shows that the Genetic Algorithm in many cases achieves results as good or even better than the optimal permutation sequence. For the instances "ta035" and "ta035", the the Genetic Algorithm shows even the optimal solution for the non-permutation case. In general the fact of having two stations with access to a resequencing buffer ($C3$ and $I22$) shows better results and limiting the physical size of the buffer sizes tends to be slightly worse. The comparison for the average values can be found in the appendix in table A.55. The solutions, obtained by the Tabu search, in general show better results for the case of n buffer places and achieves only in two cases results which are as good or better than the optimal solution of the permutation case.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta040	2782	-	2776	-	2856	**2776**	**2776**	2782	2782	**2776**	**2776**	**2776**	**2776**
ta039	2552	-	2545	2542	2599	2558	2559	2557	2557	2561	2561	2561	2561
ta038	2683	-	2683	-	2769	2697	2688	2683	2683	2683	2683	2683	2683
ta037	2725	-	2716	2715	2765	2843	2843	2725	2725	2725	**2717**	2725	2725
ta036	2829	-	2825	-	2916	2837	2829	**2828**	**2828**	2829	2829	2829	2829
ta035	2863	-	2853	-	2918	2871	2872	**2857**	**2857**	**2853**	**2853**	**2853**	**2853**
ta034	2751	-	2751	-	2888	2764	2782	2751	2751	2751	2751	2751	2751
ta033	2621	-	2612	-	2730	2632	2623	**2620**	**2620**	**2615**	**2615**	**2615**	2621
ta032	2834	-	2834	-	2913	2877	2882	2838	2838	2838	2838	2838	2838
ta031	2724	-	2724	-	2808	2729	2729	2724	2724	2724	2724	2724	2724

Table 7.13: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker for 1, 2 or n buffer places and the results obtained by the Genetic Algorithm for $I3$, $C3$ and $I22$, also for the case of physical size limitations for the buffer places (lim). The instances contain 50 jobs and 5 stations.

Instance 5: $(N = 50, M = 10)$

The comparison for the permutation case is shown in table 7.14. The comparison for the average values are shown in the appendix in table A.56. The Genetic Algorithm show results which differ between 0.9% and 3.7% from the optimal permutation sequence but in all cases show better results than the Tabu search, which differs between 22% and 30%.

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta050	3065	-	3816	3146
ta049	2897	-	3771	2968
ta048	3037	-	3722	3079
ta047	3093	-	3789	3160
ta046	3006	-	3755	3075
ta045	2976	-	3838	3060
ta044	3063	-	3844	3090
ta043	2839	-	3658	2930
ta042	2867	-	3664	2972
ta041	2991	-	3806	3068

Table 7.14: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker and solutions obtained by the Genetic Algorithm for 50 jobs and 10 stations.

In the case of non-permutation sequences, table 7.15 shows that the Genetic Algorithm differs between 0.7% and 3.2% from the optimal permutation sequence, which in any case is better than the results of the permutation case in table 7.14. The Tabu search differs from 2.2% and 8.9%.

In general the fact of having only one stations with access to a resequencing buffer ($I3$) shows better results and limiting the physical size of the buffer sizes in general tends to be worse. The comparison for the average values can be found in the appendix in table A.57. The solutions, obtained by the Tabu search, in general show better results for the case of 2 buffer places.

	Permutation		Non-Permutation										
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	Genetic Algorithm I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta050	3065	-	3065	3046	3273	3169	3201	3131	3131	3138	3138	3138	3138
ta049	2897	-	2887	2858	3114	3049	3013	2952	2962	2962	2962	2958	2962
ta048	3037	-	3026	3001	3183	3142	3150	3060	3074	3079	3079	3079	3079
ta047	3093	-	3093	-	3348	3234	3234	3156	3156	3156	3147	3147	3151
ta046	3006	-	2991	2981	3177	3126	3119	3065	3075	3075	3075	3075	3075
ta045	2976	-	2976	2935	3232	3141	3152	3037	3026	3040	3045	3043	3045
ta044	3063	-	3063	3059	3242	3129	3146	3087	3087	3087	3085	3086	3085
ta043	2839	-	2832	2828	3077	2925	2964	2926	2926	2926	2927	2929	2926
ta042	2867	-	2867	2829	3111	3003	3031	2946	2932	2957	2957	2957	2957
ta041	2991	-	2970	2970	3258	3179	3142	3063	3063	3062	3066	3062	3068

Table 7.15: Comparison of optimal solutions or the best known solutions with solutions obtained by Brucker for 1, 2 or n buffer places and the results obtained by the Genetic Algorithm for $I3$, $C3$ and $I22$, also for the case of physical size limitations for the buffer places (lim). The instances contain 50 jobs and 10 stations.

Comparison of time

Apart from the experimental results, Brucker et al., 2003, also present mean computational times. Table 7.16 shows the comparison of the mean computational time with the time consumed by the Genetic Algorithm. In the latter case, the computational time does not vary significantly when the number of resequencing possibilities is raised, see also figure 6.19, therefore the mean value is calculated over the cases with the same number of jobs and stations. The performance of the GA in terms of computational time is similar to the Tabu search, mainly when the number of stations is small. However, considering that the Tabu search (except for $b_i = 0$) uses more resequencing buffer places and the found solutions are worse, the GA outperforms the Tabu search.

		Brucker				GA
Jobs	*Stations*	b_i=0	b_i=1	b_i=2	b_i= inf	
20	5	0:23	0:19	0:14	0:15	0:13
	10	0:53	0:31	0:34	0:32	1:20
	20	1:28	0:49	0:47	0:42	2:20
50	5	6:01	2:35	1:19	0:57	1:10
	10	10:22	4:15	2:59	1:51	10:04

Table 7.16: Comparison of the mean computational time for the Tabu search from Brucker et al., 2003, and the Genetic Algorithm (in minutes : seconds).

7.10 Conclusions

In this chapter the proposed Genetic Algorithm was applied to the problem of resequencing jobs in a flowshop production line, using buffers which are located off the production line, either accessible from a single station (intermediate case) or from various stations (centralized case). These buffers are furthermore constrained in terms of the number of buffer places and the fact that a job may not be able to be stored in a certain buffer place, due to its extended physical size.

The presence of setup cost or setup time does influence the potential benefit that comes with the possibility of resequencing, particularly when both appear at the same time. When comparing the intermediate with the centralized buffer location, the centralized buffer location results in better solutions when the same number of buffer places are provided, and on the other hand open the way for a reduction of the necessary buffer area without major impact on the solution. Whereas a reduction in buffer places, while maintaining the same buffer location, reduces the objective function considerably.

However, the introduction of physical size limitations for the buffer places shows that the effect on the obtained solutions, for this type of reduction of the buffer area, is less compared to the reduction of entire buffer places. A rather big influence on the obtained solutions was demonstrated for different distributions of the input data, and very promising results were achieved for the case of semi dynamic demand with fixed job sequence for the first station, as well as for the paced production line.

For the performance study a total of 7200 experiments were conducted, including the Benchmark comparison. The GA was validated with the consideration of a variety of different plant arrangements, proved good performance, and gave evidence that the insertion of constrained resequencing buffers can improve the solutions considerable. In the case of Watson et al., 2002, the algorithm improved some of the best known solutions for the small instances and shows an average mismatch of up to 3% for the larger instances. The comparison with the Tabu search of Brucker et al., 2003, who propose to use a limited but equal number of buffer places between the stations, showed that the Genetic Algorithm performs similar in terms of computational time, but outperforms the Tabu search with respect to the found solutions, which is even more surprising due to the fact that the Tabu search apart from the permutation case does consider more buffer places for resequencing.

Furthermore, it has to be mentioned that despite the fact that the replacement of the intermediate resequencing buffers by a centralized resequencing buffer results in benefits with respect to the objective function or in a reduced buffer area, additional effort maybe necessary in terms of logistics, in order to keep track of the stored jobs, or in terms of additional hardware which may be necessary to install in order to enable the transfer of the jobs to the centralized resequencing buffer. In any case the additional effort has to be properly weighed for the specific case.

8 Performance Comparison: Exact versus Heuristic Approach

Within the present thesis, an exact and a heuristic approach were presented. The first one uses Constrained Logic Programming (CLP) and was furthermore modified to a hybrid approach which is a compound of the CLP and a simplified Genetic Algorithm. Based on this approach, a more sophisticated Genetic Algorithm was presented. Due to the fact that the performance study for the two methods was accomplished separately, this chapter presents a comparison of the two, based on the results obtained in chapter 5.

8.1 General frame

A flowshop which consists of 10 stations is used. After station 3, 5 and 8 a single intermediate buffer place is located. The range of the production time is [1...100], for the setup cost [2...8] and for the setup time [1...5]. The number of jobs is varied in the range of 4 to 10 and the objective function is the weighted sum of the makespan (factor 1.0) and the setup cost (factor 0.3), where the setup time is not concerned with a weight but is indirectly included in the calculation of the makespan.

8.2 Difference in physical size of jobs

This experimentation uses the same 10 station flowshop which was already introduced in section 5.6.2. After station 3, 5 and 8 access to resequencing buffer places exist, three differently sized buffer places (large, medium, small) are available and the ratio of jobs is $\frac{3}{10}$

large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small. The allocation of the buffer places to the buffers considers five scenarios for the intermediate case ("I111", "I231", "I132", "I222", "I333") and three scenarios for the centralized case ("C1", "C2", "C3"). "I132" represents 1 small, 1 large and 1 medium buffer place, located as intermediate resequencing buffer places after stations 3, 5 and 8, respectively. "C2" represents 1 medium buffer place, located as a centralized buffer place, accessible from stations 3, 5 and 8. "I333" and "C3" are the two cases which provide the largest flexibility in terms of physical size restrictions.

Jobs	GA_P	I111	I231	I132	I222	I333	C1	C2	C3
4	481,2	476,1	**474,9**	**474,9**	**474,9**	**474,9**	476,1	476,1	476,1
5	470,5	470,5	464,2	467,6	464,2	**449,4**	470,5	464,2	**449,4**
6	564,6	541,1	542,1	541,1	540,9	**537,6**	543,9	538,8	537,8
7	568,7	566,8	564,9	566,8	564,9	**563,7**	566,8	566,8	563,8
8	589,6	589,6	589,3	588,8	589,2	**586,0**	589,2	589,6	589,5
9	679,3	676,3	668,8	**666,7**	675,9	671,7	677,2	675,7	670,8
10	736,1	735,5	729,7	725,8	735,4	**724,0**	735,5	735,5	725,3

Table 8.1: Difference in physical size of jobs for the Genetic Algorithm.

The results of the Genetic Algorithm are presented in table 8.1. The algorithm performs best for the case "I333" and nearly as good in the case "C3", and in any case, it outperforms the solutions which are limited to permutation sequences.

			Minimum				Average			
Jobs	A_P	GA_P	A_{NP}	A_C	A_{MSS}	GA	A_{NP}	A_C	A_{MSS}	GA
4	481,2	481,2	481,2	480,0	477,7	**474,9**	481,2	480,5	478,2	**475,1**
5	470,5	470,5	470,5	470,5	451,8	**449,4**	470,5	470,5	463,7	**463,2**
6	564,6	564,6	564,6	563,4	**537,1**	537,6	564,6	564,2	**538,1**	540,6
7	568,7	568,7	568,7	566,6	565,8	**563,7**	570,3	567,9	568,2	**565,4**
8	589,6	589,6	589,6	589,6	589,1	**586,0**	605,6	589,8	593,4	**588,6**
9	684,3	679,3	684,3	681,0	685,5	**666,7**	694,8	683,1	689,6	**671,9**
10	736,1	736,1	742,4	735,5	740,9	**724,0**	750,1	738,2	744,2	**730,1**

Table 8.2: Comparison of the Genetic Algorithm and the CLP considering differences in physical size of jobs for the intermediate case.

			Minimum				Average			
Jobs	A_P	GA_P	A_{NP}	A_C	A_{MSS}	GA	A_{NP}	A_C	A_{MSS}	GA
4	481,2	481,2	481,2	480,0	477,7	**476,1**	481,2	480,0	478,1	**476,1**
5	470,5	470,5	470,5	470,5	451,8	**449,4**	470,5	470,5	462,2	**461,4**
6	564,6	564,6	564,6	564,0	**537,7**	537,8	564,6	564,0	**538,2**	540,1
7	568,7	568,7	568,7	567,5	567,0	**563,8**	570,7	567,5	567,9	**565,8**
8	589,6	589,6	589,6	589,6	**588,0**	589,2	610,9	589,6	591,0	**589,4**
9	684,3	679,3	684,3	680,7	684,8	**670,8**	696,2	682,5	689,0	**674,5**
10	736,1	736,1	742,4	735,2	736,9	**725,3**	749,1	735,4	741,6	**732,1**

Table 8.3: Comparison of the Genetic Algorithm and the CLP considering differences in physical size of jobs for the centralized case.

The comparison of the Genetic Algorithm and the Constrained Logic Programming are shown in table 8.2 and table 8.3 for the intermediate and the centralized case, respectively. The permutation solutions found by the Genetic Algorithm, GA_P, are the same as for the permutation solutions of the CLP, except in one case (684,3 versus 679,3).

Furthermore, the Genetic Algorithm in general achieves similar or slightly better results for the intermediate and the centralized case for less than 9 jobs. In the case of 9 and 10 jobs, better solutions are obtained by the Genetic Algorithm.

8.3 Semi dynamic demand

In the case of the semi dynamic demand, the same input values are used as in the previous case. The only difference is that the sequence for the first station is set to a fixed sequence. For the comparison the same sequence for the first station is used for the Constrained Logic Programming and the Genetic Algorithm.

Jobs	Perm	I111	I231	I132	I222	I333	C1	C2	C3
4	483,1	480,9	**480,7**	**480,7**	480,9	**480,7**	480,9	**480,7**	**480,7**
5	552,0	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**
6	647,5	627,0	**620,1**	**620,1**	**620,1**	**620,1**	628,5	622,2	622,2
7	636,5	627,7	627,7	625,0	625,0	**609,5**	628,0	627,7	616,1
8	673,6	646,6	646,6	644,8	644,8	644,8	646,6	644,8	**632,3**
9	744,3	719,4	719,4	716,1	716,1	716,1	719,4	716,1	**712,1**
10	813,7	786,5	763,2	**762,5**	785,9	791,2	786,5	786,5	764,0

Table 8.4: Semi dynamic demand using the Constrained Logic Programming.

Table 8.4 shows the result for the CLP. In any case, when offline resequencing buffers are considered, the results are improved compared to the permutation sequence. In the studied flowshop, an average of 4.3% is achieved for the use of the CLP. Whereas in the case of the Genetic Algorithm , see table 8.5, the average is 3.7%.

Jobs	Perm	I111	I231	I132	I222	I333	C1	C2	C3
4	483,1	480,9	**480,7**	**480,7**	**480,7**	**480,7**	480,9	**480,7**	**480,7**
5	552,0	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**	**490,7**
6	647,5	627,0	**620,1**	**620,1**	**620,1**	**620,1**	628,5	622,7	622,2
7	636,5	627,7	628,8	626,1	625,0	**609,5**	629,1	628,5	616,3
8	673,6	669,2	651,3	646,0	647,2	638,2	672,4	653,4	**637,0**
9	744,3	736,5	724,2	728,6	721,8	**709,4**	736,5	729,5	714,7
10	813,7	808,3	788,0	781,7	805,1	**757,5**	809,2	805,9	772,2

Table 8.5: Semi dynamic demand using the Genetic Algorithm.

In the case of the exact approach, as well as in the approach of the GA, the semi dynamic demand with a fixed job sequence for the first station, leads to considerably larger improvements, even for larger problem sizes. Table 8.6 shows the improvement of the CLP with respect to the GA. For only a few jobs, both methods achieve the same solutions. When 6 or more jobs are to be sequenced, the CLP in general outperforms the GA when the number of buffer places at each resequencing possibility is limited to a small number.

Jobs	Perm	I111	I231	I132	I222	I333	C1	C2	C3
4	483,1	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**
5	552,0	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**
6	647,5	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	**0,0%**	0,1%	**0,0%**
7	636,5	**0,0%**	0,2%	0,2%	**0,0%**	**0,0%**	0,2%	0,1%	**0,0%**
8	673,6	3,4%	0,7%	0,2%	0,4%	-1,0%	3,8%	1,3%	0,7%
9	744,3	2,3%	0,7%	1,7%	0,8%	-0,9%	2,3%	1,8%	0,4%
10	813,7	2,7%	3,1%	2,5%	2,4%	-4,5%	2,8%	2,4%	1,1%

Table 8.6: Comparison of the Genetic Algorithm and the Constrained Logic Programming for semi dynamic demand. The values show the improvement of the CLP with respect to the GA.

8.4 Applicability of CLP versus Genetic Algorithm

The exact approach A_{NP} is very limited with respect to the problem size. In section 5.1 a problem was solved with optimality for the case of 10 stations, 4 jobs and 2 resequencing possibilities ($M = 10, N = 4, L = 2$). Whereas, the alternative (hybrid) approaches A_{MSS} and A_C, presented in section 5.2, lead to further improvements. The consideration of *limited flexibility*, in section 5.5 positively influences the performance of the CLP. In the case of a semi dynamic demand, the CLP and the GA apparently show different behaviors.

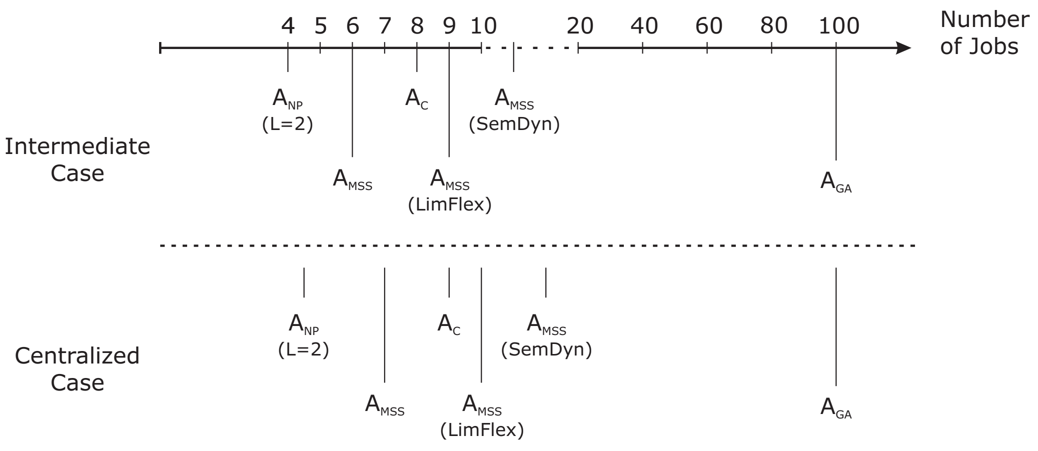


Figure 8.1: Applicability of the pure CLP and the hybrid CLP versus Genetic Algorithm.

For the case of 10 jobs, the CLP outperforms the GA for few resequencing possibilities. Whereas, the GA performs better when several buffer places but only few resequencing possibilities are considered with several buffer places (L small, D large). In general, the CLP gives better results for the case of a centralized resequencing buffer. The GA is applicable for small as well as for problems with at least up to 100 Jobs, in some studied cases, as e.g. in section 7.3, the improvements are declining when more than 60 jobs are considered.

Figure 8.1 shows the summary of the applicability of the different approaches, for the case of maximum 3 resequencing possibilities and with respect to the number of jobs to be processed. The number of stations (M) is not considered to be a very limiting factor, compared to the number of jobs (N) and the number of resequencing possibilities (L).

8.5 Conclusions

Comparing the results of the proposed approaches (exact, hybrid and heuristic), it can be concluded that the CLP, including the hybrid approach, performs good on problems with 10 stations, approximately 10 jobs and only few distributed resequencing buffer places. The CLP and the hybrid CLP furthermore are positively influenced when additional restrictions are present, as e.g. limited flexibility, or the introduction of the centralized instead of the intermediate buffer location. This is also supported by the results which were obtained in chapter 5.

As already highlighted in chapter 7, table 8.1 further underlines the fact that the GA performs better when only few stations are considered with access to offline resequencing buffers and in addition with several buffer places each.

The experimentation of the CLP, presented in chapter 5, was repeated with the Genetic Algorithm which resulted in approximately 900 experiments. The direct comparison of the GA with the CLP, together with the hybrid CLP, shown in table 8.2 and 8.3, demonstrates that the GA outperforms the CLP and the hybrid CLP in the vast majority of the cases with static demand.

In the exact approach, as well as in approach of the GA, the semi dynamic demand with a fixed job sequence for the first station, the improvements are considerably larger, even for larger problem sizes. Table 8.6 underlines the fact that the CLP outperforms the GA when the number of buffer places at each resequencing possibility is limited to a small number.

Figure 8.1 finally shows the applicability of the three different approaches (exact, hybrid and heuristic) with respect to the problem size, depending on whether the intermediate or the centralized case is considered.

9 Conclusions, contributions and future work

The present thesis is located in the area of mixed model flowshop production lines where jobs of more than one model are being processed on the same production line in an arbitrary sequence. Unlike the majority of publications in this area, Potts et al., 1991, and Liao et al., 2006 study the improvements when the possibility of resequencing jobs between selected stations is regarded; the considerable improvements are even more evident when setup cost/time exists. The problem is *NP-hard* and as highlighted by Lahmar et al., 2003, only few resequencing possibilities are necessary in order to achieve the greatest benefit.

The buffers, which were used in this work, in order to accomplish resequencing, are located off the production line, either accessible from a single station (intermediate case) or from various stations (centralized case), and are furthermore constrained in terms of the number of buffer places and the fact that a job may not be able to be stored in a certain buffer place, due to its extended physical size.

9.1 Main results

Following the extensive *State-of-the-Art*, which led to the problem under study, a *Novel Classification of Non-permutation Flowshops* is proposed. This classification was indispensable, due to the lack of an adequate classification for flowshop production lines that would consider the diversity of arrangements which permit resequencing of jobs within the production line. The classification is based on the notation used by Pinedo, 1995, but also establishes criteria that adequately categorize flowshops which provide the possibility of resequencing, including a wide scope of resequencing facilities and objectives.

In order to solve the problem under study, two distinct formulations were presented. The first is an exact approach, with *Constrained Logic Programming* (CLP). The analysis of the semi dynamic demand, with a fixed job sequence for the first station, lead to the creation of several alternative (hybrid) approaches. These hybrid approaches admitted to improve the results obtained for more complex case with a static demand. The formulation was implemented in OPL Studio version 3.7. The performance study of the exact and the hybrid approaches showed that the hybrid approach performs good until problems with 10 stations, approximately 10 jobs and only few distributed resequencing buffer places. The CLP and the hybrid approach are furthermore positively influenced when additional restrictions are present, as e.g. limited flexibility, or the introduction of the centralized instead of the intermediate buffer location.

The second formulation is a heuristic approach, a *Genetic Algorithm* (GA). Several genetic operators were used, among them inheritance, crossover and mutation. The genetic operators specify in which way the subsequent population is generated by reproduction of the present population, taking into account that ”fitter” solutions are more promising and therefore are more likely to reproduce. Even an unfeasible solution is able to reproduce, because of the fact that it may generate valuable and feasible solutions in one of the following generations. In order to further improve the Genetic Algorithm, it was partitioned into two cascades. In the first cascade, the possibility of resequencing jobs within the production line is ignored. The last generation, together with the best solution found, form the initial generation for the next cascade where the resequencing possibilities, provided by stations with access to resequencing buffers, are taken into account.

A preliminary analysis was performed on the Genetic Algorithm with 2200 individual runs, which on the one hand showed the existence of a continuous solution space for the genetic operators, and on the other hand gave evidence for the variability of the algorithm, caused by its random character. Then, the values of the genetic operators were determined with an extended experimentation, consisting of four steps (*rough adjustment*, *repeatability*, *clustering*, *fine adjustment*), ensuring flexibility and robustness.

For the performance study, a total of 7200 experiments were conducted, including the Benchmark comparison. The GA was validated with the consideration of a variety of different plant arrangements and proved good performance. In particular, the GA performs better when only few stations are considered with access to offline resequencing buffers and in addition with several buffer places each. The GA is applicable for small as well as for problems with up to 100 Jobs. However, in some studied cases, the improvements are declining when more than 60 jobs are considered.

The comparison of the proposed approaches furthermore showed that in the hybrid approach as well as in approach of the GA, the semi dynamic demand with a fixed job sequence for the first station, increases the possible improvements, even for larger problem sizes. applicability of the three different approaches (exact, hybrid and heuristic) was resumed in figure 8.1, with respect to the problem size, depending on whether the intermediate or the centralized case is considered.

During the course of this thesis, the realized studies of performance demonstrated the effectiveness of resequencing by examining certain characteristics. The results of the simulation experiments reveal the benefits that come with a centralized buffer location, compared to the intermediate buffer location. It either improve the solution or leads to the utilization of fewer resequencing buffer places. An increased number of buffer places clearly improves the objective function and including buffers, constrained by the physical size of jobs to be stored, on the one hand limits the solutions, but on the other hand minimizes the necessary buffer area.

In order to take full advantage of the possibilities of resequencing jobs in a mixed model flowshop, additional installations may be necessary to mount, as for example buffers, but also extra efforts in terms of logistics complexity may arise. The additional effort is reasonable if it pays off the necessary investment. Due to the dependency on local conditions, a general validation is not simple and was not part of this work.

The considered problem is relevant to various flowshop applications such as chemical productions dealing with client orders of different volumes and different sized resequencing tanks. Also in productions where split-lots are used for engineering purpose, such as the semiconductor industry. Even in the production of prefabricated houses with, e.g., large and small walls passing through consecutive stations where electrical circuits, sewerage, doors, windows and isolation are applied. The use of centralized buffers is especially suitable with U-shaped production line.

9.2 Contributions

As part of the thesis, a research stay of three months was performed at the Center for Heuristic Optimisation at the Kent Business School, University of Kent, Canterbury, UK. The accomplished activities consisted in the elaboration of the alternative arrangements of the Constrained Logic Programming (CLP).

The following national and international publications were contributed during the course of this thesis:

Niu, H., Coves A.M. and Färber G. (2003), Scheduling and Sequencing problem of Mix-Model-Assembly-Line, *Industrial Engineering and Engineering Management (IE&EM'2003)*, Shanghai, China.

Färber G. and Coves A.M. (2004), Secuenciación de ensamblaje mixtas con estaciones abiertas mediante un algoritmo de enumeración, Documento Técnico, IOC-DT-P-2004-10, Universitat Politècnica de Catalunya, Barcelona, Spain.

Färber G. and Coves A.M. (2005), Overview on: Sequencing in mixed model flowshop production lines with static and dynamic context, Documento Técnico, IOC-DT-P-2005-07, Universitat Politècnica de Catalunya, Barcelona, Spain.

Färber G. and Coves A.M. (2005), Modelo de programación lógica de restricciones (CLP) para una línea de producción con posibilidad de resecuenciar considerando almacenes limitados, *IX Congreso de Ingeniería de Organización (CIO 2005)*, Gijón, Spain

Färber G. and Coves A.M. (2006), Performance Study of a Genetic Algorithm for Sequencing in Mixed Model Non-Permutation Flowshops using Constrained Buffers, *Lecture Notes in Computer Science*, 3982/2006:638-648, ISSN: 0302-9743.

Färber G. and Coves A.M. (2006), Genetic Algorithm for Sequencing in Mixed Model Non-Permutation Flowshops using Constrained Buffers, *XXIX Congreso Nacional de la Sociedad de Estadística e Investigación Operativa (SEIO 2006)*, Tenerife, Spain.

Färber G. and Coves A.M. (2006), Extended Classification for Flowshops with Resequencing, *XXIX Congreso Nacional de la Sociedad de Estadística e Investigación Operativa (SEIO 2006)*, Tenerife, Spain.

Färber G. and Coves A.M. (2006), Analysis and Adjustment of a Genetic Algorithm for Non-Permutation Flowshops, *Applied Mathematical Programming and Modelling (APMOD2006)*, Madrid, Spain.

Färber G. and Coves A.M. (2006), Modelo CLP reducido para una línea de producción con posibilidad de resecuenciar considerando almacenes limitados, *X Congreso de Ingeniería de Organización (CIO 2006)*, Valencia, Spain.

Färber G., Salhi S. and Coves A.M. (2006), Heuristic Approach for solving a Flowshop with Constrained Resequencing Buffers with Multistart Constrained Logic Programming, *OR48 Annual Conference*, University of Bath, United Kingdom.

Färber G. and Coves A.M. (2006), Benchmark Results of a Genetic Algorithm for Non-Permutation Flowshops using Constrained Buffers, *10th International Research/Expert Conference (TMT 2006)*, Lloret de Mar, Spain.

The following publications are in preparation:

Färber G. and Coves A.M., Genetic Algorithm for Sequencing in Non-Permutation Flowshops using Constrained Buffers, *Journal of Heuristics*.

