Universitat Autònoma de Barcelona

# Cloud-Based Urgent Computing for Forest Fire Spread Prediction

Edigley Pereira Fraga

Universitat Autònoma de Barcelona

Escola d’Enginyeria
Departament d'Arquitectura de Computadors i Sistemes Operatius

# Cloud-Based Urgent Computing for Forest Fire Spread Prediction

Thesis submitted by **Edigley Pereira Fraga** in fulfillment of the requirements for the doctoral degree from Universitat Autònoma de Barcelona, advised by Dra. Ana Cortés, Dr. Porfidio Hernández, and Dr. Tomàs Margalef.

Barcelona (Spain), June 9th, 2023

# Cloud-Based Urgent Computing for Forest Fire Spread Prediction

Thesis submitted by **Edigley Pereira Fraga** in fulfillment of the requirements for the doctoral degree from Universitat Autònoma de Barcelona. This work has been developed in the Computer Science doctoral program (Research Line *High Performance Computing*) presented to the *Computer Architecture & Operating Systems Department* at the *Escola d'Enginyeria of Universitat Autònoma de Barcelona*. This thesis was supervised by Dra. Ana Cortés Fité, Dr. Porfidio Hernández Budé, and Dr. Tomàs Margalef Burrull.

# Acknowledgements

Apart from all the people involved in my academic career, I would like to thank the following ones:

- Dra. Ana Cortés, Dr. Porfidio Hernández, and Dr. Tomás Margalef for their expert supervision and encouragement throughout this work. Their suggestions about the best course of action to be followed during this process were invaluable, not to mention their patience in all the critical moments.
- Dr. Andrés Cencerrado and Dr. Carles Carrillo for their help at the beginning of this work. The beginning is always a fuzzy moment when someone is getting into a new area of knowledge.
- All the members of CAOS (*Computer Architecture & Operating Systems Department*), who helped with comments and suggestions in important milestones as well as in the brainstorming sessions.
- All the anonymous reviewers. Their constructive comments helped to improve the work in each paper submission.

Finally, thanks to all the members of my family for understanding all the moments I needed to be absent in the last years.

## Abstract

Fire is a natural element of many ecosystems, and even large wildfires are part of a defined disturbance regime. For that reason, the challenge from both a prevention and a suppression point of view is to anticipate and reduce the spread potential of large wildfires, and the succeeding risk for lives, properties, and land use systems. Despite the significant technological advances over the past decades, this kind of natural hazard is still difficult to be modeled and to be accurately simulated. Input data describing forest fire scenarios are subject to high levels of uncertainty that represent a serious challenge for the correctness of the prediction. To deal with this issue, the scientific community developed input data calibration methods, among them a two-stage methodology that allows the adjustment of the input parameters in a calibration phase.

An implementation based on *Genetic Algorithm* has been successfully adopted as a calibration technique. State-of-the-art solutions require the execution of many simulations to generate the best-calibrated set of input parameters, which delays the response time for the actual prediction. Unfortunately, for the task of fire suppression, an accurate prediction that comes up late is useless. While used in a fire extinction activity, a wildfire spread prediction is a *hard-deadline-driven* task. In this thesis, we have investigated the feasibility of accurately predicting wildfire spread under a strict deadline constraint and in a cost-effective way. We relied on the assumption that through cloud computing enabling technology it is possible to achieve more accurate prediction results in less time even for larger wildfire occurrences when compared to traditional *HPC*-based solutions.

We have conducted a set of activities to devise and evaluate an adequate solution to our research problem, defining a *performance-efficient* and *cost-effective cloud-based* system built upon a proven methodology for forest fire spread prediction. We exploit an elastic and scalable cloud-based solution platform implemented through coarse-grain parallel processing using a work queue. We devised a new *evaluation technique* capable of reducing the overall calibration time by 60% when compared to the current state-of-art data-driven approaches. We also proposed and evaluated a new fitness function together with a *strict deadline* policy that decreases overall processing time. Conforming to the hard-deadline-driven nature of fire extinction activities, the proposed strategies improve the genetic algorithm convergence and decrease the response time for the calibration stage, setting up an important upper bound limit to the critical compute-intensive adjustment phase.

We have evaluated the proposed solution against three historical fires that represent different and increasing levels of complexity in relation to the simulation challenges. The enhancements to the two-stage prediction framework presented in this work have been validated against fire scenarios in the Mediterranean region at Greece (*Arkadia*) and Spain (*Catalonia*), and in the United States (*California)*.

## Keywords

Forest Fire Spread Prediction, Cloud Computing, Data-Driven Simulation, Genetic Algorithm, Two-Stage Prediction, High-Performance Computing.

# Contents

# List of Tables

# List of Figures

# Chapter 1: Introduction

Fire is a natural element of many ecosystems and even large wildfires are part of a defined disturbance regime [1]. For that reason, the challenge from both a prevention and a suppression point of view is to anticipate and reduce the spread potential of large wildfires, and the succeeding risk for lives, properties, and land use systems [2]. Wildfires have a relatively unpredictable nature as their spread can vary based on the flammable material and can differ by their extent and wind speeds. Forest fire prevention strategies for detection and suppression have improved significantly through the years, both due to technological innovations and the adoption of various skills and methods. Nowadays, wildfire researchers use technologies that integrate data on weather prediction, fuel, topography, and other factors to predict how fires spread.

Nevertheless, wildfires still occur widely and represent a permanent threat whose consequences may be catastrophic both in terms of human fatalities, ecosystem degradation and economic losses. For example, years marked by intense drought and dry conditions contribute to the high levels of wildfire activity which could be turned into natural disasters. In the last years, unusually large wildfires severely damaged forests in Indonesia, Canada, United States, Spain, Chile, Portugal, and Australia, just to name a few. Actually, some studies suggest that over the past few decades, the number of wildfires has indeed increased [3] [4] [5] [6].

Considering this situation, forest fire prediction, prevention and management measures have become increasingly important over the past quarter century. Systems for wildfire prediction represent an essential asset to back up forest fire monitoring and extinction phase, to predict forest fire risks and to help in fire control planning and resource allocation.

## 1.1. On the urgent computing nature of a fire spread prediction system

When dealing with the extinction phase, an accurate prediction of the fire propagation is a critical issue to minimize its effects. As a matter of fact, to be used in a fire extinction activity, a wildfire spread prediction is a *hard-deadline-driven* task. For instance, a complex wildfire simulation that could accurately predict the perimeter of a wildfire a couple of days ahead can drive firefighters to put firebreaks where they would be most effective to stop the fire propagation [7]. In this particular case, an accurate prediction that comes up late compared to the actual event is useless to the task of fire suppression. These characteristics represent an *urgent computing* system, from which the simulation results are needed by relevant authorities in making timely informed decisions to mitigate financial losses, manage affected areas and reduce casualties [8]. The following three urgent computing requirements can be found in the forest fire propagation system at hand:

**a)** The computation operates under a strict deadline after which the computation results may give little practical value (” *late results are useless*”).

**b)** The beginning of the event that necessitates the computation is unpredictable.

**c)** The computation requires significant resource usage.

To fulfill those requirements, any viable solution must be *deadline-driven*, *on-demand provisioned* and *scalable*. To deal with those kinds of applications, High Performance Computing (*HPC*) community used to rely on dedicated high-end clusters, on supercomputers or on distributed computing platforms [9]. As deadline and priority-based resource management are key-components to urgent computing procedures, much work have been conducted to address this issue [10] [11] [12] [13] [14] [15]. High-performance computational resources (software and hardware) involved into simulation process should be provisioned and managed in a way that the computation process will be finished within a defined time-limit. The objective of these technologies is to support urgent computation dynamically, preserving overall machine utilization. One concern is not being much intrusive, allowing the productivity of the other users working daily on the platforms being used.

## 1.2. Forest fire spread simulation and data uncertainty problem

The use of simulators for forest fire propagation models requires sufficient time for the processing, a precise fuel model data and an accurate knowledge of small and large-scale interaction of weather (largely wind) and topography [5]. To start a simulation, it is necessary a plethora of input parameters, which include spatial data describing the elevation, slope, aspect, and fuel. Topography is a static element, as it changes only in space, not in time. Fuel is a semi-static element: it varies both in time and space, but its change in time is not so frequent.

In reality, the input data describing the actual scenario where the fire is taking place is usually subject to high levels of uncertainty that represent a serious drawback for the correctness of the prediction [16] [17] [18]. In a scenario with uncertainties regarding input data, to accurately predict a fire spread under a strict deadline constraint and in a cost-effective way represents a challenge for any designed solution.

Considering new advancements in cloud computing offerings, it was imperative to investigate the possibility of achieving more accurate prediction results in less time even for larger fire occurrences when compared to traditional *HPC*-based solutions. As explained, besides guaranteeing short execution time (*performance*), one must also take into consideration the total simulation cost (*total expense*) involved in the designed solution [19]. Consequently, it was necessary to define a **performance-efficient and cost-effective solution built upon a proven methodology for forest fire prediction**.

## 1.3. A cloud-based solution

So far, on-premise *HPC* solutions was usually built under the restriction of scarcity of available computing resources [20] [21] [22] or on multi-million-dollar supercomputing infrastructures [23]. In contrast, one of cloud computing characteristics is the abundance, which results in the illusion of infinite computing resources available on demand [24]. Obviously, some challenges arise since there is still some gaps between *HPC* and cloud paradigms regarding computing power and communication efficiency [25].

As an enabling technology, cloud computing allows new strategies to deal with the urgent computing challenge and requirements. Considering cloud computing paradigm characteristics of on-demand provisioning, immediate scalability, and abundant offer of resources (requirement **b**), one might consider it a natural choice as a platform for run such applications. Regarding strict deadline (requirement **a**) and resource usage (requirement **c**), although in the early days cloud offerings suffered to run *HPC* applications, nowadays they are an alternative to on-premise clusters [25] [26] [27] [28] [29].

Even though not mentioned in the general definition of urgent computing [8], one important aspect for any system is the incurred cost. Different from traditional *HPC* applications, which involve high total cost of ownership on subjacent platform, cloud-based solutions allow access to supercomputing-like features at the cost of a few dollars per hour. This is something particularly important for fire propagation system due to **1)** the seasonality of the wildfire occurrences, and **2)** because forest fire prevention services or fire brigades usually can't afford the total cost of ownership (*TCO*) to keep an infrastructure idle until eventually needed by an urgent computation.

Therefore, from an initial hypothesis that a solution based on cloud offerings would improve the accuracy of fire propagation prediction results, we have conducted a set of activities to devise and evaluate an adequate solution to this technologic and scientific research problems, defining a performance-efficient and cost-effective cloud-based system built upon a proven methodology for forest fire spread prediction. We have exploited an elastic and scalable cloud-based solution platform implemented through coarse-grain parallel processing using a work queue. Figure 1 illustrates the architecture of the proposed solution.

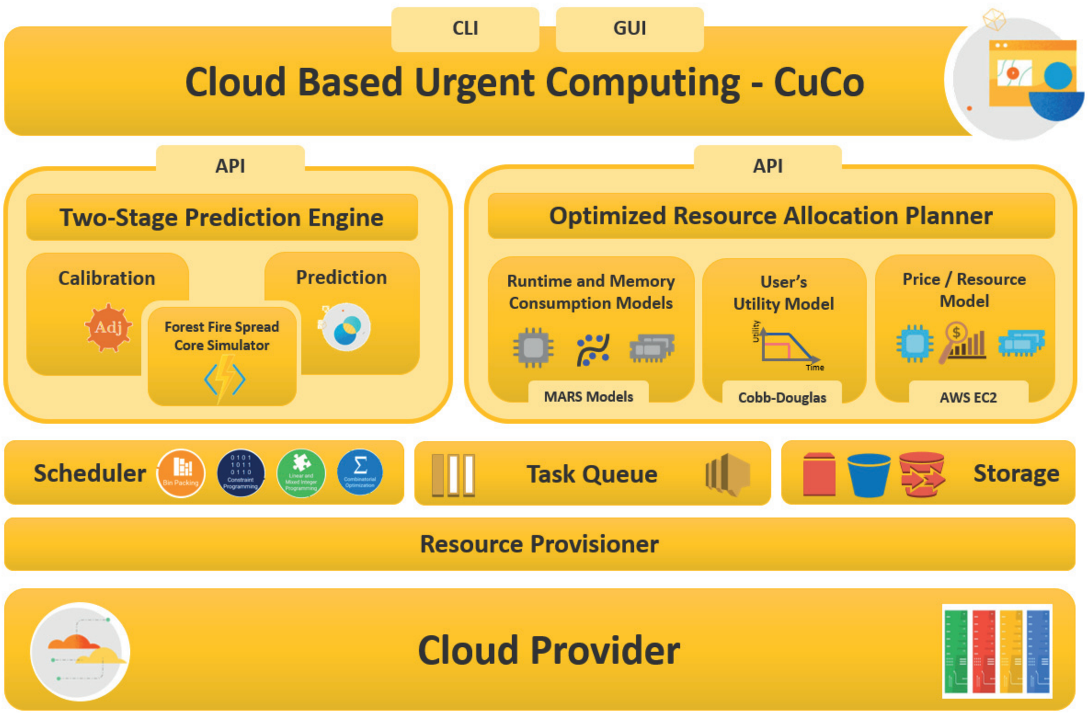

CLI
GUI
Cloud Based Urgent Computing - CuCo
API
Two-Stage Prediction Engine
Calibration
Prediction
Forest Fire Spread Core Simulator
API
Optimized Resource Allocation Planner
Runtime and Memory Consumption Models
MARS Models
User's Utility Model
Cobb-Douglas
Price / Resource Model
AWS EC2
Scheduler
Task Queue
Storage
Resource Provisioner
Cloud Provider

*Figure 1: Overall architecture of a Cloud-Based Urgent Computing solution.*

The *Two-Stage Prediction Engine* is at the core of the solution. It is a proven model to deal with the data uncertainty problem in the forest fore spread prediction simulation. To deal with the cost-performance tradeoff, an *Optimized Resource Allocation Planner* is included, incorporating domain specific statistical models for the allocation of cost and performance-efficient cloud resources. In the base of the layered architecture, are off-the-shelf cloud solutions for scheduling, queuing, and storage.

We improved the *Two-Stage Prediction* by means of a new evaluation technique (*Early Adaptive-Evaluation*) based on a periodic monitoring of the spread prediction error, avoiding the waste of computing time running undoubtedly unfit individuals. We have also proposed a new fitness function (*Goodness-of-Fit*) in tandem with a *strict deadline* policy that decreases overall processing time. In consonance with the hard-deadline-driven nature of fire extinction activities, the proposed strategies improve the genetic algorithm convergence and decrease the response time for the calibration stage, setting up an important upper bound limit to the critical compute-intensive adjustment phase.

As we could notice, apart from the technology advance of using a new computing paradigm, we contributed with new strategies in the core of the *Two-Stage Prediction Framework*, as it can be seen with the *Early Adaptive-Evaluation* strategy, the *Goodness-of-Fit* evaluation function, and the *Strict-Deadline* policy. At the same time, the proposed strategies simplified the implementation and improved the response time of the time-critical calibration stage, being a solid step forward to bring an efficient and easy to deploy cloud-version of the *Two-Stage Prediction Framework*.

## 1.4. State of the art

Related works in forest fire prediction systems can be organized according to the approach used in the investigation or implementations, including natural phenomena modeling, machine learning, computer simulation, statistical analysis, and high-performance applications.

To improve the accuracy of wildfire spread predictions, Srivas [30] extended FARSITE to incorporate data assimilation techniques based on noisy and limited spatial resolution observations of the fire perimeter. The adjustment is calculated from the Kalman filter gain in an Ensemble Kalman filter, based on a Monte-Carlo implementation of the Bayesian update problem. Uncertainty on both the measured and the simulated fire perimeter is used to formulate optimal updates for the prediction of the spread of the wildfire.

In order to cope with the input data uncertainty related with fire spread simulation, Abdalhaq [31] proposed a ***two-stage methodology*** to calibrate the input parameters in an adjustment phase so that the calibrated parameters are used in the prediction stage to improve the quality of the predictions.

Cencerrado [21] applied *Genetic Algorithms* as the calibration technique in the adjustment phase, which requires the execution of many simulations to generate the best calibrated set of input parameters. Similar work was also carried over by Mendez-Garabetti et al [32]. Cencerrado also devised one strategy based on decision trees to identify long running execution of a fire spread simulation. That strategy was the base for a classification method that allows to estimate in advance the execution time of a simulation given a certain set of input parameters. At the end, Cencerrado et al envisioned a ***two-stage framework*** to be use for urgent computing problems**.**

More recently, Artés [20], in his turn, proposed and evaluated a set of resource allocation policies to assign more computing resources to estimated long running executions and fewer resources to the fast ones, allowing to reduce the adjustment stage time to a more acceptable deadline. That was possible due to the use of a parallel version of FARSITE model that could reduce long running execution times by at least 35% [33]. To work in a time-constrained fashion, a hybrid MPI-OpenMP application based on the Master-Worker paradigm was developed to take advantage of the execution in a parallel *HPC* cluster environment, dubbed Time Aware Core allocation (*TAC*) and extended the ***two-stage framework*** to its current state-of-the-art version. Carrillo [34] compared different fitness functions and showed their impact on the prediction quality in dynamic data-driven calibrations for forest fire spread predictions.

Although being part of a comprehensive work, the implementation, evaluation, and strategies applied to the ***two-stage framework*** were designed to run on a dedicated cluster provided with a defined number of nodes, unaware of elasticity and scalability enabled by cloud computing paradigm. Therefore, the challenge faced at the time was to wisely exploit all computing resources available. A switch to a cloud environment might lead to the exploration of new strategies to improve the final prediction accuracy and the time restrictions. Nevertheless, despite its *cloud-unawareness*, the **two-stage-framework** has been proved to be a good methodology to deal with the input data uncertainties and it has been leveraged in this thesis. An opportunity of improvement in the framework implementation was the understanding of the memory consumption and runtime of *FARSITE* simulator, where schedulers can assign a suited memory or compute-intensive instances to overcome this potential points of congestion.

Regarding forest fires applications, a plenty of solution have been proposed and evaluated in the literature. Saurabh and others [35] have recently proposed a cloud-based framework to deploy and process fire models within a deadline based on a scheduling mechanism which integrates user's requirements to minimize resource usage. They also evaluated the framework with a case study using *Tasmania Bushfire Model*.

Wildfire Analyst (WFA) is a software application that allows real-time analysis of wildfire, simulating the spread of wildfires using Rothermel's model among others [36]. It is integrated with Geographic Information System *(GIS)* tools, allowing to change parameters to better reflect actual conditions. Although being originally a desktop application, it has been updated and it is also offered as a web and mobile, being designed to be used at the operations center, or directly on scene [37]. It provides a comprehensive set of outputs and tools, also including a data assimilation technique which tunes the simulations results to the actual observed fire behavior.

