



Universitat de Lleida

Cloud systems and HPC as a service of agroindustry

Jordi Mateo Fornés

<http://hdl.handle.net/10803/667343>

ADVERTIMENT. L'accés als continguts d'aquesta tesi doctoral i la seva utilització ha de respectar els drets de la persona autora. Pot ser utilitzada per a consulta o estudi personal, així com en activitats o materials d'investigació i docència en els termes establerts a l'art. 32 del Text Refós de la Llei de Propietat Intel·lectual (RDL 1/1996). Per altres utilitzacions es requereix l'autorització prèvia i expressa de la persona autora. En qualsevol cas, en la utilització dels seus continguts caldrà indicar de forma clara el nom i cognoms de la persona autora i el títol de la tesi doctoral. No s'autoritza la seva reproducció o altres formes d'explotació efectuades amb finalitats de lucre ni la seva comunicació pública des d'un lloc aliè al servei TDX. Tampoc s'autoritza la presentació del seu contingut en una finestra o marc aliè a TDX (framing). Aquesta reserva de drets afecta tant als continguts de la tesi com als seus resums i índexs.

ADVERTENCIA. El acceso a los contenidos de esta tesis doctoral y su utilización debe respetar los derechos de la persona autora. Puede ser utilizada para consulta o estudio personal, así como en actividades o materiales de investigación y docencia en los términos establecidos en el art. 32 del Texto Refundido de la Ley de Propiedad Intelectual (RDL 1/1996). Para otros usos se requiere la autorización previa y expresa de la persona autora. En cualquier caso, en la utilización de sus contenidos se deberá indicar de forma clara el nombre y apellidos de la persona autora y el título de la tesis doctoral. No se autoriza su reproducción u otras formas de explotación efectuadas con fines lucrativos ni su comunicación pública desde un sitio ajeno al servicio TDR. Tampoco se autoriza la presentación de su contenido en una ventana o marco ajeno a TDR (framing). Esta reserva de derechos afecta tanto al contenido de la tesis como a sus resúmenes e índices.

WARNING. Access to the contents of this doctoral thesis and its use must respect the rights of the author. It can be used for reference or private study, as well as research and learning activities or materials in the terms established by the 32nd article of the Spanish Consolidated Copyright Act (RDL 1/1996). Express and previous authorization of the author is required for any other uses. In any case, when using its content, full name of the author and title of the thesis must be clearly indicated. Reproduction or other forms of for profit use or public communication from outside TDX service is not allowed. Presentation of its content in a window or frame external to TDX (framing) is not authorized either. These rights affect both the content of the thesis and its abstracts and indexes.



Universitat de Lleida

TESI DOCTORAL

Cloud systems and HPC as a service of agroindustry

Jordi Mateo Fornés

Memòria presentada per optar al grau de Doctor per la Universitat de Lleida
Programa de Doctorat en Enginyeria i Tecnologies de la Informació.

Director/a
Francesc Solsona Tehàs
Lluís M. Plà Aragonès

2019

Begin at the beginning, the King
said gravely, and go on till you
come to the end: then stop

Alice in Wonderland

To you as a reader.

ABSTRACT

Background: Cloud computing is a new paradigm that opens a flexible and financial attractive door for business and consumers. In recent years, cloud services have reshaped the way business models are planned, saving costs and improving efficiency. There is an exponential growth of cloud solutions available in the public and private sector. This way, cloud services are based on a pay-per-use model, offering resources and service through the Internet. Moreover, the cloud architectures provide many advantages concerning scalability, maintainability and massive data processing. High Performance Computing (HPC) decomposition methods and Operation Research (OR) models integration inside the cloud brings the opportunity to improve the performance and the quality of the cloud services. There are infinite opportunities and challenges for exploiting the capabilities of Decision Support Systems (DSS) inside the cloud. The agribusiness sector can significantly be boost by the consideration of these possibilities.

Methods: Queuing theory and nonlinear programming were combined to model cloud architectures and to develop trade-off policies considering different metrics and ensuring a certain level of quality of service. OR decomposition methods (Benders Decomposition and Lagrangian Decomposition) are improved using HPC paradigms. Cloud services, OR methods and decision support systems are integrated to deliver knowledge from academia to society. Two-stage sStochastic programming was selected to enhance the decision-making process in the agribusiness field. Finally, two different agribusiness applications were designed, deployed and tested on a real cloud platform.

Results: This thesis explores methods, mathematical models and algorithms for managing cloud systems cost-effectively. Furthermore, this thesis analyses how to export and deliver this knowledge to society; in particular to the agribusiness field. The primary purpose is to assist decision-makers in their daily activity. The main results of this thesis highlight the future guidelines to build smart decision support systems combining HPC, OR and cloud platforms. The outcomes corroborate the benefits of applying parallel computing to decomposition methods to deal with stochastic models. Moreover, the results show the advantages of mixing cloud platforms and optimisation field offering optimisation as a service to the overall community. Furthermore, the results demonstrate that considering uncertainty and stochastic programming costs can be reduced in

agribusiness supply chains. Besides, the results show that cloud DSS's improve the decision-making process, overcoming the current barriers. It presents a cloud application to assist pig farmers in the optimal replacement of sows.

Conclusions: Several cloud models and policies were analysed and implemented using different methods and techniques. The results yielded by this thesis provide strong evidence that parallel decomposition methods, such as Lagrangian Decomposition and Benders Decomposition can beat commercial solvers such as CPLEX when dealing with the optimal resolution of real-life stochastic instances. SPOS shows how to orchestrate cloud platforms to deliver knowledge as a service while integrating mathematical solvers in a cloud environment. In a more agroindustrial context, the stochastic model designed around the Chilean dried apple supply chain proof a cost reduction by considering uncertain up to a 6.4%. Finally, a cloud DSS was successfully developed and implemented to assist sow farmers. All the results extracted from this thesis are an essential seed for modelling a much bigger service, with great potential to become a reference in the design of cloud DSS offering all users the possibility of solving any problem, even stochastic models, and also of obtaining detailed and custom results.

RESUM

Introducció: La computació al núvol és un nou paradigma que obra un univers de possibilitats en el món dels negocis i als consumidors de serveis. En els darrers anys, els serveis basats en núvol han reconfigurat la manera en què es planeja el model de negoci, estalviant costos i millorant l'eficiència. Actualment, hi ha un creixement exponencial de les solucions disponibles basades en el núvol al sector públic i privat. D'aquesta manera, els serveis basats en núvol es basen en un model de pagament per ús, que ofereix recursos i serveis a través d'Internet. A més a més, les arquitectures del sistema basades en núvol proporcionen molts avantatges en quant a l'escalabilitat de recursos, la sostenibilitat o el processament massiu de dades. Els mètodes de descomposició per a la resolució de problemes d'optimització, la computació d'altres prestacions per l'acceleració d'aquests mètodes i la seva corresponent integració dins del núvol ofereixen l'oportunitat de millorar el rendiment i la qualitat dels serveis. En el context actual, hi ha infinites oportunitats i reptes per explotar les capacitats dels sistemes de suport a la presa de decisions integrats en el núvol. Les empreses del sector agroalimentari poden beneficiar-se molt d'aquests sistemes/serveis per a ser més eficients i competitives.

Mètodes: La teoria de cues i la programació no lineal s'han utilitzat com a instrument per a modelar arquitectures basades en el núvol i desenvolupar polítiques balancejades tenint en compte diferents mètriques, com la disponibilitat o l'eficiència per garantir un cert nivell de qualitat de servei. La computació d'altres prestacions s'ha utilitzat per millorar el rendiment de les tècniques de descomposició (Benders i Lagrange). Els serveis cloud, els mètodes d'investigació operativa i els sistemes de suport a la presa de decisions s'han integrat per transferir el coneixement des de l'àmbit acadèmic a la societat. La programació estocàstica bietapa s'ha seleccionat per millor les decisions dels agronegocis. Finalment s'ha dissenyat, implementat i provat un servei d'optimització i dues aplicacions d'agronegocis diferents en una plataforma basada en el núvol.

Resultats: Aquesta tesi explora mètodes, models matemàtics i algorismes per a una gestió eficient dels sistemes en el núvol. També explora com exportar i lliurar aquest coneixement a la societat, en particular al camp dels agronegocis per millora el procés de presa de decisions. Els principals resultats d'aquesta tesi ressalten les pautes per construir sistemes intel·ligents de suport a la decisió integrats en plataformes en el núvol. Els resultats corroboren els beneficis de

l'aplicació de la computació d'altres prestacions juntament amb els mètodes de descomposició per tractar models estocàstics. A més a més, els resultats mostren els avantatges de barrejar plataformes en el núvol i els models de programació matemàtica per oferir optimització a la comunitat en general. A més a més, els resultats demostren que l'ús de la programació estocàstica permet reduir els costos de les cadenes de subministrament en l'àmbit dels agronegocis. Finalment, els resultats mostren que els DSS al núvol milloren els processos de decisió, superant les barreres actuals per adoptar aquestes tecnologies. Es presenta una aplicació al núvol per assistir a les granges de truges.

Conclusions: En aquesta tesi s'han analitzat i implementat diversos models, polítiques, arquitectures i serveis basats en el núvol. Els resultats obtinguts proporcionen evidències sòlides que els mètodes de descomposició paral·lels, com la descomposició Lagrangiana i la descomposició de Benders, poden superar solucions comercials com CPLEX quan es tracta de la resolució òptima d'instàncies estocàstiques. El servei SPOS mostra com organitzar plataformes en el núvol per oferir el coneixement com a servei integrant solucionadors matemàtics. En un context més agroindustrial, el model estocàstic dissenyat al voltant de la cadena de subministrament de poma xilena aconsegueix reduir els costos fins a un 6,4 %. Finalment, s'ha implementat un sistema de suport a la presa de decisions utilitzant les característiques del núvol existosament. Aquest treball mostra les millores en el reemplaçament de truges. Tots els resultats obtinguts en aquesta tesi són una llavor poderosa per modelar un servei molt més gran, amb un gran potencial per convertir-se en una referència en el disseny de DSS basat en núvol oferint a tots els usuaris la possibilitat de resoldre qualsevol tipus de problema, fins i tot models estocàstics i també d'obtenir resultats detallats i personalitzats.

RESUMEN

Introducción: La computación en la nube es un nuevo paradigma que abre un universo de posibilidades en el mundo de los negocios y los consumidores de servicios. En los últimos años, los servicios basados en la nube han reconfigurado la manera en que se planea los modelos de negocio, ahorrando costes y mejorando la eficiencia. Actualmente, hay un crecimiento exponencial de las soluciones disponibles basadas en la nube en el sector público y privado. De esta manera, los servicios basados en la nube se basan en un modelo de pago por uso, que ofrece recursos y servicios a través de Internet. Además, las arquitecturas del sistema basadas en nube proporcionan muchas ventajas en cuanto a la escalabilidad de recursos, la sostenibilidad o el procesamiento masivo de datos. Los métodos de descomposición para la resolución de problemas de optimización, la computación de altas prestaciones para la aceleración de estos métodos y su correspondiente integración dentro de la nube ofrecen la oportunidad de mejorar el rendimiento y la calidad de los servicios. En el contexto actual, hay infinitas oportunidades y retos para explotar las capacidades de los sistemas de apoyo a la toma de decisiones integradas en la nube. Las empresas del sector agroalimentario pueden beneficiarse mucho de estos sistemas/servicios para ser más eficientes y competitivas.

Métodos: La teoría de colas y la programación no lineal se han utilizado como instrumento para modelar arquitecturas basadas en la nube y desarrollar políticas equilibradas teniendo en cuenta diferentes métricas con la finalidad de garantizar un cierto nivel de calidad de servicio. La computación de altas prestaciones se ha utilizado para mejorar el rendimiento de los métodos de descomposición (Benders y Lagrange). Los servicios en la nube, los métodos de investigación operativa y los sistemas de ayuda a la toma de decisiones se han integrado para transferir el conocimiento académico a la sociedad. La programación estocástica de dos etapas se ha utilizado para mejorar las decisiones en los agronegocios. Finalmente, se ha diseñado, implementado y probado un servicio de optimización y dos aplicaciones de agronegocios diferentes en una plataforma basada en la nube.

Resultados: Esta tesis explora métodos, modelos matemáticos y algoritmos para una gestión eficiente de los sistemas en la nube. También explora cómo exportar y entregar este conocimiento a la sociedad, en particular en el campo de los agronegocios para mejorar el proceso de toma de decisiones. Los principales re-

sultados de esta tesis resaltan las pautas para construir sistemas inteligentes de apoyo a la decisión integrados en plataformas en la nube. Los resultados corroboran los beneficios de la aplicación de la computación de altas prestaciones junto con los métodos de descomposición para tratar modelos estocásticos. Además, los resultados muestran las ventajas de mezclar plataformas en la nube y los modelos de programación matemática para ofrecer optimización a la comunidad en general. Además, los resultados demuestran que el uso de la programación estocástica permite reducir los costes de las cadenas de suministro en el ámbito de los agronegocios. Finalmente, los resultados muestran que los DSS en la nube mejoran los procesos de decisión, superando las barreras actuales para adoptar estas tecnologías. Se presenta una aplicación en la nube para asistir a las granjas de cerdas.

Conclusiones: En esta tesis se han analizado e implementado varios modelos y políticas de arquitecturas y servicios basados en la nube. Los resultados obtenidos proporcionan evidencias de que los métodos de descomposición paralelos, como la descomposición Lagrangiana y la descomposición de Benders, pueden superar soluciones comerciales como CPLEX cuando se trata de la resolución óptima de instancias estocásticas. El servicio SPOS muestra cómo organizar plataformas en la nube para ofrecer la optimización como servicio integrando solucionadores matemáticos. En un contexto más agroindustrial, el modelo estocástico diseñado alrededor de la cadena de suministro de manzana seca chilena consigue reducir los costes hasta un 6,4 %. Finalmente, se ha implementado un sistema de apoyo a la toma de decisiones utilizando las características de la nube de forma exitosa. Este trabajo muestra las mejoras en el reemplazo de cerdas. Todos los resultados obtenidos en esta tesis son una semilla poderosa para modelar un servicio mucho mayor, con un gran potencial para convertirse en una referencia en el diseño de DSS basado en nube ofreciendo a todos los usuarios la posibilidad de resolver cualquier tipo de problema, incluso modelos estocásticos y también de obtener resultados detallados y personalizados.

ACKNOWLEDGEMENTS

The hardest arithmetic to master is that which enables us to count our blessings.

— Eric Hoffer, Reflections On The Human Condition

Doing this PhD has been an amazing experience and a great opportunity to grow as a person. It has been a long 4 years period full of difficulties and problems to overcome. A lot of times I thought If all this work was worth it. Nevertheless, my answer always was the same, If you love scabies, they don't hurt. I am very happy that I have had a chance to complete it. It would not have been possible without all those people who helped me along this journey throughout the unknown.

First of all, I would like to thank my supervisors, Dr. Francesc Solsona and Dr. Lluís M. Plà, whose constant attention and care pushed me unhurriedly and steadily throughout this adventure. Moreover, I would like to thank Dra. Adela Pagès for helping me with the mathematical concepts and for having always a clear idea of how to mitigate and avoid the issues.

I want to acknowledge Prof. Laureano Escudero and Prof. Victor Albornoz for keeping track of my advances and boost me into the right research direction, answering all the problems related to stochastic programming.

I want to acknowledge Prof. Asgeir Tomasgard for being an excellent host and supervisor during my doctoral stay at the NTNU, and for providing valuable comments and suggestions on improving my work.

I would like to thank all the past and current members of the Department of Computer Science and Industrial Engineering, at the University of Lleida. In particular, I thank Fernando Cores, Francesc Ginè, Fernando Guirado, Josep Lluís Lèrida and Concepció Roig. I also want to thank my past and current co-workers, Jordi Vilaplana, Eloi Gabaldón, Ivan Teixidó, Jordi Lladós, Ismael Arroyo, Marc Gonzalez ... for being such a wonderful bunch of people. I have had a great time with them. I also want to mention Montse Espunyes for being the best clerk a department can hope for.

Furthermore, I would like to thank Didac Florensa and Kevin Borell for being two incredible master students and helping me improving the cloud architectures and the decision support systems, coding the hard stuff.

I would also like to thank my friends and family for the support they have provided me with throughout my life, especially my family, my parents Àngel and Carme, my brother Adrià, and also my grandparents who exposed me to all sorts of opportunities as I grew up, and have always been incredibly encouraging of everything I wanted to do. Also Roman and Imma, for being a great father and mother-in-law. And in particular, I must acknowledge my girlfriend, couple and best friend, Sandra, without whose love, encouragement and assistance, I would not have finished never this thesis.

Lleida, Catalonia, June 6, 2019

A handwritten signature in black ink, appearing to read 'Jordi Mateo', written over a horizontal line.

Jordi Mateo

CONTENTS

1	INTRODUCTION AND SCOPE OF THE RESEARCH	1
1.1	Context	2
1.2	Cloud Based Decision Support Systems (DSS)	3
1.3	Cloud Computing	6
1.3.1	Cloud Platform	6
1.3.2	Cloud Research Lines	8
1.3.3	Cloud modelling	9
1.4	Cloud computing and Optimization Methods	10
1.4.1	Benders Decomposition	12
1.4.2	Lagrangian Decomposition	12
1.5	Two case studies in the agricultural sector	12
1.5.1	Pork supply chain	13
1.5.2	Dried Apple supply chain	14
1.6	Related Work and Contributions	15
1.6.1	Decision Support Systems and Agrobusiness	15
1.6.2	Cloud Computing	16
1.6.3	Decomposition Methods	18
1.6.4	Pork Supply Chain (PSC)	19
1.6.5	Fruit Supply Chain (FSC)	20
1.7	Research objectives	21
1.8	Outline	23
2	PAPER II: SLA MODEL TO KEEP QOS	25
3	PAPER III: PARALLEL LAGRANGIAN DECOMPOSITION	37
4	PAPER IV: PARALLEL CLUSTER BENDERS	55
5	PAPER IV: OPTIMIZATION AS A SERVICE	57
5.1	Introduction	59
5.2	The Cloud Service	62
5.2.1	Frontend	65
5.2.2	Backend	68
5.2.3	Computing Node	69
5.3	Results	69

5.3.1	MIPLIB Benchmark	70
5.3.2	Open Solvers versus Commercial Solvers	71
5.3.3	Comparison of resource usage	73
5.4	Conclusions and Future Work	75
6	PAPER VI: FRUIT SUPPLY CHAIN	77
6.1	Introduction	79
6.2	Supply chain for the apple dehydration process.	82
6.3	Two-stage stochastic model	85
6.4	Case Study	89
6.4.1	Scenario generation	91
6.4.2	Results and discussion	93
6.4.3	Intended use of the model	99
6.5	Conclusions	102
7	PAPER VII: PIG SUPPLY CHAIN	103
7.1	Introduction	105
7.2	Related Work	106
7.2.1	Sow replacement models	106
7.2.2	Review of Decision Support Systems	108
7.3	DSS organization	109
7.3.1	DSS workflow	112
7.4	Cloud Architecture	114
7.5	Case Study	117
7.5.1	Data integration	117
7.5.2	Data explorer	118
7.5.3	Optimal suggestions	119
7.5.4	Scenario Analysis	120
7.6	Conclusions and Future work	121
8	GLOBAL DISCUSSION OF RESULTS	125
9	GENERAL CONCLUSIONS AND FUTURE DIRECTIONS	129
9.1	Conclusions	129
9.2	Future directions	130
A	THREE MONTHS DOCTORAL STAY	133
A.1	Objectives	133
A.2	Tasks	133
A.3	Learned Skills	133
A.4	Summary	134

A.5	MultiHorizon formulation	135
B	BRIEF REVISION OF MATHEMATICAL PROGRAMMING	137
B.1	Linear Programming	137
B.2	Mixed Integer Programming	138
B.3	Stochastic Programming	139
C	LAGRANGIAN DECOMPOSITION	141
C.1	Subgradient method	142
C.2	Volume Algorithm	144
C.3	Progressive Hedging	145
C.4	Dynamic Constrained Cutting Plane method	148
D	PUBLICATIONS	151
D.1	Journal publications	151
D.2	Conference contributions	151
D.3	Other contributions	153
	NOMENCLATURE	155
	BIBLIOGRAPHY	159

INTRODUCTION AND SCOPE OF THE RESEARCH

I begin with an idea and then it becomes something else. Inspiration does exist, but it must find you working.

— Pablo Picasso

This thesis presents a collection of cloud computing techniques, platforms, and guidelines to tackle problems related to data science fields such as operations research and computer science. These problems go from seeking ways to deal with huge models to proposing services to deliver the knowledge obtained to the community.

Cloud computing opens new opportunities to design new decision support systems (DSS) to transfer qualified knowledge from academia to society, see [1]. Cloud-based services are ideal for scaling, growing and deal with uncertain demands. Furthermore, cloud computing eliminates the requirement for a powerful computer, overcoming traditional stand-alone applications with an environment dedicated to solving complex mathematical models. This way, with only a device connected to a remote decision engine, unlimited resources can be reached.

Cloud computing is applied to develop elastic, efficient and green architectures capable of offering cloud-based DSS services. These services are aimed at storing and processing the raw data, performing the computation and delivering the knowledge to the end-users. Operation research (OR) techniques play the role of modelling and solving real-world instances. High-Performance Computing (HPC) is used to boost decomposition techniques and algorithms. This way, cloud services are capable of providing practical solutions to complex problems faster.

In this context, it is essential to guarantee a certain quality of service and performance. The end users want a balanced trade-off between the response time, the usability and the quality of the solution. Furthermore, the intended user (decision maker) want to obtain the solution under an understandable form.

Proposals of this thesis are applied to help decision makers to make better strategical, tactical and operational decisions in the agribusiness field. This sector is the engine of the economy, wellness, and research of Lleida. The study performed in this thesis highlights the benefits of merging cloud systems, HPC and OR techniques to transfer expert and qualified knowledge to the society.

However, the methods and architectures developed can be applied to many other fields.

1.1 CONTEXT

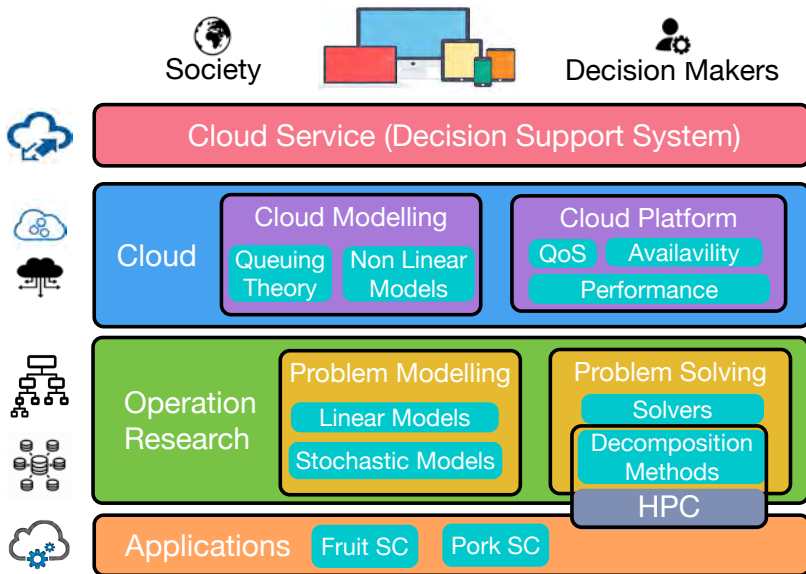


FIGURE 1.1: Overview of the cloud-based Decision Support System Infrastructure and the methodology studied, proposed and implemented in this thesis.

Figure 1.1 shows an overall scheme of the work involved in this thesis where Cloud systems are designed and modelled taking into account the Quality of Service (QoS), performance and availability. In the modelling of the cloud architecture proposed, a mixture between the queuing theory and the nonlinear programming technologies has been used. The resulting cloud platform is the testing bed used next to attend, orchestrate and service users requests.

The cloud service (DSS) deploys the required virtual architecture to carry out the resolution of the problems (stochastic or deterministic) either using a mathematical solver or a specific HPC decomposition method. This way, the cloud service is delivered as a client-server web application. The client is aimed to gather users data and requests, and the results are presented in a fancy way to

be deeply analysed. The server is the core of the application. Mainly it is aimed to manage the cloud infrastructure, orchestrate the executions and control the data processing.

Finally, note that the overall methodology developed in this thesis (mathematical programming, solvers, decomposition methods, cloud modelling and cloud platforms) are guidelines that can be adapted to any other field to offer knowledge to the society. Nevertheless, we focus the applications in the agribusiness field. This thesis deal with two applied cases, the fruit supply chain and the pork supply chain.

1.2 CLOUD BASED DECISION SUPPORT SYSTEMS (DSS)

Decision support systems (DSS) are an information system that offers solutions to help a user in the decision-making process. Figure 1.2 presents the most important problems that traditional DSS have to face. First of all, integrate the data belonging to different heterogeneous sources, such as management software, relational databases, and even though embedded devices. Secondly, they must be usable and attractive to allow a fluent and simple user interaction. Thirdly, they must be capable of generating descriptive analytics or indeed extract knowledge from the data. Finally, the last but not least, a DSS must have business credibility and successful pilot cases.

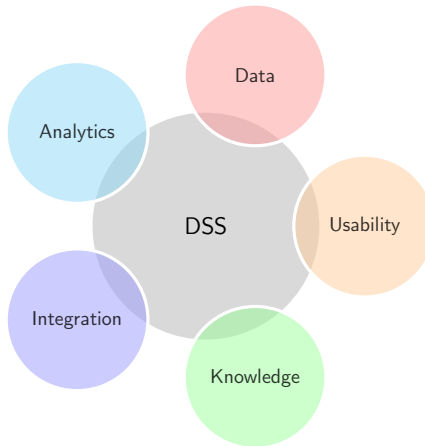


FIGURE 1.2: Main features of decisions support systems.

Nowadays, a smart DSS needs to extend these capabilities and develop on top of that valuable information either with the use of predictive or prescrip-

tive methods, by sending system alerts, by suggesting optimised alternatives or by providing the possibility to assess different scenarios. This way, DSS must evaluate and mitigate the risk of making decisions, improve the planning and manage better the resources. Note, that a smart decision support system is the combination of traditional DSS with current expert systems aimed at supporting business and organisational decision-making activities. Therefore, DSS require sophisticated Business Intelligence, Business Analytics, Artificial intelligence and data science features to deliver fundamental, consistent, data-driven and high-quality insights to decision makers. Besides, it is essential that novel DSS could be integrated within the current tools and devices that potential stakeholders are used to working.

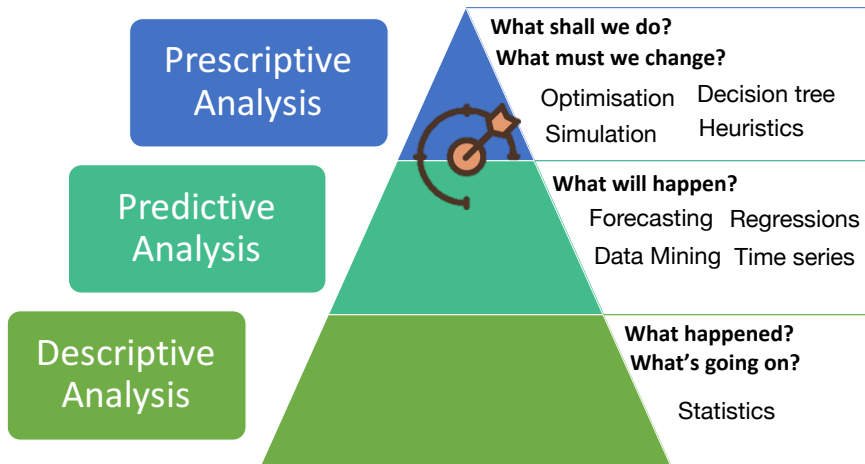


FIGURE 1.3: Comparative between traditional and novel DSS. The size of the pyramid floor depicts the relationship between % of current DSS and type of analytics offered.

Figure 1.3 summarises the main differences between the traditional DSS focused on answering the questions: What happened? and What is going on? by using basic statistical methods and the new generation of DSS, aimed at explaining: What will happen? What shall we do? or What must we change? by applying simulation or optimisation models among other techniques.

For the sake of discussion, I would like to argue that decision support systems are relatively similar to traditional client/server web application structures. On the basis of the evidence currently available, it seems fair to claim that web applications and DSS needs to deal with heterogeneous sources of data stored

locally or remotely. Moreover, web applications and the DSS needs a client to interact with end-users, gather information, process petitions and execute operations. It is important to take advantage of restful APIs and the JSON format to orchestrate and communicate smart cloud architecture components: server, client, data sources and workers. Figure 1.4 depicts these similarities and the technology involved.

A cloud-based DSS takes advantage of cloud features to overcome the limitations of traditional DSS. The cloud platform not only makes the DSS accessible for everyone also provides elasticity and scalability to create and destroy virtual resources on demand. Hence, data storage can scale properly and adapt to very different situations. In this context, hardware consuming procedures, such as processing data or model executions, run with the right hardware power. Additionally, it provides a better usage of the resources, real-time supporting and backups, load balancing and also remote disaster recovery.

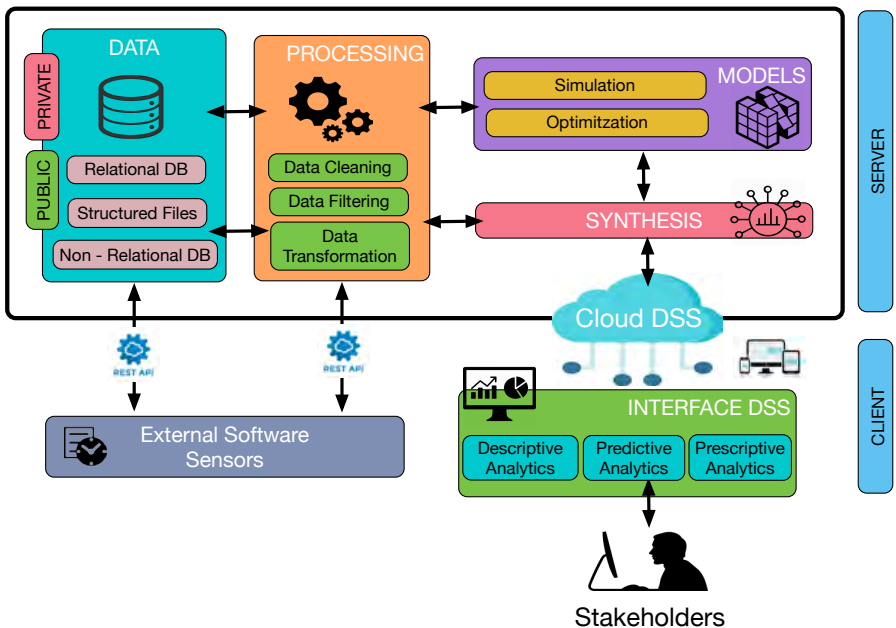


FIGURE 1.4: Cloud architecture of a DSS.

1.3 CLOUD COMPUTING

1.3.1 *Cloud Platform*

Let us start defining the main concepts of cloud computing. Mainly, there are three kinds of cloud services:

- Infrastructure as a service (IAAS): clients pay-as-you-go access to storage, networking, servers and other computing resources in the cloud.
- Platform as a service (PAAS): offers access to a cloud-based environment in which users can build and deliver applications.
- Software as a service (SAAS): provides software and data through the Internet. Users subscribe to the software and access it via the web or APIs.

In IAAS terms, there are several cloud platforms. On the one hand, in the commercial ones, users must purchase their services (Amazon EC2, Windows Azure, IBM Blue Cloud Google AppEngine, Sun Cloud and more). On the other hand, the open source platforms can be used free of costs and also help to maintain the community; for instance, OpenStack, Eucalyptus, OpenNebula, Nimbus or Apache CloudStack. OpenNebula was the selected one in this thesis. Simplicity, scalability and flexibility are the main features for choosing this framework to deploy the cloud platform and services proposed in this work.

Software-as-a-service (SAAS) eliminates the requirement for a powerful computer with an environment dedicated to executing tasks. The only thing needed is a terminal connected to the Internet. With only a web browser, end-users obtain access to almost unlimited computing power, no matter which device is used (handheld device, desktop computer or laptops).

Clouds come in different forms depending on their management and the sensitivity of data. For instance: public, private, hybrid, community and federated.

- *Public clouds* are externally available for everyone. Some examples of public clouds can be Amazon Elastic Compute Cloud (EC2), IBM's Blue Cloud and Google AppEngine.
- *Private clouds* are owned by a single company. The goal of private clouds has no control over the company data.
- *Community clouds* share their infrastructure and hardware between different companies.

- *Federated clouds* connect local infrastructure providers to a global marketplace that enables each participant to buy and sell cloud services on demand.
- *Hybrid Clouds* are a mixture of the previous clouds. Companies can either maintain crucial data inside a private cloud or rely on public clouds for other kinds of tasks and necessities.

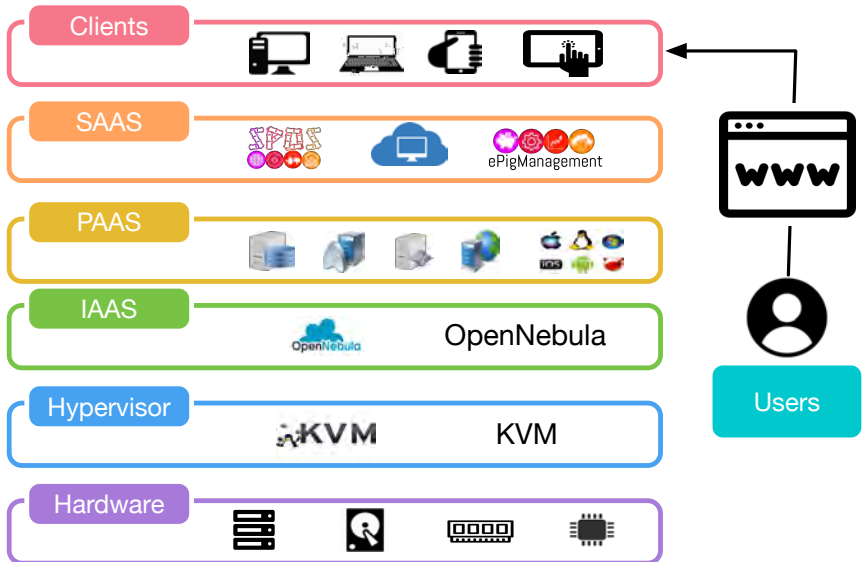


FIGURE 1.5: Cloud computing overview. Main layer schema.

Figure 1.5 highlights the main Cloud platform adopted in this Thesis. This platform relies on KVM to efficiently use the physical hardware resources to obtain the virtual ones. OpenNebula plays the vital role in creating and manage virtual infrastructure and deliver IAAS to the end-user. Further on, the virtual resources (machines, disk, and networks) deployed inside Open Nebula can be used as a remote virtual desktop, database server, application server and more to deliver PAAS. Finally, the services proposed in this thesis take advantage of the IAAS and the PAAS generated inside Open Nebula to create complex architectures to deliver real knowledge to the end-user as SAAS, for instance, ePig or SPOS will be further introduced. These are the associated layers and technologies used in this thesis to develop the cloud platform and orchestrates the proposed cloud services (DSS).

Stormy, see [2], is the cloud platform based on OpenNebula hosted at the University of Lleida used in this work.

1.3.2 *Cloud Research Lines*

Although cloud computing is a novel research topic, many challenges, and issues are related to cloud computing. Cloud architectures must provide availability, variability, reliability, elasticity, and system security while maintaining a balanced trade-off between cost and quality of the service. In this context, the quality of the service is the capability of providing better service over various technologies. It represents the problem of allocating resources to guarantee a certain level of service to the end-users. Further on, data backups, data integration, data migration and data recovery are also important aspects to be considered.

We highlight the following:

- *Availability*: points out the long-term fraction of the time of actually delivered service. Could be understood as the probability of an adequate QoS. It is a measure of the system's readiness.
- *Reliability*: the ability of uninterruptedly performing tasks. It is a measure of the continuity of correct service.
- *Serviceability*: ability to detect and correct problems. It can be understood as the flexibility to deal with faults.
- *Elasticity*: degree of fitting to the real needs. It is the smart feature that finds a balance between over-provisioning and under-provisioning resources.
- *Variability*: capacity to adapt to uncertain workloads produced from spontaneous changes in resources demand. It measures the time-varying workloads.

As the cloud computing systems continue to grow in complexity and popularity, their failures imply a high impact on the applications deployed inside the cloud. Thus, there is an increasing need to address availability, serviceability, and reliability concerns. Usually, short outages can be accepted, but more extended interruptions or accumulated disruptions exceeding a certain threshold may not be tolerable. A system that fails on average every two minutes but becomes operational again after a few milliseconds in 100 days is not very reliable but highly available [3].

In this context, the quality of the service (QoS) play a vital. The QoS is regulated by service-level agreements (SLAs) role. The SLA is a contractual relationship between a cloud service customer and a cloud service provider, where the provider guarantees a minimum QoS to the customer.

In this thesis, we focus on guaranteeing the service level agreement regarding availability and performance while keeping the QoS and user satisfaction. Nevertheless, security, variability, serviceability, elasticity or reliability are also key-stone aspects of conducting novel research.

1.3.3 Cloud modelling

In this thesis queuing theory and non-linear programming are the fundamental methodologies used to model cloud systems.

1.3.3.1 Queuing theory

Queuing theory studies the way customers are chosen for service in a waiting queue. It is a discipline that provides tools to obtain several performance metrics to deal with these situations.

A queuing model needs two distribution to represent the interarrival times and the service times. Furthermore, needs to know the number of servers. In this context, assuming an exponential distribution (Markovian), the average number of stakeholders in the system (N), the utilisation factor (ρ) or the average time in the system (T) can be obtained from the model. In contrast, for other distributions, these values can be obtained by simulation. For instance, consider one of the most widely used queues, the $M/M/1$ (1 server with exponential arrival rate and service time), the metrics can be obtained solving equations:

Equation 1.1 shows how to obtain the utilization factor (ρ):

$$\rho = \frac{\lambda}{\mu}, \quad \lambda < \mu \quad (1.1)$$

Equation 1.2 represents the average number of customers (N) and is obtained as follow:

$$N = \frac{\rho}{1 - \rho} \quad (1.2)$$

Finally, equation 1.3 (T) means the average time in the system.

$$T = \frac{\frac{1}{\mu}}{1 - \rho} \quad (1.3)$$

Nevertheless, queueing theory results are not directly applicable to Cloud modelling but assuming that the distribution of service time is unknown and the traffic intensity is uncertain. These models allow us to obtain reasonable approximations and determine the service level and relationship between the maximum number of tasks and the minimum number of resources. Furthermore, in this thesis, the times between failures and recoveries are also modelled using queues to deal with availability issues.

1.3.3.2 *Non Linear Programming*

Nonlinear programming (**NLP**) is a trending topic field in the literature with a lot of new research papers emerging. The general form of a nonlinear programming problem is to optimize a scalar-valued function f of several variables x subject to other functions (constraints) that limit or define the values of the variables. In mathematical terms,

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & c(x) = 0 \\ & l \geq x \geq u \end{aligned} \tag{1.4}$$

where $c(x)$ maps \mathbb{R}^n to \mathbb{R}^m and l, u are the lower and upper bounds that may contain infinite components. In general speaking, the objective function of the **NLP** has many locally optimal solutions. Thus, finding the best of all is often difficult. An important special case of nonlinear programming is convex programming in which all local solutions are global solutions and makes it easier to resolve. The most effective solver used in the literature to solve this kind of problems is Baron [4] or SCIP. Moreover, Couenne [5] and Bonmin [6] are the most common heuristics used to deal with this kind of problems.

This methodology has been used to design several policies to ensure a certain level of QoS while guaranteeing SLA concerning availability and performance.

1.4 CLOUD COMPUTING AND OPTIMIZATION METHODS

The optimisation models are growing exponentially in complexity. The data-sets used to feed them are also larger and heterogeneous than previously (big data). Thus, traditional environments and solvers are not powerful enough to compute the solution in a reasonable amount of time or indeed, to tackle it. A brief introduction to mathematical programming can be found in Appendix B.

Operation research can take advantage of cloud features to overcome these limitations and provide all kind of solutions to the OR MINOTAUR (Mixed, Inte-

ger, Nonlinear, Optimization, Toolkit, Algorithm, Underestimators, Relaxations). First of all, cloud computing allows OR to obtain resource on demand without human interaction. This way, clouds can automatically be provisioned and released, to meet demand fluctuations. Moreover, OR can gain access to a large pool of resources to serve multiple requests. Finally, a cloud platform allows transparency for both the consumer and the provider. These situations bring the opportunity to develop elastic cloud platforms to handle the dimensions of the data-sets properly and deliver OR tools to society.

High-Performance Computing (HPC) most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation to solve large problems in science, engineering, or business.

The cloud allows the integration of high-performance computing paradigms. This way, faster, smarter and more robust algorithms can be designed and deployed inside a cloud platform, taking advantage of the benefits of cloud elasticity, resource management and scalability.

In this thesis, efforts are focused on integrating the decomposition methods in the cloud platform and improving them using the HPC paradigms. This work takes advantage of the special mathematical structure of stochastic models. These models are characterised by having large dimensions and special matrix that makes them tractable by decomposition.

These decomposition methods are excellent candidates to apply a parallel master-worker paradigm to speed up and synchronise the resolution of the decomposition method. Furthermore, it is also possible to transform this paradigm into a collaborative parallel framework. In this case, each scenario or cluster is solved concurrently directed by the same master-worker paradigm, but the resolution of the workers can take advantage of heuristics or genetic algorithms to obtain additional information to speed up the convergence. Another possibility is implementing the heuristics in the outer-loop to obtain different candidates to be evaluated inside the inner-loop.

In this context, these HPC paradigms can be deployed inside the cloud with all the benefits commented before and also with dynamic adjustments and fault-tolerant behaviour. The last but not the least, the cloud allows delivering these optimisation methods as a service to the overall community.

A brief review of traditional decomposition methods used in this research such as Benders Decomposition and Lagrangian Decomposition are presented in this section.

1.4.1 *Benders Decomposition*

Benders Decomposition (BD) is a method used to solve stochastic linear problems via scenario analysis. The trick of benders decomposition consists in splitting the original model using a variable θ . First of all, an initial reduced master problem is generated considering only the information related to the first stage. Then, the feasibility and optimality of the solution is ensured by adding cuts produced by the second-stage scenarios subproblems.

Scenario clustering is one of its smart improvements that speed up the resolution time, taking advantage of tighter feasible cuts found by grouping scenarios into clusters. Additionally, the number of iterations can be decreased by adopting a multi or single cut approach.

1.4.2 *Lagrangian Decomposition*

Lagrangian decomposition is a method that exploits the theorems of Lagrangian relaxation and the Lagrangian dual. Scenario clustering techniques are also valid for this method. The goodness of this method is the elimination of a large number of complicated constraints with the addition of a set of parametric terms into the objective function. Although it seems to be at least as complicated than before, the reality is very different, now we get a large parametric objective function, but we have less constraint and the same number of variables. Thus, the model is faster to resolve, approximating the values of the dual parameters. A more in-depth explanation of the method can be found in Appendix C.

1.5 TWO CASE STUDIES IN THE AGRICULTURAL SECTOR

Many real-world decision-making situations arise from agribusiness. This sector is becoming more and more complex, technical and competitive. Furthermore, this sector plays a vital role in many economies around the world, in particular, Lleida. This thesis is focused on improving DSS agroindustry models and techniques.

Although significant work is being done in this decision models applied to agroindustry, there is still much to do to bring computer science into the agribusiness area. The complexity of mathematical models is an issue to apply OR solutions. Then, its implementation requires the understanding of the real problem, formulate, model, test, receive feedback, make corrections and implement the solution. The combination of traditional decision support systems with the cloud systems allow overcoming these problems.

The agribusiness sector is claiming for technification and the research developed in this thesis is aimed to deliver practical DSS tools based on cloud computing.

In this Thesis we pay attention to two agribusiness examples, the *Pork supply chain* and the *Dried Apple supply chain*.

1.5.1 *Pork supply chain*

Nowadays, the emerging trends in the pork industry are no longer based on the competition between individual farms units but rather integrated into a supply chain where the cooperatives or the integration companies deal with the animals' flow, market demand, market opportunities and even though the administration of all the facilities. This way, the farmers need only to take care of their production, to supply the overall flow. The main farms involved in the pork supply chain are the sow farms, the rearing farms, the fattening farms, the abattoirs, the cutting rooms and also the pig feed factories.

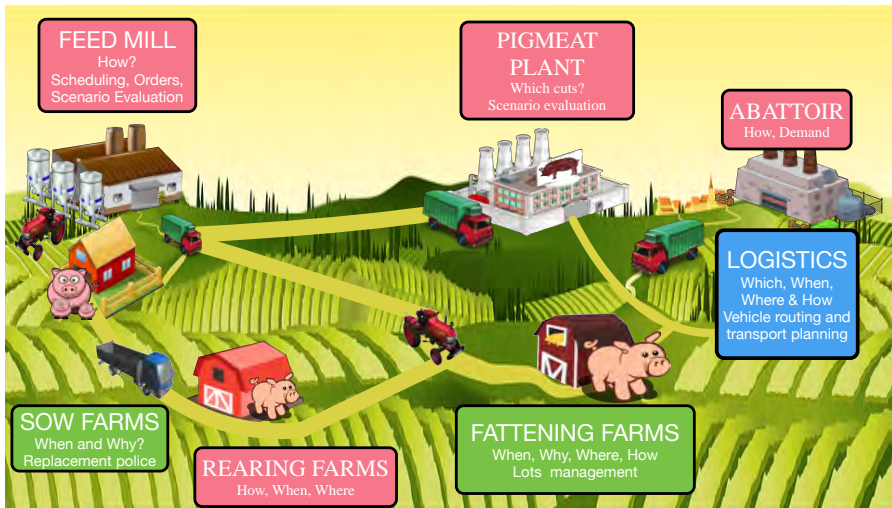


FIGURE 1.6: Overall structure of the pork supply chain. Actors and issues involved. Problems tackled in this thesis.

Figure 1.6 depicts a typical pork supply chain. This picture highlights in green colour the main issues tackled in this thesis, in red colour emphasises other important parts of the pork supply chain not covered in this thesis. Moreover,

blue colour denotes secondary problems also related to and solved by the models proposed.

This thesis focuses the efforts on improving one of the initial problems inside the supply chain, the sows replacement. The replacement of sows is a well-known problem related to pigs prolificity. This problem tries to answer the "When and Which" questions. Hence, the model determines the optimal moment to replace a sow. This thesis focus on integrating a well-studied hierarchical optimisation model inside a cloud-based DSS.

Furthermore, this thesis tackles the problem of delivering fattened pigs from the fattening farm to the abattoir. The available evidence seems to indicate that in this process are involved different decisions. First of all, at the beginning of the market windows, farmers must decide the number of weeks required to empty the current pigs batch, so a strategic decision is needed. However, in this process, other operational decisions are crucial such as the way that farmers decide how and how many animals send to abattoir each week to satisfy the strategic decisions. This thesis develops stochastic models to face the uncertainty in prices during the market window.

1.5.2 *Dried Apple supply chain*

Traditionally, fresh fruit supply chain in Chile is represented by independent agents collaborating for the same purpose and coordinated by a processing or packing plant. In particular, during the harvesting season, the manager at the dehydration plant has to make decisions about the purchase, transport, and storage of enough fresh apples for the plant to operate until the next season. The operations in this chain are quite similar to other fruit and vegetable chains preserved in cold storage during the harvest season for later processing by the agro-industry during the rest of the year. Figure 1.7 summarises all of the actors involved and the flows between them.