Färber G., Salhi. S. and Coves A.M., Multistart Constrained-Logic-Programming for solving a Flowshop with Constrained Resequencing Buffers, *European Journal of Operational Research*.

Färber G. and Coves A.M., Analysis and Adjustment of a Genetic Algorithm for Non-Permutation Flowshops, *Annals of Operations Research*.

Färber G. and Coves A.M., Overview on Resequencing within Mixed Model Flowshop Production Lines, *International Journal of Production Economics*.

Färber G. and Coves A.M., Novel Classification for Flowshops with Resequencing, *International Journal of Production Research*.

Färber G., Salhi. S. and Coves A.M., Performance Evaluation of Hybrid-CLP vs. GA: Non-Permutation Flowshop with Constrained Resequencing Buffers .

9.3 Future work

The present thesis demonstrates the effectiveness of resequencing in flowshop production lines using constrained buffers. Concerning the described investigations, the following recommendations for improvements and considerations for future work may be given.

Related to including additional characteristics for the problem under study:

- Consideration of travel and handling times, occurring when a job is transferred to an offline buffer.
- Determination of the buffer location in the production line, together with the optimal size of the buffer places.
- Consideration of a cyclic product flow and the concatenation of successive clusters of products.
- Study of the dynamic case for the incoming jobs.

Related to the optimization methods and the possible reduction of the problem size:

- Comparison of the Genetic Algorithm with other heuristic methods as for example dispatching rules.
- Division of the problem to subproblems by splitting the production line into two parts in order to solve the first part and use its outgoing sequence as a fixed ingoing sequence for the second part of the production line.
- Combination of the degression for the genetic operator crossover with the progression of the mutation (section 6.5.3).

A Appendix

A.1 Nomenclature of parameters for flowshops (Alphabetically sorted)

Buffer: Buffers were originally introduced between two consecutive stations to decouple them in order to avoid blocking and starving. Buffers are often located before and after bottleneck stations. The reason is that this already critical part of the production usually is the limiting section. In automobile productions buffers of enormous dimensions can be found, which in principle decouple the main successive production sections. This buffer is, furthermore, used to reorder the jobs, available in the buffer, on a large scale.

Completion time c_j: The time job j exits the system is called completion time and is the completion time on the last station on which it requires processing.

Degree of automation: The degree of automation determines the level of automated performed operations at a station, compared to manually performed operations. The advantages of automating a station are manifold: the processing time is smaller, resulting in higher production rates and increased productivity and, what is more important for a smooth production, the processing time is constant; the product quality is higher due to high repeatability; the use of material in general is more efficient; factory lead times are reduced. Apart from productivity, there exists the opportunity to relieve humans from repetitive, hazardous, and unpleasant labor in all forms. Automating a station also has disadvantages which are to be taken into account when designing a station: changing the task is more difficult due to major changes in programming or tools; the investment is higher for design, fabrication and installation of an automated system; higher level of maintenance is required.

The constant processing time, which comes with an advanced degree of automation, results in the use of deterministic processing times, whereas, in the case that a human operator is involved, the processing times may urge for the use of stochastic processing times. The degree of automation can be partitioned into three levels:

> *Manually-operated stations*: Only simple machinery is used by a human operator. A large variety of operations or operations which are difficult to automate are performed, also depending on the skills of the operator.
>
> *Semi-automated stations*: The station is equipped with more complex machinery which performs some of the operations at the station automatically.
>
> *Fully-automated stations*: The station is equipped with machinery that, apart from the automated operations, automatically perform load and unload operations.

Demand D: The demand describes the total volume of N jobs to be processed. In the scheduling problem the individual jobs can furthermore be specified by start-date and due-date. These values describe the earliest possible point of time to start working on a particular job and when the finished products have to be delivered to the customer. In order to fulfill the due-dates a penalty may be applied for delivering too early or too late. In the sequencing problem it is frequent to release customer orders to production once a certain number of jobs has been accumulated, and then sequence and produce these orders together as a lot, see for example Burns and Daganzo, 1987. The demand in this case is a **static demand** and only depends on the volume of jobs and the objective usually is to minimize the processing time to complete the entire order, called makespan. Whereas a **dynamic demand** implies that the customer orders arrive continuously or at least are not completely determinable beforehand.

Deterministic-stochastic models: The deterministic model is characterized by the fact that the elements, e.g. processing time, do not involve variation and that the consequences of any given decision can be predicted in a precise manner. The stochastic model is characterized by its explicit recognition of variation and uncertainty, which could exist in one or more of the elements with known probabilistic behavior. This may result, for example, in a variation of the performance of the operator.

Job j: A part, subassembly or assembly, processed by a station is called job. In a mixed model production line the jobs belong to different models which include different processing times at the stations, depending on the model type. The number of jobs to be processed is N.

Job sequence Π_i: The job sequence defines the order of jobs at station i. A job sequence that is the same for all stations is called a **permutation** sequence. In flowshops the station sequence, the order in which the individual jobs visit the stations, is the same for all jobs.

Launch-interval λ: The time between two consecutive jobs entering the production line is called launch-interval. Usually it is a constant value, also called cycle time. A constant launch-interval results in a fixed production rate (production quantity per unit of time).

Machine breakdown/maintenance: Machine breakdown/maintenance describes the state of a station which does not permit processing of any job due to failure or failure prevention. In real production systems the breakdowns occur in a stochastic way and can be simulated using the values Mean-Time-Between-Failure (MTBF) and Mean-Time-To-Repair (MTTR).

Minimal-Part-Set (MPS): see Model.

Model M: In the mixed model flowshop several variations, called models, of the same basic product are manufactured. The difference from one model to another may be due to an option that is not applied to all models or likewise in a variation of an option. Therefore the mixed-model sequencing problem consists of the determination of the consecutive order of the models. M_i determines the model of job i. **Minimal-Part-Set** (MPS): The MPS is denoted by the vector $d(d_1, d_2, ..., d_k)$ which represents a product mix, such that $d_M = D_M/h$. D_M being the number of units of model type μ which needs to be assembled during an entire planning horizon and h being the greatest common divisor of $D_1, D_2, ..., D_M$. Obviously, h times repetition of the MPS sequence meets the total demand. With the MPS the number of possible sequences is reduced to $D!/(d_1! \cdot d_2! \cdot ... \cdot d_k!)$, Korkmazel and Meral, 2001. The MPS is considered to be a good choice, however, Klundert and Grigoriev, 2001, show that many times reducing the sequence to the one of the MPS does not result in the optimal sequence.

Operation i: Processing of a job in a station is called operation. This operation can include various performed tasks at one and the same station and is determined by the processing time P_{ij}.

Paced/unpaced production line: In a paced production line the mechanical material handling equipment like conveyor belts couple the stations in an inflexible manner. The jobs are either steadily moved from station to station at constant speed or they are immediately transferred after processing. The available amount of time for the operation is the same in both cases. In the unpaced line, in contrast, the stations are decoupled by buffers. In a specific case this buffer stores jobs that can not be passed to the downstream station which is still occupied with processing the previous job.

Precedence: The precedence gives a dependency of jobs in respect to the processing. A job j is said to be predecessor of job k if job j has to be processed before job k. An immediate predecessor then is a job that has to be processed immediately before another job.

Preemptive/Nonpreemptive: Preemptive operation means that processing times may be interrupted and resumed at a later time, even on another station. Furthermore an operation may be interrupted several times. If preemption is not allowed, the operation is called nonpreemptive.

Processing time P_{ij}: Also called assembly time, is the time that job j maintains at station i while being processed. Due to the nature of a flowshop, job j that is not processed at station i has to pass this station with a processing time equal to zero.

Rework operations: The detection of defective jobs may cause either rework operations or removal of the job from the line. The occurrence of defective jobs in the production is of probabilistic nature.

Setup cost SC_{efi}: In a similar way, setup cost is concerned if an additional cost appears to change the setup of station i, in order to be able to process job $j+1$ which is of model f after job j which is of model e. If the setup cost is independent of the model, it can be simply added to the processing cost.

Setup time ST_{efi}: Setup time is concerned if an additional time appears to change the setup of station i, in order to be able to process job $j+1$ which is of model f after job j which is of model e. If the setup time is independent of the model, it can be simply added to the processing time.

Start time s_j: The time job j enters the system is called start time.

Station i: One or more tasks may be assigned to station i. In the basic flowshop problem M stations are aligned in series and all jobs j have to pass the stations in the same order. A station may be open or closed, depending on whether or not the operator working in it is allowed to cross its boundary. In the case of a paced production line, the time to realize the assigned tasks at a station, is defined by the launch interval.

Task t: Non-divisible activity which is performed in either station. The balancing problem solves the problem of assigning tasks to stations and therefore often is called the assignment problem. In the Sequencing problem this task assignation is already performed and only stations are considered.

A.2 Experimentation CLP

Station 1

	Job-out			
Job-in	1	2	3	4
1	4	6	2	5
2	4	6	6	6
3	2	4	6	4
4	8	4	7	3

Station 2

	Job-out			
Job-in	1	2	3	4
1	7	3	7	6
2	6	7	5	3
3	3	7	4	3
4	5	6	6	3

Station 3

	Job-out			
Job-in	1	2	3	4
1	4	5	7	3
2	7	5	7	3
3	4	5	7	5
4	6	2	2	2

Station 4

	Job-out			
Job-in	1	2	3	4
1	8	3	3	2
2	8	3	2	8
3	4	7	3	7
4	3	3	2	5

Station 5

	Job-out			
Job-in	1	2	3	4
1	7	3	4	5
2	7	6	3	3
3	2	2	2	3
4	2	5	5	7

Station 6

	Job-out			
Job-in	1	2	3	4
1	6	2	7	6
2	3	3	3	8
3	8	5	3	5
4	4	6	4	6

Station 7

	Job-out			
Job-in	1	2	3	4
1	6	8	6	3
2	3	2	6	6
3	4	8	6	7
4	4	6	4	6

Station 8

	Job-out			
Job-in	1	2	3	4
1	7	5	5	3
2	3	3	2	5
3	5	5	3	2
4	4	6	6	4

Station 9

	Job-out			
Job-in	1	2	3	4
1	5	5	5	5
2	4	6	2	5
3	2	3	4	2
4	6	6	2	5

Station 10

	Job-out			
Job-in	1	2	3	4
1	5	7	4	6
2	5	6	6	7
3	6	7	7	8
4	2	6	2	3

Table A.1: Setup cost for 4 jobs on a 10 station flowshop.

Station 1

	Job-out			
Job-in	1	2	3	4
1	3	2	4	3
2	2	4	5	4
3	3	3	4	4
4	4	3	5	3

Station 2

	Job-out			
Job-in	1	2	3	4
1	5	2	1	5
2	3	1	4	3
3	1	1	3	2
4	2	2	2	5

Station 3

	Job-out			
Job-in	1	2	3	4
1	4	4	3	3
2	3	4	3	3
3	2	1	2	4
4	3	2	4	2

Station 4

	Job-out			
Job-in	1	2	3	4
1	2	1	5	4
2	4	5	5	3
3	3	2	2	4
4	4	3	4	3

Station 5

	Job-out			
Job-in	1	2	3	4
1	2	4	2	4
2	2	3	2	2
3	2	4	4	3
4	4	4	5	1

Station 6

	Job-out			
Job-in	1	2	3	4
1	4	1	4	2
2	2	2	3	3
3	2	3	1	3
4	1	4	3	3

Station 7

	Job-out			
Job-in	1	2	3	4
1	5	1	4	4
2	3	2	5	4
3	4	5	4	3
4	5	4	2	1

Station 8

	Job-out			
Job-in	1	2	3	4
1	3	4	3	2
2	3	4	4	5
3	4	3	5	3
4	2	3	3	2

Station 9

	Job-out			
Job-in	1	2	3	4
1	4	3	4	2
2	2	4	2	4
3	4	1	2	1
4	1	4	3	4

Station 10

	Job-out			
Job-in	1	2	3	4
1	1	3	4	3
2	1	3	3	4
3	3	4	1	2
4	2	2	4	3

Table A.2: Setup time for 4 jobs on a 10 station flowshop.

	Perm	Non-Perm						Job Changes					
Nr.		5s	10s	30s	60s	120s	300s	5s	10s	30s	60s	120s	300s
1	240,6	238,5	238,5	238,5	238,5	238,5	238,5	2	2	2	2	2	2
2	242,2	238,0	238,0	238,0	238,0	238,0	238,0	2	2	2	2	2	2
3	242,2	241,0	241,0	241,0	238,6	238,6	235,3	1	1	1	2	2	3
4	243,1	240,1	240,1	240,1	239,8	239,8	239,8	2	2	2	1	1	1
5	244,7	241,7	241,7	241,7	241,7	241,4	241,4	1	1	1	1	2	2
6	245,1	235,4	235,4	235,4	235,4	235,4	235,4	3	3	3	3	3	3
7	247,5	244,8	244,8	244,8	244,8	244,8	243,6	1	1	1	1	1	2
8	247,9	245,2	245,2	245,2	245,2	245,2	245,2	1	1	1	1	1	1
9	248,3	247,2	247,2	246,2	246,0	246,0	246,0	4	4	3	2	2	2
10	249,3	**249,9**	247,2	247,2	245,7	245,4	245,4	3	3	3	4	3	3
11	250,3	248,8	248,8	248,8	248,8	248,8	248,8	1	1	1	1	1	1
12	250,5	237,8	237,8	237,8	237,8	237,8	237,8	1	1	1	1	1	1
13	250,6	241,9	241,9	238,4	238,4	238,4	237,2	3	3	2	2	2	3
14	253,4	252,2	252,2	252,2	252,2	252,2	252,2	3	3	3	3	3	3
15	255,0	252,3	252,3	252,3	252,3	252,3	252,3	3	3	3	3	3	3
16	255,9	250,2	250,2	250,2	250,2	250,2	250,2	2	2	2	2	2	2
17	256,6	250,7	250,7	250,7	250,7	250,7	250,7	3	3	3	3	3	3
18	258,2	256,9	256,9	244,8	244,3	244,3	244,3	2	2	1	2	2	2
19	258,6	256,2	256,2	255,3	255,3	255,3	255,3	1	1	3	3	3	3
20	259,5	245,9	245,9	245,9	245,9	245,9	245,9	3	3	3	3	3	3
21	260,3	**260,3**	**260,3**	**260,3**	**260,3**	**260,3**	**260,3**	1	1	1	1	1	1
22	261,2	249,9	249,9	249,9	249,9	249,9	249,9	3	3	3	3	3	3
23	263,2	256,0	256,0	252,2	252,2	252,2	252,2	2	2	4	4	4	4
24	264,5	**264,8**	**264,8**	263,6	262,4	262,4	262,4	1	1	1	2	2	2
25	264,9	256,6	256,6	249,6	249,6	247,7	247,7	3	3	3	3	3	3
26	265,4	264,8	264,8	263,0	263,0	263,0	262,3	2	2	4	4	4	4
27	265,7	259,4	259,4	259,4	259,4	259,4	259,4	2	2	2	2	2	2
28	266,0	256,9	256,9	256,9	256,9	256,9	256,9	1	1	1	1	1	1
29	266,9	255,7	255,7	255,7	254,5	254,5	254,5	3	3	3	3	3	3
30	266,9	260,8	260,8	256,0	256,0	256,0	256,0	3	3	3	3	3	3
31	267,0	252,0	252,0	251,4	251,4	251,4	251,4	1	1	2	2	2	2
32	267,2	263,3	260,9	260,9	260,9	260,9	260,9	2	3	3	3	3	3
33	267,6	264,3	263,1	263,1	262,2	262,2	262,2	3	4	4	4	4	4
34	268,3	266,5	266,5	266,5	266,5	266,5	266,5	2	2	2	2	2	2
35	269,2	269,0	259,1	259,1	259,1	259,1	259,1	1	2	2	2	2	2
36	269,3	**269,3**	268,7	268,3	268,3	268,3	268,3	2	1	3	3	3	3
37	269,6	258,9	258,9	258,9	255,3	255,3	255,3	3	3	3	3	3	3
38	271,9	266,9	264,8	264,8	264,8	264,8	264,8	2	2	2	2	2	2
39	272,6	263,9	263,9	263,9	263,9	263,9	263,3	3	3	3	3	3	3
40	272,9	271,4	258,7	258,7	258,7	258,7	258,7	2	3	3	3	3	3
41	274,2	250,1	250,1	248,6	248,6	247,1	247,1	2	2	2	2	2	2
42	274,4	266,4	266,4	266,4	265,8	265,8	265,8	3	3	3	3	3	3
43	274,9	257,7	257,7	251,2	251,2	251,2	250,7	3	3	2	2	2	2
44	277,2	259,8	259,8	259,8	259,6	257,8	257,8	1	1	1	2	3	3
45	278,4	273,9	267,4	267,4	267,4	267,4	267,4	2	2	2	2	2	2
46	281,6	278,0	278,0	278,0	278,0	278,0	277,4	3	3	3	3	3	4
47	281,7	269,2	269,2	269,2	269,2	269,2	269,2	2	2	2	2	2	2
48	282,9	275,3	275,3	275,3	275,3	275,3	275,3	2	2	2	2	2	2
49	285,2	280,4	270,0	270,0	270,0	270,0	270,0	2	3	3	3	3	3
50	286,4	279,0	279,0	277,5	277,5	277,5	277,5	3	3	3	3	3	3

Table A.3: Preliminary analysis of 50 randomly generated permutation sequences. The cells with grey background show an improvement with respect to the same case in the previous column. Cells with bold text show that the permutation case is not improved. The data are ordered by increasing value of the solution for the permutation case.

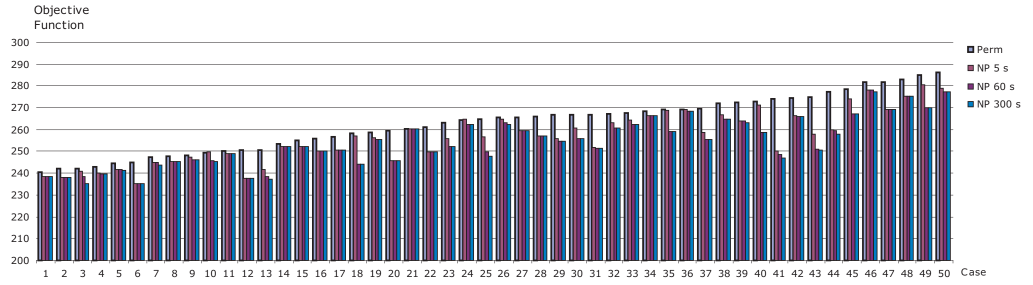


Figure A.1: Graphical representation. The case with resequencing tends to give better results but in some instances the solution of the permutation case is not be improved.

Seed	Iterations	R	time/It	t_{max}	A_{MSS}	A_{MJP}	A_{MSJ}
1	20	10	30	600	978,5	995,6	980,7
	20	5	30	600	988	983	993,1
	10	10	60	600	**970,6**	980,5	1002,7
	10	5	60	600	**960,5**	981,3	**973,2**
	5	10	120	600	977,4	1008,1	1008,1
2	20	10	30	600	978,5	980,7	982,2
	20	5	30	600	**961,1**	981,3	985,9
	10	10	60	600	977,4	981,3	988
	10	5	60	600	**961,1**	984,8	987,2
	5	10	120	600	985,9	**961,1**	1007,8
3	20	10	30	600	977	977	982,2
	20	5	30	600	**961,1**	980,5	985,9
	10	10	60	600	974,9	974,9	988
	10	5	60	600	980,5	984,8	988
	5	10	120	600	980,5	981,3	1014,3
4	20	10	30	600	978,5	978,5	980,8
	20	5	30	600	**961,1**	**961,1**	981,3
	10	10	60	600	**970,6**	984,8	977,4
	10	5	60	600	**968,7**	**961,1**	1003,8
	5	10	120	600	974,9	1003,8	1003,8
5	20	10	30	600	977	981,3	**973,2**
	20	5	30	600	**961,1**	1005	999,6
	10	10	60	600	**970,6**	**970,6**	**973,2**
	10	5	60	600	**960,5**	974,9	**961,1**
	5	10	120	600	**973,2**	1007,8	**973,2**

Table A.4: Different sets of parameters used for the adjustment of the arrangements A_{MSS}, A_{MJP} and A_{MSJ}. Each experiment is repeated five times, each time with a different seed for the randomly generated permutation sequences. Numbers in bold performed better than the arrangements A_P, A_{NP} and A_C (974,1).

Seed	Perm Sequ	R	time/It	t_{max}	A_M
1	100	20	30	600	978,2
	100	10	60	600	975,7
	100	5	120	600	1008,1
	50	20	30	600	978,2
	50	10	60	600	975,7
	50	5	120	600	1008,1
2	100	20	30	600	**961,1**
	100	10	60	600	**961,1**
	100	5	120	600	1005,3
	50	20	30	600	1010,2
	50	10	60	600	1010,2
	50	5	120	600	1010,2
3	100	20	30	600	999,4
	100	10	60	600	1007,8
	100	5	120	600	1007,8
	50	20	30	600	999,4
	50	10	60	600	995,1
	50	5	120	600	1014,3
4	100	20	30	600	**961,1**
	100	10	60	600	**960,5**
	100	5	120	600	**960,5**
	50	20	30	600	**961,1**
	50	10	60	600	**960,5**
	50	5	120	600	**960,5**
5	100	20	30	600	977
	100	10	60	600	**960,5**
	100	5	120	600	**960,5**
	50	20	30	600	977
	50	10	60	600	**973,2**
	50	5	120	600	**973,2**

Table A.5: Different sets of parameters used for the adjustment of the arrangement A_M. Each experiment is repeated five times, each time with a different seed for the randomly generated permutation sequences. Numbers in bold performed better than the arrangements A_P, A_{NP} and A_C (974,1).

4 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	481,2	481,2	480,0	477,7	477,7	480,0	480,0	488,6	480,0
	2				477,7	477,7	474,9	477,7	474,9	474,9
	3				477,7	474,9	474,9	474,9	528,7	474,9
	4				477,7	477,7	477,7	474,9	514,6	474,9
	5				480,0	477,7	480,0	480,0	514,6	480,0
	Avg	481,2	481,2	480,0	478,2	477,1	477,5	477,5	504,3	**476,9**
	1	0	0	1	2	2	1	1	1	1
	2				2	2	2	2	2	2
	3				2	2	2	2	0	2
	4				2	2	2	2	1	2
	5				1	2	1	1	1	1
	Avg	0	0	1	1,8	2	1,6	1,6	1	1,6

5 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	470,5	470,5	470,5	449,4	449,4	470,0	464,2	487,8	464,2
	2				464,2	464,2	467,6	464,2	500,3	464,2
	3				470,5	449,4	449,4	464,2	510,9	464,2
	4				449,4	449,4	467,6	478,9	499,8	478,9
	5				461,4	461,4	484,4	470,5	449,4	449,4
	Avg	470,5	470,5	470,5	459,0	**454,8**	467,8	468,4	489,6	464,2
	1	0	0	0	2	2	2	1	2	1
	2				1	1	1	1	2	1
	3				0	2	2	1	3	1
	4				2	2	1	0	1	0
	5				3	1	1	0	2	2
	Avg	0	0	0	1,6	1,6	1,4	0,6	2	1

6 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	564,6	564,6	563,4	564,9	558,3	558,3	537,5	588,5	537,5
	2				562,2	567,2	569,7	548,4	649,7	548,4
	3				559,4	563,3	534,7	559,4	623,9	559,4
	4				534,7	534,7	563,4	564,6	626,4	564,6
	5				562,2	541,1	562,2	575,6	626,7	575,6
	Avg	564,6	564,6	563,4	556,7	**552,9**	557,7	557,1	623,0	557,1
	1	0	0	3	4	4	4	2	2	2
	2				3	3	2	3	3	3
	3				2	3	2	2	3	3
	4				2	2	3	0	2	2
	5				3	2	3	2	2	2
	Avg	0	0	3	2,8	2,8	2,8	1,8	2,4	2,4

7 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	568,7	568,7	567,8	572,1	572,7	575,0	598,8	634,1	598,8
	2	-	-	-	569,3	569,3	583,1	583,1	670,5	583,1
	3	-	-	-	581,9	569,3	578,3	596,1	669,1	596,1
	4	-	-	-	592,5	581,7	571,2	569,3	640,0	569,3
	5	-	-	-	576,5	569,3	572,1	581,2	640,2	581,2
	Avg	568,7	568,7	**567,8**	578,5	572,5	575,9	585,7	650,8	585,7
	1	0	0	1	0	0	3	2	1	1
	2	-	-	-	0	0	0	0	3	3
	3	-	-	-	0	0	0	0	3	3
	4	-	-	-	0	0	3	0	1	1
	5	-	-	-	0	0	4	1	3	3
	Avg	0	0	1	0	0	2	0,6	2,2	2,2

8 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	589,6	622,1	589,6	637,4	619,0	632,7	645,1	691,4	645,1
	2				639,8	622,4	645,6	639,8	710,6	639,8
	3				642,4	609,3	632,8	642,4	654,5	642,4
	4				642,6	632,7	637,8	644,2	671,6	644,2
	5				633,9	633,0	641,2	641,4	669,5	641,4
	Avg	589,6	622,1	**589,6**	639,2	623,3	638,0	642,6	679,5	642,6
	1	0	3	0	0	0	0	0	6	6
	2				3	0	3	3	1	1
	3				0	2	0	0	2	2
	4				2	2	4	5	4	4
	5				1	1	0	0	4	4
	Avg	0	3	0	1,2	1	1,4	1,6	3,4	3,4

Table A.6: Comparison of the proposed arrangements for the intermediate case. The solutions which perform better than the best found permutation sequence is highlighted with a grey background and the best solution of each line is highlighted using a bold font.

4 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	481,2	481,2	481,2	477,7	477,7	480,0	480,0	488,6	480,0
	2	-	-	-	477,7	477,7	476,1	477,7	476,1	476,1
	3	-	-	-	477,7	476,1	476,1	476,1	528,8	476,1
	4	-	-	-	477,7	477,7	477,7	476,1	514,6	476,1
	5	-	-	-	480,0	477,7	480,0	480,0	514,6	480,0
	Avg	481,2	481,2	481,2	478,2	**477,4**	478,0	478,0	504,5	477,7
	1	0	0	0	2	2	1	1	1	1
	2	-	-	-	2	2	2	2	1	2
	3	-	-	-	2	1	1	1	0	2
	4	-	-	-	2	2	1	1	1	2
	5	-	-	-	1	2	1	1	1	1
	Avg	0	0	0	1,8	1,8	1,2	1,2	0,8	1,6

5 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	470,5	470,5	470,5	449,4	449,4	470,0	464,2	487,8	464,2
	2	-	-	-	464,2	464,2	467,6	464,2	500,3	464,2
	3	-	-	-	470,5	449,4	449,4	464,2	510,9	464,2
	4	-	-	-	449,4	449,4	467,6	478,9	499,8	478,9
	5	-	-	-	461,4	461,4	484,4	470,5	449,4	449,4
	Avg	470,5	470,5	470,5	459,0	**454,8**	467,8	468,4	489,6	464,2
	1	0	0	0	2	2	2	1	2	1
	2	-	-	-	1	1	1	1	2	1
	3	-	-	-	0	2	2	1	3	1
	4	-	-	-	2	2	1	0	1	0
	5	-	-	-	3	3	1	0	2	2
	Avg	0	0	0	1,6	2	1,4	0,6	2	1

6 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	564,6	564,6	564,0	564,9	562,7	560,2	537,5	586,9	537,5
	2	-	-	-	562,2	567,2	572,0	548,6	651,2	548,6
	3	-	-	-	559,4	563,9	534,7	559,4	597,8	559,4
	4	-	-	-	534,7	534,7	564,0	564,6	601,7	564,6
	5	-	-	-	562,2	541,1	562,2	562,2	632,9	562,2
	Avg	564,6	564,6	564,0	556,7	**553,9**	558,6	554,5	614,1	554,5
	1	0	0	2	4	3	1	1	2	2
	2	-	-	-	3	3	2	2	2	3
	3	-	-	-	2	1	2	3	2	3
	4	-	-	-	2	2	2	0	4	2
	5	-	-	-	3	2	3	2	2	2
	Avg	0	0	2	2,8	2,2	2	1,6	2,4	2,4

7 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	568,7	568,7	567,5	568,6	568,5	576,5	578,4	618,5	578,4
	2	-	-	-	577,4	568,4	583,1	583,1	607,7	583,1
	3	-	-	-	577,2	567,3	574,7	586,2	588,4	586,2
	4	-	-	-	568,4	565,6	571,2	568,4	606,8	568,4
	5	-	-	-	571,2	568,4	567,3	570,6	601,6	570,6
	Avg	568,7	568,7	**567,5**	572,6	567,6	574,6	577,3	604,6	577,3
	1	0	0	2	2	1	0	2	2	1
	2	-	-	-	3	4	0	0	2	3
	3	-	-	-	2	4	2	2	2	3
	4	-	-	-	4	3	3	4	1	1
	5	-	-	-	2	4	3	3	4	3
	Avg	0	0	2	2,6	3,2	1,6	2,2	2,2	2,2

8 Jobs	Seed	A_P	A_{NP}	A_C	A_M	A_{MSS}	A_{MJP}	A_{MSJ} (I)	A_{MSJ} (II)	A_{MSJ}
	1	589,6	589,6	589,6	633,8	619,0	624,3	624,3	645,9	624,3
	2	-	-	-	640,1	615,4	623,1	633,6	707,3	633,6
	3	-	-	-	600,2	609,3	615,0	615,0	653,3	615,0
	4	-	-	-	629,5	617,6	639,6	644,2	673,7	644,2
	5	-	-	-	625,5	630,9	607,8	607,8	652,0	607,8
	Avg	589,6	589,6	**589,6**	625,8	618,4	622,0	625,0	666,4	625,0
	1	0	0	0	2	0	1	1	3	6
	2	-	-	-	2	4	2	3	0	1
	3	-	-	-	2	2	3	3	2	2
	4	-	-	-	3	2	4	4	3	4
	5	-	-	-	2	3	2	3	4	4
	Avg	0	0	0	2,2	2,2	2,4	2,8	2,4	3,4

Table A.7: Comparison of the proposed arrangements for the centralized case. The solutions which perform better than the best found permutation sequence is highlighted with a grey background and the best solution of each line is highlighted using a bold font.