Kalabokidis [38] implemented cloud application composed of a wildfire risk and a wildfire spread simulation service. End users access the application in a *software as a service* delivery model, being charged for their consumed processing time during the actual wildfire simulation period. The application presents the flexibility to scale up or down the number of computing nodes needed for the requested processing depending on the number of simultaneous users. Although being a step toward the spread of adoption of simulation techniques to local fire agencies, the solution does not address the issues related with input data uncertainties neither was developed to be applied in urgent computing firefighting scenarios.

Altintas et al [23] conducted the comprehensive WIFIRE Firemap project, which is a dynamic data-driven system to predict wildfire progress through data analysis and map visualizations. They use *FARSITE* as one of the fire spread simulation models, coupled with a wind simulator, both fed with geo-located images of fire acquired from human observers and monitoring cameras. Compute-intensive tasks are executed in parallel on distributed computing environments, including *San Diego Supercomputing Center* infrastructure and public cloud offerings.

Miller [39] presented an integrated software system for forest fire spread prediction, which uses a user-defined algebraic spread rate to model fire propagation. The software model is run based on a modular workflow-based software environment. Garg [35] proposed a scalable cloud-based bushfire prediction framework, which allows forecasting of the probability of fire occurrences. The solution allows the selection of different bushfire models for specific regions and scheduling users' requests within their specified deadlines.

Arca [40] presented a web-based wildfire simulator for operational applications that can assist the incident command teams in charge of tactical wildfire suppression. The simulator consists of a graphical user interface, a model devoted to the downscaling of wind fields, and a module that provides the wildfire propagation. The whole solution is a client-server application, with the heavy computational work executed in parallel on a dedicated server.

Oliveira [41] developed a fire-spread prediction system tailored for the Brazilian Cerrado. Their system allows automatically upload of hotspots and satellite data to calculate maps of fuel load and moisture, and probability of burning for simulating fire spread. Results are available on an interactive web-platform, used as a tool for fire prevention and suppression. It is executed on a parallel platform that uses execution threads boosted by task-stealing algorithms, running on on-premises on a dedicated high-end server.

Considering the evaluated applications and solutions, although being a step toward the spread of adoption of simulation techniques to local fire agencies, they do not address the issues related to input data uncertainties in an agnostic approach, neither was developed to be applied in urgent computing firefighting scenarios. Both characteristics, diminishing data uncertainty and easy-to-deploy cloud-based urgent computing, are the objective of our proposed solution.

## 1.5. Objectives, assumptions, and requirements

In a scenario with uncertainties regarding input data, any designed solution faces a challenge when accurately predicting a fire spread under a strict deadline constraint and in a cost-effective way. Considering new advancements in cloud computing offerings, it was necessary to investigate the possibility of achieving more accurate prediction results in less time even for larger fire occurrences when compared to traditional *HPC* based solutions. Besides guaranteeing short execution time (*performance*), one must also take into consideration the total simulation cost (*total expense*) involved in the designed solution [25]. Consequently, it was necessary to define a **performance efficient and cost-effective cloud-based system built upon a proven methodology for forest fire prediction under data uncertainty**.

So far, on premise *HPC* solutions was usually built under the restriction of scarcity of available computing resources or on multi-million-dollar supercomputing infrastructures. In contrast, one of cloud computing characteristics is the abundance, which results in the illusion of infinite computing resources available on demand. Therefore, we stated from a plausible initial hypothesis that a solution based on cloud offerings may improve the accuracy of fire propagation prediction results.

The primary objective of this thesis was to ***propose*, *implement*, and *evaluate a cloud-based solution* for the *forest fire spread prediction under data uncertainties* problem**. To address this main objective, the following *secondary objectives* have been defined:

- Understand how a *forest fire spread* simulator works and then propose a model to characterize its runtime and memory consumptions.
- Propose and integrate an *optimized resource allocation planner* to help in the cloud resource provisioning, taking into consideration both cost and deadline constraints.
- Propose a high-level architecture for a cloud-based urgent computing solution based on the *two-stage prediction framework*.
- Implement a *proof-of-concept* to validate the proposed architecture.
- Validate the proposed solution against real case forest fire scenarios.

During the work for this thesis, to achieve the objective of accurately predicting a fire spread under a strict deadline constraint in a cost-effective way, the following activities have been conducted.

- Investigation of the most recent works on *HPC* applications which have been ported to a cloud environment and evaluation of the architectures currently available.
- Evaluation of the performance and scalability of the two-stage methodology for fire spread prediction technique. The evaluation was first performed through discrete simulation, considering both synthetic and real case forest fire scenarios.
- Proposal of efficient evaluation techniques for individual fire spread predictions using the simulator *FARSITE*. We devised an **efficient evaluation technique** based on a **periodic monitoring** of the **fire spread prediction error** estimation.
- Proposal of a new metric to compare predicted and real fire spread with the intention of obtaining more accurate ***calibration*** and ***prediction*** results.
- Proposal of a strict-deadline policy per generation to have an upper bound for worst-case scenarios.
- Implementation of an efficient *proof-of-concept* cloud-aware version of the forest fire prevention framework, taking advantage of scalability and elasticity from the cloud to obtain a speed up when compared with the current MPI-OpenMP version.
- Evaluate the proposed strategies against other real fire scenarios.

The remainder of this document is organized as follows. A contextualization on forest fires, modeling, simulation, and calibration is presented in Chapter 2. FARSITE simulation model, used for the prediction of forest fire spread in our solution, is also presented. Chapter 3 describes the architecture of the proposed cloud-base solution for the forest fire spread prediction under data uncertainty problem. Chapter 4 presents the experimental evaluation of the proposed solution, using real data from forest fires that occurred in Europe and in the United States. Finally, Chapter 5 summarizes the research findings and provides some concluding remarks.

# Chapter 2: Forest Fires, Modeling, Simulation, and Calibration

Forest fires have been widely studied by the scientific community and public authorities due to the huge environmental, social, and economic impact they produce every year around the world. They belong to the category of natural hazards, which represent a permanent thread whose consequences may be catastrophic. To manage forest fire risk, prevention and suppression activities are taken into consideration. Multiple factors are involved in forest fires regimes, including environment conditions and the effects of climate changes, what turns prevention and suppression activities a challenge even for the most capable fire brigades. In this chapter, we consider the dynamic of forest fires and the environment conditions, the types of forest fires, how they are modelled and simulated, how data are acquired and how model are calibrated.

## 2.1. Forest fires and the environment conditions

The fire behavior is a product of the environment in which the fire is burning [42]. The fire environment is determined by the surrounding conditions, factors of influences, and modifying forces. The forces that interact between each other and influence the fire environment are the *topography*, meteorological conditions (weather), vegetation (fuel) and the fire itself. Combining the fire ignition with these three components, and the interactions among them, we have the fire environment triangle. Figure 2 illustrates this concept. The fire in the center of the triangle represents the interaction between the fire and the surrounding environment. All these three components affect the fire spread and behavior [42] [43].

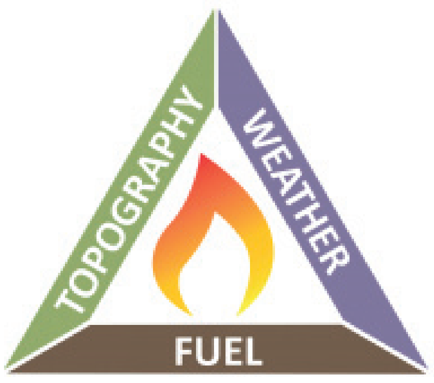
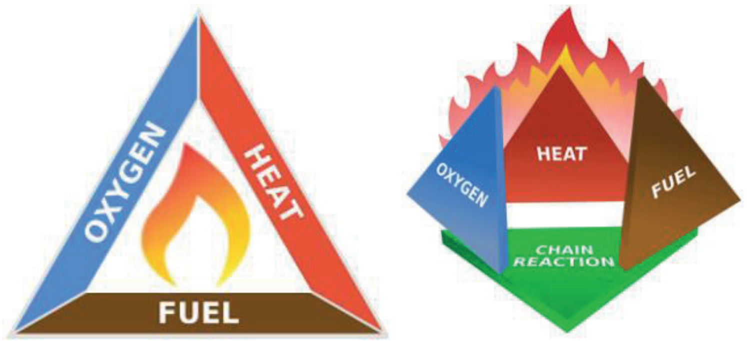


*Figure 2: The fire environment (or behavior) triangle.*

The fire environment triangle resembles the fundamental fire triangle, that combines the elements necessary for a combustion: *oxygen*, *fuel*, and *heat*. The combustion generates *heat* and byproducts like water vapor, carbon dioxide, and ashes, being a fast chemical-physical process. It is process that can be seen like the inverse of photosynthesis. When *oxygen* and *heat* are present, the vegetation (*fuel*) supplies the chemical energy for the propagation of the fire [42] [44]. The *heat* is the source of energy necessary to bring the vegetation temperature of ignition. There are four types of heat flow: radiation, conduction, convection, and mass transport. By one or more heat flows, this energy needs to be transferred to pre-heat the vegetation, so the process can continue, and the fire propagates. The *oxygen* is necessary to release chemical energy from the plant into the atmosphere. For the combustion reaction, this component acts as a stable source of oxygen (the oxidant agent).

Figure 3(a) illustrates the fundamental fire triangle: removing any one of these components will collapse the triangle and extinguish the fire; given abundance of them, a fire will be sustained in time. Further research into fire chemical reaction on a molecular level have concluded that a fourth element should be considered as necessary component of fire, which is a *chemical chain reaction*. It is the chemical process in which the fuel reacts rapidly with a source of oxygen to create fire as result [45]. This reaction demonstrates how the three sources (*oxygen*, *heat,* and *fuel*) of the fire triangle interact to create fire. The fire tetrahedron then includes this fourth component and is illustrated in Figure 3 (b).



(a) (b)

*Figure 3: The fundamental fire triangle (a) and the fire tetrahedron (b).*

The forest fire behavior is dynamic, and the propagation characteristics change over time, even when the environmental conditions remain unchanged. If we consider the timespan of a fire season, the topography changes in space, but remains unchanged in time; meteorological conditions are the most varying component, with fast variations in space and time. the vegetation (fuel) changes both in space and in time, depending on the type of vegetation and its interaction with weather and topography.

### 2.1.1. Topography

Topography is the configuration of a landscape, surface, or terrain, including geographical features and relief. Besides its effect on fire behavior, topography strongly influences weather and vegetation. In zones with complex orography, the effect of the topography is more prominent while it is very restricted in flat areas. The three general characteristics of the topography that interests in fire behavior are *Elevation*, *Aspect*, and *Slope*. The fourth element are *Barriers*, which can be natural or resulting of urban areas or human actions.

***Elevation*** in a geographic location is its height above sea level. It influences the climate and consequently the presence of burnable fuel. *Digital Elevation Maps* (*DEM*) can be easily obtained for local or global areas. For example, the *United States Geological Survey* (*USGS*) provides high resolution images for to entire Earth surface. These data have been acquired from ASTER (*Advanced Spaceborne Thermal Emission and Reflection Radiometer*) instrument from NASA Satellite. ASTER is the high spatial resolution instrument on the *Terra* satellite. *Aspect* and Slope themes can be obtained from the *elevation* map using *GIS* tools.

***Aspect*** is the compass direction that the slope faces. The *aspect* feature can be summarized as *north-facing*, *east-facing*, *south-facing*, and *west-facing* slopes. It influences on the amount of solar radiation received by the surface of the terrain. When the terrain is flat, there is *no slope* and therefore *no aspect.*

***Slope*** affects air masses movement on local scale. As the fuel bed is tilted, it allows fast heating of fuel particles, resulting in elevated rate of spread. That is why among the topographical element the slope is the one that most affects the fire spread. Figure 4 illustrates the effect of increasing the slope to the fire propagation.

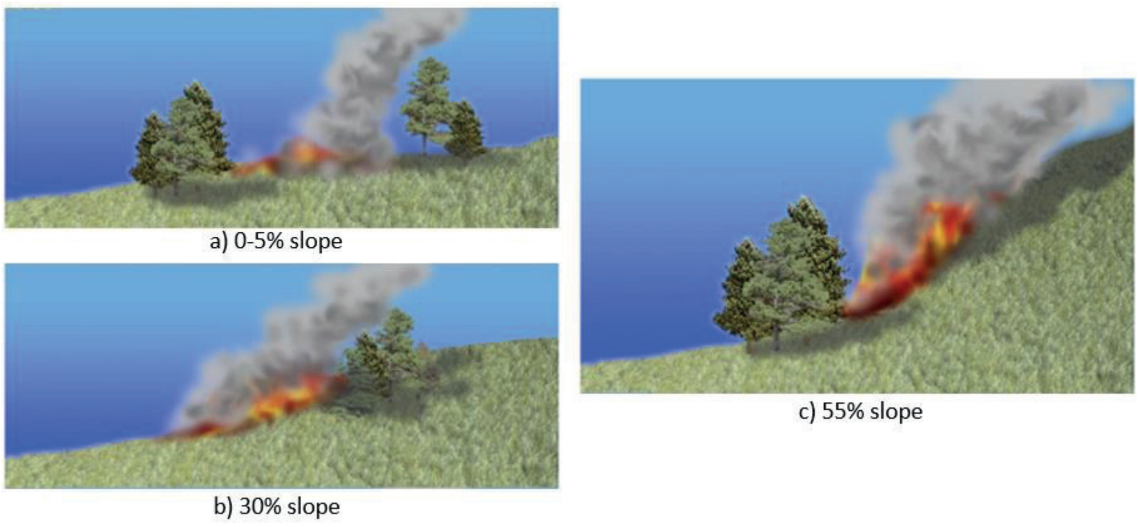


*Figure 4: The effect of increasing the slope to the fire spread.*

Wind direction and speed are influenced by these landscape features, and mountains are an obstacle to the flow of air masses on the horizontal plane. If the air current is confined by the relief elements, it increases its speed. As a result, the fire *spread rate* and *fire intensity* increase.

Finally, *barriers*, being natural or artificial, can diminishes or even stop entirely the fire propagation. As there is no fuel to burn, a barrier doesn't support wildland fires, which stops its propagation in that area of the terrain. Some examples of barriers are urban or semi-urban areas; open bodies of water; some kinds of agricultural zones; areas with snow or ice; bare ground; and areas that have been recently burned (*black areas*).

### 2.1.2. Meteorological conditions

The concept of a fire season is associated with the increase of fire activity. It increases in spring as the weather transitions from windy and dry to hot and dry. Then, the fire season is the period that covers late spring and the beginning of autumn. Fuel dryness is at the maximum level, and the temperatures are elevated. Given that the conditions of fuel are favorable, the weather is the factor that most influences the fire behavior. The fuel flammability is influenced by the dryness, being susceptible to initial fire ignitions. Moreover, the wind plays a major role during the propagation of the flame and fire front.

Whereas vegetation can be managed by human through prescribe fires and other forest fire management activities, meteorological conditions are variable and can change unexpectedly. Theses changes can be both in space and in time and are highly related with the *data uncertainty problem* in the simulation of forest fires. Nevertheless, meteorological conditions can be forecasted, and the knowledge on how meteorological condition are evolving is essential for the operational activities of fire management and fire suppression.

Regarding forest fire spread, the meteorological variables affect both the fire ignition and the fire behavior:

a) *Affecting fire ignition*: they influence the possibility of a fire ignition, as they affect the fuel moisture (dryness); they have a minor effect on the fire spread, but they increase the possibility of a fire ignition, starting the entire process. Temperature, humidity, and cloud cover belong to this group of variables.

b) *Affection fire behavior*: the rate of spread is affected by how much oxygen flows, being available for the combustion and consequently for the process of transferring the heat along the fuel; wind characteristics are in this group.

### 2.1.3. Vegetation and fuel characteristics

The knowledge about the vegetation is a key element for understanding the fire behavior. As the vegetation is the *fuel* on which the fire spread, it determines the fire characteristics and influences on its behavior. The fuel characteristics determine how easily the fire ignites, what is the fire line intensity level, and the velocity of propagation (*spread rate*), and not to mention the maximum height of flames. On a landscape, the vegetation tends to be discontinuous and heterogeneous, being difficult to characterize it by means of a single metric. On the plus side, among all the environmental factors, the vegetation is the element that the human can control or easily direct through forest fire management activities.

The vegetation fuel has intrinsic properties, defined by its *density*, *chemical characteristics*, *heat content*, and *thermal conductivity*. They also have extrinsic properties, represented by *fuel load*, fuel *dimension* and *shape*, fuel *compactness*, and fuel *arrangement*. All these properties are important and considered in the fuel models used in simulation tools.

#### *Fuel load*

The fuel load is the total of organic matter that can be potentially used in a combustion. It can be used to estimate the heat that can be produced during a forest fire. Fire intensity and its hazardous potential increase with the increase in the fuel load. Objectively, the fuel load is defined as the oven-dry weight of all the available fuels in a unit of area.

Fuel load is volatile, and it is related with the types of vegetation. Herbaceous species have lower load, while logging slash areas have greater fuel load. The layer formed by the organic soil, roots, leaves, humus, and organic matter in state of decomposition is called *duff*, and they comprise the *ground fuel load*. Ground fuel load can sustain smoldering fires, sustained by the heat evolved when oxygen directly attacks the surface of a compact and condensed fuel.

The availability of oxygen for the fuel and the heat transfer process is influenced by the size, shape, and compactness of fuel particles. Fine fuels take less energy to get dry and then be ignited when compared to bulkier fuels. Henceforth, the fine fuels make easier the fire front propagation. Apart from that, spot fires are more frequent in terrains with huge amounts of thin fuels.

### Fuel layers

Besides, being divided in live and dead fuels, forest fuel is found in different layers. They are arranged in different complexes, composed of distinct particle sizes of dead and live matter from the vegetation. These layers can be classified as *ground fuels; litter fuels; low vegetation*; *woody fuels*; *shrubs* and *small trees*; *and tree canopy*. To propagate, the forest fire feeds from these layers. To make the classification simpler, we can be divided all these fuels into only 3 large groups: *ground fuels*, *surface fuels*, and *crown fuels*.

*Ground fuels* form a compact pile of layers that include humus, roots, and matter in state of decomposition lying on the ground (called *duff*). Leaves, needles, and other materials in the earlier stages of decomposition stay in the top of the duff, while in the bottom lies the mineral soil.

*Surface fuels* are comprised of litter, small trees, shrubs, herbaceous vegetation, downed woody, and slash (logging) debris. Most forest fires have its source on this layer, being it also responsible for a major part of its propagation., with different behavior depending on the characteristics of the surface fuel bed. Rates of spread and fire intensity differ to a greater extent depends on the type of surface fuel, depending on if it is spreading on a grassland, in areas with shrubs and trees, or if there are more litter on the ground.