There is previous research, deterministic models and DSS proposed behind this problem. This research has been based on collaboration with Chilean researchers and entities. The historical data gathered by Chilean researchers claim that the quality of the raw material has a crucial impact on the final cost. This way, in this thesis, stochastic programming is used to extend the deterministic model, introducing uncertain fluctuations in the quality of the raw material. Furthermore, the results of the model are managed to deliver a more natural way to understand them, opening the path to the development of cloud-based DSS.

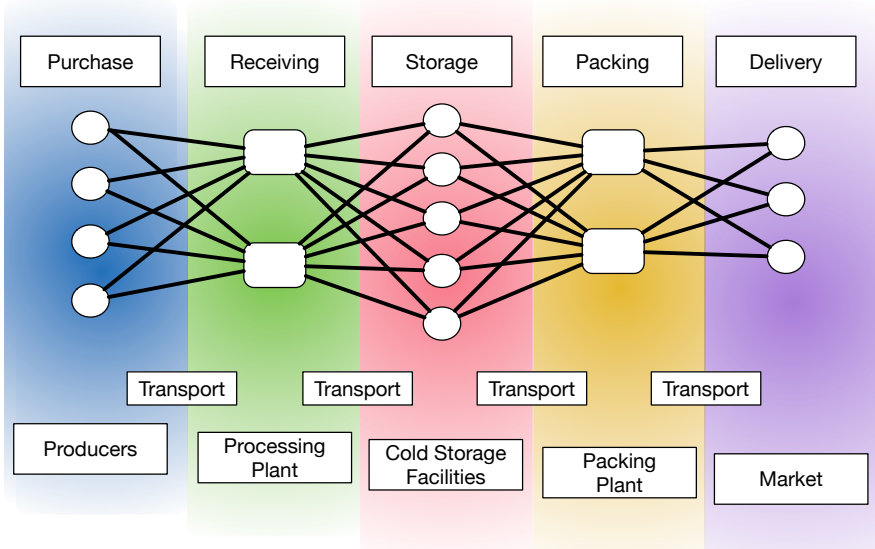


FIGURE 1.7: General structure of the apple supply chain for processing in Chile. Adapted from *Optimizing fresh food logistics for processing: Application for a large chilean apple supply chain*, by Soto-Silva et al., 2017, *Computers and Electronics in Agriculture*, 136:42-57. Adapted with permission.

1.6 RELATED WORK AND CONTRIBUTIONS

1.6.1 *Decision Support Systems and Agrobusiness*

The emerging technological advances, such as the Internet of Things, Cloud Computing, Big Data or Artificial Intelligence, during the last decades, have introduced radical changes in the business environment. In this context, Decision Support Systems (DSS) has become a relevant instrument to manage this complexity. In particular, the agribusiness sector has entered a new era, where precision, real-time information and automatization are the key-stone to succeed. In [7], an inspiring review focused on smart farming and new technologies is presented. There is a rapidly growing literature in this field such as [7], [8], or [9], which indicates the novelty and the increasing interest of consolidating the emerging technologies in the agribusiness sector. Hence, decision making is a vital aspect to survive and be efficient in this 4.0 world.

The majority of the decision support systems (DSS) described in the literature are based on simulation and optimisation models. Nevertheless, as has been observed by several authors [10, 11] there are several reasons why the majority of OR tools does not impact on the society. The main reasons can be summarised as a) unclear focus of the project, b) steep learning curve, c) lack of interface with the current management information systems, or d) hard system maintenance. [12] analyses the current state of the agricultural systems science, which emphasises the definition of a Use Case as the central pivot for the development of the interactions between models and users, and expose the difficulties for the final user to find easily accessible and usable OR models. It highlights the necessity to develop cloud-based DSS to transfer the knowledge and make a meaningful impact on the society.

Moreover, uncertainty and risk play a vital role in the development of real-based models, and therefore they are crucial to improve the decision support systems. Nevertheless, in this sector, there is a considerable lack of stochastic models and risk aversion proposals. Pla [13] made an in-depth and exhaustive literature review. This work put forward the claim that there are a huge heterogeneous set of successful deterministic models proposed to deal with different problems related to the agribusiness. This thesis pretends to fill this gap proposing models that deal with uncertainty and can be integrated into a complete DSS.

The main contribution of this work is putting complex mathematical models, algorithms, and tools as a service of the society comfortably and intuitively, to help and support strategical, tactical, and operational decisions. Thus, the cloud-based DSS proposed in this thesis merge and integrate the data, feed mathematical models and relies on parallel algorithms to transform the theoretical research into useful applications.

1.6.2 *Cloud Computing*

The Internet is often used for transaction-based applications, such as online banking, stock trading, and shopping. Social networking services, e-commerce, and cloud computing-based businesses have been behind the enormous growth in data-centre systems in recent years [14]. As the cloud computing systems continue to grow in complexity and popularity, it is vital to ensure the availability, the reliability while building green platforms even in the presence of faults. These failures impose tremendous implications on the applications deployed inside the cloud, so there is an increasing need to address availability and performance concerns.

In *Queueing Theory*, previous research showed that cloud systems could be modelled through an $M/M/m$ open network [15]. Frequently used methods of modelling computer service performance subjected to such *QoS* metrics as response time, throughput and network utilisation have been extensively studied in the literature [16–20]. In [17], the authors obtained the response time distribution for a cloud with an $M/M/m/m+r$ system model. Both inter-arrival and service distribution times were assumed to be exponential, and the system had a finite number of $m+r$ size buffers. The complexity of other queues ($G/M/m$, $M/G/m$, $G/G/m$) comes from the impossibility of obtaining a closed formula to represent the probability distributions of the response or waiting times of customers in the queue, and therefore approximate models must be found [21].

The main contribution in this field is the analysis and design of a cloud-based system, modelled using different queues taking into account the performance and the quality of service capabilities. Furthermore, the literature abounds with linear models and heuristics to solve scheduling problems. The main contribution in this regard is the proposal of models to ensure a certain level of quality of service, defined in the SLA contracts.

1.6.2.1 Availability

In the field of cloud computing, research has recently been focused on designing highly available and fault-tolerant systems. Authors in [22] highlights the priority for this research in robust business continuity plans in any big company. Although many efforts have been dedicated to analysing the availability of cloud (or web) hosts using measurement-based techniques [23, 24], less emphasis has been put on modelling the web service availability taking into account the impact of server node failures and performance degradation [3]. The modelling of the availability of fault-tolerant cloud computing systems using a Markov model to fulfil the client request was proposed in [25]. This work discusses various heterogeneous availability models for a few failure structures and designs and compares different recovery techniques.

Moreover, in business management, research on the prevention and the recovery from a disaster is a top priority. First of all, disaster recovery (*DR*) focuses on the operation ability of business operations and resources to resume activities after disaster occurrence [26–28] whereas disaster tolerance *DT* emphasises resilience and continuity of business operations and computing systems and resources when a disaster occurs. Therefore, the industry has been switching from *DR* (which may mitigate system downtime in days to weeks) to *DT*.

Meaningful information generated by scheduling problems should be merged with job availability in clouds. This way, we detect an increasing number of

research papers studying scheduling problems that are subject to machine or job availability constraints [29–31]. In this context, machines or jobs may be unavailable for specific time intervals. In [29], job behaviour was analysed regarding unavailability constraints. An integer linear programming (*ILP*) algorithm was proposed for scheduling problems that occur when the weighted number of late jobs that are subject to deterministic machine availability constraints must be minimised.

Furthermore, authors in [32] developed analytical models of the availability of Cassandra [33] (implementing Distributed Hash Tables (*DHT*) [34], a type of structured peer-to-peer (*P2P*) storage system) when confronting transient failures. This paper also deals with transient failures, the ones that imply the recovery of a failed node after some time, without any loss of the previously-stored information. Faulting nodes will lose updates during their off-line period [35]. This type of failure can be caused by network or power outages and node disconnections caused by users (i.e. overfull, churning, etc.). The literature survey makes apparent that traffic models are suitable to represent the patterns of real web-based environments. In this context, some of the properties considered in the literature deal with file size distributions, self-similarity [36], and the reference locality [37]. The results in [38] suggest that the aggregate web traffic tends to smooth out as Poisson traffic (i.e. exponential distribution) in observations of short time windows. It was further highlighted in [39].

The main contribution in this field is to propose an SLA decision model capable of optimising and analysing the cost and quality impact in the hosting of cloud-based applications (i.e. *SaaS*) concerning availability and performance. This solution is aimed to guarantee a predetermined service level agreement regarding availability while minimising the response time and cost of the system.

1.6.3 *Decomposition Methods*

There are several methods in the literature proposed to solve stochastic models via decomposition methods. Nevertheless, not all the techniques can be applied to real applications.

In particular, an important branch of algorithms is related to Benders decomposition (BD) methods [40]. For instance, [41] proposed the L-Shaped algorithm to solve two-stage stochastic linear problems when uncertainty in the model is represented by random parameters depending on scenarios with discrete and finite distributions. The improvement of the L-shaped method is highly relevant and a subject of ongoing research [42] because the problem may become rather large with the increase in the number of scenarios. Recent achievements

include the generation of tighter feasible cuts, as it is proposed in [43] for the so-called Cluster Benders Decomposition algorithm (CBD). The scenarios are grouped into clusters to fit the problem size into the underlying architecture to optimise the finding of tighter feasible cuts in the second stage. The benders decomposition algorithm has been successfully applied to a wide range of optimisation problems. Authors in [44] describe the current state of the art of this technique very profoundly. They present a vast variety of situations where the algorithm is applied successfully and also highlights the impact and the necessity of improving this technique by using new parallel collaborative frameworks.

Lagrangian decomposition is a method that exploits the theorems of Lagrangian relaxation and the Lagrangian dual. Lagrangian decomposition is prominent in the literature in the OR field. There is a huge range of works where Lagrangian decomposition was used to solve several real-life problems efficiently. First of all, in the year 2000, a research conducted by Nowak et al. [45] proves the goodness of applying Lagrangian relaxation to a stochastic power scheduling in the hydrothermal field. Since then, a lot of authors applied the method successfully to a huge range of problems and fields. More recently, Ghaddar et al. (2015) [46] also uses this technique to solve a scheduling problem related to water networks. Given the centrality of this issue thought the recent history, other authors do deeper theoretical research to improve the technique and also to adapt the technique to more complex stochastic trees. An example of that is the usage of an augmented Lagrangian to deal with multi-stage stochastic programs, see [47]. The professor Escudero proposed more works, see [48] and [49]. These works adapt classical clustering techniques to speed up the serial resolution of the method. Furthermore, Escudero's team extend also the method to generic multi-stage trees. On logical grounds, there is no compelling reason to argue that this resolution method does not lead to successful results in a feasible time in most cases.

The main contribution in this area is aimed at drawing firm parallel proposals to improve the traditional OR decomposition methods. In this thesis, a parallel improvement of Benders and Lagrangian decomposition are proposed to solve large instances of two-stage stochastic models. Besides, these proposals alleviate the computational load in serial procedures, reduce computational time, and make huge cases treatable.

1.6.4 *Pork Supply Chain (PSC)*

In the literature, several models seek the sow replacement problem. Porkchop is a spread-sheet based model developed in the eighties by Dijkhuizen [50]. They

introduced a new index called the Retention Pay-Off that indicates the total extra profit that retention of a sow is expected to yield over her replacement. Huirne *et al.* [51] presents a stochastic dynamic programming model that maximises the present value of the expected annual net returns over a specified planning horizon of a sow herd. This model was refined later by introducing other attributes such as the piglet mortality rate or the number of undersized piglets at birth [52]. Kirchner *et al.* [53] propose the use of decision trees based on the gain ratio criterion to decide whether a sow should be replaced or not. In this case, sows are not ranked but classified. A multi-level hierarchical Markov process with decisions on multiple time scales is presented by Kristensen *et al.* [54, 55]. The model incorporates the biological performance of sows through a dynamic linear model, the dynamic dropout structure and a feed intake model. Latter [56] the model was also extended to include clinical observations. Pla *et al.* [57] uses an equivalent semi-Markov model to represent farm dynamics to optimize the expected rewards. Finally, Rodríguez *et al.* [58] present a two-stage stochastic programming model to optimise the sow replacement and the scheduling of gilt purchases.

The main contribution in this area is to present an elastic architecture that can be used to develop new decision support systems in almost every industrial process. In particular, this work is focused on proposing a cloud-based service to support sow replacement that can be easily used by farmers or farm managers. Users can take advantage of the cloud features to get a tool that can be fitted and integrated with his/her current farm management software. Besides, traditional farmers that work today with spreadsheets and notebooks also have the opportunity to use this tool to improve their decisions.

1.6.5 Fruit Supply Chain (FSC)

The fresh fruit supply chain is a challenging problem. It is characterised by long supply lead times combined with significant supply and demand uncertainties, and relatively thin margins.

Soto-Silva *et al.* [59] recently reviewed operational research models applied to FSCs showing studies focused on different stages of the chain. They remark the methods referred to supplier selection have concentrated mainly on multi-criteria analysis, mathematical programming and artificial intelligence [60] and [61]. The main decision criteria used in supplier selection are quality, delivery time, price, manufacturing capacity, service, management, technology, research and development, flexibility, reputation, relationship, risk, safety and environment [62], [63] [64]. Anojkumar *et al.* [65] advocated determining simple and

logical selection criteria to make efficient decisions in the shortest time. Soto-Silva et al. [59] proposed the coordination of cold chamber opening and closing according to the refrigeration technology. Rong et al. [66] and Nakandala et al. [67] took the degradation of quality along the supply chain into account, the latter emphasising transportation from producers to retailers. Routing problems are found more often in the distribution of perishable foods [68] than in fruit production or processing [69].

Most of the models mentioned above are deterministic showing that decision-making models under uncertainty are one of the main challenges of the agricultural sector in general and the fruit industry in particular [70]. Recently, Borodin et al. [71] revised the latest advances and developments in the application of operations research methodologies to handling uncertainty in agricultural supply-chain management problems. Out of over a hundred papers that Borodin et al. [71] reviewed, only a couple of them were related to FSCs [72] [73].

The main contribution of this thesis is the development of a two-stage stochastic model to optimise the purchase and storage policies for seasonal fruits. Furthermore, we define the guidelines to build a smart DSS to bring this model from the research world to the society.

1.7 RESEARCH OBJECTIVES

The main purpose of this thesis is to achieve agribusiness solutions delivering high-performance computing methods and mathematical models through cloud-based technologies and architectures, especially to allow decision makers to make better decisions. Hence, to deal with this challenge, the following objectives were defined:

1. To develop cloud systems capable of delivering SAAS with SLA guarantees.
2. To develop parallel algorithms capables to solve large stochastic mathematical programming instances.
3. To implement a cloud architecture integrating the guidelines developed previously to provide optimisation as a service.
4. To design practical models to assist decision-makers in agribusiness.
5. To propose cloud DSS services and architectures to bring the knowledge to the farmers in a transparent, automatic and straightforward way.

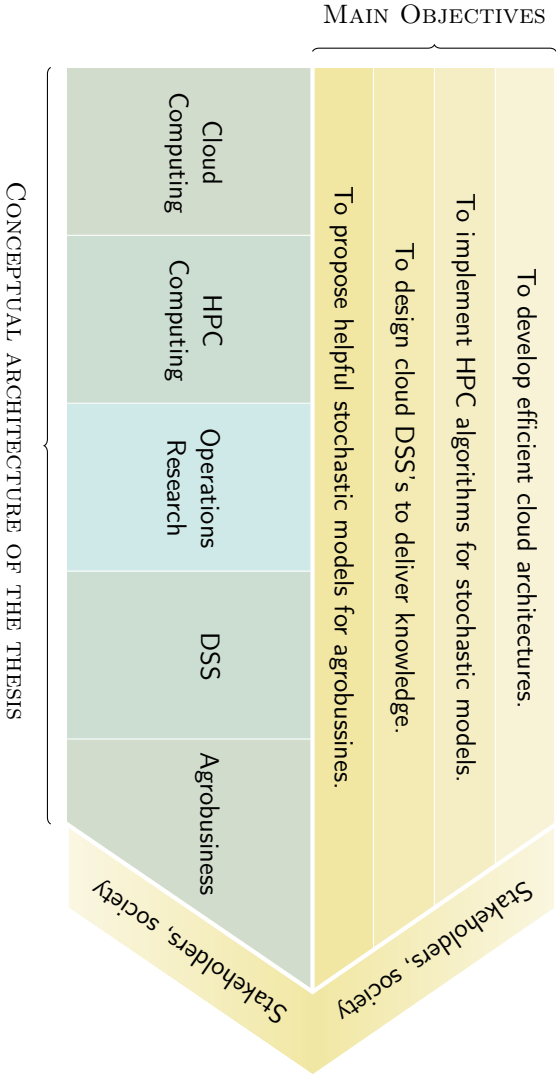


FIGURE 1.8: Conceptual architecture of the thesis. Main Objectives, topics and research directions.

1.8 OUTLINE

The remainder of the thesis is organised as follows. Chapter 1 presents the introduction, the motivations and a general overview of the work done in this thesis. Chapter 2 is related to the first research objective proposing an SLA decision model capable of optimising and analysing the cost and quality impact in the hosting of cloud-based applications (i.e. SaaS) concerning availability and performance. Chapter 3 and 4 are related to the second research objective. In these chapters, a parallel improvement of Benders and Lagrangian decomposition are proposed to solve large instances of two-stage stochastic models. Furthermore, Chapter 5 deals with the third research objective presenting a cloud service (SPOS) capable of solving deterministic linear and non-linear models as a service. Next, Chapter 6 proposes a two-stage model to assist managers in the purchase and storage for seasonal fruits seeking with the fourth research objective. Chapter 7 shows the implementation of a cloud-based decision support system to help pig farmers in the sow replacement filling the last research objective. Finally, the Global discussion of the results, the Conclusions and Future Work are detailed in Sections 8 and 9.

Authors: *Jordi Mateo, Francesc Solsona, Jordi Vilaplana, Ivan Teixido and Josep Rius*

Journal: IEEE Access. The Multidisciplinary Open Access Journal

Publisher: IEEE

Year: 2019

DOI: <https://doi.org/10.1109/ACCESS.2019.2905870>

ISSN: 2169-3536

Keywords: *cloud computing, availability, response time, service level agreement, quality of the service*

CART, a decision SLA model for SaaS providers to keep QoS regarding availability and performance

Abstract: Cloud systems are becoming a powerful tool for business. The evidence of the advantages of offering infrastructure, hardware or software as a service (IaaS, PaaS, SaaS) is overwhelming. Therefore, for SaaS providers, it is essential to know the virtual resources required to optimise the service offered and keep the quality of service (QoS) at the desired levels. This paper presents an analytic model CART (Cloud availability and response time) for obtaining the best trade-off between performance, cost, and availability in a cloud system aimed to provide software as a service. The model aims to guarantee a predetermined availability and response time (RT) agreed with customers in a *service level agreement* (SLA) contract while minimising the cost of the system. A client-transparent error recovery strategy has been considered along with a Poisson traffic model. A sensitivity analysis of the simulated and real workloads is presented to study how a specific service level agreement influences the cloud service performance regarding the guaranteed response time and availability. The results corroborate the goodness of the model proposed, adjusting the real system accurately. Furthermore, the results obtained provide useful guidelines for cloud or web server designers.

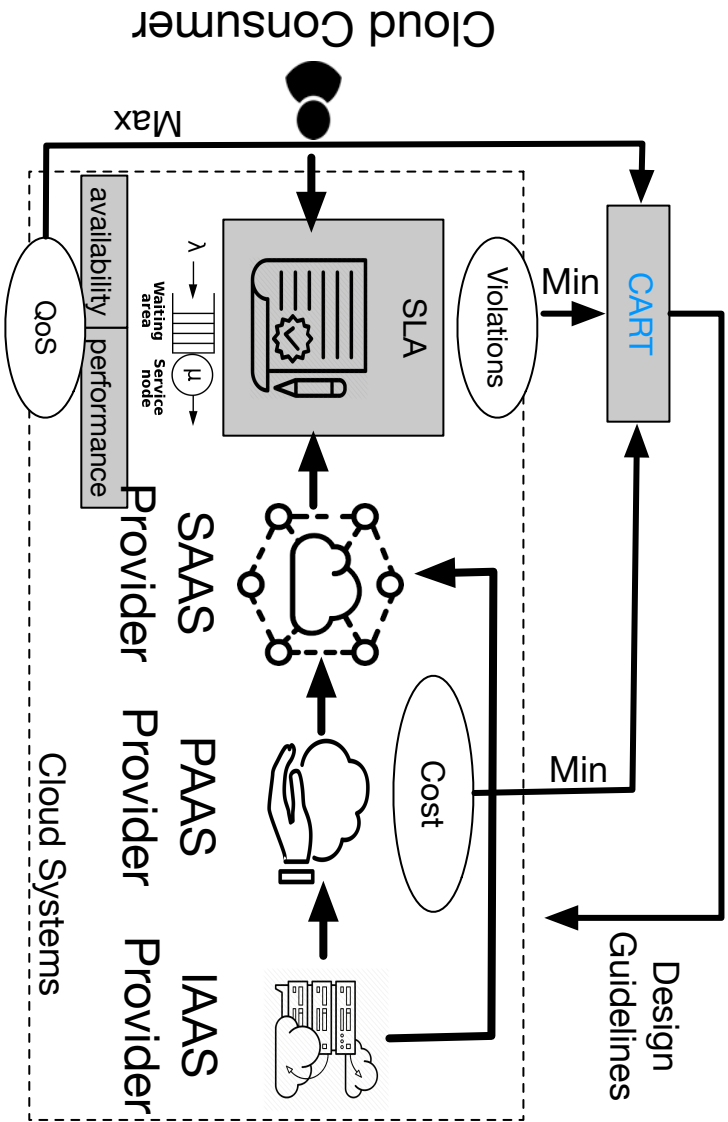


FIGURE 2.1: Graphical Abstract.

Received February 14, 2019, accepted March 7, 2019, date of publication March 18, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905870

CART, a Decision SLA Model for SaaS Providers to Keep QoS Regarding Availability and Performance

JORDI MATEO-FORNÉS¹, FRANCESC SOLSONA-TEHÀS¹, JORDI VILAPLANA-MAYORAL,
IVAN TEIXIDÓ-TORRELLES, AND JOSEP RIUS-TORRENTÓ

Department of Computer Science, University of Lleida, 25001 Lleida, Spain
INSPIRES, University of Lleida, 25001 Lleida, Spain

Corresponding author: Francesc Solsona-Tehàs (francesc@diei.udl.cat)

This work was supported in part by the Intelligent Energy Europe (IEE) Programme, in part by the Ministerio de Economía y Competitividad under Contract TIN2017-84553-C2-2-R, and in part by the European Union FEDER through CAPAP-H6 network under Grant TIN2016-81840-REDT. The work done by F. Solsona-Tehàs was supported by the Generalitat de Catalunya.

ABSTRACT Cloud systems are becoming a powerful tool for business. The evidence of the advantages of offering infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (SaaS) is overwhelming. Therefore, for SaaS providers, it is essential to know the virtual resources required to optimize the service offered and to keep the quality of service (QoS) at the desired levels. This paper presents an analytic model cloud availability and response time (CART) for obtaining the best tradeoff between performance, cost, and availability in a cloud system aimed at providing software as a service. The model aims to guarantee the predetermined availability and response time (RT) agreed with customers in a service-level agreement (SLA) contract while minimizing the cost of the system. A client-transparent error recovery strategy has been considered along with a Poisson traffic model. A sensitivity analysis of the simulated and real workloads is presented to study how a specific SLA influences the cloud service performance regarding the guaranteed RT and availability. The results corroborate the goodness of the model proposed, adjusting the real system accurately. Furthermore, the obtained results provide useful guidelines for cloud or Web-server designers.

INDEX TERMS Cloud computing, availability, response time, service level agreement, quality of service.

I. INTRODUCTION

Over the last decade, cloud computing has revolutionized society and the IT industry [1], [2]. According to the National Institute of Standards and Technology (NIST) [3], Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

In this context, computational resources can be offered to the users as a service (XaaS). The most common are infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS). Therefore, the quality of service (QoS) plays a vital role [4]. Nowadays, the *QoS*

is regulated by service-level agreements (SLAs). These are contracts between client and providers that express the price for a service, the *QoS* levels required during the service provisioning and the penalties associated with the *SLA* violations. Hence, service providers must design these contracts very carefully to maintain user confidence and avoid revenue loss [5]. The quality of service and user satisfaction have a significant relation: A lower level of *QoS* due to an *SLA* violation leads to a decrease in user satisfaction. The main challenge for a service provider is to determine the best trade-off between profit and customer satisfaction. This work is aimed at assisting service providers to design the optimal *SLA* contract regarding cost, performance and availability to maintain user satisfaction and the quality of service at the desired levels.

There is a rapidly growing trend in developing *SaaS* and real-time applications that open the door to new challenges in cloud-based hosting. The most crucial are to ensure

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

high availability (*HA*) and a reasonable response time (*RT*). To address these challenges, the central objective of this work is to design a decision model that minimizes the cost of the system and maximizes the *SLA* guarantees (based on the availability and response time of the system).

Availability is defined as the probability of an adequate *QoS* [6]. Moreover, availability is the long-term fraction of the time of service actually delivered. Short outages can usually be accepted, but more prolonged interruptions or accumulated disruptions exceeding a certain threshold may not be tolerable. The IEEE defines *HA* as the ability to wake components that have failed in the system. Morrill *et al.* [7] consider that the minimum level of *HA* is around 99.7%. Other cloud-computing issues, such as variability [8], system security [9] and reliability [10] were discarded.

Inspired by the work in [11]–[13], we represent the availability of cloud service as a Markov model, where the times between failures are exponentially distributed. Furthermore, a *M/M/1/k* queue was chosen to model the performance of a Cloud offering *SaaS*. The model consists of a cloud made up of *N* virtual machines (VMs). Considering the number of tasks as *b*, from here on $k = bN$, and the queue is renamed *M/M/1/bN* [14]. This queue was chosen not only as it fits in this Cloud environment properly but also because it allows the cloud's administrator to compute the response time quickly and reliably.

Optimizing the performance of data centers is prominent in the literature on cloud computing, see [12], [15], [16]. In this paper, the proposal is focused on designing a non-linear multi-criteria model (NLP) that minimizes the cost of the system while preserving *SLA* guarantees (based on the performance and availability). It is known that *NLP* problems are commonly harder to resolve to require more time and computational power. Nevertheless, in this case, a non-linear function represents the real behavior of the cloud service more accurately and the experiments suggest that it can be solved quickly.

There are different metrics for evaluating the performance of computer applications, such as the response time or throughput. The mean response time is perhaps the most significant performance metric in a cloud-computing context [17] dealing with *HPC* applications and real-time services, and so, this was the performance parameter chosen for this work. Hence, it is a negotiated *SLA* required to guarantee the *QoS* considered in our formulation. Moreover, the model proposed is also focused on guaranteeing a negotiated level of *QoS* regarding availability. Thus, another *SLA* is required to guarantee the aforementioned specific availability rate.

To address *QoS* requirements regarding availability, performance (response time) and cost, this paper makes the following contributions:

- (i) Modeling the cloud architecture using a queuing network model.
- (ii) Modeling the availability using a Markov model.
- (iii) Proposing a *SLA* decision model (CART) capable of optimizing and analyzing the cost and quality impact in

the hosting of cloud-based applications (i.e. *SaaS*). This model keeps user satisfaction and quality of service (availability and performance) at a negotiated *SLA*.

- (iv) Making an accessible CART model through a cloud-based service (AOS, Application Optimization Service) to assist cloud designers in the evaluation of their models. CART is available for everyone in *AOS*,¹ by clicking on “New session”, and then “Cloud/CART”.

II. LITERATURE REVIEW

Research related to *SLA*-based cost optimization and customer satisfaction is becoming an important branch in the Cloud computing area. Most of the works are focused on models oriented towards *IaaS* providers. For example, García *et al.* [18] propose an *SLA*-driven architecture for automatic provision, scheduling, allocation and dynamic management of cloud resources.

Nevertheless, one of the main challenges of cloud service providers is to ensure the quality of service by guaranteeing the *SLA* contract. Serrano *et al.* [19] propose a method that combines *QoS* with *SLA* in clouds aiming at facing challenges such as better performance, dependability or cost reduction of online cloud services. The paper shows advantages from providing *IaaS* and *PaaS*. Moreover, Hussain *et al.* [20] describe this situation for small-medium enterprises (SMEs). This work presents an exhaustive review of the current state of the art and highlights the main gaps in the research. They claim that a lack of a viable *SLA* management framework could lead to service interruption and contract violation penalties. Furthermore, service interruption and contract violation penalties affect customer satisfaction and cloud service trustworthiness directly.

In this context, availability, cost, reliability, dependability, performance and other cloud metrics are essential quality factors to design cloud-based services. The literature abounds with different modeling approaches that mix a subset of these metrics, see [21]. Wu *et al.* [22] present an optimization model to minimize cost and improve customer satisfaction level considering *SLA* violations and response time. Kouki and Ledoux [23] propose an analytical model to predict cloud service performance regarding the cost and the dependability of the service.

High Availability in cloud computing services is one of the most crucial quality metrics [24]. *HA* for cloud services is essential for maintaining customer confidence and preventing revenue losses due to *SLA* violation penalties [25]. Snyder *et al.* [26] claim that about \$285 million have been lost yearly due to cloud service failures.

Although many efforts have been dedicated to analyzing the availability of cloud (or web) hosts using measurement-based techniques [27], [28], less emphasis has been placed on modeling web service availability taking into account the impact of server node failures and performance degradation [10]. The modeling of the availability of

¹<http://stormy02.udl.cat/aos>

fault-tolerant cloud computing systems using a Markov model to fulfill the client request was proposed in [29]. This work discusses various heterogeneous availability models for a few failure structures and designs and compares different recovery techniques.

Meaningful information generated by scheduling problems should be merged with job availability in clouds. This way, we detected an increasing number of research papers studying scheduling problems that are subject to machine or job availability constraints [12], [15], [16]. In this context, machines or jobs may be unavailable for distinct time intervals. In [12], job behavior regarding unavailability constraints was analyzed. An integer linear programming (ILP) algorithm was proposed for scheduling problems that occur when the weighted number of late jobs that are subject to deterministic machine availability constraints must be minimized.

Frequently used methods of modeling computer service performance subjected to such *QoS* metrics as response time, throughput and network utilization have been extensively studied in [30]–[35]. Xiong and Perros [30] obtained the response time distribution of a cloud system modeled on a classic *M/M/m*, assuming an exponential density function for the inter-arrival and service times. Yang et al. [32] obtained the response time distribution for a cloud with an *M/M/m/m+r* system model. Both inter-arrival and service distribution times were assumed to be exponential, and the system had a finite number of *m+r* size buffers. The complexity of other queues (*G/M/m*, *M/G/m*, *G/G/m*) comes from the impossibility of obtaining a closed formula to represent the probability distributions of the response or waiting times of customers in the queue, and therefore approximate models must be found [36]. Several authors propose analytical models based on queuing theory to estimate the performance of a heterogeneous data center accurately, for instance [37] and [38].

Recent research has indicated that the speed of service, response time and service cost are not the only crucial factors in a cloud system [39]. The growth in the literature is particularly concentrated on power consumption, with some works related to the possibility of dynamically consolidating traffic flows dynamically on as few links as possible and turning off unused links and switches [40], [41]. Other authors recommend shutting down the spare nodes to make the system greener and more efficient concerning energy consumption [42], [43]. Another option is to select the right data center considering energy efficiency, *QoS* and *SLA* [44]. This way, Rossi et al. [45] propose an eco-orchestration approach to balance the energy-efficiency of a data center with a smaller impact on performance. Despite the importance of energy-efficient systems in cloud computing, this work only evaluates the *QoS* based on guaranteeing the *SLA* concerning availability, response time and cost.

III. CART MODEL

In this section, the cloud architecture is presented. Next, an availability model is proposed. A novel and attractive

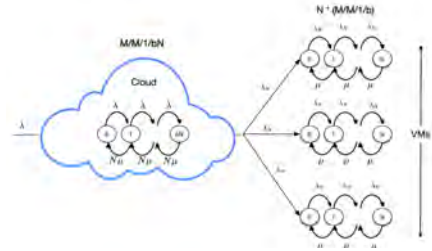


FIGURE 1. Cloud architecture model. One *M/M/1/bN* or *N M/M/1/b* queues. This figure represents the behavior of a web cloud made up of *N* slaves governed by a server.

feature of this approach is the use of non-linear optimization techniques to take advantage of the strengths of both well-studied models in the current state of the art (see the literature review, Section II).

A. CLOUD ARCHITECTURE MODEL

The performance of the cloud system proposed in this work is modeled by finite customers and a single server queue (*M/M/1/bN*), where *N* and *b* respectively represent the number of virtual machines and the capacity of each VM in the cloud. The cloud is made up of *N* virtual machines. Accordingly, the performance of a VM in the cloud is modeled by a queue (*M/M/1/b*). Figure 1 depicts the cloud system architecture. This figure shows that the entire cloud can be modeled using an *M/M/1/bN* queue. This can be broken down into *N M/M/1/b* queues.

The input traffic is modeled by a Poisson process with rate λ tasks/s (tasks per second). The authors chose a Poisson process for simplicity and for fitting the behavior of the web traffic properly. Since the cloud system has *N* available VMs, there will be *N* independent Poisson arrivals each with a $\lambda_N = \frac{\lambda}{N}$ rate. It is modeled like this to obtain a balanced system where each virtual resource has the same capacity. It means that each virtual resource receives a proportional amount of the workload. Moreover, each VM in the system has a service rate of μ tasks/s. Provided that the queuing model on this occasion is an *M/M/1/bN*, the response time (RT) can be obtained as follows:

$$RT = \frac{\hat{N}}{\lambda(1 - \rho_{bN})} \quad (1)$$

where \hat{N} represents the mean number of VMs in the system and is defined as:

$$\hat{N} = \rho * \rho_0 * \sum_{k=1}^{bN} \rho^k \quad (2)$$

and ρ represents the utilization factor and is defined as:

$$\rho = \frac{\lambda}{N\mu} \quad (3)$$



FIGURE 2. Markov model for availability.

Besides, p_i is the steady probability of having i tasks in this queue and is computed as:

$$p_i = \begin{cases} \frac{\rho^i(1-\rho)}{1-\rho^{bN+1}}, & \text{if } \lambda \neq N\mu \\ \frac{1}{bN+1}, & \text{otherwise} \end{cases}$$

If $i = bN$, the queue is full. This means that the system has no more free resources to execute new incoming tasks. Therefore, until the finalization of some tasks currently being processed, the system rejects all new incoming tasks.

B. AVAILABILITY

In this section, the availability model, inspired by the work presented in [10], is presented. The times between virtual machine failures are exponentially distributed at the rate α . We suppose that failures are not detected immediately. When failing, the system is reconfigured by restarting the virtual machine. After a virtual machine failure, the sum of the detection failure, system reconfiguration, restoration and reintegration times is assumed to be exponentially distributed at rate β .

Figure 2 shows the queuing model that describes this process. The N states of the Markov model represent the availability of the N virtual machines making up the cloud system used in the previous section.

The unavailability of the system can be defined as:

$$unavailability = \sum_{n=1}^N P_n L(n) + P_0, \tag{4}$$

where n represents the number of available VMs in the system. The state of the system represents the number of available VMs at any given moment. Thus, the steady-state probabilities, defined in eq. 5, represent the probability of having n available VMs at a specific time.

$$P_n = \frac{N!}{n!} \left(\frac{\alpha}{\beta}\right)^{N-n} P_N \tag{5}$$

Specifically, P_N and P_0 are defined in Eqs. 6 and 7 respectively.

$$P_N = \left[\sum_{i=0}^N \frac{N!}{(N-i)!} \left(\frac{\alpha}{\beta}\right)^i \right]^{-1} \tag{6}$$

$$P_0 = N! \left(\frac{\alpha}{\beta}\right)^N P_N \tag{7}$$

$L(n)$, defined in eq. 8 represents the loss probability due to a lack of capacity at VM n .

$$L(n) = np_b, \tag{8}$$

where p_b as defined in eq. 9 is the probability of loss due to a buffer overflow.

$$p_b = \begin{cases} \frac{\rho^b(1-\rho)}{1-\rho^{b+1}}, & \text{if } \lambda \neq \mu \\ \frac{1}{b+1}, & \text{otherwise} \end{cases} \tag{9}$$

Considering that the probability must be inside the interval $[0-1]$, the availability can be defined as 1 minus the unavailability rate. Thus:

$$availability = 1 - unavailability \tag{10}$$

C. CART MODEL

In this section, an optimization model is presented to provide the ideal number of virtual machines N required to guarantee a given level of availability (passed as an argument by some clients), while minimizing the response time and also the cost of the system. Let us consider the system in the cloud. The goal of the model is to find a balanced solution between cost and response time to guarantee a certain level of availability. In this context, the model finds the best trade-off to ensure the availability and performance of the QoS . This is achieved by mixing two mathematical approaches, queuing theory and nonlinear programming.

To that end, an objective function (OF) and its constraints are presented. The OF corresponds to the response time defined in eq. 1. In this case, we are interested in finding the number of virtual machines in the system that minimizes eq. 1. The requirements of this model are to set two main parameters that act as constraints in the model presented in this section. The first one is $SLA_Availability$, which represents the availability agreed with the customers. This way, the model can guarantee a negotiated level of availability for the QoS . The second one is the $SLA_ResponseTime$, which represents an upper-bound for the response time that the administrator of the system considers quick enough to serve the jobs. Moreover, this is also a crucial factor that the *SaaS* provider must agree with the customers in the SLA contract. In this work, the metric used to estimate this bound is efficiency. The response time used in this model is the minimum time that does not improve the efficiency significantly, taking into account a mean number of users. The efficiency can be evaluated considering the speed-up in executing a task in a single node and the time to execute a task when more processors are added to the system, always taking into consideration a mean number of user petitions. There is overwhelming evidence of the relationship between the cost and minimization of the response time. Thus, this constraint is needed to make a trade-off between the cost and the response time.

The system is used to design the central virtual architecture we need to maintain the availability levels while serving users with high efficiency. It is true that this architecture will not always be static. The cloud is elastic and capable of adapting to changes in the workload. Thus, a cloud-based platform must be capable of adjusting to the workload. If the

workload decreases, some virtual resources can be stopped. This amount can be obtained by recalculating the model. The same can be applied if the workload increases. So, the model can be recalculated when significant changes in the workload are detected. Furthermore, the results of the model are the minimum resources the system needs in a typical situation. However, the model can be recalculated to fit changes in the workload.

This function is formally defined by the following non-linear programming model:

$$OF : \min [RT] \quad (11)$$

$$s.t. : Availability \geq SLA_Availability \quad (12)$$

$$ResponseTime \leq SLA_ResponseTime \quad (13)$$

(11) is the OF to be minimized. Note that the resolution of such an equation is a non-linear problem. (12) is the SLA constraint regarding availability (defined in 10) the system must guarantee. (13) is another constraint for ensuring the agreed performance regarding response time. Prior to designing the SLA with the customers, cloud providers can restrict the unnecessary use of resources (computing nodes, cores, etc.) to satisfy users and keep the QoS at the desired level. Given the constants α , β , λ and μ , the solution that minimizes OF will obtain the value of the variable N , representing the number of virtual machines. (13) is used to obtain the most efficient N (i.e. minimum N), that is, an optimal OF but taking into account the number of resources.

The N obtained when applying the model will be the minimum number of virtual resources to ensure the QoS concerning the availability and performance required by the client.

IV. RESULTS

In this section, we present the results that corroborate the advantages of the model proposed. First of all, we describe the test-bed used in this study. Then, the model is evaluated to ensure the correctness of the results and highlight the influence of the availability and response time of the SLA on the design of cloud services. Finally, a real cloud service is deployed and monitored to show how the model proposed can simulate real systems and also to give useful guidelines for designing cloud services.

A. TEST BED

A small range of cloud architectures was considered and analyzed to show the good behavior of the model presented in section III-C. The main characteristics of each architectural-cloud design (Clouds 1, 2 and 3) are described in Table 1. λ is the average arrival rate of tasks. μ is the task service rate of the system. α is used to obtain $1/\alpha$, which represents the mean time for failure detection. β is used to calculate $1/\beta$, which represents the mean time for restarting a virtual machine; b represents the capacity of each virtual machine.

Table 2 describes the initial decision to optimize the system. As explained in section III-C, $SLA_ResponseTime$

TABLE 1. Cloud architectural characteristics.

Name	μ	α	β	b	λ
Cloud 1	5 tasks/s	20 s	200 s	20 tasks	20 tasks/s
Cloud 2	5 tasks/s	20 s	200 s	20 tasks	40 tasks/s
Cloud 3	15 tasks/s	20 s	200 s	20 tasks	800 tasks/s

TABLE 2. Numerical values of the decision parameters.

Name	$SLA_Availability$	$SLA_ResponseTime$
P1	0.7	0.01
P2	0.8	0.001
P3	0.85	0.08

represents the minimum response time that the system must ensure and $SLA_Availability$ represents the rate of availability that the system must guarantee.

In an attempt to test the veracity of the model proposed in this paper, the authors selected cloud characteristics randomly. In spite of this, the decision parameters displayed in Table 2 were carefully chosen to reflect different trends and enrich the discussion of the results.

The model presented and discussed in Section III-C was implemented and solved using Python [46] and the Sage 8.2 mathematical software [47].

B. MODEL EVALUATION

This section presents the results obtained by the model to study the impact of the minimization on the response time that guarantees a negotiated SLA regarding availability when increasing the number of VMs.

The principal results obtained from the test-bed alternatives (Cloud 1, 2 and 3) and the three combinations of the decision parameters $SLA_Availability$ and $SLA_ResponseTime$ (P1, P2 and P3) are presented in this section. These show the relationship between the number of virtual machines, the response time and availability.

Figures 3 and 4 depict the behavior of the availability and response time in all the clouds studied. Thus, the results show the impact on availability and response time when the cloud service is offered with a certain number of virtual machines (N).

Those figures highlight the influence of the number of virtual machines on the minimization of the RT , guaranteeing the availability and response time of the SLA . This way, a range between 1 and 50 virtual machines was used to evaluate Cloud 1, Cloud 2 and Cloud 3 architectures on the test-bed. The results show the different trends in the metrics analyzed.

These figures also show the minimum number of virtual machines required to ensure the level of the response time and availability of the SLA agreed with the customer, represented by marker-lines (–, P1), (+, P2) and (x, P3).

Figure 3 presents a decreasing trend between response time and the number of virtual machines. Thus, the more VMs the system has, the quicker it will reply. On the contrary,

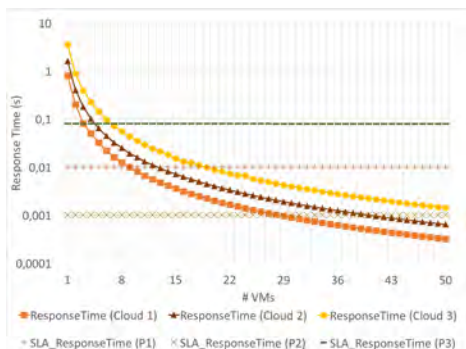


FIGURE 3. Response time.

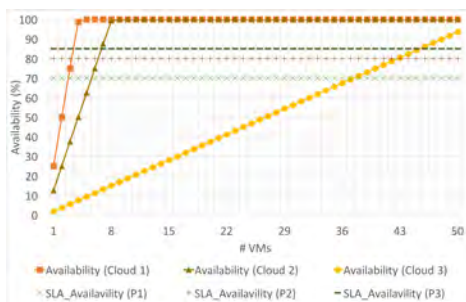


FIGURE 4. Availability.

an incremental trend between the availability and number of virtual machines can be appreciated in Figure 4. Therefore, the more virtual machines the system has, the more available it will be. Cloud 3 and $SLA_ResponseTime(P3)$ show a case in which there is not enough with 50 VMs to ensure a reasonable *Response Time*. In this situation, the cloud designer must invest in more than 50 VMs to satisfy their consumers. However, this bound on the response time can be incremented to reduce the overall cost. The final decision of whether to increase either this bound or the number of virtual machines will only depend on the trade-off between customer satisfaction and the system cost.

The minimum response time is found with 50 VMs in all the clouds. Note that after a certain number of VMs, the response time barely decreases, and ceases to be an essential metric to consider. The system cost increases with the number of VMs and the type of the contract agreed. Thus, a multi-objective minimization objective is justified for designing cloud services.

For Cloud 1 and 2, the principal influencing factor in economic terms (investing in VMs) is the response time, since the *SLA* is easy to achieve regarding availability in this

context (7 VMs to guarantee $SLA_Availability$ as much for P2, see Figure 4). However, in Cloud 3, availability is the most critical aspect. Thus, Cloud 1 needs 3 VMs to ensure $SLA_Availability$ (P1) and 4 for $SLA_Availability$ (P2) and $SLA_Availability$ (P3), whereas Cloud 3 needs 37, 45 and 47 VMs respectively. The system cost increases with the number of VMs and the type of the contract agreed. This assumption again justifies the need for a multi-objective minimization model for designing cloud services.

These results provide confirmatory evidence about the differences between designing clouds to ensure the response time and availability of the *SLA*.

C. CASE STUDY

In this section, a practical case study was designed to show how the proposed model can be applied in a real cloud environment. The cloud architecture analyzed offers software-as-a-service (*SaaS*). The application tested is a traditional cloud service. The client based on a web application performs tasks using a back-end to manage virtual machines in the cloud. The main steps are to deploy the VM, send and execute the job, send the results to the client and destroy the virtual machine. Furthermore, there is a queue to store the undone tasks. The fundamental objective is to show how the model proposed reflects reality and is helpful for *SaaS* providers to deploy the cloud architecture.

The experimental results were obtained using the Apache JMeter tool [48]. This can be used to simulate loads in a web service to test its strength or analyze overall performance by sending scheduled hypertext transfer protocol (*HTTP*) requests. The results presented in this section show the response time and availability of the system described above when monitoring the system with 1, 2 and 3 virtual machines contracted in the Amazon Web Services (*AWS*) to serve the incoming petitions. This study draws on the monitorization conducted by the JMeter tool for 1 minute and only contracted resources were used.

The average input and service task rate used in the study were $\lambda = 15$ tasks/s and $\mu = 5$ tasks/s. Therefore, the maximum capacity of each VM was restricted to five tasks (this is $b = 5$). The virtual machines (VMs) used were *1.micro*. The main features of this type of VM instance are computational power provided by 1.7 GB of RAM and 1 vCPU (Virtual CPU). These are aimed at general-purpose applications.

First of all, the simulation tool was used to verify the behavior of the cloud service. With the parameters described above, the cloud system solved by simulation is depicted in Figure 5. This figure shows that with contracting 3 VMs, the response time and availability of the *SLA* are satisfied.

Figures 6 and 7 show the number of incoming tasks per second (demand) generated by JMeter (right axis) and the response time or availability obtained by each cloud configuration (1, 2 or 3 VMs). Moreover, these pictures contain the desired levels (agreed with the customers) of response time and availability.

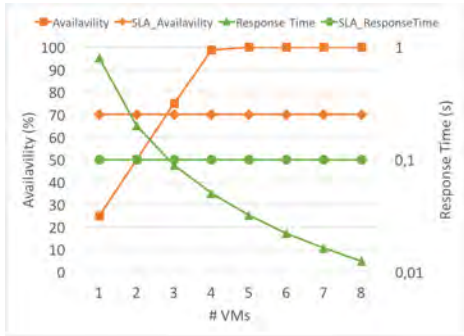


FIGURE 5. Case Study. Model results.