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
7 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	568,7	568,7	567,8	572,7	568,7	566,6	568,5	568,7	566,6	568,5	568,7	566,6	569,4
	2	-	-	-	569,3	-	-	563,8	-	-	563,8	-	-	568,7
	3	-	-	-	569,3	-	-	565,5	-	-	564,3	-	-	569,3
	4	-	-	-	581,7	-	-	567,8	-	-	563,8	-	-	569,3
	5	-	-	-	569,3	-	-	566,6	-	-	568,5	-	-	568,7
	Avg	568,7	568,7	567,8	572,5	568,7	566,6	566,4	568,7	566,6	**565,8**	568,7	566,6	569,1
	1	0	0	3	0	0	3	1	0	3	1	0	3	3
	2	-	-	-	0	-	-	4	-	-	4	-	-	0
	3	-	-	-	0	-	-	3	-	-	5	-	-	0
	4	-	-	-	0	-	-	2	-	-	3	-	-	0
	5	-	-	-	0	-	-	3	-	-	3	-	-	0
	Avg	0	0	3	0	0	3	2,6	0	3	3,2	0	3	0,6

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
8 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	589,6	622,1	589,6	619,0	589,6	589,6	596,2	621,6	589,6	588,7	589,6	591,7	596,2
	2	-	-	-	622,4	-	-	596,6	-	-	590,2	-	-	596,6
	3	-	-	-	609,3	-	-	604,9	-	-	590,9	-	-	604,9
	4	-	-	-	632,7	-	-	589,6	-	-	584,2	-	-	589,6
	5	-	-	-	633,0	-	-	607,8	-	-	591,7	-	-	607,8
	Avg	589,6	622,1	589,6	623,3	589,6	589,6	599,0	621,6	589,6	**589,1**	589,6	591,7	599,0
	1	0	3	0	0	0	0	0	0	0	4	0	2	2
	2	-	-	-	3	-	-	0	-	-	3	-	-	0
	3	-	-	-	0	-	-	0	-	-	2	-	-	0
	4	-	-	-	2	-	-	0	-	-	5	-	-	0
	5	-	-	-	1	-	-	0	-	-	3	-	-	0
	Avg	0	3	0	1,2	0	0	0	0	0	3,4	0	2	0,4

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
9 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	684,3	694,5	732,2	697,6	694,5	692,6	697,6	711,1	684,3	685,9	694,5	684,0	697,6
	2	-	-	-	690,7	-	-	690,7	-	-	683,2	-	-	688,4
	3	-	-	-	690,8	-	-	690,9	-	-	686,6	-	-	690,6
	4	-	-	-	695,4	-	-	695,4	-	-	681,8	-	-	695,4
	5	-	-	-	704,8	-	-	696,8	-	-	690,2	-	-	698,3
	Avg	684,3	694,5	732,2	695,9	694,5	692,6	694,3	711,1	684,3	685,5	694,5	**684,0**	694,1
	1		0	0	0	0	3	0	0	0	3	0	3	0
	2	-	-	-	0	-	-	0	-	-	1	-	-	3
	3	-	-	-	2	-	-	0	-	-	2	-	-	4
	4	-	-	-	0	-	-	0	-	-	3	-	-	0
	5	-	-	-	5	-	-	4	-	-	4	-	-	2
	Avg	#¡DIV/0!	0	0	1,4	0	3	0,8	0	0	2,6	0	3	1,8

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
10 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	736,1	755,8	755,1	754,6	752,3	813,4	749,8	755,8	746,1	749,8	752,3	748,1	749,8
	2	-	-	-	728,2	-	-	739,2	-	-	735,9	-	-	739,2
	3	-	-	-	754,6	-	-	754,6	-	-	753,7	-	-	753,6
	4	-	-	-	750,8	-	-	750,8	-	-	750,8	-	-	750,8
	5	-	-	-	742,4	-	-	742,4	-	-	742,4	-	-	742,4
	Avg	736,1	755,8	755,1	746,1	752,3	813,4	747,4	755,8	746,1	746,5	752,3	748,1	747,2
	1	0	0	5	0	0	7	0	0	3	0	0	5	0
	2	-	-	-	0	-	-	0	-	-	5	-	-	0
	3	-	-	-	0	-	-	0	-	-	6	-	-	0
	4	-	-	-	0	-	-	0	-	-	0	-	-	0
	5	-	-	-	0	-	-	0	-	-	0	-	-	0
	Avg	0	0	5	0	0	7	0	0	3	2,2	0	5	0

Table A.8: Limited flexibility for the intermediate case. The solutions which perform better than the best found permutation sequence is highlighted with a grey background and the best solution of each line is highlighted using a bold font.

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
7 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	568,7	568,7	567,5	568,5	568,7	567,5	568,5	572,7	567,5	568,5	568,7	567,5	568,5
	2	-	-	-	568,4	-	-	563,8	-	-	563,8	-	-	563,8
	3	-	-	-	567,3	-	-	567,0	-	-	569,3	-	-	568,3
	4	-	-	-	565,6	-	-	567,8	-	-	565,8	-	-	566,7
	5	-	-	-	568,4	-	-	567,5	-	-	567,5	-	-	568,4
	Avg	568,7	568,7	567,5	567,6	568,7	567,5	**566,9**	572,7	567,5	567,0	568,7	567,5	567,1
	1	0	0	2	1	0	2	1	0	2	1	0	2	1
	2	-	-	-	4	-	-	4	-	-	4	-	-	4
	3	-	-	-	4	-	-	3	-	-	0	-	-	4
	4	-	-	-	3	-	-	2	-	-	3	-	-	2
	5	-	-	-	4	-	-	2	-	-	2	-	-	4
	Avg	0	0	2	3,2	0	2	2,4	0	2	2	0	2	3

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
8 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	589,6	589,6	589,6	619,0	589,6	589,6	588,7	621,6	589,6	588,7	621,6	589,6	588,7
	2	-	-	-	615,4	-	-	587,2	-	-	590,2	-	-	587,2
	3	-	-	-	609,3	-	-	590,9	-	-	590,9	-	-	588,7
	4	-	-	-	617,6	-	-	586,9	-	-	586,9	-	-	586,9
	5	-	-	-	630,9	-	-	588,7	-	-	590,2	-	-	588,7
	Avg	589,6	589,6	589,6	618,4	589,6	589,6	588,5	621,6	589,6	589,4	621,6	589,6	**588,0**
	1	0	0	0	0	0	0	4	0	0	4	0	0	4
	2	-	-	-	4	-	-	2	-	-	3	-	-	2
	3	-	-	-	2	-	-	2	-	-	2	-	-	3
	4	-	-	-	2	-	-	4	-	-	4	-	-	4
	5	-	-	-	3	-	-	3	-	-	2	-	-	3
	Avg	0	0	0	2,2	0	0	3	0	0	3	0	0	3,2

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
9 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	684,3	694,5	682,5	697,6	694,5	683,4	687,4	711,1	680,7	688,9	694,5	682,5	697,6
	2	-	-	-	685,7	-	-	684,1	-	-	685,9	-	-	685,7
	3	-	-	-	681,5	-	-	688,7	-	-	678,0	-	-	684,3
	4	-	-	-	695,4	-	-	687,9	-	-	681,8	-	-	684,3
	5	-	-	-	697,1	-	-	696,5	-	-	689,3	-	-	696,8
	Avg	684,3	694,5	682,5	691,5	694,5	683,4	688,9	711,1	**680,7**	684,8	694,5	682,5	689,7
	1	0	0	2	0	0	2	2	0	3	3	0	2	0
	2	-	-	-	3	-	-	1	-	-	4	-	-	3
	3	-	-	-	3	-	-	3	-	-	3	-	-	2
	4	-	-	-	0	-	-	1	-	-	3	-	-	4
	5	-	-	-	3	-	-	2	-	-	4	-	-	4
	Avg	0	0	2	1,8	0	2	1,8	0	3	3,4	0	2	2,6

			No Limitation			W_{up}= 2; W_{down} = 2			W_{up}= 5; W_{down} = 1			W_{up}= 1; W_{down} = 5		
10 jobs	Seed	A_P	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}	A_{NP}	A_C	A_{MSS}
	1	736,1	752,3	735,1	737,2	752,3	735,5	741,9	755,8	735,5	738,4	742,4	735,2	740,1
	2	-	-	-	737,7	-	-	734,4	-	-	732,9	-	-	733,8
	3	-	-	-	754,6	-	-	751,7	-	-	744,9	-	-	751,1
	4	-	-	-	746,2	-	-	748,0	-	-	732,6	-	-	747,7
	5	-	-	-	742,4	-	-	742,4	-	-	735,6	-	-	740,3
	Avg	736,1	752,3	**735,1**	743,6	752,3	735,5	743,7	755,8	735,5	736,9	742,4	735,2	742,6
	1	0	0	2	4	0	1	4	0	1	3	0	2	3
	2	-	-	-	3	-	-	4	-	-	3	-	-	4
	3	-	-	-	0	-	-	3	-	-	4	-	-	4
	4	-	-	-	2	-	-	2	-	-	6	-	-	3
	5	-	-	-	0	-	-	3	-	-	3	-	-	2
	Avg	0	0	2	1,8	0	1	3,2	0	1	3,8	0	2	3,2

Table A.9: Limited flexibility for the centralized case. The solutions which perform better than the best found permutation sequence is highlighted with a grey background and the best solution of each line is highlighted using a bold font.

A.3 Heuristic Approach: Genetic Algorithm

A.3.1 Preliminary analysis of parameters

	Crossover-I ($p_{C\text{-}I}$)										
Crossover-II ($p_{C\text{-}II}$)	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
0,0	153,6	153,4	152,0	149,4	148,2	147,7	149,6	150,0	151,2	150,8	151,0
0,1	153,0	152,4	149,7	148,5	148,8	149,8	150,5	151,0	151,1	150,6	153,6
0,2	152,8	150,1	149,4	148,5	149,8	150,1	150,7	149,2	150,9	153,6	153,6
0,3	151,4	150,0	149,4	149,8	149,5	150,5	148,4	150,3	153,6	153,6	153,6
0,4	149,4	148,7	149,5	150,0	150,6	150,8	149,7	153,6	153,6	153,6	153,6
0,5	148,7	149,2	150,6	150,5	151,0	150,1	153,6	153,6	153,6	153,6	153,6
0,6	150,2	150,3	151,1	149,6	151,1	153,6	153,6	153,6	153,6	153,6	153,6
0,7	149,4	148,3	149,6	151,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6
0,8	149,1	151,2	151,3	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6
0,9	151,2	151,7	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6
1,0	150,4	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6	153,6

Table A.10: Details on the preliminary analysis: $p_{\text{c-I}}$ versus $p_{\text{c-II}}$. The sum ($p_{\text{c-I}}+p_{\text{c-II}}$) may not exceed 1.0. In order to be able to plot the graph, the numbers with grey background are complemented values.

	Mutation-I(f) ($p_{m\text{-}I(f)}$)										
Mutation-I(b) ($p_{m\text{-}I(b)}$)	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
0,0	149,8	148,7	150,2	148,2	149,0	146,8	147,6	148,9	149,6	151,9	152,3
0,1	150,0	149,0	149,0	148,8	148,1	147,7	148,1	149,1	150,5	150,8	152,5
0,2	149,2	147,9	146,6	148,1	146,9	149,0	148,6	150,2	153,2	151,8	152,1
0,3	148,0	146,9	146,9	147,8	148,0	149,0	148,9	152,1	152,2	153,3	153,7
0,4	147,9	148,4	147,6	147,7	149,1	150,7	150,7	152,3	153,6	151,4	153,4
0,5	147,9	148,2	147,7	149,1	152,0	151,8	151,2	153,0	154,1	153,5	152,8
0,6	149,5	149,4	149,4	150,4	152,8	152,8	152,0	153,1	153,0	152,7	151,5
0,7	150,9	151,0	150,8	151,9	153,0	153,1	153,8	152,8	154,3	152,5	154,1
0,8	151,9	152,2	153,0	153,6	153,1	152,8	152,2	154,9	153,3	154,3	153,8
0,9	152,7	152,4	153,7	153,4	152,8	154,3	154,2	153,8	153,4	153,9	152,5
1,0	152,8	152,8	153,1	152,2	153,4	153,4	153,4	153,1	154,5	152,9	153,6

Table A.11: Details on the preliminary analysis: $p_{\text{m-I(f)}}$ versus $p_{\text{m-I(b)}}$

	Mutation-I(f) ($p_{m\text{-}I(f)}$)										
Mutation-II ($p_{m\text{-}II}$)	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
0,0	148,8	147,5	149,8	147,6	148,3	148,0	147,9	149,5	149,9	152,3	152,4
0,1	149,2	147,9	146,6	148,1	146,9	149,0	148,6	150,2	153,2	151,8	152,1
0,2	149,4	147,4	146,9	148,2	148,8	149,2	150,4	151,9	152,9	153,9	153,3
0,3	148,0	148,2	148,8	149,8	148,8	151,9	151,9	152,0	152,7	152,5	154,3
0,4	147,8	149,2	149,5	148,8	151,4	152,2	152,6	153,9	154,3	153,2	152,8
0,5	150,1	149,5	150,1	151,3	152,1	152,3	152,5	152,1	153,4	153,9	151,8
0,6	150,4	151,7	151,7	153,0	153,7	154,5	153,3	152,4	153,6	152,9	153,8
0,7	151,9	151,8	151,4	153,1	153,9	151,2	154,2	153,4	154,2	153,2	153,9
0,8	153,2	152,4	152,5	153,0	153,1	153,5	154,8	153,3	153,5	154,3	152,5
0,9	153,4	152,7	154,9	154,1	153,5	153,7	153,9	153,2	153,6	153,4	153,3
1,0	152,3	151,8	154,1	152,9	153,3	152,3	153,4	152,7	153,8	153,3	153,7

Table A.12: Details on the preliminary analysis: $p_{\text{m-I(f)}}$ versus $p_{\text{m-II}}$

A.3.2 Performance analysis

	Absolut time		Relative Time	
M	Abort-300	Abort-No	Abort-300	Abort-No
5	1,47	**3,14**	0,47	**1,00**
10	2,94	6,08	0,94	1,94
15	3,45	8,94	1,10	2,85
20	4,33	11,78	1,38	3,75
25	5,84	14,67	1,86	4,67
30	5,64	17,64	1,80	5,62
35	11,69	20,41	3,72	6,50
40	8,16	23,25	2,60	7,40
45	8,78	26,20	2,80	8,35
50	15,92	29,06	5,07	9,26
55	11,67	31,88	3,72	10,15
60	11,67	34,69	3,72	11,05
65	12,59	37,74	4,01	12,02
70	13,30	40,74	4,24	12,97
75	13,80	43,49	4,39	13,85
80	15,63	46,39	4,98	14,77
85	20,47	49,57	6,52	15,79
90	18,67	52,22	5,95	16,63
95	28,31	55,39	9,02	17,64
100	21,00	57,97	6,69	18,46

Table A.13: Number of stations (M).

	Absolut time		Relative Time	
L	Abort-300	Abort-No	Abort-300	Abort-No
0	3,69	**11,78**	0,31	**1,00**
1	4,55	12,17	0,39	1,03
2	5,09	12,42	0,43	1,05
3	9,53	12,74	0,81	1,08
4	6,22	13,05	0,53	1,11
5	6,92	13,31	0,59	1,13
6	6,91	13,44	0,59	1,14
7	5,55	13,63	0,47	1,16
8	6,88	13,86	0,58	1,18
9	9,47	14,06	0,80	1,19
10	10,50	14,25	0,89	1,21
11	12,33	14,47	1,05	1,23
12	7,28	14,61	0,62	1,24
13	7,36	14,72	0,62	1,25
14	7,31	14,97	0,62	1,27
15	14,41	15,30	1,22	1,30
16	9,74	15,59	0,83	1,32
17	14,59	15,83	1,24	1,34
18	15,91	15,92	1,35	1,35
19	11,11	16,25	0,94	1,38

Table A.14: Number of resequencing possibilities (L).

	Absolut time		Relative Time	
N	Abort-300	Abort-No	Abort-300	Abort-No
5	0,42	**1,41**	0,30	**1,00**
10	1,73	3,16	1,23	2,24
15	1,81	5,58	1,29	3,97
20	6,73	8,63	4,79	6,14
25	5,42	12,28	3,86	8,74
30	8,06	16,58	5,73	11,79
35	21,28	21,55	15,14	15,33
40	16,19	27,18	11,51	19,33
45	22,75	33,46	16,18	23,80
50	37,44	40,44	26,63	28,76
55	39,16	47,99	27,85	34,13
60	27,36	56,43	19,46	40,13
65	62,66	65,76	44,57	46,77
70	74,88	75,59	53,26	53,76
75	57,44	86,18	40,85	61,30
80	57,83	97,34	41,13	69,23
85	90,27	110,34	64,21	78,48
90	73,98	123,22	52,61	87,64
95	117,43	137,38	83,52	97,71
100	131,67	153,13	93,65	108,91

Table A.15: Number of jobs (N).

	Absolut time		Relative Time	
R	Abort-300	Abort-No	Abort-300	Abort-No
5	2,47	**3,84**	0,64	**1,00**
10	7,56	7,80	1,97	2,03
15	6,17	11,77	1,61	3,06
20	8,86	15,72	2,31	4,09
25	13,34	19,75	3,47	5,14
30	16,13	23,75	4,20	6,18
35	19,24	27,80	5,01	7,23
40	17,24	31,94	4,48	8,31
45	26,33	36,10	6,85	9,39
50	37,44	40,31	9,74	10,49
55	42,42	44,49	11,04	11,58
60	27,77	48,75	7,23	12,69
65	32,61	53,07	8,49	13,81
70	57,39	57,39	14,93	14,93
75	44,25	61,78	11,51	16,08
80	34,88	66,39	9,08	17,28
85	47,52	70,82	12,36	18,43
90	51,16	75,47	13,31	19,64
95	51,31	79,77	13,35	20,76
100	79,00	84,27	20,56	21,93

Table A.16: Number of parents (R).

	Absolut time		Relative Time	
G	Abort-300	Abort-No	Abort-300	Abort-No
250	10,12	**10,29**	0,98	**1,00**
500	20,18	21,07	1,96	2,05
750	30,23	30,97	2,94	3,01
1000	37,45	40,97	3,64	3,98
1250	37,47	51,28	3,64	4,98
1500	37,45	61,49	3,64	5,97
1750	37,43	74,96	3,64	7,28
2000	37,45	81,00	3,64	7,87
2250	37,43	91,13	3,64	8,86
2500	37,43	101,15	3,64	9,83
2750	37,44	110,88	3,64	10,77
3000	37,44	121,06	3,64	11,76
3250	37,41	133,38	3,64	12,96
3500	37,43	144,27	3,64	14,02
3750	37,43	154,05	3,64	14,97
4000	37,43	160,94	3,64	15,64
4250	37,43	171,04	3,64	16,62
4500	37,41	181,05	3,64	17,59
4750	37,44	191,04	3,64	18,56
5000	37,42	201,35	3,64	19,57

Table A.17: Number of generations (G).

	Absolut time		Relative Time	
Number	Abort-300	Abort-No	Abort-300	Abort-No
1	12,44	**41,09**	0,30	**1,00**
2	12,34	40,97	0,30	1,00
3	12,20	40,49	0,30	0,99
4	12,22	40,53	0,30	0,99
5	12,69	40,50	0,31	0,99
6	14,56	40,50	0,35	0,99
7	19,74	40,56	0,48	0,99
8	13,39	40,58	0,33	0,99
9	14,94	40,55	0,36	0,99
10	20,59	40,77	0,50	0,99
11	20,11	40,61	0,49	0,99
12	18,70	40,80	0,46	0,99
13	21,52	40,72	0,52	0,99
14	31,17	40,83	0,76	0,99
15	40,38	40,83	0,98	0,99
16	21,03	40,80	0,51	0,99
17	20,03	40,80	0,49	0,99
18	35,67	40,84	0,87	0,99
19	30,34	40,99	0,74	1,00
20	20,41	40,92	0,50	1,00

Table A.18: Number of types of models.

Data	Obj Func	Probability
1	494,10	0,009
2	497,70	0,074
3	500,80	0,135
4	509,00	0,003
5	502,10	0,125
6	505,50	0,041
7	496,40	0,042
8	504,40	0,068
9	502,90	0,108
10	500,70	0,134
11	503,60	0,090
12	507,00	0,017
13	500,30	0,132
14	498,90	0,106
15	507,50	0,012
16	497,90	0,080
17	495,90	0,032
18	493,60	0,006
19	502,90	0,108
20	495,60	0,027
21	499,70	0,124
22	499,40	0,118
23	501,40	0,133
24	500,00	0,128
25	499,10	0,111
26	503,00	0,105
27	503,00	0,105
28	500,20	0,131
29	497,00	0,056
30	498,70	0,101
31	502,60	0,115
32	497,40	0,066
33	499,70	0,124
34	497,80	0,077
35	497,00	0,056
36	502,00	0,126
37	497,10	0,058
38	503,20	0,100
39	502,70	0,113
40	502,90	0,108
41	500,70	0,134
42	500,80	0,135
43	500,50	0,133
44	509,10	0,003
45	498,20	0,088
46	501,90	0,128
47	503,00	0,105
48	500,60	0,134
49	500,90	0,135
50	499,60	0,122
51	499,70	0,124
52	496,60	0,046
53	502,10	0,125
54	501,20	0,134
55	496,20	0,038
56	496,90	0,053
57	502,40	0,119
58	500,30	0,132
59	498,30	0,091
60	504,30	0,071
61	500,80	0,135
62	502,00	0,126
63	503,90	0,081
64	499,40	0,118
65	499,00	0,109
66	502,20	0,123
67	499,80	0,125
68	501,90	0,128
69	499,00	0,109
70	498,90	0,106
71	497,00	0,056
72	501,00	0,135
73	503,10	0,103
74	501,90	0,128
75	500,80	0,135
76	503,60	0,090
77	502,20	0,123
78	501,00	0,135
79	500,40	0,132
80	499,40	0,118
81	503,70	0,087
82	501,10	0,134
83	504,40	0,068
84	504,80	0,057
85	498,60	0,099
86	500,90	0,135
87	504,20	0,073
88	500,60	0,134
89	505,60	0,039
90	505,80	0,035
91	503,00	0,105
92	499,50	0,120
93	504,50	0,065
94	499,20	0,114
95	499,60	0,122
96	498,40	0,094
97	500,70	0,134
98	499,50	0,120
99	504,10	0,076
100	503,50	0,092

Table A.19: Variability of solutions: Permutation case.

Data	Obj Func	Probability
1	486,90	0,070
2	488,30	0,087
3	498,10	0,016
4	487,90	0,083
5	478,00	0,001
6	488,50	0,089
7	487,80	0,081
8	490,70	0,096
9	484,10	0,032
10	491,10	0,094
11	487,40	0,077
12	496,00	0,036
13	489,20	0,094
14	485,80	0,055
15	490,10	0,097
16	491,70	0,090
17	490,50	0,096
18	492,70	0,080
19	489,60	0,096
20	483,10	0,022
21	493,40	0,072
22	494,00	0,063
23	499,20	0,009
24	498,40	0,014
25	486,30	0,062
26	488,00	0,084
27	493,40	0,072
28	490,10	0,097
29	486,10	0,059
30	484,00	0,031
31	491,00	0,095
32	489,90	0,096
33	495,30	0,045
34	490,40	0,096
35	491,50	0,092
36	492,90	0,078
37	490,00	0,096
38	497,10	0,024
39	489,30	0,094
40	495,00	0,049
41	486,60	0,066
42	489,60	0,096
43	489,80	0,096
44	498,50	0,013
45	491,70	0,090
46	495,40	0,044
47	490,30	0,097
48	492,90	0,078
49	490,10	0,097
50	486,30	0,062
51	491,00	0,095
52	495,50	0,043
53	495,10	0,048
54	490,50	0,096
55	494,20	0,061
56	491,00	0,095
57	487,20	0,074
58	486,60	0,066
59	481,60	0,011
60	484,30	0,035
61	488,00	0,084
62	484,20	0,034
63	496,10	0,035
64	488,60	0,090
65	488,60	0,090
66	485,50	0,050
67	484,90	0,042
68	491,40	0,093
69	489,00	0,093
70	486,50	0,065
71	489,60	0,096
72	487,00	0,071
73	484,50	0,037
74	497,70	0,019
75	489,00	0,093
76	488,80	0,091
77	489,50	0,095
78	488,80	0,091
79	486,60	0,066
80	488,80	0,091
81	486,70	0,067
82	489,50	0,095
83	489,00	0,093
84	494,00	0,063
85	488,80	0,091
86	487,40	0,077
87	495,10	0,048
88	491,80	0,090
89	495,50	0,043
90	496,40	0,031
91	486,90	0,070
92	488,30	0,087
93	483,90	0,030
94	494,30	0,059
95	496,40	0,031
96	489,90	0,096
97	496,90	0,026
98	491,90	0,089
99	488,20	0,086
100	495,70	0,040

Table A.20: Variability of solutions: Non-permutation case.

Data	Obj Func	Jobchanges
1	486,90	12
2	488,30	5
3	498,10	2
4	487,90	17
5	478,00	19
6	488,50	14
7	487,80	13
8	490,70	11
9	484,10	13
10	491,10	9
11	487,40	15
12	496,00	12
13	489,20	11
14	485,80	12
15	490,10	15
16	491,70	10
17	490,50	9
18	492,70	2
19	489,60	14
20	483,10	10
21	493,40	11
22	494,00	10
23	499,20	5
24	498,40	5
25	486,30	12
26	488,00	12
27	493,40	10
28	490,10	13
29	486,10	9
30	484,00	17
31	491,00	10
32	489,90	11
33	495,30	5
34	490,40	9
35	491,50	7
36	492,90	9
37	490,00	7
38	497,10	6
39	489,30	9
40	495,00	10
41	486,60	9
42	489,60	10
43	489,80	10
44	498,50	6
45	491,70	11
46	495,40	11
47	490,30	15
48	492,90	11
49	490,10	11
50	486,30	12
51	491,00	6
52	495,50	7
53	495,10	8
54	490,50	11
55	494,20	5
56	491,00	9
57	487,20	8
58	486,60	12
59	481,60	14
60	484,30	21
61	488,00	9
62	484,20	15
63	496,10	13
64	488,60	9
65	488,60	8
66	485,50	15
67	484,90	18
68	491,40	6
69	489,00	12
70	486,50	9
71	489,60	8
72	487,00	14
73	484,50	14
74	497,70	6
75	489,00	13
76	488,80	15
77	489,50	8
78	488,80	12
79	486,60	10
80	488,80	8
81	486,70	14
82	489,50	6
83	489,00	11
84	494,00	7
85	488,80	10
86	487,40	10
87	495,10	7
88	491,80	10
89	495,50	12
90	496,40	6
91	486,90	13
92	488,30	11
93	483,90	16
94	494,30	11
95	496,40	6
96	489,90	6
97	496,90	6
98	491,90	9
99	488,20	11
100	495,70	9

Table A.21: Variability of solutions: Number of jobchanges for non-permutation case.