*Crown fuels* include tree crowns and high shrubs. Their biomass has an elevated level of moisture content, and the crown get burned only if the fire releases heat enough to maintain itself for long time. Another possibility is when flames burn the tree crowns directly. Apart from having the radiation as the principal mechanism of heat transfer, the forest fire needs to climb through the shrubs and small trees to reach the crown fuel. These layers of dead branches or small trees connecting the surface fire to the crown fuels is called the *ladder fuel*, and are an important factor on fire transition, being considered in the simulation models.

### *Fuel moisture*

The combustion process depends on the presence of water or humidity in the live and dead plants, namely the fuel moisture. The level of dryness is ultimately responsible for the total of fuel that are effectively available to be consumed by a fire. Before igniting a fuel, it is needed the evaporation of the water from the plants or downed woods, litter, tinder, and sticks. More moisture content means that more heat is needed to remove the water from the vegetation layer. Weather conditions in the long and short term are responsible for the compound effect in the fuel moisture. Like the vegetation itself, the fuel moisture varies in space and in time, what influences flammability of the available fuel. *Fuel moisture content* is determined or evaluated in terms of percentage, considering the dry weight of the fuel as the reference. Moisture properties are present both in live and dead vegetation matter.

The environment humidity conditions the levels of water present in the fuel, and these conditions change throughout the day. For the dead fuel, the moisture gain occurs due to the balance between vapor of water in the air and water in liquid state (mainly due to precipitation) in the fuel. In its turn, the drying of the fuel depends on the evaporation. On the other hand, live fuel moisture is affected by weather, type of fuel, and topography. In particular, the location, size, and fuel composition are the components that most effect moisture content.

Regarding the location, dead fuel on the ground has an elevated moisture content on average. In their case, there is not much variability in the water level in relation to dead fuel on the surface. Considering the weather, the moisture on dead fuel is affected by the solar radiation, temperature, wind, precipitations, and atmospheric humidity. Solar radiation is the meteorological condition that mainly influences the dead fuel moisture, by directly affecting its temperature. Its influences depends on the latitude, slope, aspect, hour, day, month, and season. Wind factor has both wetting and drying effects, depending on the atmospheric humidity.

Moisture and flammability of dead fuels depends strongly on the weather conditions around them. As the sun rises and sets, fine dead fuels (grass and litter) gain and lose moisture content from hour to hour. As the temperature rises and falls, moisture moves between the fuel and the air. The burning conditions are in its maximum level typically late in the afternoon. In the case of large dead woody fuels, they dry more slowly, and they usually burn most readily during the peak of the fire season. Figure 5 illustrates some examples of dead fuels.

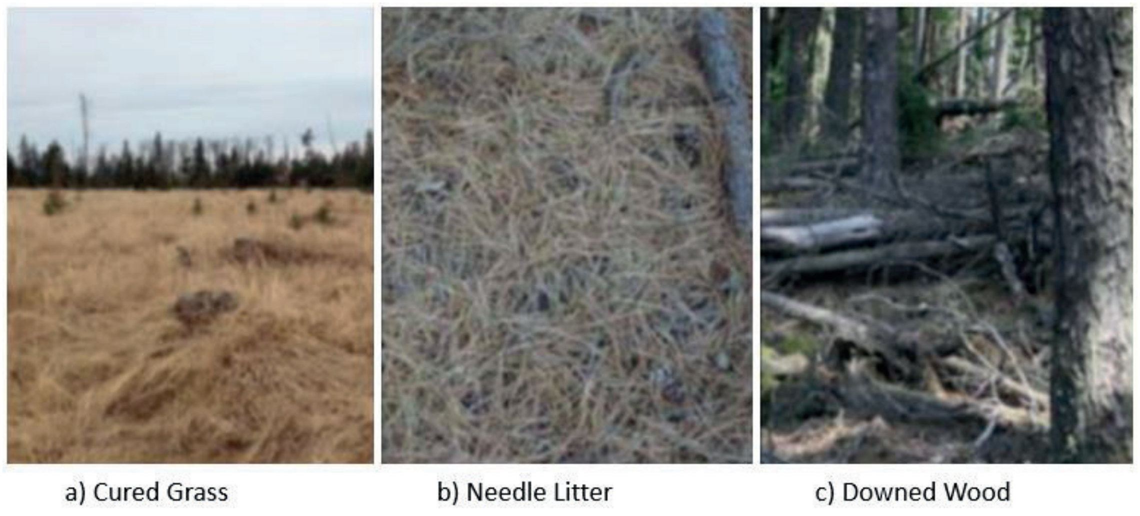


*Figure 5: Dead fuels examples.*

Although live fuel provides an extra source of energy along the combustion, it also consumes more energy for water evaporation and fire ignition, retarding the intensity and the propagation of the fire. The moisture content on live fuel varies extraordinarily during the year, depending on the conditions of the climate, the weather and the particular characteristics of the plants.

For example, in herbs, the minimum level of moisture occurs when the plant dies. In shrubs, the leaf moisture is at its minimum level right before the fall of the leaves. In shrub and other evergreen species, the minimum leaf moisture is elevated when compared to the value in other plants, because there are leaves in their branches the entire year. The volatility of the leaf moisture is marked by irregular values. Herbaceous species presents an irregular tendency in moisture content variation, as they are remarkably sensible to changes in the meteorological conditions. Figure 6 illustrates some examples of live fuels.

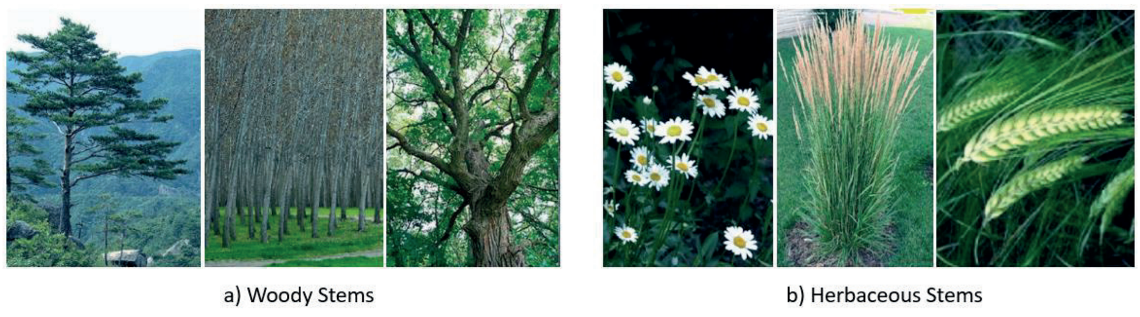


*Figure 6: Live fuels examples.*

### *Fuel classification*

Fuel models are the basis for a fuel classification system. They are mathematical description of the fuels on the surface, representing all the variables needed to define the fire behavior (the *spread rate* and the *fire line intensity*). A fuel model can represent a wide range of fuel types, as they are not defined solely on parameters visible on the plants. They are physical parameters of the fuel bed. Moreover, the fire behavior characteristics are computed for the entire range of topographical and meteorological conditions. On the minus side, fuel models are hard to validate, and it is not possible to rely on a mathematical equation to cover all the idiosyncrasies of the fuel types on the surface area. Figure 7 illustrates surface fuels classified into 6 groups to aid identification [46].

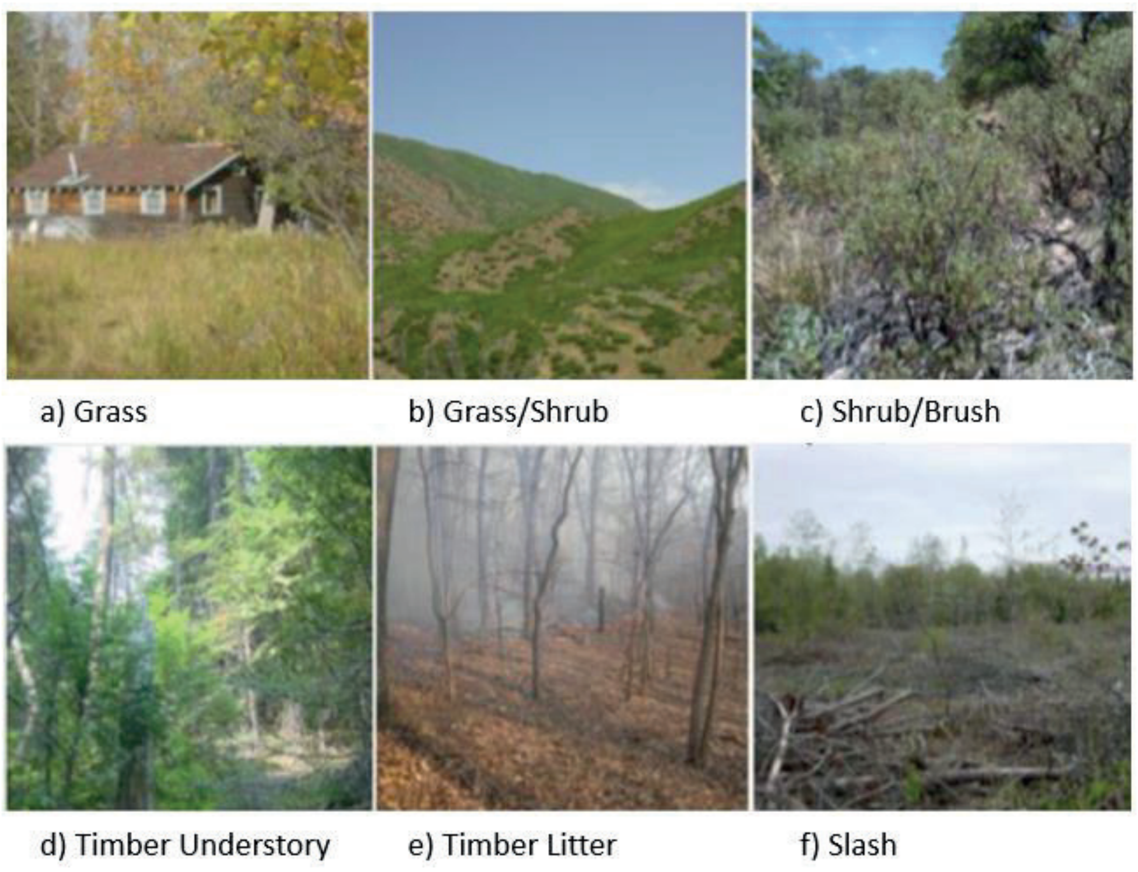


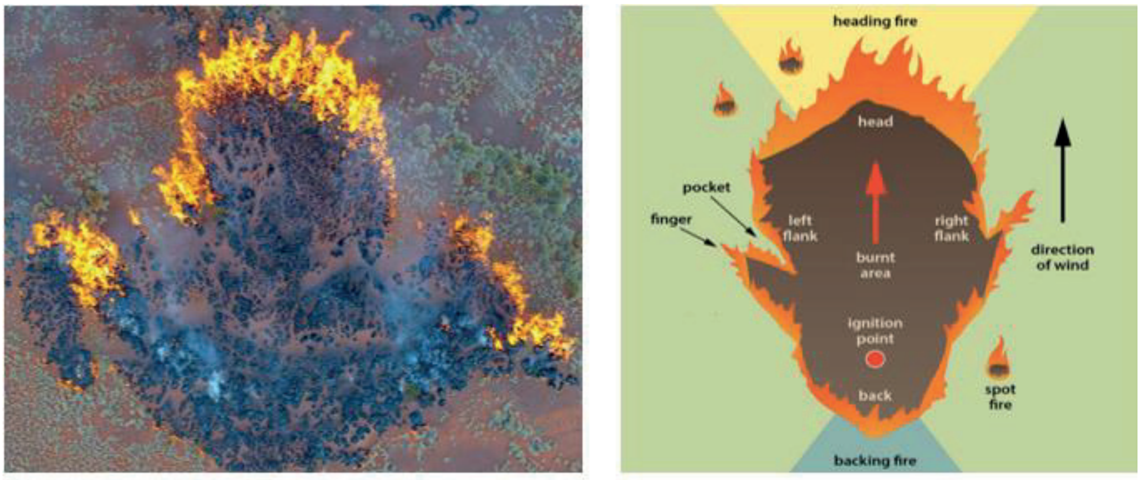
*Figure 7: Surface fuels divided into 6 groups to aid identification.*

### 2.1.4. Ignition

Ignition sources are usually divided into natural and human causes. Natural causes are primarily lightning, but they can also come from geological (volcanic eruptions), sparks from rock falls, spontaneous combustion, and even fires being ignited by the sun's heat. Human causes include both accidentally and intentionally ignited fires. Intentional or malicious setting of fire to property, especially public lands, intending to cause harm or defraud is called *arson* and is a criminal act. Another reason for human caused fires are vehicle crashes or malfunctioning, unextinguished cigarettes, irresponsible campers, burning debris and damaged electrical power. In general terms, lightning is responsible for over 10% of wildland fire ignition, while humans are responsible for starting almost all of the 90% remaining ignitions [47] [48].

## *Forest fire part names*

Starting from the ignition source, the components of fire behavior (fuel, weather, and topography) influences how fires spread and behave. They ultimately define what becomes the head, flank, finger, pockets or back of the fire. The parts of a wildfire follow the nomenclature described below.



a) Aerial image of real prescribed fire    b) Wildfire nomenclature

*Figure 8: The parts of a forest fire.*

- The *ignition* source is the location inside the fire perimeter where the ignition first occurred. It can be multiple points or even an entire perimeter.
- The *head* of the fire is the part of the fire perimeter that presents the greatest rate of spread and fire line intensity. It is in general on the downwind or upslope section of the fire. It is the fastest-moving, hottest, and most dangerous part of a wildfire.
- The *back* of a fire is the parcel of the fire perimeter located in opposition to the head. This part represents the slowest spreading part of the fire, usually opposed to the wind direction.

- The *fingers* of a fire are elongated burned areas that are projected from the main part (or body) of the fire. They result in a fire perimeter with a very irregular shape.
- The *pockets* are indentations in the fire perimeter, mainly located among two fingers.
- The *flanks* or sides are the fragments of the fire perimeter that are between the head and the back of the fire. These portions are approximately parallel to the leading direction of fire spread.
- An *island* (not pictured) is an area comprised of unburned fuels that is located inside the fire perimeter.
- *Spot fire* occurs when sparks, firebrands, and embers travel transported by the convection column or by the wind current and ends up landing to ignite new fires located outside the main fire perimeter.
- The *perimeter* is the entirety of the outside edge (or boundary) of a burning area or fire.

### 2.1.5. Types of fires

There are basically three types of forest fires: ground fire, surface fire, and crown fire [49] [50]. During a forest fire, it is not uncommon to have all three types of fire at once in the same event. The proportion of each type varies depending on the components of the environment fire triangle. Topography, fuel, and weather conditions are the main determinants of a fire behavior. A change in any of three components can make a ground fire to emerge as a surface fire, as well as make a surface fire to transition into a crown fire. The cycle can repeat in any direction during a wildfire.

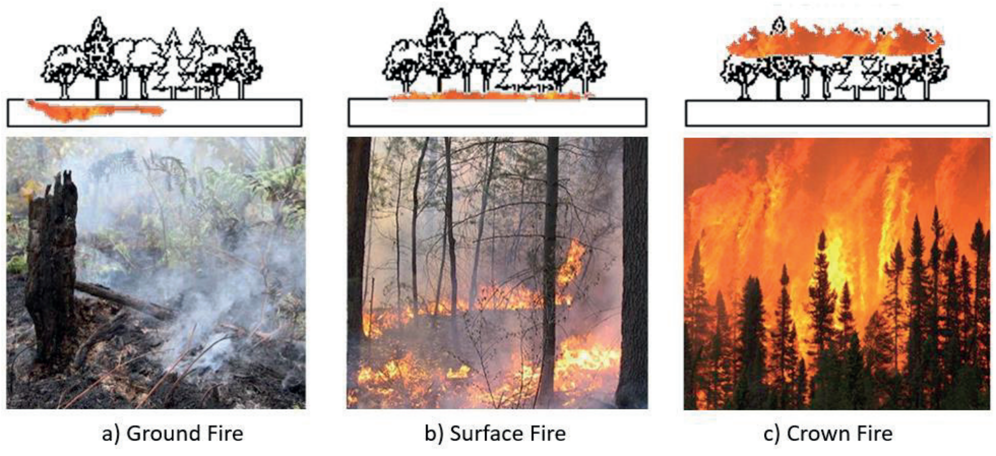


*Figure 9: Types of fire considering the major fuel being consumed.*

### *Ground fires*

Ground fires burn the ground fuel, the duff layer composed of matter in state of decomposition, and roots below the ground. Downed leaves, bark, twigs, sticks, and needles are presented in compacted form and feeds a smoldering fire. Glowing combustion (without flames) sustains these types of fires and they can spread hidden, slowly and undetected for a long period of time as they produce almost no visible smoke.

### *Surface fires*

Surface fires feed on sticks on the ground, tinder, needles, even moss and lichen, shrubs, small trees, herbaceous vegetation, and plants that are at or near the surface, over the ground level. They spread by flaming combustion, determining most of a fire perimeter, scorching large areas and growing vastly in extension. Surface fires can also transition and consume the forest canopy, a characteristic that is seen in crown fires, depending on:

- the amount of surface fuel.
- fuel moisture content.
- slope and/or wind speed.
- the resultant surface flame length.
- the height to the base of tree crowns.
- and the density and compactness of tree crowns.

### *Crown fires*

Crown fires feed on the canopy of the tree, which include live and dead branches and leaves, and tall shrubs [51]. As already mentioned, a crown fire can have its origin in a surface fire. Firebrands (also called *embers*) can be lifted outside the fire perimeter by the smoke column and start new fires (*spotting*) [52]. Figure 10 illustrates a crown fire and the occurrence of spot fires.

Depending on the environmental conditions and on the rate of spread, crown fires can also be passive, active, and independent. Passive crown fires burn individual trees or small groups of them (so called torching). An active crown fire, also called a running crown fire, is formed by a solid front vertical line of flame including the surface and the canopy fuel. Active crown fires propagate from one crown to the next one using canopy pathway. Figure 11 illustrates each one of these types of crown fires, always in relation to the surface fire spread.