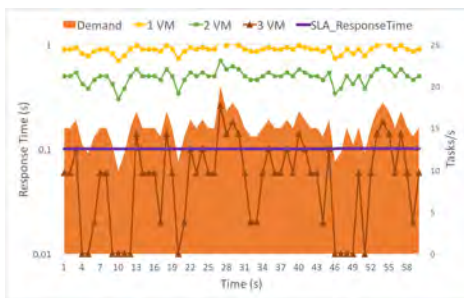


FIGURE 6. Case Study. Response time obtained with JMeter.

A closer look at the data indicates that the model solved by simulation (Figure 5) generates similar information (response time and availability) as the output obtained from monitoring the real system. Note that the percentage of availability increases while the response time decreases when more virtual machines are contracted.

Nevertheless, in the data obtained from monitoring the real system, moments in time were found when the response time and availability of the SLA are not satisfied. Therefore, the response time and availability exceed the SLA_ResponseTime and SLA_Availability respectively. In all these situations, we can ensure response time and availability by deploying pay-per-use instances but with a higher cost, as explained above.

Thus, using the model proposed, we can optimize the computational resources deployed. These demand peaks are produced by the high variability of the input tasks generated by JMeter, which attempts to emulate the real load of cloud data centers and websites. Considering the results of the model, this service will contract 3 virtual machines to guarantee SLA. Despite this, pay-per-use instances will be required when demand increases. In this study, pay-per-use instances will be required around seconds 13, 18, 27-30, 40,

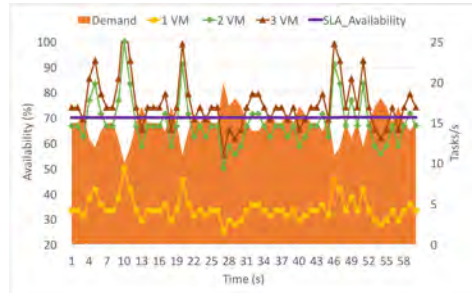


FIGURE 7. Case Study. SLA availability obtained with JMeter.

53-57 in the response time and 13, 19, 17-31, 40-41 and 52-56 in the availability.

Regarding the cost, for certain cloud providers like Amazon, the user can invest in pay-per-use or reserved capacity. With reserved resources, up to 74% can be saved compared with equivalent on-demand capacity. For example, the yearly fee of an instance *t1.micro* in the pay-per-use modality is around €175.68. Meanwhile, with a reserved capacity contract, the price is reduced to €103. In this study, the model suggests 3 VMs (reserved capacity), so we will have an annual cost of around $3\text{ VMs} \times \text{€}103 = \text{€}309$. However, without planning and using the pay-per-use modality, the yearly cost would be around $3\text{ VMs} \times \text{€}175 = \text{€}525$. In this case, the savings amount to 41%. Thus, using the solution in the model proposed, it is possible to plan how many reserved capacity contracts to deploy to ensure the SLA and then the demand variations can be satisfied with pay-per-use resources.

This experiment shows how the model proposed represents the behavior of a real service and can be used as a guideline to evaluate and design the response time and availability in new cloud services.

V. DISCUSSION

The cloud computing literature abounds with fruitful applications for ensuring the SLA either through availability [11] or response time [36]. However, there are fewer applications and models in the SaaS that consider both metrics to ensure QoS. The model presented in this paper contributes to providing a decision SLA model for SaaS providers to maintain QoS regarding availability and performance.

The industry demands tools to minimize overall costs by offering the best QoS to keep the user satisfaction and avoid paying SLA violations. The model presented in this paper is aimed at a practical application for the design of cloud services. It provides multi-criteria features for evaluating availability, response time and cost together simultaneously.

The results presented show that sometimes the number of VMs needed to ensure availability is higher than the one required to satisfy the response time condition, and vice-versa. The relationship between the availability of an SaaS

and client satisfaction is well known. In an *SaaS* context, the risk of losing clients due to a lack of service is very high. The same is also true for the *RT* criteria. Thus, for real businesses, it is crucial to have a multi-criteria model that ensures the satisfaction of the clients.

The results presented in Section IV provide convincing evidence about the correctness of the proposed model. It is proved that the information gathered from monitoring the real service is similar to that obtained by solving the model.

Morrill *et al.* [7] consider that the minimum level of *SLA* regarding availability is around 99.7%. With the proposed model, we provide a tool to design a cloud-based service with the best trade-off between cost and *SLA* concerning response time and availability.

Given the interest in deciding the number of contracts required for a service, cloud designers can use the model proposed to find out the minimum number of these needed to ensure a certain level of *SLA* concerning availability and response time. In the case study presented, this was contracting three reserved instances. This way, the system's credibility of good practices and service increases among the customers.

The data gathered in the results section suggest significant advantages to using the model, including the neglective cost and short time required to obtain a solution. Besides, the model can also be used to simulate different scenarios of *SLAs* regarding availability or response time. This way, cloud designers can check the differences between these configurations and check the best scenario for availability the cloud provider can offer its customers without compromising the cost.

To sum up, the model presented is capable of dealing with a considerable amount of information to make the best trade-off to keep *QoS* regarding availability, performance and cost and be used as a basis for designing the associated *SLA* contracts.

VI. CONCLUSIONS AND FUTURE WORK

The question under discussion in this study was the applicability of an optimization model capable of designing and evaluating cloud services. The results of this research provide confirmatory evidence that the combination of queuing theory models and optimization techniques can be applied efficiently in offering *SaaS*. Furthermore, the model was tested by comparing the solution of the model with the data gathered from monitoring a real system working under the same conditions. The data yielded by this study provide substantial evidence that the implementation of this technique is effortless and low-cost. Moreover, the results presented concerning availability and response time show the strength and advantages of using this technique to ensure *SLA*. This way, cloud providers can offer the users better and more reliable contracts. In the future, this model will be extended to consider another essential cloud *QoS* metric, the reliability of a cloud system. Hence, it would be fascinating to extend the proposed model for jointly evaluating the cloud service regarding availability, reliability, performance, energy

savings and cost in order to design more accurate *SLA* contracts and increase user confidence and satisfaction.

REFERENCES

- [1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generat. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X08001957>
- [2] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Gener. Comput. Syst.*, vol. 79, pp. 849–861, Feb. 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X17302224#b1>
- [3] P. M. Mell and T. Grance, "The NIST definition of cloud computing," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-145, 2011. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>
- [4] A. Abdelmaboud, D. N. A. Jawawi, I. Ghani, A. Elsaifi, and B. Kitchenham, "Quality of service approaches in cloud computing: A systematic mapping study," *J. Syst. Softw.*, vol. 101, pp. 159–179, Mar. 2015. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0164121214002830#bbib0066>
- [5] M. H. Goharamani, M. Zhou, and C. T. Hon, "Toward cloud computing QoS research: Analysis of cloud systems and cloud services," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 6–18, Jan. 2017. [Online]. Available: <http://ieeexplore.ieee.org/document/7815547/>
- [6] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation With Computer Science Applications*, 2nd ed. Hoboken, NJ, USA: Wiley, 2006.
- [7] H. Morrill, M. Beard, and D. Clitherow, "Achieving continuous availability of IBM systems infrastructures," *IBM Syst. J.*, vol. 47, no. 4, pp. 493–503, 2008.
- [8] A. Iosup, N. Yigibasi, and D. Epema, "On the performance variability of production cloud services," in *Proc. 11th IEEE/ACM Int. Symp. Cluster. Cloud Grid Comput.*, May 2011, pp. 104–113. [Online]. Available: <http://ieeexplore.ieee.org/abstract/document/5948601/>
- [9] J. Varia. (2010). *Architecting for the Cloud: Best Practices*. [Online]. Available: <http://docs.huioho.com/amazon/aws/whitepapers/AWS-Cloud-Best-Practices-January-2011.pdf>
- [10] M. Martinello, M. Kaäniche, and K. Kanoun, "Web service availability—Impact of error recovery and traffic model," *Rel. Eng. Syst. Saf.*, vol. 89, no. 1, pp. 6–16, Jul. 2005. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0951832004001723>
- [11] T. A. Nguyen, D. S. Kim, and J. S. Park, "Availability modeling and analysis of a data center for disaster tolerance," *Future Gener. Comput. Syst.*, vol. 56, pp. 27–50, Mar. 2016. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0167739X15002824>
- [12] B. Detienne, "A mixed integer linear programming approach to minimize the number of late jobs with and without machine availability constraints," *Eur. J. Oper. Res.*, vol. 235, no. 3, pp. 540–552, Jun. 2014. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0377217113008758>
- [13] D. A. Menasce, "Performance and availability of Internet data centers," *IEEE Internet Comput.*, vol. 8, no. 3, pp. 94–96, May 2004. [Online]. Available: <http://ieeexplore.ieee.org/document/1297280/>
- [14] L. Kleinrock, *Queueing Systems: Computer Applications*, vols. 2 and 66. Hoboken, NJ, USA: Wiley, 1976. [Online]. Available: http://www.biruni.tn/gw_2009_4_3/thumbs/contents-table/38/TM.383052.pdf
- [15] N. Hashemian, C. Diallo, and B. Vizvári, "Makespan minimization for parallel machines scheduling with multiple availability constraints," *Ann. Oper. Res.*, vol. 213, no. 1, pp. 173–186, Feb. 2014. [Online]. Available: <http://link.springer.com/10.1007/s10479-012-1059-8>
- [16] W. Kubiak, J. Blazewicz, P. Formanowicz, J. Breit, and G. Schmidt, "Two-machine flow shops with limited machine availability," *Eur. J. Oper. Res.*, vol. 136, no. 3, pp. 528–540, Feb. 2002. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0377217101000832>
- [17] R. Aversa, B. Di Martino, M. Rak, S. Venticinque, and U. Villano, "Performance prediction for HPC on clouds," *Cloud Comput., Princ. Paradigms*, pp. 437–456, 2011. doi: [10.1002/9780470940105.ch17](https://doi.org/10.1002/9780470940105.ch17).
- [18] A. G. García, I. V. Espert, and V. H. García, "SLA-driven dynamic cloud resource management," *Future Gener. Comput. Syst.*, vol. 31, pp. 1–11, Feb. 2014. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X1300215X>

- [19] D. Serrano et al., "SLA guarantees for cloud services," *Future Gener. Comput. Syst.*, vol. 54, pp. 233–246, Jan. 2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X15008081>
- [20] W. Hussain, F. K. Hussain, O. K. Hussain, E. Damiani, and E. Chang, "Formulating and managing viable SLAs in cloud computing from a small to medium service provider's viewpoint: A state-of-the-art review," *Inf. Syst.*, vol. 71, pp. 240–259, Nov. 2017. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0306437917020697>
- [21] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang, "Quality-of-service in cloud computing: Modeling techniques and their applications," *J. Internet Services Appl.*, vol. 5, no. 1, p. 11, Dec. 2014. [Online]. Available: <http://jisajournal.springeropen.com/articles/10.1186/s13174-014-0011-3>
- [22] L. Wu, S. K. Garg, S. Versteeg, and R. Buyya, "SLA-based resource provisioning for hosted software-as-a-service applications in cloud computing environments," *IEEE Trans. Services Comput.*, vol. 7, no. 3, pp. 465–485, Jul. 2014. [Online]. Available: <http://ieeexplore.ieee.org/document/6654162/>
- [23] Y. Kouki and T. Ledoux, "SLA-driven capacity planning for cloud applications," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 135–140. [Online]. Available: <http://ieeexplore.ieee.org/document/6427519/>
- [24] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: A reference roadmap," *Hum.-Centric Comput. Inf. Sci.*, vol. 8, no. 1, p. 20, Dec. 2018. [Online]. Available: <https://hccis-journal.springeropen.com/articles/10.1186/s13673-018-0143-8>
- [25] J. Dantas, R. Matos, J. Araujo, and P. Maciel, "Eucalyptus-based private clouds: Availability modeling and comparison to the cost of a public cloud," *Computing*, vol. 97, no. 11, pp. 1121–1140, Nov. 2015. [Online]. Available: <http://link.springer.com/10.1007/s00607-015-0447-8>
- [26] B. Snyder, J. Ringenberg, R. Green, V. Devabhaktuni, and M. Alam, "Evaluation and design of highly reliable and highly utilized cloud computing systems," *J. Cloud Comput.*, vol. 4, no. 1, p. 11, Dec. 2015. [Online]. Available: <http://www.journalofcloudcomputing.com/content/4/1/11>
- [27] D. Oppenheimer and D. A. Patterson, "Architecture and dependability of large-scale Internet services," *IEEE Internet Comput.*, vol. 6, no. 5, pp. 41–49, Sep. 2002. [Online]. Available: <http://ieeexplore.ieee.org/document/1036037/>
- [28] M. Kalyanakrishnan, R. K. Iyer, and J. U. Patel, "Reliability of Internet hosts: A case study from the end user's perspective," *Comput. Netw.*, vol. 31, nos. 1–2, pp. 47–57, Jan. 1999. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0169755298002293>
- [29] D. Mani and A. Mahendran, "Availability modelling of fault tolerant cloud computing system," *Int. J. Intell. Eng. Syst.*, vol. 10, no. 1, pp. 1–12, 2017. [Online]. Available: <http://www.inass.org/share/2017022817.pdf>
- [30] K. Xiong and H. G. Perros, "Service performance and analysis in cloud computing," in *Proc. SERVICES-I*, Jul. 2009, pp. 693–700. [Online]. Available: <http://ieeexplore.ieee.org/document/5190711/>
- [31] L. P. Slothouber, "A model of Web server performance," in *Proc. 5th Int. World Wide Web*, 1996, pp. 571–576. [Online]. Available: <http://www.occities.org/webserververperformance/modelpaper.html>
- [32] B. Yang, F. Tan, Y.-S. Dai, and S. Guo, "Performance evaluation of cloud service considering fault recovery," in *Cloud Computing*. Berlin, Germany: Springer, 2009, pp. 571–576. [Online]. Available: <http://link.springer.com/content/pdf/10.1007/978-3-642-10665-1.pdf#page=590>
- [33] B. N. W. Ma and J. W. Mark, "Approximation of the mean queue length of an M/G/c queueing system," *Oper. Res.*, vol. 43, no. 1, pp. 158–165, 1995. doi: [10.1287/opre.43.1.158](https://doi.org/10.1287/opre.43.1.158)
- [34] H. Karlapudú and J. Martin, "Web application performance prediction," Ph.D. dissertation, Dept. Comput. Sci., Clemson Univ., Clemson, SC, USA, 2004.
- [35] R. D. Van Der Mei and H. B. Meeuwissen, "Modelling end-to-end quality-of-service for transaction-based services in multi-domain environments," in *Proc. IEEE Int. Conf. Web Services*, Feb. 2006, pp. 453–462.
- [36] J. Vilaplana, F. Solsona, I. Teixidó, J. Mateo, F. Abella, and J. Rius, "A queuing theory model for cloud computing," *J. Supercomput.*, vol. 69, no. 1, pp. 492–507, 2014. [Online]. Available: <http://link.springer.com/article/10.1007/s11227-014-1177-y>
- [37] W.-H. Bai, J.-Q. Xi, J.-X. Zhu, and S.-W. Huang, "Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model," *Math. Problems Eng.*, vol. 2015, Jun. 2015, Art. no. 980945.
- [38] S. E. Kafhali and K. Salah, "Stochastic modelling and analysis of cloud computing data center," in *Proc. 20th Conf. Innov. Clouds, Internet Netw. (ICIN)*, Mar. 2017, pp. 122–126.
- [39] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *J. Netw. Comput. Appl.*, vol. 89, pp. 96–108, Jul. 2017. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S1084804517301108>
- [40] B. Heller et al., "ElasticTree: Saving energy in data center networks," in *Proc. 7th USENIX Symp. Netw. Syst. Design Implement.*, vol. 10, pp. 249–264. [Online]. Available: <https://www.usenix.org/legacy/event/nsdi10./doi/10.1.1.174.3815>
- [41] Z. Guo, S. Hui, Y. Xu, and H. J. Chao, "Dynamic flow scheduling for Power-efficient Data Center Networks," in *Proc. IEEE/ACM 24th Int. Symp. Quality Service (IWQoS)*, Jun. 2016, pp. 1–10. [Online]. Available: <http://ieeexplore.ieee.org/document/7590399/>
- [42] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, 2012. [Online]. Available: <http://onlinelibrary.wiley.com/doi/10.1002/cpe.1867/full>
- [43] D. Kliazovich, P. Bouvry, and S. U. Khan, "GreenCloud: A packet-level simulator of energy-aware cloud computing data centers," *J. Supercomput.*, vol. 62, no. 3, pp. 1263–1283, 2012. [Online]. Available: <https://link.springer.com/article/10.1007/s11227-010-0504-1>
- [44] B. Aldawsari, T. Baker, and D. England, "Trusted energy-efficient cloud-based services brokerage platform," *Int. J. Intell. Comput. Res.*, vol. 6, no. 4, pp. 1–10, 2015. [Online]. Available: <http://infonomics-society.org/wp-content/uploads/ijicr/published-papers/volume-6-2015/Trusted-Energy-Efficient-Cloud-Based-Services-brokerage-Platform.pdf>
- [45] F. D. Rossi, M. G. Xavier, C. A. F. De Rose, R. N. Calheiros, and R. Buyya, "E-eco: Performance-aware energy-efficient cloud data center orchestration," *J. Netw. Comput. Appl.*, vol. 78, pp. 83–96, Jan. 2017. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S1084804516302569>
- [46] *Python Software Foundation*. Accessed: Jun. 6, 2018. [Online]. Available: <http://www.python.org>
- [47] *Sage Mathematics Software*. Accessed: Jun. 6, 2018. [Online]. Available: <http://www.sagemath.org>
- [48] *Apache JMeter*. Accessed: Jun. 6, 2018. [Online]. Available: <http://jmeter.apache.org>



JORDI MATEO-FORNÉS received the ETIG degree and the B.Sc. and M.S. degrees in computer engineering from the Escola Politècnica Superior, Universitat de Lleida (UdL) in 2006, 2012, and 2013, respectively. He is currently pursuing the Ph.D. degree. His research interests and expertise include data science fields, such as operations research (stochastic optimization), high-performance computing (parallelization of algorithms), cloud computing, big data, and the Internet of Things (IoT), decision support systems, and Agriculture 4.0.



FRANCESC SOLSONA-TEJÁS received the B.S., M.S., and Ph.D. degrees in computer science from the Universitat Autònoma de Barcelona, Spain, in 1991, 1994, and 2002, respectively. He is currently a Full Professor with the Department of Computer Science, University of Lleida, Spain. His research interests include parallelization of algorithms, coscheduling, cluster computing, multi-cluster computing, P2P computing, cloud computing, desktop grid computing, scheduling, optimization, multi-stage linear and stochastic programming, reconstruction of metabolic pathways, bioinformatics, ecosystems simulation, mHealth, big data, and the Internet of Things (IoT).



JORDI VILAPLANA-MAYORAL received the Ph.D. degree in computer science. He is currently a Professor with the University of Lleida and also a member of the Distributed Computing Group. He is involved in multiple interdisciplinary projects alongside clinicians, psychologists, statisticians, and mathematicians. He uses telemedicine techniques, web-based applications, smartphone applications, image recognition, data entry, big data storage, processing and visualization, machine learning algorithms, and deep learning techniques. His research interests include cloud computing, eHealth and mHealth, big data, and machine learning applied to the health field.



JOSEP RIUS-TORRENTÓ received the B.S. and M.S. degrees in computer science from the Universitat de Lleida, Spain, in 2006 and 2008, respectively, the B.S. and M.S. degrees in business administration, in 2009 and 2011, respectively, and the Ph.D. degree in computer science from the Universitat de Lleida, in 2012, where he is currently an Associate Professor with the Department of Business Administration. His research interests include parallel and distributed computing, cloud infrastructures, big data, and data analysis. . . .



IVAN TEIXIDÓ-TORRELLES received the ETIS degree, the B.S. degree in computer engineering, and the M.S. in applied science to engineering from the Escola Politècnica Superior, Universitat de Lleida (UdL), Lleida, in 2007, 2009, and 2011, respectively. He is currently a Research Support Staff with UdL. His research interests include cloud computing, big data analysis, machine learning, and parallelization of scientific and technological applications.

Authors: *Jordi Mateo, Lluís M. Plà, Francesc Solsona and Adela Pagès*

Journal: Springerplus

Publisher: Springer International Publishing

Year: 2016

DOI: <https://doi.org/10.1186/s40064-016-2556-z>

ISSN: 2193-1801

Keywords: *Stochastic linear optimization, Parallelization, Scalability, Clustering in stochastic optimization, Benders Decomposition*

A production planning model considering uncertain demand using two-stage stochastic programming in a fresh vegetable supply chain context.

Abstract: Production planning models are achieving more interest for being used in the primary sector of the economy. The proposed model relies on the formulation of a location model representing a set of farms susceptible of being selected by a grocery shop brand to supply local fresh products under seasonal contracts. The main aim is to minimize overall procurement costs and meet future demand. This kind of problem is rather common in fresh vegetable supply chains where producers are located in proximity either to processing plants or retailers. The proposed two-stage stochastic model determines which suppliers should be selected for production contracts to ensure high quality products and minimal time from farm-to-table. Moreover, Lagrangian relaxation and parallel computing algorithms are proposed to solve these instances efficiently in a reasonable computational time. The results obtained show computational gains from our algorithmic proposals in front of the usage of plain CPLEX solver. Furthermore, the results ensure the competitive advantages of using the proposed model by purchase managers in the fresh vegetables industry.

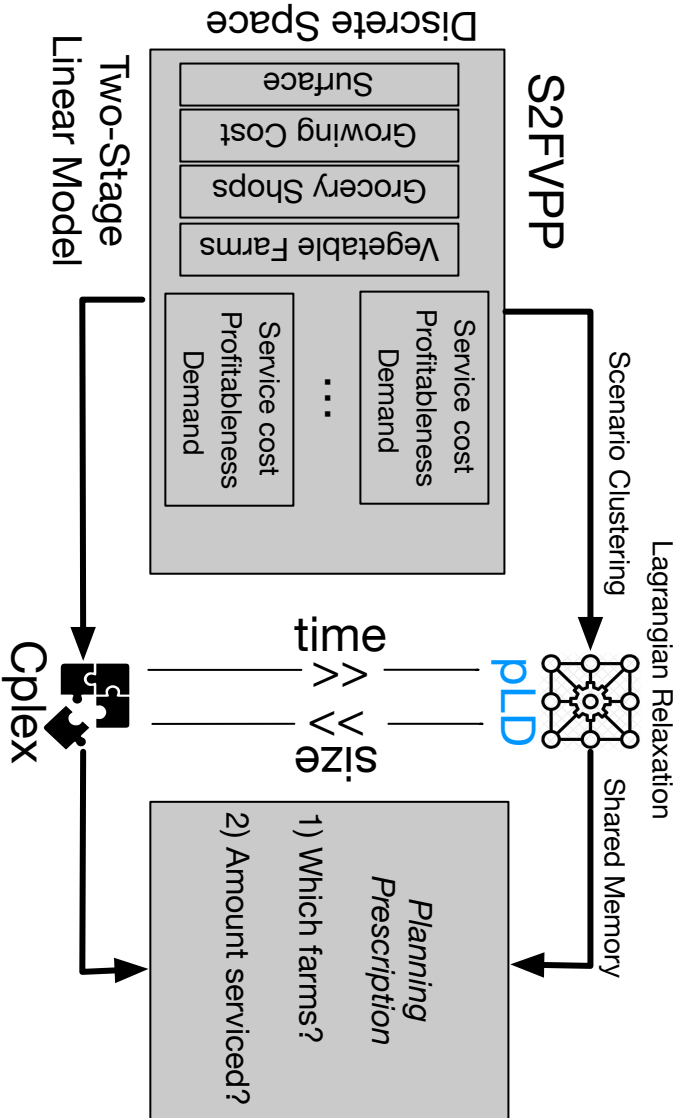



FIGURE 3.1: Graphical Abstract.

RESEARCH

Open Access



A production planning model considering uncertain demand using two-stage stochastic programming in a fresh vegetable supply chain context

Jordi Mateo^{1*} , Lluís M. Pla², Francesc Solsona¹ and Adela Pagès²

*Correspondence:

jmateo@diei.udl.cat

¹ Computer Science

Department and INSPIRES,
University of Lleida, Jaume II
69, 25001 Lleida, Spain

Full list of author information
is available at the end of the
article

Abstract

Production planning models are achieving more interest for being used in the primary sector of the economy. The proposed model relies on the formulation of a location model representing a set of farms susceptible of being selected by a grocery shop brand to supply local fresh products under seasonal contracts. The main aim is to minimize overall procurement costs and meet future demand. This kind of problem is rather common in fresh vegetable supply chains where producers are located in proximity either to processing plants or retailers. The proposed two-stage stochastic model determines which suppliers should be selected for production contracts to ensure high quality products and minimal time from farm-to-table. Moreover, Lagrangian relaxation and parallel computing algorithms are proposed to solve these instances efficiently in a reasonable computational time. The results obtained show computational gains from our algorithmic proposals in front of the usage of plain CPLEX solver. Furthermore, the results ensure the competitive advantages of using the proposed model by purchase managers in the fresh vegetables industry.

Keywords: Production planning, Supplier selection, Fresh vegetable supply chain, Two stage mixed 0–1 models, Lagrangian relaxation, Parallel computing

Background

Nowadays, little by little are appearing studies in which operations research are applied to solve agricultural problems for the agroindustry. See Pla et al. (2013) to review some opportunities and perspectives in this field. Some examples can be found for different sectors in the current literature, for example, a review of models for transport planning for the fresh fruit supply chain is presented in Soto-Silva et al. (2015). A collection of a variety of models applied to the agroindustry is included in Plà-Aragónés (2015), while Ahumada and Villalobos (2009a) review several agrifood supply chain models.

The fresh vegetable industry is a very important economic activity in Spain. Novel trends in Spain are related to the consumption of quality-local products. Many grocery shops prefer to add value to their products offering local fresh vegetables (fresh vegetables produced in the surroundings) rather than importing the same vegetables from other countries or regions at a lower cost. Green and sustainable products of proximity

are labels associated many times to this kind of products well appreciated by a more awareness consumer. Nowadays, the quality of a vegetable is not only measured through its appearance or flavour, but also by where it has been grown or how much traceable information can be offered to the final customer.

In this context, professional purchase managers in these brands have to deal with the problem of choosing a set of farms to contract production from season to season so as to minimize the overall cost while satisfying the future demand. In this way, these grocery shops are able to sell tasty and healthy locally-produced vegetables with full traceability, making the activity sustainable over time with beneficial effects for the local economy.

However, selecting a farm to grow vegetables on is a mid/long-term decision. The decision has to be made beforehand with the uncertainty of the future behaviour or conditions of production and demand for the following season. Thus, a two-stage model with a “here and now” strategy is needed to deal with this kind of problem. For more information about stochastic programming see Birge and Louveaux (2011), Shapiro et al. (2014) and Prékopa (2013).

The current literature on fresh vegetable supply chain contains several examples of successful implementations which are focused on the maximisation of the total revenue for the grower. For instance, see Ahumada and Villalobos (2009b). However, there are no successful model focused on minimising the costs in a competitive market where companies can rent or contract farms to grow up fresh vegetables. The model proposed in this paper is oriented to grocery shops, big retailers and distributors inside this particular context.

The objectives of the present study are:

- Design a model capable of dealing with the decision of choosing the best set of farms to contract their production in the fresh vegetable production context.
- A parallel algorithm is proposed to solve the model in order to alleviate computational load in serial procedures, to reduce computational time, and also to make possible the usage of the model by purchase managers in the agroindustry.

One of the main contributions of this paper is the proposal of a production planning model by a two-stage model capable of dealing with the problem of selecting suppliers in a fresh vegetable supply chain. This production planning model is based on the adaptation of the linear Uncapacitated Facility Location Problem (Cornuejols et al. 1983; Erlenkotter 1978), in which facilities are replaced by production contracts with suppliers. The model presented is very flexible and can be easily adapted to other financial contexts with similar characteristics and restrictions.

Numerous methods have been proposed to solve the Uncapacitated Facility Location Problem. For example, neighbourhood search heuristic (Ghosh 2003), Lagrangian based heuristic (Beasley 1993), specific heuristics (Avella et al. 2009), benders decomposition (de Camargo et al. 2008), branch-and-price (Ro and Tcha 1984), tabu search (Al-Sultan and Al-Fawzan 1999), Lagrangian relaxation (Wu et al. 2015), etc.

Another important contribution of this paper is the proposal of an algorithmic approach to solve huge real instances of two-stage mixed 0–1 models using parallel computing paradigms. This work presents the parallelization of the Subgradient Method

proposed in Escudero and Garín (1984). The combination of Lagrangian relaxation and parallel computing techniques makes this model flexible for purchase managers in their daily tasks. Current research seems to validate the potential of parallel computing in stochastic programming. Thus, this paper presents a novel way of dealing with production planning by farm selection inside a supply chain in the vegetable production environment.

A two stage model for uncertain fresh vegetable production planning

This model is adapted from the classic uncapacitated facility location problem (Cornuejols et al. 1983; Erlenkotter 1978). From here on, *S2FVPP* is used as the model name.

The objective of the proposed model is to evaluate the available fresh vegetable farms and determine the ones that minimize the overall cost so as to satisfy the uncertain demand of the potential customers. The model is intended for cooperatives or private companies to accept or recommend farms to be part of them or to distributors to agree production contracts.

The parameters and the decision variables used to formulate the model are listed in Table 1. The *S2FVPP* configuration decisions consist of choosing whether to use a fresh vegetable farm to grow fresh vegetables or not. A binary variable is associated with the selection of these fresh vegetables farms in such a way that $y_i = 1$ whether the fresh vegetable farm i is used to grow up the fresh vegetables; otherwise $y_i = 0$. Let x_{ij} denote the fraction of demand serviced from farm i to the customer j under a specific stochastic scenario ω . Furthermore, the cost of growing a unit of fresh vegetable in the fresh vegetable farm i is represented by cgv_i , and the last but not least, the cost of serving a customer j from fresh vegetable farm i under the scenario ω is denoted by csv_{ij}^ω . Furthermore, a customer cannot be served from a fresh vegetable farm unless we contract its production, see (1e). Besides, each customer j must be full served so (1b) is needed in the model. Moreover, the model must ensure that the total amount of demand is satisfied by the final farm selection, see (1c). Last but not least, the amount of demand served for each fresh vegetable farm can not exceed its maximum productivity, see (1d). Finally, (1f) defines the variable y_i as binary (Fig. 1) shows this model.

$$(S2FVPP) \quad \min \sum_{\omega \in \Omega} \pi^\omega \left[\sum_{i \in F} \sum_{j \in C} x_{ij}^\omega csv_{ij}^\omega \right] + \sum_{i \in F} cgv_i y_i \tag{1a}$$

Table 1 Notations used in the mathematical parameters

F	Set of farms available in our production field
Ω	Set of different uncertain scenarios
C	Set of different potential customers for our vegetables
y_i	Represents whether farm i is used or not
x_{ij}^ω	% of demand serviced from farm i to customer j under scenario ω
csv_{ij}^ω	Cost of servicing the customer i from farm j under scenario ω
cgv_i	Cost for growing a vegetable on farm i
d_j^ω	Demand of customer j under scenario ω
r_i^ω	Profitableness per hectare of farm i under scenario ω
s_i	Surface in hectares of farm i
π^ω	Represents the probability of scenario ω

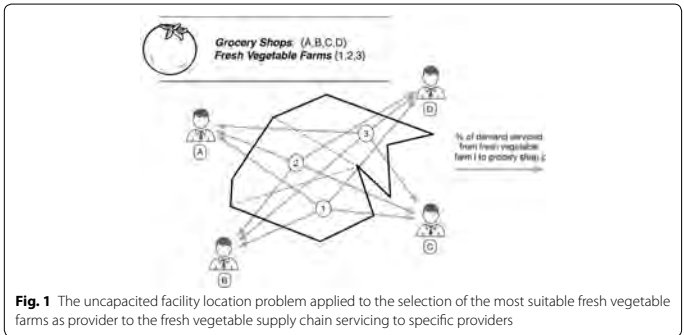


Fig. 1 The uncapacitated facility location problem applied to the selection of the most suitable fresh vegetable farms as provider to the fresh vegetable supply chain servicing to specific providers

$$s.t. : \sum_{i \in F} x_{ij}^\omega = 1, \quad \forall j \in C, \quad \forall \omega \in \Omega \tag{1b}$$

$$\sum_{i \in F} y_i s_i r_i^\omega \geq \sum_{j \in C} d_j^\omega, \quad \forall \omega \in \Omega \tag{1c}$$

$$\sum_{j \in C} d_j^\omega x_{ij}^\omega \leq r_i^\omega s_i, \quad \forall \omega \in \Omega, \quad \forall i \in F \tag{1d}$$

$$0 \leq x_{ij}^\omega \leq y_i, \quad \forall i \in F, \quad \forall j \in C, \quad \forall \omega \in \Omega \tag{1e}$$

$$y_i \in [0, 1], \quad \forall i \in F \tag{1f}$$

Problem (1), called *S2FVPP*, is expressed in compact representation, which is equivalent to its equivalent deterministic model (DEM), that, in the splitting variable representation, can be expressed as:

$$(S2FVPP) \min \sum_{\omega \in \Omega} \pi^\omega \left[\sum_{i \in F} \sum_{j \in C} x_{ij}^\omega c s v_{ij}^\omega + \sum_{i \in F} c g v_i y_i^\omega \right] \tag{2a}$$

$$s.t. : \sum_{i \in F} x_{ij}^\omega = 1, \quad \forall j \in C, \quad \forall \omega \in \Omega \tag{2b}$$

$$\sum_{i \in F} y_i^\omega s_i r_i^\omega \geq \sum_{j \in C} d_j^\omega, \quad \forall \omega \in \Omega \tag{2c}$$

$$\sum_{j \in C} d_j^\omega x_{ij}^\omega \leq r_i^\omega s_i, \quad \forall \omega \in \Omega, \quad \forall i \in F \tag{2d}$$

$$y_i^\omega - y_i^{\omega+1} \leq 0, \quad \forall \omega \in 1 \dots \Omega - 1, \quad \forall i \in F \tag{2e}$$

$$y_i^\Omega - y_i^1 \leq 0, \quad \forall i \in F \tag{2f}$$

$$0 \leq x_{ij}^\omega \leq y_i^\omega, \quad \forall i \in F, \quad \forall j \in C, \quad \forall \omega \in \Omega \tag{2g}$$

$$y_i \in [0, 1], \quad \forall i \in F \tag{2h}$$

Note that in the splitting variable model Problem (2), the first stage decisions are replicated for each scenario. The non-anticipativity constraints (NAC), see (2e) and (2f), are needed to make the problem equivalent to Problem (1). NAC constraints are expressed in that form to avoid the use of non-negative vectors of Lagrangian multipliers in the dualization of equality constraints. The advantages of dealing with the splitting variable model is the possibility of decomposing it into different independent scenarios and solving them using the Lagrangian decomposition (Escudero and Garín 1984).

Lagrangian relaxation

The Lagrangian relaxation (LR) of the *S2FVPP* for a given non-negative vector of weight $\mu_y = (\mu_{y_1}, \dots, \mu_{y_{|F|}})$ refers to the mixed 0–1 *LR* minimization Problem presented in this section.

$$(LR) \quad \min \sum_{\omega \in \Omega} \pi^\omega \left[\sum_{i \in F} \sum_{j \in C} x_{ij}^\omega c s v_{ij}^\omega + \sum_{i \in F} (c g v_i + (\mu_i^\omega - \mu_i^{\omega-1})) y_i^\omega \right] \tag{3a}$$

$$s.t. : \sum_{i \in F} x_{ij}^\omega = 1, \quad \forall j \in C, \quad \forall \omega \in \Omega \tag{3b}$$

$$\sum_{i \in F} y_i^\omega s i v_i^\omega \geq \sum_{j \in C} d_j^\omega, \quad \forall \omega \in \Omega \tag{3c}$$

$$\sum_{j \in C} d_j^\omega x_{ij}^\omega \leq r_i^\omega s i, \quad \forall \omega \in \Omega, \quad \forall i \in F \tag{3d}$$

$$0 \leq x_{ij}^\omega \leq y_i^\omega, \quad \forall i \in F, \forall j \in C, \quad \forall \omega \in \Omega \tag{3e}$$

$$y_i \in [0, 1], \quad \forall i \in F \tag{3f}$$

It can be shown that Problem *LR* is a relaxation of Problem (2) because:

- The feasible set of Problem *LR* contains the feasible set of Problem *S2FVPP*.
- For any feasible solution (x, y) from Problem *S2FVPP*, and also any positive μ , the solution of Problem *LR* is a lower bound on the optimal value of Problem *S2FVPP*, thus $Z_{LR} \leq Z_{S2FVPP}$.

The Problem *LR*(ω) provides a suitable structure to be decomposed in scenarios (ω).

$$(LR(\omega)) \quad \min \pi^\omega \left[\sum_{i \in F} \sum_{j \in C} x_{ij}^\omega c s v_{ij}^\omega + \sum_{i \in F} (cgv_i + (\mu_i^\omega - \mu_i^{\omega-1})) y_i^\omega \right] \quad (4a)$$

$$s.t. : \quad \sum_{i \in F} x_{ij}^\omega = 1, \quad \forall j \in C \quad (4b)$$

$$\sum_{i \in F} y_i^\omega s_i r_i^\omega \geq \sum_{j \in C} d_j^\omega \quad (4c)$$

$$\sum_{j \in C} d_j^\omega x_{ij}^\omega \leq r_i^\omega s_i, \quad \forall i \in F \quad (4d)$$

$$0 \leq x_{ij}^\omega \leq y_i^\omega, \quad \forall i \in F, \quad \forall j \in C \quad (4e)$$

$$y_i \in [0, 1], \quad \forall i \in F \quad (4f)$$

One of the most common approaches to solving the Lagrangian relaxation is the subgradient method (Fisher 2004; Escudero et al. 2004; Escudero and Garín 1984), known as a general-purpose method. It is used often to solve generic non-smooth convex optimisation problems. Hereafter, the question under discussion is whether or not a parallel implementation of Lagrangian decomposition using the subgradient method is suitable to solve efficiently models similar to the one proposed in this work.

Parallel Lagrangian decomposition

In this section, a parallel implementation of the Lagrangian decomposition method is proposed so as to gain computational efficiency in the resolution of problems such as model *S2FVPP*; see Problem (1). A serial implementation of Lagrangian decomposition using the subgradient method for dealing with two-stage stochastic mixed 0–1 models was presented and proposed in Escudero and Garín (1984).

The underlying argument in favour of designing a parallel implementation of Lagrangian decomposition using the subgradient method (**pSM**), is the independence between the problems generated by the scenario decomposition of model *LR*; see Problem (3). Furthermore, the operations performed by the subgradient method in each iteration are suitable to be executed in a parallel context too. Thus, the effort made in this work was focused on designing a parallel version of the subgradient method presented in Escudero and Garín (1984).

The parallel version **pSM** is identical to the serial version, with additional coding for shared memory data and synchronisation steps among the available computing cores. Instead of running a single computation task, the parallel implementation is able to run as many tasks as cores in the computing node are available.

The parallel algorithm proceeds to update first the objective function of the model *LR*(ω) from scenario 1 to scenario Ω with the value $\mu_{\omega^k}^k$, where k represents the current iteration of the method and ω represents a specific scenario, $\omega \in \Omega$. Then, these updated *LR*(ω) problems are solved concurrently. Each thread, stores information about

the solution such as the values of the variables $x_{ij}^{\omega k}$ and $y_i^{\omega k}$ inside the matrix structures, which belong to shared memory. The execution of this step does not represent any synchronisation problem in shared-memory environments. Since one of the dimensions of these matrices represents the assigned scenario, the threads, even executed in different cores, will not overwrite the same solution. After this step, **pSM** reduces all the partial solution of $LR(\omega)$, $z_{LR(\omega)}^k$ in the summatory z_{LR}^k . Hence, in this point, it is premised on the presumption of Eq. 5.

$$z_{LR}^k = \sum_{\omega=1}^{\Omega} z_{LR(\omega)}^k \tag{5}$$

In a reduce operation, a private copy for each variable is created for each core. At the end, the reduction operation (sum) is applied to all private copies of the shared variable, and the final result is written in the global shared variable. Next, **pSM** computes all the subgradients, S^k , taking into account the solution of the first stage variables $y_i^{\omega k}$, the computation is performed using Algorithm 1.

The pseudo code of Algorithms 1 and 2 uses the notation of OpenMP to represent the

Algorithm 1 Compute S^k

```

1: #pragma omp for
2: for all  $\omega \in \Omega$  do
3:   for all  $i \in y^{(k)\omega}$  do
4:     if  $\omega = 1$  then
5:        $S_{\Omega}^k(i) = y^{(k)\Omega}(i) - y^{(k)1}(i)$ 
6:     else
7:        $S_{\Omega}^k(i) = y^{(k)\omega}(i) - y^{(k)\omega+1}(i)$ 
8:     end if
9:   end for
10: end for
    
```

$$S^k = \begin{pmatrix} y^{(k)1} - y^{(k)2} \\ \vdots \\ y^{(k)\Omega-1} - y^{(k)\Omega} \\ y^{(k)\Omega} - y^{(k)1} \end{pmatrix}$$

shared-memory parallel approach.

There is overwhelming evidence that two threads are accessing concurrently to the same memory region, because not only the values of $y^{(k)\omega}$ in the first stage belonging to a specific thread are needed to compute the Subgradient, but also the values of the solution of the following thread, $y^{(k)\omega+1}$. The basic premises of parallel shared memory paradigm is that the same memory region can be read concurrently for multiple threads if and only if no one writes on this memory region. Thus, the algorithm does not end in any memory exception.

Once all the subgradients are computed, **pSM** needs to evaluate in a single thread the status of the algorithm at iteration k using the current global solution. This process consists on checking whether the algorithm is able to improve the solution in future iterations. The criteria used for taking this decision are deeply explained in Escudero and Garín (1984). After this phase, in case that the algorithm improve the current solution, some convergence parameters are updated so as to boost the ending of the method. For detailed information about the choice and the improvement of these parameters, check Escudero and Garín (1984) too.

Following these serial steps, **pSM** updates the matrix μ^{k+1} this procedure is described in Algorithm (2).

Algorithm 2 Compute μ^k

```

1: #pragma omp for
2: for all  $\omega \in \Omega$  do
3:   for all  $i \in y^{(k)\omega}$  do
4:      $\mu^{k+1}(i) = \mu^{(k)\Omega}(i) + (b \cdot S^{(k)\Omega}(i))$ 
5:   end for
6: end for
    
```

$$\mu^{k+1} = \mu^k + \alpha^k \cdot b \cdot S^k$$

$$b = \frac{\bar{z}_{LR} - \underline{z}_{LR}}{\|S^k\|_2^2}$$

There is no need to argue about the correctness of Algorithm (2). Note that b is fixed for all threads. Thus, this value is computed by a single thread and stored in the shared memory in order to be accessible for all threads. Moreover, this value is computed in the previous step, just before checking the stopping criteria. \bar{z}_{LR} represents an upper bound of the solution value of $S2FVPP$, see Problem (1).

Once the matrix μ^{k+1} is updated for all threads, **pSM** goes to the next iteration, $k = k + 1$.

The current literature on Lagrangian decomposition abounds with different examples of small modifications to improve the behaviour and the convergence of the method. The implementation of **pSM** takes advantage of the introduction of the scenario cluster concept. The proposed method is able to deal with scenario clusters, see Escudero et al. (2004).

Figure 2 shows the proposed **pSM** scheme. This scheme summarises the iterative process and highlights the steps realised in parallel by all available threads and the ones realised in serial by a single thread.

Case study

In this section, a real study for $S2FVPP$ instance is studied. A local chain of grocery shops is dealing with the problem of supplying tomatoes, grown by locally producers, at the minimum cost for the next year. Thus, the aim is to determine which tomato farms have to be contracted this season to satisfy future demand with local products.

Moreover, the local chain of grocery shops is made up of eight shops {C1–C8} and has to select tomato farms {A–J} to fulfil the future demand. These shops and tomato farms are distributed throughout Catalonia, see Fig. 3. This map represent the location of each shop and farm, approximately.

The main characteristics of the tomato farms used in this study are summarised in Table 2. *Farm* represents the name of the farm, *Location* indicates the place and *Hectares* the land surface. Quality field ranges from 1 to 10, where 1 represent the lowest quality. This index is computed using both the knowledge of historical data for past seasons and customers feedback information. Note that in Table 2, farms {A, D} has not quality index (–), representing that neither historical data nor customers feedback information are recorded for these farms. Besides, each *cgv* coefficient is computed taking into account the quality index and the size of the yields.

Scenarios are built considering production, demand and cost uncertainty of servicing each shop. By this way, 3 different scenarios are generated: poor, fair and boom with probability 0.22, 0.70 and 0.08 respectively. Table 3 shows the variation of the demand at each shop under the 3 scenarios considered in this study. Besides, the yield per hectare is

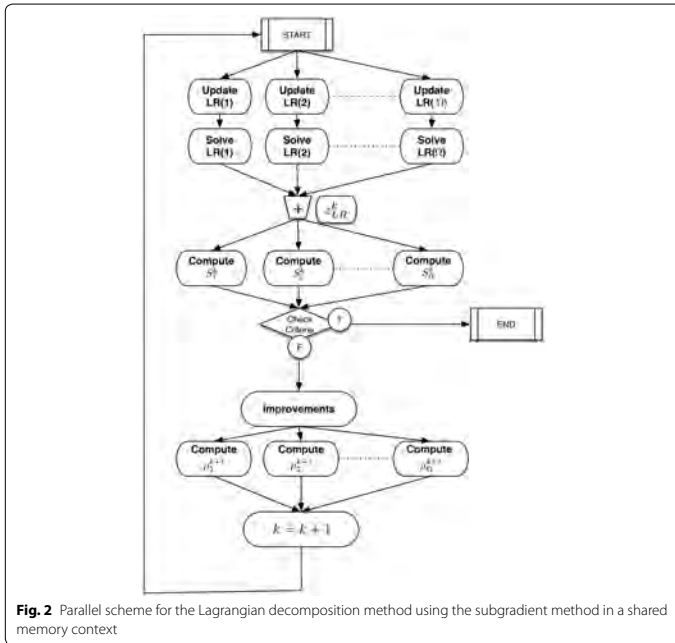


Fig. 2 Parallel scheme for the Lagrangian decomposition method using the subgradient method in a shared memory context

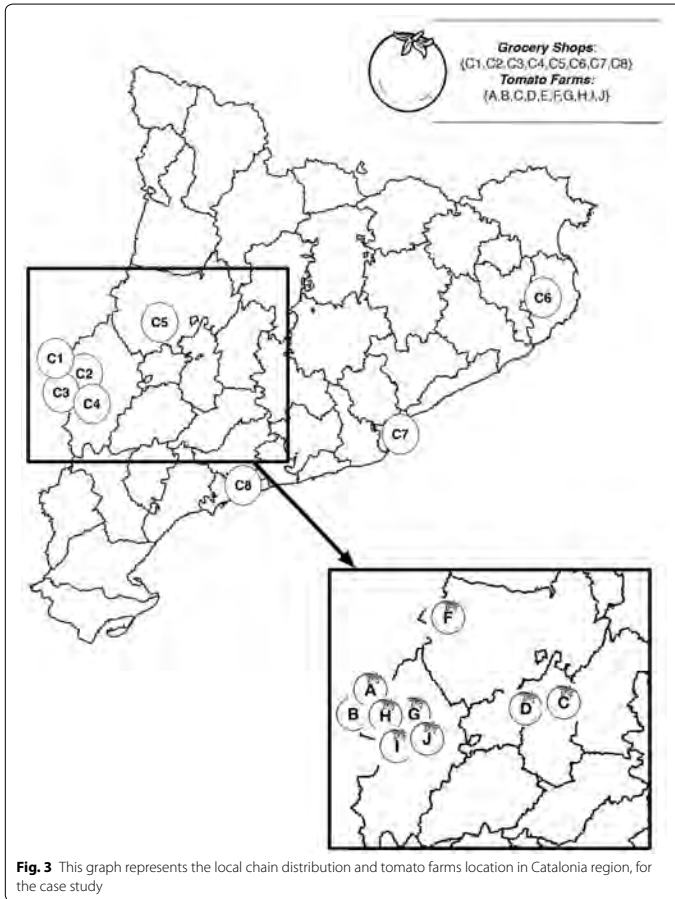
assumed to be fixed. The scenario values considered are 50,000, 72,000 and 80,000 kg/ha for poor, fair and boom respectively.