A.4 Experimentation Genetic Algorithm

A.4.1 Setup considerations

				Permutation				Non-permutation					
Case	Setup	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
1	NSC, NST	30	5	410,0	0	0	**410,0**	405,0	0	0	1,0	**405,0**	**1,22%**
1	NSC, NST	30	10	468,0	0	0	**468,0**	466,8	0	0	2,4	**466,8**	**0,27%**
1	NSC, NST	30	15	541,0	0	0	**541,0**	538,5	0	0	2,6	**538,5**	**0,46%**
1	NSC, NST	30	20	605,6	0	0	**605,6**	599,0	0	0	7,0	**599,0**	**1,10%**
1	NSC, NST	30	25	684,1	0	0	**684,1**	678,6	0	0	8,6	**678,6**	**0,80%**
1	NSC, NST	30	30	748,9	0	0	**748,9**	743,5	0	0	9,9	**743,5**	**0,72%**
1	NSC, NST	30	35	814,8	0	0	**814,8**	803,4	0	0	14,4	**803,4**	**1,40%**
1	NSC, NST	30	40	884,4	0	0	**884,4**	873,8	0	0	17,9	**873,8**	**1,20%**
1	NSC, NST	30	45	953,5	0	0	**953,5**	947,1	0	0	17,4	**947,1**	**0,67%**
1	NSC, NST	30	50	1011,6	0	0	**1011,6**	998,6	0	0	22,6	**998,6**	**1,28%**
2	NSC, ST	30	5	421,0	0	326,0	**421,0**	421,0	0	326,0	0,0	**421,0**	**0,00%**
2	NSC, ST	30	10	495,0	0	767,0	**495,0**	491,3	0	771,9	1,6	**491,3**	**0,76%**
2	NSC, ST	30	15	590,0	0	1231,6	**590,0**	587,8	0	1218,5	4,1	**587,8**	**0,38%**
2	NSC, ST	30	20	671,4	0	1677,3	**671,4**	667,0	0	1660,5	4,9	**667,0**	**0,65%**
2	NSC, ST	30	25	759,6	0	2122,8	**759,6**	754,3	0	2096,6	7,8	**754,3**	**0,71%**
2	NSC, ST	30	30	844,5	0	2557,3	**844,5**	835,9	0	2553,8	12,4	**835,9**	**1,02%**
2	NSC, ST	30	35	925,1	0	2984,8	**925,1**	913,8	0	2943,6	12,1	**913,8**	**1,23%**
2	NSC, ST	30	40	1012,8	0	3415,9	**1012,8**	998,8	0	3382,6	19,1	**998,8**	**1,38%**
2	NSC, ST	30	45	1098,6	0	3838,9	**1098,6**	1085,5	0	3813,9	17,0	**1085,5**	**1,19%**
2	NSC, ST	30	50	1175,1	0	4296,5	**1175,1**	1157,6	0	4250,4	24,4	**1157,6**	**1,49%**
3	SC, NST	30	5	410,0	541,0	0	**572,3**	413,0	516,0	0	2,0	**567,8**	**0,79%**
3	SC, NST	30	10	472,0	1189,0	0	**828,7**	474,0	1147,0	0	5,0	**818,1**	**1,28%**
3	SC, NST	30	15	568,1	1808,4	0	**1110,6**	567,9	1731,0	0	7,5	**1087,2**	**2,11%**
3	SC, NST	30	20	629,4	2469,3	0	**1370,2**	629,6	2389,6	0	7,4	**1346,5**	**1,72%**
3	SC, NST	30	25	714,1	3134,1	0	**1654,4**	715,4	3010,1	0	12,1	**1618,4**	**2,17%**
3	SC, NST	30	30	789,3	3774,5	0	**1921,6**	789,9	3602,1	0	15,1	**1870,5**	**2,66%**
3	SC, NST	30	35	854,3	4396,6	0	**2173,2**	855,1	4212,1	0	17,5	**2118,8**	**2,51%**
3	SC, NST	30	40	935,4	4999,0	0	**2435,1**	931,1	4796,3	0	18,6	**2370,0**	**2,67%**
3	SC, NST	30	45	1011,8	5702,6	0	**2722,5**	1008,9	5427,0	0	23,6	**2637,0**	**3,14%**
3	SC, NST	30	50	1069,8	6308,4	0	**2962,3**	1066,1	6041,3	0	22,6	**2878,5**	**2,83%**
4	SC, ST	30	5	421,0	541,0	326,0	**583,3**	421,0	541,0	326	0,0	**583,3**	**0,00%**
4	SC, ST	30	10	499,0	1191,0	779,0	**856,3**	494,0	1189,0	773	2,0	**850,7**	**0,65%**
4	SC, ST	30	15	610,8	1811,1	1196,4	**1154,1**	608,1	1744,9	1195,25	6,3	**1131,6**	**1,95%**
4	SC, ST	30	20	689,9	2480,3	1679,9	**1434,0**	692,3	2387,0	1669	8,5	**1408,4**	**1,78%**
4	SC, ST	30	25	798,1	3121,4	2171,3	**1734,5**	796,1	2999,0	2154,625	10,3	**1695,8**	**2,23%**
4	SC, ST	30	30	885,3	3788,4	2560,0	**2021,8**	883,4	3631,1	2566,375	14,8	**1972,7**	**2,43%**
4	SC, ST	30	35	974,5	4382,6	2980,1	**2289,3**	974,4	4153,4	2995,75	17,4	**2220,4**	**3,01%**
4	SC, ST	30	40	1058,6	5015,0	3469,6	**2563,1**	1061,1	4770,4	3468,125	18,8	**2492,2**	**2,76%**
4	SC, ST	30	45	1151,1	5687,0	3887,4	**2857,2**	1154,1	5430,1	3900	22,6	**2783,2**	**2,59%**
4	SC, ST	30	50	1237,6	6312,9	4329,4	**3131,5**	1235,9	6024,8	4347,125	25,6	**3043,3**	**2,82%**

Table A.22: **Intermediate case**. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

				Permutation				Non-permutation					
Case	Setup	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST		Obj-Func	Improv
1	NSC, NST	30	5	410,0	0	0	**410,0**	405,0	0	0	1,0	**405,0**	**1,22%**
1	NSC, NST	30	10	468,0	0	0	**468,0**	466,4	0	0	3,3	**466,4**	**0,35%**
1	NSC, NST	30	15	541,0	0	0	**541,0**	538,5	0	0	3,3	**538,5**	**0,46%**
1	NSC, NST	30	20	605,6	0	0	**605,6**	600,1	0	0	6,6	**600,1**	**0,91%**
1	NSC, NST	30	25	684,1	0	0	**684,1**	678,6	0	0	5,8	**678,6**	**0,80%**
1	NSC, NST	30	30	748,9	0	0	**748,9**	742,1	0	0	11,1	**742,1**	**0,90%**
1	NSC, NST	30	35	814,8	0	0	**814,8**	803,0	0	0	14,9	**803,0**	**1,44%**
1	NSC, NST	30	40	884,4	0	0	**884,4**	873,5	0	0	17,9	**873,5**	**1,23%**
1	NSC, NST	30	45	953,5	0	0	**953,5**	945,9	0	0	16,6	**945,9**	**0,80%**
1	NSC, NST	30	50	1011,6	0	0	**1011,6**	997,4	0	0	24,4	**997,4**	**1,41%**
2	NSC, ST	30	5	421,0	0	326,0	**421,0**	421,0	0	326,0	0,0	**421,0**	**0,00%**
2	NSC, ST	30	10	495,0	0	767,0	**495,0**	491,3	0	768,5	2,1	**491,3**	**0,76%**
2	NSC, ST	30	15	590,0	0	1231,6	**590,0**	587,5	0	1225,9	3,5	**587,5**	**0,42%**
2	NSC, ST	30	20	671,4	0	1677,3	**671,4**	667,9	0	1669,0	4,4	**667,9**	**0,52%**
2	NSC, ST	30	25	759,6	0	2122,8	**759,6**	752,1	0	2097,0	8,3	**752,1**	**0,99%**
2	NSC, ST	30	30	844,5	0	2557,3	**844,5**	836,1	0	2525,1	12,0	**836,1**	**0,99%**
2	NSC, ST	30	35	925,1	0	2984,8	**925,1**	913,0	0	2952,4	14,9	**913,0**	**1,31%**
2	NSC, ST	30	40	1012,8	0	3415,9	**1012,8**	998,8	0	3409,8	18,9	**998,8**	**1,38%**
2	NSC, ST	30	45	1098,6	0	3838,9	**1098,6**	1085,1	0	3815,0	17,5	**1085,1**	**1,23%**
2	NSC, ST	30	50	1175,1	0	4296,5	**1175,1**	1157,6	0	4242,3	25,5	**1157,6**	**1,49%**
3	SC, NST	30	5	410,0	541,0	0	**572,3**	413,0	516,0	0	2,0	**567,8**	**0,79%**
3	SC, NST	30	10	472,0	1189,0	0	**828,7**	473,1	1150,6	0	4,9	**818,3**	**1,25%**
3	SC, NST	30	15	568,1	1808,4	0	**1110,6**	567,1	1728,6	0	7,5	**1085,7**	**2,24%**
3	SC, NST	30	20	629,4	2469,3	0	**1370,2**	628,3	2385,6	0	9,3	**1343,9**	**1,91%**
3	SC, NST	30	25	714,1	3134,1	0	**1654,4**	714,9	3004,6	0	12,9	**1616,3**	**2,30%**
3	SC, NST	30	30	789,3	3774,5	0	**1921,6**	789,5	3603,4	0	14,9	**1870,5**	**2,66%**
3	SC, NST	30	35	854,3	4396,6	0	**2173,2**	855,1	4187,4	0	18,1	**2111,3**	**2,85%**
3	SC, NST	30	40	935,4	4999,0	0	**2435,1**	932,8	4776,1	0	18,8	**2365,6**	**2,85%**
3	SC, NST	30	45	1011,8	5702,6	0	**2722,5**	1007,1	5434,4	0	24,6	**2637,4**	**3,12%**
3	SC, NST	30	50	1069,8	6308,4	0	**2962,3**	1071,4	6009,4	0	25,3	**2874,2**	**2,97%**
4	SC, ST	30	5	421,0	541,0	326,0	**583,3**	421,0	541,0	326	0,0	**583,3**	**0,00%**
4	SC, ST	30	10	499,0	1191,0	779,0	**856,3**	495,0	1186,9	780,875	1,9	**851,1**	**0,61%**
4	SC, ST	30	15	610,8	1811,1	1196,4	**1154,1**	610,4	1733,9	1200,625	7,1	**1130,5**	**2,04%**
4	SC, ST	30	20	689,9	2480,3	1679,9	**1434,0**	689,6	2402,0	1674,75	7,6	**1410,2**	**1,65%**
4	SC, ST	30	25	798,1	3121,4	2171,3	**1734,5**	796,5	3002,0	2160,375	11,3	**1697,1**	**2,16%**
4	SC, ST	30	30	885,3	3788,4	2560,0	**2021,8**	882,8	3643,4	2560,5	13,3	**1975,8**	**2,28%**
4	SC, ST	30	35	974,5	4382,6	2980,1	**2289,3**	974,9	4173,0	2967,75	16,6	**2226,8**	**2,73%**
4	SC, ST	30	40	1058,6	5015,0	3469,6	**2563,1**	1060,9	4784,8	3482,875	16,4	**2496,3**	**2,61%**
4	SC, ST	30	45	1151,1	5687,0	3887,4	**2857,2**	1153,4	5437,3	3870,875	23,0	**2784,6**	**2,54%**
4	SC, ST	30	50	1237,6	6312,9	4329,4	**3131,5**	1236,5	6047,4	4319,5	24,8	**3050,7**	**2,58%**

Table A.23: **Centralized case**. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.2 Intermediate versus centralized buffer location

			Permutation flowshop				Non-permutation flowshop					
Case	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
I22	5	10	164,0	148	110	**208,4**	164,0	140	102	1,6	**206,0**	**1,15%**
I22	5	20	293,5	288,625	207,625	**380,1**	292,0	261,25	207,25	7,4	**370,4**	**2,56%**
I22	5	30	412,4	408,375	295,25	**534,9**	407,9	366,25	284,25	13,5	**517,8**	**3,20%**
I22	5	40	538,4	567,5	409,375	**708,6**	534,3	492	401,25	19,5	**681,9**	**3,78%**
I22	5	50	659,0	688,875	515,375	**865,7**	651,5	606,125	501,875	20,8	**833,3**	**3,74%**
I22	5	60	746,1	815,875	582,375	**990,9**	741,8	716,125	568	26,6	**956,6**	**3,46%**
I22	5	70	888,6	975,875	707,25	**1181,4**	880,6	860,875	684,375	31,4	**1138,9**	**3,60%**
I22	5	80	1003,0	1072,25	798,5	**1324,7**	998,6	941,75	781,625	29,9	**1281,2**	**3,28%**
I22	5	90	1140,6	1232,625	898,5	**1522,0**	1136,0	1103,625	892,375	32,9	**1467,1**	**3,61%**
I22	5	100	1267,6	1397,25	975,625	**1686,8**	1262,1	1233	962,25	35,1	**1632,0**	**3,25%**
C3	5	10	164,0	148	110,0	**208,4**	164,0	140	102,0	1,1	**206,0**	**1,15%**
C3	5	20	293,5	288,625	207,6	**380,1**	291,0	263,625	204,1	6,6	**370,1**	**2,63%**
C3	5	30	412,4	408,375	295,3	**534,9**	407,3	366,25	285,9	13,9	**517,1**	**3,32%**
C3	5	40	538,4	567,5	409,4	**708,6**	534,0	497,875	397,5	15,8	**683,4**	**3,56%**
C3	5	50	659,0	688,875	515,4	**865,7**	653,3	602,375	498,6	20,3	**834,0**	**3,66%**
C3	5	60	746,1	815,875	582,4	**990,9**	743,4	720,375	574,9	21,8	**959,5**	**3,17%**
C3	5	70	888,6	975,875	707,3	**1181,4**	884,5	868	695,4	22,9	**1144,9**	**3,09%**
C3	5	80	1003,0	1072,25	798,5	**1324,7**	998,1	961,5	785,5	23,8	**1286,6**	**2,87%**
C3	5	90	1143,1	1263	891,9	**1522,0**	1134,3	1139,25	866,1	28,0	**1476,0**	**3,02%**
C3	5	100	1267,6	1397,25	975,6	**1686,8**	1263,1	1259,875	976,1	28,6	**1641,1**	**2,71%**
C4	5	10	164,0	148,0	110	**208,4**	164,0	140,0	102	1,6	**206,0**	**1,15%**
C4	5	20	293,5	288,6	207,625	**380,1**	292,6	258,9	204,25	7,8	**370,3**	**2,58%**
C4	5	30	412,4	408,4	295,25	**534,9**	409,3	360,4	288,625	16,5	**517,4**	**3,28%**
C4	5	40	538,4	567,5	409,375	**708,6**	535,9	482,3	404,875	20,9	**680,6**	**3,96%**
C4	5	50	659,0	688,9	515,375	**865,7**	653,5	586,1	498,5	23,9	**829,3**	**4,20%**
C4	5	60	746,1	815,9	582,375	**990,9**	742,0	701,9	563,75	27,9	**952,6**	**3,87%**
C4	5	70	888,6	975,9	707,25	**1181,4**	881,0	830,3	685,125	30,8	**1130,1**	**4,34%**
C4	5	80	1003,0	1072,3	798,5	**1324,7**	994,8	928,1	779,375	33,5	**1273,2**	**3,89%**
C4	5	90	1143,1	1263,0	891,875	**1522,0**	1136,0	1081,6	870,125	38,3	**1460,5**	**4,04%**
C4	5	100	1267,6	1397,3	975,625	**1686,8**	1260,3	1223,0	959	37,6	**1627,2**	**3,54%**

Table A.24: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.3 Number of buffer places

			Permutation flowshop				Non-permutation flowshop					
Case	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
C2	5	10	164,0	148	110	**208,4**	164,0	140	102	1,6	**206,0**	**1,15%**
C2	5	20	293,5	288,625	207,625	**380,1**	292,6	258,875	204,25	7,8	**370,3**	**2,58%**
C2	5	30	412,4	408,375	295,25	**534,9**	409,3	360,375	288,625	16,5	**517,4**	**3,28%**
C2	5	40	538,4	567,5	409,375	**708,6**	535,9	482,25	404,875	20,9	**680,6**	**3,96%**
C2	5	50	659,0	688,875	515,375	**865,7**	653,5	586,125	498,5	23,9	**829,3**	**4,20%**
C2	5	60	746,1	815,875	582,375	**990,9**	742,0	701,875	563,75	27,9	**952,6**	**3,87%**
C2	5	70	888,6	975,875	707,25	**1181,4**	881,0	830,25	685,125	30,8	**1130,1**	**4,34%**
C2	5	80	1003,0	1072,25	798,5	**1324,7**	994,8	928,125	779,375	33,5	**1273,2**	**3,89%**
C2	5	90	1143,1	1263	891,875	**1522,0**	1136,0	1081,625	870,125	38,3	**1460,5**	**4,04%**
C2	5	100	1267,6	1397,25	975,625	**1686,8**	1260,3	1223	959	37,6	**1627,2**	**3,54%**
C3	5	10	164,0	148	110,0	**208,4**	164,0	140	102,0	1,1	**206,0**	**1,15%**
C3	5	20	293,5	288,625	207,6	**380,1**	291,0	263,625	204,1	6,6	**370,1**	**2,63%**
C3	5	30	412,4	408,375	295,3	**534,9**	407,3	366,25	285,9	13,9	**517,1**	**3,32%**
C3	5	40	538,4	567,5	409,4	**708,6**	534,0	497,875	397,5	15,8	**683,4**	**3,56%**
C3	5	50	659,0	688,875	515,4	**865,7**	653,3	602,375	498,6	20,3	**834,0**	**3,66%**
C3	5	60	746,1	815,875	582,4	**990,9**	743,4	720,375	574,9	21,8	**959,5**	**3,17%**
C3	5	70	888,6	975,875	707,3	**1181,4**	884,5	868	695,4	22,9	**1144,9**	**3,09%**
C3	5	80	1003,0	1072,25	798,5	**1324,7**	998,1	961,5	785,5	23,8	**1286,6**	**2,87%**
C3	5	90	1143,1	1263	891,9	**1522,0**	1134,3	1139,25	866,1	28,0	**1476,0**	**3,02%**
C3	5	100	1267,6	1397,25	975,6	**1686,8**	1263,1	1259,875	976,1	28,6	**1641,1**	**2,71%**
C4	5	10	164,0	148,0	110	**208,4**	164,0	140,0	102	1,3	**206,0**	**1,15%**
C4	5	20	293,5	288,6	207,625	**380,1**	292,5	258,5	206,25	6,1	**370,1**	**2,64%**
C4	5	30	412,4	408,4	295,25	**534,9**	410,6	369,6	295	10,0	**521,5**	**2,50%**
C4	5	40	538,4	567,5	409,375	**708,6**	534,4	518,3	399,625	12,0	**689,9**	**2,65%**
C4	5	50	659,0	688,9	515,375	**865,7**	654,9	618,1	502	13,6	**840,3**	**2,93%**
C4	5	60	746,1	815,9	582,375	**990,9**	743,4	748,5	580,25	15,4	**967,9**	**2,32%**
C4	5	70	888,6	975,9	707,25	**1181,4**	883,9	891,6	688,125	16,5	**1151,4**	**2,54%**
C4	5	80	1003,0	1072,3	798,5	**1324,7**	999,1	984,1	786,375	16,6	**1294,4**	**2,29%**
C4	5	90	1143,1	1263,0	891,875	**1522,0**	1139,6	1158,9	881,75	18,5	**1487,3**	**2,28%**
C4	5	100	1267,6	1397,3	975,625	**1686,8**	1266,5	1315,1	972,125	17,9	**1661,0**	**1,53%**

Table A.25: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.4 Difference in physical size of jobs

				Permutation flowshop				Non-permutation flowshop					
Case	Buffer	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
Int	012	5	10	164,0	148	110	**208,4**	164,0	140	102	1,0	**206,0**	**1,15%**
Int	012	5	20	293,5	288,625	207,625	**380,1**	292,0	269	209	5,6	**372,7**	**1,94%**
Int	012	5	30	412,4	408,375	295,25	**534,9**	410,9	385,375	296	7,4	**526,5**	**1,57%**
Int	012	5	40	538,4	567,5	409,375	**708,6**	535,3	535,625	404,875	9,3	**695,9**	**1,79%**
Int	012	5	50	659,0	688,875	515,375	**865,7**	657,5	646	508,625	10,3	**851,3**	**1,66%**
Int	012	5	60	746,1	815,875	582,375	**990,9**	743,0	767	566,5	13,9	**973,1**	**1,79%**
Int	012	5	70	888,6	975,875	707,25	**1181,4**	882,5	925	694,75	14,6	**1160,0**	**1,81%**
Int	012	5	80	1003,0	1072,25	798,5	**1324,7**	1000,5	1013,25	789,875	12,3	**1304,5**	**1,53%**
Int	012	5	90	1143,1	1263	891,875	**1522,0**	1140,3	1190,25	884,25	17,4	**1497,3**	**1,62%**
Int	012	5	100	1267,6	1397,25	975,625	**1686,8**	1265,6	1335,375	973	15,5	**1666,2**	**1,22%**
Int	102	5	10	164,0	148	110,0	**208,4**	164,0	140	102,0	1,3	**206,0**	**1,15%**
Int	102	5	20	293,5	288,625	207,6	**380,1**	292,8	263,625	205,6	5,5	**371,8**	**2,17%**
Int	102	5	30	412,4	408,375	295,3	**534,9**	411,3	385,25	296,9	7,0	**526,8**	**1,51%**
Int	102	5	40	538,4	567,5	409,4	**708,6**	534,0	527,125	396,4	11,4	**692,1**	**2,33%**
Int	102	5	50	659,0	688,875	515,4	**865,7**	654,8	643,625	500,0	12,1	**847,8**	**2,06%**
Int	102	5	60	746,1	815,875	582,4	**990,9**	743,8	756,375	568,6	15,4	**970,7**	**2,04%**
Int	102	5	70	888,6	975,875	707,3	**1181,4**	885,0	910,625	701,0	15,4	**1158,2**	**1,97%**
Int	102	5	80	1003,0	1072,25	798,5	**1324,7**	999,6	985,125	792,9	18,3	**1295,2**	**2,23%**
Int	102	5	90	1143,1	1263	891,9	**1522,0**	1141,4	1171,75	889,5	17,4	**1492,9**	**1,91%**
Int	102	5	100	1267,6	1397,25	975,6	**1686,8**	1265,4	1313,5	967,1	18,0	**1659,4**	**1,62%**
Int	111	5	10	164,0	148,0	110	**208,4**	164,0	140,0	102	1,1	**206,0**	**1,15%**
Int	111	5	20	293,5	288,6	207,625	**380,1**	293,0	262,5	207,625	5,8	**371,8**	**2,19%**
Int	111	5	30	412,4	408,4	295,25	**534,9**	409,6	374,1	294,25	10,9	**521,9**	**2,43%**
Int	111	5	40	538,4	567,5	409,375	**708,6**	533,5	515,3	398,25	11,1	**688,1**	**2,90%**
Int	111	5	50	659,0	688,9	515,375	**865,7**	655,4	623,8	507,625	14,5	**842,5**	**2,68%**
Int	111	5	60	746,1	815,9	582,375	**990,9**	742,5	761,6	575,625	15,6	**971,0**	**2,01%**
Int	111	5	70	888,6	975,9	707,25	**1181,4**	884,3	892,5	695,375	18,3	**1152,0**	**2,49%**
Int	111	5	80	1003,0	1072,3	798,5	**1324,7**	998,9	990,3	782,625	18,4	**1296,0**	**2,17%**
Int	111	5	90	1143,1	1263,0	891,875	**1522,0**	1141,5	1175,1	889,25	20,3	**1494,0**	**1,84%**
Int	111	5	100	1267,6	1397,3	975,625	**1686,8**	1264,1	1304,4	964,25	21,0	**1655,4**	**1,86%**
Int	300	5	10	164,0	148,0	110,0	**208,4**	164,0	140,0	102	1,9	**206,0**	**1,15%**
Int	300	5	20	293,5	288,6	207,6	**380,1**	291,6	264,4	206,125	6,9	**370,9**	**2,41%**
Int	300	5	30	412,4	408,4	295,3	**534,9**	408,3	370,1	283,375	11,9	**519,3**	**2,92%**
Int	300	5	40	538,4	567,5	409,4	**708,6**	534,9	506,0	401,625	14,0	**686,7**	**3,10%**
Int	300	5	50	659,0	688,9	515,4	**865,7**	657,3	608,0	511,5	16,5	**839,7**	**3,01%**
Int	300	5	60	746,1	815,9	582,4	**990,9**	742,8	744,3	572	19,1	**966,0**	**2,51%**
Int	300	5	70	888,6	975,9	707,3	**1181,4**	884,4	888,4	697,125	20,9	**1150,9**	**2,58%**
Int	300	5	80	1003,0	1072,3	798,5	**1324,7**	998,1	962,4	790,125	22,5	**1286,8**	**2,85%**
Int	300	5	90	1143,1	1263,0	891,9	**1522,0**	1138,9	1165,3	875,5	21,9	**1488,5**	**2,20%**
Int	300	5	100	1267,6	1397,3	975,6	**1686,8**	1266,3	1279,0	960,125	25,1	**1650,0**	**2,19%**
Cent	012	5	10	164,0	148	110	**208,4**	164,0	140	102	1,0	**206,0**	**1,15%**
Cent	012	5	20	293,5	288,625	207,625	**380,1**	291,8	269,5	208,5	4,6	**372,6**	**1,97%**
Cent	012	5	30	412,4	408,375	295,25	**534,9**	411,6	373,5	293,75	9,3	**523,7**	**2,09%**
Cent	012	5	40	538,4	567,5	409,375	**708,6**	535,1	521,125	400,625	10,3	**691,5**	**2,42%**
Cent	012	5	50	659,0	688,875	515,375	**865,7**	655,9	637,875	507	14,1	**847,2**	**2,13%**
Cent	012	5	60	746,1	815,875	582,375	**990,9**	743,9	741,375	572,75	17,8	**966,3**	**2,48%**
Cent	012	5	70	888,6	975,875	707,25	**1181,4**	883,8	907,375	690,75	16,4	**1156,0**	**2,15%**
Cent	012	5	80	1003,0	1072,25	798,5	**1324,7**	998,3	979	786,5	20,5	**1292,0**	**2,47%**
Cent	012	5	90	1143,1	1263	891,875	**1522,0**	1140,0	1171,75	877,125	20,1	**1491,5**	**2,00%**
Cent	012	5	100	1267,6	1397,25	975,625	**1686,8**	1262,6	1306,75	968,875	20,0	**1654,7**	**1,90%**
Cent	102	5	10	164,0	148	110,0	**208,4**	164,0	140	102,0	1,5	**206,0**	**1,15%**
Cent	102	5	20	293,5	288,625	207,6	**380,1**	292,9	257,875	202,9	6,1	**370,2**	**2,59%**
Cent	102	5	30	412,4	408,375	295,3	**534,9**	409,0	377,125	289,3	10,3	**522,1**	**2,38%**
Cent	102	5	40	538,4	567,5	409,4	**708,6**	535,6	518	402,9	12,8	**691,0**	**2,48%**
Cent	102	5	50	659,0	688,875	515,4	**865,7**	654,5	623,5	504,0	14,6	**841,6**	**2,79%**
Cent	102	5	60	746,1	815,875	582,4	**990,9**	743,4	733,75	577,4	19,1	**963,5**	**2,76%**
Cent	102	5	70	888,6	975,875	707,3	**1181,4**	883,4	895,5	696,8	17,6	**1152,0**	**2,48%**
Cent	102	5	80	1003,0	1072,25	798,5	**1324,7**	998,9	981,125	792,9	20,6	**1293,2**	**2,37%**
Cent	102	5	90	1143,1	1263	891,9	**1522,0**	1139,0	1168,5	880,6	20,1	**1489,6**	**2,13%**
Cent	102	5	100	1267,6	1397,25	975,6	**1686,8**	1260,9	1292,75	961,8	22,4	**1648,7**	**2,26%**
Cent	111	5	10	164,0	148,0	110	**208,4**	164,0	140,0	102	1,5	**206,0**	**1,15%**
Cent	111	5	20	293,5	288,6	207,625	**380,1**	291,9	260,5	203,625	7,4	**370,0**	**2,65%**
Cent	111	5	30	412,4	408,4	295,25	**534,9**	409,0	368,1	289,75	11,6	**519,4**	**2,89%**
Cent	111	5	40	538,4	567,5	409,375	**708,6**	533,6	499,1	396,75	15,4	**683,4**	**3,56%**
Cent	111	5	50	659,0	688,9	515,375	**865,7**	654,9	613,0	501,875	17,5	**838,8**	**3,11%**
Cent	111	5	60	746,1	815,9	582,375	**990,9**	745,0	726,9	580,25	19,9	**963,1**	**2,81%**
Cent	111	5	70	888,6	975,9	707,25	**1181,4**	881,8	881,9	693	23,3	**1146,3**	**2,97%**
Cent	111	5	80	1003,0	1072,3	798,5	**1324,7**	994,3	969,4	780,25	23,0	**1285,1**	**2,99%**
Cent	111	5	90	1143,1	1263,0	891,875	**1522,0**	1138,9	1154,4	878,125	25,0	**1485,2**	**2,42%**
Cent	111	5	100	1267,6	1397,3	975,625	**1686,8**	1262,1	1265,6	965,125	26,4	**1641,8**	**2,67%**
Cent	300	5	10	164,0	148,0	110,0	**208,4**	164,0	140,0	102	1,1	**206,0**	**1,15%**
Cent	300	5	20	293,5	288,6	207,6	**380,1**	291,0	263,6	204,125	6,6	**370,1**	**2,63%**
Cent	300	5	30	412,4	408,4	295,3	**534,9**	407,3	366,3	285,875	13,9	**517,1**	**3,32%**
Cent	300	5	40	538,4	567,5	409,4	**708,6**	534,0	497,9	397,5	15,8	**683,4**	**3,56%**
Cent	300	5	50	659,0	688,9	515,4	**865,7**	653,3	602,4	498,625	20,3	**834,0**	**3,66%**
Cent	300	5	60	746,1	815,9	582,4	**990,9**	743,4	720,4	574,875	21,8	**959,5**	**3,17%**
Cent	300	5	70	888,6	975,9	707,3	**1181,4**	884,5	868,0	695,375	22,9	**1144,9**	**3,09%**
Cent	300	5	80	1003,0	1072,3	798,5	**1324,7**	998,1	961,5	785,5	23,8	**1286,6**	**2,87%**
Cent	300	5	90	1143,1	1263,0	891,9	**1522,0**	1134,3	1139,3	866,125	28,0	**1476,0**	**3,02%**
Cent	300	5	100	1267,6	1397,3	975,6	**1686,8**	1263,1	1259,9	976,125	28,6	**1641,1**	**2,71%**