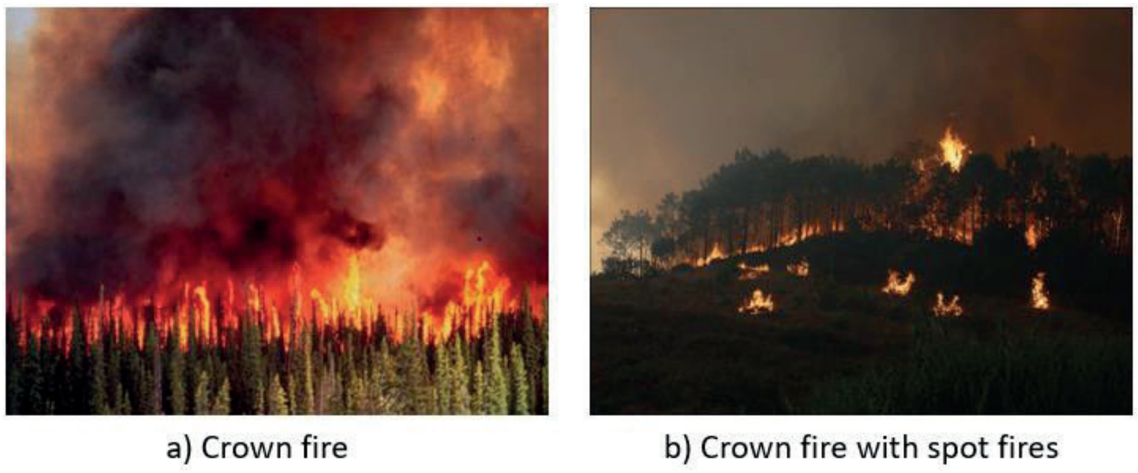


*Figure 10: A crown fire (a) and a crown fire with spot fires (b).*

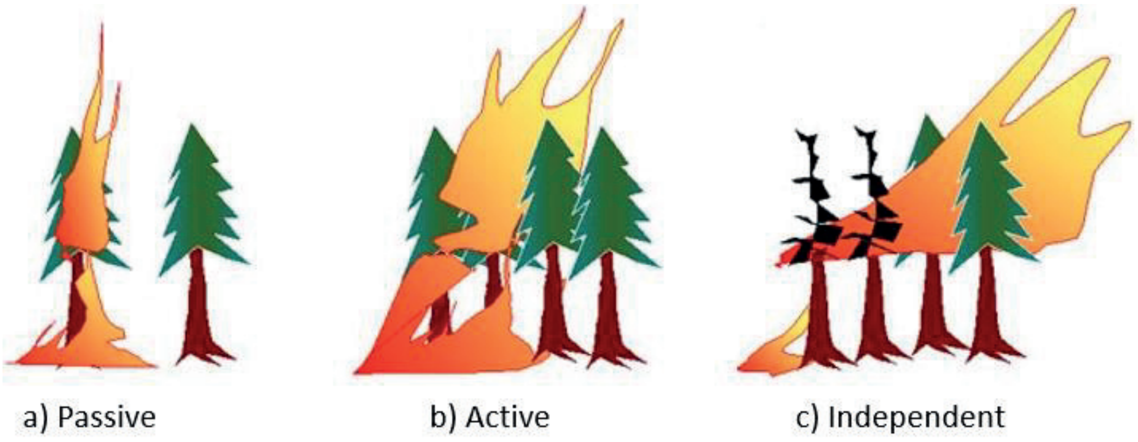


*Figure 11: Types of crown fires regarding spread rate.*

## 2.2. Forest fire modeling and simulation

Forest fire simulation systems (*fire simulator* for short) are software applications that predict future fire spread and behavior. A fire simulator is composed of one or more inter-related fire prediction models. Fire prediction models are the core of any forest fire simulator. It considers the environmental conditions for simulating the fire propagation. The fire simulation technique determines how the parameters that describe the fire spread and behavior are compounded throughout the terrain, surface, and the entire landscape.

Objectively, fire prediction models are a collection of equations. For any of them, the solution gives numerical values for variables representing the fire evolution. Are examples of these variables, the ignition risk, fuel consumption, rate of spread, fire intensity, and flame height. The fire evolution occurs both in time and in space, and there is a variety of classification systems for fire prediction models. These classifications consider how the equations models the flow of energy and heat transfer, the variables being studied, or the system being modelled, whether physical, experimental, empirical, or theoretical.

### 2.2.1. Classification based on equation modeling

The objective of the mathematical models is to capture the flow of energy produced during the process of combustion. Their release of energy and its propagation to neighboring unburned fuels is the essential part of a fire spread. In this way, unburned fuels are heated to reach the temperature of ignition. Then, they predict the evolution and propagation of a fire based on the defined flow of energy, both in time and in space. There are distinct methods that calculate the values for these processes, both in terms of quantity and quality. Among them, the most common are *physical*, *empirical*, *probabilistic*, and semi-empirical.

***Physical*** (or theoretical) models result from mathematical analysis of combustion, heat transfer (radiation, convection, and conduction), and physical-chemical laws of fluid mechanics. In a real scenario, they need a vast amount of information and data regarding all parameters of the chemical processes, using various variables as input, like flame height and temperature. Considering the complexity of equations and fuel structure, many simplifications are necessary, like uniformity of forest fuel and a constant process of heat transfer.

***Empirical*** (or statistical) models use correlations for fire spread and behavior obtained from controlled fires. As each testing fire has unique environmental conditions, statistical correlations are difficult to be generalized. Their application is restricted to the conditions like the ones present in the fires used for test.

***Probabilistic*** models use contingency tables instead of statistical or physical equations. They are not used to predict fire behavior, but to estimate probability of spread considering hypothetical fires over a landscape or to simulate ignitions.

**Semi-*empirical*** (or semi-*physical*) models combine physical (theoretical) and empirical (statistical) models. They use physical theory about combustion and heat transfer processes with knowledge acquired from analysis of fire-experiments in laboratory and field observations. From the empirical observations, mathematical models are formulated to describe the fire behavior. In these models, the source of heat comes from the burning fuel, whereas moisture represents sinks. Physical properties of fuel, slope, and weather contribute to each one of these variables, and experiments in laboratory and in the field are conducted to determine in each extend these contributions occur. From the semi-empirical predictive models, ***Rothermel's model (1972)*** is the most used.

Semi-empirical models require validation, as environmental conditions might be different from the ones found in controlled scenarios, in laboratory or from field observations. Nevertheless, even for challenging scenarios with steep slope, strong winds, and high temperatures, it is easier to validate semi-empirical models than the purely physical ones. Such fire propagation models are crafted to acquire the physical variables used to the calculate the advance of the fire perimeter. The main variables are *rate of spread*, *fire line intensity* and *fuel consumption*. Fire front property models, in turn, describe geometric flame features such as length, height, angle of inclination, and depth.

### 2.2.2. Classification based on the fuel being consumed

Considering the major type of fuel being consumed, a forest fire model can be classified as *ground*, *surface*, *crown*, and *spot* fire.

#### *Ground fire model*

The predictive models for ground fire determine the behavior of these fires that affect the ground layer. Ground fires present extremely low spread rate and propagate by smoldering combustion; as the ground fires feed on the organic soil and heat the inorganic matter, they damage the biologic activities of the forest. Unfortunately, there aren't many studies modelling ground fires. Most of the research is on the heat transfer process and on the probability of ignition.

#### *Surface fire model*

The predictive models forecast the fire spread on surface fuels, composed of small trees, shrubs, herbaceous vegetation, and other surface fuels. In general, fuel lower than 2m are considered surface fuel. The elliptical wave propagation technique, which considers that fire spreads approximately in an elliptical shape, is used in the majority of the empirical surface fire models. This technique considers that the major axis is in the same direction of the wind, the maximum speed also depends on the wind velocity; and the ignition point determines the ellipse focus.

The parameters needed to predict the fire behavior depends on several features of environmental conditions, like type of fuel, fuel moisture, wind speed and direction, precipitation, air temperature and humidity. From these parameters, the model needs to determine spread rate and fire front intensity.

In the beginning, these predictive models were mostly theoretical and consider only the hypothesis of a one-dimensional, steady, fire line spread, advancing against a homogeneous fuel bed. This fuel bed was characterized by moisture content, packing ratio and surface-area-to-volume (*SAV*) ratio of its constituent particles, which were assumed to be uniformly distributed in all directions. The semi-empirical model of *Rothermel* (1972) has been validated against different scenarios with a major success. It models the fire behavior by considering the global balance of the energy. Using data related to the fuels (i.e., heat content; fuel load; and moisture; etc.). This model predicts the parameters that describes the fire propagation.

### *Crown fire model*

A crown fire is a fire that in conjunction with the surface fire advances through the crown fuel layer. They can be classified according to the level of dependence on the surface fire. Crowning happens when a fire ascends into the crowns of trees and spreads from crown to crown. Predictive models for a crown fire forecast the behavior of fires spreading both on surface and aerial fuels. Independently if it is an empirical or theoretical equations, modelling of crown fires is complex, not to mention the validation process. The predictive models related to crown fires are divided in *crown fire spread models* and *crown fire initiation models*.

### *Spot fire model*

The models for spot fire predicts the aerial movement of embers (firebrand), due to the effect of convective column or the wind current. In this way, the ignition of fires totally new and outside the main fire perimeter is possible, even crossing barriers. The distance that a traveling ember can cover, the probability of ignition of new fires, and the maximum distance that a firebrand (considering cylindrical shape) can travel are the main characteristics of spot fire models. The spot fire modelling problem is attacked by considering the production of firebrand, the combustion in the wind, its trajectory, landing, and the ignition of new fires

## 2.3. FARSITE: Fire Area Simulator

FARSITE (*Fire Area Simulator*) [53] is a well-known fire growth simulation modeling system which uses spatial information on topography and fuels along with weather and wind inputs. With FARSITE it is possible to compute wildfire growth and behavior for long time periods under heterogeneous conditions of terrain, fuels, and weather. Models for surface fire, crown fire, spotting, post-frontal combustion, and fire acceleration are incorporated into a two-dimensional fire growth model. A FARSITE simulation generates a sequence of fire perimeters representing the growth of a fire under given input condition. For that purpose, it incorporates, among others, the simple but effective Rothermel's surface fire spread behavior model [43] along with the *Huygens's Principle* of wave propagation [53]. Although being a deterministic modeling system, a forest fire spread simulated with FARSITE is a process inherently complex, from which a long execution time for an individual simulation is not atypical. It is worthwhile to mention that apart from being a compute-intensive simulator, for complex scenarios involving large fires, the memory consumption can also be a bottleneck.

### 2.3.1. Incorporated fire models

FARSITE incorporates separate models for distinct types of fire behavior: point-source fire acceleration, post frontal combustion, surface fire, crown fire, and spotting. Ground fire is the only type of fire that is not incorporated in FARSITE.

The incorporated fire behavior models are linked and inter-related. As a sequence of fire activities, all the types of fires can be linked to each other given the favorable conditions. A fire can start propagating only as a surface fire, but. If the environmental conditions are favorable, the fire accelerates reaching a new condition of equilibrium. Given favorable weather, topography, and sufficient fuels, the fire might transition to a crown fire, using the so-called ladder fuels. If crown fuels are ignited, trees torch and may loft firebrands, potentially given origin to new fires (spotting).

#### *Surface fire model*

As the surface fires burn the surface fuels, a surface fire model is responsible for characterizing how the fire consumes the available fuel and propagates into the surrounding area. Three empirical characteristics are considered in modeling a surface fire: the elliptical shape of the surface fire propagation; the Huygens's principle of wave propagation; and the Rothermel's fire spread equations.

## The elliptical shape of two-dimensional fire propagation

Two-dimensional fire shapes are assumed to be generally ellipsoidal under uniform conditions, i.e., when factors affecting fire behavior are spatially and temporally constant. A fire propagating in a terrain with fuel conditions that remain constant and uniform, the same occurring for the slope, the wind, and the fuel moisture, is assumed to present a simple ellipse shape. Under these conditions, shapes from ovoid to pair of ellipses, and fan shaped have been experimentally observed. The difference between the shapes from simple ellipse to the alternate shapes are in the rear part of the fire. In this region, a small portion of the area is burned when compared to the head and the flank sides.

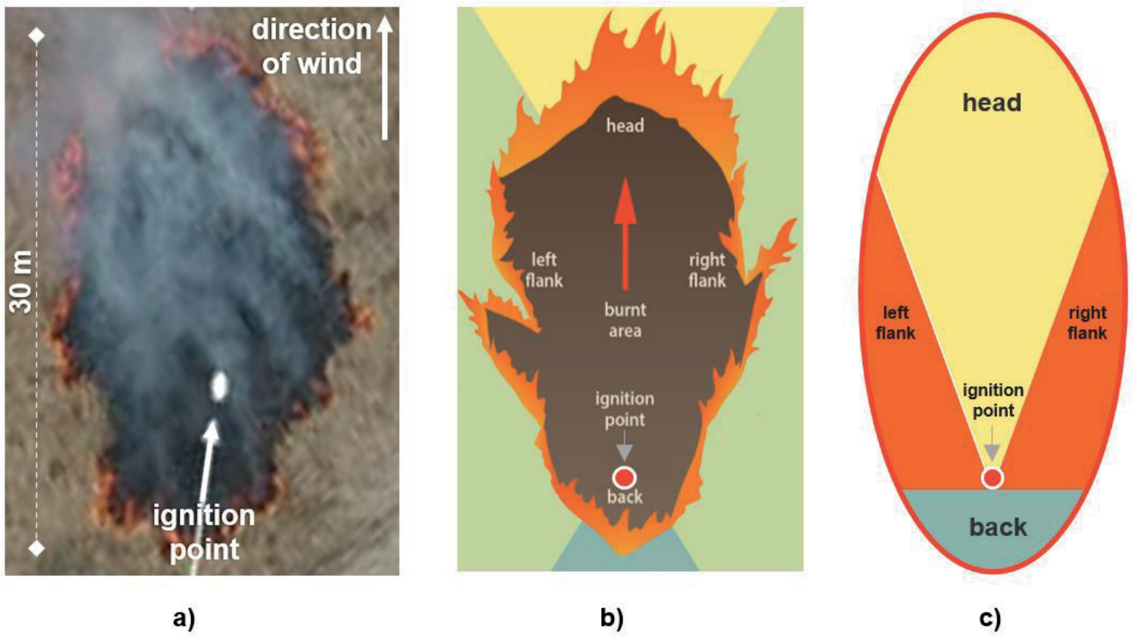


*Figure 12: A controlled fire (a), its schematic (b) and ellipsoidal shape (c) considering uniform conditions.*

Although not necessarily correct, it is assumed that the ignition point coincides with the focus in the rear of the ellipse. As advantage, this strategy provides also a backing fire spread rate. The location of the origin of the fire in other point of the major axis can be calculated, given and independent backing spread rate. The first strategy, using the source of the fire as the rear focus is largely adopted.

## Huygens's principle of wave propagation

The Huygens–Fresnel principle states that every point on a wavefront may be regarded as the source of new wavelets, and the aggregation of these spherical wavelets forms a new wavefront. Following Huygens's principle, Richards [54] developed a set of differential equations used to describe the fire propagation, by the expansion of an elliptical wave front. His method considers non-uniform environmental conditions, in which shape and size of the ellipses are dependent on fuel, wind, and slope. The fire can also propagate based on distinct elliptical shapes when the conditions change, i.e., for variations in slope or wind speed.

Using the elliptical wave propagation technique developed by Richards, FARSITE simulates the fire spread, in which the fire front propagates as an expanding fire polygon, growing at specified timesteps. Based on Huygens' principle, regularly spaced points on the fire perimeter are considered individually as part of the propagation: each point, or vertex, is the source of ignition of a small fire that spreads outward. Figure 13 illustrates this method for uniform and nonuniform conditions. In (a) uniform conditions use wavelets of constant shape and size, and in (b) nonuniform conditions showing the dependency of wavelet size and shape on fuel type and wind-slope vector.

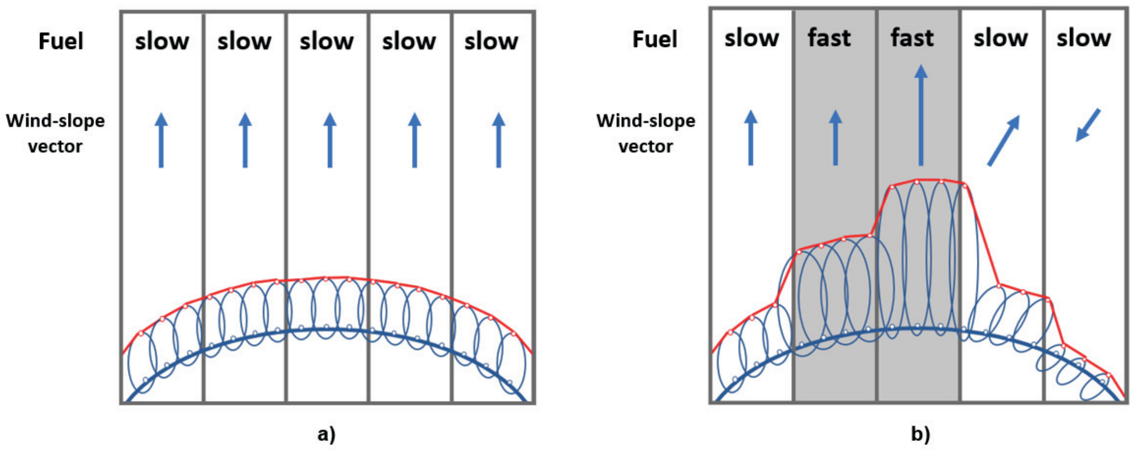


*Figure 13: Huygens's principle of wave propagation.*

The fire perimeter for each timestep is formed by the aggregation of the small ellipses that propagate from single vertices. By computing the spread rate and direction from each vertex and multiplying by the duration of each timestep is possible to expand the fire polygon. The wind-slope vector determines the shape and the direction of the small ellipses, while spread rate and length of the timestep determine its size. The shape of the ellipse, its direction and dimensions depends on these environmental conditions (see Figure 13b).

## Rothermel's fire spread equation

FARSITE uses ***Rothermel***'s fire spread model [43], a semi-empirical model validate on several experiments on the field. It has been developed to determine the variables defining the propagation of a two-dimensional steady surface fire, with uniform slope and uniform wind conditions, burning a homogeneous fuel bed. The main output produced by Rothermel's equation is the *rate of spread* ($R$), calculated as follows:

$$R = \frac{I_R \xi (1 + \Phi_w + \Phi_s)}{\rho_b \varepsilon Q_{ig}}$$

where:

| | |
|---|---|
| $R$ | is the heading fire steady state spread rate. |
| $I_R$ | is the reaction intensity. |
| $\xi$ | is the propagation flux ratio. |
| $\Phi_w$ | is the wind coefficient. |
| $\Phi_s$ | is the slope coefficient. |
| $\rho_b$ | is the ovendry bulk density. |
| $\varepsilon$ | is the effective heating number. |
| $Q_{ig}$ | is the heat of pre-ignition. |

The surface fire rate of spread is calculated for each vertex in the fire polygon (see Figure 13), burning in stationary regime, and spreading on a plane parallel to the ground surface. In this model, the numerator represents the total heat that is released by the fire, while the denominator represents the heat that can be absorbed by the fuel.