The main features of the grocery shops used in this study are summarised in Table 4. *Grocery Shop* represents the name of the grocery shop and *Location* indicates the place. The *Serving Cost; csv*, is computed using information related with the transport and delivery costs estimated for each pair of tomato farm and grocery shop. Table 4 describes the cost of service under the fair scenario.

Figure 4 shows the results and tactical decisions of the proposed model. These results show the farms selected to be used the following season. The selection of these farms minimizes the overall cost into 132.973. The proposed model is shown to deal with that kind of problem efficiently.

Practical implications

The S2FVPP model presented in this paper is aimed at practical application for the fresh vegetable agri-food industry. There is overwhelming evidence corroborating the idea that industry needs to understand the model before rely on it. Choosing the best farms to contract production is a critical decision in order to design an efficient model to select suppliers and explore alternative solutions by purchase managers. This position allow the manager to gain knowledge about the range of prices he can fix on contracts considering



the uncertainty represented by each scenario. Moreover, decisions makers are able to use the huge amount of big data gathered from their industrial context to feed the model with accurate coefficients for csv and cgv and enlarging the number of scenarios to be considered. Historical data sets for past seasons, consumers feedback, traceability, transport cost, among others, can be used to model these input parameters. Furthermore, the stochasticity of the model introduces the market uncertainty and makes possible to improve predictions about the future trends so as to make better decisions in the whole agri-food business context.

Given the interest of selling local fresh vegetables, the model helps purchase manager to choose the best fresh vegetable farms to sign production contracts and hence fulfil

Table 2 Summary stored in the data center with the main information for available tomato farms

Farm	Location	Hectares	Quality	cgv (0.09 €/kg)
A	Torreserona	0.005	–	9
B	Torreserona	0.008	8	8
C	Tàrrrega	0.008	8	8
D	Anglesola	0.012	–	9
E	Mollerusa	0.016	7	18
F	Alfaràs	0.012	9	12
G	Torrefarrera	0.018	7	18
H	Alpicat	0.025	7	25
I	Alpicat	0.003	8	5
J	Torrefarrera	0.025	6	30

Table 3 Information about the demand under each specific scenario

Name	Demand (kg)		
	Poor	Fair	Boom
C1	350	400	450
C2	200	300	350
C3	275	300	325
C4	250	325	400
C5	75	150	200
C6	150	250	400
C7	400	650	800
C8	250	300	350

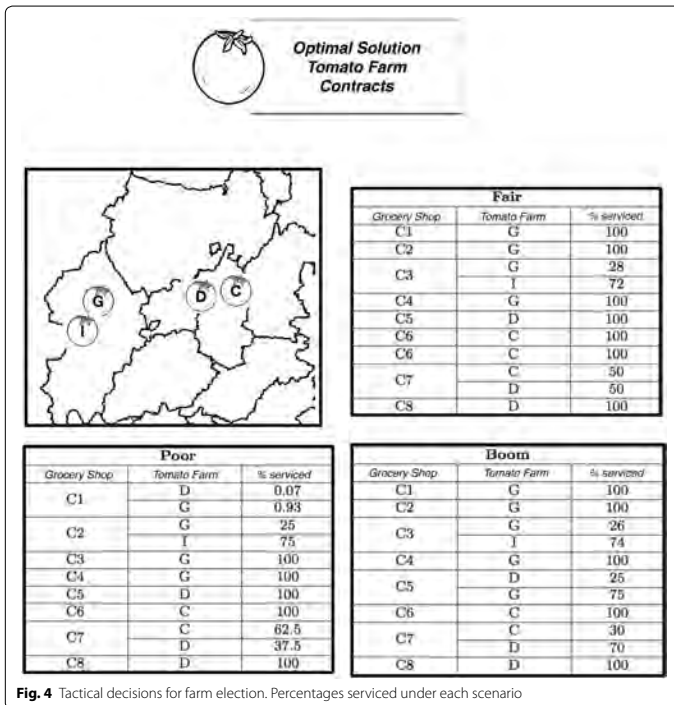
Table 4 Main information about the grocery shops

Grocery shop	Location	Serving cost per unit (0.09 €/kg)									
		A	B	C	D	E	F	G	H	I	J
C1	Lleida	5	5	10	10	7	10	6	6	6	6
C2	Lleida	5	5	10	10	7	10	6	6	6	6
C3	Lleida	6	6	11	11	8	11	7	7	7	7
C4	Lleida	5	5	10	10	6	10	6	6	6	6
C5	Balaguer	7	7	6	6	8	4	7	7	7	7
C6	Girona	30	30	20	22	25	37	28	28	28	28
C7	Barcelona	25	25	15	18	20	30	24	24	24	24
C8	Tarragona	16	16	10	11	20	25	16	16	16	16

Highlighting the serving cost csv under fair scenario per unit (0.09 €/kg) of demand

final consumers demand. By the analysis of the optimal solution, the industry can negotiate production contracts with the more suitable farms. In contrast, purchase managers without the model have to trust only on their expertise in the field in order to decide whether or not selecting a farm as supplier.

The model presented is a powerful tool capable of dealing with huge amount of information and a number of scenarios coping with the uncertainty of future fresh vegetable



production to make the best decision. Much of the current solutions deals with the minimization of the global cost of choosing the most suitable neighbouring farms to supply products. However, the last but no the least, the model not only helps to minimise this cost but also helps purchase managers to reduce work time and effort. The time freed by the model allows managers to develop other activities or exploring different alternatives making them more efficient in their daily job.

Computational performance

In this section, the computational experiments to assess the behaviour of the proposed model is presented. The algorithm to solve the model was coded in C++ using the OpenMP library (2016), the Eigen C++ library (2016) and OPL, CPLEX 12.6 C++ API (2015). Moreover, tests were conducted on a virtual scientific computing platform of the University of Lleida, known as Stormy. More information about this infrastructure can be found in Stormy (2016). The virtual machine chosen to develop the experiments was configured with 10 CPU of 2 cores each one, 25 GB of RAM and also 100 GB of HDD. The operating system used was Ubuntu 13.04.

Table 5 Testbench

Instance	#Farms	#Grocery shops	Ω
S1	30	70	20
S2	30	70	50
S3	30	70	80
S4	30	70	100
S5	30	70	200
S6	30	70	300

Main configuration parameters

A collection of benchmark instances, considering different possible scenarios was generated. A set of 30 farms and a set of 70 grocery shops are considered, so $F = 30$ and $C = 70$. The tests go from a small set of stochastic scenarios, where $\Omega = 20$ to a huge set of them, where $\Omega = 300$. Moreover, the uncertain scenarios were generated taking into account different combinations of financial and economic situations in the production and demand. Table 5 summarise the size and configuration of the whole instances. The main parameters are #Farms, number of available farms; #Grocery shops, number of customers; and Ω , number of stochastic scenarios.

Table 6 shows the dimensions of the instances in the compact and splitting variable representations. This table extends the information presented in Table 5, showing the real complexity of the problems solved. The heading are as follows: m , number of constraints; n_y , number of 0–1 first stage variables; n_x , number of continuous second stage variables; Ω , number of scenarios.

There is overwhelming evidence that **pSM** can be configured using a huge range of parameters. However, this was not the purpose of this paper. The results presented were computed using the parameters described in Table 7.

Table 8 shows the results obtained using as solver CPLEX with automatic setting and using all the available CPU's inside the machine, solving the original S2FVPP in both the compact and splitting variable representation. The results of applying the parallel method **pSM** presented in this paper are highlighted too. The headings are as follows: T_{CR} , CPLEX elapsed time (seconds) for obtaining the optimal solution for S2FVPP in compact representation; T_{SV} , CPLEX elapsed time (seconds) for obtaining the optimal solution for 2SVPP in splitting variable representation; T_{pSM} elapsed time (seconds) for obtaining the optimal solution using the **pSM** proposed in this paper; Sp_{pSM-CR} , the

Table 6 Model dimensions

Instance	Compact representation			Splitting variable representation			Ω
	m	n_y	n_x	m	n_y	n_x	
S1	44,020	30	42,000	44,620	600	42,000	20
S2	110,050	30	105,000	111,550	1500	105,000	50
S3	176,080	30	168,000	178,480	2400	168,000	80
S4	220,100	30	210,000	223,100	3000	210,000	100
S5	440,200	30	420,000	446,200	6000	420,000	200
S6	660,300	30	630,000	669,300	9000	630,000	300

Size of instance sets

Table 7 Initial parameters for calibrating the pSM method

Instance	μ_0	α_0	Norm	G	# cores
S1	0	0.1	2	1	20
S2	0	0.9	2	2	20
S3	0	0.9	2	4	20
S4	0	0.001	2	5	20
S5	0	0.001	2	10	20
S6	0	0.001	2	15	20

Norm norm type, G cluster size

Table 8 Performance of parallel Lagrangian decomposition solving the relaxed splitting variable model over the resolution of the equivalent models using CPLEX solver

Instance	T_{CR}	T_{SV}	T_{pSM}	SP_{pSM-CR}	SP_{pSM-SV}
S1	58.7	60.9	49	1.19	1.24
S2	378	386	156.74	2.41	2.46
S3	3051	3522	1419	2.15	2.48
S4	4397	4844	3568	1.23	1.43
S5	62,266	63,000	12,348	5.04	5.1

improvement in speed in the execution of the parallel method compared with the execution using the commercial solver CPLEX, solving the compact representation model; and SP_{pSM-SV} , the improvement in speed in the execution of the parallel method compared with the execution using the commercial solver CPLEX, solving the splitting variable representation model.

The results presented show the strengths and weakness of using a commercial solver such as CPLEX, compared with the usage of the parallel method proposed in this paper. The use of pSM only depends on the model size. The results provide confirmatory evidence that the method proposed is very suitable to deal with these kind of problems. These results highlight the goodness of applying parallel decomposition techniques instead of using commercial solution to deal with full stochastic models.

Small instances, such as S1 and S2 are very far from real-life problems. Whereas, the bigger the problem is, the closer to the real problems. The analysis of the biggest instances show a huge reduction in computing time, and proves the applicability and efficiency of the algorithm for dealing with the resolution of the model with real-life data. On logical grounds, there is no compelling reason to argue that instances S3, S4, or S5 are more suitable to be solved by the method proposed. The main advantage of the method proposed is the considerable improvement in computing performance, taking the computing time as a metric to compare the proposed method and CPLEX solver. Finally, the method seems to be very scalable, because the bigger the problem is, the more speed up reached.

The initial upper bound of the solution value of the original problem is obtained by using an intuitive heuristic. This heuristic obtained the greatest possible feasible solution by fixing to 1 all the values of the boolean 0–1 first stage variables and then solving the problem. To portray the issue in farms terms, the basic idea is solving the model by choosing all the farms. By this way, it is possible to obtain a feasible upper bound

in a few seconds. The results presented show that this upper bound is good enough to achieve competitive results.

On the other hand, the results show that the solution of the splitting variable representation takes much more time than the compact representation for solving the model. Therefore more computation effort is needed to obtain the Lagrangian multipliers vector (μ_0) in the case of being initialized as the dual variables of the non-anticipativity constraints. This vector is initialized to zero in order to saving time by avoiding the calibration phase of the method.

Finally, these results boost the usage of the model by purchase managers in the agroindustry because models are solved in a reasonable computing time.

Conclusions

This work is focused on the design and resolution of a model to deal with the selection of suppliers for a chain of grocery shops. The objective of this selection is to contract the production of fresh vegetables. Hence the grocery shop can offer local products and have a better position to control quality and traceability. The model takes into account the future demand of a set of customers and the uncertain production of a farm during a season. The objective was to develop a two stage mixed 0–1 model considering uncertain production and demand. Thus, the model presented seems to be a good approach for solving this kind of problems. Moreover, the good behaviour of parallel Lagrange Decomposition for resolving the model shows its applicability to the real-life problems. The integration of the model in the software of purchase managers give them competitive advantage when contracting production.

The proposed model is very practical and flexible and it will be very easy to adapt to other contexts with the same necessities.

In the future, this model will be transformed into a fully supply chain model considering all the uncertain costs of production, transportation, storage and delivery so as to make a decision model capable of taking tactical and strategic decisions for the full vegetable supply chain.

Finally, the parallel algorithm presented in this paper can be improved using other methods to solve the Lagrangian relaxation, as a cutting plane algorithm, the progressive hedging algorithm, or the parallel combination of each of these.

Authors' contributions

JM designed the model, developed the testbench and drafted the manuscript. LMP checked the correctness of the model, assisted in the development and solution of the fresh vegetable SC, provided guidance and also improved the manuscript. FS and AP collected and reviewed the literature, provide guidance and also improved the manuscript. All authors read and approved the final manuscript.

Author details

¹Computer Science Department and INSPIRES, University of Lleida, Jaume II 69, 25001 Lleida, Spain. ²Department of Mathematics, University of Lleida, Jaume II 73, 25001 Lleida, Spain.

Acknowledgements

This work was supported by the MEyC under contracts TIN2011-28689-C02-02, TRA2013-48180-C3-P and TIN2014-53234-C2-2-R. The authors are members of the research group 2014-SGR163 and 2014-SGR151, funded by the Generalitat de Catalunya.

Competing interests

The authors declare that they have no competing interests.

Received: 7 January 2016 Accepted: 10 June 2016

Published online: 22 June 2016

References

- Ahumada O, Villalobos JR (2009a) A tactical model for planning the production and distribution of fresh produce. *Ann Oper Res* 190(1):339–358
- Ahumada O, Villalobos JR (2009b) Application of planning models in the agri-food supply chain: a review. *Eur J Oper Res* 196(1):1–20
- Al-Sultan KS, Al-Fawzan MA (1999) A tabu search approach to the uncapacitated facility location problem. *Ann Oper Res* 86:91–103
- Avella P, Boccia M, Sforza A, Vasiliev I (2009) An effective heuristic for large-scale capacitated facility location problems. *J Heuristics* 15(6):597–615
- Beasley JE (1993) Lagrangean heuristics for location problems. *Eur J Oper Res* 65(3):383–399
- Birge JR, Louveaux F (2011) Introduction to stochastic programming. Springer, Berlin
- Cornuéjols G, Nemhauser GL, Wolsey LA (1983) The uncapacitated facility location problem (No. MSRR-493). Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group
- de Camargo RS, Miranda G, Luna HP (2008) Benders decomposition for the uncapacitated multiple allocation hub location problem. *Comput Oper Res* 35(4):1047–1064
- Eigen 3.2.6 API C+++. <http://eigen.tuxfamily.org/>. Accessed: 31 April 2016
- Erlenkotter D (1978) A dual-based procedure for uncapacitated facility location. *Oper Res* 26(6):992–1009
- Escudero LF, Garín MA, Pérez G, Unzueta A (2011) Lagrangian decomposition for large-scale two-stage stochastic mixed 0–1 problems. *TOP* 20(2):347–374
- Escudero LF, Garín MA, Pérez G, Unzueta A (2013) Scenario cluster decomposition of the Lagrangian dual in two-stage stochastic mixed 0–1 optimization. *Comput Oper Res* 40(1):362–377
- Fisher ML (2004) The Lagrangian relaxation method for solving integer programming problems. *Manag Sci* 50(12 Supplement):1861–1871
- Ghosh D (2003) Neighborhood search heuristics for the uncapacitated facility location problem. *Eur J Oper Res* 150(1):150–162
- ILoG Cplex 12.6 API C+++. <http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud/>. Accessed: 05 Jan 2015
- OpenMP. <http://openmp.org/>. Accessed: 31 April 2016
- Plà LM, Sandars DL, Higgins AJ (2013) A perspective on operational research prospects for agriculture. *J Oper Res Soc* 65(7):1078–1089
- Plà-Aragónes LM (ed) (2015) Handbook of operations research in agriculture and the agri-food industry, International Series in Operations Research & Management, vol 224. Springer, New York
- Prékopa A (2013) Stochastic programming. Springer, Berlin
- Ro H, Tcha D (1984) A branch and bound algorithm for the two-level uncapacitated facility location problem with some side constraints. *Eur J Oper Res* 18(3):349–358
- Shapiro A, Dentcheva D, Ruszczyński A (2014) Lectures on stochastic programming: modeling and theory, 2nd edn. SIAM, Philadelphia
- Soto-Silva WE, Nadal-Roig E, González-Araya MC, Plà-Aragónes LM (2016) Operational research models applied to the fresh fruit supply chain. *Eur J Oper Res* 251(2):345–355
- Stormy. <http://stormy.udl.cat>. Accessed: 28 May 2016
- Wu T, Yang Z, Chu F, Zhou Z (2015) A Lagrangean relaxation approach for a two-stage capacitated facility location problem with choice of depot size. In: 2015 IEEE 12th International conference on networking, sensing and control. IEEE, pp 39–44

Submit your manuscript to a SpringerOpen® journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com

Authors: *Jordi Mateo, Francesc Solsona, Lluís M. Plà and Adela Pagès*

Journal: Cluster Computing

Publisher: Springer Nature

Year: 2018

DOI: <https://doi.org/10.1007/s10586-018-2878-4>

ISSN: 1573-7543

Keywords: *Stochastic linear optimization, Parallelization, Scalability, Clustering in stochastic optimization, Benders Decomposition*

A scalable parallel implementation of the Cluster Benders Decomposition algorithm.

Abstract: Benders Decomposition (BD) is a method used to solve stochastic linear problems via scenario analysis. Cluster BD (CBD) is one of its smart improvements that speed up the execution time, taking advantage of tighter feasible cuts found by grouping scenarios into clusters. In this paper, we propose a new design for CBD, one which takes into account the role played by optimal cuts in the solution. Besides, we propose a new parallel scheme for CBD to deal with large-scale two-stage stochastic linear problems. Moreover, we characterise the problems for which our proposal performs best. The results obtained show computational gains from our proposal compared with the plain use of CPLEX, serial BD, parallel BD, serial CBD and parallel CBD.

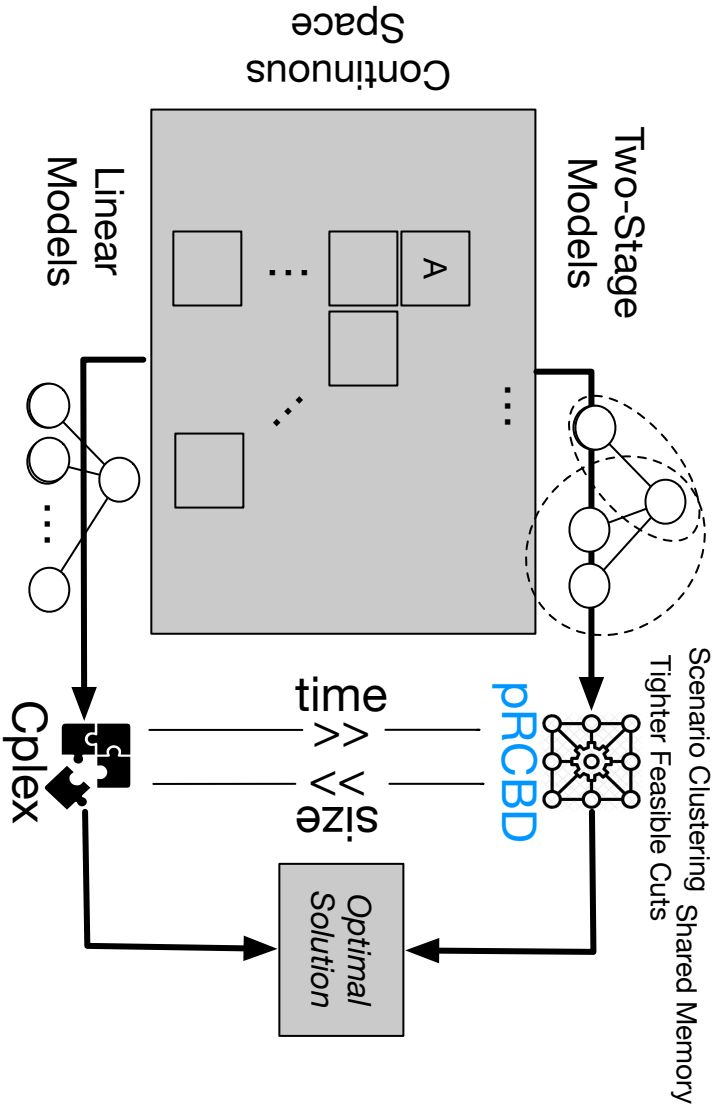


FIGURE 4.1.: Graphical Abstract.

Authors: *Jordi Mateo, Kevin Borrell, Francesc Solsona, Lluís M. Plà, Adela Pagès and Jordi Vilaplana*

Journal: The International Journal on Software Tools for Technology Transfer

Status: Under Review

Keywords: *cloud computing, SAAS, optimization*

SPOS, a new cloud-based service for solving optimization models

Abstract: Optimisation models are being more extensively used to tackle real-life problems. This way, better decisions can be taken to reduce risks and cost or increase profits. Large companies are improving their economic performance by investing in business analytics or operations research. However, small to medium enterprises (SME) are unable to afford this kind of investment. This paper proposes a new software as a service (SAAS) to bring the huge potential and benefits of optimisation to the daily activity of SMEs, the academic community and the public in general. The service described in this article allows optimisation models to be solved through a simple step-by-step user interface. Thus, the only requirement for executing a model and analysing its results is an electronic device connected to the Internet. The results obtained show the usability and competitive advantages of using the proposed service for decisions makers and academic users. The service presented is available at stormy02.udl.cat/spos.

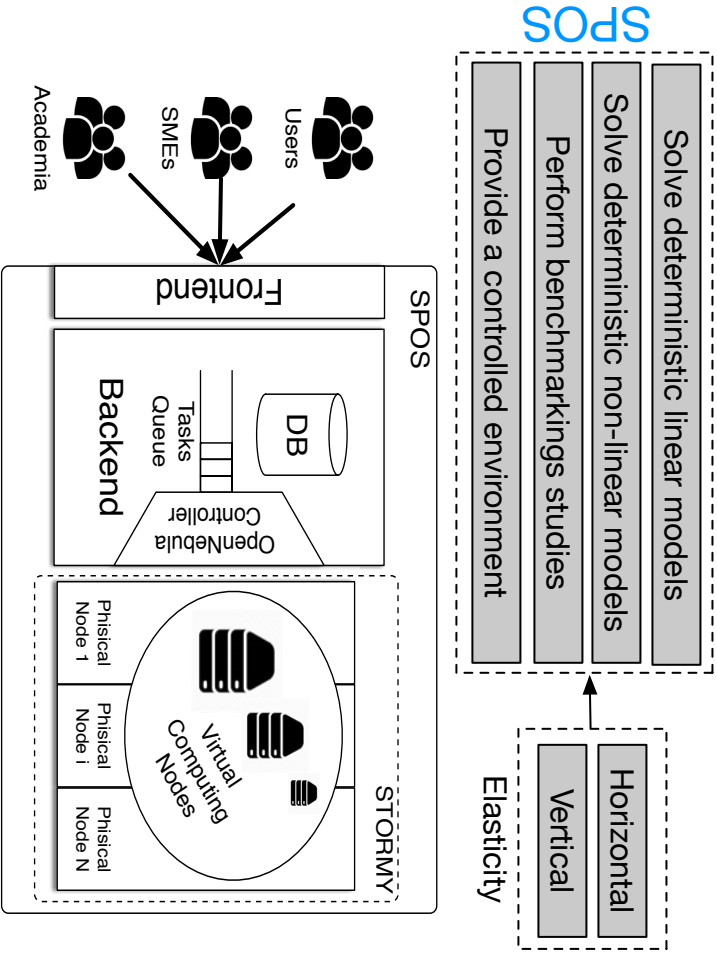


FIGURE 5.1.: Graphical Abstract.

5.1 INTRODUCTION

In the words of (Leonhard Euler, 1744) "Nothing in the world takes place without optimisation and there is no doubt that all aspects of the world that have a rational basis can be explained by optimisation methods". Globally, thousands of new products are continually coming into production, hundreds of companies are being created, a massive number of business agreements are reached and more. Commercial and business activity around the world is very frenetic and intense, with many companies fighting each other to dominate one or more commercial sectors, where each decision can enable the company to succeed or lead it to ruin. Optimisation, simulation, and big data technology are widely used to tackle these critical decisions where maximising profit or reducing risk are increasingly the keystones for every business development plan.

Linear and non-linear programming have proven to be important tools for modelling and solving real-life problems. A real-life problem is translated into a mathematical model where the primary goal is to maximise or minimise an objective function within a set of numerical constraints. The constraints are used to represent the practical bounds of the problem whereas the variables are used to model the decisions to be made. A closer look into the decision-making process indicates that different types of variables are needed to model different decisions. For example, a Boolean variable will be used to decide whether to buy a product from a specific producer or not, whereas a continuous variable is used to determine the amount of raw material that needs to be purchased.

There are a huge number of problems that can be solved or simplified with the help of optimization models. Problems related to logistics [74–77], supply chain management [78, 79], production planning [80], or portfolio optimization [81, 82], are the most common among these. The current literature on Operations Research abounds with examples of successful applications. Optimising transportation networks to set vehicle routes to perform transportation at minimum cost is described in [83]. The authors in [84] propose a model capable of reducing costs and pollution in the iron and steel industry. In 2014, an article published in the American Society of Animal Science journal [85] described how they created a linear programming model to help a company located in Catalonia to take two type of decisions: short-term decisions, like scheduling the weekly transports of pigs between farms, or between farms and the abattoir, and mid-term or long-term decisions, like enlarging or shrinking the company, increasing or decreasing the capacity of its farms and others. Another successful application was presented in [86] where the optimisation of coal mining planning is useful for choosing the best production strategy to meet market demand. Moreover,

in [87], a model for deciding the necessary maintenance crew to cover all the maintenance needs with the lowest possible cost, taking data like the budget or duration of maintenance cycles as inputs, was proposed. An article published in the *Journal of Cleaner Production* explained how linear programming helped a real refinery company by optimising their hydrogen network [88]. Moreover, lastly, another successful application is presented in [89], where linear programming helps to reduce the outlay of a company and to preserve the environment.

Software-as-a-Service (SAAS) and cloud computing are becoming powerful tools for business. Cloud providers claim that cloud is elastic and is aimed to control costs allowing resources as pay-as-you-go and pay-per-use. Cloud elasticity is a vital property to improve the quality of the services. Cloud providers can add or remove features without interruption to adapt load and traffic variation on demand. For cloud users, the available resources seem to be unlimited, see [90] and [91]. This way, the cloud is ready to fit unpredictable demand and planned business growth. Furthermore, clouds can ensure the availability, reliability and performance of applications and resources. These features make clouds and SAAS attractive to large companies, but more attractive to SMEs.

Large companies can afford to have a specialised team to create mathematical models for different problems, run these problems in powerful solvers and computers and make an exhaustive analysis of the results. However, for people who have a SME or want to use these tools for personal reasons, the situation is somewhat more complicated, because although various free solvers are available, preparing and configuring a dedicated machine to run these solver executions is not possible for everyone, either because they lack the necessary resources or because the necessary knowledge to prepare a computer or install and use these solvers is outwith their grasp.

Combining cloud computing and software-as-a-service (SAAS) and mathematical programming generates an incredibly powerful alliance. Cloud computing and SAAS eliminate the requirement for a powerful computer with an environment dedicated to solving complex optimisation models. The only thing needed is an electronic device connected to the Internet. With only a web browser, the user obtains access to almost unlimited computing power, no matter which device is used (hand held device, desktop computer or laptop).

Some free services are already available on the Internet, for example, the NEOS Server [92]. The latter combines the power of optimisation and remote computing, in this case using high-performance distributed machines, to bring the huge potential of optimisation to all kinds of user, allowing to launch executions, where you can choose from a wide range of solvers and obtain the results once the execution has finished. However, this service has one limitation to run

academical experiments because it does not give monitoring results and also it is difficult to compare different executions in a controlled environment. There are also commercial services, like IBM Decision Optimization on Cloud, powered by the Bluemix platform [93], that offer to solve optimisation problems in the cloud taking advantage of cloud elasticity. Therefore, this service allows users to run optimisation problems in the cloud through a modern and easy-to-use user interface and also offers the possibility of customising the machine where the solver will be executed. However, this service is limited to *Cplex* and *Opl* compatible models.

To address the main limitations of these services, and to keep the main advantages of them, this paper makes the following contributions:

1. Model an elastic cloud architecture to offer optimisation as a service.
2. Provide a service to solve optimisation problems in the cloud automatically.
3. Implement a service that grants access to the most used open source linear and non-linear solvers.
4. Provide a service that monitors the use of computational resources during the executions.
5. Provide tools to customize virtual machines to perform experiments in a controlled environment.

The application presented in this article, known as SPOS, goes one step further into Optimization-As-A-Service. SPOS mix the best points of the two services explained above, offering not only the possibility of launching executions on the cloud with different solvers, but also allowing the user to customise the virtual machine where the solver will be executed. Users can choose the number of CPUs or the amount of memory manually. Moreover, it also aims to keep this process as simple as possible for those users who only want to see the results of their problem quickly, so it has a Simple mode with a smooth and straightforward step-by-step process where the user only has to introduce some necessary information to launch the execution, letting the system decide the best virtual machine configuration to solve the problem. This “Keep it simple” principle also applies to the results of execution, so the application shows the user a summarised version of the results with only the critical information, while still offering the full solver output for advanced users.

SPOS is also designed to be useful for advanced users interested in the performance of their executions. Accordingly, this application does allow you not only to customise the virtual machine used in the execution but also displays

performance graphics for massive executions, which show the use of memory and CPU during execution. This is another important characteristic that makes the service unique and different from NEOS. Hence, in summary, the service presented is a combination of NEOS and cloud providers, which takes advantage of cloud elasticity to adapt dynamically to the service traffic and demand. Furthermore, it gives the users the opportunity to customise their virtual machines and obtain a controlled environment to realise experiments. Finally, the service extends the solver results with monitoring information. These features make the service unique and very useful to perform academic benchmarks, experiments and studies.

The major contribution of this work is the design and implementation of a new SAAS optimisation service that combines the potential of some of the most popular open-source solvers with the capabilities of cloud computing. Furthermore, the service is implemented in a user-friendly way to be used by the entire community. Finally, the service presented is a highly valuable tool for researchers not only for resolving models but also for monitoring computational resource capabilities. This feature complements the information about the resolution of the model helping the user to a better understanding of the complexity and the resources required.

5.2 THE CLOUD SERVICE

The platform chosen to develop this architecture was OpenNebula, see [94]. The main reason for choosing this was to take advantage of the Stormy server. This server is a private cloud deployed with the OpenNebula platform that belongs to the University of Lleida. See [2] for more information.

On logical grounds, this service can be replicated or extended using other cloud platforms such as Amazon Web Services or Microsoft Azure. The authors choose Stormy because is a platform supported by public funding and it is aimed to improve research and to transfer the knowledge from the academia to the society. The architecture of SPOS service is composed of three parts: the frontend, the backend and the computing nodes. Figure 5.2 depicts each part inside the platform. Moreover, all these parts were deployed using Centos7 images as the OS.

The frontend is the part with which the user interacts with. This part is executed on the users' device. The service presented is responsive and can be adapted to multiple devices. It contains the website and is responsible for collecting all the user input data, forwarding data and tasks to the backend and displaying the responses and results.

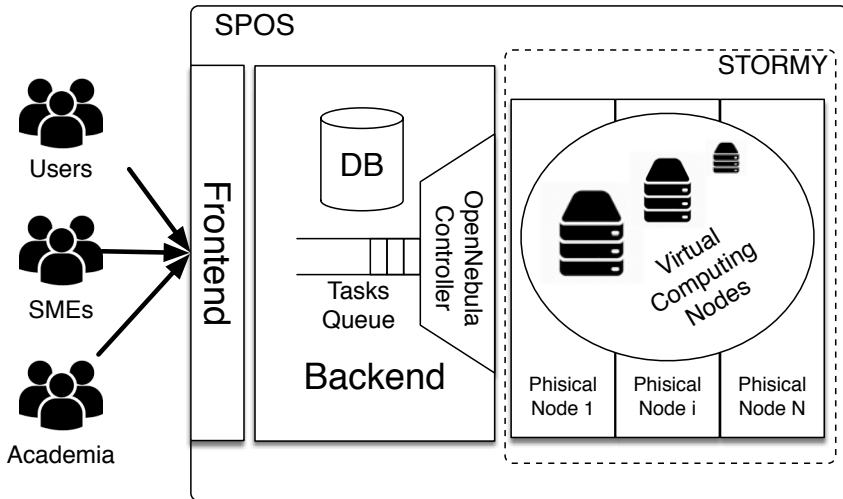


FIGURE 5.2: Main parts of the cloud service (SAAS).

The backend is the core of the application. The main features are to manage the user's data (input and output), to handle and maintain the virtual resources of Stormy (computing nodes) and to launch executions in the computing nodes. It contains a relational database to store user's data, a task queue to keep alive all the unhandled tasks and an OpenNebula controller to manage the computing nodes on demand.

Finally, the computing nodes are the parts where the mathematical models are solved. They receive the solver input information and execute the specified solver. When execution ends, the results are sent to the backend to be stored at the database. These computing nodes are virtual machines deployed in Stormy that are dynamically created on demand.

The main feature of this architecture is the elasticity. Cloud elasticity is the ability of a system to add and remove resources dynamically to adapt to real-time variation of the load and traffic. In cloud computing, there are two kinds of elasticity, horizontal and vertical. Horizontal elasticity consists of instantiating volatile resources that are created and destroyed on demand and associated with an application. Vertical elasticity deals with increasing or decreasing the computing power of specific instances. Figure 5.3 depicts this principle.

SPOS service takes advantage of elasticity to instantiate a virtual machine (computing node) of specific power to solve an optimisation task and also to

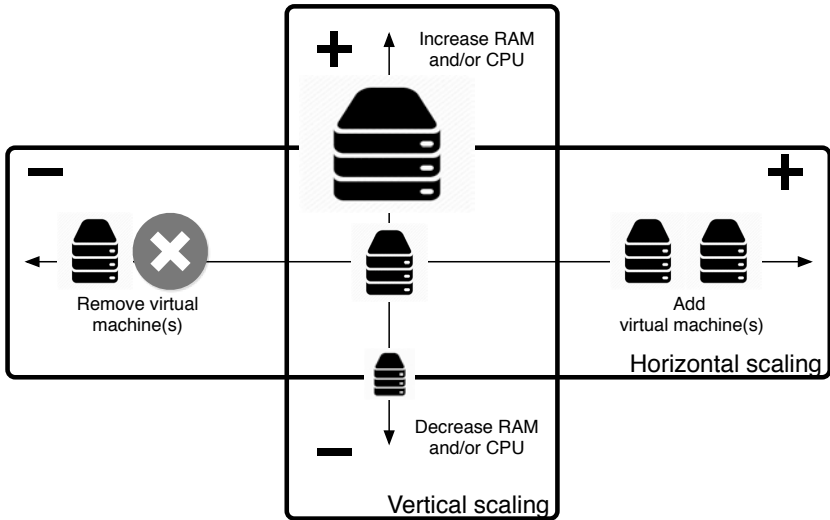


FIGURE 5.3: Elasticity of cloud services.

delete this virtual machine once the execution is finished to release the associated resources. Figure 5.4 is used as an example of these features. Let us consider the situation where two incoming tasks arrive at the system (T1, T2), and then the system needs to instantiate two virtual machines (A, B), handling the unused resources of Stormy assigned to SPOS. In particular, computing node B is more powerful than computer node A. This is indicated using the size of the server and also the size of the partition. After a while, T2 is completed. Then, the system disassociates all the virtual resources of node B to be reused again to serve other tasks.

Vertical elasticity or the capacity to resizing instances in real time can lead to leverage as best as possible the resources at a given time. Elasticity is often confused with scalability. Scalability is the ability of a system to add more resources to meet a more significant workload, while elasticity consists of adapting the resources in a given time. In this context, elasticity is built on top of scalability. It means that the system can scale by increasing the application quota or just by adding more physical nodes to the architecture while preserving the elasticity advantages.

The system presented deals with errors and failures in a simple way. The central premise is that when an error or failure happens in a session, the user receives the proper feedback and the session must be checked and recreated

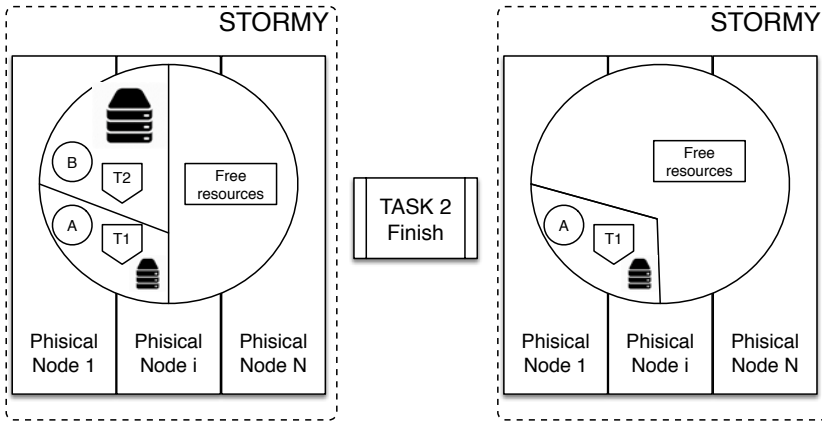


FIGURE 5.4: SPOS. Elasticity example.

again. The administrator of the system handles the internal service failures such as the errors in the communication between the virtual instances or the errors in the instantiation of virtual machines. When the administrator receives a notification of the error, he solves it and then the affected tasks automatically start again. This process is entirely transparent to the users. In contrast, where an error causes failures in the solving step, such as a false problem, or an error triggered by the solver, the users are notified with proper feedback, and the session must be corrected and recreated again manually by them. Moreover, the system keeps track of all the transaction in a logging system.

Finally, another important feature of the full system is security. The virtual machines are only accessible using the cloud service (through the client) or with a VPN connection. The virtual machine that acts as the client (frontend) is an exception, and it is the open the door to the virtual resources. The client is the middleman, responsible for managing the remote connection, and it uses a firewall that denies all connections except the ones started from the application.

5.2.1 Frontend

The frontend or the client is a web application accessible from any modern device connected to the Internet. The users can create new sessions, upload their

problem files with the models and data, check the progress of their executions and see the results.

The user interface of the client makes the process of creating a session for a solver execution as easy as possible, with a usable and short step-by-step process, with help texts that explain what the user should do at each step.

The client is responsible for gathering all the necessary data from the user to launch a solver execution in the cloud. When creating a new session, users can choose between using an automatic configuration or customising the resources by introducing the amount of memory and the number of real and virtual CPUs. The next step is to choose the type of model to solve. Currently, it only allows linear and non-linear deterministic models.

Once the type of model has been selected, the application lists all available solvers for the selected problem type. There are currently six solvers available for dealing with deterministic linear models: *Ip-solve* [95] under GNU LGPL, *Glpk* [96] under GNU, *Cbc* [97], *Symphony* [98], *Clp* [99] and *Dip* [100] under Eclipse Public License. On the other hand, there are only three solvers available for dealing with deterministic non-linear models *Couenne* [5], *Bonmin* [6] and *Ipopt* [101] under Eclipse Public License. The main characteristic of this set of solvers is their non-commercial license. So, they are free to be used by everybody.

Next step after choosing the desired solver is to upload the problem model files. Depending on the solver selected, the application allows different types of files to be uploaded. Right now, *MPS*, *LP* and *Pyomo* models [102] can be solved in the system. Moreover, for advanced users, complex and larger deterministic integer problems can be introduced in a decomposed format and solved by Dantzig-Wolfe decomposition, Lagrangian relaxation, and various cutting plane methods. More information about how to use this functionality can be found in the official documentation of the *Dip* solver. See [100].

The last step to configure the execution is to choose the type of solution. Here the users have two options: an optimal solution, where the solver tries to find the best solution, regardless of how long it takes (up to 24 hours), and a feasible solution, where the user defines a maximum duration, and the solver returns the best solution found before the limit time.

Finally, once the execution configuration is complete, it only remains to introduce the user email in order to be notified when execution finishes. If the session has been successfully created, a page appears providing the user with the necessary credentials to check the status of his/her session.

After these steps, the users receive their credentials for the execution and a randomly generated password. With this information, the users can access the

results section and check if the execution has finished or not and obtain the results when these are available.

When execution finishes, a summarised version of the solver output appears. This summary contains the objective value, the execution time and all the values of non-zero variables. The full solver output is also available for download. Moreover, charts representing the usage of CPU and memory are also available in the results section.

All the web application was implemented using AngularJS [103], a robust JavaScript framework. The decision to use this framework responded to the incredible potential to create dynamic, modern web pages, and this has been used in this application to create dynamic forms, like the one used for creating sessions. Hence, making this process more straightforward and more manageable, as the form reacts instantly to the user input, to warn if some information is wrong, if a problem has occurred and also provides some help or information, all this without even the necessity to submit the form to the server. Another important reason for choosing AngularJS is that it uses the Model-View-Controller (MVC) pattern, which makes a flexible, robust, loosely coupled and easy to update application.

In combination with AngularJS, other tools were used to create this web application. One of these tools is Bower [104], a package manager to manage all the application dependencies, like libraries, frameworks, or utilities, automatically. On the other side, the JavaScript task runner Grunt [105] was used to build and run the application on an HTTP server.

All these tools were combined and configured to work together with an excellent tool named Yeoman [106], which uses a set of plugins named generators to scaffold a complete project in a wide variety of web application frameworks, in this case, AngularJS.

Besides, many other tools, plug-ins and libraries were used to create this web application. For example, the Bootstrap library was used to give some visual style to the application, many AngularJS extensions, like Angular Chart, were used for the charts in the result section, with other libraries which work together with AngularJS, like jQuery.

The web service uses Representational State Transfer (REST) architecture, all communication between the client application and the server used the resource service provided by AngularJS, which provides an object to interact with REST server-side data sources easily.

5.2.2 *Backend*

The backend is the core of the application. It contains all the application logic and is the communication bridge between the client, the database, the queue, and the different computing nodes.

The server was implemented in Java using the Spring application framework [107], which is also based on the Model-View-Controller (MVC) pattern, and all its dependencies are managed using Maven [108].

The backend has to provide all the information needed by the client, so it can display it in a user-friendly way, and also receive all the information provided by the user and save or transform it as needed.

As explained above, all communication between the server and client is done through REST API calls. The main advantage of the REST protocol is the separation between the client (frontend) and the server (backend). Hence, it leads to increase the scalability of the project by allowing different components to evolve independently and also increase the portability of the interface to other types of platforms, such as a mobile application context. These calls are intercepted by the Spring controllers in the backend, where different API calls are mapped to Java methods which can receive input and generate a response which is sent back to the client.

Moreover, all the information about sessions, solvers, results and more, needs to be stored in the database so users can retrieve it later. The server is responsible for communicating with the database to create, save, destroy or modify the data contained in it.

All the data entities stored in the database are represented in the application as Java classes, forming the application model. Communication between the application and the database uses the Hibernate framework [109]. This framework offers a set of Java Annotations to specify how the data stored in these Java classes are mapped into database tables. Once the model entities are mapped to the database, they can be stored or retrieved from the database using the Spring repository controllers.

The backend controls the cloud quota to instantiate tasks in computing nodes. When the system is saturated, and there are not enough virtual resources for proper execution of the task, then the task keeps in the queue until the system has enough resources to deal with it. Task queue aims at dispatching task requests to computing nodes. Tasks are dispatched in a safe, steady rate and with a guarantee. Hence, the system is providing a reliable service by ensuring the performance of all the tasks and also a quality service by fitting the virtual resources to each system state.

The OpenNebula controller is the component that keeps track of the status of the system and the queue. Through the API offered by OpenNebula, the backend can perform operations such as instantiate a new computing node, destroy a computing node or rescale a computing node. Moreover, using SFTP the backend can communicate and transfer data between the computing nodes and also using SSH, the backend can perform remote executions in the computing nodes.

5.2.3 *Computing Node*

The computing node is the place where the work is done. Each Computing Node is a virtual machine with all the supported solvers installed, and an environment prepared to launch solver executions.

The computing nodes are created and destroyed for every execution: the virtual machine is created using an image with all the necessary software installed, the solvers are launched and, once the results have been collected, the virtual machine is destroyed.

Each computing node contains a set of shell scripts, one per solver, to execute all the necessary steps automatically to launch solver executions. These steps are:

1. Check if there are any input files on the node.
2. Execute the solver.
3. Generate a file with execution duration and results.
4. Send the results file to the backend.
5. Collect data about CPU and memory usage during execution and send it to the backend.
6. Send an email notification to the user explaining that the execution has finished.

5.3 RESULTS

In this section, a testbed based on linear models is used to show the usability and the goodness of the cloud service presented in this paper. Given the centrality of this issue, a set of benchmarks is used to compare the performance of different solvers, models, and hardware configurations using the cloud service proposed.

Next, the results show how to obtain an added value for reporting using the monitorization generated information. All the results were obtained using the same virtual machine configuration with 10 virtual CPUs from 5 real CPUs, and 10 GB RAM.

5.3.1 MIPLIB Benchmark

This study evaluates the performance of the solvers for dealing with a set of deterministic linear models. These models were obtained from the mixed integer programming library (MIPLIB), see [110] for more details. All the models were used in MPS format. Table 5.1 summarises the main characteristics of the selected models to perform the benchmark, where *Name* represents the name of the instance selected; *Type* represents the typology of the problem; *#Bin*, *#Con*, *#Int* represents the number of binary, continuous and integer variables respectively.

Name	Type	#Bin	#Con	#Int
noswot	MIP	75	28	25
50v-10	MIP	1464	366	183
k16x240	MBP	240	240	0
bienst2	MBP	35	470	0
cov1075	BP	120	0	0
enlight	MIP	169	0	169

TABLE 5.1: Main instances. Testbench.