Table A.26: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.5 Distribution of data

A.4.5.1 Range

				Permutation flowshop				Non-permutation flowshop					
Case	Range	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
I22	16-24	20	10	634,8	814	517,5	**879,0**	634,8	814	517,5	0,0	**879,0**	**0,00%**
I22	16-24	20	30	1119,3	2558,375	1685,75	**1886,8**	1124,5	2530,125	1671,375	2,9	**1883,5**	**0,17%**
I22	16-24	20	50	1593,3	4360,25	2816,75	**2901,3**	1596,4	4329,375	2818,5	3,3	**2895,2**	**0,21%**
I22	16-24	20	70	2077,1	6174,625	4086,125	**3929,5**	2087,4	6080,5	4073,125	11,6	**3911,5**	**0,46%**
I22	16-24	20	90	2548,5	7987,875	5289,25	**4944,9**	2559,4	7850	5287	15,0	**4914,4**	**0,62%**
I22	12-28	20	10	651,0	843	547,0	**903,9**	651,9	836,375	532,6	1,1	**902,8**	**0,12%**
I22	12-28	20	30	1140,5	2594,625	1674,3	**1918,9**	1144,6	2556	1669,5	5,0	**1911,4**	**0,39%**
I22	12-28	20	50	1629,6	4451,375	2849,9	**2965,0**	1633,4	4365,375	2854,3	10,8	**2943,0**	**0,74%**
I22	12-28	20	70	2165,8	6322,375	4075,8	**4062,5**	2172,1	6109,5	4043,0	24,9	**4005,0**	**1,11%**
I22	12-28	20	90	2598,4	8079,375	5240,5	**5022,2**	2611,0	7837,625	5237,5	30,1	**4962,3**	**1,19%**
I22	8-32	20	10	664,0	872,0	537	**925,6**	664,0	872,0	537	0,0	**925,6**	**0,00%**
I22	8-32	20	30	1164,3	2645,9	1691,625	**1958,0**	1163,3	2604,1	1681,25	5,3	**1944,5**	**0,69%**
I22	8-32	20	50	1670,5	4481,9	2863,375	**3015,1**	1677,0	4311,4	2843,25	17,8	**2970,4**	**1,48%**
I22	8-32	20	70	2165,8	6322,4	4075,75	**4062,5**	2172,1	6109,5	4043	24,9	**4005,0**	**1,42%**
I22	8-32	20	90	2646,4	8151,4	5269,5	**5091,8**	2652,5	7861,5	5222,375	33,4	**5011,0**	**1,59%**
C3	16-24	20	10	634,8	814,0	517,5	**879,0**	634,8	814,0	517,5	0,0	**879,0**	**0,00%**
C3	16-24	20	30	1119,3	2558,4	1685,8	**1886,8**	1125,4	2524,8	1672,125	3,8	**1882,8**	**0,21%**
C3	16-24	20	50	1593,3	4360,3	2816,8	**2901,3**	1596,5	4329,1	2817,625	3,5	**2895,2**	**0,21%**
C3	16-24	20	70	2077,1	6174,6	4086,1	**3929,5**	2081,9	6114,1	4088,125	6,6	**3916,1**	**0,34%**
C3	16-24	20	90	2548,5	7987,9	5289,3	**4944,9**	2557,9	7849,1	5282,75	14,6	**4912,6**	**0,65%**
C3	12-28	20	10	651,0	843	547	**903,9**	652,0	835,625	530	1,1	**902,7**	**0,13%**
C3	12-28	20	30	1140,5	2594,625	1674,25	**1918,9**	1144,0	2564,75	1666,875	4,1	**1913,4**	**0,28%**
C3	12-28	20	50	1629,6	4451,375	2849,875	**2965,0**	1632,6	4376,75	2838,5	10,8	**2945,7**	**0,65%**
C3	12-28	20	70	2165,8	6322,375	4075,75	**4062,5**	2167,8	6126	4054,25	25,3	**4005,6**	**1,10%**
C3	12-28	20	90	2598,4	8079,375	5240,5	**5022,2**	2609,6	7828,875	5210,25	30,1	**4958,3**	**1,27%**
C3	8-32	20	10	664,0	872	537,0	**925,6**	664,0	872	537,0	0,0	**925,6**	**0,00%**
C3	8-32	20	30	1164,3	2645,875	1691,6	**1958,0**	1166,0	2590,75	1676,6	6,0	**1943,2**	**0,76%**
C3	8-32	20	50	1670,5	4481,875	2863,4	**3015,1**	1674,0	4351,25	2868,1	14,9	**2979,4**	**1,18%**
C3	8-32	20	70	2165,8	6322,375	4075,8	**4062,5**	2167,8	6126	4054,3	25,3	**4005,6**	**1,40%**
C3	8-32	20	90	2646,4	8151,375	5269,5	**5091,8**	2652,8	7846,5	5236,8	34,8	**5006,7**	**1,67%**

Table A.27: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.5.2 Spreading

				Permutation flowshop				Non-permutation flowshop					
Case	Range	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
I22	16-24	20	10	634,8	814	517,5	**879,0**	634,8	814	517,5	0,0	**879,0**	**0,00%**
I22	16-24	20	30	1119,3	2558,375	1685,75	**1886,8**	1124,5	2530,125	1671,375	2,9	**1883,5**	**0,17%**
I22	16-24	20	50	1593,3	4360,25	2816,75	**2901,3**	1596,4	4329,375	2818,5	3,3	**2895,2**	**0,21%**
I22	16-24	20	70	2077,1	6174,625	4086,125	**3929,5**	2087,4	6080,5	4073,125	11,6	**3911,5**	**0,46%**
I22	16-24	20	90	2548,5	7987,875	5289,25	**4944,9**	2559,4	7850	5287	15,0	**4914,4**	**0,62%**
I22	12-28	20	10	651,0	843	547,0	**903,9**	651,9	836,375	532,6	1,1	**902,8**	**0,12%**
I22	12-28	20	30	1140,5	2594,625	1674,3	**1918,9**	1144,6	2556	1669,5	5,0	**1911,4**	**0,39%**
I22	12-28	20	50	1629,6	4451,375	2849,9	**2965,0**	1633,4	4365,375	2854,3	10,8	**2943,0**	**0,74%**
I22	12-28	20	70	2165,8	6322,375	4075,8	**4062,5**	2172,1	6109,5	4043,0	24,9	**4005,0**	**1,11%**
I22	12-28	20	90	2598,4	8079,375	5240,5	**5022,2**	2611,0	7837,625	5237,5	30,1	**4962,3**	**1,19%**
I22	8-32	20	10	664,0	872,0	537	**925,6**	664,0	872,0	537	0,0	**925,6**	**0,00%**
I22	8-32	20	30	1164,3	2645,9	1691,625	**1958,0**	1163,3	2604,1	1681,25	5,3	**1944,5**	**0,69%**
I22	8-32	20	50	1670,5	4481,9	2863,375	**3015,1**	1677,0	4311,4	2843,25	17,8	**2970,4**	**1,48%**
I22	8-32	20	70	2165,8	6322,4	4075,75	**4062,5**	2172,1	6109,5	4043	24,9	**4005,0**	**1,42%**
I22	8-32	20	90	2646,4	8151,4	5269,5	**5091,8**	2652,5	7861,5	5222,375	33,4	**5011,0**	**1,59%**
C3	16-24	20	10	634,8	814,0	517,5	**879,0**	634,8	814,0	517,5	0,0	**879,0**	**0,00%**
C3	16-24	20	30	1119,3	2558,4	1685,8	**1886,8**	1125,4	2524,8	1672,125	3,8	**1882,8**	**0,21%**
C3	16-24	20	50	1593,3	4360,3	2816,8	**2901,3**	1596,5	4329,1	2817,625	3,5	**2895,2**	**0,21%**
C3	16-24	20	70	2077,1	6174,6	4086,1	**3929,5**	2081,9	6114,1	4088,125	6,6	**3916,1**	**0,34%**
C3	16-24	20	90	2548,5	7987,9	5289,3	**4944,9**	2557,9	7849,1	5282,75	14,6	**4912,6**	**0,65%**
C3	12-28	20	10	651,0	843	547	**903,9**	652,0	835,625	530	1,1	**902,7**	**0,13%**
C3	12-28	20	30	1140,5	2594,625	1674,25	**1918,9**	1144,0	2564,75	1666,875	4,1	**1913,4**	**0,28%**
C3	12-28	20	50	1629,6	4451,375	2849,875	**2965,0**	1632,6	4376,75	2838,5	10,8	**2945,7**	**0,65%**
C3	12-28	20	70	2165,8	6322,375	4075,75	**4062,5**	2167,8	6126	4054,25	25,3	**4005,6**	**1,10%**
C3	12-28	20	90	2598,4	8079,375	5240,5	**5022,2**	2609,6	7828,875	5210,25	30,1	**4958,3**	**1,27%**
C3	8-32	20	10	664,0	872	537,0	**925,6**	664,0	872	537,0	0,0	**925,6**	**0,00%**
C3	8-32	20	30	1164,3	2645,875	1691,6	**1958,0**	1166,0	2590,75	1676,6	6,0	**1943,2**	**0,76%**
C3	8-32	20	50	1670,5	4481,875	2863,4	**3015,1**	1674,0	4351,25	2868,1	14,9	**2979,4**	**1,18%**
C3	8-32	20	70	2165,8	6322,375	4075,8	**4062,5**	2167,8	6126	4054,3	25,3	**4005,6**	**1,40%**
C3	8-32	20	90	2646,4	8151,375	5269,5	**5091,8**	2652,8	7846,5	5236,8	34,8	**5006,7**	**1,67%**

Table A.28: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.5.3 Normal distribution

				Permutation flowshop				Non-permutation flowshop					
Case	Stand Dev	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
I22	10 %	5	10	165,9	184,625	122,375	**221,3**	165,9	184,625	122,375	0,0	**221,3**	**0,00%**
I22	10 %	5	20	299,8	382,875	248,375	**414,6**	300,4	380,5	249	0,3	**414,5**	**0,02%**
I22	10 %	5	30	420,3	599,375	375,75	**600,1**	419,9	591,25	372,25	2,3	**597,3**	**0,47%**
I22	10 %	5	40	548,5	797,125	518	**787,6**	550,5	776,5	510,75	4,6	**783,5**	**0,53%**
I22	10 %	5	50	671,1	1008,375	658,25	**973,6**	673,9	974,375	644,875	7,6	**966,2**	**0,76%**
I22	10 %	5	60	779,4	1211,75	788,625	**1142,9**	781,6	1158,5	774,875	11,6	**1129,2**	**1,20%**
I22	10 %	5	70	908,8	1427,5	931,875	**1337,0**	910,0	1368,25	905	14,5	**1320,5**	**1,24%**
I22	10 %	5	80	1012,5	1613,5	1057,125	**1496,6**	1017,0	1534,5	1038,125	17,4	**1477,4**	**1,28%**
I22	10 %	5	90	1146,0	1855,125	1196	**1702,5**	1150,8	1767,875	1182,625	18,0	**1681,1**	**1,26%**
I22	10 %	5	100	1257,4	2064,5	1337,75	**1876,7**	1260,8	1932,875	1307,5	27,5	**1840,6**	**1,92%**
I22	20 %	5	10	164,0	184	121,0	**219,2**	164,0	184	121,0	0,0	**219,2**	**0,00%**
I22	20 %	5	20	304,9	377,375	257,5	**418,1**	305,1	375,25	258,4	0,3	**417,7**	**0,09%**
I22	20 %	5	30	426,5	596,875	377,9	**605,6**	426,5	567,5	373,1	5,0	**596,8**	**1,45%**
I22	20 %	5	40	550,4	806	526,6	**792,2**	551,8	787	517,1	3,9	**787,9**	**0,55%**
I22	20 %	5	50	673,6	1022,75	648,5	**980,5**	675,1	984,75	634,8	10,5	**970,6**	**1,01%**
I22	20 %	5	60	774,3	1224,5	798,0	**1141,6**	775,5	1147,25	776,0	15,5	**1119,7**	**1,92%**
I22	20 %	5	70	907,9	1416,125	940,4	**1332,7**	907,9	1340,875	911,0	19,3	**1310,1**	**1,69%**
I22	20 %	5	80	1008,3	1618,625	1077,0	**1493,8**	1004,8	1548,625	1042,8	19,4	**1469,3**	**1,64%**
I22	20 %	5	90	1142,1	1841,375	1201,9	**1694,5**	1142,5	1751,375	1181,5	21,3	**1667,9**	**1,57%**
I22	20 %	5	100	1247,1	2059,375	1332,1	**1864,9**	1247,6	1954,375	1299,5	25,8	**1833,9**	**1,66%**
I22	30 %	5	10	164,5	182,9	126,875	**219,4**	164,9	181,0	123,5	0,4	**219,2**	**0,09%**
I22	30 %	5	20	301,0	386,5	248,375	**417,0**	301,6	380,1	245,5	1,5	**415,7**	**0,31%**
I22	30 %	5	30	426,8	601,6	381,625	**607,2**	425,8	570,4	368,375	7,3	**596,9**	**1,71%**
I22	30 %	5	40	549,1	817,8	513	**794,5**	550,4	780,5	504,5	8,1	**784,5**	**1,25%**
I22	30 %	5	50	675,9	1020,0	650	**981,9**	675,9	957,8	636,75	14,6	**963,2**	**1,90%**
I22	30 %	5	60	770,1	1225,4	796,5	**1137,7**	772,3	1165,1	780,625	16,4	**1121,8**	**1,40%**
I22	30 %	5	70	903,6	1437,0	932,875	**1334,7**	904,0	1363,0	920,125	17,9	**1312,9**	**1,64%**
I22	30 %	5	80	1004,9	1631,5	1072,625	**1494,3**	1003,5	1548,0	1048	22,6	**1467,9**	**1,77%**
I22	30 %	5	90	1139,6	1849,9	1213,625	**1694,6**	1140,5	1732,9	1193,5	28,0	**1660,4**	**2,02%**
I22	30 %	5	100	1244,5	2053,5	1331,25	**1860,6**	1241,6	1918,5	1298,25	31,6	**1817,2**	**2,33%**
I22	40 %	5	10	163,4	186,0	122,875	**219,2**	163,5	183,8	122	0,5	**218,6**	**0,25%**
I22	40 %	5	20	305,3	382,1	253,25	**419,9**	306,1	372,5	252,375	1,4	**417,9**	**0,48%**
I22	40 %	5	30	428,3	601,3	387,875	**608,6**	427,4	570,5	375,375	6,4	**598,5**	**1,66%**
I22	40 %	5	40	551,5	802,8	516,25	**792,3**	553,4	759,8	510,625	10,1	**781,3**	**1,39%**
I22	40 %	5	50	675,6	1022,6	662,75	**982,4**	675,8	954,3	650,25	16,8	**962,0**	**2,07%**
I22	40 %	5	60	777,1	1230,1	812,875	**1146,2**	771,9	1150,3	775,125	20,1	**1117,0**	**2,55%**
I22	40 %	5	70	896,6	1433,8	927,375	**1326,8**	894,3	1348,9	904,5	23,6	**1298,9**	**2,10%**
I22	40 %	5	80	1006,9	1646,4	1077,125	**1500,8**	1005,0	1532,5	1036,875	27,9	**1464,8**	**2,40%**
I22	40 %	5	90	1133,4	1862,0	1216,125	**1692,0**	1132,6	1736,0	1190,375	28,5	**1653,4**	**2,28%**
I22	40 %	5	100	1236,4	2064,0	1343,75	**1855,6**	1229,6	1937,5	1309,375	30,0	**1810,9**	**2,41%**
C3	10 %	5	10	165,9	184,625	122,4	**221,3**	165,9	184,625	122,4	0,0	**221,3**	**0,00%**
C3	10 %	5	20	299,8	382,875	248,4	**414,6**	300,4	380,5	249,0	0,3	**414,5**	**0,02%**
C3	10 %	5	30	420,3	599,375	375,8	**600,1**	421,0	580,125	368,9	4,0	**595,0**	**0,84%**
C3	10 %	5	40	548,5	797,125	518,0	**787,6**	550,3	778,5	513,1	3,9	**783,8**	**0,49%**
C3	10 %	5	50	671,1	1008,375	658,3	**973,6**	672,4	974,375	646,1	7,9	**964,7**	**0,92%**
C3	10 %	5	60	779,4	1211,75	788,6	**1142,9**	781,0	1169	783,5	9,9	**1131,7**	**0,98%**
C3	10 %	5	70	908,8	1427,5	931,9	**1337,0**	911,4	1366,125	910,1	13,8	**1321,2**	**1,18%**
C3	10 %	5	80	1012,5	1613,5	1057,1	**1496,6**	1015,9	1545,25	1039,8	15,9	**1479,5**	**1,14%**
C3	10 %	5	90	1146,0	1855,125	1196,0	**1702,5**	1149,9	1763,375	1179,5	19,1	**1678,9**	**1,39%**
C3	10 %	5	100	1257,4	2064,5	1337,8	**1876,7**	1261,5	1934	1300,5	25,8	**1841,7**	**1,87%**
C3	20 %	5	10	164,0	184,0	121	**219,2**	164,0	184,0	121	0,0	**219,2**	**0,00%**
C3	20 %	5	20	304,9	377,4	257,5	**418,1**	305,1	375,3	258,375	0,3	**417,7**	**0,09%**
C3	20 %	5	30	426,5	596,9	377,875	**605,6**	426,3	572,6	373	4,4	**598,0**	**1,24%**
C3	20 %	5	40	550,4	806,0	526,625	**792,2**	552,3	782,0	511,125	5,5	**786,9**	**0,67%**
C3	20 %	5	50	673,6	1022,8	648,5	**980,5**	675,4	985,6	638	9,1	**971,1**	**0,96%**
C3	20 %	5	60	774,3	1224,5	798	**1141,6**	775,1	1149,3	774,25	15,5	**1119,9**	**1,90%**
C3	20 %	5	70	907,9	1416,1	940,375	**1332,7**	907,1	1345,4	913,75	17,8	**1310,7**	**1,65%**
C3	20 %	5	80	1008,3	1618,6	1077	**1493,8**	1007,3	1538,4	1048,5	20,0	**1468,8**	**1,68%**
C3	20 %	5	90	1142,1	1841,4	1201,875	**1694,5**	1142,3	1757,9	1176,875	21,1	**1669,6**	**1,47%**
C3	20 %	5	100	1247,1	2059,4	1332,125	**1864,9**	1247,8	1933,9	1300,25	29,3	**1827,9**	**1,99%**
C3	30 %	5	10	164,5	182,9	126,9	**219,4**	164,9	181,0	123,5	0,4	**219,2**	**0,09%**
C3	30 %	5	20	301,0	386,5	248,4	**417,0**	301,3	383,0	246,375	1,1	**416,2**	**0,19%**
C3	30 %	5	30	426,8	601,6	381,6	**607,2**	427,5	566,4	378,75	8,3	**597,4**	**1,62%**
C3	30 %	5	40	549,1	817,8	513,0	**794,5**	552,4	776,9	504,875	8,6	**785,4**	**1,13%**
C3	30 %	5	50	675,9	1020,0	650,0	**981,9**	675,0	962,0	632,875	14,6	**963,6**	**1,86%**
C3	30 %	5	60	770,1	1225,4	796,5	**1137,7**	772,5	1153,3	778,875	17,6	**1118,5**	**1,69%**
C3	30 %	5	70	903,6	1437,0	932,9	**1334,7**	901,6	1368,3	915,625	18,6	**1312,1**	**1,70%**
C3	30 %	5	80	1004,9	1631,5	1072,6	**1494,3**	1002,1	1546,8	1042,5	21,0	**1466,2**	**1,89%**
C3	30 %	5	90	1139,6	1849,9	1213,6	**1694,6**	1137,9	1754,4	1178,75	26,1	**1664,2**	**1,79%**
C3	30 %	5	100	1244,5	2053,5	1331,3	**1860,6**	1244,3	1927,0	1306,625	30,6	**1822,4**	**2,05%**
C3	40 %	5	10	163,4	186,0	122,9	**219,2**	163,5	183,8	122	0,5	**218,6**	**0,25%**
C3	40 %	5	20	305,3	382,1	253,3	**419,9**	306,4	372,3	255	1,4	**418,1**	**0,44%**
C3	40 %	5	30	428,3	601,3	387,9	**608,6**	427,5	569,1	376	7,3	**598,2**	**1,71%**
C3	40 %	5	40	551,5	802,8	516,3	**792,3**	551,1	769,4	507,375	9,4	**781,9**	**1,31%**
C3	40 %	5	50	675,6	1022,6	662,8	**982,4**	675,0	958,5	646,5	15,6	**962,6**	**2,02%**
C3	40 %	5	60	777,1	1230,1	812,9	**1146,2**	773,6	1146,4	782,75	19,8	**1117,5**	**2,50%**
C3	40 %	5	70	896,6	1433,8	927,4	**1326,8**	893,1	1347,0	906,625	23,4	**1297,2**	**2,23%**
C3	40 %	5	80	1006,9	1646,4	1077,1	**1500,8**	1005,5	1532,3	1046,625	25,4	**1465,2**	**2,37%**
C3	40 %	5	90	1133,4	1862,0	1216,1	**1692,0**	1130,6	1755,6	1185,75	29,4	**1657,3**	**2,05%**
C3	40 %	5	100	1236,4	2064,0	1343,8	**1855,6**	1232,1	1929,0	1314,5	34,3	**1810,8**	**2,41%**

Table A.29: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.4.6 Semi dynamic demand

			Permutation flowshop				Non-permutation flowshop					
Case	Stations	Jobs	Makespan	SC	ST	Obj-Func	Makespan	SC	ST	Job-Ch	Obj-Func	Improv
I20	5	10	204,8	195,25	122,125	**263,3**	197,0	180,625	110,5	3,3	**251,2**	**4,58%**
I20	5	20	404,9	383,125	235,875	**519,8**	392,3	320	233,375	6,8	**488,3**	**6,00%**
I20	5	30	535,4	554,375	335,75	**701,7**	525,5	472	321	9,6	**667,1**	**4,93%**
I20	5	40	695,0	774,625	469	**927,4**	658,5	685,125	458	10,9	**864,0**	**6,78%**
I20	5	50	826,6	930,5	578	**1105,8**	795,5	836,5	563,125	13,0	**1046,5**	**5,37%**
I20	5	60	944,6	1085	672,375	**1270,1**	907,4	1008,375	662,5	10,3	**1209,9**	**4,74%**
I20	5	70	1111,3	1328,875	779,625	**1509,9**	1065,8	1243,5	768,375	10,1	**1438,8**	**4,72%**
I20	5	80	1257,4	1438,625	915,125	**1689,0**	1203,9	1409,625	899,125	7,1	**1626,8**	**3,70%**
I20	5	90	1370,4	1677	1021	**1873,5**	1340,4	1595	1020	9,1	**1818,9**	**2,92%**
I20	5	100	1527,4	1820,375	1155,125	**2073,5**	1483,4	1799,375	1141,875	4,4	**2023,2**	**2,43%**
I02	5	10	204,8	195,25	122,1	**263,3**	200,8	177,5	115,1	2,9	**254,0**	**3,53%**
I02	5	20	404,9	383,125	235,9	**519,8**	395,5	332,125	235,0	7,1	**495,1**	**4,68%**
I02	5	30	535,4	554,375	335,8	**701,7**	528,5	487,5	330,0	9,5	**674,8**	**3,83%**
I02	5	40	695,0	774,625	469,0	**927,4**	661,6	692,625	453,8	11,1	**869,4**	**6,21%**
I02	5	50	826,6	930,5	578,0	**1105,8**	803,1	849,875	560,9	12,0	**1058,1**	**4,33%**
I02	5	60	944,6	1085	672,4	**1270,1**	922,4	1017,375	660,0	11,4	**1227,6**	**3,37%**
I02	5	70	1111,3	1328,875	779,6	**1509,9**	1085,6	1241,5	771,3	13,1	**1458,1**	**3,44%**
I02	5	80	1257,4	1438,625	915,1	**1689,0**	1225,8	1357,25	893,9	11,4	**1632,9**	**3,32%**
I02	5	90	1370,4	1677	1021,0	**1873,5**	1353,3	1596	1019,3	10,5	**1832,1**	**2,21%**
I02	5	100	1527,4	1820,375	1155,1	**2073,5**	1505,9	1724,125	1142,6	13,5	**2023,1**	**2,43%**
I22	5	10	204,8	195,3	122,125	**263,3**	197,9	166,8	110,375	5,6	**247,9**	**5,83%**
I22	5	20	404,9	383,1	235,875	**519,8**	392,4	306,1	231,375	11,6	**484,2**	**6,80%**
I22	5	30	535,4	554,4	335,75	**701,7**	526,8	449,4	320,5	16,5	**661,6**	**5,72%**
I22	5	40	695,0	774,6	469	**927,4**	668,0	676,5	454,25	15,8	**871,0**	**6,09%**
I22	5	50	826,6	930,5	578	**1105,8**	809,1	819,4	570,125	18,8	**1054,9**	**4,62%**
I22	5	60	944,6	1085,0	672,375	**1270,1**	915,1	961,4	652	22,4	**1203,5**	**5,28%**
I22	5	70	1111,3	1328,9	779,625	**1509,9**	1084,4	1194,9	770,125	20,9	**1442,8**	**4,47%**
I22	5	80	1257,4	1438,6	915,125	**1689,0**	1229,0	1363,3	899,875	15,3	**1638,0**	**3,06%**
I22	5	90	1370,4	1677,0	1021	**1873,5**	1355,8	1604,4	1017,625	10,4	**1837,1**	**1,95%**
I22	5	100	1527,4	1820,4	1155,125	**2073,5**	1503,6	1774,1	1142,625	9,0	**2035,9**	**1,81%**
C2	5	10	204,8	195,25	122,1	**263,3**	197,9	170,625	110,9	4,1	**249,1**	**5,38%**
C2	5	20	404,9	383,125	235,9	**519,8**	393,5	319,25	226,9	8,6	**489,3**	**5,82%**
C2	5	30	535,4	554,375	335,8	**701,7**	525,4	472,25	321,1	11,9	**667,1**	**4,93%**
C2	5	40	695,0	774,625	469,0	**927,4**	663,8	697,75	459,3	10,9	**873,1**	**5,87%**
C2	5	50	826,6	930,5	578,0	**1105,8**	806,6	849	563,6	13,9	**1061,3**	**4,02%**
C2	5	60	944,6	1085	672,4	**1270,1**	926,8	1027,5	661,6	8,5	**1235,0**	**2,79%**
C2	5	70	1111,3	1328,875	779,6	**1509,9**	1088,8	1264,375	774,1	9,8	**1468,1**	**2,78%**
C2	5	80	1257,4	1438,625	915,1	**1689,0**	1227,5	1374,75	897,8	10,3	**1639,9**	**2,92%**
C2	5	90	1370,4	1677	1021,0	**1873,5**	1353,0	1619,125	1011,4	8,8	**1838,7**	**1,86%**
C2	5	100	1527,4	1820,375	1155,1	**2073,5**	1507,5	1776,625	1149,3	6,8	**2040,5**	**1,59%**
C3	5	10	204,8	195,3	122,125	**263,3**	197,9	167,3	110,5	5,0	**248,1**	**5,78%**
C3	5	20	404,9	383,1	235,875	**519,8**	393,0	303,5	226,125	10,8	**484,1**	**6,82%**
C3	5	30	535,4	554,4	335,75	**701,7**	526,1	452,0	326,875	16,1	**661,7**	**5,70%**
C3	5	40	695,0	774,6	469	**927,4**	662,3	664,4	441	15,9	**861,6**	**7,04%**
C3	5	50	826,6	930,5	578	**1105,8**	801,1	809,5	550	20,1	**1044,0**	**5,61%**
C3	5	60	944,6	1085,0	672,375	**1270,1**	911,0	964,6	662	22,6	**1200,4**	**5,48%**
C3	5	70	1111,3	1328,9	779,625	**1509,9**	1076,3	1171,3	763,5	24,8	**1427,6**	**5,45%**
C3	5	80	1257,4	1438,6	915,125	**1689,0**	1216,6	1323,8	901,625	19,8	**1613,8**	**4,47%**
C3	5	90	1370,4	1677,0	1021	**1873,5**	1351,6	1570,1	1001,375	16,5	**1822,7**	**2,71%**
C3	5	100	1527,4	1820,4	1155,125	**2073,5**	1490,8	1727,3	1131,75	16,3	**2008,9**	**3,11%**
C4	5	10	204,8	195,3	122,125	**263,3**	197,9	166,8	110,375	5,6	**247,9**	**5,83%**
C4	5	20	404,9	383,1	235,875	**519,8**	393,0	306,6	228,875	13,1	**485,0**	**6,65%**
C4	5	30	535,4	554,4	335,75	**701,7**	527,5	437,6	322,625	18,4	**658,8**	**6,12%**
C4	5	40	695,0	774,6	469	**927,4**	655,0	623,3	437,5	21,9	**842,0**	**9,11%**
C4	5	50	826,6	930,5	578	**1105,8**	792,5	771,0	549,125	28,4	**1023,8**	**7,43%**
C4	5	60	944,6	1085,0	672,375	**1270,1**	912,8	895,4	653,25	32,0	**1181,4**	**7,01%**
C4	5	70	1111,3	1328,9	779,625	**1509,9**	1065,1	1133,3	757,875	32,4	**1405,1**	**6,95%**
C4	5	80	1257,4	1438,6	915,125	**1689,0**	1200,0	1252,5	886,875	33,1	**1575,8**	**6,71%**
C4	5	90	1370,4	1677,0	1021	**1873,5**	1343,6	1493,4	1008,5	28,3	**1791,6**	**4,36%**
C4	5	100	1527,4	1820,4	1155,125	**2073,5**	1484,4	1703,9	1133,375	21,9	**1995,5**	**3,76%**