The *surface fire intensity* ($I_b$) is obtained based the rate of spread ($R$) as follows:

$$I_b = hwR/60$$

where:

| | |
|---|---|
| $I_b$ | is the surface fire intensity. |
| $h$ | is the heat yield of the fuel. |
| $w$ | is the weight of the fuel per unit area. |
| $R/60$ | is the fire spread rate (in SI units). |

The variables describing the fuel characteristics are indispensable to determine the values of the Rothermel's equations. They are added in FARSITE fuel models as input. The user sets them implicitly before the simulation, by informing the fuel model being used. *FARSITE* calculates the frontal fire characteristics (*spread rate* and surface *fire intensity*) for a fire in steady state regime and each calculation depends on the current environmental conditions.

### Crown fire model

The crown fire model used in FARSITE determines if the fire remains burning only in surface fuels or if it transitions to burn in the crown fuels. Once the transition is made, it also determines if it will spread actively through tree crowns (*active crown fire*) or simply torches individual trees (*passive crown fire*). This crown fire model [55] assumes that exists a threshold ($I_o$) for transition to a crown fire: this value depends on the *crown foliar moisture* content and on the *height to crown base* (also considering the presence of *ladder fuels*). Transition to crown fire occurs if the surface fire intensity $I_b$ is greater than or at least equals to the defined threshold $I_o$.

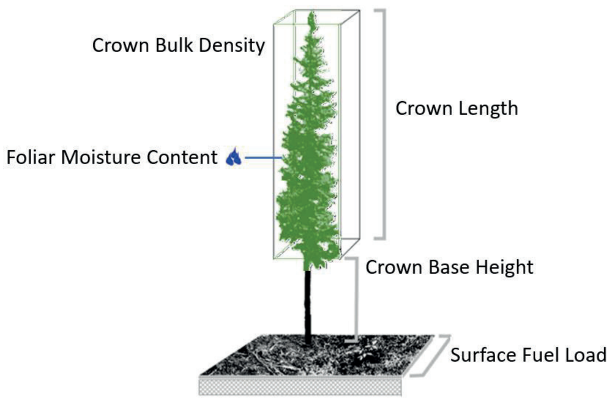


*Figure 14: Parameters involved in a crown fire modelling.*

If the crown fire will be *active* it will depend on the actual and critical energy flux in the advancing fire direction. The spread rate of a passive crown fire is assumed equal to that of the surface fire, whereas the actual active crown fire spread rate is determined from the maximum crown fire spread rate. The crown fire model involves statistical correlations and other empirical values to estimate crown fraction burned, active crown fire spread rate, and considerations to relate it with the spotting model. The reference material from Finney [53] and Van Wagner [55] can be consulted for a comprehensive understanding of the crown fire modelling.

### *Spotting model*

FARSITE uses a spotting model based on the originated from torching trees in a crown fire. The distance an ember travels depends on the ember size, the vertical wind speed, and the topography in the ember travelling direction. The larger the firebrand, the longer it can burn and the higher is the final velocity, but it won't be lofted as high as the small ones. The lofting height of embers with a defined diameter and characteristics can be calculated considering some assumptions:

a) particles are cylinders with constant specific gravity and drag coefficient.
b) particles are assumed to originate at the top of the tree canopy.
c) flame base is assumed equal to half the stand height.
d) particles are lofted vertically above the burning tree canopy.

The firebrands start a descending trajectory based on the wind speed and direction: the wind speed is assumed to have only a horizontal component at 6.1m (20 feet) above the ground, and it increases logarithmically considering the velocity of reference. Once the firebrand loses density and volume during the burning period, it follows at a decreasing rate, also considering the gravity force. It is up to the user to define a frequency of ignition as input, as several firebrands that drop on the ground can't originate a new fire. Many elements influence the frequency of ignition, but the most important are the moisture content, heterogeneity, and temperature of the fuel. Figure 15 illustrates the main factors affecting the spotting model.

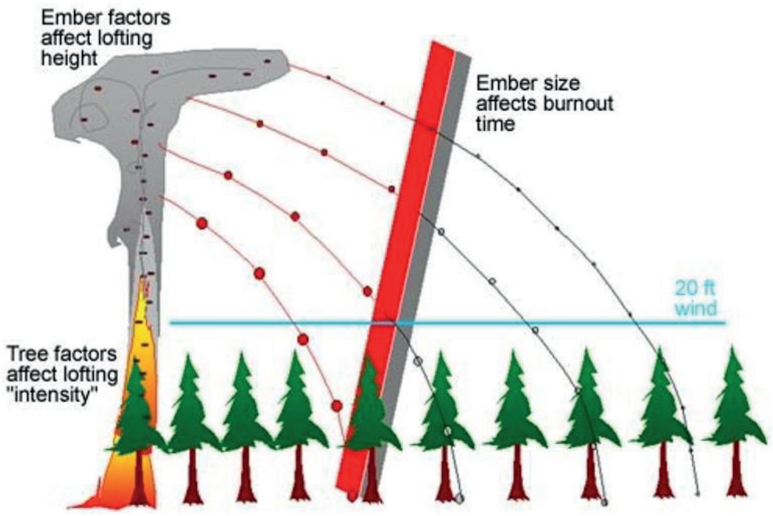

Ember factors affect lofting height
Ember size affects burnout time
20 ft wind
Tree factors affect lofting "intensity"

*Figure 15: Factors affecting spotting.*

### *Burn up model*

FARSITE uses a *burn up* model to emulate the fuel combustion. This model is inter-related with flaming and smoldering fire activities, that occurs after the main fire front. This model considers the fuel complex history, the flux of heating, the emissions of gases, and the heating of the soil heating. Each class of fuel model has its own fuel complex, moisture, heat content, and density. As input, the model requires the initial fire conditions responsible for igniting the fuel, i.e., the *residence time* and *fire intensity* level. Moisture contents of woody fuels, duff, the wind speed, and other environmental conditions are also required. Apart from these, this model also needs the duff and woody fuel load. As outputs, it produces at each timestep the *fire intensity* and *fuel weight loss*.

### *Point-source fire acceleration*

One important model incorporated in FARSITE is the fire acceleration model. Such model allows smooth transition from slow to faster spread rates, avoiding instantaneous unrealistic jumps to faster spread rates after changes to faster fuel types, sudden increases in windspeed, or occurrence of steeper slopes. Fire acceleration is the rate of increase in spread rate for a given ignition source assuming all environmental conditions remain constant. Fire acceleration occurs also due to the ignition of additional fuels, due to dry of fuels during the day.

As the simulation progresses through dynamic environmental conditions, these changes might create higher potential levels of a fire propagation rate equilibrium. FARSITE acceleration model is based on the simple logarithmic formula for point-source, that assumes the fire spread rate is dependent on only the time allowed for accelerating to the maximum rate possible under the current conditions. The model also includes acceleration coefficients depending on fuel type and other constants defined empirically.

Although a smooth transition from slow to faster spread rates is necessary, fires are assumed to decelerate instantly when encountering new environmental conditions that produce a slower fire spread rate. The reference material from Finney [53] can be consulted for a detailed view of the fire acceleration model and its equations.

### 2.3.2. Fuel model

A fuel model is a set of fuel bed characteristics used as input for a forest fire modeling application. Fire behavior models consider numerous empirical variables that are often difficult and time-consuming to measure for each fuel bed. A fuel model then defines these input variables for a stylized set of quantitative vegetation characteristics that can be visually identified in the field.

Fuel models are tools to help the user realistically estimate fire behavior [46]. The major fuel model classification systems for use in the United States include the National Fire Danger Rating System, the 13 original fuel models of Anderson and Albini [46], and the subsequent set of 40 fuels produced by Scott and Burgan [56]. These two most recent works present conversion crosswalk to allow backwards compatibility.

#### *National Fire Danger Rating System*

The concept of a fuel model was first introduced in 1972 with the *National Fire Danger Rating System* (NFDRS), a standardized set of equations to determine fire danger at specific points on the landscape [57]. Fuel models were at the core of these calculations, with each of its 20 models containing information about the relative loading of different fuel components. Each model is described by the volume of dead and live fuels present as well as the fuel bed depth and moisture of extinction.

### *Albini and Anderson's Models*

In 1976, Albini [58] presented 13 fuel models. Anderson [46] expanded the work in 1982. These fuel models were designed for use with Rothermel's spread model and are designed to be used at much smaller spatial scales when compared to the *NFDRS* models. These fuel models are designed to be used during the dry season, when the fuel bed becomes more uniform.

These models are grouped into four classes:

- Grass Group: short grass, timber Grass and understory, and tall grass.
- Shrub Group: chaparral, brush, dormant brush, and southern rough.
- Timber Group: compact timber litter, hardwood litter, timber understory.
- Slash Group: light slash, medium slash, heavy slash.

### *Scott and Burgan's Dynamic Models*

Scott and Burgan's dynamic fuel models were published [56] in 2005 to eliminate the assumption that the fuel bed was uniform during the dry season. They use dynamic herbaceous fuel beds, in which the live herbaceous load is transferred to dead depending on its moisture content. Fire behaviors in herbaceous fuel beds are more realistically modelled by using a curing coefficient for each live fuel.

## 2.4. Input data acquisition

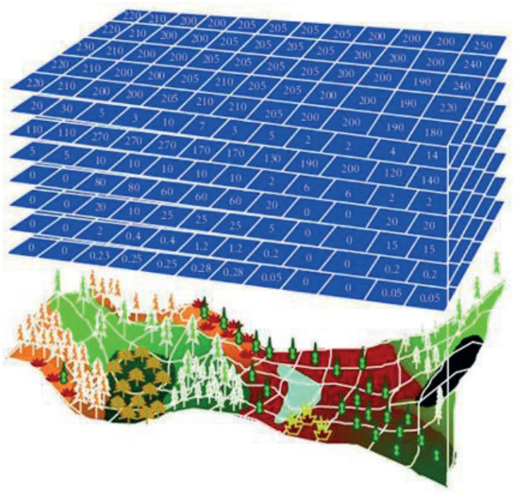
A limiting factor to geospatial fire modeling is the data acquisition. FARSITE landscape file (*LCP*) preparation is sometimes limited to geographic areas with the expertise, financial support, and need for fire modeling. Fire and fuel management need to evaluate, reproject, modify, and maintain these geospatial datasets to guarantee an accurate information. Apart from that, selecting the right personnel with the appropriate level of fire behavior and modeling experience and adequate local knowledge is critical to a successful outcome.

The analysis team should be composed of individuals that: have observed fires locally in various fuel and weather conditions; with geospatial fire modeling experience to help bridge the gap between field observations and the modeling systems; familiar with vegetation type, distribution, and characteristics; with geographic information system (*GIS*) expertise, particularly in grid analysis with raster datasets.

Unfortunately, geospatial fire data receive attention only when there is a need for decision support for a major incident. When the analysis or product is needed, the first few days of the incident are spent assembling the data, becoming familiar with it, tracking down supporting documentation, and updating the layers. Of course, to update critical data under pressure is far from ideal. Nevertheless, once the data is prepared, fire specialists can use the data year-round for prescribed fire and fuels planning.

### 2.4.1. Landscape file

There are several ways to obtain geospatial data for fire modeling. The most common data format for geospatial fire analysis is the FARSITE landscape file (or LCP for short), a single binary file consisting of elevation, slope, aspect, fuel model, and canopy cover, as well as optional themes of stand height, crown base height, canopy bulk density, duff loading, and coarse woody debris. Figure 16 presents a schematic and the units of these layers.



a) Schematic

| Layer | Units |
|---|---|
| Elevation | meters |
| Slope | degrees |
| Aspect | azimuth degrees |
| Fuel model | n/a, 1-999 |
| Canopy cover | percent |
| Stand height | meters, * 10 |
| Crown base height | meters, * 10 |
| Canopy bulk density | Kg/m^3, * 100 |
| Duff loading | metric tons per hectare |
| Coarse woody debris | n/a, 1-99 |

b) Layers

*Figure 16: FARSITE Landscape file (LCP) schematic and its layers units.*

The LCP is built by importing each theme as an ASCII Raster – a common file format for an exported *GIS* layer. A map projection is a mathematical calculation to portray all or part of the earth on a flat surface. A local area will likely use a different coordinate system and data may need to be reprojected to the local projection to minimize error and enable local *GIS* data overlays.

The fire analyst needs to scrutinize the input data both numerically and visually, and only later they are ready to examine the data and models based on fire behavior output. Analyzing the outputs can validate suspicions one has when critiquing the inputs and reveal new problems. Although nowadays the topographic area characterization needs to be made only once (elevation, slope, and aspect), the other layers need to be maintained. *GIS* tools are generally used to make straightforward changes to landscape files, accounting for recent burns new urban structures, changes to fuel models). Nevertheless, these data don't change often and can be considered semi-static and be prepared up-front.

### 2.4.2. Meteorological conditions

Weather data for simulations can be obtained from *Remote Automatic Weather Stations* (*RAWS*). The basic weather reading consists of precipitation, relative humidity, temperature, wind direction, and wind speed.

As wind is one of the most influential environmental factors affecting wildland fire behavior, fire models can be coupled with wind models for better prediction results. For example, one of these wind models is WindNinja [59], a microscale wind model developed for use in wildland fire applications. It computes spatially varying wind fields, generating high resolution wind prediction in complex terrain. It is specifically designed to simulate the effect of terrain on wind flow, and it can use information from standard weather forecasts to help determine the future wind inputs. It can be used to generate wind direction and speed vectors using the data from the weather stations at hourly time steps with finer spatial resolution (e. g. 200m). One interesting use in the suppression activities in the forecast mode, where it uses coarser resolution mesoscale weather model data from the weather services to forecast wind at future times.

### 2.4.3. Vegetation and fuel model

Vegetation data can be acquired from national and continental agencies, like *Copernicus Land Monitoring Service* (*CLMS*) for Europe and the *National Land Cover Database* (*NLCD*) for the United States. Copernicus is the Program of the European Union for Earth Observation that offers information services based on satellite observation and in situ data. Likewise, *NLCD* is the national (USA) land cover product created by the Multi-Resolution Land Characteristics (*MRLC*) Consortium. These information services are freely and openly accessible to their users. They offer 100-meter and 30-meter resolution land cover maps that show the global distribution of 10 major land cover classes: water bodies, wetland, artificial surfaces, cultivated land, permanent snow and ice, forests, grasslands, shrubland, bare land, and tundra. To be used to generate a landscape file, the data from *CLMS* and *NLCD* databases needs post-processing for reprojection, selection, and fuel type conversion, if needed.

### 2.4.4. Fire perimeters

Fire perimeters allow the definition of an initial state of the forest fire, as well as to compare prediction against actual fire spread. Unfortunately, even ignition points and intermediate perimeters are subject to high levels of uncertainties in the moment of a disaster occurrence. Besides field observations, moderate resolution satellite images are used to estimate fire perimeters [60] [61]. In particular, Terra and Aqua satellites [62], and Landsat [63] and Copernicus [64] programs have optimal ground resolution and spectral bands to efficiently track land use and to document land change due to wildfire and a host of other natural and human-caused changes. Landsat has a finer resolution of 15 meters and 30 meters, but unfortunately its combined (Landsat 8 + Landsat 9) revisit time for data collection is eight days, too much longer for disaster response. For Terra and Aqua, the images are taken at a resolution from 250 meters to 1 km, and the satellites pass through a particular area twice a day. Using publicly available data from these global projects is the way-to-go option, but, with proprietary access to other satellite constellation, more coverage and higher resolution, intra-daily revisit can be also an option.

Observing surface temperatures from space can be difficult, however, as atmospheric moisture and air temperature can skew the signals. Despite of cloud cover, every surface emits thermal infrared radiation, or heat. By detecting radiation in two thermal wavelengths, satellite's instrument can measure the temperature of an observed area. Fire managers also analyze images from the shortwave-infrared band, which reflects strongly from exposed ground, to identify scorched areas, as well as the thermal observations to locate the perimeters of active fires. Since the thermal bands can penetrate smoke that might otherwise obscure the view, the resulting images can be used to estimate the fire perimeters of interest.

When a fire is detected, the first perimeter describing the burned area is computed by monitoring agencies. This perimeter, with a high degree of accuracy, is then made available in their systems, like in the European Forest Fire Information System (EFFIS) and the Fire Information for Resource Management System (FIRMS) in the United States and Canada. For example, Figure 17 shows the location of the 2013 Silver Fire in New Mexico, USA. The image on the left represents the status "before" the fire event, acquired on May 28, 2013. The middle image shows the location of the fire (bright red dot) and burn scar (dark red) on June 13, 2013, "during" event, while the fire was still growing. The image on the right is an example of a Burned Area Emergency Response, showing areas with high (red), moderate (yellow) and low (green) severity burns. We use perimeters from map like these to define the ignition shape that is fed into the forest fire simulator.

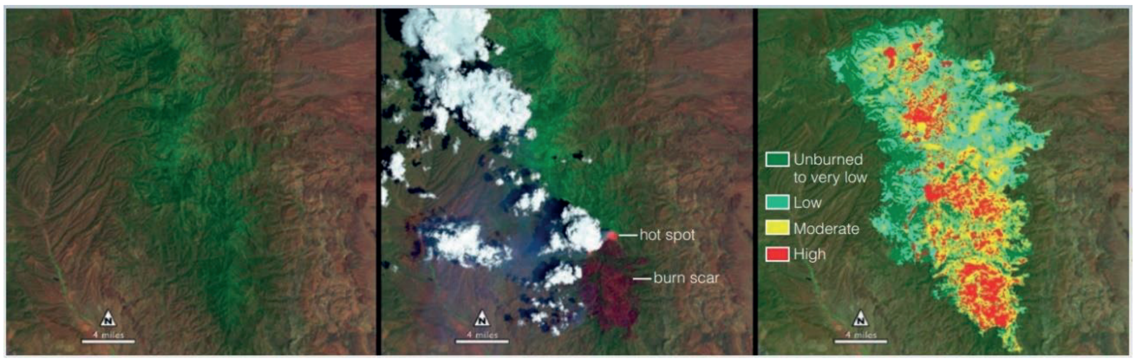


Figure 17: Burned Area Emergency Response Maps.

## 2.5. Model calibration

Calibration is a critical step to the analysis of any model. In the field of metrology, it is the comparison of measurement values delivered by a device being tested with the ones from a calibration standard of known accuracy. The term *calibration* is used freely in the wildland fire community, nevertheless the meaning is quite similar. To produce fire growth and behavior outputs consistent with observations, model checking, modifications, and comparisons are done with known fire perimeters and weather conditions [53]. If the model has not been calibrated to local fires, analysts and managers will have less confidence in the output. On the other hand, when fire behavior outputs have been critiqued and the model calibrated adequately, one can have a higher degree of confidence in future simulations [65].

In wildfire community, calibration is model specific, whether the calibration is being performed for fuel model or simulation parameters. Using FARISTE, the calibration can be made for longer period, it can be adjusted to crowning, spotting, and account for changes in weather, wind, and fuel moisture. Usually, the fire perimeter is the output that needs to be calibrated, considering the changes in different parameters in FARSITE modelling or simulation.