The benchmark proposed is based on the selection of different types of linear deterministic models, such as mixed-integer (MIP), mixed-binary (MBP) or pure binary (BP) models. In Figure 5.5, the time required by each solver to solve the proposed benchmark is compared. The results reveal significant differences between solvers. First of all, it is shown that *Symphony* and *Cbc* perform best. Moreover, *Cbc* performs better with integer models while *Symphony* performs better with binary ones. Secondly, the worst solver to be applied in this benchmark is *Ip-solve*. However, *Glpk* has a performance very close to *Ip-solve*. Mittelmann [111] presents some benchmarks with similar results. They compare the solving time on a set of MIP problems also belonging to the MIPLIB between commercial solvers such as *Cplex* and *Gurobi* with OpenSource solvers such as *Cbc*, *Glpk* and *Ip-solve*. Their results are quite similar corroborating that *Cbc* is

one of the best open source solvers dealing with MIP capable of solving 53/87 MIPs. On the other hand, our benchmark was a mixture between MIP, MBP and BP suggesting that *Glpk* perform better than *lp-solve*. In contrast, in Hans' results for only MIP problems, *lp-solve* was capable of solving 7/87 instances and *Glpk* only 2/87. Moreover, authors in [112] presents another benchmark based on solving the cell suppression problem with similar results too. However, *Symphony* behaviour was not evaluated in these benchmarks. A closer look at benchmarks on the topic suggests that different types of problem affect the behaviour of solvers in the optimality resolution. Hence, this information can be used to develop smart policies or recommendation systems to select the best solver depending on problem topology.

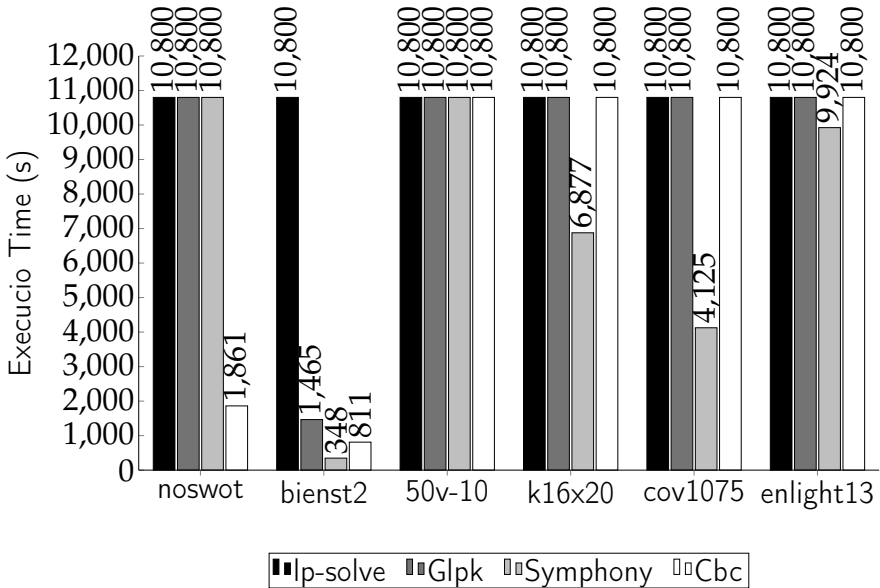


FIGURE 5.5: Solution time of different solvers in the resolution of the instances belonging to the MIPLIB.

5.3.2 Open Solvers versus Commercial Solvers

The purpose of this section is to check the differences between commercial solvers such as *Cplex* or *Gurobi*, and open source solvers like the ones proposed in the SPOS service. *Cplex* and *Gurobi* are both installed on the template used

for instantiating the computing nodes. However, the license does not allow us to share them with the public, so they are used internally and only for research purposes.

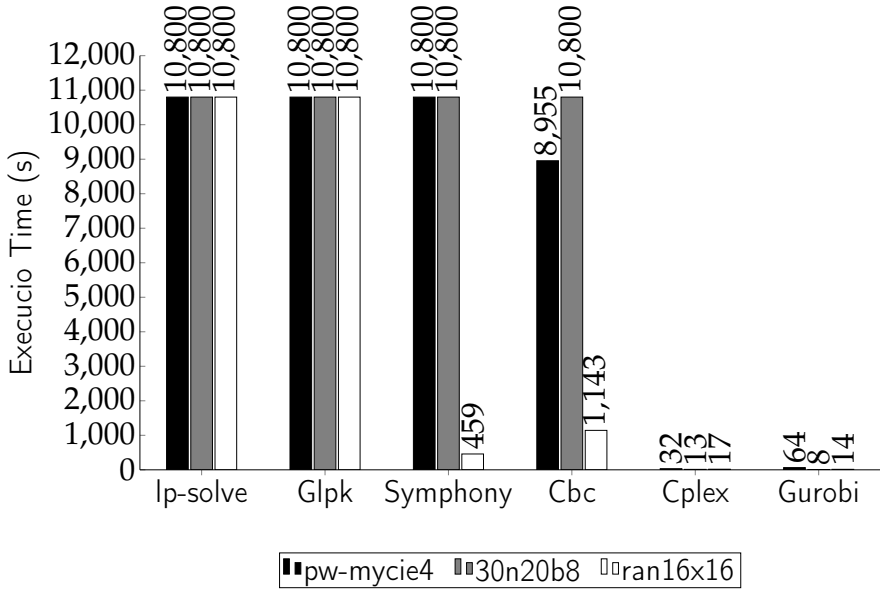


FIGURE 5.6: Commercial Solvers versus Open Source Solvers in the resolution of a set of instances belonging to the MIPLIB.

Figure 5.6 shows a comparison using the same virtual machine described in the previous section, and using three other MIP problems extracted from the MIPLIB. On logical grounds, the results corroborate that commercial solvers are more efficient and effective. The three instances are solved obtaining the optimal solution and also are solved very fast, whereas open source solvers are slower and most of the times incapable of obtaining the optimal solution within the time limit of 10800 seconds. Sometimes, these solvers obtained the optimal solution, but they could not prove that the current solution was optimal within the time limit. *Cbc* is the best option in this case because is capable of solving 2/3 instances. However, if *Symphony* can obtain an optimal solution, it is the faster one and the only one capable of competing with commercial solvers in this study, see *ran16x16*. These results are also very similar to the ones reported by the benchmarks described above. Evaluating whether *Cplex* or *Gurobi* perform better is out of the scope of this study. Moreover, this work prioritises giving to all the community access to powerful optimisation tools rather than the velocity

in computing the solution, but it is interesting to evaluate how far are the open source solution from the commercial ones.

5.3.3 Comparison of resource usage

This study is concerned with highlighting the capabilities of the tool, SPOS, presented in this paper to realise computational monitorization and benchmarks. Besides, it is interesting to show the real behaviour of open-source solvers dealing with complex instances. Hence, the instance named *enlight13* was selected to monitor its execution. The faster and most efficient solver for this study was *Symphony*.

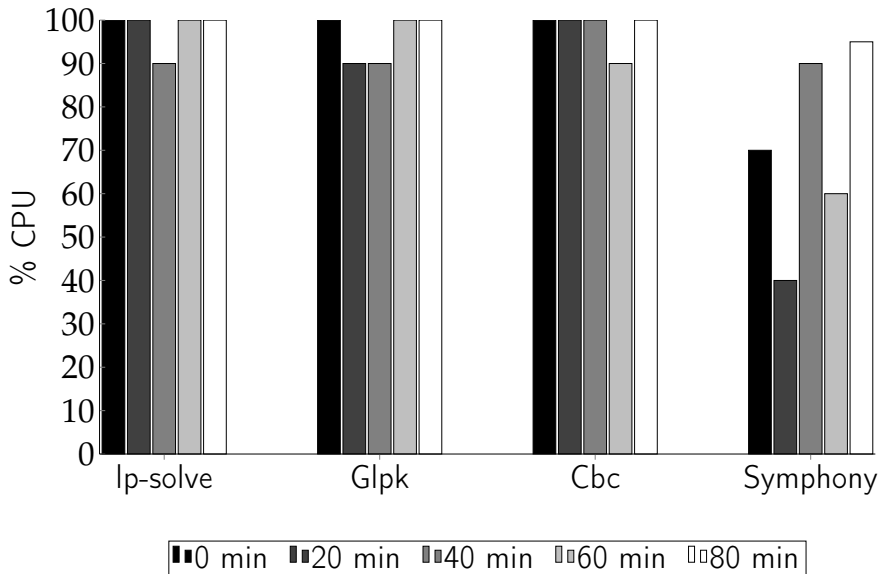


FIGURE 5.7: Instance *enlight 13*. Evolution over the time of CPU usage for all solvers.

Figures 5.7 and 5.8 reveal that the CPU and memory usage of *Cbc*, *Glpk* and *lp-solve* is fairly similar. On logical grounds, all solvers tend to use the maximum percentage of CPU, see Figure 5.7. However, slight differences can be appreciated due to the different internal algorithms performed by the solvers to resolve the problem. For example, the node processing module of *Symphony* is the most CPU consuming step of the five iterative ones of the branch and price implementation. Hence, there seems to be no trade-off between these

steps. There are some periods of execution where the amount of processing step tasks increase (0, 40 and 80 minutes) and decrease (20 and 60 minutes) the average. In contrast, the rest of the solvers seems to perform more similar along the time.

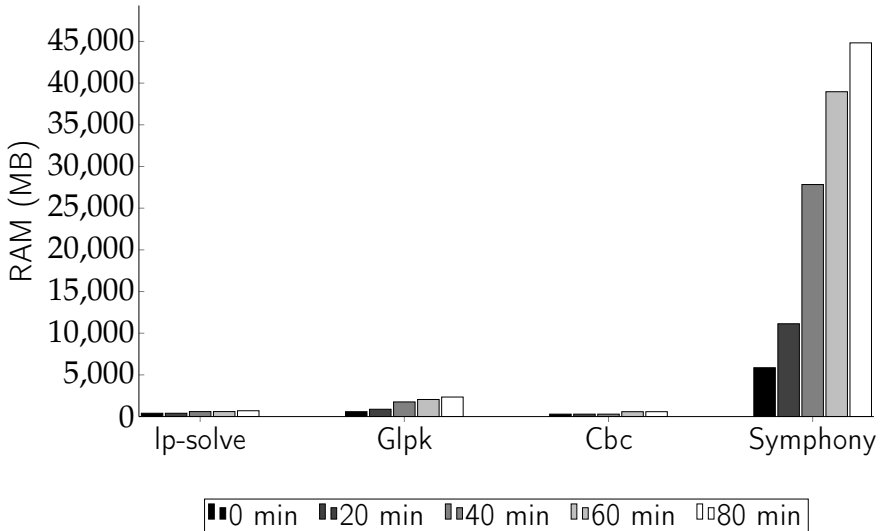


FIGURE 5.8: Instance *enlight 13*. Evolution over the time of memory consumption for all solvers.

Figure 5.8 depicts RAM consumption along time. *Symphony* required up to 4GB of RAM while the other solvers only used up to 0.5GB. These results suggest that *Symphony* stores more nodes of the search tree in the RAM. Note that the levels obtained by the *Symphony* solver increased progressively with execution. Hence, the need for RAM increase with the number of nodes explored by *Symphony*. In contrast, the other solvers seem to store less information on the RAM and keep more information in the secondary memory.

The results presented in this section are very useful in the academic context, where resource comparatives or scalability analyses are increasingly trending topics. In this sense, this capacity is an added value that highlights and differentiates this service from other similar services. From the results obtained in this section, it seems fair to suggest that the performance of the solvers is strongly linked to the size and kind of the problem. Furthermore, these results are provided by the usage of the service, so this SPOS functionality shows the benefits for making benchmarks and comparing executions in different virtual environments, between different solvers or checking the number of resources required over time.

Finally, these results corroborate that anyone with optimisation problem will be able to solve it and exploit the capacity of optimisation models.

5.4 CONCLUSIONS AND FUTURE WORK

The application presented in this article offers an alternative to the cloud optimisation services already established on the market by combining the best points of these into one single, free and easy-to-use cloud service. Combining the power of optimisation solvers, allowing stakeholders to choose which solver and model to use, together with the cloud computing platform OpenNebula and many more technologies, optimisation is accessible for anyone who has an Internet connection and a web browser.

SMEs can take advantage of cloud and optimisation tools delivered by SPOS to increase their productivity. Academic users can obtain a platform where their models can be tested, evaluated and also reproduced in the same configuration environment. These features can enrich Operations Research applied manuscripts. Hence, SPOS is a cloud-based service aimed to transfer the knowledge to the society.

Furthermore, it also allows users to customise their executions by creating custom virtual machines on demand, making the application suitable for both simple and advanced uses. The users' freedom to choose the solver and model and customise their virtual machine also allows different application usages, as demonstrated in the solvers' benchmarks done with SPOS, allowing them to analyse which solver performs best with a specific type of problem.

Although the service now offers a full range of utilities to solve deterministic optimisation models, the application has many areas open for improvement and which can make it even better. One of these improvements will be to support stochastic problems and solution methods to allow stochastic models to be solved in the cloud. Another area of improvement should be the generation of automatic experiments inside the service. Furthermore the option to build specific and personalised results visualizations, that adapts to the results of the problem solved, would make the results more understandable. This feature could be accomplished by generating maps displaying the optimised routes for a transport company or locating the best suppliers for a manufacturing company. Finally, the system can be improved with an automatic recommendation system based on a neuronal network, to suggest the best configuration concerning solver selection, machine configuration and solver options.

The service presented in this article is a powerful seed for a much larger service, with great potential to become a reference in the optimisation of the

cloud market, offering to all the users the possibility of solving any problem, even stochastic models, and of obtaining detailed and personalised results.

Authors: *Jordi Mateo, Wladimir Soto-Silva, Marcela C. Gonzalez-Araya, Lluís M.Plà-Aragonès and Francesc Solsona*

Journal: The Omega Journal

Status: Under Review

Keywords: *Supply Chain, Dried Apple, Two-stage mixed 0-1, Agroindustry, Tactical planning*

Supplier selection and cold storage management for agri-food products. The case of a dehydrated apple plant.

Abstract: Quality of commercial agri-food products is related to the quality of freshly harvested products and connected with the wastage reduction along the agri-food supply chain. Supplier selection is important to assure the quality of fresh products and refrigerated systems serve to mitigate deterioration and guarantee quality preservation. This paper explores the benefits of a two-stage stochastic programming model to consider the selection of producers supplying fresh fruit and the management of cold storage. The quality of fruit is reflected in its conversion rate into dried fruit. Bad conversion rates may force the purchase of additional raw material, incurring in unforeseen costs and increasing wastage. Fruit quality, purchase cost uncertainty is represented through scenarios which are generated based on historical data. Recourse actions include the purchase of additional fruit and renting of additional cold chambers. The results obtained show a cost reduction of 6.4% with our stochastic approach compared with the deterministic one. The results ensure the applicability of this model in practice before and during the harvesting season for planning and re-planning as far as uncertainty is revealed.

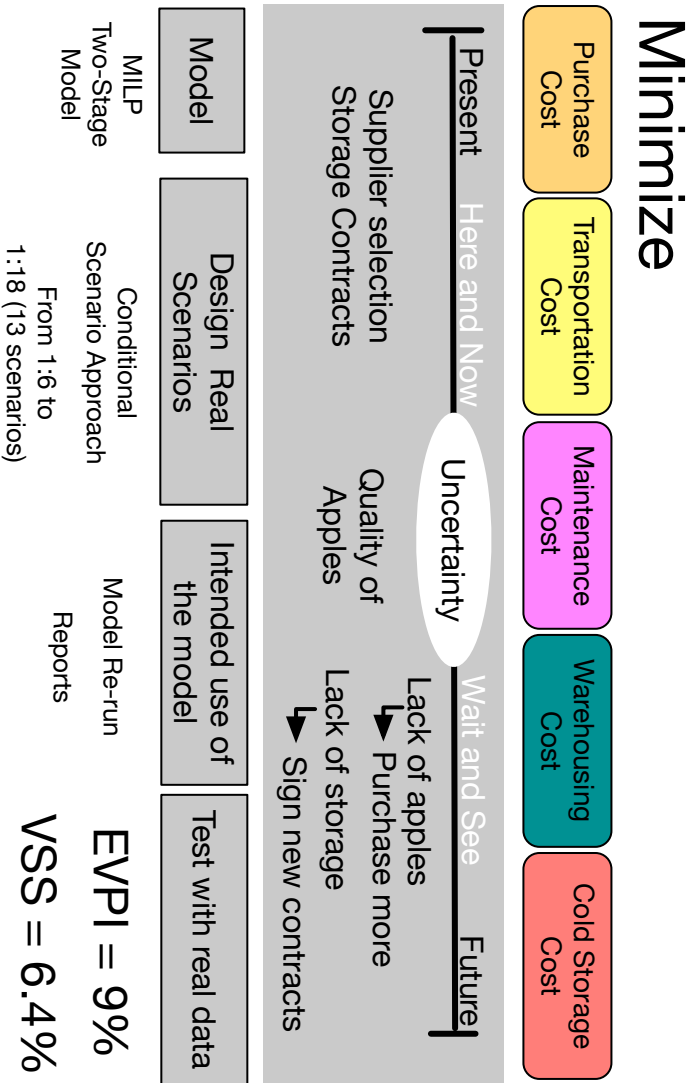


FIGURE 6.1: Graphical Abstract.

6.1 INTRODUCTION

Quality of commercial agri-food products is related to the quality of freshly harvested products and the wastage along the agri-food supply chain. As with other fresh agricultural products, fresh fruit production has a seasonal component that strongly affects the quality of the end-product. Special considerations for product quality connected in some way to the optimal ripeness of the fruit require more controlled storage conditions, delivery deadlines and minimising losses due to deterioration [113, 114]. Also, many uncertainties affect Fruit Supply Chain (FSC) management and are related to fruit biology, diseases, climate and market, plus added logistical complexities [59], making the underlying supply chain more complicated and harder to manage than other supply chains [115]. These challenges generate a need for more accurate and efficient decision-making tools to permit rapid adjustments at the operational or tactical level [116–118].

The structure of modern FSCs may vary regarding the number of agents collaborating in the different stages involved, ranging from farming (i.e. land cultivation, fruit production and harvesting), warehousing, processing, packaging and transportation to distribution. In general, the principal agent in the FSC is a packing or processing plant because, at the beginning of the season, plants have to make relevant tactical decisions, like the selection of suppliers and cold storage to ensure plant operation all year until the following season, forcing other agents to coordinate with them [69]. This paper adopts the perspective of managers at a packing or processing plant at the beginning of the season. Other important decisions, like harvest planning and processing, are made at different moments or levels and are out of the scope of this paper. In this context, supplier selection tries to assure the quantity and quality of fresh products and cold storage serve to mitigate deterioration preserve fruit quality and guarantee plant operation until next harvest season. Both aspects are even more critical to coordinate for processing plant managers when farmers and warehouse facilities belong to independent companies. Regarding supplier selection, processing plants have to identify reliable producers with high-quality raw material and agree to the purchase plan accordingly. Selection is complicated and considerable time is required to achieve stable relationships between good producers and the plant [65, 119]. Regarding the renting of cold storage, processing plants have to book facilities with a range of refrigeration technologies to allocate the purchased raw material until it is processed. [114] pointed out the necessity for proper storage handling with more controlled conditions for the fruit to prolong its profitability. This way, the final product sorted out from cold storage benefits from higher quality and the plant reduces wastage as the raw material can be

stored correctly to preserve its quality longer, reducing spoilage and so giving competitive advantages [120–123]. The transportation of fruit from the orchards to cold storage and from there to the plant is a production cost to consider for the FSC. Decision models integrating tactical decisions about production, transport and route planning are still scarce, and applications to real cases even more limited [124, 125].

[59] reviewed operational research models applied to FSCs showing studies focused on different stages of the chain. They remark the methods applied to supplier selection have focused mainly on multi-criteria analysis, mathematical programming and artificial intelligence [60, 61]. The main decision criteria used in supplier selection are quality, delivery time, price, manufacturing capacity, service, management, technology, research and development, flexibility, reputation, relationship, risk, safety and environment [62–64]. [65] advocated determining logical and straightforward selection criteria to make efficient decisions in the shortest time. Regarding cold storage, the critical decision is the capacity and type of cold storage to rent. In this sense, [59] proposed the coordination of cold chamber opening and closing according to the refrigeration technology. As mentioned, product quality is related to the optimal ripeness of the fruit and requires controlled storage conditions to minimise losses due to deterioration. Hence, [126] formulated a linear programming model to minimise the loss of fruit quality when stored in different types of cold storage. Regarding transportation decision models, the formulation of integer linear programming models minimising plant transportation costs is the most frequent method. For instance, [127] considered a distribution centre, while [128] focused on storage costs for the finished product or [129], on the use of means of transport for multiproduct supply chains. [66] and [67] took the degradation of quality along the supply chain into account, the latter emphasising transportation from producers to retailers. Routing problems are found more often in the distribution of perishable foods [68] than in fruit production or processing [69]. Other decision models dealing with technical particularities combining cold storage and product quality, like the optimal temperature for perishable foods or frozen fruit [130–133] or those considering environmental issues of transportation [134], are beyond the scope of this study.

Most of the models mentioned above are deterministic showing that decision-making models under uncertainty are one of the main challenges of the agricultural sector in general and the fruit industry in particular [70]. Stochastic programming permits explicitly incorporating uncertainty of some parameters via scenarios and have been proposed for solving a range of problems in other industrial fields (e.g. [135–137]). Recently, [71] revised the latest advances

and developments in the application of operations research methodologies to handling uncertainty in agricultural supply-chain management problems. Out of over a hundred papers that [71] reviewed, only a couple of them were related to FSCs [72, 73]. After this review, [138] propose a two-stage model applied to agriculture and related to supply chain problems.

Hence, in this paper, we intend to formulate a tactical planning stochastic programming model for the selection of the fruit supplier and cold storage and transportation to the processing plant. In particular, two-stage stochastic programs with recourse are an important class of stochastic models widely used in multi-period planning problems [139]. In such models, there are two kinds of decision variables. The first-stage decisions (i.e. here-and-now) represent proactive decisions taken before the uncertainty is revealed. In this paper, these decisions are related to the purchase of fruit from selected farmers and the renting of the required cold storage in the harvesting season to preserve the fruit and keep the plant operating till the next harvesting season.

On the other hand, the second-stage or recourse variables (i.e. wait-and-see) represent reactive decisions made in recourse or response to compensate for the decisions made in the first stage after the uncertainty is revealed. In our case, revealed uncertainty is regarding as the observed quality of the fruit for processing, which affects wastage and the conversion rate into the end-product. Thus, recourse variables correspond to additional purchases of fruit and renting of additional cold storage to meet final product demand. Hence, the objective of this paper is precisely to formulate and solve a two-stage stochastic programming model for planning the purchase and storing of apples for the operation of a dehydrated fruit plant. The model includes a list of farmers producing apples of different varieties, some private companies offering storing capacity, a fleet of trucks in charge of transportation and the total annual demand for the dehydrated product. Finally, the model proposed in this paper is a stochastic extension of the deterministic model presented by [140].

In the following section, the activities involved in the purchase and storage of fresh produce destined for the dehydration process in Chile are described. Section 6.3 presents the two-stage stochastic optimisation model formulation for purchase and storage decisions. In Section 6.4, the results and discussion of the case study are presented with the data obtained from a dehydration company in the Maule Region of Chile. Finally, in Section 6.5, the study conclusions are given, and future research is indicated.

6.2 SUPPLY CHAIN FOR THE APPLE DEHYDRATION PROCESS.

Fruit production is consumed as fresh or processed. The dehydration industry has undergone rapid growth in recent years and represents an interesting alternative for diversifying fruit related products [141]. Typical FSCs are represented by independent agents collaborating for the same purpose and coordinated by a processing or packing plant. Figure 1.7 summarises all of the actors involved and the flows between them. In our approach, soon before starting the harvesting season, the manager at the dehydration plant has to make decisions about the purchase of enough fresh apples for the plant to keep it operating until the next season. Once the season starts, the manager receives the fresh fruit and select where to store it. After that, regular decisions in a dehydrated plant are related to the retrieval and transport of fruit from storage to the plant for dehydration. However, if the foreseen quality of fresh apples processed in the plant does not correspond with expectations, complementary purchases may be needed to satisfy the annual demand. These decisions are important in the fresh fruit supply chain since, according to [142], the main production cost of a dehydrated apple is the purchase, transportation and storage (85%), and the rest is energy (5%), salaries (5%) and others (5%). This chain operates similarly to others fresh fruit and vegetables preserved in cold storage during the harvest season for later processing by the agro-industry during the rest of the year.

The processing plant manager would like to ensure the purchase of all the fruit needed to keep the plant operating until the next season at full capacity with minimal waste. However, the processed quality of the apples is uncertain as it depends on freshness and storage conditions. The conversion rate (fc) of fresh apple into a dehydrated apple is directly related to the apple quality and fruit segregation [142]. Thus, bad quality of apples can be originated on the field (damages when harvesting or bad weather), bad storage conditions (inappropriate temperature or atmosphere control) or a combination of both representing a waste increment. A reference in the industry is $fc = 1/11$ representing 11 kg of fresh apple needed to obtain one kg of dehydrated apple. A better conversion rate allows a reduction in the weight of fresh fruit to be purchased and so, to satisfy the same demand. Table 6.1 shows the historical expectation of fc^{-1} and its variability in a dehydration plant from season to season. Based on that and on field sampling, the plant manager calculates prior expectations of fc^{-1} for the next season (Table 6.1). For instance, in 2008 a 23% of increment in additional purchases of fruit was observed representing a 20% of the extra cost (i.e. US \$ 338,948.36) to meet the annual demand. The averaged deviation between the years 2006 and 2015 due to bad quality apples

was of US \$ 125,560.58 and resulted in a worse than expected conversion rate. Hence, the year when this happens, the plant manager has to go out to buy more fresh fruit, and also to rent additional cold chambers, not considered in the initial budget incurring in additional expenses.

Years	Apples (t)	End Product (t)	fc^{-1} expected	fc^{-1} real	Variation (%)	Variation (US \$)
2006	22,990.222	1,968.448	10.49	11.68	11%	151,735.44
2007	21,296.269	1,835.457	10.70	11.60	8%	102,222.12
2008	24,564.081	1,785.125	11.19	13.76	23%	338,984.36
2009	17,476.698	1,702.707	10.62	10.38	-3%	-31,458.25
2010	23,850.444	2,150.544	10.60	11.09	5%	71,551.37
2011	22,280.223	2,006.221	10.54	11.11	5%	66,840.75
2012	26,294.952	2,392.509	10.49	10.99	5%	78,884.94
2013	21,059.401	1,818.895	10.49	11.58	10%	126,356.42
2014	20,564.419	1,735.997	10.52	11.85	13%	160,402.51
2015	19,800.631	1,625.558	10.46	12.18	16%	190,086.11

TABLE 6.1: Variation over time of fc^{-1} for an apple dehydration plant.

Apples are picked up by producers, who are generally the orchard owners. The fruit is collected in bins with a load capacity of 0.380 t. In agreement with the producer, the processing plant collects the bins from the orchard and receives them for payment, accounting and selection [143]. Later on, the fruit is sent to different cold storage units, either in refrigerated chambers belonging to the plant or rented ones (see Figure 1.7). The decision to store the fruit in a specific refrigerated chamber is made by the processing plant on receiving the fruit. During the harvest season, prices depend on demand and can be lower or higher than out of season. Notice that the dehydrated plant competes for fresh apple with fresh fruit providers for human consumption in fresh. Besides, the amount of available fresh fruit in the market decreases around an 80% soon after the harvest season affecting directly to the prices, e.g. [142] claim that the prices could rise around a 30% per kg. These market conditions generate uncertainty in the fresh fruit supply. Data showed in Table 6.1 provide strong evidence about the variability in the amount of fruit to process yearly and the necessity of introducing these aspects into the optimisation model. As any corrective action to purchase more fruit beyond the harvesting season could represent an extra or a lesser cost depending on the market, purchasing decisions are more critical

when plants, warehouses and orchards belong to different private companies or farmers and they have to coordinate among them [69].

The type of cold chamber where the fruit is kept must also be selected and the fruit divided up (segregated) according to the storage period considered (short, medium or long-term storage). This segregation is done according to the variety and physiological indexes that are controlled by professionals in the reception area [144]. Among the most common indicators are the Brix degree, pressure, starch index and physical damage. In general, good quality fruit, i.e. with right quality indexes, implies a more extended storage capability with lower deterioration. In addition, an updated historical data of fruit quality per producer may help the plant to implement segregation quicker. A conventional processing plant with a capacity for approximately 30,000 tonnes of fresh produce can efficiently deal with about 250 producers who can offer six varieties of fruit suitable for different storage periods.

The storage period depends on the type of cold chamber and the technology used. Moreover, the ripening process for all the stored fruit must be controlled by the refrigeration system [69]. Available cold technology is conventional (CR) with only thermostat control, smart-fresh (SF) adding a phytohormone diffusion system which minimises the synthesis of ethylene and delays maturation, and controlled atmosphere (CA) where the concentrations of oxygen, carbon dioxide and nitrogen are regulated along with the temperature and humidity. CR, SF and CA allow the fruit to be preserved for periods of about three, six or nine months (short, medium and long-term respectively). Generally, plants need more than one warehouse to keep fruit stored for processing throughout the year, from season to season. A warehouse can have chambers with different cold technologies. It is common and advisable to have one type of fruit stored in a cold chamber at a time [142]. When a processing company does not have enough capacity for cold storage, it must lease it. The 86% of cold storage capacity in Chile is CR or SF, and the rest CA, being the latter the more expensive. The storage available to rent is required by different agro-industrial processing companies related to fruit, vegetables and meat products competing on the same market for cold storage. Therefore, although there is a minor variation in hiring cost along the year, it is true that the offer of cold chambers decreases considerably during the harvest season, with the risk of ending up in the surroundings. This situation implies looking for more expensive storage in farther locations. Then, an accurate decision to lease cold storage should be made at the beginning of the season or even before. Obtaining enough storage contracts to fulfil processing plant needs will be a competitive advantage for the companies. Storage cost is usually charged per tonne of stored fruit depending on the refrigeration

technology and, hence, corresponds to a variable cost. On the other hand, there is a fixed storage cost related to the administrative costs of renting cold storage.

Transportation from the orchards to the plant for selection and from the plant to the warehouses is usually outsourced since it is expensive to maintain a fleet of trucks, with their maintenance and drivers' salaries [143]. Payment policies vary; among the most common are payment per kilogram or bins transported, by kilometre covered or even an agreed fixed quantity for the whole season. However, transport costs generally have two components: a variable cost that depends on the quantity of fruit transported and the distance travelled; and a fixed cost corresponding to the transportation freight. In practice, processing plant managers are in charge of the logistics and transport of fresh fruit from suppliers to the plant and from the plant to the warehouses. They estimate a variable cost as a unit value per ton transported considering the distance between the plant and the suppliers, and from the plant to the warehouses. Concerning the fixed cost, the total load capacity of each type of truck is also considered.

6.3 TWO-STAGE STOCHASTIC MODEL

This section is concerned with the issue of proposing a stochastic approach to minimise production cost in the dried apple supply chain. The model proposed in this work is a stochastic extension of the deterministic model presented by [59] focused on the three initial steps of the chain (from the purchase to the storage), see Figure 1.7 presented in Section 6.2. The main contribution of this stochastic approach is considering the quality of the raw material uncertain. This way, the model can be outlined regarding taking advantage of the strengths of the deterministic model adding the raw material uncertainty to obtain a more flexible solution capable of adapting to uncertain quality of fresh apples in a realistic environment.

The supplier selection and cold storage contracts are critical decisions to be made during the harvesting season. The raw material (apples) can be purchased from the beginning to the end of the harvesting season, but the market prices are going to be influenced by different factors. One of the most important factors is the quality of the apples. Besides, the quality of the apple also influences the final demand needed to supply. Likewise, storage contracts can be signed during all the season as long as storage chambers are available. In this context, purchasing managers must make blind decisions at the beginning of the season without knowing the processing quality of the apples they have to buy neither the storage capacity needed to contract.

We propose a two-stage stochastic programming model to handle these uncertainties. The set of decisions are split into first stage decisions, made a priori before the harvesting season starts where the quality of the apple is unknown, and second stage decisions made once the uncertainty is revealed, knowing the performance (quality) of the current apples and the market prices. Hence, the second stage decisions are recourse action to correct first stage decisions if they fail and to fulfil committed demand. Thus, the model makes purchase decisions and signs storage contracts in the first stage, but now, depending on the future uncertainty, both can be corrected with recourse actions in the second stage.

First of all, the nomenclature used to model the input parameters and the required sets belonging to the dried apple supply chain are described in the nomenclature section. The tactical and operational decisions are also presented.

One of the purchase decisions consists of choosing whether to use a producer to satisfy the future dried apple demand or not. A binary variable is associated with the selection of these producers in such a way that $Z_p^\omega = 1$ whether producer p is used to satisfy the dried apple demand; otherwise $Z_p^\omega = 0$. Another critical decision closely related to the previous one consists of selecting which variety of apples are purchased from each producer. So, two binary variables are associated with this selection in such a way that $XP_{pq}^0 = 1$ and $XP_{pq}^\omega = 1$ if the variety q is purchased from producer p in the first stage and the second stage. Moreover, W_{pqt}^ω denotes the number of apples purchased from producer p that is used to satisfy the demand for the variety q and type t . Let Y_{lpq}^ω represent the number of trips between producer p and the processing plant using truck l to transport the variety q . On the other hand, one of the storage decisions consists of choosing whether using a cold storage chamber n inside a warehouse c to store a specific variety of apples is of interest or not. That is why a binary variable ME_{cnqt}^ω is required. Moreover, X_{qtcn}^ω denotes the number of apples to be stored. Finally, YC_{lcn}^ω is needed to calculate the number of trips made between the processing plant and the warehouses. The first component is the objective function that seeks to minimise costs associated with:

- The fruit purchase, see equation 6.1;

$$\begin{aligned}
 C(p) = & \sum_{p \in P} \sum_{q \in Q} (CC_{pq}^0 + CP_p) * XP_{pq}^0 * O_{pq} \\
 & + \sum_{\Omega \in \omega} \pi^\omega \left[\sum_{p \in P} \sum_{q \in Q} (CC_{pq}^\omega + CP_p) * XP_{pq}^\omega * O_{pq} \right]
 \end{aligned} \tag{6.1}$$

- Transport from the producers to the processing plant for segregation, see equation 6.2:

$$C(t_1) = \sum_{\Omega \in \omega} \pi^\omega \left[\sum_{l \in L} \sum_{p \in P} \sum_{q \in Q} CFT_l * Y_{lpq}^\omega \right] \quad (6.2)$$

- The maintenance and administrative cost for the producers from whom the fruit is purchased, see equation 6.3:

$$C(m) = \sum_{\Omega \in \omega} \pi^\omega \left[\sum_{p \in P} CM_p * Z_p^\omega \right] \quad (6.3)$$

- The transport from the processing plant to warehouses, see equation 6.4:

$$C(t_2) = \sum_{\Omega \in \omega} \pi^\omega \left[\sum_{l \in L} \sum_{c \in C} \sum_{n \in N} CFT_l * YC_{lcn}^\omega \right] \quad (6.4)$$

- Finally, the cost for keeping the fruit in cold chambers and fixed cost for transport is represented by equation 6.5.

$$\begin{aligned} C(s) = & \sum_{c \in C} \sum_{n \in N} \sum_{q \in Q} \sum_{t \in T} CF_{cn}^0 * ME_{cnqt}^0 + \sum_{\Omega \in \omega} \pi^\omega \left[\sum_{c \in C} CA_c * A_c^\omega \right. \\ & + \sum_{q \in Q} \sum_{t \in T} \sum_{c \in C} \sum_{n \in N} \left((CT_{cq} + CE_{cn}) * X_{qtcn}^\omega \right) \\ & \left. + \left(CF_{cn}^\omega * ME_{cnqt}^\omega \right) \right] \quad (6.5) \end{aligned}$$

In this context, the objective function represents the minimisation of the cost related to both stages involving purchase, transport and storage.

- First stage cost C^0 represent the contracts made at the beginning of the season related to suppliers (purchase cost) and warehouses (storage cost).

$$\begin{aligned} C^0 = & \sum_{p \in P} \sum_{q \in Q} (CC_{pq}^0 + CP_p) * XP_{pq}^0 * O_{pq} \\ & + \sum_{c \in C} \sum_{n \in N} \sum_{q \in Q} \sum_{t \in T} CF_{cn}^0 * ME_{cnqt}^0 \quad (6.6) \end{aligned}$$

- Second stage cost C^ω represent the corrective actions made at each scenario when the uncertainty is materialised.

$$\begin{aligned}
C^\omega = & \sum_{p \in P} \sum_{q \in Q} (CC_{pq}^\omega + CP_p) * XP_{pq}^\omega * O_{pq} + \sum_{p \in P} CM_p * Z_p^\omega \\
& + \sum_{l \in L} CFT_l * \left(\sum_{p \in P} \sum_{q \in Q} Y_{lpq}^\omega + \sum_{c \in C} \sum_{n \in N} YC_{lcn}^\omega \right) + \sum_{c \in C} CA_c * A_c^\omega \\
& + \sum_{q \in Q} \sum_{t \in T} \sum_{c \in C} \sum_{n \in N} \left((CT_{cq} + CE_{cn}) * X_{qtcn}^\omega \right) + \left(CF_{cn}^\omega * ME_{cnqt}^\omega \right)
\end{aligned} \tag{6.7}$$

$$\min \quad C^0 + \sum_{\Omega \in \omega} \pi^\omega C^\omega;$$

$$\text{s.t. : } Z_p^\omega \geq (XP_{pq}^0 + XP_{pq}^\omega), \quad \forall \omega \in \Omega, \quad \forall p \in P, \quad \forall q \in Q \tag{6.8a}$$

$$XP_{pq}^0 + XP_{pq}^\omega \leq 1, \quad \forall \omega \in \Omega, \quad \forall p \in P, \quad \forall q \in Q \tag{6.8b}$$

$$\sum_{t \in T} W_{p,q,t}^\omega = (XP_{pq}^0 + XP_{pq}^\omega) * O_{p,q}, \quad \forall \omega \in \Omega, \quad \forall p \in P, \quad \forall q \in Q \tag{6.8c}$$

$$\sum_{p \in P} W_{p,q,t}^\omega \geq D_{qt}^\omega, \quad \forall \omega \in \Omega, \quad \forall q \in Q, \quad \forall t, t' \in T : t \leq t' \tag{6.8d}$$

$$\sum_{t \in T} W_{p,q,t}^\omega \leq \sum_{l \in L} QMax_l * Y_{lpq}^\omega, \quad \forall \omega \in \Omega, \quad \forall p \in P, \quad \forall q \in Q \tag{6.8e}$$

$$\sum_{q \in Q} \sum_{t \in T} (ME_{cnqt}^0 + ME_{cnqt}^\omega) \leq 1, \quad \forall \omega \in \Omega, \quad \forall c \in C, \quad \forall n \in N \tag{6.8f}$$

$$X_{qtcn}^\omega \leq WC_{cn} (ME_{cnqt}^0 + ME_{cnqt}^\omega), \quad \forall \omega, c, n, q, t \in \Omega, C, N, Q, T \tag{6.8g}$$

$$\sum_{n \in N} \sum_{t \in T} \sum_{q \in Q} ME_{cnqt}^0 + ME_{cnqt}^\omega \leq |C| * A_c^\omega, \quad \forall \omega \in \Omega, \quad \forall c \in C \tag{6.8h}$$

$$\sum_{t \in T} \sum_{q \in Q} X_{qtcn}^\omega \leq QMax_l * YC_{lcn}^\omega, \quad \forall \omega, c, n, l \in \Omega, C, N, L \tag{6.8i}$$

$$ME_{cnqt}^0 = 0, \quad \forall q, c, n, t \in Q, C, N, T : t < TE_{cn} \tag{6.8j}$$

$$ME_{cnqt}^\omega = 0, \quad \forall \omega, q, c, n, t \in \Omega, Q, C, N, T : t < TE_{cn} \tag{6.8k}$$

$$\sum_{c \in C} \sum_{n \in N} X_{qtcn}^\omega \geq \sum_{p \in P} W_{pqt}^\omega, \quad \forall \omega \in \Omega, \quad \forall q \in Q, \quad \forall t \in T \tag{6.8l}$$

The model 6.8 (2SDASC) proposed in this section can be divided into two main blocks. The first block corresponds to the purchasing decisions and it is

made considering the constraints (6.8a - 6.8e). The second block represents the storage decisions and it is made using the restrictions (6.8f - 6.8k). Despite this, the two blocks are not independent, so constraint (6.8l) is required to ensure that the amount of apples stored is greater or equal to the number of apples purchased to satisfy the demand.

Let us consider the constraints related to the first block and corresponding to the purchase step. Constraint (6.8a), ensures that if a producer is selected to satisfy the demand, at least one apple variety is purchased from this producer. This way, it is essential to take care of the decisions made at different stages, so (6.8b) is required to coordinate the decisions in the first and second stages, so it is guaranteed that a decision can only have been taken in one specific stage. It is important to highlight that if a producer is selected to supply a variety of apples (Q), all the apples belonging to this variety must be purchased from this producer. This behaviour is represented in (6.8c); This constraint also guarantees that the number of apples used to satisfy the demand is available from the producers. Furthermore, all the apples purchased must be transported to the plant by trucks, see (6.8e). Finally, (6.8d) ensures that the demand for (dried apples) is met. Note that apples purchased to satisfy the long-term demand can also be used to satisfy short-term demand. However, apples belonging to the short-term demand cannot be used to satisfy long-term demand.

The second block (the storage step) ensures that all the apples purchased can be stored in the selected warehouses, see (6.8g). Another critical restriction that must be taken into account is that only one variety of apple can be stored in any one cold store, so (6.8f) is needed, and logically (6.8h) is used to guarantee that a cold store could be used only if its warehouse is selected. As in the other block, all the apples stored must be transported by trucks, see (6.8i). Last but not least, (6.8j) and (6.8k) are used to sort the apples depending on the different cold storage technologies.

6.4 CASE STUDY

In this section, real data from an agribusiness company are used to illustrate and assess the suitability of the stochastic approach. The results presented highlight the advantages of the stochastic solution. Furthermore, these results seek to demonstrate the applicability of the model to a real-world context dealing with uncertainty in fruit quality and purchase cost. The data was gathered from a leader Chilean agroindustry located in the Maule Region. These data are from the 2010 season which can be considered as a regular season. Figure 6.2 shows the main dimensions of the case study considered.

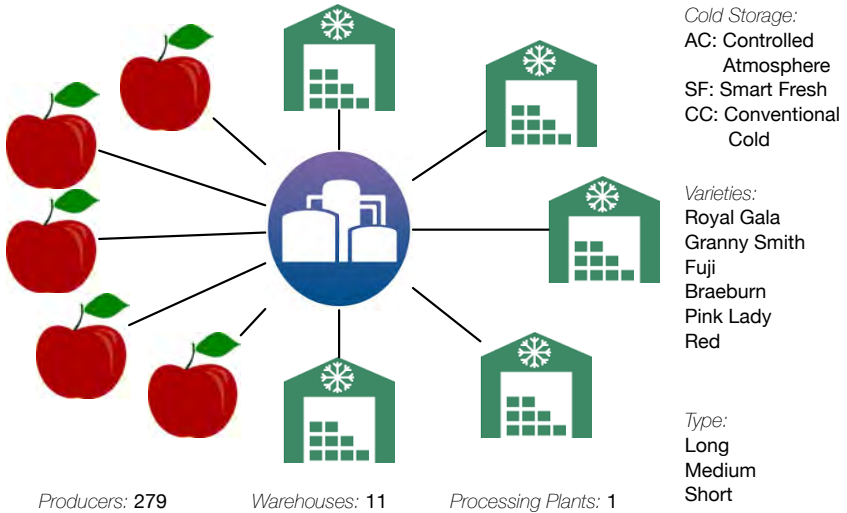


FIGURE 6.2: Main dimensions for the case study evaluated.

All tests were performed on a virtual scientific computing platform, known as Stormy, at the University of Lleida. More information about this infrastructure can be found in Stormy¹. The virtual machine chosen for the experiments was configured with 5 CPUs each with 2 cores, 25 GB of RAM and also 100GB of HDD. The operating system used was Ubuntu 13.04. However, a typical laptop would be enough to run the model. The model was implemented using the Pyomo modelling language [145, 146]. The main reasons for this selection are the flexibility, the open source feature, and also the potential of the Python language to deploy data manipulation and data representations. This way, the model presented is not linked to a specific solver, so it can be solved using different solvers, such as Cplex or Gurobi. Moreover, the model can be quickly provided with Python functionalities to feed and improve the decision support system in the dried apple supply chain context. In this case study, the solver selected is Gurobi 7.0.2².

1 <http://stormy.udl.cat>

2 <http://www.gurobi.com/>

6.4.1 Scenario generation

The generation of the stochastic scenarios lies at the heart of the discussion in which the managers of the dehydration plant also took part. The raw material quality, represented by the conversion rate fc , and the purchase cost at the second stage, C^ω , were selected for the design of the full range of scenarios (Table 6.2). Uncertainty can be modelled with a closer look at the historical data showing the range of principal random fluctuations. For this purpose, historical data on quality based on fc was analysed and used to compose Table 6.2. It was observed that a reasonable range of values was from 1:6 to 1:18. In this continuous space formed by these range of values, there are infinite possibilities to generate scenarios. To overcome this situation, the authors propose the usage of the Conditional Scenario Approach presented in [147]. Using this method we can obtain an excellent approximation to the real solution. Mainly, this method computes and discretise a set of conditional expectations of the random vector that models the uncertain problem parameters. This way, we build 13 typical scenarios representing the whole space. Therefore, to implement this technique, it is vital to computing a proper probability for each scenario π^ω . In this case, historical trends have been used to approximate this value.

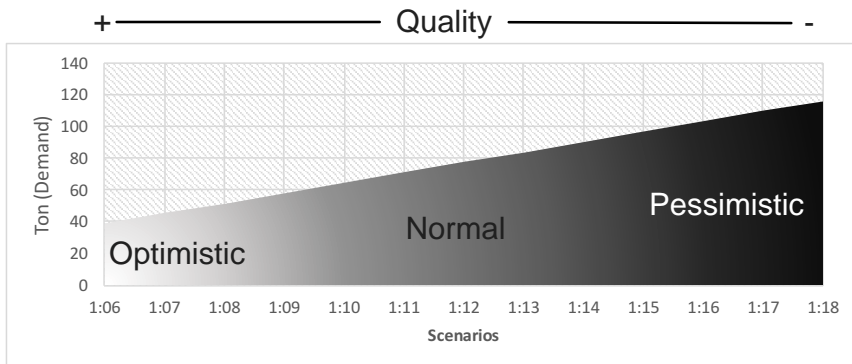


FIGURE 6.3: Main scheme for the generation of uncertain scenarios. From optimistic ($fc = 1 : 6$) to pessimistic ($fc = 1 : 18$).

Figure 6.3 shows the scenarios generated and illustrates the practical implications of fruit quality concerning the purchase amount per scenario (i.e. from 40 to 120 tonnes for scenarios 1 to 13 respectively). A heat map (white to black) was used to highlight the influence of the fluctuations on the conversion rate as the quality indicator. For example, scenario 1 is the most optimistic scenario

because it evaluates the apples with the highest quality. There is overwhelming evidence to corroborate that the higher the quality, the lower the amount of raw material required to satisfy the demand for dried apple, i.e. $fc = 1 : 6$.

Ω	π^ω	Quality	CC^0	%	C^ω
1	04%	1:06	12.5 \$	-45%	06.87 \$
2	04%	1:07	12.5 \$	-31%	08.62 \$
3	06%	1:08	12.5 \$	-25%	09.37 \$
4	09%	1:09	12.5 \$	-19%	10.12 \$
5	10%	1:10	12.5 \$	-16%	10.50 \$
6	16%	1:11	12.5 \$	-12%	11.00 \$
7	15%	1:12	12.5 \$	10%	13.75 \$
8	12%	1:13	12.5 \$	12%	14.00 \$
9	08%	1:14	12.5 \$	16%	14.50 \$
10	07%	1:15	12.5 \$	20%	15.00 \$
11	04%	1:16	12.5 \$	33%	16.62 \$
12	04%	1:17	12.5 \$	42%	17.75 \$
13	01%	1:18	12.5 \$	49%	18.62 \$

TABLE 6.2: Generation of the second-stage cost.