Table A.30: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.5 Paced flowshop production line

			Permutation flowshop	Non-permutation flowshop		
Case	Stations	Jobs	Obj-Func	Job-Ch	Obj-Func	Improv
I22	5	10	**36,9**	5,3	**28,8**	**22,01%**
I22	5	20	**68,7**	11,8	**58,0**	**15,57%**
I22	5	30	**97,8**	15,1	**83,0**	**15,21%**
I22	5	40	**142,2**	15,6	**125,9**	**11,58%**
I22	5	50	**174,3**	15,4	**158,1**	**9,31%**
I22	5	60	**208,5**	16,6	**192,5**	**7,68%**
I22	5	70	**243,0**	9,3	**232,5**	**4,35%**
I22	5	80	**276,2**	7,8	**264,8**	**4,15%**
I22	5	90	**317,6**	8,9	**307,4**	**3,25%**
I22	5	100	**350,1**	6,6	**340,6**	**2,73%**
C4	5	10	**36,9**	8,0	**27,7**	**25,09%**
C4	5	20	**68,7**	11,4	**57,5**	**16,33%**
C4	5	30	**97,8**	18,6	**79,1**	**19,13%**
C4	5	40	**142,2**	18,1	**120,3**	**15,55%**
C4	5	50	**174,3**	17,6	**155,3**	**10,95%**
C4	5	60	**208,5**	19,9	**186,0**	**10,80%**
C4	5	70	**243,0**	15,0	**226,5**	**6,79%**
C4	5	80	**276,2**	11,6	**263,4**	**4,63%**
C4	5	90	**317,6**	9,6	**306,5**	**3,50%**
C4	5	100	**350,1**	7,4	**341,7**	**2,41%**
C3	5	10	**36,9**	5,8	**29,2**	**21,01%**
C3	5	20	**68,7**	9,8	**59,8**	**12,97%**
C3	5	30	**97,8**	14,1	**81,8**	**16,40%**
C3	5	40	**142,2**	12,4	**127,6**	**10,35%**
C3	5	50	**174,3**	13,6	**159,3**	**8,66%**
C3	5	60	**208,5**	11,6	**194,2**	**6,89%**
C3	5	70	**243,0**	10,0	**231,1**	**4,90%**
C3	5	80	**276,2**	8,4	**265,5**	**3,87%**
C3	5	90	**317,6**	6,0	**312,0**	**1,80%**
C3	5	100	**350,1**	8,1	**341,0**	**2,63%**
C2	5	10	**36,9**	4,1	**30,6**	**17,07%**
C2	5	20	**68,7**	6,4	**61,8**	**10,05%**
C2	5	30	**97,8**	8,5	**87,6**	**10,46%**
C2	5	40	**142,2**	7,8	**134,0**	**5,82%**
C2	5	50	**174,3**	5,8	**165,9**	**4,86%**
C2	5	60	**208,5**	6,6	**200,1**	**4,02%**
C2	5	70	**243,0**	4,9	**237,7**	**2,17%**
C2	5	80	**276,2**	4,8	**270,4**	**2,11%**
C2	5	90	**317,6**	4,0	**312,5**	**1,63%**
C2	5	100	**350,1**	3,5	**345,2**	**1,41%**

Table A.31: The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.5.1 Benchmark comparison for Watson, Barbulescu, Whitley, Howe

Instance 1: 20 jobs, 20 stations

		GA Genetic Algorithm							
Instance	Best known solution	(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
		Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	2228	2232	**2223**	**2223**	-0,18%	**0,22%**	**0,22%**	0,40%	0,40%
19	2151	2156	2151	2151	-0,23%	0,00%	0,00%	0,23%	0,23%
18	2290	2298	**2284**	2296	-0,35%	**0,26%**	-0,26%	0,61%	0,09%
17	2281	2286	2286	2286	-0,22%	-0,22%	-0,22%	0,00%	0,00%
16	2101	2115	2106	2107	-0,67%	-0,24%	-0,29%	0,43%	0,38%
15	2272	2287	2283	2282	-0,66%	-0,48%	-0,44%	0,17%	0,22%
14	2185	2192	2188	2188	-0,32%	-0,14%	-0,14%	0,18%	0,18%
13	2300	2317	2308	2310	-0,74%	-0,35%	-0,43%	0,39%	0,30%
12	2188	2194	**2183**	**2183**	-0,27%	**0,23%**	**0,23%**	0,50%	0,50%
11	2299	2311	2307	2307	-0,52%	-0,35%	-0,35%	0,17%	0,17%
10	2111	2118	2114	2114	-0,33%	-0,14%	-0,14%	0,19%	0,19%
9	2258	2271	2260	2260	-0,58%	-0,09%	-0,09%	0,48%	0,48%
8	2096	2099	**2093**	**2091**	-0,14%	**0,14%**	**0,24%**	0,29%	0,38%
7	2244	2256	**2237**	**2237**	-0,53%	**0,31%**	**0,31%**	0,84%	0,84%
6	2194	2205	2201	2198	-0,50%	-0,32%	-0,18%	0,18%	0,32%
5	2207	2207	2207	2207	0,00%	0,00%	0,00%	0,00%	0,00%
4	2142	2144	**2133**	**2133**	-0,09%	**0,42%**	**0,42%**	0,51%	0,51%
3	2307	2327	2323	2324	-0,87%	-0,69%	-0,74%	0,17%	0,13%
2	2155	2155	2155	2155	0,00%	0,00%	0,00%	0,00%	0,00%
1	2213	2221	2219	2215	-0,36%	-0,27%	-0,09%	0,09%	0,27%

Table A.32: Comparison of solutions with the best known solution for permutation sequences for 20 jobs and 20 stations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal solution or the sofar best known solution.

	GA Genetic Algorithm				
Instance	(Absolute improvement to best known solution)			(Improvement to Perm)	
	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	2848,30	2839,60	2840,60	0,31%	0,27%
19	2765,30	2748,60	2748,60	0,60%	0,60%
18	2910,30	2900,00	2900,10	0,35%	0,35%
17	2905,00	2889,00	2889,00	0,55%	0,55%
16	2726,20	2711,60	2711,60	0,54%	0,54%
15	2901,00	2890,40	2890,40	0,37%	0,37%
14	2804,40	2802,00	2802,00	0,09%	0,09%
13	2928,50	2918,10	2918,40	0,36%	0,34%
12	2800,10	2790,60	2790,60	0,34%	0,34%
11	2920,20	2902,80	2902,80	0,60%	0,60%
10	2728,70	2713,70	2713,70	0,55%	0,55%
9	2889,50	2865,80	2865,80	0,82%	0,82%
8	2713,40	2700,20	2706,80	0,49%	0,24%
7	2876,60	2867,30	2863,60	0,32%	0,45%
6	2824,50	2814,90	2809,80	0,34%	0,52%
5	2850,60	2840,40	2839,70	0,36%	0,38%
4	2760,80	2725,60	2725,60	1,27%	1,27%
3	2924,10	2916,50	2916,90	0,26%	0,25%
2	2754,60	2745,70	2745,70	0,32%	0,32%
1	2820,70	2809,20	2809,20	0,41%	0,41%

Table A.33: Comparison of solutions with the permutation case for 20 jobs and 20 stations, taking into account setup considerations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

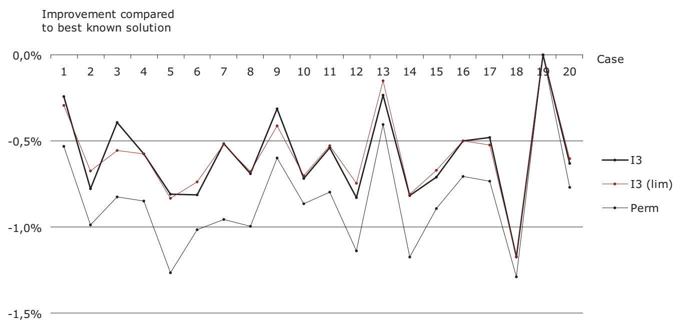


Figure A.2: Comparison of the average of eight runs with the best known solution for permutation sequences for 20 jobs and 20 stations.

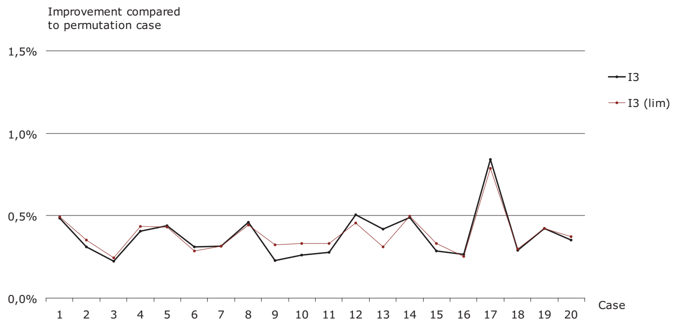


Figure A.3: Comparison of the average of eight runs with the permutation case for 20 jobs and 20 stations, with setup considerations.

		GA Genetic Algorithm							
Instance	Best known solution	(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
		Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	2228	2239,88	2233,38	2234,50	-0,53%	-0,24%	-0,29%	0,29%	0,24%
19	2151	2172,25	2167,75	2165,50	-0,99%	-0,78%	-0,67%	0,21%	0,31%
18	2290	2308,88	2299,00	2302,75	-0,82%	-0,39%	-0,56%	0,43%	0,27%
17	2281	2300,38	2294,13	2294,13	-0,85%	-0,58%	-0,58%	0,27%	0,27%
16	2101	2127,63	2118,00	2118,50	-1,27%	-0,81%	-0,83%	0,45%	0,43%
15	2272	2295,13	2290,50	2288,75	-1,02%	-0,81%	-0,74%	0,20%	0,28%
14	2185	2205,88	2196,25	2196,38	-0,96%	-0,51%	-0,52%	0,44%	0,43%
13	2300	2322,88	2315,88	2315,63	-0,99%	-0,69%	-0,68%	0,30%	0,31%
12	2188	2201,13	2194,88	2197,00	-0,60%	-0,31%	-0,41%	0,28%	0,19%
11	2299	2318,88	2315,50	2315,13	-0,86%	-0,72%	-0,70%	0,15%	0,16%
10	2111	2127,88	2122,38	2122,13	-0,80%	-0,54%	-0,53%	0,26%	0,27%
9	2258	2283,75	2276,75	2274,88	-1,14%	-0,83%	-0,75%	0,31%	0,39%
8	2096	2104,50	2100,88	2099,13	-0,41%	-0,23%	-0,15%	0,17%	0,26%
7	2244	2270,38	2262,38	2262,13	-1,18%	-0,82%	-0,81%	0,35%	0,36%
6	2194	2213,63	2209,63	2208,75	-0,89%	-0,71%	-0,67%	0,18%	0,22%
5	2207	2222,63	2218,00	2218,00	-0,71%	-0,50%	-0,50%	0,21%	0,21%
4	2142	2157,75	2152,25	2153,25	-0,74%	-0,48%	-0,53%	0,25%	0,21%
3	2307	2336,75	2334,13	2334,00	-1,29%	-1,18%	-1,17%	0,11%	0,12%
2	2155	2155,00	2155,00	2155,00	0,00%	0,00%	0,00%	0,00%	0,00%
1	2213	2230,00	2227,00	2226,38	-0,77%	-0,63%	-0,60%	0,13%	0,16%

Table A.34: Comparison of the average of eight runs with the best known solution for permutation sequences for 20 jobs and 20 stations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data. Fields with grey background indicate that the obtained solution is equal to the optimal solution or the sofar best known solution.

Instance	GA Genetic Algorithm				
	(Absolute improvement to best known solution)			(Improvement to Perm)	
	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	2864,60	2850,70	2850,45	0,49%	0,49%
19	2771,74	2763,16	2761,93	0,31%	0,35%
18	2916,76	2910,28	2909,66	0,22%	0,24%
17	2913,33	2901,50	2900,66	0,41%	0,43%
16	2737,16	2725,19	2725,36	0,44%	0,43%
15	2919,49	2910,43	2911,19	0,31%	0,28%
14	2817,70	2808,81	2808,81	0,32%	0,32%
13	2940,08	2926,60	2926,99	0,46%	0,45%
12	2814,19	2807,75	2805,04	0,23%	0,33%
11	2926,00	2918,39	2916,26	0,26%	0,33%
10	2735,01	2727,41	2725,93	0,28%	0,33%
9	2902,86	2888,15	2889,66	0,51%	0,45%
8	2722,04	2710,61	2713,60	0,42%	0,31%
7	2890,21	2876,05	2875,86	0,49%	0,50%
6	2827,85	2819,75	2818,50	0,29%	0,33%
5	2857,08	2849,46	2849,86	0,27%	0,25%
4	2769,39	2746,13	2747,55	0,84%	0,79%
3	2927,83	2919,39	2919,14	0,29%	0,30%
2	2758,88	2747,24	2747,24	0,42%	0,42%
1	2828,43	2818,46	2817,84	0,35%	0,37%

Table A.35: Comparison of the average of eight runs with the permutation case for 20 jobs and 20 stations, taking into account setup considerations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

Instance 2: 50 jobs, 20 stations

		GA Genetic Algorithm							
Instance	Best known solution	(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
		Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	3768	3873	3838	3826	-2,79%	-1,86%	-1,54%	0,90%	1,21%
19	3754	3838	3819	3822	-2,24%	-1,73%	-1,81%	0,50%	0,42%
18	3822	3921	3875	3867	-2,59%	-1,39%	-1,18%	1,17%	1,38%
17	3644	3802	3734	3737	-4,34%	-2,47%	-2,55%	1,79%	1,71%
16	3925	4031	4000	4011	-2,70%	-1,91%	-2,19%	0,77%	0,50%
15	3818	3919	3867	3865	-2,65%	-1,28%	-1,23%	1,33%	1,38%
14	3741	3841	3790	3794	-2,67%	-1,31%	-1,42%	1,33%	1,22%
13	3865	3984	3940	3953	-3,08%	-1,94%	-2,28%	1,10%	0,78%
12	3573	3704	3664	3667	-3,67%	-2,55%	-2,63%	1,08%	1,00%
11	3767	3907	3860	3868	-3,72%	-2,47%	-2,68%	1,20%	1,00%
10	3923	4039	3998	3997	-2,96%	-1,91%	-1,89%	1,02%	1,04%
9	3795	3909	3860	3878	-3,00%	-1,71%	-2,19%	1,25%	0,79%
8	3702	3829	3788	3788	-3,43%	-2,32%	-2,32%	1,07%	1,07%
7	3713	3843	3790	3787	-3,50%	-2,07%	-1,99%	1,38%	1,46%
6	3904	4021	3984	3988	-3,00%	-2,05%	-2,15%	0,92%	0,82%
5	3830	3960	3925	3916	-3,39%	-2,48%	-2,25%	0,88%	1,11%
4	3821	3929	3892	3884	-2,83%	-1,86%	-1,65%	0,94%	1,15%
3	3731	3849	3829	3836	-3,16%	-2,63%	-2,81%	0,52%	0,34%
2	3599	3745	3695	3692	-4,06%	-2,67%	-2,58%	1,34%	1,42%
1	3860	3966	3947	3957	-2,75%	-2,25%	-2,51%	0,48%	0,23%

Table A.36: Comparison of solutions with the best known solution for permutation sequences for 50 jobs and 20 stations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	GA Genetic Algorithm				
Instance	(Absolute improvement to best known solution)			(Improvement to Perm)	
	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	5366,20	5301,10	5312,40	1,21%	1,00%
19	5419,30	5317,60	5327,90	1,88%	1,69%
18	5432,10	5353,50	5353,50	1,45%	1,45%
17	5316,70	5229,60	5204,30	1,64%	2,11%
16	5551,00	5490,20	5485,30	1,10%	1,18%
15	5454,30	5379,20	5358,10	1,38%	1,76%
14	5381,00	5285,00	5291,10	1,78%	1,67%
13	5497,40	5402,60	5415,00	1,72%	1,50%
12	5261,40	5172,80	5179,40	1,68%	1,56%
11	5441,00	5342,00	5348,40	1,82%	1,70%
10	5539,30	5461,20	5443,20	1,41%	1,73%
9	5441,40	5357,50	5363,90	1,54%	1,42%
8	5371,20	5283,90	5265,30	1,63%	1,97%
7	5346,20	5272,10	5283,70	1,39%	1,17%
6	5544,00	5471,30	5474,70	1,31%	1,25%
5	5479,20	5379,10	5402,80	1,83%	1,39%
4	5496,10	5414,60	5429,90	1,48%	1,20%
3	5381,00	5305,80	5315,70	1,40%	1,21%
2	5281,20	5198,40	5219,20	1,57%	1,17%
1	5481,90	5409,10	5387,10	1,33%	1,73%

Table A.37: Comparison of solutions with the permutation case for 50 jobs and 20 stations, taking into account setup considerations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

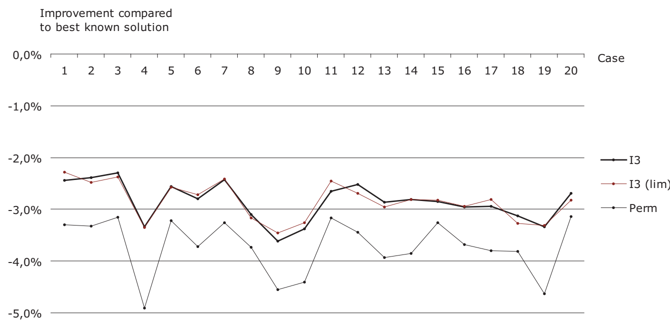


Figure A.4: Comparison of the average of eight runs with the best known solution for permutation sequences for 50 jobs and 20 stations.

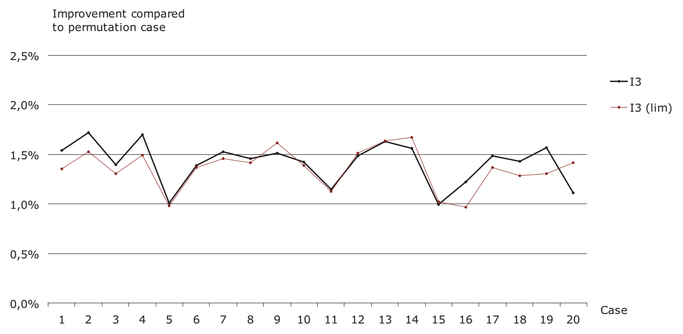


Figure A.5: Comparison of the average of eight runs with the permutation case for 50 jobs and 20 stations, with setup considerations.

		GA Genetic Algorithm							
Instance	Best known solution	(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
		Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	3768	3892,38	3859,88	3854,13	-3,30%	-2,44%	-2,29%	0,83%	0,98%
19	3754	3878,75	3843,63	3846,88	-3,32%	-2,39%	-2,47%	0,91%	0,82%
18	3822	3942,75	3909,75	3912,63	-3,16%	-2,30%	-2,37%	0,84%	0,76%
17	3644	3823,00	3765,50	3766,25	-4,91%	-3,33%	-3,35%	1,50%	1,48%
16	3925	4051,25	4025,25	4026,13	-3,22%	-2,55%	-2,58%	0,64%	0,62%
15	3818	3960,00	3925,00	3921,63	-3,72%	-2,80%	-2,71%	0,88%	0,97%
14	3741	3862,75	3831,75	3831,13	-3,25%	-2,43%	-2,41%	0,80%	0,82%
13	3865	4009,25	3984,75	3987,63	-3,73%	-3,10%	-3,17%	0,61%	0,54%
12	3573	3735,63	3702,38	3696,38	-4,55%	-3,62%	-3,45%	0,89%	1,05%
11	3767	3932,88	3894,00	3889,63	-4,40%	-3,37%	-3,26%	0,99%	1,10%
10	3923	4047,00	4027,00	4019,25	-3,16%	-2,65%	-2,45%	0,49%	0,69%
9	3795	3925,88	3890,75	3897,00	-3,45%	-2,52%	-2,69%	0,89%	0,74%
8	3702	3847,50	3808,13	3811,25	-3,93%	-2,87%	-2,95%	1,02%	0,94%
7	3713	3855,88	3817,25	3817,50	-3,85%	-2,81%	-2,81%	1,00%	1,00%
6	3904	4031,13	4015,25	4014,13	-3,26%	-2,85%	-2,82%	0,39%	0,42%
5	3830	3970,88	3943,00	3942,50	-3,68%	-2,95%	-2,94%	0,70%	0,71%
4	3821	3966,38	3933,38	3928,13	-3,80%	-2,94%	-2,80%	0,83%	0,96%
3	3731	3873,13	3847,75	3853,25	-3,81%	-3,13%	-3,28%	0,66%	0,51%
2	3599	3765,88	3719,00	3718,13	-4,64%	-3,33%	-3,31%	1,24%	1,27%
1	3860	3981,00	3964,00	3968,75	-3,13%	-2,69%	-2,82%	0,43%	0,31%

Table A.38: Comparison of the average of eight runs with the best known solution for permutation sequences for 50 jobs and 20 stations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

	GA Genetic Algorithm				
Instance	(Absolute improvement to best known solution)			(Improvement to Perm)	
	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	5420,04	5336,58	5346,61	1,54%	1,35%
19	5437,94	5344,35	5354,98	1,72%	1,53%
18	5474,05	5397,75	5402,75	1,39%	1,30%
17	5355,26	5264,16	5275,29	1,70%	1,49%
16	5573,78	5517,71	5519,18	1,01%	0,98%
15	5488,08	5411,71	5413,08	1,39%	1,37%
14	5405,55	5323,13	5326,63	1,52%	1,46%
13	5528,75	5448,16	5450,49	1,46%	1,42%
12	5286,59	5206,50	5201,33	1,51%	1,61%
11	5463,50	5385,86	5387,74	1,42%	1,39%
10	5559,31	5495,74	5496,71	1,14%	1,13%
9	5464,21	5383,04	5381,40	1,49%	1,52%
8	5390,54	5302,85	5302,40	1,63%	1,64%
7	5401,14	5316,93	5310,71	1,56%	1,67%
6	5560,65	5505,41	5503,78	0,99%	1,02%
5	5496,18	5428,81	5442,91	1,23%	0,97%
4	5522,40	5440,56	5446,98	1,48%	1,37%
3	5402,94	5325,53	5333,41	1,43%	1,29%
2	5309,19	5225,98	5239,78	1,57%	1,31%
1	5503,95	5442,83	5426,18	1,11%	1,41%

Table A.39: Comparison of the average of eight runs with the permutation case for 50 jobs and 20 stations, taking into account setup considerations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

Instance 3: 100 jobs, 20 stations

		GA Genetic Algorithm							
		(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
Instance	Best known solution	Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	6328	6594	6568	6570	-4,20%	-3,79%	-3,82%	0,39%	0,36%
19	6430	6628	6586	6589	-3,08%	-2,43%	-2,47%	0,63%	0,59%
18	6415	6666	6648	6623	-3,91%	-3,63%	-3,24%	0,27%	0,65%
17	6456	6642	6626	6632	-2,88%	-2,63%	-2,73%	0,24%	0,15%
16	6299	6433	6419	6421	-2,13%	-1,91%	-1,94%	0,22%	0,19%
15	6407	6614	6578	6588	-3,23%	-2,67%	-2,83%	0,54%	0,39%
14	6448	6684	6630	6651	-3,66%	-2,82%	-3,15%	0,81%	0,49%
13	6192	6457	6416	6424	-4,28%	-3,62%	-3,75%	0,63%	0,51%
12	6314	6482	6453	6454	-2,66%	-2,20%	-2,22%	0,45%	0,43%
11	6264	6514	6477	6464	-3,99%	-3,40%	-3,19%	0,57%	0,77%
10	6216	6466	6425	6426	-4,02%	-3,36%	-3,38%	0,63%	0,62%
9	6476	6695	6663	6663	-3,38%	-2,89%	-2,89%	0,48%	0,48%
8	6183	6392	6369	6356	-3,38%	-3,01%	-2,80%	0,36%	0,56%
7	6214	6495	6446	6453	-4,52%	-3,73%	-3,85%	0,75%	0,65%
6	6301	6538	6480	6494	-3,76%	-2,84%	-3,06%	0,89%	0,67%
5	6495	6722	6690	6680	-3,49%	-3,00%	-2,85%	0,48%	0,62%
4	6514	6647	6623	6623	-2,04%	-1,67%	-1,67%	0,36%	0,36%
3	6415	6607	6573	6583	-2,99%	-2,46%	-2,62%	0,51%	0,36%
2	6417	6640	6617	6602	-3,48%	-3,12%	-2,88%	0,35%	0,57%
1	6328	6581	6530	6532	-4,00%	-3,19%	-3,22%	0,77%	0,74%

Table A.40: Comparison of solutions with the best known solution for permutation sequences for 100 jobs and 20 stations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	GA Genetic Algorithm				
	(Absolute improvement to best known solution)			(Improvement to Perm)	
Instance	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	9669,00	9465,40	9502,30	2,11%	1,72%
19	9721,30	9522,60	9589,70	2,04%	1,35%
18	9763,00	9590,80	9593,30	1,76%	1,74%
17	9667,50	9496,70	9506,30	1,77%	1,67%
16	9489,90	9282,90	9305,60	2,18%	1,94%
15	9654,60	9443,70	9475,70	2,18%	1,85%
14	9734,80	9527,30	9587,00	2,13%	1,52%
13	9530,70	9329,90	9330,90	2,11%	2,10%
12	9549,70	9368,40	9388,10	1,90%	1,69%
11	9524,90	9372,30	9416,50	1,60%	1,14%
10	9558,40	9347,90	9366,00	2,20%	2,01%
9	9731,30	9542,70	9538,00	1,94%	1,99%
8	9518,40	9301,90	9323,90	2,27%	2,04%
7	9573,40	9387,70	9373,30	1,94%	2,09%
6	9602,90	9413,30	9419,20	1,97%	1,91%
5	9770,20	9583,80	9606,10	1,91%	1,68%
4	9663,80	9448,80	9508,00	2,22%	1,61%
3	9699,10	9508,40	9573,50	1,97%	1,29%
2	9713,30	9488,00	9569,60	2,32%	1,48%
1	9656,90	9448,00	9483,50	2,16%	1,80%

Table A.41: Comparison of solutions with the permutation case for 100 jobs and 20 stations, taking into account setup considerations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

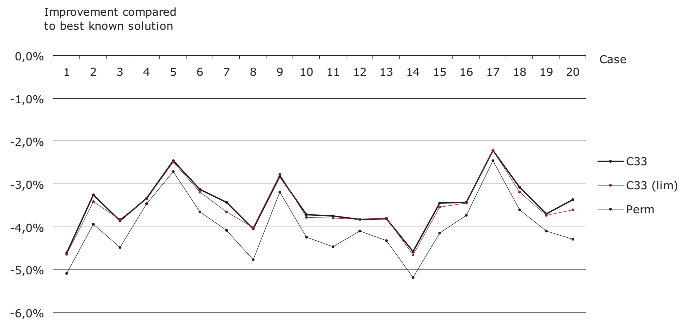


Figure A.6: Comparison of the average of eight runs with the best known solution for permutation sequences for 100 jobs and 20 stations.