### 2.5.1. Manual calibration for a forest fire spread model

Considering the manual calibration, one can select a reference fire or run to calibrate the model under moderate conditions, and another for extreme conditions with and without a high wind event, for example. Apart from the landscape file, to perform a proper calibration, the fire analyst needs: a precise starting location; accurately burn period by the starting and ending time; progression layer or detailed field observations that identify the position and time of the fire; and representative weather and wind information.

To evaluate and calibrate forest fire growth and rate of spread, holding all environmental conditions constant, the fire analyst sets the initial conditions. Then after defining an ignition on the landscape, they start the simulation. When the simulation finishes, its output reveals the fire perimeter positions at a later instant in time. They can then compare the result with historical fire progression in *GIS* visualization tools.

The process can then be repeated, iteratively, importing an ignition file in the same projection (e.g., the previous perimeter). It is important to set the simulation time (in minutes) to match the spread event being calibrated. The target is to compare the output with the actual fire perimeter and determine how is the overall *fit* on the head, rear, and flank of the fire spread. If there is a major discrepancy, it is necessary to conduct additional runs to identify and solve the problem. Calibration is done by varying the maximum simulation time, fuel moisture, wind speed and direction, torching, crowning, spotting, etc.

If the changes in model parameters and environmental factors proven to be adequate to fit the historical fire perimeter, then the calibration is successful. If not, other variables could be a factor, like fuel model, crown base height, canopy cover, barriers not taken in account, or even the result human intervention (fire suppression activities).

A fire analyst wants to incorporate his calibration's findings to improve forest fire model output. Unfortunately, there are time restraints that limit how long one can take preparing the input data to be operationally used in the fire simulation. It is important to focus on the changes that affect the most ground and not be worrying too much with details. It is crucial to understand that, in the end, we use landscape fire growth modeling with imperfect data and models.

There are three sources of error: data, user, and model. Not only the data can be the reason for misfit results, but the model can also be the source of the problem. The analyst needs to adjust fuel model layer, vegetation, and crown canopy cover. Changes need to be traced, documented, and might even need to be backtracked. Of course, adjustments need to be made incrementally, as it is difficult to assess how a specific change affected a given fire behavior output if several changes are made all at once.

Given model assumptions and limitations, data inaccuracies, effects of fire suppression, variability in the weather and wind, a perfect calibration in a real scenario is not feasible. Even a calibration that is good enough takes time, skill, and patience. Every fire, landscape, and conditions are different; hence every calibration exercise will vary from one another. The manual calibration is crucial in forest fire management to understand the fire behavior and fuel model definition for a particular area, but it can not be performed during a real fire scenario, as it takes too much time to be performed.

### 2.5.2. Two-stage forest fire spread prediction framework

The tedious, error-prone approach described in the previous section can be systematized, automated, and then included in an advanced two-stage prediction model. If a time-constrained solution is provided, the two-stage model can be used in a real fire scenario, helping to deal with data uncertainties in real-time. This model uses a stream of real data to automatically adjust the unknown FARSITE parameters.

Usually, to predict forest fire behavior, a simulator takes the initial state of the fire front perimeter (***$P_0$***) along with other parameters as input. As output, the simulator then returns the fire front spread prediction for a later instant in time (***$P_1$***). After comparing the simulation result with the actual advanced fire front, the predicted fire line tends to differ from the actual one. Besides the natural phenomena modeling complexity uncertainty, the reason for this mismatch is that the classic scheme calculation is based solely on a single set of input parameters, affected by the data uncertainty. To overcome this drawback, a simulator independent data-driven prediction scheme was proposed to calibrate model input parameters [31]. Figure 18 illustrates this ***two-stage*** prediction method.

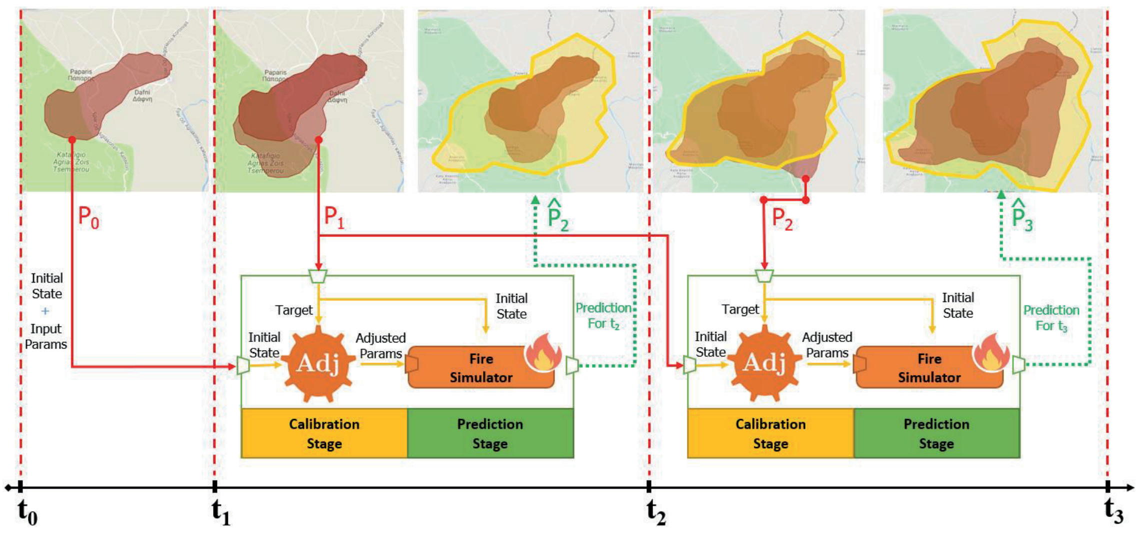


*Figure 18: Two-Stage Prediction Method.*

Introducing a previous adjustment stage, the set of input parameters is calibrated before every prediction step. Thus, the solution comes from reversing the problem, coming up with a parameter configuration such that the fire simulator would produce predictions that match the actual fire behavior. After detecting the simulator input that better reproduces the observed fire propagation, the same set of parameters is used to describe the conditions for the next prediction $\widehat{\boldsymbol{P}}_{\mathbf{2}}$, assuming that the meteorological circumstances remain constant during the next prediction interval. Then, the prediction becomes the result of a series of automatically adjusted input configurations. The process can be applied again for subsequent fire perimeters $\widehat{\boldsymbol{P}}_{\mathbf{3}}, \widehat{\boldsymbol{P}}_{\mathbf{4}}, \widehat{\boldsymbol{P}}_{\mathbf{5}}$ , and so on.

To enhance the quality of the predictions, as a data-driven prediction scheme, the two-stage method is applied continuously, providing calibrated parameters at different time intervals, and taking advantage of observed fire behavior and helping to reduce the negative effects related to the input-data uncertainty. The two-stage approach has been proven to be appropriate to enhance the quality of the predictions. In particular, a *Genetic Algorithm* (GA) based adjustment technique gives accurate results.

The genetic algorithm implemented in a Master-Worker paradigm starts from an initial random population of individuals, each one representing a scenario to be simulated. An individual is composed of a set of different genes that represent input variables such as dead fuel moisture, live fuel moisture, wind speed and direction, among others. Each one of the individuals is simulated, and it is evaluated comparing the predicted and the real fire propagation by estimating the fitness function (or prediction error function) based on the normalized symmetric difference (*NSD* for short) between predicted and real burned areas. Equation 1 defines how such difference is calculated, where ***Real*** is the area burned by the real fire at a certain time and ***Pred*** is the area burned by the predicted fire at the same time instant.

$$NSD = \frac{\bigcup(Real, Pred) - \bigcap(Real, Pred)}{Real} \tag{1}$$

As illustrated in Figure 19, the areas around the simulation map that have not been burned by neither the real fire nor the simulated fire are considered ***Correct Negatives***. Areas that have been burned by both fires are called ***Hits***. The areas that have been burned only in the real fire are called Misses, whereas areas that have been burned only in the simulated fire are called ***False Alarms***.

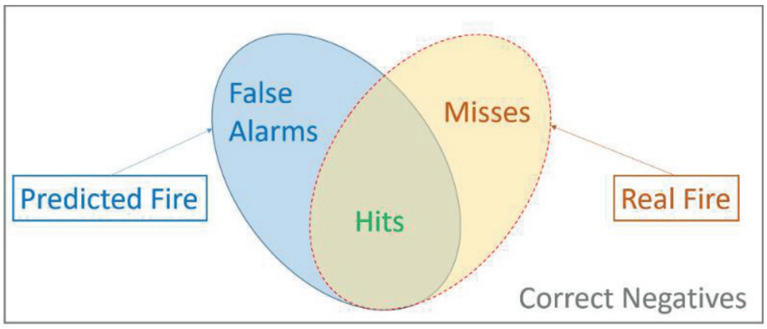


*Figure 19: Different categories present in forecast verification.*

With this categorization, the normalized symmetric difference can be rewritten as showed in Equation 2. This difference considers the wrongly predicted burned cells (*false alarms*) and the real burned cells that were not predicted (*misses*). The ultimate objective of a fire prediction strategy is to reduce this error function respecting a strict deadline.

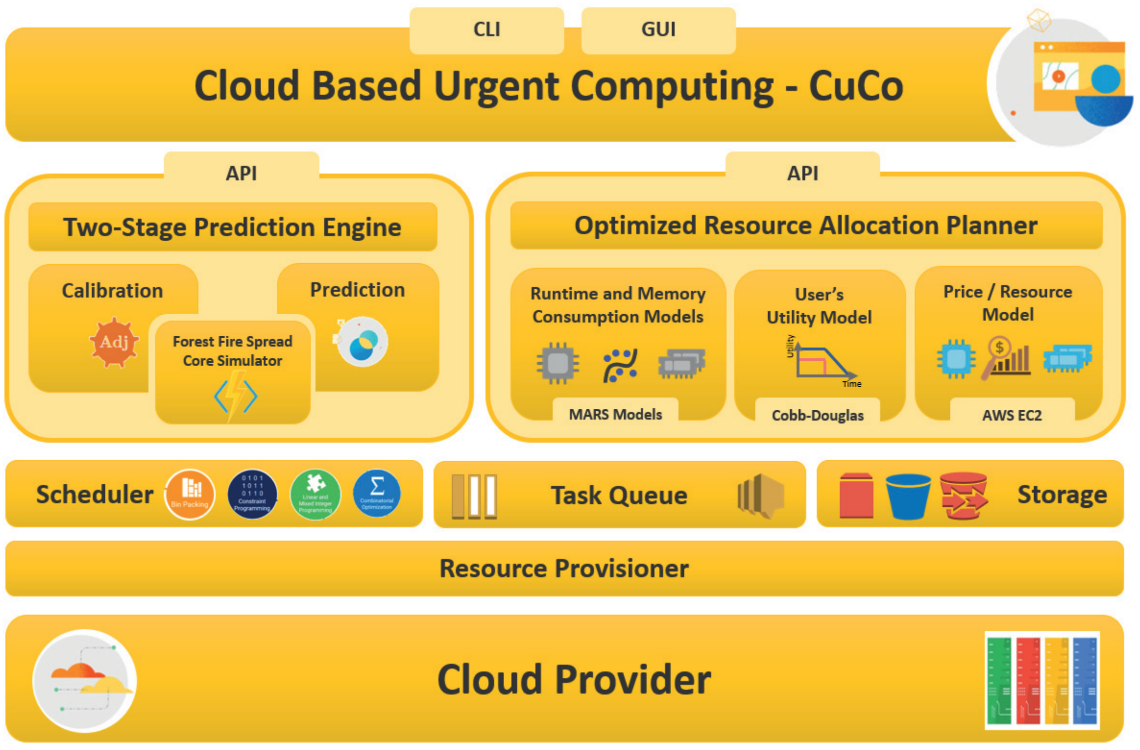
$$NSD = \frac{Misses + FalseAlarms}{Misses + Hits} \tag{2}$$

# Chapter 3: Cloud-Based Forest Fire Spread Prediction

During the work for this thesis, we have proposed a high-level architecture for a cloud-based urgent computing solution. In this chapter, we present this main contribution, together with and a proof-of-concept implementation that uses state-of-the-art cloud technologies. As secondary contributions, we have proposed new strategies to help decrease the overall calibration time of the Two-Stage Prediction Framework based on an early adaptive-evaluation technique, and a methodology to define a strict-deadline per generation.

## 3.1. CuCo's architecture

Figure 20 shows the high-level layered architecture for a *Cloud-based urgent Computing* (dubbed ***CuCo***) solution. The *Two-Stage Prediction Engine* is the heart of the application, being responsible for both the *calibration* and the *prediction* stages. There is also a component called *Optimized Resource Allocation Planner*, capable of minimizing cost while maintaining a prediction deadline.



CLI
GUI
Cloud Based Urgent Computing - CuCo
API
Two-Stage Prediction Engine
Calibration
Prediction
Adj
Forest Fire Spread Core Simulator
API
Optimized Resource Allocation Planner
Runtime and Memory Consumption Models
MARS Models
User's Utility Model
Utility
Time
Cobb-Douglas
Price / Resource Model
AWS EC2
Scheduler
Bin Packing
Constraint Programming
Linear and Mixed Integer Programming
Combinatorial Optimization
Task Queue
Storage
Resource Provisioner
Cloud Provider

*Figure 20: High-level architecture of the cloud-based two-stage fire spread prediction solution.*

The main components presented in this layered architecture are the following:

**Two-Stage Prediction Engine**: it is based on the two-stage prediction method, running a calibration phase before the actual fire spread prediction. The calibration is orchestrated by an iterative data-driven compute-intensive genetic algorithm that runs *FARSITE* as the *Core Simulator*.

**Optimized Resource Allocation Planner**: it is the component responsible for defining the resources where the calibration and the prediction will be executed. As a black box, it simply returns the list of available machines. But, in the details, it is a domain-specific module, tailored to statistically model the core simulator based on the specified computational resources properties, and determine the user's preferences through strict deadlines or utility functions.

- **Runtime and Memory Consumption Models**: statistical models are used to estimate the execution time of an individual fire spread simulation.
- **User's Utility Model**: utility functions are used to capture user's satisfaction over prediction time and monetary cost of the overall fire spread prediction.
- **Price / Resource Model**: we consider an *infrastructure-as-a-service* (*IaaS*) cloud system, in which data centers deliver on-demand storage and compute capacities over the Internet, charging on a *pay-as-you-go* model.

**Scheduler**: its function is to assign resources to perform computing tasks. It determines which resources are valid placements for each task according to their constraints.

**Task Queue**: it integrates the different components providing an Application Programming Interface (API) to create queues as well as send, receive, and delete messages. Messages are used to represent simulation tasks, and to queue states to trigger orchestration or scheduling activities.

**Storage**: as an urgent computing solution, it is necessary to have on-demand access to the input data. It is needed a persistent repository consisting of all the static data for the areas with a greater probability of forest fire occurrence (those with high fire danger indices). For these static data, as a data lake, the solution leverage object storage services that offers industry-leading scalability, data availability, security, and performance.

**Resource Provisioner**: it is responsible for providing the virtual resources in which the actual simulation will be run. After understanding the memory consumption and runtime of the core simulator, a suitable memory or compute-intensive pool of instances can then be provisioned.

**Cloud Provider**: A third-party company offering a cloud infrastructure, application, or storage services. Users access the resources on-demand, being charged on a pay-as-you-go basis. These computational resources are provided in the form of an abstract unit of compute and storage called Virtual Machine (VM for short), object storage, or remote file systems volumes. In a cloud, VMs are offered in different types, each of which has different characteristics such as different numbers of CPUs, amount of memory, and network bandwidths capacity.

### 3.1.1. Proof-of-concept implementation

In this section, the proof-of-concept implementation of the high-level architecture of the cloud-based solution is presented and discussed. We decided to leverage well-established cloud tools, favoring a decoupled solution that can be easily deployed on a public cloud or on-premises infrastructure. It also relies on scalable object-storage service to allow the efficient parallel retrieval and storage of input and output data.

The *Resource Provisioner* is linked to the *AWS* infrastructure (*Cloud Provider*) using *Elastic Compute Cloud (EC2) service*, *FARSITE* is the *Core Simulator,* and it has been isolated as a containerized application running on *Docker* runtime engine (*Container*). The stream of generations from the *Genetic Algorithm* is responsible for generating all the compute demands that will be fulfilled by the orchestrating component (*Kubernetes*), which takes the role of the *Scheduler*. For each generation, a *Kubernetes Job* performs a *coarse parallel processing using a work queue* (from *AWS Simple Queue Service* - SQS). For Storage of input data (*Landscape* and configuration files) as well as output results (prediction files) we use *AWS Simple Storage Service* – S3.

**Integration using a message queue service:** The initial population is randomly generated and, for each one of its individuals, a task definition is created and pushed to the *Task Queue*. Although we use *Simple Queue Service* from *AWS*, any local queue service could also be used. SQS has the advantage of being more scalable and dependable.

**Storing input and output:** To deal with the data acquisition latency problem, we rely on a well-established scalable object-storage service to allow the efficient parallel retrieval and storage of input and output data (*AWS S3*).

**Kubernetes Job:** The *Kubernetes Job* starts one *FARSITE* pod for each task in order to obtain maximum parallelism. Each of them takes one task from the task queue, processes it, and gets terminated. Once the current generation is entirely evaluated, a new one can be computed, and the task queue can be filled up again.

**Filling the queue with tasks:** One task represents a scenario describing parameters for every single *FARSITE* execution, taking the role of an individual of the *Genetic Algorithm*. In each message describing a task it also informed the *input bucket* and the *input directory name*. Likewise, the *output bucket* needs to be informed, accompanied by the name of the *directory* to which the output files are pushed.

**Dockerizing FARSITE:** A Docker template has been prepared to build a containerized version of *FARSITE*. In its first execution, the container attempts to consume a message from the task queue. In case there is at least one task, it proceeds to run the scenario that the task describes and, in the end, push the results to the output directory. At the end of the execution, the message gets deleted from the queue. Once there is no message to be consumed, the pod gets terminated.

**Reconciling**: Once the queue is empty, the *Genetic Algorithm* can take back the control of the execution and, if the evolution is not complete yet, applies the genetic operators (*elitism*, *selection*, *crossover*, and *mutation*) resulting in a new generation. The queue is filled up again and *Kubernetes* takes the control, proceeding to deploy, start, and at the end terminate all the pods used to run the individuals.