The second stage costs were generated assuming that high quality is related to an excellent production season and lower prices than what expected before harvesting. On the other hand, bad quality of fruits is assumed to be associated with a bad season, with small or scarce fruits leading to higher prices because of this scarcity. The ranges of purchase cost, price per kg, falls into empirical observations. See on Table 6.2 the mean purchase cost of the first stage, represented by CC^0 and being the same for all scenarios. On the same table, CC^ω represents the mean of the second stage unitary cost per scenario.

Regarding storage cost, CF^ω , it is considered to be 35% greater than CF^0 and homogeneous over the all the scenarios. Figure 6.4 illustrates the positive variation of purchase cost as quality deteriorates per scenario. Naturally, other considerations can be made to generate other unitary purchase costs per scenario, such as considering a “v” shape, or maybe an “n” shape, or using Montecarlo to forecast the new prices based on historical trends. However, the focus of this

paper is on showing that uncertainty about the quality of the raw material can be treated and is a critical factor in the apple supply chain.

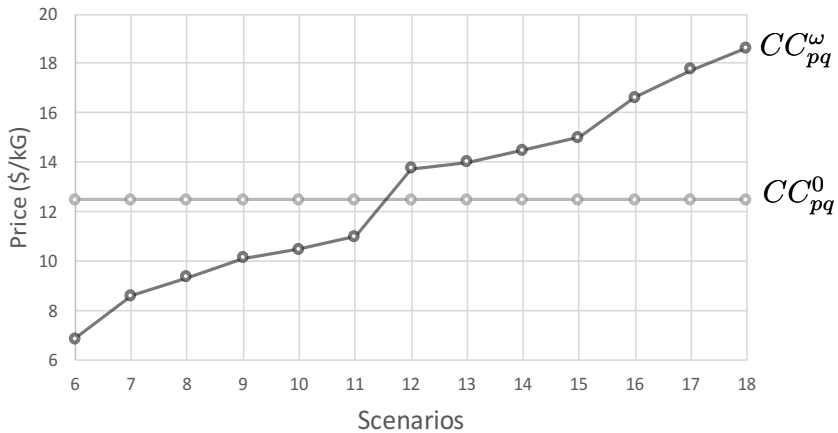


FIGURE 6.4: Fluctuation of purchasing prices over the time horizon for each scenario.

6.4.2 Results and discussion

6.4.2.1 Results of single scenarios

On logical grounds, there is no compelling reason to argue that if the purchasing managers knew the purchase cost, quality of the raw material and processing performance beforehand, the decision would be easier to make at the moment of purchase. In that case, the optimal decision would be calculated from the deterministic model corresponding to a scenario without uncertainty on fruit quality and purchase cost.

Table 6.3 shows the solutions for different scenarios solved independently. It highlights the outcome differences of dealing with a range of different fruit quality and unitary purchase cost represented by the scenarios (Table 6.2). In Table 6.3 we observe how the worse is the scenario more producers are selected and warehouses rented. In particular, the number of producers selected indicates the number of purchase contracts that would have to be signed under each scenario. Furthermore, the usage of a warehouse on the same table not only marks the number of cold storage to be committed but also the current levels of use in

per cent. As a consequence of the declining quality of fruit, a higher number of warehouses is observed.

Ω	Producers	Usage of Warehouses (%)										
		1	2	3	4	5	6	7	8	9	10	11
1	61	-	75	100	-	-	44	100	-	-	-	-
2	71	-	100	100	-	-	44	100	33	-	-	-
3	72	-	100	100	-	-	56	100	67	-	-	-
4	82	-	100	100	40	0	67	100	100	-	-	-
5	83	-	100	100	20	12	67	100	100	-	-	-
6	93	-	100	100	40	12	67	100	100	-	-	-
7	110	-	100	100	80	12	78	100	100	-	-	-
8	126	-	100	100	100	25	78	100	100	-	-	-
9	138	-	100	100	80	38	78	100	100	-	50	-
10	149	-	100	100	100	75	89	100	100	-	-	-
11	156	-	100	100	100	75	89	100	100	-	50	-
12	166	-	100	100	100	75	100	100	100	0	100	-
13	186	25	100	100	100	100	100	100	100	0	100	20

TABLE 6.3: The relationship between the number of producers selected and the warehouses' percentage of usage for each scenario. Obtained considering perfect information.

Moreover, Table 6.4 decomposes the total cost of ensuring the demand for purchase storage and transport. The purchase and the storage cost are presented in three components: first stage cost (1^{st}), second stage cost (2^{st}), and others (Others). The others cost represents the administrative cost, commitments with preferred producers or the cost of using a specific cold chamber inside a warehouse. General speaking, others cost are representing all the costs belonging either to the purchase or the storage step, and are also independent of the stage. Results on Table 6.4 highlight that recourse actions with perfect information are useless. If the decision makers know the unitary purchase cost at each stage, they always will select the lowest. Note, that the decision maker purchases the apples at the stage, first or second, they are cheaper. Hence, for the first six scenarios, the second stage unitary purchase cost is lower, so the outcome recommends to purchase at this stage. Otherwise, for the rest of the scenarios, the second stage is more expensive, and purchases are recommended at the

first stage. Furthermore, all the storage contracts are signed at the first stage, because signing storage contracts at the second stage has a penalty cost, mainly, due to the risk of a future lack of storage. As the number of producers and warehouses increase when the quality of fruit gets worse, the corresponding cost and transportation cost increase accordingly. Thus, the information displayed on the table can be summarised by saying that the model using perfect information sign all contracts when the price is lower and the total cost increase as the scenario get worse.

Ω	Cost in million of US Dollars (\$M)							Transport	Total (tc)
	Purchase			Storage					
	1 st	2 st	Others	1 st	2 st	Others			
1	-	0.355	0.001	0.002	-	0.089	0.056	0.504	
2	-	0.515	0.001	0.003	-	0.104	0.065	0.688	
3	-	0.639	0.001	0.003	-	0.119	0.075	0.838	
4	-	0.776	0.001	0.004	-	0.134	0.084	1.001	
5	-	0.894	0.001	0.004	-	0.150	0.093	1.144	
6	-	1.030	0.001	0.005	-	0.165	0.103	1.305	
7	1.269	-	0.001	0.006	-	0.181	0.112	1.571	
8	1.377	-	0.002	0.007	-	0.197	0.122	1.705	
9	1.486	-	0.002	0.008	-	0.213	0.131	1.841	
10	1.596	-	0.002	0.009	-	0.228	0.140	1.977	
11	1.706	-	0.002	0.010	-	0.244	0.150	2.114	
12	1.817	-	0.002	0.012	-	0.260	0.159	2.252	
13	1.928	-	0.003	0.014	-	0.277	0.169	2.391	

TABLE 6.4: Deterministic solution for each scenario (Costs).

6.4.2.2 Stochastic solution

The most important premise behind this study is that the quality of the apples (raw material) and purchase cost are uncertain. In particular fruit quality is not known when the selected producers sign production contracts at the beginning of the season. It is observed that quality may vary from season to season on a random basis and may cause disruptions in cold storage management. So

that, this study is concerned with highlighting the advantages of the stochastic approach and the need for accounting for uncertainty when planning the operation of a dehydrated apple plant. This approach represents an additional model complexity impacting on solving times. Table 6.5 depicts the size of a single deterministic model (one scenario) and the size of the full formulation of the stochastic model involving $\Omega = 13$ scenarios. Although the deterministic model for one scenario was solved in a reasonable time, that was not the case for the stochastic one. The size of the model beside the nature of variables justifies the need of fixing a stopping time criterion to solve the stochastic model and obtain a good enough feasible solution with a small GAP.

Instance	Variables			Constraints			Non zeros
	(0 – 1)	Z	R	≤	≥	=	
Stochastic Model	54926	69567	91038	118885	468	26226	1048127
One scenario	7598	5352	7014	9145	36	2322	84779

TABLE 6.5: Complexity for the models evaluated in the case study.

The stochastic solution (**SS**) was of \$1.56M. This value represented the overall cost considering all the uncertain scenarios. Moreover, this value has been obtained with an execution time limit of one hour and with a specific GAP of 0.21%. Despite this GAP, the solution is good enough to be taken into account. Different criteria have been proposed to measure the goodness of a stochastic approach concerning the related deterministic one [135–137]. The common criterion requires the calculation of the Value of the stochastic solution (**VSS**) and the expected value of perfect information (**EVPI**). The foregoing discussion implies computing **EVPI**, that is, the price that the plant manager would be willing to pay to gain access to perfect information. Equation 6.9 shows the way to compute this metric. In our case study, the **EVPI** value was of \$0.14M which represent a reduction in the cost of a 9%.

$$EVPI = SS - \sum_{\omega \in \Omega} \pi^\omega t c^\omega \tag{6.9}$$

Given the centrality of this issue for purchasing managers, and by the evidence that the future is unpredictable, another situation is considered. So, let us assume that the purchasing managers always decide to purchase using the optimal amount according to the expected mean of the quality of the apples (raw material). This way, the purchasing managers will use the stochastic approach with only the values of a mean scenario, to compute a purchasing policy (**P**). Then,

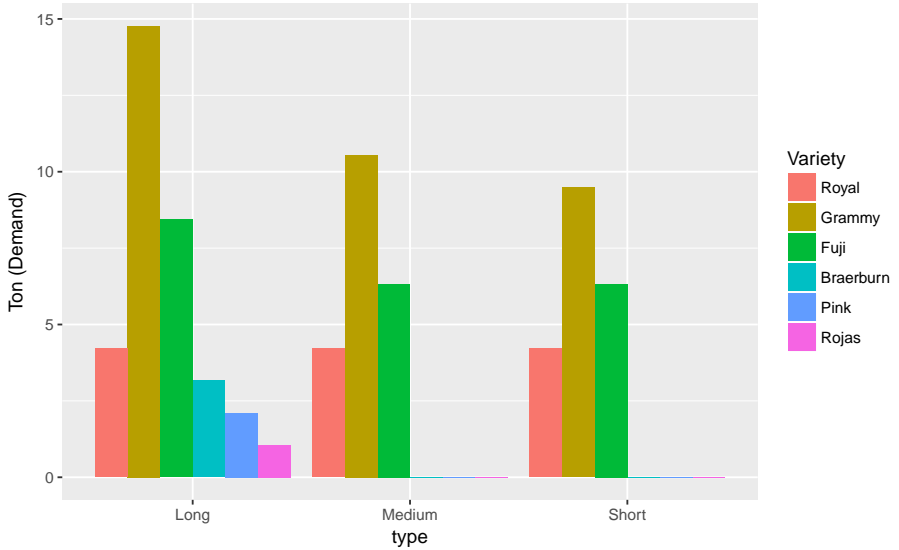


FIGURE 6.5: Expected demand for computing the purchasing policy (**P**).

the purchasing managers using the policy (**P**) will know the optimal amount according to this expected quality. Figure 6.5 shows the demand considered in the generation of the mean scenario. We can observe a trend to increase the number of tonnes placed in cold chambers with extended storage to assure the fulfilment of the demand during the last part of the year. Next, we retain the first stage solution of (**P**) and solve the problem for each scenario with a minimum cost (mv^ω). This way, we can compute the overall cost according to the expected quality solution (**EQS**), the only requirement is to sum all these partial scenario costs (see equation 6.10).

$$EQS = \sum_{\omega \in \Omega} \pi^\omega mv^\omega \quad (6.10)$$

From **EQS** and the stochastic solution (**SS**) we can compute also another measure to value the suitability of the stochastic approach, the value of the stochastic solution (**VSS**) [135]. The **VSS** represents the loss from not considering the random variations and is the difference between the optimal solution according to the expected quality and the stochastic solution (see equation 6.11):

$$VSS = EQS - SS \quad (6.11)$$

Table 6.6 and Table 6.7 summarize the results obtained from computing the optimal solution considering the stochastic model (**SS**). These tables contain the information presented by scenario without accounting for the corresponding probability, π^ω .

If we compare results of Table 6.6 with Table 6.3 we observe that the **SS** reduce the number of producers selected. In both cases, warehouse #9 is never used. On the one hand, if we observe Table 6.7 we can see how first stage decisions are the same for all the scenarios as expected. However, second stage decisions regarding purchases and renting of storage are more important than those in the first stage. We can interpret the importance of the recourse actions to adapt decisions to the uncertain scenario. On the other hand, the purchase and storage needs increase as the quality of the product is being reduced impacting accordingly in higher transport cost.

Ω	Producers	Usage of Warehouses (%)										
		1	2	3	4	5	6	7	8	9	10	11
1	47	-	100	100	40	-	44.4	100	66.6	-	-	-
2	51	-	100	100	20	12.5	44.4	100	100	-	-	-
3	63	-	100	100	40	12.5	55.5	100	66.6	-	-	-
4	72	-	100	100	40	12.5	66.6	100	66.6	-	25	-
5	74	-	100	66.6	80	25	66.6	100	100	-	25	-
6	88	-	100	100	100	25	66.6	100	100	-	-	-
7	101	-	100	100	80	25	77.7	100	100	-	75	20
8	118	-	100	100	80	37.5	77.7	100	100	-	75	20
9	133	-	100	100	80	62.5	77.7	100	100	-	75	0
10	139	-	100	100	100	37.5	88.8	100	100	-	100	40
11	147	-	100	100	100	75	88.8	100	100	-	100	-
12	164	25	100	100	100	100	100	100	100	-	100	80
13	176	25	100	100	100	100	100	100	100	-	100	100

TABLE 6.6: The relationship between the number of producers selected and the warehouses' percentage of usage for each scenario according to the stochastic solution (**SS**).

The model evaluated using real data was shown to be good enough for use in a real context. The evaluation of the stochastic indicators, **VSS**, shows that the stochastic solution can reduce the costs by around 6.4%, due to the recourse

actions. Putting all the information together helps to understand the results and why the stochastic model proposed performs better than the deterministic approach when fruit quality and purchase cost is uncertain.

Ω	Cost in million of US Dollars (\$M)							
	Purchase			Storage			Transport	Total
	1 st	2 st	Others	1 st	2 st	Others		
1	0.060	0.324	0.004	0.002	0.002	0.080	0.057	0.473
2	0.060	0.476	0.005	0.002	0.003	0.093	0.066	0.639
3	0.060	0.596	0.006	0.002	0.003	0.107	0.075	0.774
4	0.060	0.728	0.007	0.002	0.005	0.120	0.085	0.922
5	0.060	0.845	0.007	0.002	0.006	0.133	0.094	1.053
6	0.060	0.978	0.008	0.002	0.006	0.147	0.104	1.201
7	0.060	1.325	0.009	0.002	0.009	0.160	0.113	1.566
8	0.060	1.468	0.011	0.002	0.010	0.173	0.122	1.724
9	0.060	1.642	0.012	0.002	0.011	0.187	0.132	1.914
10	0.060	1.826	0.013	0.002	0.012	0.201	0.141	2.114
11	0.060	2.157	0.013	0.002	0.013	0.214	0.150	2.459
12	0.060	2.448	0.015	0.002	0.019	0.228	0.160	2.772
13	0.060	2.723	0.016	0.002	0.018	0.241	0.169	3.060

TABLE 6.7: Costs for each scenario according to the stochastic solution (SS).

6.4.3 Intended use of the model

The discussion in this study revolved around how the purchasing managers can mitigate the drawback of the uncertain raw cost and conversion rate fc when signing purchase and storage contracts. Given the difficulties that inaccurate fc is expected to provoke in the company (cf. problem description in section 6.2) the stochastic models offer an approach to protect managers from the uncertainty related to apple production through recourse actions. This way, the model fed with these scenarios representing fruit quality and purchase expenses fluctuations shows economic profits for the agroindustry.

We have observed in Table 6.6 and Table 6.7 how the number of producers selected increase according to the decline in fruit quality. The increment in purchases also implies an increment in storage needs. Then, for a practical purpose, we can consider the producers and warehouses being always selected regardless of the scenario. Python scripts were developed to exploit the results of the model and deliver useful information and reports to the companies as done with Figures 6.6 and 6.7. These Figures represent a visual way to access the solution quick and efficient. Therefore, Figure 6.6 shows the producers selected in all scenarios and representing a purchase of 1,233 tonnes, the 3.5% of total purchase. Note, that each square in the grid represents an individual producer, and all the markers draw inside indicate their apple’s offer. Moreover, the colours are used to differentiate the apple varieties, and the markers depict the state. For example, if the variety is purchased, the marker (x) will be used, however, if the same variety is available but not purchased, then is the point marker. Hence, each square with only point markers means that the model in some scenario discards this producer. Similarly, Figure 6.7 depicts all storage rented in all the scenarios and which chambers are requested in the first stage.

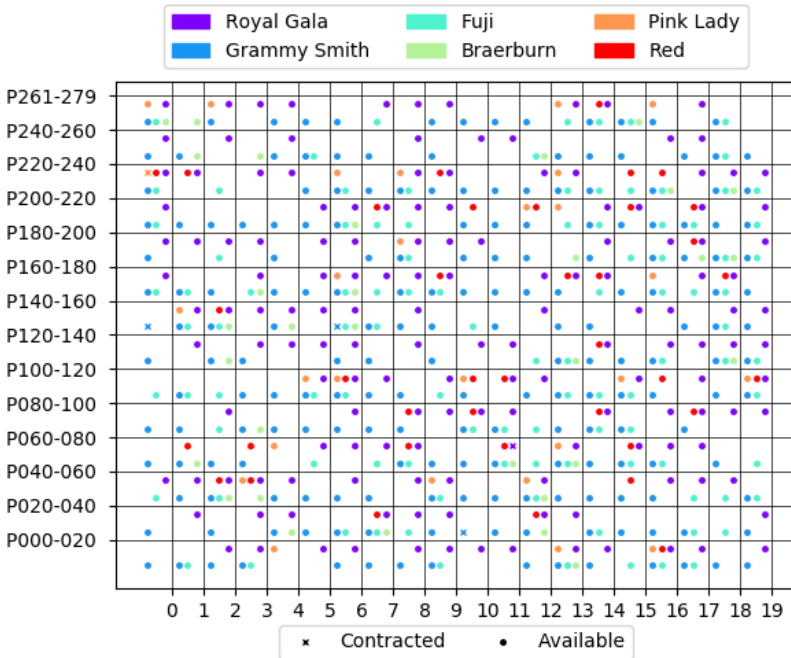


FIGURE 6.6: Common producers contracts for stochastic solution.

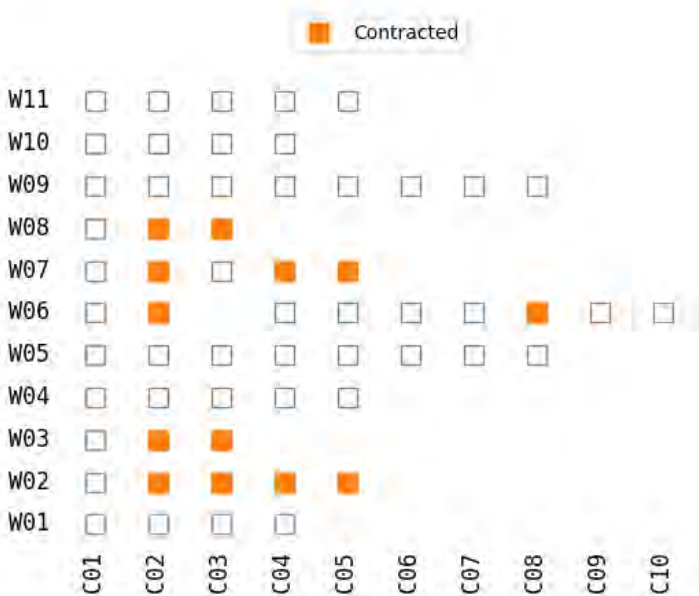


FIGURE 6.7: Warehouse common contracts for stochastic solution. Note that W denotes the warehouse and C the chambers.

The planning of the plant begins before the harvesting season of apple that takes around three months. Before the harvest season, the plant manager can perform the first planning according to samples on orchards to foresee the quality of fruits on the trees and guessing unitary purchase cost and implement the first stage decisions. This way, they can sign the contracts with selected producers at the first stage assuring the startup in the dehydrated plant. The harvest season gives the manager a chance to update the parameters in the model and adapt the solution according to the performance of the plant during this period. Thus, once starting to process fruit the quality of apples can be valued and proceed with the second stage decisions signing the rest of contracts assuring the operation of the plant till the next season. At any time, the plant manager can postpone second stage decisions or implement it partially and re-run the model and confirm or amend the original plan selecting new producers or deselecting them according to the progression of the season. Once the harvesting season will be finished, the uncertainty of quality of apples, purchase cost and rental cost of warehouses had been revealed, and a deterministic model for the corresponding

scenario can produce eventual second stage decisions that could be updated if conversion factor of apples gets worse. This fact may lead to consider a multi-stage stochastic problem to cover all the stages where a decision may have sense.

6.5 CONCLUSIONS

This work is focused on the design and resolution of a model to deal with the tactical decisions of purchasing and storing apples in the dried apple supply chain. The objective of this model is to minimise the costs of a dehydrated apple plant from a season by taking the uncertain quality of the raw materials into account. This way, the decision makers can purchase the amount of raw material prescribed by the model and be in a better position to control the quality factor of the raw material by recourse actions. Moreover, additional benefits of the model for decision makers are the reduction in storage and transportation costs.

Thus, a two-stage mixed 0-1 model considering the uncertainty of the raw material quality was designed and analysed. The model proposed (2SDASC) seemed to be realistic and also a right approach to tackling this kind of decisions because it allows decision makers to purchase apples and sign storage contracts at the beginning of the season. This situation is relatively close to the real context. Moreover, the results corroborate the gains from using this model in economic terms. Thus, the value of the stochastic solution shows the applicability of the model compared with the deterministic one. Last but not least, the results also show that the computation time for building the stochastic solution is not very high, making it suitable for use in practice.

The proposed model is practical and flexible since common producers are identified for all scenarios. The formulation of the model will be straightforward to adapt to other contexts with similar requirements (selection of producers, cold storage and processing plant). Note that similar supply chains in agriculture can be modelled using this two-stage approach and also consider a set of recourse actions. For example the strawberries. So, with slight modifications, mainly due to the different time horizon of the harvest season, this model can be used as a template to design similar models in other agroindustry contexts or with more stages. For instance, a multi-stage stochastic model to cover more than two decision stages would also seem reasonable.

Authors: *Jordi Mateo, Dídac Florensa, Adela Pagès, Lluís M. Plà, Francesc Solsona and Anders R. Kristensen*

Journal: Computer and Electronics in Agriculture

Status: Under Review

Keywords: *Decision Support System, Cloud Computing, Sow Replacement, Optimization, Agribusiness*

A cloud-based Decision Support System to support decisions in sow farms.

Abstract: The pig farming sector is a very competitive business where innovation is a key factor. On the research side, there exists a large body of models and analysis that too often do not reach the final user. In this paper we propose a cloud-based Decision Support System architecture that should overcome the barriers spotted in the literature. Our approach is presented on a sow farm model, which offers a set of analytic tools to help the farmers to make better strategic, tactical and operational decisions based on data.

A cloud-based service can take advantage and be integrated with the most popular farm software used to manage all the data related to the sow farm, either the historical data or the economic costs. This work extends the advantages of optimization and simulation models with the potential of cloud computing to offer knowledge and analysis to farmers. The results show that the cloud-based decision support system proposed helps decisions makers in pig farms to make a better planning and obtaining the competitive advantages of using the proposed model in a usable, flexible and transparent way.

DSS - ePigManagement

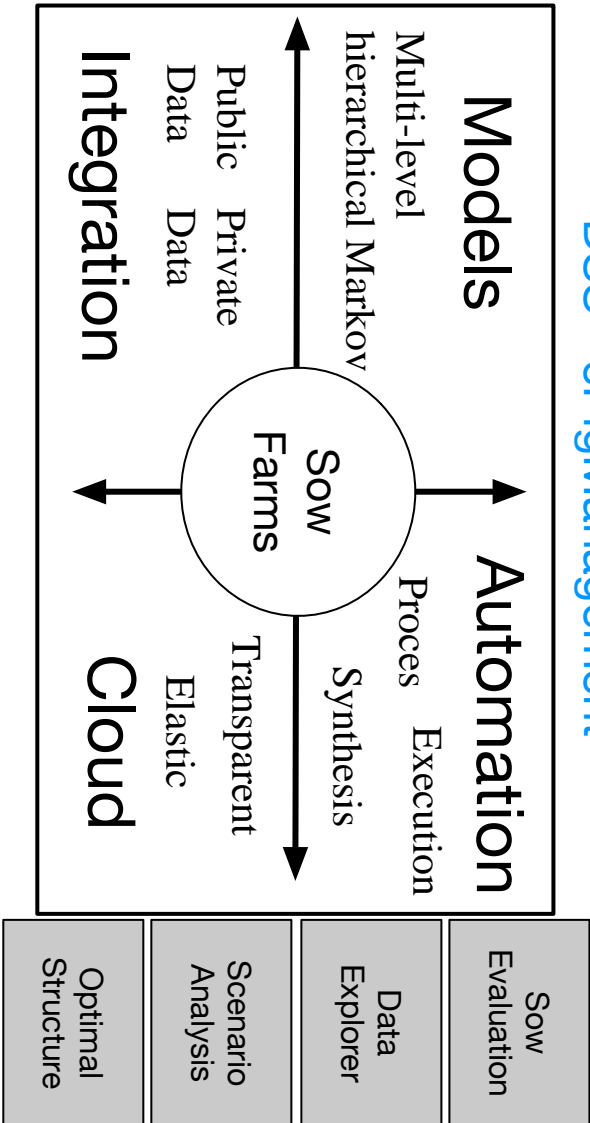


FIGURE 7.1: Graphical Abstract.

7.1 INTRODUCTION

The development of decision support tools for the pig sector has been the subject of research for several decades. Already in the late 1950s descriptive and mathematical modeling was applied to solve agricultural problems [148]. Since then, researchers have developed new models aiming at improving any process within the pig supply chain, from the optimization of the feed supply [149] to the control of the farm climate [150] or the sow replacement problem among others [55]. With the development of Internet and Communication Technologies (ICT) an early adoption was the collection of sow data from individual farms in a systematic way. Several commercial software companies developed software applications to register, organize, monitor and visualize the farm activity, which in itself had a great value and constituted the basis for the development of mathematical models that would help further in the decision making process.

On the other hand researchers continued to develop advanced models aimed at improving the overall performance of the several units in the pig supply chain [151, 152]. Despite the initial vision for great potential for a new set of decision tools for the sector, the success was limited [11]. Current state-of-the-art commercial solvers for farm management tends to be adapted to the new ICT paradigm (multi-site, cloud-based services) but with no added capabilities in terms of business intelligence related capabilities. As has been observed by several authors [10, 11] there are several reasons why a model does not arrive to the farmer's hands. The main reasons can be summarized as: a) unclear focus of the project, b) steep learning curve, c) lack of interface with the current management information systems, or d) hard system maintenance. In the same line and from the broader perspective of agricultural systems, Jones *et al.* [12] analyses the current state of the agricultural systems science, who emphasizes the definition of a Use Case as the central pivot for the development of the interactions between models and users, and expose the difficulties for the final user to find easily accessible and usable applications.

In this paper, we present a decision support system (DSS) oriented to help farmers in the replacement of sows in a breeding farm. The design of the DSS avoids the barriers spotted in the literature, as those mentioned above, by developing an easy-to-use cloud-based service. The main advantage of the tool is to reduce the learning curve of the final user, who only needs to provide data which he or she already has available and can obtain the final result in a direct and understandable way. Of course, the model needs to be calibrated, submodels need to be run, but all this knowledge is encapsulated within the cloud platform and is integrated seamlessly in the web application.

The development of applications as Software-as-a-service (SAAS) hosted in a cloud eliminate the requirement for a powerful computer with an environment dedicated to solve complex mathematical models. The only thing required to the user is an electronic device connected to the Internet. With only a web browser, decision makers obtain access to almost unlimited computing power, no matter which device is used (hand-handled device, desktop computer or laptops).

The novelty of this work is to present an elastic architecture that can be used to develop emergent decision support systems in almost every industrial process. In particular, this work is focused on presenting a cloud-based service that can be easily used by farmers or farm managers. Users can take advantage of the cloud features to get a tool that can be fitted and integrated with his/her current farm management software. In addition traditional farmers that work today with spreadsheets and notebooks also have the opportunity to use this tool to improve their decisions. The cloud-based DSS can be tested on *epigManagement website*.

The main contribution of this work is putting complex mathematical models as a service to the farmers in an easy and intuitive way in order to help and support the decisions related to the sow replacement. Moreover, the integration, management, analysis and organization of the data is also an important aspect considered in this work.

The remainder of the paper is organized as follows: Section 7.2 presents an overall vision of related work of sow replacement models and decision support systems. Section 7.3 presents our proposal for a decision support system in the replacement of sows in piglet production farms. It follows Section 7.4 which describes the cloud architecture that orchestrates the decision support system. In Section 7.5 the results from a real case study are evaluated and discussed. Finally, Section 7.6 outlines the main conclusions and future work.

7.2 RELATED WORK

This section presents a literature review and other related papers concerning the application used in this work, *i.e.* the sow replacement problem, and the development of decision support systems.

7.2.1 Sow replacement models

Livestock systems are complex and involve models at different levels (the animal and the herd) and types (*e.g.* biological models or feed models among others) in addition to the interaction among them. Being the piglet production one of

the key parts of the whole pig supply chain (it is the start of the production cycle), the optimal management of sow farms are of paramount importance.

Currently the replacement of sows is mostly based on the farmer's experience which relies on technical performance indicators, such as litter size, the age or the parity. There can be other thumb rules based on the company policy such as culling sows after an abortion or with a litter size below some threshold. But the final choice is based on technical measures and it is not backed by an economic analysis.

The fact that economic measures are not part of the decision process is mainly due to the fact that it requires advanced calculations such as determining the ideal culling rate and associated herd structure (i.e. distribution of sows over parities), litter size curve, feeding cost and prices. Farms are dynamic systems and this add complexity in analyzing whether a decision is economically sound or not. The goal of a farm producing piglets is to produce as much piglets as possible, and for that a balanced structure of the herd is needed. Even when this is determined, variability in sow production also happen, so stochastic models are introduced.

In the literature, there are several models that approach the sow replacement problem. Porkchop is a spread-sheet based model developed in the 80's by Dijkhuizen [50]. They introduced a new index called the Retention Pay-Off that indicates the total extra profit that retention of a sow is expected to yield over her replacement. Huirne *et al.* [51] presents a stochastic dynamic programming model that maximizes the present value of the expected annual net returns over a specified planning horizon of a sow herd. This model was refined later by introducing other attributes such as the piglet mortality rate or the number of undersized piglets at birth [52]. Kirchner *et al.* [53] propose the use of decision trees based on the gain ratio criterion to decide whether a sow should be replaced or not. In this case, sows are not ranked but classified. A multi-level hierarchical Markov processes with decisions on multiple time scales is presented by Kristensen *et al.* [54, 55]. The model incorporates the biological performance of sows by means of a dynamic linear model, the dynamic dropout structure and a feed intake model. Latter [56] the model was extended to also include clinical observations. Plà *et al.* [57] uses an equivalent semi-Markov model to represent farm dynamics to optimize the expected rewards. Finally, Rodríguez *et al.* [58] present a two-stage stochastic programming model to optimize the sow replacement and the scheduling of gilt purchases.

In the present study, the model proposed in [55] is used to obtain the optimal sow replacement policy that maximizes the benefits of the farms considering the current herd dynamics, predictions about the productive behaviour, casualties

and also the derived cost of the activity. The model optimize sow replacement from an economic point of view and not only accounting for litter size performance. A deeper presentation of the model is presented in Section 7.3.

7.2.2 *Review of Decision Support Systems*

Decision support systems can be defined as information systems that offer solutions (one or more alternatives) to help a user, which can be a single person or an organization, in the decision-making process. We focus on DSS which provides smart (*i.e.* data-driven and prescriptive, see 1.3) and usable solutions capable of managing the data, feeding the mathematical models with precise data and to integrate into the system all available resources.

The first generation of DSSs focused on simplifying the tasks of storing data, tracking changes and provided custom reports based on descriptive statistics. Advanced smart DSSs extend these capabilities and develop on top of that valuable information either with the use of predictive or prescriptive methods, by sending system alerts, by suggesting optimized alternatives or by providing the possibility to assess different scenarios. Another important characteristic is the capability to be integrated into the systems and devices that farmers are using in their daily tasks. Figure 1.3 summarizes the main differences between the first generation of DSS focused on answering the questions: What happened? and What is going on? by using basic statistical methods and the new generation of DSS aimed at explaining: What will happen? What shall we do? or What must we change? by applying simulation or optimization models among other techniques.

In order to identify the current state-of-the-art on DSS, we have surveyed what commercial applications are offering at the moment. The search has been limited to farm management software which offer cloud-based services. We have analyzed the following features:

- Storage which represents the capacity of the application to store data in an organized and hierarchical way.
- Traceability is the capacity of the application to track data changes.
- Data format is the ability to combine data from disparate and heterogeneous sources into meaningful and valuable information.
- Reports refer to the delivery of (usually aggregated data) information in form of descriptive analysis.

- Analytics is the ability to explore and interpret data, discovering patterns and meaningful information.
- Integration is the capacity to cooperate with the other agents in the current context.
- Alerts indicate whether the system is able to alert the users of special situations, either to anticipate or to mitigate the detected problems quickly or for taking advantage of some situations. This type of messages is usually the output of predictive statistical models.
- Suggestions indicate whether the system is able to generate and deliver proposals for improvements. This type of messages is usually the output of prescriptive mathematical models.
- Scenario analysis represents the capacity of the application to explore different unreal situations or scenarios, reproducing what would happen if this situation becomes true. This is usually the result of an underlying model of the system by means of mathematical modeling or a simulation model.

Table 7.2 shows the features that are offered by four commercial farm management applications. As it can be observed, current DSSs do not offer optimization or scenario analysis capabilities. The work presented here aims at filling this gap.

7.3 DSS ORGANIZATION

The DSS integrates the multi-level hierarchical Markov process presented in [55] in a modular way. The DSS is designed in a way that a step-by-step process is performed by the user leading to a smooth learning curve.

Figure 7.3 represents the overall structure of the DSS and the flow of information. To start with, the DSS requires a set of data, either public such as prices or private related to the farm performance or policies. The set of data required however is based on data available to the user and for which current farm management software already collects. With the evolution of sensors and smart farming, it is likely that in the near future some of the current models can be updated with more precise information [7, 157]. This data is used by different models to estimate the prolificity curve, the dropout structure or the biological production model. This processing stage is named calibration.

After the model is run, several outputs are generated. To keep the focus of the project, the application offers the main results obtained and needed to

	Agrivi	Agroptima	FarmLogic	CloudFarms
	[153]	[154]	[155]	[156]
Storage	✓	✓	✓	✓
Traceability	✓	✓	✓	✓
Data format	✓	✓	✓	✓
Reports	✓	✓	✓	✓
Analytics	✓	✗	✓	✓
Integration	✗	✗	✗	✓
Alerts	✓	✗	✗	✗
Suggestions	✗	✗	✗	✗
Scenarios	✗	✗	✗	✗

FIGURE 7.2: Comparison of different commercial software applications.

support decision making: the value of each sow in the farm, the comparison between the current herd structure and its ideal composition and metrics of the farm in the ideal state. The sow valuation allows the user to rank them and to get to know those sows which are below expectations, being therefore those the candidates to cull. This information has a clear operational focus. The other results provide information about the current state of the farm and what is the expected value in the ideal (optimal) situation. This information is of great interest to managers that look at the farm evolution with tactical/strategical vision. Also in the strategic domain, a manager may wonder what would be the effects of changing some of the current practices.

The main features of the decision support system designed can be summarized as:

- **Data integration:** The DSS proposed can be fed with data of any farm management software that the farmer is already using or even if the farmer does not have any digital data source, the farmer can introduce manually the data and start a digital transformation using the DSS.
- **Data explorer:** After the data processing, the user can obtain an overview of the current status of their sow farm.
- **Optimal suggestions:** One of the novel features of the DSS presented in this work is the capability of making suggestions. These suggestion are

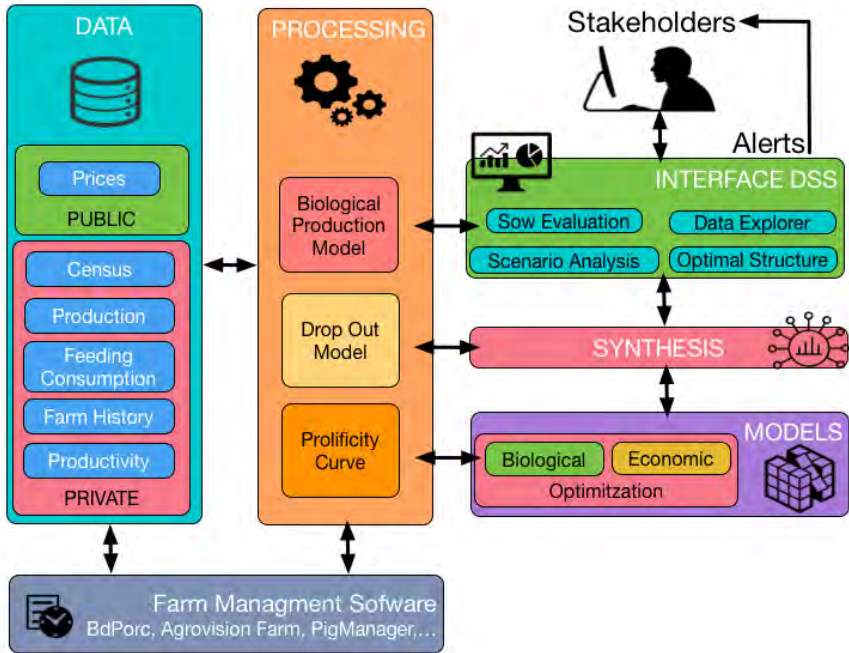


FIGURE 7.3: Representation of the different parts of the data structure, models integrated and output delivered by the current DSS.

based on the execution of a complex mathematical model that is able to obtain an optimal ranking of candidate sows to be replaced. This ranking is based on economic indicators.

- Scenario Analysis:** Another important contribution of this work is the capability of simulating and analyzing different scenarios. Farmers can evaluate what is going to happen if they decide to make changes in the farm. For example, they may wish to evaluate the influence and the consequences in terms of farm structure of modifying their prolificity curve by making some tactical decisions. Generally speaking, this feature helps the farmer to anticipate and mitigate undesired effects of some decisions.

These features are achieved after automatizing and encapsulating two important phases: the calibration of the farm parameters and the optimization process.

- **Calibration of the farm parameters:** The operations involved at the calibration stage are time consuming and require dedicated computer resources. It consists on the execution of different models coded in R, and the proper data transformation to connect the user's data and the R scripts output to the proper inputs of the following steps inside the calibration process. This stage is only performed the first time that the farm is registered.
- **Optimization:** In this stage the optimizer is executed and the optimal policy for sow replacement is obtained. Two type of outputs can be generated, either the indicators in the ideal state or the value of the sows present in the farm. From this point on, the suggestions and the ranking table are displayed. Note, that all the data required by the optimizer is heterogeneous and belongs to different sources, thus in this step, it is also required data manipulation scripts to unify the information.

7.3.1 DSS workflow

Figure 7.4 illustrates the work-flow to use the DSS, separating the steps corresponding to the first use from the rest. It also depicts the common actions that the farmer is allowed to do inside the DSS.

The main part of the decision support system is the **Dashboard**, the central part of Figure 7.4. Here, farmers can interact and manage the data, launch operations, check the results, consult the suggestions, configure their sessions, alerts and notifications and more. This area is private and contains all the information about the farm.

First of all, the farmer needs to create a new session inside the DSS to start using it. When a new farm is registered a usable step-by-step form allows the user to introduce the data and perform the first calibration and optimization of the farm. This form allows the user to introduce historical files, feed parameters, cost parameters, production parameters, census parameters and also productivity parameters.

One of the most important results that can be derived out of the optimal policy is the structure of the farm. This information summarizes the optimal herd structure that comes out after using the specific farm parameters, both combining technical performance and economic indicators. Hence, the farmer can compare the ideal structure with the present data and ultimately detect unbalanced situations. Related to the optimal policy is the prolificity curve of the farm. This has been estimated from the farm data and is displayed for informative purposes.

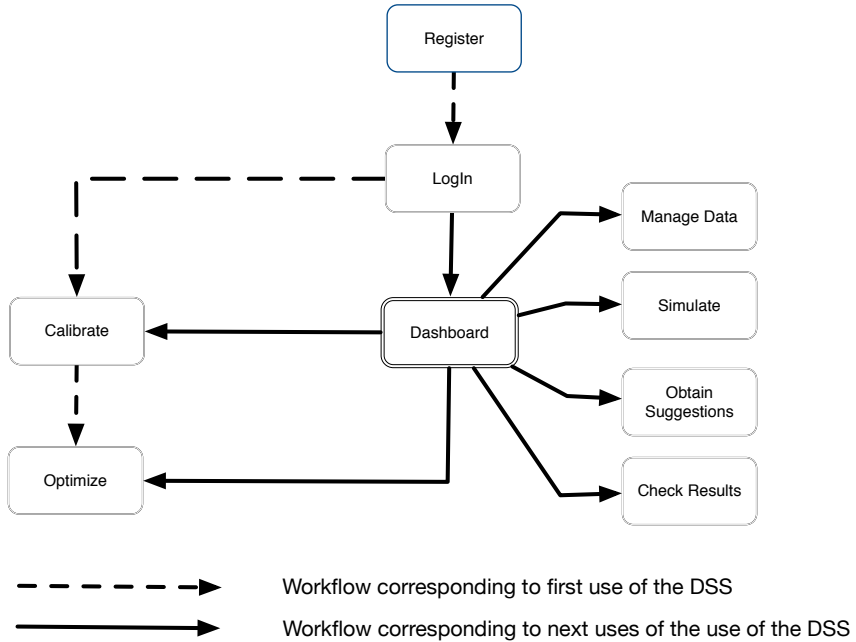


FIGURE 7.4: Workflow within the DSS. Dashed lines depicts the steps required the first time the DSS is used. Normal lines, depicts all the operations the farmer can realize inside the DSS.

The main information delivered by the tool is the suggestions about the candidates' sows to be replaced. This information is displayed as an interactive ranking. This ranking can be sorted using different criteria, being the most useful the economic criterion. The farmer has a tool, readily accessible, to support the critical decisions of replacing sows based not only in their experience but also based on mathematical models and economic indicators.

The DSS is merged with cloud computing to avoid the barrier of software installation and maintenance inside the farmer computer. Moreover, a cloud-based service can perform software updates seamlessly and can deliver a generic product that can be used by everyone independently of the type of device used to manage the farm and independently of the knowledge in data-science.

7.4 CLOUD ARCHITECTURE

This section presents the cloud architecture that provides a platform to offer knowledge as a service to the pig farmers. This architecture orchestrates the execution of the models, the presentation of the results and the administration of the data. The architecture is composed of three layers to split the functionality: the presentation layer, the business logic, and the data layer. This separation makes this architecture flexible and hybrid; capable to adapt to either endogenous or exogenous mathematical models, to introduce novel algorithms and even incorporate more heterogeneous data sources, such as a non relational database.

The architecture is built under the platform OpenNebula [94]. The main reason for choosing OpenNebula was to take advantage of the Stormy server. This is a private cloud deployed with the OpenNebula platform that belongs to the University of Lleida [2]. Nevertheless, the architecture is capable to be deployed or integrated into any commercial cloud, such as Microsoft Azure or Amazon Cloud. All the parts of the cloud-based service were deployed on this platform using virtual instances of Centos7 images as the operating system.

Figure 7.5 shows the layers skeleton. The architecture was designed to be flexible, elastic, scalable, easy to maintain and multi-purpose. The main function of the data layer is storing the information using different heterogeneous sources, such as a file server and a relational database. The business logic represents the mathematical models, the data manipulation, the report generation and the analytical computations. Finally, the main purpose of the presentation layer is the interaction with the farmer, gathering and displaying all the information.

First of all, the data layer represents all the tools that deal with the storage and organization of the data. Nowadays, this concept is considered a keystone in the digital transformation of any sector. Big data, small data, smart data or even not structured data have their space in this layer. Hence, the data layer proposed in this work helps farmers to collect, group, organize and store the data involved in the daily sow farms activity. In this work, the data layer is implemented using two different servers, a relational database, and a file server to store historical and log files.

Secondly, the presentation layer is the visible part of the application. This represents the tool that the farmers see and which interacts with. The presentation layer is a web application accessible from any modern browser and any device connected to the Internet. This is a soft layer that does not require too many computational resources. This layer is executed on the farmers' devices. Thus, the only requirement to obtain the reports and the analytics is a device with an active internet connection. The main purposes of this layer are allowing

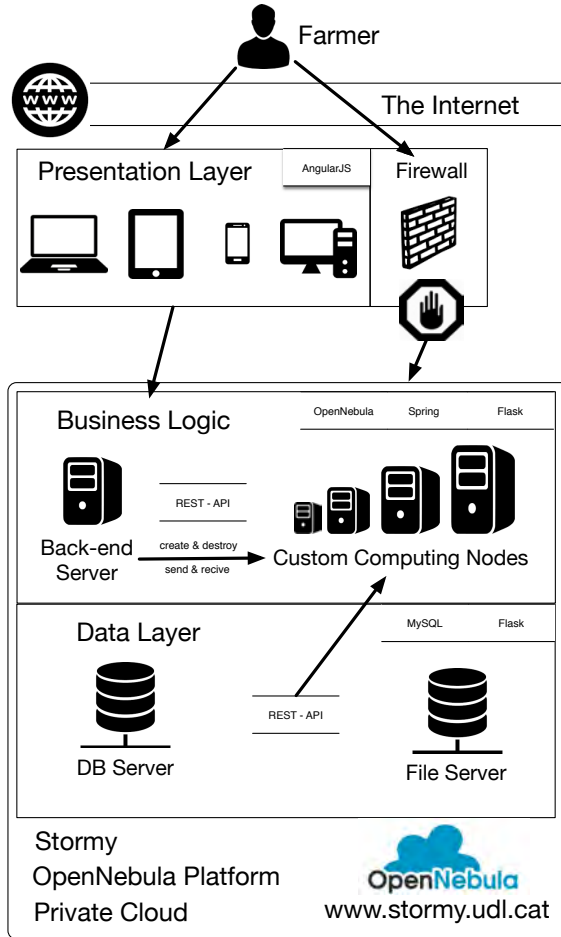


FIGURE 7.5: Cloud architecture divided into their three main layers. Highlighting the communication and the technology used to deploy the platform.

the farmers to register to the decision support system, upload and store their data, obtain the analytics, get access to an interactive dashboard and support their tactical and strategical decisions. To sum up, the user interface of the presentation layer makes the interaction between the mathematical models, the data, the analytics and the farmers as easy, transparent and usable as possible.