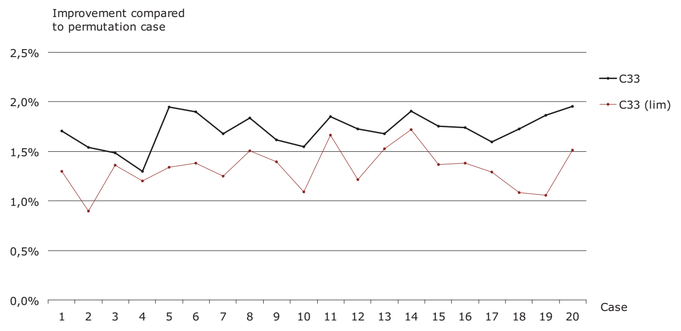


Figure A.7: Comparison of the average of eight runs with the permutation case for 100 jobs and 20 stations, with setup considerations.

Instance	Best known solution	GA Genetic Algorithm							
		(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
		Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	6328	6650,00	6620,25	6622,13	-5,09%	-4,62%	-4,65%	0,45%	0,42%
19	6430	6683,00	6639,50	6649,88	-3,93%	-3,26%	-3,42%	0,65%	0,50%
18	6415	6702,50	6662,50	6660,75	-4,48%	-3,86%	-3,83%	0,60%	0,62%
17	6456	6680,00	6671,00	6672,13	-3,47%	-3,33%	-3,35%	0,13%	0,12%
16	6299	6470,00	6453,50	6456,25	-2,71%	-2,45%	-2,50%	0,26%	0,21%
15	6407	6641,63	6607,38	6611,75	-3,66%	-3,13%	-3,20%	0,52%	0,45%
14	6448	6711,63	6668,75	6683,13	-4,09%	-3,42%	-3,65%	0,64%	0,42%
13	6192	6487,50	6443,13	6441,38	-4,77%	-4,06%	-4,03%	0,68%	0,71%
12	6314	6515,63	6492,75	6489,63	-3,19%	-2,83%	-2,78%	0,35%	0,40%
11	6264	6530,25	6496,75	6501,38	-4,25%	-3,72%	-3,79%	0,51%	0,44%
10	6216	6493,25	6448,75	6452,38	-4,46%	-3,74%	-3,80%	0,69%	0,63%
9	6476	6742,00	6724,00	6724,00	-4,11%	-3,83%	-3,83%	0,27%	0,27%
8	6183	6450,13	6418,88	6418,25	-4,32%	-3,81%	-3,80%	0,48%	0,49%
7	6214	6536,63	6498,13	6503,13	-5,19%	-4,57%	-4,65%	0,59%	0,51%
6	6301	6562,75	6517,75	6524,63	-4,15%	-3,44%	-3,55%	0,69%	0,58%
5	6495	6737,13	6717,63	6719,00	-3,73%	-3,43%	-3,45%	0,29%	0,27%
4	6514	6674,50	6658,13	6658,50	-2,46%	-2,21%	-2,22%	0,25%	0,24%
3	6415	6646,25	6612,63	6620,00	-3,60%	-3,08%	-3,20%	0,51%	0,39%
2	6417	6680,50	6654,50	6656,38	-4,11%	-3,70%	-3,73%	0,39%	0,36%
1	6328	6599,88	6541,13	6555,75	-4,30%	-3,37%	-3,60%	0,89%	0,67%

Table A.42: Comparison of the average of eight runs with the best known solution for permutation sequences for 100 jobs and 20 stations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

Instance	GA Genetic Algorithm				
	(Absolute improvement to best known solution)			(Improvement to Perm)	
	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	9731,04	9564,74	9604,54	1,71%	1,30%
19	9772,75	9622,38	9684,90	1,54%	0,90%
18	9785,38	9639,76	9652,54	1,49%	1,36%
17	9697,58	9571,45	9580,73	1,30%	1,20%
16	9543,73	9358,19	9416,06	1,94%	1,34%
15	9696,56	9512,30	9562,80	1,90%	1,38%
14	9778,85	9614,64	9656,84	1,68%	1,25%
13	9574,55	9398,49	9430,26	1,84%	1,51%
12	9589,95	9434,94	9456,01	1,62%	1,40%
11	9619,98	9471,36	9515,31	1,54%	1,09%
10	9575,33	9398,36	9416,18	1,85%	1,66%
9	9779,10	9610,35	9660,44	1,73%	1,21%
8	9545,73	9385,58	9399,95	1,68%	1,53%
7	9619,20	9435,95	9453,58	1,91%	1,72%
6	9642,04	9473,16	9510,00	1,75%	1,37%
5	9803,56	9633,00	9668,34	1,74%	1,38%
4	9718,20	9563,48	9592,43	1,59%	1,29%
3	9740,20	9571,84	9634,80	1,73%	1,08%
2	9741,13	9559,56	9638,21	1,86%	1,06%
1	9676,45	9487,08	9530,35	1,96%	1,51%

Table A.43: Comparison of the average of eight runs with the permutation case for 100 jobs and 20 stations, taking into account setup considerations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

Instance 4: 200 jobs, 20 stations

		GA Genetic Algorithm							
		(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
Instance	Best known solution	Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	11455	11758	11749	11748	-2,65%	-2,57%	-2,56%	0,08%	0,09%
19	11601	11975	11943	11962	-3,22%	-2,95%	-3,11%	0,27%	0,11%
18	11678	11908	11880	11895	-1,97%	-1,73%	-1,86%	0,24%	0,11%
17	11219	11640	11599	11627	-3,75%	-3,39%	-3,64%	0,35%	0,11%
16	11253	11590	11573	11583	-2,99%	-2,84%	-2,93%	0,15%	0,06%

Table A.44: Comparison of solutions with the best known solution for permutation sequences for 200 jobs and 20 stations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	GA Genetic Algorithm				
	(Absolute improvement to best known solution)			(Improvement to Perm)	
Instance	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	17849,0	17696,6	17702,2	0,85%	0,82%
19	18063,4	17917,8	17919,7	0,81%	0,80%
18	17992,9	17818,2	17805,7	0,97%	1,04%
17	17696,4	17451,1	17548,7	1,39%	0,83%
16	17781,3	17416,5	17549,5	2,05%	1,30%

Table A.45: Comparison of solutions with the permutation case for 200 jobs and 20 stations, taking into account setup considerations. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

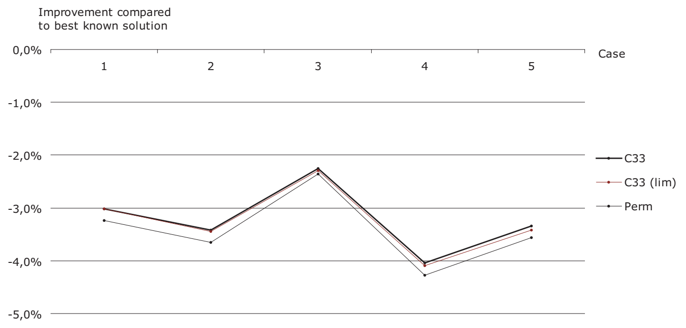


Figure A.8: Comparison of the average of eight runs with the best known solution for permutation sequences for 200 jobs and 20 stations.

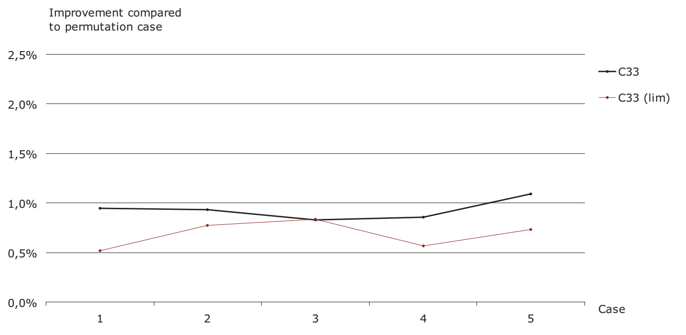


Figure A.9: Comparison of the average of eight runs with the permutation case for 200 jobs and 20 stations, with setup considerations.

		GA Genetic Algorithm							
		(Absolute improvement to best known solution)			(Relative improvement to known best solution)			(Improvement to Perm)	
Instance	Best known solution	Perm	Non-perm	Non-perm (lim)	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	11455	11826,00	11800,38	11801,38	-3,24%	-3,02%	-3,02%	0,22%	0,21%
19	11601	12025,50	11997,50	12000,75	-3,66%	-3,42%	-3,45%	0,23%	0,21%
18	11678	11952,63	11940,88	11946,25	-2,35%	-2,25%	-2,30%	0,10%	0,05%
17	11219	11698,88	11671,75	11678,88	-4,28%	-4,04%	-4,10%	0,23%	0,17%
16	11253	11654,50	11629,63	11637,38	-3,57%	-3,35%	-3,42%	0,21%	0,15%

Table A.46: Comparison of the average of eight runs with the best known solution for permutation sequences for 200 jobs and 20 stations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

	GA Genetic Algorithm				
	(Absolute improvement to best known solution)			(Improvement to Perm)	
Instance	Perm	Non-perm	Non-perm (lim)	Non-perm	Non-perm (lim)
20	17933,59	17763,64	17840,18	0,95%	0,52%
19	18174,70	18005,70	18033,68	0,93%	0,78%
18	18065,68	17916,08	17915,05	0,83%	0,83%
17	17797,46	17644,84	17696,10	0,86%	0,57%
16	17815,36	17620,64	17684,90	1,09%	0,73%

Table A.47: Comparison of the average of eight runs with the permutation case for 200 jobs and 20 stations, taking into account setup considerations. The result of each line is obtained by the average of a cluster of eight experiments. Each of the eight experiments uses a unique seed for the same set of input data.

A.5.2 Benchmark comparison for Brucker, Heitmann, Hurink

Instance 1: $(N = 20,\ M = 5)$

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta010	1108	-	1302	1110,25
ta009	1230	-	1433	1232,88
ta008	1206	-	1436	1207,13
ta007	1234	-	1436	1248,00
ta006	1195	-	1434	1205,00
ta005	1235	-	1370	1239,13
ta004	1293	-	1471	1303,75
ta003	1081	-	1353	1084,63
ta002	1359	-	1451	1359,00
ta001	1278	-	1437	1281,63

Table A.48: Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal to the optimal solution or the sofar best known solution.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta010	1108	-	1103	-	1134	1115	**1103**	**1106,75**	**1107,38**	1109,50	1109,13	1108,38	1109,13
ta009	1230	-	1210	-	1289	1249	**1210**	1232,38	1232,38	1232,88	1232,88	1232,88	1232,88
ta008	1206	-	1199	-	1227	1215	**1199**	**1205,38**	**1205,38**	1207,13	1207,13	1207,13	1207,13
ta007	1234	-	1234	-	1266	1251	1234	1248,00	1248,00	1248,00	1248,00	1248,00	1248,00
ta006	1195	-	1193	-	1268	1217	**1193**	1200,75	1199,25	1205,00	1203,63	1205,00	1203,88
ta005	1235	-	1231	-	1262	1250	**1231**	1238,25	1237,50	1239,13	1239,13	1239,13	1239,13
ta004	1293	-	1292	-	1329	1329	**1292**	1299,38	1299,00	1303,38	1303,38	1303,38	1303,38
ta003	1081	-	1073	-	1132	1098	**1073**	**1080,50**	**1080,63**	1081,50	1082,38	1081,50	1082,38
ta002	1359	-	1358	-	1365	1365	**1358**	1359,00	1359,00	1359,00	1359,00	1359,00	1359,00
ta001	1278	-	1278	-	1287	1287	1278	1279,75	1279,75	1280,75	1281,13	1280,50	1280,75

Table A.49: Non-Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal solution or the sofar best known solution.

Instance 2: $(N = 20, M = 10)$

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta020	1591	-	1806	1613,50
ta019	1593	-	1772	1610,50
ta018	1538	-	1788	1561,13
ta017	1484	-	1673	1490,88
ta016	1397	-	1632	1410,50
ta015	1419	-	1678	1429,13
ta014	1377	-	1620	1388,88
ta013	1496	-	1755	1513,38
ta012	1659	-	1875	1682,50
ta011	1582	-	1758	1595,38

Table A.50: Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta020	1591	-	1559	1525	1632	1642	1642	1610,13	1610,13	1612,25	1611,63	1611,13	1611,38
ta019	1593	-	1586	1558	1672	1628	1626	1609,63	1610,50	1610,25	1610,25	1610,25	1610,25
ta018	1538	-	1527	1446	1582	1585	1580	1551,63	1554,13	1558,13	1558,50	1558,00	1558,63
ta017	1484	-	1428	1400	1559	1521	1505	1484,63	1485,63	**1483,88**	**1483,13**	**1483,13**	1485,75
ta016	1397	-	1369	1347	1424	1413	1419	1406,13	1405,38	1406,13	1407,50	1406,88	1405,25
ta015	1419	-	1413	1374	1501	1476	1463	1424,25	1424,38	1424,63	1425,38	1425,00	1425,50
ta014	1377	-	1368	1356	1433	1413	1402	1387,25	1386,75	1386,25	1385,88	1386,00	1387,38
ta013	1496	-	1486	1450	1565	1544	1540	1511,38	1511,38	1509,00	1509,13	1509,00	1510,13
ta012	1659	-	1644	1603	1763	1737	1737	1675,88	1674,88	1677,13	1677,13	1677,13	1676,75
ta011	1582	-	1560	1549	1681	1659	1659	1592,88	1593,00	1594,63	1594,63	1594,63	1594,13

Table A.51: Non-Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal solution or the sofar best known solution.

Instance 3: $(N = 20, M = 20)$

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta030	2178	-	2361	2200,25
ta029	2237	-	2411	2251,00
ta028	2200	-	2367	2222,25
ta027	2273	-	2436	2294,00
ta026	2226	-	2414	2240,88
ta025	2291	-	2507	2312,88
ta024	2223	-	2389	2234,88
ta023	2326	-	2527	2349,50
ta022	2099	-	2292	2116,38
ta021	2297	-	2512	2316,25

Table A.52: Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta030	2178	-	2178	1992	2257	2257	2257	2187,63	2188,00	2191,25	2191,75	2192,63	2194,13
ta029	2237	-	2227	1941	2310	2298	2300	2248,00	2247,38	2248,25	2248,25	2248,25	2248,25
ta028	2200	-	2183	1961	2247	2249	2249	2218,75	2217,50	2219,38	2219,38	2219,38	2219,00
ta027	2273	-	2267	2010	2383	2383	2347	2289,13	2288,38	2287,38	2285,13	2288,38	2289,63
ta026	2226	-	2221	1971	2285	2270	2349	2234,00	2234,00	2234,75	2233,50	2233,50	2234,38
ta025	2291	-	2291	2086	2365	2369	2369	2307,25	2307,75	2303,63	2304,75	2303,25	2303,25
ta024	2223	-	2223	2001	2242	2242	2242	**2216,63**	**2215,50**	2231,25	2231,25	2231,25	2230,13
ta023	2326	-	2313	2006	2396	2401	2404	2342,13	2342,88	2346,50	2346,88	2346,38	2344,13
ta022	2099	-	2092	1847	2134	2134	2134	2110,50	2110,38	2112,63	2113,63	2112,38	2114,00
ta021	2297	-	2293	2021	2428	2404	2428	2309,25	2306,88	2313,13	2313,13	2313,13	2313,13

Table A.53: Non-Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal solution or the sofar best known solution.

Instance 4: $(N = 50,\ M = 5)$

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta040	2782	-	3350	2782,00
ta039	2552	-	3045	2562,50
ta038	2683	-	3187	2696,63
ta037	2725	-	3166	2730,13
ta036	2829	-	3364	2833,50
ta035	2863	-	3350	2863,88
ta034	2751	-	3334	2762,38
ta033	2621	-	3265	2622,13
ta032	2834	-	3385	2840,38
ta031	2724	-	3238	2724,00

Table A.54: Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal to the optimal solution or the sofar best known solution.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta040	2782	-	2776	-	2856	**2776**	**2776**	2782,00	2782,00	**2776,25**	**2776,13**	**2776,25**	**2776,00**
ta039	2552	-	2545	2542	2599	2558	2559	2561,63	2561,63	2562,50	2562,50	2562,50	2562,50
ta038	2683	-	2683	-	2769	2697	2688	2696,63	2696,63	2696,63	2696,63	2696,63	2696,63
ta037	2725	-	2716	2715	2765	2843	2843	2729,38	2729,38	2730,13	2728,50	2730,13	2729,50
ta036	2829	-	2825	-	2916	2837	2829	2831,25	2831,50	2831,63	2831,75	2832,13	2831,63
ta035	2863	-	2853	-	2918	2871	2872	**2860,38**	**2862,13**	**2854,63**	**2855,38**	**2855,38**	**2855,38**
ta034	2751	-	2751	-	2888	2764	2782	2762,38	2762,38	2762,38	2762,38	2762,38	2762,38
ta033	2621	-	2612	-	2730	2632	2623	2621,63	2621,63	**2618,38**	**2619,00**	**2618,63**	2622,13
ta032	2834	-	2834	-	2913	2877	2882	2840,00	2840,00	2840,38	2840,38	2840,13	2840,38
ta031	2724	-	2724	-	2808	2729	2729	2724,00	2724,00	2724,00	2724,00	2724,00	2724,00

Table A.55: Non-Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used. Fields with grey background indicate that the obtained solution is equal, fields with grey background and bold text that the solution is smaller than the optimal solution or the sofar best known solution.

Instance 5: $(N = 50,\ M = 10)$

Instance	Best Known/ Opt. solution	Lower bound	Brucker b_i = 0	Genetic Algorithm
ta050	3065	-	3816	3163,38
ta049	2897	-	3771	2988,00
ta048	3037	-	3722	3093,38
ta047	3093	-	3789	3168,63
ta046	3006	-	3755	3090,50
ta045	2976	-	3838	3073,38
ta044	3063	-	3844	3110,75
ta043	2839	-	3658	2958,38
ta042	2867	-	3664	2978,63
ta041	2991	-	3806	3091,38

Table A.56: Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

	Permutation		Non-Permutation					Genetic Algorithm					
Instance	Best Known/ Opt. solution	Lower bound	Best Known/ Opt. solution	Lower bound	Brucker b_i = 1	Brucker b_i = 2	Brucker b_i = n	I3	I3 (lim)	C3	C3 (lim)	I22	I22 (lim)
ta050	3065	-	3065	3046	3273	3169	3201	3158,00	3157,13	3160,63	3159,50	3160,25	3161,25
ta049	2897	-	2887	2858	3114	3049	3013	2977,50	2976,63	2980,00	2977,88	2974,75	2978,63
ta048	3037	-	3026	3001	3183	3142	3150	3085,63	3087,88	3091,25	3090,00	3091,63	3091,25
ta047	3093	-	3093	-	3348	3234	3234	3167,13	3167,25	3166,50	3165,38	3165,88	3166,13
ta046	3006	-	2991	2981	3177	3126	3119	3081,75	3084,63	3085,13	3087,50	3086,88	3086,88
ta045	2976	-	2976	2935	3232	3141	3152	3058,13	3055,50	3060,13	3065,13	3062,00	3064,88
ta044	3063	-	3063	3059	3242	3129	3146	3109,63	3110,13	3110,38	3110,13	3110,25	3110,13
ta043	2839	-	2832	2828	3077	2925	2964	2950,63	2950,13	2952,88	2951,50	2950,38	2954,75
ta042	2867	-	2867	2829	3111	3003	3031	2969,38	2968,88	2971,88	2974,25	2972,75	2973,38
ta041	2991	-	2970	2970	3258	3179	3142	3080,25	3080,88	3083,00	3083,13	3082,63	3084,75

Table A.57: Non-Permutation case. The result of each line shows the best solution out of eight runs each. For each of the eight runs a different seed is used.

List of Figures

1.1 Gant-graph of permutation sequence versus non-permutation sequence . . . 2
1.2 Scheme of a flowshop with constrained resequencing buffer places. 3

2.1 Illustration of the demand horizon 40
2.2 Sectioning of static and dynamic demand 41
2.3 Sequencing problems in assembly lines 43

4.1 Constraints of the schedule calculation at a station 85

5.1 CLP structure 1: Non-permutation 95
5.2 Analysis of semi dynamic demand 97
5.3 CLP structure 2: Cascading of Permutation and Non-permutation 98
5.4 CLP structure 3: Multistart Cascading without feedback (1) 99
5.5 CLP structure 5: Multistart cascading with feedback (1) 100
5.6 CLP structure 6: Multistart cascading with feedback (2) 101
5.7 CLP structure 7: Multistart cascading with feedback (3) 104

6.1 Chromosome reduction for the Genetic Algorithm 122
6.2 Data structure used for one generation of the Genetic Algorithm 123
6.3 Overview on the application of the genetic operators 131
6.4 Genetic operator Crossover-I (case 1) 135
6.5 Genetic operator Crossover-I (case 2) 135
6.6 Genetic operator Crossover-II (case 1) 136
6.7 Genetic operator Crossover-II (case 2) 136
6.8 Linear degression of Crossover-I and Crossover-II (case 1) 137
6.9 Linear degression of Crossover-I and Crossover-II (case 2) 138
6.10 Genetic operator Mutation-I (forward) 140
6.11 Genetic operator Mutation-I (backward) 140

6.12 Genetic operator Mutation-II 141
6.13 Preliminary analysis: $p_{\text{c-I}}$ versus $p_{\text{c-II}}$ 144
6.14 Preliminary analysis: $p_{\text{m-I(f)}}$ versus $p_{\text{m-I(b)}}$ 146
6.15 Preliminary analysis: $p_{\text{m-II}}$ versus $p_{\text{m-I(f)}}$ 147
6.16 Variability of the solutions. 147
6.17 Dependency of the number of job changes on the objective function. 148
6.18 Performance, depending on the number of stations (M). 153
6.19 Performance, depending on the number of resequencing possibilities (L). . . 154
6.20 Performance, depending on the number of jobs (N). 155
6.21 Performance, depending on the number of parents (R). 156
6.22 Performance, depending on the number of generations (G). 157

7.1 Consideration of setup (SC/ST) for the intermediate case. 161
7.2 Consideration of setup (SC/ST) for the centralized case. 162
7.3 Intermediate versus centralized buffer location: Objective function. 163
7.4 Intermediate versus centralized buffer location: Number of jobchanges . . . 164
7.5 Number of buffer places for the centralized case: Objective Function 164
7.6 Number of buffer places for the centralized case: Number of jobchanges . . 165
7.7 Variation of the physical size of the buffer places for the intermediate case. 166
7.8 Variation of the physical size of the buffer places for the centralized case. . 167
7.9 Range analysis for the intermediate case. 168
7.10 Range analysis for the centralized case. 169
7.11 Conversion of the homogenous data to spread out data. 170
7.12 Analysis of spreading out the input data for the intermediate case. 170
7.13 Analysis of spreading out the input data for the centralized case. 171
7.14 Probability density function $\varphi(x)$. 172
7.15 Analysis of an increasing standard deviation for the intermediate case. . . . 172
7.16 Analysis of an increasing standard deviation for the centralized case. 173
7.17 Semi dynamic demand for the intermediate case. 174
7.18 Semi dynamic demand for the centralized case. 174
7.19 Paced production line. 175
7.20 Watson: Comparison with best known solutions for permutation sequences (20 jobs, 20 stations) 178
7.21 Watson: Comparison with permutation case (20 jobs, 20 stations, with setup considerations) 178

7.22 Watson: Comparison with best known solutions for permutation sequences (50 jobs, 20 stations) 179

7.23 Watson: Comparison with permutation case (50 jobs, 20 stations, with setup considerations) 179

7.24 Watson: Comparison with best known solutions for permutation sequences (100 jobs, 20 stations) 180

7.25 Watson: Comparison with permutation case (100 jobs, 20 stations, with setup considerations) 180

7.26 Watson: Comparison with best known solutions for permutation sequences (200 jobs, 20 stations) 181

7.27 Watson: Comparison with permutation case (200 jobs, 20 stations, with setup considerations) 181

8.1 Applicability of CLP versus GA. 194

A.1 Peliminary analysis. 209

A.2 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (20 jobs, 20 stations) 228

A.3 Watson: Comparison of the average of eight runs with the permutation case (20 jobs, 20 stations, with setup considerations) 228

A.4 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (50 jobs, 20 stations) 230

A.5 Watson: Comparison of the average of eight runs with the permutation case (50 jobs, 20 stations, with setup considerations) 231

A.6 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (100 jobs, 20 stations) 232

A.7 Watson: Comparison of the average of eight runs with the permutation case (100 jobs, 20 stations, with setup considerations) 233

A.8 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (200 jobs, 20 stations) 234

A.9 Watson: Comparison of the average of eight runs with the permutation case (200 jobs, 20 stations, with setup considerations) 234

List of Tables

1.1 Numerical example of processing time of two jobs and four stations 2

2.1 Resequencing methods I 34

2.2 Resequencing methods II 35

2.3 Resequencing methods III 36

2.4 Resequencing methods IV 37

4.1 Calculation of start time difference for two consecutive jobs j and k 79

5.1 Basic Arrangement: Processing time 96

5.2 Peliminary analysis. 97

5.3 Mnemonic abbreviations and the relevant parameters for the execution time. 105

5.4 Different parameter sets for the adjustment of arrangements A_{MSS}, A_{MJP} and A_{MSJ}. 106

5.5 Different parameter sets for the adjustment of arrangement A_M. 107

5.6 Final set of parameters for the different arrangements. 107

5.7 Comparison of arrangements for the intermediate case. 107

5.8 Comparison of arrangements for the centralized case. 108

5.9 Limited flexibility for the intermediate case. 109

5.10 Limited flexibility for the centralized case. 109

5.11 Intermediate versus centralized buffer location. 111

5.12 Detailed information for the intermediate buffer location. 111

5.13 Detailed information for the centralized buffer location. 111

5.14 Difference in physical size of jobs: A_{NP} 112

5.15 Difference in physical size of jobs: A_C 113

5.16 Difference in physical size of jobs: A_{MSS} 113

5.17 Difference in physical size of jobs: Comparison 114

5.18 Difference in physical size of jobs: Intermediate versus centralized case . . . 114

5.19 Semi dynamic demand. 114

6.1 Preliminary analysis: constant parameters 143
6.2 Preliminary analysis: Range of variable parameters $p_{\text{c-I}}$ and $p_{\text{c-II}}$ 144
6.3 Preliminary analysis: Range of variable parameters $p_{\text{m-I(f)}}$ and $p_{\text{m-I(b)}}$ . . 145
6.4 Preliminary analysis: Range of variable parameters, $p_{\text{m-II}}$ and $p_{\text{m-I(f)}}$ . . . 146
6.5 Rough parameter adjustment of the Genetic Algorithm 149
6.6 Variable parameters of the flowshop (Permutation case). 150
6.7 Variable parameters of the flowshop (Non-permutation case). 150
6.8 Adjustment of variable parameters. 151

7.1 Four scenarios for the setup consideration. 161
7.2 Allocation of the buffer places to the buffers. 166
7.3 Range of the processing times. 168
7.4 Stations with access to resequencing buffer for benchmark comparison. . . . 182
7.5 Allocation of the buffer places to buffers. 182
7.6 Brucker: Comparison for permutation sequences (20 jobs, 5 stations) 183
7.7 Brucker: Comparison for non-permutation sequences (20 jobs, 5 stations) . 183
7.8 Brucker: Comparison for permutation sequences (20 jobs, 10 stations) . . . 184
7.9 Brucker: Comparison for non-permutation sequences (20 jobs, 10 stations) . 185
7.10 Brucker: Comparison for permutation sequences (20 jobs, 20 stations) . . . 185
7.11 Brucker: Comparison for non-permutation sequences (20 jobs, 20 stations) . 185
7.12 Brucker: Comparison for permutation sequences (50 jobs, 5 stations) 186
7.13 Brucker: Comparison for non-permutation sequences (50 jobs, 5 stations) . 187
7.14 Brucker: Comparison for permutation sequences (50 jobs, 10 stations) . . . 187
7.15 Brucker: Comparison for non-permutation sequences (50 jobs, 10 stations) . 188
7.16 Brucker: Comparison of computational time. 188