The process is repeated until the pre-defined number of generations is reached. In the end, the parameters and files from the best individuals are made available in the output bucket and they can then be used in the *Prediction Stage*. Figure 21 illustrates the interaction flow between the main CuCo's components for both the *Calibration* and *Prediction* stages. The wildfire analyst informs the scenario's configuration **(1)**, with all the static data available in a storage service. For each generation, the *Optimized Resource Allocation Planner* gets consulted **(2)**, and an allocation plan is defined **(3)**. The list of resources to be acquired is then informed to **(4)** and provisioned by the *Resource Provisioner*. It is responsible for requesting **(5)**, receiving **(6)**, and passing them to the *Two-Stage Prediction Engine* **(7)**. The *Scheduler* **(8)** receives the list of resources used to start a *Kubernetes Job* **(9)**. The *Task Queue* gets fed with the tasks **(10)** used to define each *FARSITE* scenario that later will be consumed **(11)** by the pods running in the *Kubernetes cluster*. Results are sent to the *output bucket* **(12)** until the end of the *Calibration Stage*. Once the *calibration* finishes, the adjusted parameters get passed on to the *Prediction Stage* **(13)** that runs the actual prediction on a high-performance container **(14)**. In the end, the prediction result **(15)** is made available to the wildfire analyst.

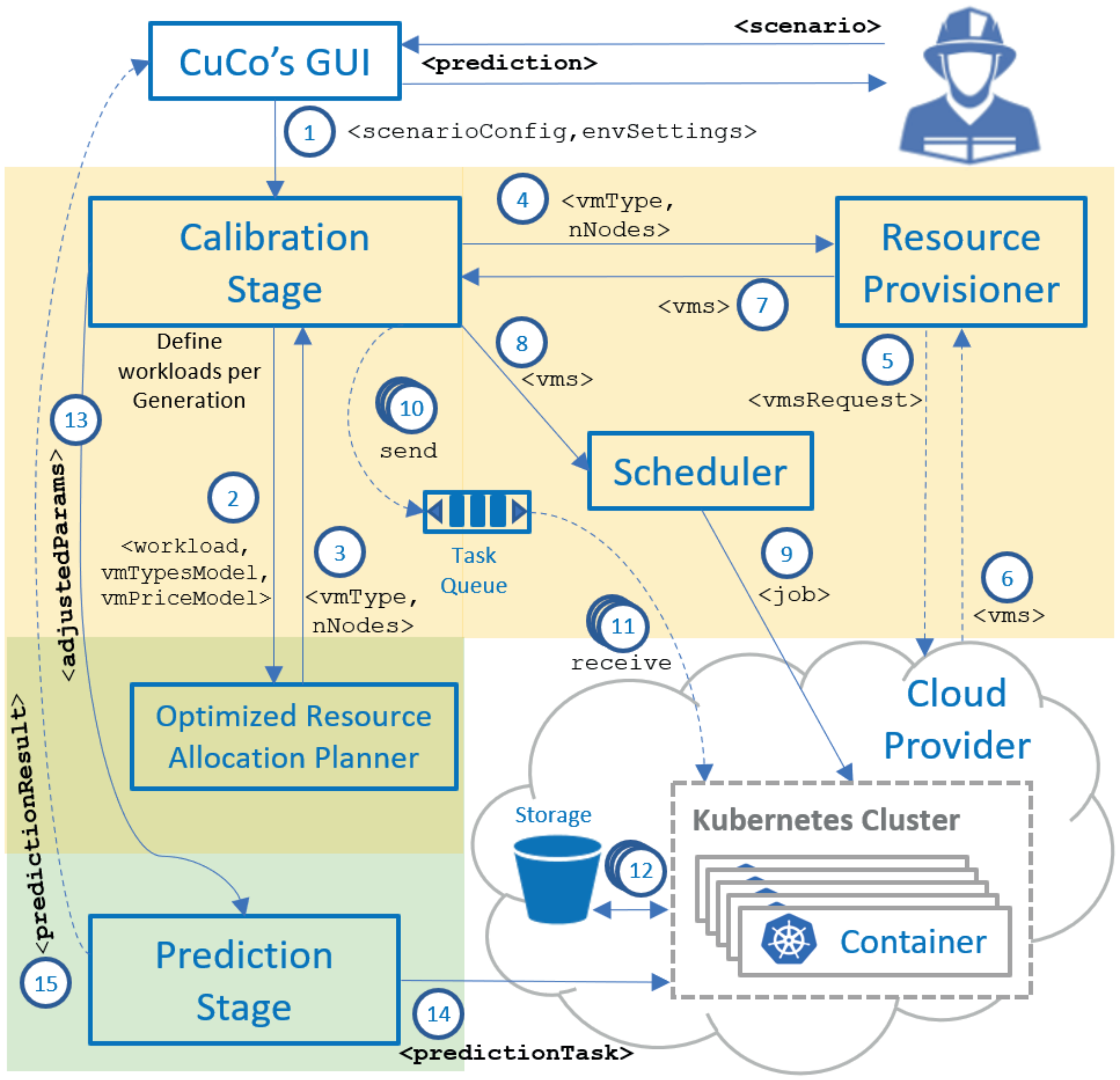


*Figure 21: Interaction flow between the main CuCo's components*

### 3.1.2. Optimized resource allocation planner

The *Optimized Resource Allocation Planner* is the component responsible for defining the resources where the calibration and the prediction will be executed. As a black box, it simply returns the list of available machines. But, to its fullest extent, it is a domain-specific module, crafted to statistically model the forest fire core simulator based on the specified computational resources properties, and determine the user preferences through strict deadlines or utility functions. Altogether, this information is used to acquire the computing resources that best fit the user needs. Its three sub-components, as illustrated in Figure 20, are the following.

**Runtime and Memory Consumption Models**: to estimate the execution time of a fire spread simulation, it is necessary to perform a large set of executions of the underlying simulator on the target architecture and then analyze its behavior from the obtained results.

**User's Utility Model**: to capture user's satisfaction over prediction time and monetary cost of the overall fire spread prediction. Utility functions are commonly used to communicate the value of work and other quality of service aspects such as its timely completion [66]. A utility function commonly used is the simplified version of the *Cobb-Douglas* production function.

**Price / Resource Model**: we consider an *infrastructure-as-a-service* (*IaaS*) cloud system, in which data centers deliver on-demand storage and compute capacities over the Internet. These computational resources are provided in the form of an abstract unit of compute and storage called Virtual Machine (VM for short), object storage, or remote file systems volumes. In a cloud, VMs are offered in several types, each of which has distinct characteristics such as different numbers of CPUs, amount of memory, and network bandwidth capacity.

We have chosen Amazon Web Services (AWS) [67] as the public cloud provider. AWS is a comprehensive and universally adopted cloud platform, offering fully featured services from data centers globally. As computing resources are treated as commodity in the cloud era, any major provider that offer computing and object storage could be used.

AWS *EC2* characteristics, including their pay-as-you-go prices, are then used to characterize the *Price/Resource Model*, against which we build the *Runtime and Memory Consumption Models*. Once the models are built, they can be actioned upon the utility functions to define the list of machines that will be provisioned to run the prediction engine.

### *AWS resource and cost models*

Assessing the requirements of the forest fire prediction model and selecting the appropriate instance family is the starting point for the application performance testing. It is crucial to identify how the model needs compare to different instance families (e.g., if it is compute-bound, memory-bound, network-bound, etc.), and to size the workload to identify the appropriate instance size.

**Resource Model**: Amazon Web Services (AWS) *Elastic Compute Cloud* (*EC2*) provides a variety of instance types, each one providing different combinations of CPU, memory, disk, and networking. As of May 2023, there are six families optimized for several types of applications and, for each one, ten different instance types.

Table 1 presents the families, the number of instances available, the number of CPUs, and the range of on-demand cost per hour for each one of them.

| Family | Recommendation | # | vCPU | Cost per Hour |
|---|---|---|---|---|
| **General Purpose** | Variety of diverse workloads. Balance of compute, memory, and networking resources. | 114 | 1 to 192 | 0.005 to 9.677 |
| **Compute Optimized** | Compute bound applications that benefit from high performance processors. | 85 | 1 to 192 | 0.042 to 9.158 |
| **Memory Optimized** | Workloads that process large data sets in memory. | 96 | 1 to 128 | 0.056 to 34.68 |
| **Accelerated Computing** | Use hardware accelerators, or co-processors, to perform functions more efficiently than is possible in software running on CPUs. | 14 | 4 to 96 | 0.274 to 9.389 |
| **Storage Optimized** | Workloads that require high, sequential read and write access to very large data sets on local storage. | 27 | 2 to 128 | 0.172 to 12.109 |
| **HPC Optimized** | Built to offer the best price performance for running *HPC* workloads at scale. | - | - | - |

*Table 1: AWS EC2 Families of Instances.*

**Cost Models**: as with everything, there are costs associated with the cloud usage. With the many economic benefits of the cloud, this may often seem negligible for simple workloads. For applications with highly parallel workloads, the cost may easily surpass a thousand of dollars in a couple of hours if the instance provision is not made in a diligent way. Table 2 presents the different cost models used to pay for Amazon *EC2* instances. Although the *Saving Plans* might be an interesting model to high-demanding fire analysts, we focus on the *On-Demand* and *Spot Instances* models to perform the characterizations.

| Model | Description |
|---|---|
| **On-Demand** | pay for compute capacity by the hour or the second depending on which instances are run. No longer-term commitments or upfront payments are needed. |
| **Spot Instances** | request spare *EC2* computing capacity for up to 90% off the On-Demand price. |
| **Savings Plans** | offers low prices on *EC2* usage, in exchange for a commitment to a consistent amount of usage (measured in $/hour) for a one- or three-year term. |
| **Dedicated Host** | a physical *EC2* dedicated server. |

*Table 2: AWS EC2 Cost Models.*

## *Characterizing runtime and memory consumption*

To obtain the missing knowledge about the execution time of each individual beforehand, the fire model is executed for a certain area with a representative number of individuals (different input data sets). Its parameters, together with the runtime and memory consumption of each execution is then recorded. These steps are made offline, which means that they must be carried out before the fire occurrence.

As forest fires show seasonal behavior and an occurrence pattern that affects the same zones every year, several indexes have been developed to determine the fire risk and fire spread potential. Besides past occurrences, these indexes consider the underlying vegetation and the meteorological conditions on a given area. The indexes most used are the *Fire Weather Index* (*FWI*) [68] and the Haines Index [69]. *FWI* is a meteorologically based index used worldwide to estimate fire danger that accounts for the effects of fuel moisture and wind on fire behavior and spread. Haines uses the meteorological conditions as an indicator of the potential risk of wildland fires, considering the stability and moisture content of the lower atmosphere. In this way, fire analysts can run the executions and prepare these models in the moment prior to the fire occurrence.

Our methodology is comprised of three steps: **1)** prepare the training database; **2)** trace the information regarding runtime and memory consumption; **3)** build the regression models.

**Training database**: Currently, we work with training databases composed of 10,000 up to almost 40,000 different scenarios. Each one represents a random individual, an input configuration for the target scenario. For the use case presented in this study, the distribution of each input parameter corresponds to the ones specified in Table 3. The vegetation models correspond to the 13 standard Northern Forest Fire Laboratory (NFFL) fuel models [46].

| Input | Distribution | μ; σ | Min; Max |
|---:|:---:|:---:|:---:|
| **Vegetation model** | Uniform | - | 1;13 |
| **Wind speed** | Normal | 70; 10.0 | - |
| **Wind direction** | Normal | 45; 5.0 | - |
| **Dead fuel moisture** | Uniform | - | 2;15 |
| **Live fuel moisture** | Uniform | - | 50; 100 |
| **Adjustment factor** | Uniform | - | 0.0; 2.0 |

*Table 3: Input parameters distributions.*

**Determination of execution time and memory consumption**: For each execution we trace the *Runtime* and the maximum *Resident Set Size* (the portion of memory occupied by a process held in main memory) metrics. Figure 22 depicts the multivariate relation between runtime and the most important individual parameters. As it can be seen, after calculating the *Pearson Coefficient*, we conclude that wind speed and humidity are the two parameters that most impact on the runtime. The scattering plot confirms visually such importance. Similar relations happen considering memory consumption.

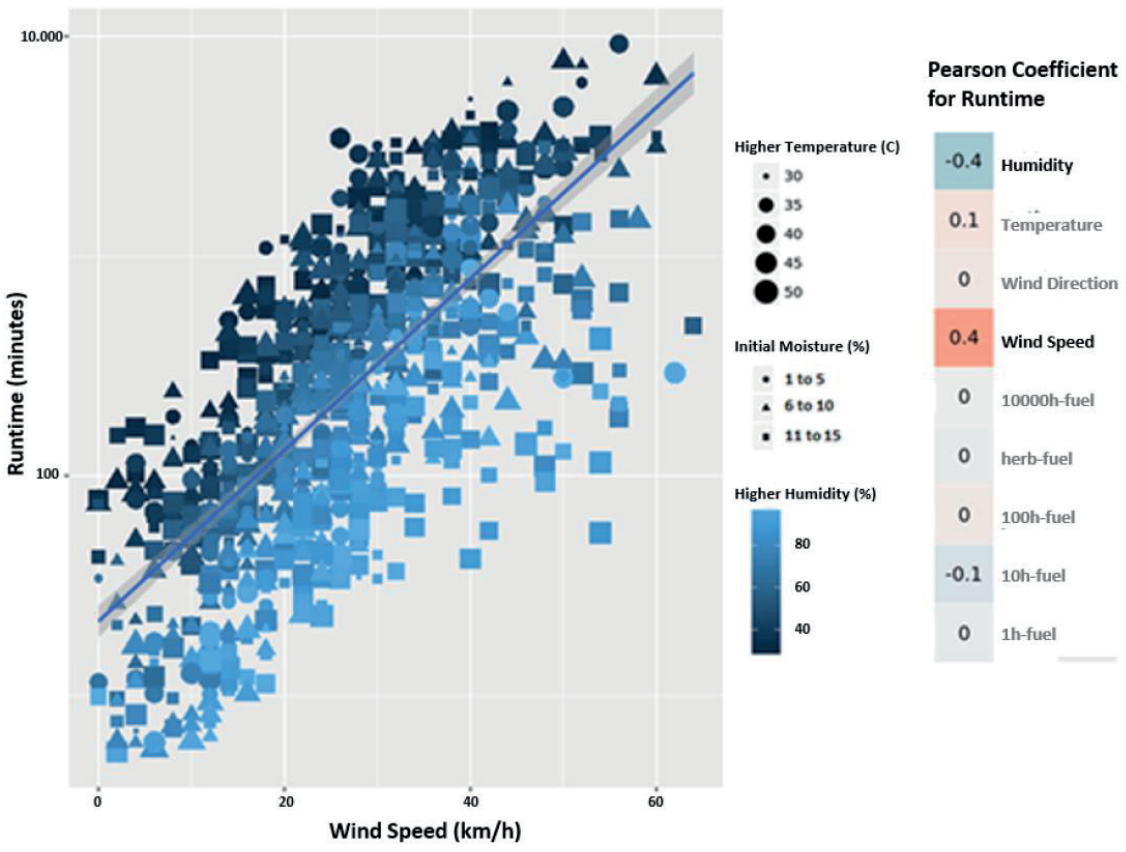


*Figure 22: Multivariate analysis of FARSITE's runtime.*

**Building the regression models**: We evaluated 5 multiple regression techniques to build a cross-validated regression model representing the relation between the parameters and the response metrics. The models are: *Multiple Linear Regression*, *Principal Component Regression*, *Partial Least Square*, *Regularized Regression Elastic Net* and *Multivariate Adaptive Regression Splines (MARS)*. From the built models we choose the best one based on the standard deviation of the residuals measure, i.e., the prediction errors, also called *Root Mean Square Error* (*RMSE*). Table 4 shows a summary of the prediction performance for each model.

| Model | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
|---|---|---|---|---|---|---|
| **Multiple Linear Regression** | 1022.458 | 1267.519 | 2215.918 | 2591.679 | 3984.889 | 4889.377 |
| **Principal Component Regression (PCR)** | 989.443 | 1263.651 | 2234.553 | 2590.982 | 3992.400 | 4867.281 |
| **Partial Least Square (PLS)** | 1018.913 | 1266.828 | 2216.746 | 2590.798 | 3984.444 | 4887.721 |
| **Regularized Regression Elastic Net** | 760.384 | 1030.713 | 2202.121 | 2533.858 | 4049.466 | 4978.295 |
| **Multivariate Adaptive Regression Splines (MARS)** | 746.422 | 1112.445 | **1670.995** | **2249.241** | 3430.974 | 4624.153 |

*Table 4: Standard deviation of the prediction errors.*

Based on the prediction performance, we have chosen MARS as the best model to characterize FARSITE runtime based on input parameters. Considering median and mean columns, we can see that it shows results with the smaller errors. MARS is a form of non-parametric regression analysis technique, and it is an extension of linear models that automatically models non-linearities and interactions between predictor variables [70]. Equivalent results apply to the memory consumption analysis. Figure 23 shows the graphical representation of the two built models. From the two graphics we can observe that there is a region where runtime and memory consumption grow significantly. Considering an informed two-stage prediction scheduler, individuals located in such region should be allocated in the most efficient available resources to decrease the overall calibration and the final fire spread prediction time.

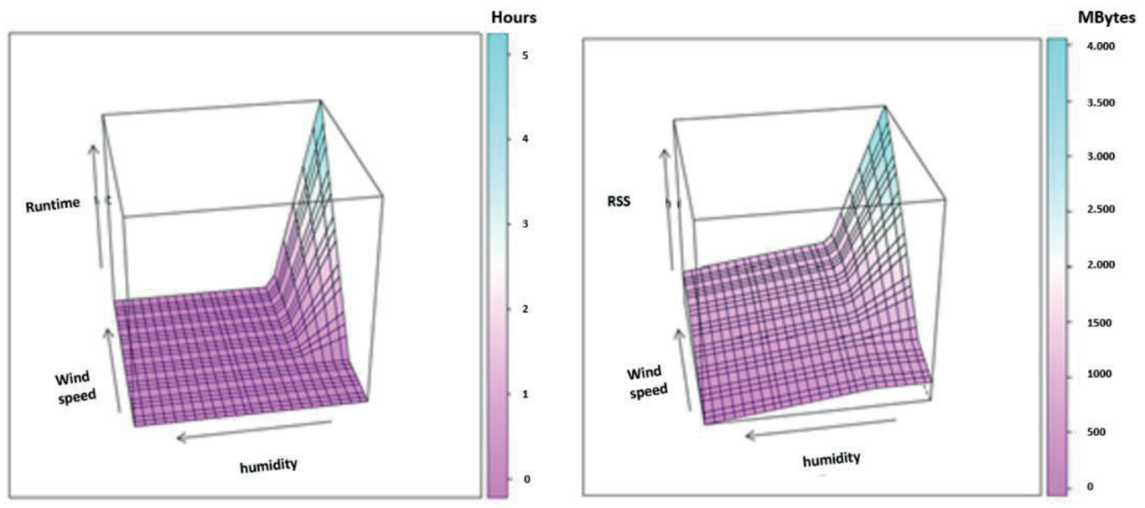


*Figure 23: MARS' models for FARSITE runtime and memory consumption.*

| Model | #vCPU | Memory (GiB | Cost per Hour |
|---|---|---|---|
| **c5.large** | 2 | 4 | $ 0.085 |
| **c5.xlarge** | 4 | 8 | $ 0.17 |
| **c5.2xlarge** | 8 | 16 | $ 0.34 |
| **c5.4xlarge** | 16 | 32 | $ 0.68 |
| **c5.9xlarge** | 36 | 72 | $ 1.53 |
| **c5.18xlarge** | 72 | 144 | $ 3.06 |

*Table 5: AWS EC2 Compute Instances Family (May 2023)*

Once this methodology is followed, only a last step remains: the application of the resulting model with the scenario describing the ongoing fire (to assess in advance the execution time its simulation will produce). This action supposes a negligible cost, in terms of time overhead (on the order of a few seconds). Accurate *FARSITE* runtime and memory consumption models are used to predict performance of actual *GA* individuals in the specified cloud resources and then determine the resulting monetary cost.