Finally, the core that makes everything work is the business logic. It contains all the application logic and is the communication bridge between the client,

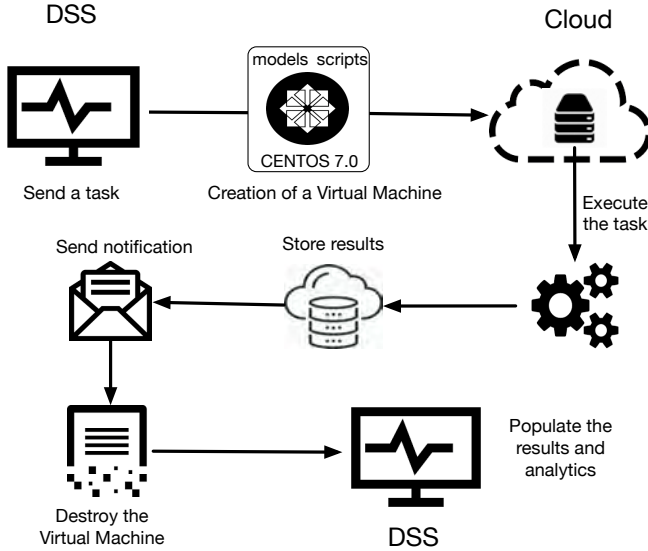


FIGURE 7.6: Life-cycle of a computing node. Steps to execute the tasks.

the database, and the different computing nodes. The business logic receives all the data from the presentation layer, transforms and manipulates this data and ensures that this data is stored in the data layer. On the one hand, the Backend server can be considered the brain that orchestrates all the skeleton. On the other hand, the computing nodes can be understood as the heart and the lungs, responsible for solving the hard computational tasks. The server was implemented using novel technologies such as Spring framework, flask framework and others. All communication between the server and client is done through REST API calls.

The virtual machines where the mathematical models are executed are hosted by the computing nodes. These nodes are elastic virtual machines that are dynamically created and destroyed under demand. Figure 7.6 depicts the life-cycle of a virtual machine inside the platform that manages the execution of a farmer task.

7.5 CASE STUDY

In this section, a realistic case study is presented to illustrate the capabilities of the cloud-based DSS proposed in this work. The results presented in this section revolves around the functionality of the DSS proposed. The potential advantages of using the model in real environments have already been presented in other works [55, 148, 158]. The focus of this work is to close the gap between model development and practical use. In this section we present the main outputs from where the user can extract information and knowledge.

The case study presented here uses realistic data from a Spanish company with 900 sows. Some of the data displayed has been truncated to preserve the privacy of the information. For illustrative purposes, the reader can test the application at *ePigManagement website* with a test user (details are given in the main page). Input data is not required by the test user, although the test user can check the data that is required to an active user.

In this section we start by presenting the data required from the user (data integration). This is the base over which the calibration step and the optimizer produce informative results for a manager, such as the prolificity curve or ideal structure of the farm (data explorer). This is a descriptive analysis (although it is based on advanced statistical and modeling tools) of the farm useful to understand the suggestions on sow replacement. Finally, the tool allows a manager to investigate how the change on one input parameter would impact the ideal structure.

7.5.1 *Data integration*

Data integration is one of the most important features of the DSS. After registration, the user is required to introduce the farm parameters through a simple step by step form. The steps are organized by areas of information: Production, Census, Feeding, Prices and Production. Some of the information refers to farm's policy on management and production system (such as the number of weeks of lactation) while other refers to the farm specific results (historical census of sow productivity or mortality and fertility rates among others). All the information required can be easily imported from almost all commercial farm management software. For an active user this step may be automatized with a background script that collects and sends the required information to our application.

Note that the input boxes provides automatically default values to speed up the input procedure. If the farmer is the first time that uses the DSS the boxes will be filled with common default values defined by the administrators.

The screenshot displays the 'ePigManagement' web application interface. At the top, there is a navigation bar with a menu icon, the application name 'ePigManagement', and links for 'New', 'Dashboard', 'Contact', and a user profile 'user@test.udl.cat'. Below the navigation bar is a progress indicator with six steps: 1 CONFIGURATION, 2 PRODUCTION, 3 GENUS, 4 FEEDING, 5 PRICES, and 6 PRODUCTIVITY. The 'PRODUCTION' step is currently active. The main form area contains several input fields and tables:

- Insmination (kg/day)**: A text input field containing the value '2'.
- During gestation (kg)**: A table with three columns representing different weeks:

Weeks 1-4	Weeks 5-12	Weeks 13-16
2.75	2.4	3.3
- During gestation, 2-3 days before birth (kg/day)**: A text input field containing the value '3'.
- For Piglets (kg)**: A table with three columns representing different weeks:

Week 3	Week 4	Week 5
0.03	0.12	0.6

At the bottom of the form, there are three buttons: 'Cancel', 'Back', and 'Next'.

FIGURE 7.7: Step-by-step form. Parameters used in the case study.

Nevertheless, if it is not the first time for the farmer the boxes are going to be fulfilled with the values used in the last calibration. The input data step is equipped with basic error-checking filters to avoid involuntary errors.

7.5.2 Data explorer

After the input data is provided, the calibration step is performed silently to the user. The most informative results are the ideal structure of the farm and its associated performance indexes. These results, compared with the current state of the farm, help to understand better the latter. Figure 7.8 compares the structure of the herd by cycles in the ideal situation with the current state. The structure of the herd in the ideal situation shows a population of sows at different cycles with a descendent proportion for higher parities. The real situation however is that there is a decompensation in terms of sows in the second parity.

Based on the ideal structure, several performance indicators of the farm can be computed, as those presented in Figure 7.9. These indicators are of common usage to farmers and farm managers. In this way, the final user can compare

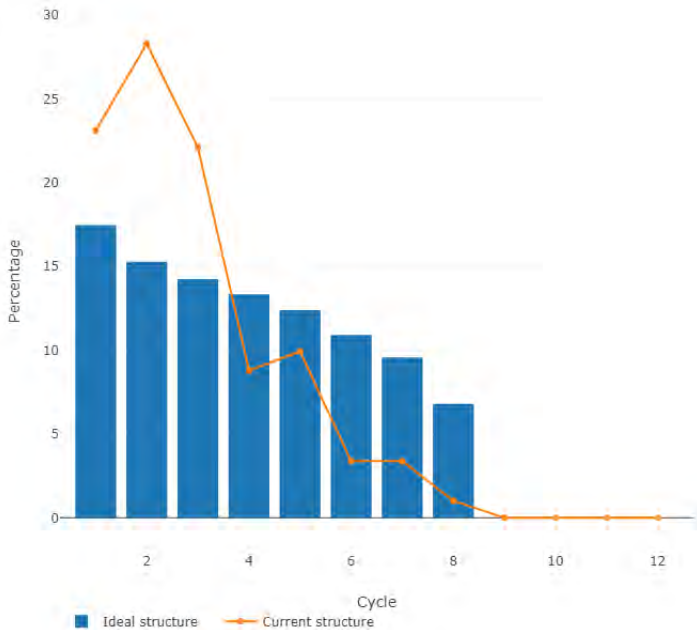


FIGURE 7.8: Case Study. Comparative between the ideal and the real farm structure.

the current farm's performance and the ideal situation. Although getting to the ideal situation may be an utopian aim, the comparison helps the user to position the state of the farm with respect the desirable situation.

7.5.3 Optimal suggestions

The most powerful feature of the DSS is the ability to offer the user with a list of sows candidate to be replaced ranked according an economic index. Given a list of sows with her current reproduction state and productivity history, a retention pay-off is computed by optimization [54, 55] based on the optimal policy. Sows with negative valuation have a reproductive potential lower than the potential offered by a gilt. This indicator summarizes the potential productive of the sow in an economic manner.

Figure 7.10 shows how the ranking is presented to the farmer. Each sow present in the farm is evaluated and a valuation indicator is computed. This

Census	
Average lifetime of sows (in years)	2.22
% Sow Replacement per year	45.05

Reproductive Rate	
Number of parities (per sow and year)	2.47
Average number of days between parities	147.85

Productivity	
Average number of piglets weaned per sow	56.22
Average number of piglets weaned per sow and year	25.19
Average litter size at weaning	10.26
Average number of litters produced per sow	5.48

Income	
Average net returns per year and sow (EUR)	178.85

FIGURE 7.9: Performance indicators under the ideal structure.

indicator is accompanied by extra information such as the current cycle of the sow, the number of piglets produced in the last parity and the state of the sow at present. By using this information, the farmer gets access to almost all the crucial information needed to evaluate whether to replace a sow or not. Hence, not only the economic value is important, but also the current state and the history are required to make a smart decision. With the values obtained for this case study, the farmer has a tool to support the sows replacement decision, corroborating their intuition or realizing for other better possibilities.

7.5.4 Scenario Analysis

From a managerial point of view, the farm manager may study future scenarios or perform *what-if* analysis based on the results of the ideal situation. The DSS offers the possibility of simulating scenarios and comparing with the current business-as-usual situation. In this case, the evaluation performed a change in the policy on the maximum number of parities allowed for a sow to stay. This is a strategic parameter that farmers can decide. The DSS proposed allows the farmers to simulate different scenarios in terms of farm structure to check the

Display 10 Search:

ANIMAL	CYCLE	PIGLETS	VALUATION	STATUS
H869161	7	7	-100.7377	PREGNANT
BT911174	5	4	-96.46921	PREGNANT
BT911251	5	5	-96.46918	PREGNANT
BR06482	2	2	-90.183075	PREGNANT
BR08632	2	5	-90.18306	PREGNANT
BT911689	5	4	-68.6794	LACTANT
BT911574	5	5	-68.67937	LACTANT
BR010034	2	4	-55.501724	OPEN
BR010202	2	5	-55.50171	LACTANT
BR010291	2	5	-55.50171	OPEN

Showing page 1 of 89 Previous 1 2 3 4 5 ... 89 Next

FIGURE 7.10: Sow ranked according to the retention pay-off (animal id needs to be randomized)

convenience or non-convenience to make this decision and the effects that would cause to the productive flow.

Figure 7.11 depicts the comparison between the base scenario and the situation where the maximum parities are reduced from 8 to 7. As in section 7.5.2, the graphic is backed by the technical and economic indicators, and the comparison between the two situations can be analyzed.

7.6 CONCLUSIONS AND FUTURE WORK

The transfer of models devised to support decisions from the research community to the final users is not an easy endeavor. The novelty of this research is the development of a usable, flexible and scalable DSS to support the decision making process in the agricultural context. This is presented by means of a cloud-based service aimed to support farmers in a reproduction sow farm.

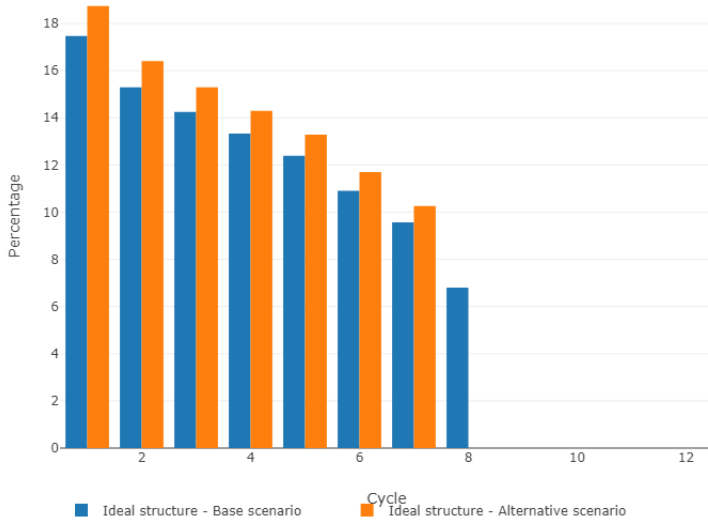


FIGURE 7.11: Case Study. Simulating scenarios based on parameters.

Four main causes of failure in the adoption of DSS tools by the final user have been identified and avoided as much as possible. To put a clear focus on the project, the application shows only those results that are informative for the decision-making process. The learning curve for using the model is rather flat, as the user needs to provide basic information that already collects and the complex parts of the calibration and optimization are performed silently for the user. From the output part, the results are also concise and uses mainly measures that the farmer is already familiar with. With regards to the interface between management systems and our application, we have analyzed several outputs from farm management software and the conclusion was that, even the format are different, all the required information can be provided by those. Finally, the barrier on how to maintain the system is overcome by developing a cloud-based service. This feature also avoids problems in terms of computation power needs and system-versions nuances. Moreover, the service presented in this article has great potential because is a web application, accessible with all devices from anywhere. Thus, the service presented is a powerful seed for a much larger service with a great potential to become a reference in the agribusiness world.

Merging the potential of cloud computing, advanced statistical methods and optimization models with usability and portability makes the farmers decision-making process smoother and data-driven. Therefore tactical and operational decisions can be taken on a model before they are implemented, avoiding undesired counter-effects.

Regarding the future work it is important to highlight the ability of the model to simulate different scenarios. Implementing further this feature will allow farmers to make better strategic decisions. Another enhancement in terms of usability and the comfort of the service is the implementation of a notification service. Finally, the storage and evaluation of farm historical data opens the world for a non-relational database. Integrating Mongo and Spark could be a key factor. The exploitation of these data could change the pig farms reality.

GLOBAL DISCUSSION OF RESULTS

I dream my painting and I paint my dream.

— Vincent van Gogh

The main purpose of this thesis was producing cloud-based systems to improve, deliver and transfer expert knowledge from the data science fields (such as computer science and operations research) to the society. The work proposed in this thesis answer questions belonging to the agribusiness field. Therefore, this thesis presented agribusiness solutions delivering high-performance computing methods and mathematical models through cloud-based technologies and architectures, mainly to allow decision makers to make better decisions.

At the beginning of this research, the efforts were focused on modelling cloud systems taking care of different metrics such as the quality of the service, the availability and the response time without putting aside the power consumption. The results of this research provide affirmative evidence that the combination of queuing theory models and optimisation techniques can be applied efficiently in real-world cloud environments. There is a rapidly growing trend in developing SAAS and real-time applications that open the door to new challenges in cloud-based hosting. The most crucials are ensuring high availability (*HA*) and reasonable response time (*RT*). QoS and SLA lie at the heart of the work realised in this thesis aimed to address these challenges. The industry demand tools to minimise overall costs by offering the best QoS to keep user satisfaction and avoid to pay SLA violations. The authors in [159] present an optimisation model to minimise cost and improve customer satisfaction level considering *SLA* violations and response time. In [160], the authors propose an analytical model to predict cloud service performance regarding the cost and the dependability of the service. In this thesis, we proposed a model based on queuing theory to optimising and analysing the impact in the hosting of cloud-based applications (i.e., SAAS) to guarantee a negotiated level of QoS regarding availability, performance, and cost. The model presented was evaluated in theory to highlight the impact in cost of offering different SLA contracts to the stakeholders. Moreover, the model was analysed and compared with a real service to verify the applicability. In this context, the model can be adopted as a modelling tool to simulate and evaluate different scenarios of SLAs regarding availability or response time. Finally, the results yielded by this study provides strong evidence

that the multicriteria model proposed can be used as a basis to design associated SLA contracts minimising the risk to pay for future SLA violations. Hence, the model leads to a better service design, which increases the system credibility and user satisfaction.

At this point, we were capable of offering an efficient layer of IAAS. Thus, we focus on improving the SAAS inside this IAAS layer. This thesis proposed the integration of mathematical solvers and the parallelisation of several decompositions methods to deliver the power of optimisation to the stakeholders. In Operations Research field there were a vast range of decomposition methods that assist the resolution of large complex optimisation models, such as Cluster Benders Decomposition [43] and Lagrangian Decomposition [48]. In this thesis, we improved these two methods by using high-performance computing paradigms to speed up the solving time. We developed a parallel implementation to solve discrete two-stage stochastic models based on the cluster benders algorithm. Then, we developed a parallel implementation to solve mixed integer two-stage stochastic models based on the lagrangian decomposition. The results of both implementations show computational performance and reduction of solving time around 18% using parallel benders and computational speed up around 5 in the lagrangian proposal. Moreover, the results also corroborated that decomposition methods can handle huge problems that commercial solvers can not (memory limitations) and parallel computing can take advantage of the particular structure of the decomposed matrix to speed up the resolution of very complex problems.

The next milestone of this thesis was improving the optimisation as a service. Inspired on NEOS [92], we developed a SAAS that takes advantage of cloud elasticity to offer an alternative to the cloud optimisation services already established on the market by combining the best points of these into one single, free and easy-to-use cloud service. SMEs can take advantage of cloud and optimisation tools delivered by SPOS to increase their productivity. Furthermore, academic users can obtain a platform where their models can be tested, evaluated and also reproduced in the same configuration environment. The results showed the advantages of using the service to perform benchmarking experiments. Therefore, SPOS is a cloud-based service aimed to transfer the knowledge of OR to the society. Currently, deterministic linear and non-linear problems are tackled by this service.

Finally, two agribusiness applications were presented. Agribusiness processes and activities are very uncertain. In this field, there were a lot of deterministic proposals aimed at solving common daily problems. These proposals obtain good solutions to be applied in practice. However, a more accurate and robust

solution can be achieved taking into account the uncertainty of some parameters. Hence, in this thesis, we improved a deterministic model using two-stage stochastic modelling to help decisions makers in the dried apple supply chain [79]. The evaluation of the model under real data and conditions showed that the stochastic solution can reduce the costs by around 6.4%, due to the recourse actions. Before the harvest season, the plant manager can perform the first planning according to samples on orchards to foresee the quality of fruits on the trees and guessing unitary purchase cost to implement first stage decisions. This way, they can sign the contracts with selected producers at the first stage assuring the startup in the dehydrated plant. The harvest season gives the manager a chance to update the parameters in the model and adapt the solution according to the performance of the plant during this period. Thus, once starting to process fruit, the quality of apples can be valued and proceed with the second stage decisions signing the rest of contracts assuring the operation of the plant till the next season. At any time, the plant manager can postpone second stage decisions or implement it partially and re-run the model and confirm or amend the original plan selecting new producers or deselecting them according to the progression of the season. When the harvesting season is finished, the uncertainty of quality of apples, purchase cost and rental cost of warehouses had been revealed, and a deterministic model for the corresponding scenario can produce eventual second stage decisions that could be updated if conversion factor of apples gets worse.

The field of agribusiness has much successful research works aimed to improve the decision-making process. Despite this, only a few are used or integrated by companies to be more competitive and make better tactical, operational and strategical decisions, [11] and [12]. In this context, we proposed cloud-based decision support systems to overcome these limitations and transfer the knowledge from academia to companies. In particular, we designed ePig to assist sow farmers. The results yielded by this study illustrated the capabilities of the cloud-based DSS proposed in this work to overcome the traditional limitations. Besides, the results corroborated the capabilities of the DSS proposed to give prescriptions (optimal ranking of candidate sows to be replaced), to analyse what-if scenarios and to assist the farmer in obtaining a better farm structure. The results also presented a successful case study for a local farm.

GENERAL CONCLUSIONS AND FUTURE DIRECTIONS

This chapter summarises the research work on Cloud systems and architectures applied to agribusiness that is presented in this thesis and highlights the main findings. It also discusses open research problems in the area and outlines a number of future research directions.

9.1 CONCLUSIONS

The objectives of this thesis were to contribute the research in cloud computing to transfer qualified knowledge from academia to the society. The research of this thesis abounded three important fields in data science: Cloud computing, Operations Research, and High-Performance Computing.

The first paper leads to improving the design of cloud architectures focused on offering services. The next two papers allow us to beat commercial solvers in the resolution of two-stage large instances. Hence, the next paper is the first step to OaaS (Optimization as a Service). Furthermore, the next work shows the benefits of applying two-stage models into the fresh fruit supply chain. This work highlights the reduction in cost due to a more robust solution. Finally, the last paper is the culmination of the work, improving the decision making in sow farms by overcoming traditional limitations. The main conclusion that can be drawn from this thesis are:

- The combination of Queuing theory models and mathematical programming leads to the design of better SLA contracts regarding QoS metrics.
- Decomposition methods and parallel computing are a perfect alliance to deal with large real-life instances and overcome commercial solvers.
- The cloud features such as elasticity, scalability or variability are crucial to transfer qualified knowledge from academia to society.
- SPOS service gives the opportunity to perform controlled experiments and grant access to linear and non-linear solvers to the overall community.
- In agribusiness supply chains, two-stage programming approach helps decision makers to face the uncertainty and make better and more realistic decisions.

- Merging cloud architectures, advanced statistical methods and optimisation models with usability and portability make the farmers' decision-making process smoother and data-driven.
- Agribusiness decision makers need natural, smart, robust and transparent decision support systems to perform their daily activities.
- XaaS (Anything as a Service) is the high-road to the future.

To sum up, the work on this thesis was focused on achieving agribusiness solutions to delivering high-performance computing methods and mathematical models through Cloud-based technologies and architectures, mainly to allow decision makers to make better decisions. Consequently, the introduction of this methodology into the agribusiness sector allow particulars, small-medium, and bigger companies benefit from academic expertise and also bring the opportunity to these industries to be more efficient and competitive.

9.2 FUTURE DIRECTIONS

Despite the contributions of the current thesis in the Cloud, High-Performance computing, OR and Agribusiness fields, there are many open research challenges to address to further advance in these areas.

Regarding operation research, several methods can be improved using parallel collaborating frameworks. Moreover, elastic cloud platform can provide a new layer of collaboration where not only information related to the solution is exchanged, but also the methods can dynamically adapt the resources of the different steps between the collaborating methods.

Regarding the transmission of knowledge to the society, a crucial step is to expand SPOS service to deal with stochastic models. In this context, the primary research challenge is standardising the way to generate a stochastic model. Furthermore, integrating parallel decomposition methods such as the ones developed in this thesis into SPOS is another exciting area of improvement. Besides, building a recommendation system capable of choosing for a given problem the best method or solver with the best configuration would be fabulous.

Cloud-based big data analytics and the IoT (Internet of Things) technology performs a vital role in the future of agribusiness. Data science applied to agribusiness implies the introduction of sensors to monitor the activity of animals, farms or orchards. The data from the sensors can generate alarms and perform precise actuation preventing future problems. Furthermore, sensor data can be used to feed operations research models and decisions support system

to represent the reality with more precision and obtain better decisions. Moreover, the analysis of all the historical information collected from sensors can help to identify patterns and perform better predictions and prescriptions. Thus, researching platforms that integrate IoT, Big Data, Cloud computing, Artificial Intelligence, Operations Research, and High-performance computing is the future of the field. Therefore, modelling complex models that can be solved faster assisted by a parallel collaborative framework and fed with smart data to be delivered with cloud-based decision support tools are the future. This thesis plants the seed of a much bigger and ambitious idea.



THREE MONTHS DOCTORAL STAY

During the course of this thesis, I spent a three-month doctoral stay at the Norwegian University of Science and Technology (NTNU) in Trondheim. My supervisor was Prof. Asgeir Tomasgard from the Department of Industrial Economics and Technology Management.

A.1 OBJECTIVES

The main objective of this stay was understanding a novel methodology to deal with strategical and operational decision inside a stochastic scenario based tree. We developed a multistage and a multi-horizon model to deal with the stochastic components of the optimal delivery of pigs from fattening farms to the slaughterhouse (ODFP). Progressive Hedging Algorithm (PHA) is a scenario-based decomposition technique for solving stochastic models. Hence, we planned to develop a parallel version to speeds up the resolution of multistage stochastic models and to propose an extension to multi-horizon stochastic trees.

A.2 TASKS

@DONE: Design the multistage stochastic version for deterministic multi-period ODFP model.

@DONE: Design the multihorizon stochastic version for deterministic multi-period ODFP model.

@DONE: Parallel implementation of PHA and testing the multistage ODFP instances.

@TODO: Adaptation of PHA to multihorizon ODFP instances.

A.3 LEARNED SKILLS

Usage of XPRESS Mossel Language.

Design and implementation of multi-horizon stochastic models.

Design and implementation of multi-tage stochastic models.

Implementation of scenario generation tools.

Implementation of markets simulation tools.

A.4 SUMMARY

The work done at NTNU was focused first on the design and implementation of a multistage stochastic model to deal with the price market uncertainty in the optimal delivery of fattened pigs from fattening farms to the abattoir (ODFP). The starting point was a deterministic multi-period model developed by Lluís M. Plà and Adela Pagès in OPL modeling language. During my research stay, I propose two stochastic adaptations of the ODFP. One using a multi-stage approach and another using a multi-horizon approach. Furthermore, initially, I developed all the code using XPRESS Mossel language, the language that I learned at NTNU. Next, I translate everything into Python and Pyomo modeling language. The main purpose was to extend the Pyomo framework to build and manage stochastic multi-stage and multi-horizon models. Moreover, we also implement and integrate simulation tools and scenario generation routines. Finally, MPI and C++ were used to implement a parallel approach for the progressive hedging algorithm to solve the multistage ODFP instances generated.

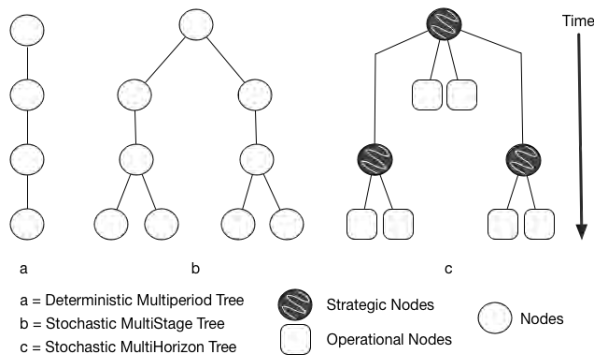


FIGURE A.1: Model Tree differences.

Figure A.1 shows the main differences of the tree implementations developed in this research stay.

A.5 MULTIHORIZON FORMULATION

$$\begin{aligned} \max \quad & \sum_{s \in sN} \sum_{o \in oN} \pi_{g(s)} \left[\sum_{f \in F} \sum_{c \in C} (FB_{\delta_o, c} p_{g(s)}) FW_{\delta_o, c} x_{f, s, o, c} \right. \\ & \left. - \sum_{f \in F} \sum_{c \in C} \left(FC_{\delta_o, c} fC + Alt + Cg \right) x_{f, s, o, c} - \sum_{f \in F} y_{f, s, o} tC \right] \end{aligned} \quad (\text{A.1a})$$

s.t. :

$$g(s) == (SS[s] - 1)mw, \quad \forall s \in sN \quad (\text{A.1b})$$

$$\begin{aligned} n_{f, s, o, c} &= qS_f, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C : o &== 0 \end{aligned} \quad (\text{A.1c})$$

$$\begin{aligned} n_{f, s, o, c} &\leq qS_f, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C \end{aligned} \quad (\text{A.1d})$$

$$\begin{aligned} n_{f, s, o, c} &= n_{f, s, P_o, c} - x_{f, s, P_o, c}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C : c > 1, o > 0 \end{aligned} \quad (\text{A.1e})$$

$$\begin{aligned} n_{f, s, o, c} &\leq n_{f, s, o, c-1}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C : c > 1 \end{aligned} \quad (\text{A.1f})$$

$$\begin{aligned} n_{f, s, o, c} &\leq qS_f * (1 - d_{f, s, P_o, c-1}), \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C : c > 1, o > 0 \end{aligned} \quad (\text{A.1g})$$

$$\begin{aligned} x_{f, s, o, c} &\leq n_{f, s, o, c}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C \end{aligned} \quad (\text{A.1h})$$

$$\begin{aligned} \sum_{c \in C} x_{f, s, o, c} &\leq qT * y_{f, s, o}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN \end{aligned} \quad (\text{A.1i})$$

$$\begin{aligned} z1_{f, s, o} + z2_{f, s, o} &\leq 1, \\ \forall f \in F, \forall s \in sN, \forall o \in oN \end{aligned} \quad (\text{A.1j})$$

$$\begin{aligned} (1 - z2_{f, s, o}) * qS_f &\geq n_{f, s, o, c}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C \end{aligned} \quad (\text{A.1k})$$

$$\begin{aligned} z1_{f, s, o} * qS_f &\geq n_{f, s, o, c}, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C \end{aligned} \quad (\text{A.1l})$$

$$\begin{aligned} z2_{f, s, o} &== 1, \\ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C : SO_s &== mw \end{aligned} \quad (\text{A.1m})$$

$$\sum_{o \in oN} (z2_{f,s,o} * SO_o / \Omega) == sW_{f,s},$$

$$\forall f \in F, \forall s \in sN \quad (\text{A.1n})$$

The ODFP multi-horizon approach is presented in A.1. This model is maximizing the net profit, considering that the net profit is computed as the profit minus the costs. The formula presented in A.1b indicates the way to compute the specific position to access as a global node, using the information of strategic and operational tree. This way, constraints A.1c and A.1d are needed to initialize the batch at every strategic node and to ensure that the batch does not exceed the farm capacity in any week. Furthermore, to control the flow of pigs between the fattening farm and the abattoir, constraint A.1e is added. Moreover, real fattening farms have the constraints that bigger pigs must be sent first, so constraints A.1f and A.1g ensures this behavior. In this point, the model needs to have control of the batch and also ensure that at most in fw weeks the batch is closed and all the animals are sent. Thus, constraints A.1j,A.1k,A.1l and A.1m are required. Finally, A.1n represents the connection between the operational and the strategical horizons.

B

BRIEF REVISION OF MATHEMATICAL PROGRAMMING

In this thesis, Operation Research (**OR**) is one of the essential pillars. **OR** is the discipline that deals with the generation of models that depicts somehow the reality and applies smart analytical methods to help make better decisions. One of the most powerful tools inside **OR** is mathematical programming. This tool helps decision makers to model problems where one seeks to minimize or maximize an objective function. This way, the models are capable to choose the values from real, integer or 0-1 variables from an allowed set and also subject to a set of constraints. Once the model is formulated, **OR** uses optimization algorithms and heuristics to solve them up to optimally or just up to feasibility. Figure B.1 depicts the range of features that characterize a mathematical model.

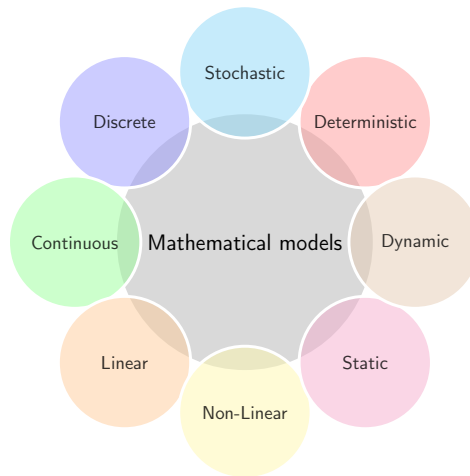


FIGURE B.1: Types of mathematical models.

B.1 LINEAR PROGRAMMING

Linear programming (**LP**) is the process of modelling complex relationships through a linear objective function and a set of linear inequalities. The role of inequalities is defining the feasible set for all the variables involved. The

linear objective function is optimized over a convex polyhedron specified by the linear and non-negative constraints. The optimal solution is always at the intersection of some constraints. A general LP can be formulated in the following way:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned} \tag{B.1}$$

In this example, formulation B.1, the decision variables are given by vector x , while c^T represents the coefficients for the objective function. Moreover, the matrix A represents the coefficients for the constraints and b are the right hand side limits. There are a lot of efficient solvers and techniques in the literature to solve this kind of problems, such as CPLEX [161], GUROBI [162] or GLPK [96].

B.2 MIXED INTEGER PROGRAMMING

Mixed Integer Programming (**MIP**) is a process of modelling where a set of variables must take discrete values. Integer programs can be linear, non-linear, static, dynamic, deterministic or indeed stochastic. Let us consider the deterministic linear case. This way, if some but not all variables are integer we have a (Linear) Mixed Integer Program (MILP), written as:

$$\begin{aligned} \min_{x,y} \quad & cx + hy \\ \text{s.t.} \quad & Ax + Gy \leq b \\ & x \geq 0 \\ & y \in \mathbb{Z} \end{aligned} \tag{B.2}$$

where A is a coefficients matrix with dimensions $[n, m]$ and G is another coefficients matrix with dimensions $[m, p]$, c is an n -dimensional row vector and h is a p row vector, b is the right hand side column vector of dimension m . Moreover, x is an n -dimensional column vector of continuous variables and y is also a column vector of p dimension of discrete variables.

Nevertheless, if all variables are integer, the problem becomes a (Linear) Integer Problem (IP), see formulation B.3:

$$\begin{aligned} \min_x \quad & cx \\ \text{s.t.} \quad & Ax \leq b \\ & X \in \mathbb{Z} \end{aligned} \tag{B.3}$$

The last but not the least if all variables are restricted to 0-1 values, the problem is known as a Binary Integer Program (BILP):

$$\begin{aligned} \min_x \quad & cx \\ \text{s.t.} \quad & Ax \leq b \\ & X \in \{0,1\}^n \end{aligned} \tag{B.4}$$

B.3 STOCHASTIC PROGRAMMING

Stochastic programming (**SP**) is commonly used to model problems where a set of parameters are uncertain. In particular, it allows incorporating probability distribution of the uncertain parameters explicitly into the model and provides an opportunity to take corrective actions in the future (recourse) based on the actual outcomes. See formulation B.5, where x is an n -dimensional column vector, ω represents the distribution of the data and f_i is the problem data corresponding to a specific set of x and a certain distribution ω .

$$\begin{aligned} \min_x \quad & F_0(x) = \mathbf{E}f_0(x, \omega) \\ \text{s.t.} \quad & F_i(x) = \mathbf{E}f_i(x, \omega) \leq 0 \quad | \quad i = 1, \dots, m \end{aligned} \tag{B.5}$$

An important feature of **SP** is the type of uncertainty either *exogenous*; where the true parameters values are revealed independently of decisions or *endogenous*; where the parameters realizations are influenced by the decisions. A basic assumption in **SP** is that the probability distribution of the uncertain parameters is known. A common approach is to design scenario trees, where each scenario represents a possible realization of the uncertain parameters and has a probability of occurrence link to it.

One of the main features of a scenario based tree is that each level of the tree represents a stage where the decision makers can evaluate the solution of certain variables and apply corrective action related with the occurrence of the uncertainty.

The typical approach for optimization under exogenous uncertainty is the two-stage case. Here, first-stage decisions are made *here and now* at the beginning of the first time period, without knowing how the uncertainty will reveal. Hence, the decision-maker *wait and see* for the outcome, then at some point the uncertainty is resolved and the decision-maker is able to take corrective actions.

Formulation B.6 represents the same two-stage stochastic model where Ω denotes the set of discretizations of the possible outcomes of the uncertain

parameters (scenarios) and π represents their probability of occurrence. The decisions that are taken before knowing the outcome of the stochastic parameters are modeled by the first stage variables, in this case, x . Moreover, the decisions that are taken to correct or adapt the past decisions once the uncertainty is revealed (ξ_1) are known as recourse variables, in this case, y .

$$\begin{aligned}
 \min_{x,y} \quad & c^T x + \sum_{\omega \in \Omega} \pi_{\omega} d_{\omega}^T y_{\omega} \\
 \text{s.t.} \quad & Ax = b \\
 & T_{\omega} x + W_{\omega} y_{\omega} = h_{\omega}, \quad \forall \omega \in \Omega \\
 & x \geq 0 \\
 & y_{\omega} \geq 0, \quad \forall \omega \in \Omega
 \end{aligned} \tag{B.6}$$

Nevertheless, in practise, multi-stage is even more common than two-stage. This argument implies that decision-makers needs additional *here and now* decisions at the beginning of each time period.

The available evidence extracted from the real problems in the world seems to suggest that a time horizon is not enough for tackling strategical, tactical and operational decisions. For the sake of the discussion, I would like to argue that introducing more than one-time horizon inside a multistage tree it is not affordable due to the combinatory explosion, the complexity and the size of the tree. An alternative formulation of the problem that combines suitable two-time scales was proposed by Knaut. This methodology is known as a multi-horizon approach, and it is a clever and stylized idea for embedding strategical and operation decision inside a solvable and tractable scenario tree.

LAGRANGIAN DECOMPOSITION

Lagrangian decomposition is a method that exploits the theorems of Lagrangian relaxation and the Lagrangian dual. Let us consider a general mixed 0-1 multi-stage stochastic model with a given number of scenarios ω . Scenario clustering techniques proposed by Laureano are also valid for this example. However, for a sake of simplicity, let us consider single scenarios. Formulation C.1 represents the Lagrangian relaxation of the non-anticipative constraints (NAC) of the problem. The goodness of this method is the elimination of $2 * |x| * |\delta| * \Omega$ constraints with the addition of a set of parametric terms into the objective function. Although it seems to be at least as complex than before, the reality is very different, now we get a large parametric objective function, but we have less constraint and the same number of variables. Thus, the model is faster to resolve, approximating the values of the dual parameters.

$$\begin{aligned}
\min_{x,y,\delta,\gamma} \quad & c_1^T x + c_2^T \delta + \sum_{\omega \in \Omega} \pi_\omega q_1^{T\omega} \gamma_\omega + q_2^{T\omega} y_\omega \\
& + \sum_{\omega \in \Omega-1} \mu_\delta(\delta^\omega - \delta^{\omega+1}) + \mu_\delta(\delta^\Omega - \delta^1) \\
& + \sum_{\omega \in \Omega-1} \mu_x(x^\omega - x^{\omega+1}) + \mu_x(x^\Omega - x^1) \\
\text{s.t.} \quad & A \begin{pmatrix} \delta \\ x \end{pmatrix} = b \\
& T_\omega \begin{pmatrix} \delta \\ x \end{pmatrix} + W_\omega \begin{pmatrix} \gamma_\omega \\ y_\omega \end{pmatrix} = h_\omega, \quad \forall \omega \in \Omega \\
& x \geq 0, \delta \in \{0,1\} \\
& y_\omega \geq 0, \gamma_\omega \in \{0,1\} \quad \forall \omega \in \Omega
\end{aligned} \tag{C.1}$$

The question of how the Lagrangian dual is solved has caused much debate in the field over the years. Several methods can be applied to dynamically update the Lagrangian dual to get closer the optimal solution. A brief review of the common methods is presented below:

C.1 SUBGRADIENT METHOD

The procedure proceeds to update first the objective function of the model $LR(\omega)$ from scenario 1 to scenario Ω with the value μ_ω^k , where k represents the current iteration of the method and ω represents a specific scenario, $\omega \in \Omega$. Then, these updated $LR(\omega)$ problems are solved. After this step, the sum z_{LR}^k is computed. Hence, at this point, it is premised on the presumption of equation C.2.

$$z_{LR}^k = \sum_{\omega=1}^{\Omega} z_{LR(\omega)}^k \quad (C.2)$$

Next, all the subgradients S^k are computed, taking into account the solution of the first stage variables, the computation is performed using Algorithm 1.

Algorithm 1 Compute S^k (Subgradient Method)

```

1: for all  $\omega \in \Omega$  do
2:   for all  $i \in x^{(k)\omega}$  and  $j \in \delta^{(k)\omega}$  do
3:     if  $\omega == \Omega$  then
4:        $S_{\Omega}^k(i, j) = \binom{\delta}{x}^{(k)\Omega}(j) - \binom{\delta}{x}^{(k)1}(j)$ 
5:     else
6:        $S_{\omega}^k(i, j) = \binom{\delta}{x}^{(k)\omega}(j) - \binom{\delta}{x}^{(k)\omega+1}(j)$ 
7:     end if
8:   end for
9: end for

```

Equation C.3 represents the formula to update the Lagrangian multipliers at each iteration.

$$\mu^{k+1} = \mu^k + \alpha^k \cdot b \cdot S^k \quad (C.3)$$

Algorithm 2 Compute μ^k

```

1: for all  $\omega \in \Omega$  do
2:   for all  $i \in x^{(k)\omega}$  and  $j \in \delta^{(k)\omega}$  do
3:      $\mu_{\omega}^{k+1}(i, j) = \mu^{(k)\Omega}(i, j) + (b \cdot S^{(k)\Omega}(i, j))$ 
4:   end for
5: end for

```

Whereas equation C.4 describes the formula to compute the step direction at each iteration. \bar{z}_{LR} represents an upper bound of the solution value and α^k represents a real parameter to increase or decrease the speed of the movement.

$$b = \frac{\bar{z}_{LR} - z_{LR}^k}{\|S^k\|^2} \quad (\text{C.4})$$

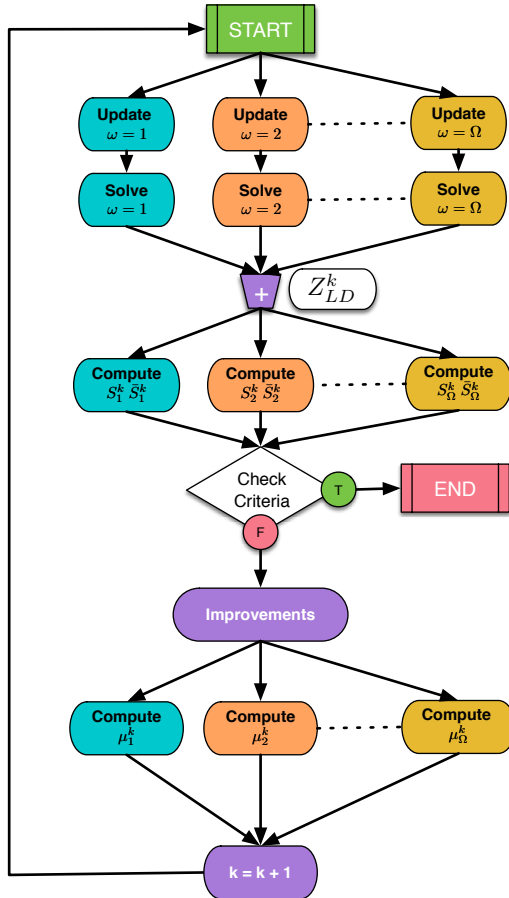


FIGURE C.1: Parallel scheme for the Lagrangian Decomposition method using the Subgradient Method.

Once all the subgradients are computed, the procedure evaluates its status at iteration k using the current solution. This process consists of checking whether

the algorithm is able to improve the solution in future iterations. The criteria used for taking this decision are deeply explained in [48]. After this phase, in case that the algorithm improves the current solution, some convergence parameters are updated so as to boost the ending of the method. For detailed information about the choice and the improvement of these parameters, check [48] and [49] too.

Following these steps, the method updates the matrix μ^{k+1} this procedure is described in Algorithm (2). On logical grounds, there is no compelling reason to argue that this method is not suitable for a parallel scheme. So, in Figure C.1 it represents a parallel approach to solve the overall decomposition algorithm in an efficient way.

C.2 VOLUME ALGORITHM

The procedure called Volume Algorithm is a slight modification of the Subgradient Algorithm presented above. The base theorem of this method is that in the neighbourhood of an optimal solution, the probability to produce a face i is μ_i ; this is an optimal solution of the global primal problem. Thus, Barahona et.al in [163] modifies the subgradient method so as to estimate these probabilities.

Let us start considering the same structure as Subgradient Method and also \bar{Z}_{LD} and $\bar{\mu}$ to store the best coefficients till the current moment. Let also f^k be a real parameter related to the convex combination for obtaining the incumbent solution at each iteration; $f^k \in (0, 1)$. Thus, equation C.5 describes how the neighbourhood solution is computed considering this parameter at each iteration. The overall procedure is depicted in Figure C.2.

$$(\bar{\delta}, \bar{x}, \bar{\gamma}, \bar{y}) = f^k(\delta^k, x^k, \gamma^k, y^k) + (1 - f^k)(\bar{\delta}, \bar{x}, \bar{\gamma}, \bar{y}) \quad (\text{C.5})$$

First of all the subgradients S^k and \bar{S}^k are computed, taking into account the solution of the first stage variables, the computation is performed using Algorithms 1 and 3.

In contrast to the subgradient method, VA uses \bar{S}^k and $\bar{\mu}$ to update the Lagrangian multipliers, check equation C.6.

$$\mu^{k+1} = \bar{\mu}^k + \alpha^k \cdot b \cdot \bar{S}^k \quad (\text{C.6})$$

Moreover, the formula to obtain the new step direction is fairly different, check (eq. C.7).

$$b = \frac{\bar{Z}_{LR} - \bar{Z}(\mu)}{\|\bar{S}^k\|^2} \quad (\text{C.7})$$

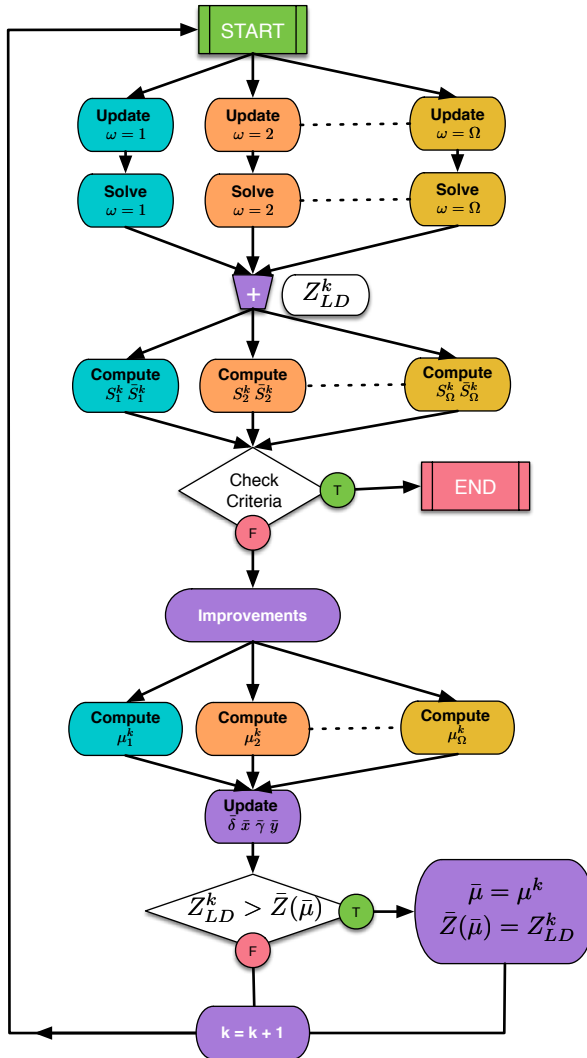


FIGURE C.2: Parallel scheme for the Lagrangian Decomposition method using the Volumn Algorithm.

C.3 PROGRESSIVE HEDGING

The basic ideas of Progressive Hedging Algorithm can also be used to update the Lagrangian multipliers. The procedure needs the same information as Sub-

Algorithm 3 Compute \bar{S}^k (Volume Algorithm)

```

1: for all  $\omega \in \Omega$  do
2:   for all  $i \in \bar{x}^{(k)\omega}$  and  $j \in \bar{\delta}^{(k)\omega}$  do
3:     if  $\omega == \Omega$  then
4:        $\bar{S}_{\Omega}^k(i, j) = \binom{\bar{\delta}}{\bar{x}}^{(k)\Omega}(j) - \binom{\bar{\delta}}{\bar{x}}^{(k)1}(j)$ 
5:     else
6:        $\bar{S}_{\omega}^k(i, j) = \binom{\bar{\delta}}{\bar{x}}^{(k)\omega}(j) - \binom{\bar{\delta}}{\bar{x}}^{(k)\omega+1}(j)$ 
7:     end if
8:   end for
9: end for

```

gradient method. However, instead of computing only the basic subgradient, the procedure takes advantage of the progressive hedging subgradient and step direction formulas.

The overall procedure is depicted in Figure C.3. The solution estimation required by PHA is given by:

$$\hat{\delta}^{(k+1)} = \sum_{\omega \in \Omega} \pi_{\omega} \delta^{(k+1)\omega} \quad (\text{C.8})$$

$$\hat{x}^{(k+1)} = \sum_{\omega \in \Omega} \pi_{\omega} x^{(k+1)\omega} \quad (\text{C.9})$$

Correspondingly, the subgradients S^k and \hat{S}^k are computed, taking into account the solution of the first stage variables, the computation is performed using Algorithms 1 and 4.