8.1 Difference in physical size of jobs: Genetic Algorithm 192
8.2 Difference in physical size of jobs: Intermediate case. 192
8.3 Difference in physical size of jobs: Centralized case. 192
8.4 Semi dynamic demand: CLP. 193
8.5 Semi dynamic demand: Genetic Algorithm. 193
8.6 Semi dynamic demand: GA versus CLP. 194

A.1 Basic Arrangement: Setup cost 208
A.2 Basic Arrangement: Setup time 208

A.3 Peliminary analysis. 209
A.4 Parameter analysis (A_{MSS}, A_{MJP} and A_{MSJ}) 210
A.5 Parameter analysis (A_M). 210
A.6 Comparison of the proposed arrangements for the intermediate case. 211
A.7 Comparison of the proposed arrangements for the centralized case. 212
A.8 Limited flexibility for the intermediate case. 213
A.9 Limited flexibility for the centralized case. 214
A.10 Details on the preliminary analysis: Crossover-I versus Crossover-II 215
A.11 Details on the preliminary analysis: Mutation-I(f) versus Mutation-I(b) . . 215
A.12 Details on the preliminary analysis: Mutation-I(f) versus Mutation-II . . . 215
A.13 Number of stations. 216
A.14 Number of resequencing possibilities. 216
A.15 Number of jobs. 216
A.16 Number of parents. 217
A.17 Number of generations. 217
A.18 Number of types of models. 217
A.19 Variability of solutions: Permutation case. 218
A.20 Variability of solutions: Non-permutation case. 218
A.21 Variability of solutions: Number of jobchanges for non-permutation case. . 218
A.22 Setup considerations for the intermediate case. 219
A.23 Setup considerations for the centralized case. 220
A.24 Intermediate versus centralized buffer location. 221
A.25 Number of buffer places. 221
A.26 Difference in physical size of jobs. 222
A.27 Distribution of data. 223
A.28 Spreading. 223
A.29 Normal distribution. 224
A.30 Semi dynamic demand. 225
A.31 Semi dynamic demand. 226
A.32 Watson: Comparison of solutions with the best known solution for permutation sequences (20 jobs, 20 stations) 227
A.33 Watson: Comparison of solutions with the permutation case, with setup considerations (20 jobs and 20 stations) 227
A.34 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (20 jobs, 20 stations) 228
A.35 Watson: Comparison of the average of eight runs with the permutation case, with setup considerations (20 jobs and 20 stations) 229

A.36 Watson: Comparison of solutions with the best known solution for permutation sequences (50 jobs, 20 stations) 230

A.37 Watson: Comparison of solutions with the permutation case, with setup considerations (50 jobs and 20 stations) 230

A.38 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (50 jobs, 20 stations) 231

A.39 Watson: Comparison of the average of eight runs with the permutation case, with setup considerations (50 jobs and 20 stations) 231

A.40 Watson: Comparison of solutions with the best known solution for permutation sequences (100 jobs, 20 stations) 232

A.41 Watson: Comparison of solutions with the permutation case, with setup considerations (100 jobs and 20 stations) 232

A.42 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (100 jobs, 20 stations) 233

A.43 Watson: Comparison of the average of eight runs with the permutation case, with setup considerations (100 jobs and 20 stations) 233

A.44 Watson: Comparison of solutions with the best known solution for permutation sequences (200 jobs, 20 stations) 234

A.45 Watson: Comparison of solutions with the permutation case, with setup considerations (200 jobs and 20 stations) 234

A.46 Watson: Comparison of the average of eight runs with the best known solution for permutation sequences (200 jobs, 20 stations) 235

A.47 Watson: Comparison of the average of eight runs with the permutation case, with setup considerations (200 jobs and 20 stations) 235

A.48 Comparison for data by Brucker, Permutation case, ($N = 20$, $M = 5$) . . . 235

A.49 Comparison for data by Brucker, Non-Permutation case, ($N = 20$, $M = 5$) . 235

A.50 Comparison for data by Brucker, Permutation case, ($N = 20$, $M = 10$) . . . 236

A.51 Comparison for data by Brucker, Non-Permutation case, ($N = 20$, $M = 10$) 236

A.52 Comparison for data by Brucker, Permutation case, ($N = 20$, $M = 20$) . . . 236

A.53 Comparison for data by Brucker, Non-Permutation case, ($N = 20$, $M = 20$) 236

A.54 Comparison for data by Brucker, Permutation case, ($N = 50$, $M = 5$) . . . 237

A.55 Comparison for data by Brucker, Non-Permutation case, ($N = 50$, $M = 5$) . 237

A.56 Comparison for data by Brucker, Permutation case, ($N = 50$, $M = 10$) . . . 237

A.57 Comparison for data by Brucker, Non-Permutation case, ($N = 50$, $M = 10$) 237

References

Aldowaisan, T. and Allahverdi, A.(1998). Total flowtime in no-wait flowshops with separated setup times. *Computers & Operations Research*, 25(9):757–765.

Allahverdi, A.(2003). The two- and m-machine flowshop scheduling problems with bicriteria of makespan and mean flowtime. *European Journal of Operational Research*, 147(2):373–396.

Allahverdi, A., Gupta, J., and Aldowaisan, T.(1999). A review of scheduling research involving setup considerations. *Omega*, 27(2):219–239.

Azizoglu, M., Cakmak, E., and Kondakci, S.(2001). A flexible flowshop problem with total flow time minimization. *European Journal of Operational Research*, 132(3):528–538.

Baker, K.(1974). *Introduction to Sequencing and Scheduling*. John Wiley & Sons, Inc., New York.

Baker, K. and Scudder, G.(1990). Sequencing with earliness and tardiness penalties: A review. *Operations Research*, 38:22–36.

Balasko, B., Abonyi, J., and Feil, B.(2005). Fuzzy clustering and data analysis toolbox; for use with matlab.

Bard, J., Dar-El, E., and Shtub, A.(1992). An analytic framework for sequencing mixed model assembly lines. *International Journal of Production Research*, 30(1):35–48.

Bautista, J., Companys, R., and Corominas, A.(1996). Heuristics and exact algorithms for solving the monden problem. *European Journal of Operational Research*, 88(1):101–113.

Beaty, S.(1992). Genetic algorithms for instruction sequencing and scheduling. In *Workshop on Computer Architecture Technology and Formalism for Computer Science Research and Applications*. Naples, Italy.

Becker, C. and Scholl, A.(2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168:694–715.

Bellman, R.(1957). *Dynamic Programming*. Princeton University Press.

Bellman, R., Esogbue, A., and Nabeshima, I.(1982). Mathematical aspects of scheduling and applications. *International Series in Modern Applied Mathematics and Computer Science*, 33(5):769–772. Pergamon Press, Volume 4.

Blackstone, J., Phillips, D., and Hogg, G.(1982). A state-of-the-art survey of dispatching rules for manufacturing job shop operations. *International Journal of Production Research*, 20(1):27–45.

Bolat, A.(1994). Sequencing jobs on an automobile assembly line: objectives and procedures. *International Journal of Production Research*, 32(5):1219–1236.

Bratley, P., Fox, B., and Schrage, L.(1983). *A guide to simulation*. Springer-Verlag, New York.

Brucker, P.(1998). *Scheduling Algorithms*. Springer-Verlag.

Brucker, P., Heitmann, S., and Hurink, J.(2003). Flow-shop problems with intermediate buffers. *OR Spectrum*, 25:549–574.

Bukchin, J.(1998). A comparative study of performance measures for throuput of a mixed model assembly line in a jit environment. *International Journal of Production Research*, 36(10):2669–2685.

Burns, L. and Daganzo, C.(1987). Assembly line job sequencing principles. *International Journal Production Research*, 25(1):71–99.

Campbell, H., Dudek, R., and Smith, M.(1970). A heuristic algorithm for the n-job,m-machine sequencing problem. *Management Science*, 20(10):630–637.

Carlier, J. and Rebai, I.(1996). Two branch and bound algorithms for the permutation flowshop problem. *European Journal of Operational Research*, 90(2):238–251.

Cepek, O., Okada, M., and Vlach, M.(2002). Nonpreemptive flowshop scheduling with machine dominance. *European Journal of Operational Research*, 139(2):245–261.

Choi, W. and Shin, H.(1997). A real-time sequence control system for the level production of the automobile assembly line. *Computers and Industrial Engineering*, 33(3-4):769–772.

Companys, R.(1992). Algoritmo lomnicki pendular. *Technical Report, Universitat Politècnica de Catalunya, Spain.*

Dannenbring, D.(1977). An evaluation of flow shop sequencing heuristics. *Management Science*, 23(11):1174–1182.

Dantzig, G.(1948). *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ.

Dar-El, E. and Cucuy, S.(1977). Optimal mixed-model sequencing for balanced assembly lines. *Omega*, 5(3):333–342.

Díaz, A., Tchernykh, A., and Ecker, K.(2003). Algorithms for dynamic scheduling of unit execution time tasks. *European Journal of Operational Research*, 146(2):403–416.

Ding, F. and Sun, H.(2004). Sequence alteration and restoration related to sequenced parts delivery on an automobile mixed-model assembly line with multiple departments. *International Journal of Production Research*, 42(8):1525–1543.

Dorigo, M., Maniezzo, V., and A., C.(1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B*, 26(1):29–41.

Dudek, R. A., Panwalkar, S. S., and Smith, M. L.(1992). The lessons of flowshop scheduling research. *Operations Research*, 40:7–13.

Dutta, S. and Cunningham, A.(1975). Sequencing two-machine flow-shops with finite intermediate storage. *Management Science*, 21:989–996.

Engel, C., Zimmermann, J., and Steinhoff, A.(1997). *Objectives for Order-Sequencing in Automobile Production: Drexl, A. and Kimms, A. 1998, Beyond Manufacturing Resource Planning (MRP II) - Advanced Models and Methods for Production Planning.* Springer, Berlin. 307-331.

Engstrőm, T., Jonsson, D., and Johansson, B.(1996). Alternatives to line assembly: Some swedish examples. *International Journal of Industrial Ergonomics*, 17(3):235–245.

Erel, E. and Sarin, S.(1998). A survey of the assembly line balancing procedures. *Production Planning and Control*, 9(5):414–434.

Festa, P. and Resede, M.(2004). An annotated bibliography of grasp. Technical Report TD-5WYSEW, AT and T Labs Research.

Framinan, J., Gupta, J., and Leisten, R.(2002). A review and classification of heuristics for permuation flowshop scheduling with makespan objective. *Technical Report OI/PPC-2001/02.* Version 1.2.

Framinan, J. and Leisten, R.(2003). Comparison of heuristics for flowtime minimisation in permuation flowshops. *Technical Report IO-2003/01.* Version 0.5.

French, S.(1982). *Sequencing and Scheduling.* John Wiley, New York.

Garey, M. and Johnson, D.(1979). *The complexity and Intractability-A guide to the theory of NP-Completeness.* W.H.Freeman and Company, NY.

Garey, M., Johnson, D., and Sethi, R.(1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1:117–29.

Gendreau, M., Laporte, G., and Guimaraes, E.(2001). A divide and merge heuristic for the multiprocessor scheduling problem with sequence dependent setup-times. *European Journal of Operational Research*, 133(1):183–189.

Ghosh, S. and Gagnon, R.(1989). A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *International Journal of Production Research*, 27(4):637–670.

Glover, F.(1990). Tabu search: A tutorial. *Interfaces*, 20(4):74–94.

Gupta, J. and Reddi, S.(1978). Improved dominance conditions for the three-machine flowshop scheduling problem. *Operations Research*, 26:200–203.

Ha, J., Chang, H., Lee, E., Lee, I., Lee, B., and Yi, G.(2000). Intermediate storage tank operation strategies in the production scheduling of multi-product batch processes. *Computers and Chemical Engineering*, 24:1633–1640.

Herrmann, J., Chung-Yee, L., and Snowdon, J.(1993). A classification of static scheduling problems. *Complexity in Numerical Optimization*, pages 203–253. P.M. Pardalos, Editor, World Scientific Publishing Co., Singapore.

Holland, J.(1973). Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput.*, 2(2):88–105.

Holland, J.(1975). Adaptation in natural and artificial systems. *University of Michigan Press, Ann Arbor*, page s.

Holthaus, O. and Rajendran, C.(2002). A study on the performance of scheduling rules in buffer-constrained dynamic flowshops. *International Journal of Production Research*, 40(13):3041–3052.

Hyun, C., Kim, Y., and Kim, Y.(1998). A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines. *Computers & Operations Research*, 25(7/8):675–690.

Ignall, E. and Schrage, L.(1965). Application of the branch and bound technique to some flow-shop problems. *Operations Research*, 13(3):400–412.

Inman, R.(2003). Asrs sizing for recreating automotive assembly sequences. *International Journal of Production Research*, 41(5):847–863.

Ishibuchi, H., Yoshida, T., and Murata, T.(2003). Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223.

Jain, V. and Grossmann, I.(2000). A disjunctive model for schedulin in a manufacturing and packing facility with intermediate storage. *Optimization and Engineering*, 1:215–231.

Johnson, S.(1954). Optimal two and three stage production schedules with setup times included. *Naval Research Logistics Quaterly*, 1(1):61–68.

Kim, Y., Hyun, C., and Kim, Y.(1996). Sequencing in mixed model assembly lines: A genetic algorithm approach. *Computers & Operations Research*, 23(12):1131–1145.

Kim, Y. and Kim, J.(2000). A coevolutionary algorithm for balancing and sequencing in mixed model assembly lines. *Applied Intelligence*, 13(3):247–258.

Kirkpatrick, S., Gelatt Jr., C., and Vecchi, M.(1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Klundert, J. and Grigoriev, A.(2001). Throuput rate optimization in high multiplicity sequencing problems. In *Conference proceedings of the 17 International Symposium on Mathematical Programming*.

Korkmaz, O., Wood, D., and Gunal, A. Throughput analysis with respect to buffer and batch sizes using simulation. *http://www.pmcorp.com/pub_simulation.shtm*. consulted on 26.08.2004.

Korkmazel, T. and Meral, S.(2001). Bicriteria sequencing methods for the mixed-model assembly line in just-in-time production systems. *European Journal of Operational Research*, 131(1):188–207.

Koshla, I.(1995). The scheduling problem where multiple machines compete for a common local buffer. *European Journal of Operational Research*, 84:330–342.

Lageweg, B., Lenstra, J., and Rinnooy, K.(1978). A general bounding scheme for the permutation flowshop problem. *Operations Research*, 26:53–67.

Lahmar, M. and Benjaafar, S.(2003). Sequencing with limited flexibility. *Manufacturing and Service Operations Management*. in review.

Lahmar, M., Ergan, H., and Benjaafar, S.(2003). Resequencing and feature assignment on an automated assembly line. *IEEE Transactions on Robotics and Automation*, 19(1):89–102.

Lee, H. and Schaefer, S.(1997). Sequencing methods for automated storage and retrieval systems with dedicated storage. *Computers and Industrial Engineering*, 32(2):351–362.

Lee, I. and Shaw, M.(2000). A neural-net approach to real time flow-shop sequencing. *Computers and Industrial Engineering*, 38(1):125–147.

Leisten, R.(1990). Flowshop sequencing problems with limited buffer storage. *Internation Journal of Production Research*, 28(11):2085–2100.

Leu, Y., Matheson, L., and Rees, L.(1996). Sequencing mixed-model assemly lines with genetic algorithms. *Computers and Industrial Engineering*, 30(4):1027–1036.

Li, S.(1997). A hybrid two-stage flowshop with part family, batch production, major and minor set-ups. *European Journal of Operational Research*, 102(1):142–156.

Liao, C., Liao, L., and Tseng, C.(2006). A performance evaluation of permutation vs. non-permutation schedules in a flowshop. *International Journal of Production Research*, 44(20):4297–4309.

Liesegang, G. and Schirmer, A.(1975). Heuristische verfahren zur maschinenbelegungsplanung bei reihenfertigung. *Zeitschrift für Operations Research*, 19:195–211.

Little, J., Murty, K., Sweeney, D., and Karel, C.(1963). An algorithm for the traveling salesman problem. *Operations Research*, 11:972–989.

Lomnicki, Z.(1965). A branch and bound algorithm for the exact solution of the three machine scheduling problem. *Operations Research Quaterly*, 16:89–100.

Luenberger, D.(1984). *Linear and Nonlinear Programming*. Addison-Wesley.

Michaelewicz, Z.(1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 3rd edition.

Monden, Y.(1983). *Toyota Production System*. Institute of Industrial Engineering Press, Norcross, GA.

Morton, T. and Pentico, D.(1993). *Heuristic scheduling systems: with applications to production systems and project management*. Wiley, New York.

Nawaz, M., Enscore, E., and Ham, I.(1983). A heuristic algorithm for the m-machine, n-job flow shop sequencing problem. *Omega*, 11(1):91–95.

Nicholson, T.(1971). *Optimization in industry, optimization techniques*, volume 1. Longman Group Limited.

Niu, H.(2005). Sequencing and balancing problem of mixed-model-assembly-line with window cycle time. *Doctoral Thesis*. Universitat Politécnica de Catalunya, Barcelona, Spain.

Papadimitriou, C. and Kanellakis, P.(1980). Flowshop scheduling with limited temporary storage. *Journal of the Association for Computing Machinery*, 27:533–554.

Park, Y., Pegden, C., and Enscore, E.(1984). A survey and evaluation of static flowshop scheduling heuristics. *International Journal of Production Research*, 22:127–141.

Pastor, R.(1999). *Metalgoritmo de optimización combinatoria mediante la exploración de grafos*. PhD thesis, Universitat Politécnica de Catalunya.

Pastor, R. and Corominas, A.(2004). Branch and win: Or tree search algorithms for solving combinatorial optimisation problems. *Sociedad de Estadística e Investigación Operativa*, 12(1):169–191.

Péridy, L., Pinson, E., and Rivreau, D.(1999). Elimination rules for the flow-shop and permuatation flow-shop problems. *http://www.math-appli-uco.fr*.

Pinedo, M.(1995). *Scheduling. Theory, Algorithms and Systems*. Prentice Hall Inc., Englewood Cliffs, New Jersey, 1 edition.

Plans, J.(1999). Classificació, modelització i resolució dels problemes de disseny i assignació de tasques en línies de producció. *Tesis Doctoral*. Universitat Politécnica de Catalunya, Barcelona, Spain.

Plans, J. and Corominas, A.(2000). A classification scheme for assembly line problems. *Technical Document IOC-DF-P-2000-20*. Universitat Politécnica de Catalunya, Barcelona, Spain.

Potts, C.(1980). An adaptive branching rule for the permutation flowshop problem. *European Journal of Operational Research*, 5(2):19–25.

Potts, C., Shmoys, D., and Williamson, D.(1991). Permutation vs. non-permutation flow shop schedules. *Operations Research Letters*, 10(5):281–284.

Pugazhendhi, S. and Thiagarajan, S.(2004). Generating non-permutation schedules in flowline-based manufacturing systems with sequence-dependent setup times of jobs: a heuristic approach. *The International Journal of Advanced Manufacturing Technology*, 23(1/2):64–78.

Pugazhendhi, S., Thiagarajan, S., Rajendran, C., and Anantharaman, N.(2002). Performance enhancement by using non-permutation schedules in flowline-based manufacturing systems. *Computers and Industrial Engineering*, 44(1):133–157.

Rachakonda, P. and Nagane, S.(2000). Simulation study of paint batching problem in automobile industry. *http://sweb.uky.edy/˜pkrach0/Projects/MFS605Project.pdf*. consulted 14.07.2004.

Raghavachari, M.(1988). Scheduling problems with non-regular penalty functions: A review. *Opsearch*, 25:144–164.

Rajendran, C. and Chaudhuri, D.(1992). An efficient heuristic approach to the scheduling of jobs in a flowshop. *European Journal of Operational Research*, 61(3):318–325.

Rajendran, C. and Ziegler, H.(2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, 155:426–438.

Reddi, S.(1976). Sequencing with finite intermediate storage. *Management Science*, 23:19–25.

Reeves, C.(1995). A genetic algorithm for flowshop sequencing. *Computers & Operations Research*, 22(1):5–13.

Riane, F., Artiba, A., and Elmaghraby, S.(1998). A hybrid three-stage flowshop problem: Efficient heuristics to minimize makespan. *European Journal of Operational Research*, 109(2):321–329.

Ríos-Mercado, R. and Bard, J.(1999). A branch-and-bound algorithm for permutation flow shops with sequence-dependent setup times. *IIE Transaction*, 31:721–731.

Roy, B.(1962). Cheminement et connexité dans les graphes-applications aux problémes d'ordonnancement. *Revue METRA, série spéciale*, (1). 130 pages.

Rubin, P. and Ragatz, G.(1995). Scheduling in a sequence dependent setup environment with genetic search. *Computers & Operations Research*, 22(1):85–99.

Ruiz, R., Maroto, C., and Alcaraz, J.(2006). Two new robust genetic algorithms for the flowshop scheduling problem. *Omega*, 34:461–476.

Sarker, B. and Pan, H.(1998). Designing a mixed-model assembly line to minimize the costs of idle and utility times. *Computers and Industrial Engineering*, 34(3):609–628.

Sarker, B. and Pan, H.(2001). Design configuration for a closed-station, mixed-model assembly line: a filing cabinet manufacturing system. *International Journal of Production Research*, 39(10):2251–2270.

Sawik, T.(2000). Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers. *Mathematical and Computer Modelling*, 31(13):39–52.

Scholl, A.(1995). *Balancing and Sequencing of Assembly lines*. Heidelberg: Physica-Verlag.

Scholl, A. and Becker, C.(2006). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168:666–693.

Sevaux, M. and Dauzère-Péres, S.(2003). Genetic algorithms to minimize the weighted number of late jobs on a single machine. *Computers and Industrial Engineering*, 151:296–306.

Shakhlevich, N., Sotskov, Y., and Werner, F.(2000). Complexity of mixed shop scheduling problems: A survey. *European Journal of Operational Research*, 120:343–351.

Silver, E.(2004). An overview of heuristic solution methods. *Journal of the operational research society*, 55(9):936–956.

Smed, J., Johnsson, M., Johtela, T., and Nevalainen, O.(1999). Techniques and applications of production planning in electronics manufacturing systems. *Technical Report No. 320*. Turku Centre for Computer Science.

Stafford, E. and Tseng, F.(2002). Two models for a family of flowshop sequencing problems. *European Journal of Operational Research*, 142(2):282–293.

Sun, X., Morizawa, K., and Nagasawa, H.(2003). Powerful heuristics to minimize makespan in fixed, 3-machine, assembly-type flowshop scheduling. *European Journal of Operational Research*, 146(3):498–516.

Szwarc, W.(1971). Elimination methods in the *mxn* sequencing problem. *Naval Research Logistics Quaterly*, 18:295–305.

Taillard, E.(1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64(2):278–285.

Vieira, G., Herrmann, J., and Lin, E.(2003). Rescheduling manufacturing systems: A framework of strategies, policies and methods. *Journal of Scheduling*, 6(1):39–62.

Watson, J., Barbulescu, L., Whitley, L., and Howe, A.(2002). Contrasting structured and random permutation flow-shop scheduling problems: Search-space topology and algorithm performance. *INFORMS Journal on Computing*, 14(2):98–123.

Wilhelm, W. and Shin, H.(1985). Effectiveness of alternative operations in a flexible manufacturing system. *International Journal of Production Research*, 23(1):65–79.

Witt, A. and Voss, S.(2005). Simple heuristics for scheduling with limited intermediate storage. *Computers & Operations Research*. Arcticle in Press.

Index of first Authors

A

Aldowaisan, T. . . . . , 19

Allahverdi, A. . . . . , 17, 18, 20, 22, 24

Azizoglu, M. . . . . , 20

B

Baker, K. . . . . , 16, 20, 21

Balasko, B. . . . . , 151

Bard, J. . . . . , 17, 39, 42

Bautista, J. . . . . , 16, 47

Beaty, S. . . . . , 63

Becker, C. . . . . , 1, 18, 33, 55, 70

Bellman, R. . . . . , 46, 47

Blackstone, J. . . . . , 48

Bolat, A. . . . . , 23, 25

Bratley, P. . . . . , 51

Brucker, P. , 29, 36, 45, 51, 159, 182, 184, 188, 189

Bukchin, J. . . . . , 16

Burns, L. . . . . , 9, 23, 24, 38

Scholl, A. . . . . , 1

C

Campbell, H. . . . . , 21

Carlier, J. . . . . , 22, 32

Cepek, O. . . . . , 19

Choi, W. . . . . , 27, 32, 34

Companys, R. . . . . , 22

D

Díaz, A. . . . . , 38

Dannenberg, D. . . . . , 50

Dantzig, G. . . . . , 46

Dar-El, E. . . . . , 13, 41

Ding, F. . . . . , 28, 34

Dorigo, M. . . . . , 49

Dudek, R.A. . . . . , 45

Dutta, S.K. . . . . , 28, 35

Ishibuchi, H. . . . . , 17

E

Engel, C. . . . . , 38

Engstrőm, T. . . . . , 31, 37

Erel, E. . . . . , 1

F

Färber, G. . . . . , 200, 201

Festa, P. . . . . , 50

Framinan, J. . . . . , 22, 51

French, S. . . . . , 14, 15, 46

G

Garey, M. . . . . , 21, 45

Gendreau, M. . . . . , 20, 31, 37

Ghosh, S. . . . . , 1

Glover, F. . . . . , 49

Gupta, J. . . . . , 21

H

Ha, J.K. . . . . , 30, 36

Herrmann, J. . . . . , 18, 33, 55, 70

Holland, J. . . . . , 118

Holthaus, O. . . . . , 29, 35

Hyun, C. , 39, 49, 117

I

Ignall, E. . . . . , 22, 46
Inman, R. , 28, 32, 34

J

Jain, V. . . . . , 29, 36
Johnson, S. . . . . , 19, 21, 51, 55

K

Kim, Y. , 2, 17, 25, 49
Kirkpatrick, S. . . . . , 48
Klundert, J. . . . . , 10, 39
Korkmaz, O. . . . . , 32, 34
Korkmazel, T. . . . . , 10
Koshla, I. . . . . , 29

L

Lageweg, B. . . . . , 18, 22, 33, 55, 70
Lahmar, M. . . . , 3, 6, 29, 35, 53, 175, 197
Lee, H. . . . . , 27, 34, 47
Leisten, R.L. . . . . , 28, 35
Leu, Y. . . . . , 117
Li, S. . . . . , 20, 31, 37
Liao, C.J. . . . . , 3, 51, 197
Liesegang, G. . . . . , 27
Little, J. . . . . , 46
Lomnicki, Z. . . . . , 22, 46
Luenberger, D. . . . . , 46

M

Michaelewicz, Z. , 117
Monden, Y. . . . . , 47
Morton, T. . . . . , 49

N

Nawaz, M. . . . . , 22
Nicholson, T. . . . . , 45–47
Niu, H. . . . . , 18, 33, 55, 70, 117, 200

P

Péridy, L. . . . . , 21, 32, 34
Papadimitriou, C.H. . . . . , 28, 35
Park, Y.B.. . . . . , 27
Pastor, R. . . . . , 47
Pinedo, M. . . , 5, 8, 18, 21–23, 25, 28, 31, 33, 37, 45, 48, 55, 70, 197
Plans, J. . . . . , 18, 33, 55, 70
Potts, C. . . . . , 3, 21, 22, 32, 51, 197
Pugazhendhi, S. . . . . , 32, 34

R

Ríos-Mercado, R. . . . . , 22, 42
Rachakonda, P. . . . . , 31, 37
Raghavachari, M. , 16
Rajendran, C. . . . . , 49, 51
Reddi, S.S. . . . . , 28, 35
Reeves, C. . . . . , 117
Riane, F. . . . . , 20, 21, 30, 37
Roy, B. . . . . , 22, 27, 34
Rubin, P.A. . . . . , 117
Ruiz, R. . . . . , 117

S

Sarker, B. . . . . , 16, 39, 41
Sawik, T. . . . . , 30, 31, 37
Scholl, A. . . . . , 1
Sevaux, M. . . . . , 117
Shakhlevich, N. . . . . , 45
Silver, E. . . . . , 48, 50
Smed, J. . . . . , 39
Stafford, E. . . . . , 41
Sun, X. . . . . , 20, 31, 37
Szwarc, W. . . . . , 22

T

Taillard, E. . . . . , 51, 176

V

Vieira, G. . . . . , 18, 33, 39, 55, 70

W

Watson, J. . . . . , 51, 159, 176, 177, 189
Wilhelm, W.E. . . . . , 28, 35
Witt, A. . . . . , 29, 36