Algorithm 4 Compute \hat{S}^k (Progressive Hedging)

```

1: for all  $\omega \in \Omega$  do
2:   for all  $i \in \hat{x}^{(k)\omega}$  and  $j \in \hat{\delta}^{(k)\omega}$  do
3:     if  $\omega == \Omega$  then
4:        $\hat{S}_{\Omega}^k(i, j) = \binom{\hat{\delta}}{\hat{x}}^{(k)\Omega}(j) - \binom{\hat{\delta}}{\hat{x}}^{(k)1}(j)$ 
5:     else
6:        $\hat{S}_{\omega}^k(i, j) = \binom{\hat{\delta}}{\hat{x}}^{(k)\omega}(j) - \binom{\hat{\delta}}{\hat{x}}^{(k)\omega+1}(j)$ 
7:     end if
8:   end for
9: end for

```

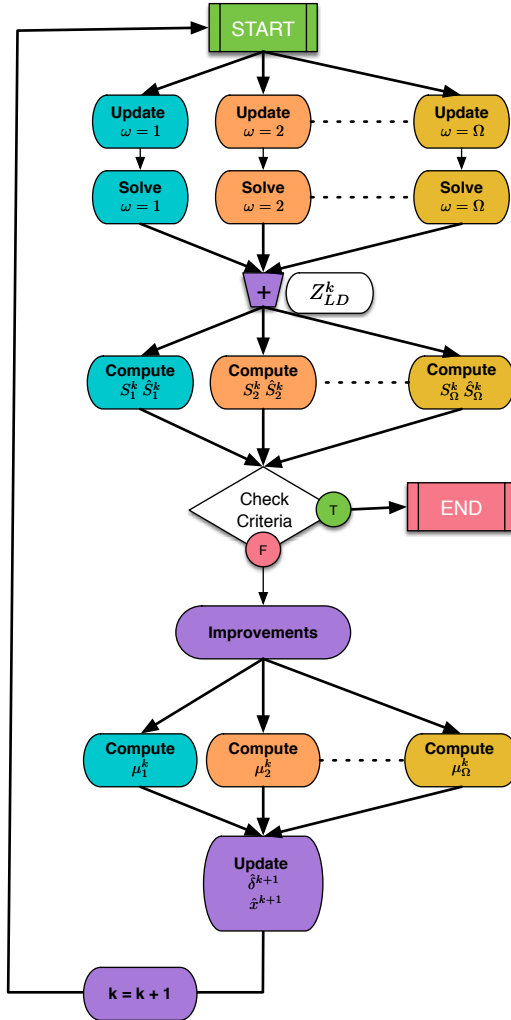


FIGURE C.3: Parallel scheme for the Lagrangian Decomposition method using the Progressive Hedging Algorithm.

As it is said, PHA uses \hat{S}^k to update the Lagrangian multipliers, check equation C.10.

$$\mu^{k+1} = \bar{\mu}^k + a^k \cdot b \cdot \hat{S}^k \quad (\text{C.10})$$

Moreover, the formula to obtain the new step direction is described in equation C.11.

$$b = \frac{\bar{z}_{LR} - z_{LR}^k}{\|\hat{S}^k\|^2} \tag{C.11}$$

C.4 DYNAMIC CONSTRAINED CUTTING PLANE METHOD

The Dynamic Constrained method is related to the trust region methods. Thus, this method is going to build a feasible region for the Lagrangian multipliers at each iteration and manage a set of hyper-planes (cuts). Let us consider i as an hyper-plane belonging to the set I and s^i the subgradient corresponding to this cutting plane. This method is based on the truncation of a Taylor expansion series.

$$\begin{aligned} & \max_{\mu \in C^k} Z \\ \text{s.t.} \quad & Z \leq Z_{LD}(\mu^i) + \sum_{\omega \in \Omega} (\mu^\omega - \mu^{i,\omega})s^i, \quad \forall i \in I \end{aligned} \tag{C.12}$$

Logically, C^k denotes the feasible region of the Lagrangian multipliers at iteration k . Equation C.13 describes how to compute this region.

$$C^k = \mu, \mu^k \leq \mu \leq \bar{\mu}^k \tag{C.13}$$

Moreover, Equations C.14 and C.15 represents the limits of the region. Note, that b (the step direction) is computed equally as the subgradient method, see Equation C.4.

$$\underline{\mu}_j^{k+1} = \mu_j^k - \alpha^k b^k |s_j^k| \tag{C.14}$$

$$\bar{\mu}_j^{k+1} = \mu_j^k + \alpha^k b^k |s_j^k| \tag{C.15}$$

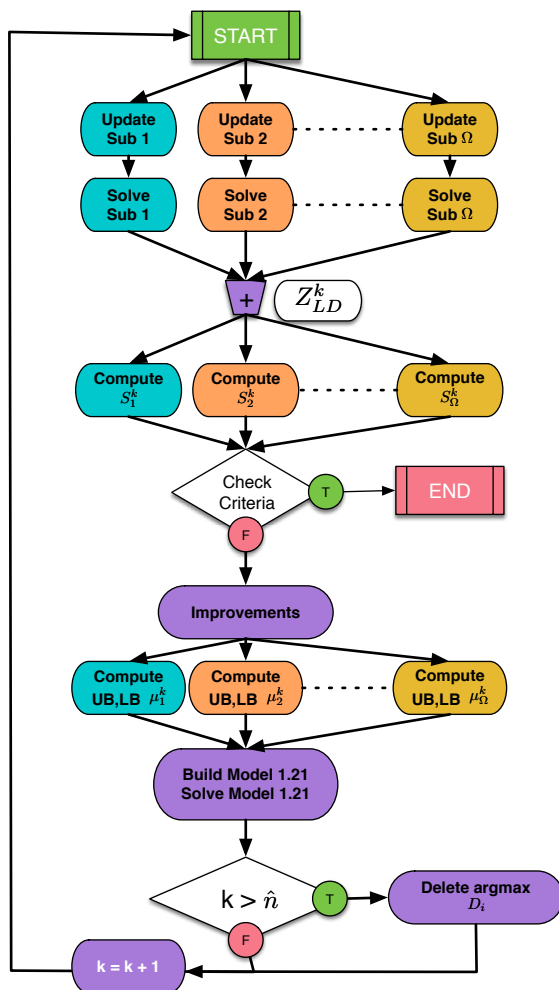


FIGURE C.4: Parallel scheme for the Lagrangian Decomposition method using the Dynamic Constrained Cutting Plane.

PUBLICATIONS

The following publications have been derived from the work on this thesis.

D.1 JOURNAL PUBLICATIONS

The following publications in research journals are derived from the work in this thesis:

1. Mateo-Fornés, J., Plà-Aragonès, L. M., Solsona, F. & Pagès-Bernaus, A. A scalable parallel implementation of the Cluster Benders Decomposition algorithm. *Cluster Computing* (2018).
2. Mateo-Fornés, J., Plà-Aragonès, L. M., Solsona, F. & Pagès-Bernaus, A. A production planning model considering uncertain demand using two-stage stochastic programming in a fresh vegetable supply chain context. *SpringerPlus* **5**, 1 (2016).
3. Mateo-Fornés, J., Vilaplana, J., Plà-Aragonès, L. M., Lèrida, J. L. & Solsona, F. A green strategy for federated and heterogeneous clouds with communicating workloads. *The Scientific World Journal* **2014** (2014).

D.2 CONFERENCE CONTRIBUTIONS

4. Pagès-Bernaus, A., Ortíz-Araya, V., Plà-Aragonés, L. M., Mateo-Fornés, J. & Solsona, F. *Optimizing feed formulations: from the market to production in II International Conference on Agro Big Data and Decision Support Systems in Agriculture* (2018).
5. Pagès-Bernaus, A., Miquel Plà-Aragonés, L., Mateo-Fornés, J. & Solsona, F. *Optimal timing for sending pigs to the abattoir: a stochastic programming approach in International Conference on Computational Management Science* (2018).
6. Pagès-Bernaus, A., Ortíz-Araya, V., Plà-Aragonés, L. M., Mateo-Fornés, J., Solsona, F. & Florensa, D. *Multi-diet formulation model under activity-based costing for animal feed production in 29th European Conference on Operational Research* (2018).

7. Mishra, S., Bordin, C., Mateo-Fornés, J. & Palu, I. *Reliability framework for power network assessment in International Conference on Renewable Energy and Environment Engineering* (2018).
8. Mateo-Fornés, J., Borrell, K., Solsona, F., Plà-Aragonés, L. M. & Pagès-Bernaus, A. *Deterministic linear optimization as a service in 21 th Conference of the International Federation of Operational Research Societies* (2017).
9. Pagès-Bernaus, A., Plà-Aragonés, L. M., Mateo-Fornés, J. & Solsona, F. *Dynamic diet formulation responsive to price changes: a feed mill perspective in II International Conference on Agro BigData and Decision Support Systems in Agriculture* (2017).
10. Mateo-Fornés, J., Soto-Silva, W., Gonzalez-Araya, M., Solsona, F., Plà-Aragonés, L. M., Pagès-Bernaus, A. & Pifarré, M. *A new cloud DSS for tactical planning in a FSC in II International Conference on Agro BigData and Decision Support Systems in Agriculture.* (2017).
11. Pagès-Bernaus, A., Plà-Aragonés, L. M., Oriz-Araya, V., Mateo-Fornés, J. & Solsona, F. *Optimal scheduling of production tasks in an animal feed mill in 12th Metaheuristics International Conference* (2017).
12. Plà-Aragonés, L. M., Mateo-Fornés, J., Soto-Silva, W., Gonzalez-Araya, M. & Solsona, F. *Stochastic contribution to the Fruit Supply Chain planning in 28th European Conference on Operational Research* (2016).
13. Castellà, G., Boix, R., Colls, C., García-Altés, A., Teixidó, I., Mateo-Fornés, J., Solsona, F., Escarrabil, J., Sánchez-de-la-Torre, M., Barbé, F. & Valls, J. *Assessing the heterogeneity in the individuals to fit different models: an application to predict mortality in a large sample of sleep apnoea patients in Biostatnet Workshop on Biomedical (Big) Data* (2015).
14. Mateo-Fornés, J., Solsona, F., Lèrida, J. L. & Plà-Aragonés, L. M. *Parallel Lagrangian Decomposition for large-scale two-stage stochastic mixed 0-1 problems. in 15th International Conference Computational and Mathematical Methods in Science and Engineering* (2015).
15. Mateo-Fornés, J., Solsona, F., Plà-Aragonès, L. M. & Lèrida, J. L. *Comparative Study between the Parallelization of Cluster Benders Decomposition and the Parallelization of Lagrange Decomposition Applied to Stochastic Discrete Models in 27th European Conference on Operational Research* (2015).

16. Solsona, F., Mateo-Fornés, J., Plà-Aragonès, L. M. & Lèrida, J. L. *A Green Scheduling Policy Model for Federated Clouds in 27th European Conference on Operational Research* (2015).
17. Solsona, F., Mateo-Fornés, J., Plà-Aragonès, L. M. & Lèrida, J. L. *Parallelization and High Performance Computing adjustment of the Cluster Benders Decomposition algorithm in 20th Conference of the International Federation of Operational Research Societies* (2014).
18. Vilaplana, J., Mateo-Fornés, J., Teixido, I. & Solsona, F. *A Green Job Scheduling Policy for Heterogeneous Clouds in 14th International Conference Computational and Mathematical Methods in Science and Engineering* (2014).
19. Vilaplana, J., Solsona, F., Teixido, I., Abella, F., Mateo-Fornés, J. & Rius, J. *An SLA&Power Aware Strategy for a Cloud in International Conference on Information Technology and Management Engineering* (2014).
20. Lladós, J., Mateo-Fornés, J., Cores, F., Lèrida, J. L. & Gine, F. *Incentivación del aprendizaje de la programación en las ingenierías. Un caso práctico in XXIV Jornadas de paralelismo* (2013).
21. Mateo-Fornés, J., Lladós, J., Lèrida, J. L., Plà-Aragonès, L. M. & Solsona, F. *Paralelización del algoritmo de descomposición de cluster benders in XXIV Jornadas de paralelismo* (2013).
22. Plà-Aragonès, L. M., Solsona, F., Lèrida, J. L. & Mateo-Fornés, J. *Parallelisation of the Cluster Benders Decomposition method in 26th European Conference on Operational Research* (2013).
23. Solsona, F., Plà-Aragonès, L. M., Mateo-Fornés, J. & Lèrida, J. L. *Parallelisation of the so-called Cluster Benders Decomposition algorithm for solving two-stage stochastic linear problems in 25th European Conference on Operational Research* (2012).

D.3 OTHER CONTRIBUTIONS

1. Vilaplana, J., Alves, R., Solsona, F., Mateo, J., Teixidó, I. & Pifarré, M. MetReS, an Efficient Database for Genomic Applications. *Journal of Computational Biology* **25** (2018).
2. Piñol, M., Alves, R., Teixidó, I., Mateo, J., Solsona, F. & Vilapinyó, E. Rare Disease Discovery: An Optimized Disease Ranking System. *IEEE Transactions on Industrial Informatics* **13** (2017).

3. Cuadrado, J., Vilaplana, J., Mateo, J., Solsona, F., Solsona, S., Rius, J., Alves, R. & Camafort, M. HBPF: a Home Blood Pressure Framework with SLA guarantees to follow up hypertensive patients. *PeerJ Computer Science* **2**, e69 (2016).
4. Carrera, A., Pifarré, M., Vilaplana, J., Cuadrado, J., Solsona, S., Mateo, J. & Solsona, F. A mobile app to monitor hypertensive patients. *Applied Clinical Informatics* **7** (2016).
5. Vilaplana, J., Solsona, F., Abella, F., Cuadrado, J., Teixido, I., Mateo, J. & Rius, J. H-PC: A Cloud Computing Tool for Supervising Hypertensive Patients. *Journal of Supercomputing* **71**, 591 (2015).
6. Vilaplana, J., Mateo, J., Teixido, I., Solsona, F., Giné, F. & Roig, C. An SLA and power-saving scheduling consolidation strategy for shared and heterogeneous clouds. *Journal of Supercomputing* **71**, 1817 (2015).
8. Vilaplana, J., Solsona, F., Aballa, F., Cuadrado, J., Alves, R. & Mateo, J. S-PC: An e-treatment application for management of smoke-quitting patients. *Computer Methods and Programs in Biomedicine* **115**, 33 (2014).
9. Vilaplana, J., Solsona, F., Teixido, I., Mateo, J., Abella, F. & Rius, J. A queuing theory model for cloud computing. *Journal of Supercomputing* **69**, 492 (2014).
11. Vilaplana, J., Solsona, F., Mateo, J. & Teixido, I. SLA-Aware Load Balancing in a Web-Based Cloud System over OpenStack. *Lecture Notes in Computer Science* **8377**, 281 (2014).
12. Mateo, J., Lerida, J. L., Guirado, F., Gonzalez, M., Colomer, A. & Filgueira, R. Design of a prototype for modelling membrane computation based on ecosystems in a cloud environment. *Iberian Journal of Information Systems and Technologies*, 1 (2014).

NOMENCLATURE

Research Stay: Stochastic Tree Components

- δ_o Vector containing the corresponding fattening week, where $o \in O$.
- γ_n Vector containing the corresponding price, where $n \in N$.
- Ω Number of stochastic scenarios.
- π_n Vector containing the corresponding probability, where $n \in N$.
- N Set of nodes in the stochastic tree.
- oN Set of nodes that belongs to the operational horizon.
- oS Set of nodes that belongs to the strategical horizon.
- P_n Vector containing the parent nodes, where $n \in N$.
- SO_o Vector containing the corresponding tree stage, where $o \in O$.
- SS_s Vector containing the corresponding tree stage, where $s \in S$.

Research Stay: Fattening Pigs Farms Parameters

- Alt Other costs.
- C Set of growth stages.
- Cg Cost of growing a pig.
- F Set of fattening farms.
- $FB_{w,c}$ Matrix with the bonus of pigs, where $w \in FW$, $c \in C$.
- fC Food Cost.
- $FC_{w,c}$ Matrix with the cost of pigs, where $w \in FW$, $c \in C$.
- fw Size of fattening window.
- $FW_{w,c}$ Matrix with the weight of pigs, where $w \in FW$, $c \in C$.
- $m\omega$ Size of market window.

qF_f Max capacity of the farms, where $f \in F$.

qS_f Pig stock at each category, where $f \in F$.

qT_f Truck Capacity.

tC Truck cost.

Research Stay: Strategic Decisions

sW_s Number of weeks to close the batch in the strategic node s .

Research Stay: Operational Decisions

$d_{f,s,o,c}$ Indicator of whether animals from farm f and group c are sent to abattoir for strategic node s and operational node o , where $d_{f,s,o,c} \in (0,1)^+$.

$n_{f,s,o,c}$ Number of animals in farm f at growth stage c for strategic node s and operational node o .

$x_{f,s,o,c}$ Number of animals transfer to abattoir from farm f at growth stage c for strategic node s and operational node o .

y_f Number of trucks required to transfer the pigs, where $y_f \in \mathbb{Z}^+$, $\forall f \in F$.

$z1_{f,s,o,c}$ Indicator whether the batch is started, where $z1_{f,s,o,c} \in (0,1)^+ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C$.

$z2_{f,s,o,c}$ Indicator whether all batch is sent to the abattoir, where $z2_{f,s,o,c} \in (0,1)^+ \forall f \in F, \forall s \in sN, \forall o \in oN, \forall c \in C$.

Paper 5: Indices and index sets

Ω Set of different uncertain scenarios. $\omega \in \Omega$

C Set of warehouses. $c \in C$

L Set of trucks. $l \in L$

N Set of cold storages inside a warehouse. $n \in N$

P Set of available producers. $p \in P$

Q Set of apple varieties. $q \in Q$

T Set of apple types in sense of storage times. $t \in T$

Paper 5: Parameters

- π^ω Probability of scenario ω .
- CA_c Fixed cost for storing inside warehouse c .
- CC_{pq}^0 Cost for purchasing in the first stage; from a producer p and the variety q .
- CC_{pq}^ω Cost for purchasing in the second stage; from a producer p and the variety q under scenario ω .
- CE_{cn} Cost for storing using a cold tech inside warehouse c using the cold storage n .
- CF_{cn}^0 Fixed cost for storing using a cold tech inside warehouse c using the cold storage n , in the first stage.
- CF_{cn}^ω Fixed cost for storing using a cold tech inside warehouse c using the cold storage n , in the second stage, under scenario ω .
- CFT_l Cost for using truck l .
- CM_p Cost to maintain a producer p .
- CP_p Cost to transport from a producer p .
- CT_{cq} Cost for transportation; a variety q to the warehouse c .
- D_{qt}^ω Amount of demand for variety q and type t under scenario ω .
- O_{pq} Amount of raw material from variety q produced by producer p .
- $QMax_l$ Maximum capacity of truck l .
- TE_{cn} Type of cold tech used inside warehouse c using room n .

Paper 5: First Stage Decisions

- ME_{cnqt}^0 Cold storage chamber n inside warehouse c is or not contracted to store fruit of quality q to satisfy the demand for a certain type of apple t in sense of storage time. (B)
- XP_{pq}^0 Variety q is or not purchased from producer p . (B)

Paper 5: Second Stage Decisions

- A_c^ω Warehouse c is or not used under scenario ω (B)
- ME_{cnqt}^ω Cold storage chamber n inside warehouse c is or not contracted to store fruit of quality q to satisfy the demand for a certain type of apple t in sense of storage time, in the second stage under scenario ω . (B)
- W_{pqt}^ω Amount of apples purchased from producer p that are used to satisfy the demand for variety q and type t . (\mathbb{Z}^+)
- X_{qtcn}^ω Amount of apples to be stored for variety q and storage time t in cold storage chamber n inside warehouse c . (\mathbb{Z}^+)
- XP_{pq}^ω Variety q is or not purchased from producer p in the second stage under scenario ω . (B)
- Y_{lpq}^ω Number of trips to the processing plant using truck l under scenario ω . (\mathbb{Z}^+)
- YC_{lcn}^ω Number of trips from the processing plant using truck l under scenario ω . (\mathbb{Z}^+)
- Z_p^ω Producer p is or not used to satisfy the dried apple demand. (B)

BIBLIOGRAPHY

1. Kamp, J. Knowledge based systems: from research to practical application: pitfalls and critical success factors. *Computers and Electronics in Agriculture* **22**, 243 (1999).
2. *Stormy* <http://stormy.udl.cat>. Online; Accessed: October, 2018.
3. Martinello, M., Kaâniche, M. & Kanoun, K. Web service availability—impact of error recovery and traffic model. *Reliability Engineering & System Safety* **89**, 6 (2005).
4. *baron* <http://www.minlp.com/home>. Online; Accessed: December, 2018.
5. *couenne* <https://projects.coin-or.org/Couenne>. Online; Accessed: December, 2018.
6. *bonmin* <https://projects.coin-or.org/Bonmin>. Online; Accessed: December, 2018.
7. Wolfert, S., Ge, L., Verdouw, C. & Bogaardt, M.-J. Big Data in Smart Farming – A review. *Agricultural Systems* **153**, 69 (2017).
8. Jiang, S., Chen, T. & Dong, J. in *Computer and Computing Technologies in Agriculture IX: 9th IFIP WG 5.14 International Conference, CCTA 2015, Beijing, China, September 27-30, 2015, Revised Selected Papers, Part II* 60 (Springer International Publishing, Cham, 2016).
9. Park, H., Lee, E.-J., Park, D.-H., Eun, J.-S. & Kim, S.-H. PaaS offering for the big data analysis of each individual APC in *Information and Communication Technology Convergence (ICTC), 2016 International Conference on* (2016), 30.
10. Kamp, J. Knowledge based systems: from research to practical application: pitfalls and critical success factors. *Computers and Electronics in Agriculture* **22**, 243 (1999).
11. Manos, B., Paparrizos, K., Matsatsinis, N. & Papathanasiou, J. *Decision support systems in agriculture, food and the environment: Trends, applications and advances* (2010).

12. Jones, J. W., Antle, J. M., Basso, B., Boote, K. J., Conant, R. T., Foster, I., Godfray, H. C. J., Herrero, M., Howitt, R. E., Janssen, S., Keating, B. A., Munoz-Carpena, R., Porter, C. H., Rosenzweig, C. & Wheeler, T. R. Toward a new generation of agricultural system data, models, and knowledge products: State of agricultural systems science. *Agricultural Systems* **155**, 269 (2017).
13. Plà Aragonés, L. M. *Handbook of Operations Research in Agriculture and the AgriFood Industry* (Springer, 2015).
14. Nguyen, T. A., Kim, D. S. & Park, J. S. Availability modeling and analysis of a data center for disaster tolerance. *Future Generation Computer Systems* **56**, 27 (2016).
15. Xiong, K. & Perros, H. *Service Performance and Analysis in Cloud Computing in 2009 Congress on Services - I* (IEEE, 2009), 693.
16. Slothouber, L. A model of web server performance. *Proceedings of the 5th International World wide web* (1996).
17. Yang, B., Tan, F., Dai, Y. & Guo, S. Performance evaluation of cloud service considering fault recovery. *Cloud computing* (2009).
18. Ma, B. & Mark, J. Approximation of the mean queue length of an M/G/c queueing system. *Operations Research* (1995).
19. Karlapudi, H. & Martin, J. *Web application performance prediction* PhD thesis (Clemson University, 2004).
20. Mei, R. & Meeuwissen, H. Modelling end-to-end Quality-of-Service for transaction-based services in multidomain environment. *Proceedings IEEE International Conference on Web* (2006).
21. Vilaplana, J., Solsona, F., Teixido, I., Mateo, J., Rius, J. & Abella, F. *A green scheduling policy for cloud computing* (2014).
22. Tusler, W. H. Designing for disasters. *Health facilities management* **20**, 33 (2007).
23. Oppenheimer, D. & Patterson, D. Architecture and dependability of large-scale internet services. *IEEE Internet Computing* **6**, 41 (2002).
24. Kalyanakrishnan, M., Iyer, R. & Patel, J. Reliability of Internet hosts: a case study from the end user's perspective. *Computer Networks* **31**, 47 (1999).
25. Mani, D. & Mahendran, A. Availability Modelling of Fault Tolerant Cloud Computing System. *International Journal of Intelligent Engineering and Systems* **10** (2017).

26. Lump, T., Schneider, J., Holtz, J. & Mueller, M. From high availability and disaster recovery to business continuity solutions. *IBM Systems* (2008).
27. Clitherow, D., Brookbanks, M. & Clayton, N. Combining high availability and disaster recovery solutions for critical IT environments. *IBM Systems* (2008).
28. Adeshiyani, T., Attanasio, C. & Farr, E. Using virtualization for high availability and disaster recovery. *IBM Journal of* (2009).
29. Detienne, B. A mixed integer linear programming approach to minimize the number of late jobs with and without machine availability constraints. *European Journal of Operational Research* **235**, 540 (2014).
30. Hashemian, N., Diallo, C. & Vizvári, B. Makespan minimization for parallel machines scheduling with multiple availability constraints. *Annals of Operations Research* **213**, 173 (2014).
31. Kubiak, W., Błażewicz, J., Formanowicz, P., Breit, J. & Schmidt, G. Two-machine flow shops with limited machine availability. *European Journal of Operational Research* **136**, 528 (2002).
32. Pérez-Miguel, C., Mendiburu, A. & Miguel-Alonso, J. Modeling the availability of Cassandra. *Journal of Parallel and Distributed Computing* **86**, 29 (2015).
33. Lakshman, A. & Malik, P. Cassandra. *ACM SIGOPS Operating Systems Review* **44**, 35 (2010).
34. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D., Kaashoek, M., Dabek, F. & Balakrishnan, H. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking* **11**, 17 (2003).
35. Griffith, W. S. Optimal Reliability Modeling: Principles and Applications. *Technometrics* **46**, 112 (2004).
36. Crovella, M. E., Bestavros, A., Crovella, M. E. & Bestavros, A. Self-similarity in World Wide Web traffic. *ACM SIGMETRICS Performance Evaluation Review* **24**, 160 (1996).
37. Almeida, V., Bestavros, A., Crovella, M. & De Oliveira, A. *Characterizing Reference Locality in the WWW in Parallel and Distributed Information Systems (PDIS)* (IEEE Comput. Soc. Press, 1996), 92.
38. Iyengar, A., Squillante, M. & Zhang, L. Analysis and characterization of large-scale Web server access patterns and performance. *World Wide Web* (1999).

39. Morris, R. & Lin, D. *Variance of aggregated Web traffic in INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* **1** (IEEE, 2000), 360.
40. Benders, J. F. Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik* **4**, 238 (1962).
41. Van Slyke, R. M. & Wets, R. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics* **17**, 638 (1969).
42. Wolf, C. & Koberstein, A. Dynamic sequencing and cut consolidation for the parallel hybrid-cut nested L-shaped method. *European Journal of Operational Research* **230**, 143 (2013).
43. Aranburu, L., Escudero, L., Garín, M. & Pérez, G. A so-called Cluster Benders Decomposition approach for solving two-stage stochastic linear problems. *Top* **20**, 279 (2012).
44. Rahmaniani, R., Crainic, T. G., Gendreau, M. & Rei, W. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research* (2016).
45. Nowak, M. P. & Römisich, W. Stochastic Lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. *Annals of Operations Research* **100**, 251 (2000).
46. Ghaddar, B., Naoum-Sawaya, J., Kishimoto, A., Taheri, N. & Eck, B. A Lagrangian decomposition approach for the pump scheduling problem in water networks. *European Journal of Operational Research* **241**, 490 (2015).
47. Rosa, C. H. & Ruszczyński, A. On augmented Lagrangian decomposition methods for multistage stochastic programs. *Annals of Operations Research* **64**, 289 (1996).
48. Escudero, L. F., Garín, M. A., Pérez, G. & Unzueta, A. Scenario cluster decomposition of the Lagrangian dual in two-stage stochastic mixed 0--1 optimization. *Computers & Operations Research* **40**, 362 (2013).
49. Escudero, L. F., Garín, M. A. & Unzueta, A. Cluster Lagrangean decomposition in multistage stochastic optimization. *Computers & Operations Research* **67**, 48 (2016).
50. Dijkhuizen, A. A., Morris, R. S. & Morrow, M. Economic optimization of culling strategies in swine breeding herds, using the "porkchop computer program". *Preventive Veterinary Medicine* **4**, 341 (1986).

51. Huirne, R., Dijkhuizen, A. & Renkema, J. Economic optimization of sow replacement decisions on the personal computer by method of stochastic dynamic programming. *Livestock Production Science* **28**, 331 (1991).
52. Huirne, R. B. M. & Hardaker, J. B. A multi-attribute utility model to optimise sow replacement decisions. *European Review of Agricultural Economics* **25**, 488 (1998).
53. Kirchner, K., Tölle, K. H. & Krieter, J. Decision tree technique applied to pig farming datasets. *Livestock Production Science* **90**, 191 (2004).
54. Kristensen, A. R. & Søllested, T. A. A sow replacement model using Bayesian updating in a three-level hierarchic Markov process: I. Biological model. *Livestock Production Science* **87**, 13 (2004).
55. Kristensen, A. R. & Søllested, T. A. A sow replacement model using Bayesian updating in a three-level hierarchic Markov process: II. Optimization model. *Livestock Production Science* **87**, 25 (2004).
56. Rodríguez, S. V., Jensen, T. B., Plà, L. M. & Kristensen, A. R. Optimal replacement policies and economic value of clinical observations in sow herds. *Livestock Science* **138**, 207 (2018).
57. Plà, L. M., Pomar, C. & Pomar, J. A sow herd decision support system based on an embedded Markov model. *Computers and Electronics in Agriculture* **45**, 51 (2004).
58. Rodríguez, S. V., Albornoz, V. M. & Plà, L. M. A two-stage stochastic programming model for scheduling replacements in sow farms. *TOP* **17**, 171 (2009).
59. Soto-Silva, W. E., Nadal-Roig, E., González-Araya, M. C. & Pla-Aragones, L. M. Operational research models applied to the fresh fruit supply chain. *European Journal of Operational Research* **251**, 345 (2016).
60. Hammervoll, T. Value-creation logic in supply chain relationships. *Journal of Business-to-Business Marketing* **16**, 220 (2009).
61. Chai, J., Liu, J. N. & Ngai, E. W. Application of decision-making techniques in supplier selection: A systematic review of literature. *Expert Systems with Applications* **40**, 3872 (2013).
62. Guneri, A. F., Yucel, A. & Ayyildiz, G. An integrated fuzzy-lp approach for a supplier selection problem in supply chain management. *Expert Systems with Applications* **36**, 9223 (2009).
63. Terrazas, R., Kecojevic, V. & Komljenovic, D. Multi-attribute fuzzy methodology for the selection of mining shovels (2008).

64. Zimmer, K., Fröhling, M. & Schultmann, F. Sustainable supplier management—a review of models supporting sustainable supplier selection, monitoring and development. *International Journal of Production Research* **54**, 1412 (2016).
65. Anojkumar, L., Ilangkumaran, M. & Sasirekha, V. Comparative analysis of MCDM methods for pipe material selection in sugar industry. *Expert Systems with Applications* **41**, 2964 (2014).
66. Rong, A., Akkerman, R. & Grunow, M. An optimization approach for managing fresh food quality throughout the supply chain. *International Journal of Production Economics* **131**, 421 (2011).
67. Nakandala, D., Lau, H. & Zhao, L. Development of a hybrid fresh food supply chain risk assessment model. *International Journal of Production Research*, 1 (2016).
68. Osvald, A. & Stirn, L. Z. A vehicle routing algorithm for the distribution of fresh vegetables and similar perishable food. *Journal of food engineering* **85**, 285 (2008).
69. Nadal-Roig, E. & Plà-Aragónés, L. M. in *Handbook of Operations Research in Agriculture and the Agri-Food Industry* 163 (Springer, 2015).
70. Plà, L. M., Sandars, D. L. & Higgins, A. J. A perspective on operational research prospects for agriculture. *Journal of the Operational Research Society* **65**, 1078 (2014).
71. Borodin, V., Bourtembourg, J., Hnaïen, F. & Labadie, N. Handling uncertainty in agricultural supply chain management: a state of the art. *European Journal of Operational Research* **254**, 348 (2016).
72. Tan, B. & Çömnden, N. Agricultural planning of annual plants under demand, maturation, harvest, and yield risk. *European Journal of Operational Research* **220**, 539 (2012).
73. Munhoz, J. R. & Morabito, R. Optimization approaches to support decision making in the production planning of a citrus company: a Brazilian case study. *Computers and Electronics in Agriculture* **107**, 45 (2014).
74. Lin, Y., Pan, F. & Srivastava, A. A Linear Programming Optimization Model Of Woody Biomass Logistics Integrating Infield Drying As A Cost Saving Pre-Process In Michigan. <http://dx.doi.org/10.13073/FPJ-D-15-00077> (2016).

75. Franco, J. F., Rider, M. J. & Romero, R. A Mixed-Integer Linear Programming Model for the Electric Vehicle Charging Coordination Problem in Unbalanced Electrical Distribution Systems. *IEEE Transactions on Smart Grid* **6**, 2200 (2015).
76. Amin, S. H. & Zhang, G. A multi-objective facility location model for closed-loop supply chain network under uncertain demand and return. *Applied Mathematical Modelling* **37**, 4165 (2013).
77. Liu, S., Alhasan, I. & Papageorgiou, L. G. A mixed integer linear programming model for the optimal operation of a network of gas oil separation plants. *Chemical Engineering Research and Design* **111**, 147 (2016).
78. López-Milán, E. & Plà-Aragonés, L. M. A decision support system to manage the supply chain of sugar cane. *Annals of Operations Research* **219**, 285 (2014).
79. Soto-Silva, W. E., Nadal-Roig, E., González-Araya, M. C. & Pla-Aragones, L. M. Operational research models applied to the fresh fruit supply chain. *European Journal of Operational Research* **251**, 345 (2016).
80. Mateo, J., Pla, L. M., Solsona, F. & Pagès, A. A production planning model considering uncertain demand using two-stage stochastic programming in a fresh vegetable supply chain context. *SpringerPlus* **5**, 839 (2016).
81. Mansini, R., Ogryczak, W. & Speranza, M. G. *Linear and Mixed Integer Programming for Portfolio Optimization* (Springer International Publishing, Cham, 2015).
82. Mansini, R., Ogryczak, W. & Speranza, M. G. in *Linear and Mixed Integer Programming for Portfolio Optimization* 47 (Springer, 2015).
83. Luathep, P., Sumalee, A., Lam, W. H., Li, Z.-C. & Lo, H. K. Global optimization method for mixed transportation network design problem: A mixed-integer linear programming approach. *Transportation Research Part B: Methodological* **45**, 808 (2011).
84. KONG, H.-n. A Green Mixed Integer Linear Programming Model for Optimization of Byproduct Gases in Iron and Steel Industry. *Journal of Iron and Steel Research, International* **22**, 681 (2015).
85. Nadal-Roig, E. & Pla, L. M. Multiperiod planning tool for multisite pig production systems. *Journal of Animal Science* **92**, 4154 (2014).
86. de Carvalho Junior, J., Koppe, J. & Costa, J. A case study application of linear programming and simulation to mine planning. *Journal of the Southern African Institute of Mining and Metallurgy* **112**, 477 (2012).

87. Kareem, B. & Aderoba, A. A. Linear Programming based Effective Maintenance and Manpower Planning Strategy: A Case Study. *International Journal of The Computer, the Internet and Management* **16**, 26 (2008).
88. Fonseca, A., Sá, V., Bento, H., Tavares, M. L., Pinto, G. & Gomes, L. A. Hydrogen distribution network optimization: a refinery case study. *Journal of Cleaner Production* **16**, 1755 (2008).
89. Kuzle, I., Zdrilic, M. & Pandzic, H. *Virtual power plant dispatch optimization using linear programming* in *2011 10th International Conference on Environment and Electrical Engineering* (IEEE, 2011), 1.
90. Coutinho, E. F., de Carvalho Sousa, F. R., Rego, P. A. L., Gomes, D. G. & de Souza, J. N. Elasticity in cloud computing: a survey. *annals of telecommunications - annales des télécommunications* **70**, 289 (2015).
91. Han, R., Ghanem, M. M., Guo, L., Guo, Y. & Osmond, M. Enabling cost-aware and adaptive elasticity of multi-tier cloud applications. *Future Generation Computer Systems* **32**. Special Section: The Management of Cloud Systems, Special Section: Cyber-Physical Society and Special Section: Special Issue on Exploiting Semantic Technologies with Particularization on Linked Data over Grid and Cloud Architectures, 82 (2014).
92. *NEOS SERVER* <https://neos-server.org/neos/>. Online; Accessed: December, 2018.
93. *IBM DO CLOUD* <http://onboarding-oaas.docloud.ibmcloud.com/software/analytics/docloud/>. Online; Accessed: December, 2018.
94. *OpenNebula* <http://opennebula.org/>. Online; Accessed: December, 2018.
95. *lpsolve* <http://lpsolve.sourceforge.net/5.5/>. Online; Accessed: December, 2018.
96. *glpk* <https://www.gnu.org/software/glpk/>. Online; Accessed: December, 2018.
97. *cbc* <https://projects.coin-or.org/Cbc>. Online; Accessed: December, 2018.
98. *symphony* <https://projects.coin-or.org/SYMPHONY>. Online; Accessed: December, 2018.
99. *clp* <https://projects.coin-or.org/Clp>. Online; Accessed: December, 2018.
100. *Dip* <https://projects.coin-or.org/Dip>. Online; Accessed: December, 2018.

101. *ipopt* <https://projects.coin-or.org/Ipopt>. Online; Accessed: December, 2018.
102. *Pyomo* <https://software.sandia.gov/trac/pyomo>. Online; Accessed: December, 2018.
103. *AngularJS — Superheroic JavaScript MVW Framework* <https://angularjs.org/>. Online; Accessed: December, 2018.
104. *Bower — a package manager for the web* <https://bower.io/>. Online; Accessed: December, 2018.
105. *Grunt: The JavaScript Task Runner* <http://gruntjs.com/>. Online; Accessed: December, 2018.
106. *Yeoman - The web's scaffolding tool for modern webapps* <http://yeoman.io/>. Online; Accessed: December, 2018.
107. *Spring* <https://spring.io/>. Online; Accessed: December, 2018.
108. *Maven – Welcome to Apache Maven* <https://maven.apache.org/>. Online; Accessed: December, 2018.
109. *Hibernate. Everything data.* <http://hibernate.org/>. Online; Accessed: December, 2018.
110. *MIPLIB* <http://miplib.zib.de/>. Online; Accessed: December, 2018.
111. *H. Mittelmann. Benchmarks for optimization software* <http://plato.asu.edu/bench.html>. Online; Accessed: December, 2018.
112. Meindl, B. & Templ, M. *Analysis of Commercial and Free and Open Source Solvers for the Cell Suppression Problem* tech. rep. (2013), 147.
113. Dabbene, F., Gay, P. & Sacco, N. Optimisation of fresh-food supply chains in uncertain environments, Part I: Background and methodology. *Biosystems Engineering* **99**, 348 (2008).
114. Verdouw, C., Beulens, A. J., Trienekens, J. & Verwaart, T. Mastering demand and supply uncertainty with combined product and process configuration. *International Journal of Computer Integrated Manufacturing* **23**, 515 (2010).
115. Ahumada, O. & Villalobos, J. R. Application of planning models in the agri-food supply chain: A review. *European journal of Operational research* **196**, 1 (2009).
116. Lowe, T. J. & Preckel, P. V. Decision technologies for agribusiness problems: A brief review of selected literature and a call for research. *Manufacturing & Service Operations Management* **6**, 201 (2004).

117. Akkerman, R., Farahani, P. & Grunow, M. Quality, safety and sustainability in food distribution: a review of quantitative operations management approaches and challenges. *Or Spectrum* **32**, 863 (2010).
118. Catalá, L. P., Durand, G. A., Blanco, A. M. & Bandoni, J. A. Mathematical model for strategic planning optimization in the pome fruit industry. *Agricultural Systems* **115**, 63 (2013).
119. Zutshi, A., Creed, A. & Sohal, A. Child labour and supply chain: profitability or (mis) management. *European Business Review* **21**, 42 (2009).
120. Narasimhan, R. & Mahapatra, S. Decision models in global supply chain management. *Industrial Marketing Management* **33**, 21 (2004).
121. Li, G., Yan, H., Wang, S. & Xia, Y. Comparative analysis on value of information sharing in supply chains. *Supply Chain Management: An International Journal* **10**, 34 (2005).
122. Blackburn, J. & Scudder, G. Supply chain strategies for perishable products: the case of fresh produce. *Production and Operations Management* **18**, 129 (2009).
123. Aung, M. M. & Chang, Y. S. Traceability in a food supply chain: Safety and quality perspectives. *Food control* **39**, 172 (2014).
124. Mula, J., Peidro, D., Díaz-Madroño, M. & Vicens, E. Mathematical programming models for supply chain production and transport planning. *European Journal of Operational Research* **204**, 377 (2010).
125. Díaz-Madroño, M., Peidro, D. & Mula, J. A review of tactical optimization models for integrated production and transport routing planning decisions. *Computers & Industrial Engineering* **88**, 518 (2015).
126. Broekmeulen, R. A. Operations management of distribution centers for vegetables and fruits. *International Transactions in Operational Research* **5**, 501 (1998).
127. Eskigun, E., Uzsoy, R., Preckel, P. V., Beaujon, G., Krishnan, S. & Tew, J. D. Outbound supply chain network design with mode selection, lead times and capacitated vehicle distribution centers. *European Journal of Operational Research* **165**, 182 (2005).
128. Kawamura, M., Ronconi, D. & Yoshizaki, H. Optimizing transportation and storage of final products in the sugar and ethanol industry: a case study. *International Transactions in Operational Research* **13**, 425 (2006).
129. Lamsal, K., Jones, P. C. & Thomas, B. W. Harvest logistics in agricultural systems with multiple, independent producers and no on-farm storage. *Computers & Industrial Engineering* **91**, 129 (2016).

130. Pittia, P., Nicoli, M. C., Comi, G. & Massini, R. Shelf-life extension of fresh-like ready-to-use pear cubes. *Journal of the Science of Food and Agriculture* **79**, 955 (1999).
131. McHugh, T. & Senesi, E. Apple wraps: A novel method to improve the quality and extend the shelf life of fresh-cut apples. *Journal of Food Science* **65**, 480 (2000).
132. Róth, E., Berna, A., Beullens, K., Yarramraju, S., Lammertyn, J., Schenk, A. & Nicolai, B. Postharvest quality of integrated and organically produced apple fruit. *Postharvest Biology and Technology* **45**, 11 (2007).
133. Aung, M. M. & Chang, Y. S. Temperature management for the quality assurance of a perishable food supply chain. *Food Control* **40**, 198 (2014).
134. Bortolini, M., Ferrari, E., Galizia, F. & Mora, C. A Reference Framework Integrating Lean and Green Principles within Supply Chain Management. *World Academy of Science, Engineering and Technology, International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering* **10**, 884 (2016).
135. Birge, J. & Louveaux, F. *Introduction to Stochastic Programming. Series in Operations Research and Financial Engineering* 1997.
136. Ruszczyński, A. P. & Shapiro, A. *Stochastic programming* (Elsevier Amsterdam, 2003).
137. Wallace, S. W. & Ziemba, W. T. *Applications of stochastic programming* (Siam, 2005).
138. Mateo-Fornés, J., Plà-Aragonés, L. M., Solsona, F. & Pagès-Bernaus, A. A production planning model considering uncertain demand using two-stage stochastic programming in a fresh vegetable supply chain context. *SpringerPlus* **5**, 1 (2016).
139. Kall, P. W. *SW: Stochastic programming* 1994.
140. Soto-Silva, W. E., González-Araya, M. C., Oliva-Fernández, M. A. & Plà-Aragonés, L. M. Optimizing fresh food logistics for processing: Application for a large Chilean apple supply chain. *Computers and Electronics in Agriculture* **136**, 42 (2017).
141. Bravo, J. Chile y el mercado mundial de la fruta industrializada (2010).
142. Oliva, M. Optimization Model for Agribusiness' Apple Supply Chain. Programa de Magister en Gestión de Operaciones (2011).

143. González-Araya, M. C., Soto-Silva, W. E. & Espejo, L. G. A. in *Handbook of Operations Research in Agriculture and the Agri-Food Industry 79* (Springer, 2015).
144. Gil, G. & Gonzalo, F. Fruticultura: madurez de la fruta y manejo de post-cosecha: fruta de climas templados, subtropical y uva de vino. *Santiago, Chile. Ediciones Universidad Católica de Chile* (2001).
145. Hart, W. E., Watson, J.-P. & Woodruff, D. L. Pyomo: modeling and solving mathematical programs in Python. *Mathematical Programming Computation* **3**, 219 (2011).
146. Hart, W. E., Laird, C., Watson, J.-P. & Woodruff, D. L. *Pyomo—optimization modeling in python* (Springer Science & Business Media, 2012).
147. Beltran-Royo, C. Two-stage stochastic mixed-integer linear programming: The conditional scenario approach. *Omega* **70**, 31 (2017).
148. Cornou, C. & Kristensen, A. R. Use of information from monitoring and decision support systems in pig production: Collection, applications and expected benefits. *Livestock Science* **157**, 552 (2013).
149. Dubeau, F., Julien, P. O. & Pomar, C. Formulating diets for growing pigs: Economic and environmental considerations. *Annals of Operations Research* **190**, 239 (2011).
150. Gray, J., Banhazi, T. M. & Kist, A. A. Wireless data management system for environmental monitoring in livestock buildings. *Information Processing in Agriculture* **4**, 1 (2017).
151. Fountas, S., Carli, G., Sørensen, C., Tsiropoulos, Z., Cavalaris, C., Vatsanidou, A., Liakos, B., Canavari, M., Wiebensohn, J. & Tisserye, B. Farm management information systems: Current situation and future perspectives. *Computers and Electronics in Agriculture* **115**, 40 (2015).
152. *Handbook of Operations Research in Agriculture and the Agri-Food Industry* (ed Plà-Aragonés, L. M.) (Springer New York, New York, NY, 2015).
153. *Agrivi* <http://www.agrivi.com/>. Online; Accessed: December, 2018.
154. *Agroptima* <https://www.agroptima.com/>. Online; Accessed: December, 2018.
155. *Farmlogics* <http://farmlogics.com/>. Online; Accessed: December, 2018.

156. *Cloudfarms* <https://cloudfarms.com/>. Online; Accessed: December, 2018.
157. O'Grady, M. J. & O'Hare, G. M. Modelling the smart farm. *Information Processing in Agriculture* **4**, 179 (2017).
158. Fernández, Y., Bono, C., Babot, D. & Plà, L. Impact of prolificity on sow replacement policies | Impacto de la prolificidad en la política de reemplazamiento de cerdas reproductoras. *ITEA Informacion Tecnica Economica Agraria* **111**, 127 (2015).
159. Wu, L., Garg, S. K., Versteeg, S. & Buyya, R. SLA-Based Resource Provisioning for Hosted Software-as-a-Service Applications in Cloud Computing Environments. *IEEE Transactions on Services Computing* **7**, 465 (2014).
160. Kouki, Y. & Ledoux, T. *SLA-driven capacity planning for Cloud applications* in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings* (IEEE, 2012), 135.
161. *cplex* <https://www.ibm.com/products/ilog-cplex-optimization-studio>. Online; Accessed: December, 2018.
162. *gurobi* <http://www.gurobi.com/>. Online; Accessed: December, 2018.
163. Barahona, F. & Anbil, R. The volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming* **87**, 385 (2000).