## *The cost-performance trade-off*

We consider the FARSITE characterization described in last subsections to define the most cost-effective AWS *EC2* instance types to be used as the underlying virtual infrastructure. As a result, the *Compute Optimized* family listed in Table 5 has been chosen. *The Optimized Resource Allocation Planner* is fed with these prices and resource characteristics and, based on the *Core Runtime and Memory Consumption Models* (see Figure 20), generates as output the average *cost per individual per calibration time*. In Figure 24 we can see the result for configuration spanning a potential cluster with 1, 2, 4, 8, 16, and 32 nodes. Estimations are grouped per potential cluster size, as instances are immediately terminated when there is no workload to be processed.

The last step is to decide which configuration will be used based on the user's preference. We model the user's preference as a utility function, a mathematical function that ranks alternatives according to their utility to an individual. The Cobb-Douglas production function $u(x_1, x_2) = x_1^{b_1} x_2^{b_2}$, is a utility function commonly used, where $b_1$ and $b_2$ are positive numbers describing the preferences of the consumer. In its simplified version, applying it for $b_1 = \frac{1}{2}$ and $b_2 = \frac{1}{2}$, i.e., the user is equally concerned with price and runtime, the resulting ranking of preference is shown in Table 6. There are 6×6 = 36 configurations in total (the number of instance types × the number of options for cluster size). Considering the results, the configuration that gives more utility to user preferences is the one with 4 nodes of type c5.2xlarge. For different parameters of the utility function, a new ranking is provided, but it is up to the user, i.e., the wildfire analyst, to inform his or her preferences. The Resource Provisioner module can provision the corresponding virtual resources and then trigger Kubernetes to perform the scheduling task.

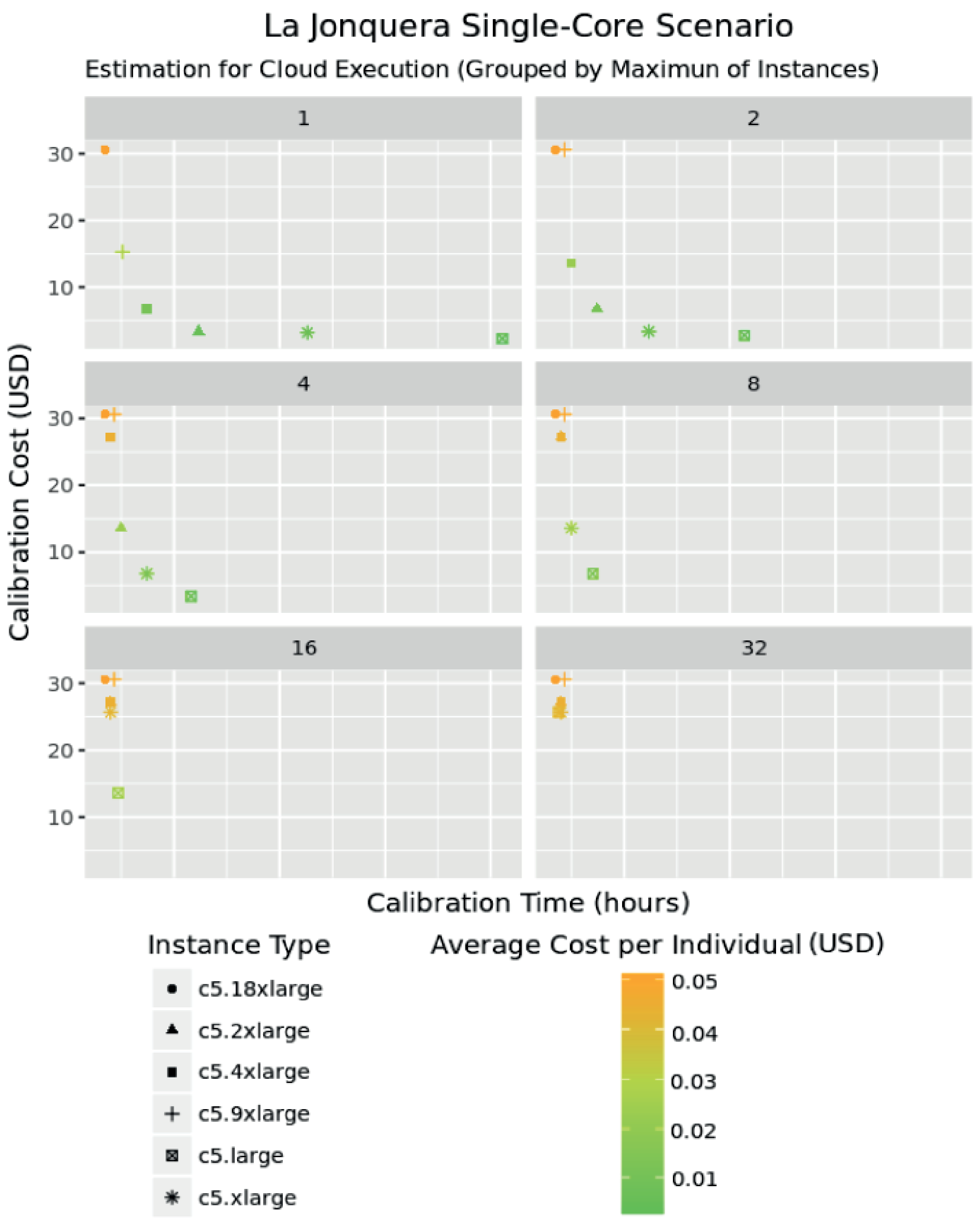


*Figure 24: Cloud cost prediction for the calibration phase.*

| Rank | Utility | #Nodes | Model |
|---|---|---|---|
| **1** | 0.5529 | 4 | c5.2xlarge |
| **2** | 0.5037 | 2 | c5.xlarge |
| **3** | 0.4976 | 1 | c5.xlarge |
| **4** | 0.4593 | 16 | c5.large |
| **5** | 0.3828 | 2 | c5.2xlarge |
| **…** | … | … | … |
| **35** | 0.0847 | 2 | c5.large |
| **36** | 0.0603 | 1 | c5.large |

*Table 6: Ranking of preference to run the calibration.*

The importance of an accurate FARSITE runtime and memory consumption model is twofold: a) be used as a workload generator for large scale performance analysis of the two-stage prediction framework in a simulated cloud environment; b) be used as input for an optimized cost and runtime estimator for a cloud based urgent computing solution based on the two-stage prediction framework, as illustrated in Figure 20.

## 3.2. Enhancements in the two-stage prediction model

Despite the use of cloud-based computing resources, the calibration stage is inherently time-consuming as the genetic algorithm that guides it needs to wait the end of the current generation before starting the next one. Being able to decrease the overall calibration depends not only on fast and vastly parallel computing resources, but also in clever strategies to deal with the sequentially nature of the genetic algorithm evolution. We have proposed three strategies to enhance the two-stage prediction model: **1)** an early adaptive evaluation strategy, **2)** definition of strict deadlines per generation and **3)** proposal of a new fitness function. The following subsections details each one of these strategies.

### 3.2.1. Early adaptive-evaluation strategy

To overcome the slow time-to-result characteristic of the genetic algorithm, we proposed an adaptive evaluation technique based on a periodic monitoring of the fire spread prediction error ε estimated by the normalized symmetric difference, as defined in Equation 1, for each execution of the individuals. Figure 25 shows the steps done by the monitoring process to determine whether the individual being monitored should be early terminated or not. Figure 26 depicts two individuals being executed and monitored according to the monitor flow defined in Figure 25. The one described in Figure 26 (I) terminates normally whereas the other described in Figure 26 (II) is early terminated by the monitoring agent due to its unfitness based on its ongoing prediction error.

The reasoning behind the early termination is to avoid wasting computing time running individuals that are doomed to unfitness. If along its execution the monitor agent detects that the prediction deviates too much from the actual fire spread, it is considered safe to early terminate the individual.

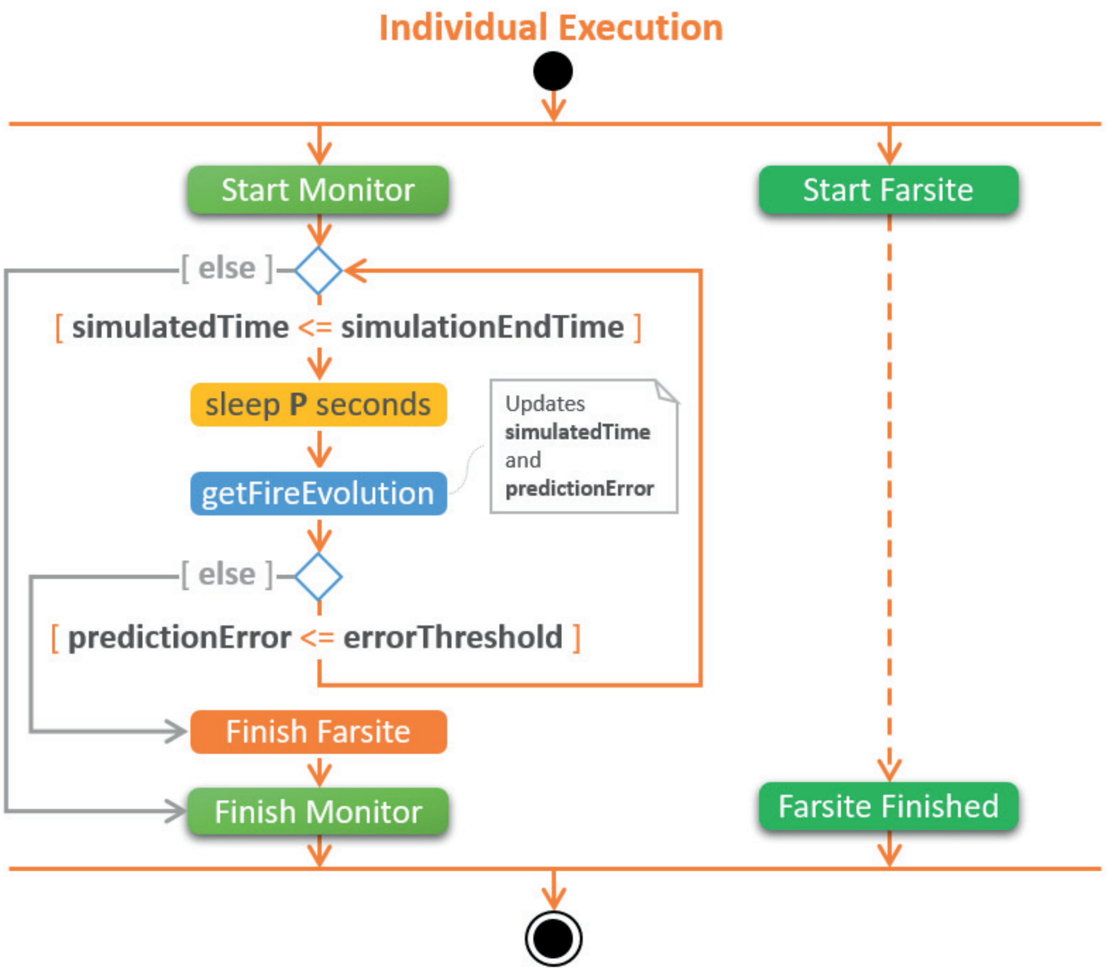


*Figure 25: Activity diagram for the FARSITE Monitoring agent.*

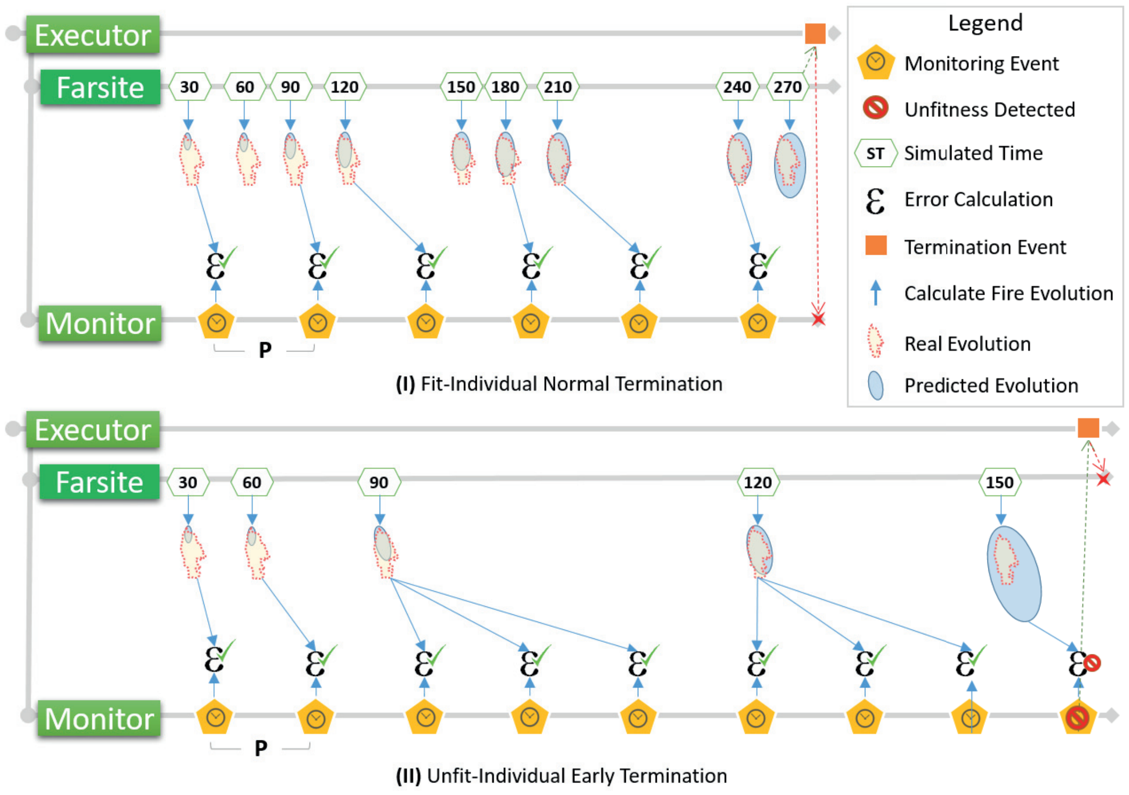


*Figure 26: Two distinct termination strategies for a FARSITE individual.*

Each monitoring agent is launched with a *prediction error threshold* **T**, i.e., a maximum tolerable error above which the monitoring agent must early terminate the individual, and a *monitoring period* **P**, usually defined in the order of dozens of seconds. For example, if we consider the canonical scenario of a fire prediction evolution described in Figure 27, together with their corresponding *normalized symmetric difference* (prediction error ε) calculated as described in Equation 1, we notice that as the prediction overspread beyond the real fire, the prediction error increases at a rate higher than the ones circumscribed to the real fire perimeter. Therefore, if at an earlier stage of the prediction a monitor configured with a threshold **T=2.0** detects a fire evolution like the one showed in Figure 27 (f), then the execution can be terminated as the individual is considered unfit to represent the real fire.

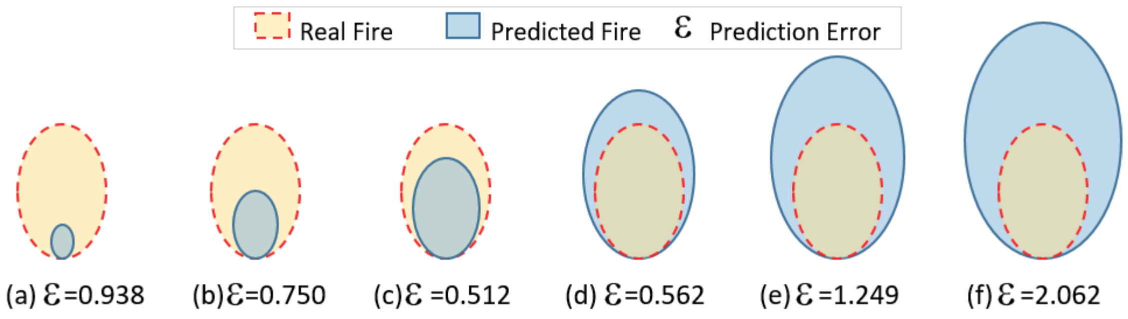


*Figure 27: Normalized Symmetric Difference calculated along fire spread prediction.*

So far, the strategy used to early terminate and then discard individuals was based solely on a deadline previously defined to avoid delaying an entire generation due to a few longer individuals. Or, even worse, previously filtering out those individuals whose execution time is estimated to be longer than a preset value. The problem regarding early termination is its impact on the population diversity, a crucial characteristic to the genetic algorithm's ability to continue fruitful exploration of the search space to find a solution.

Another difference from previous works is that, in this proposal, we use a *weighted version* of the normalized symmetric difference to consider early terminations. In Equation 3 we can see the formula for a weighted version of the normalized symmetric difference (*NSD*). In it, the ***PredictionTime*** is the total time expected to be simulated whereas ***SimulatedTime*** is the total time effectively simulated (until a normal or an early termination takes place).

$$fitness = \frac{PredictionTime}{SimulatedTime} \times NSD \qquad (3)$$

In Figure 26 (I), in the scenario of a normal termination, for a total simulated time equals to 270 minutes, we can see that the simulated and prediction time are equals, then the fitness function is equals to the normalized symmetric difference. On the other hand, in the early termination scenario showed in Figure 26 (II), the penalty is directly proportional to the simulated time left to a normal termination, i.e.,

$$fitness = \frac{270}{150} \times NSD = 1.8 \times NSD \qquad (4)$$

The idea is to be able to compare unfinished individuals results against the actual burned area, considering how much has been simulated until the moment of the early termination: the sooner the termination, the greater should be the penalty. Therefore, the main contribution of this strategy is to avoid wasting too much computing time running individuals that are indubitably unfit. Nevertheless, thanks to the *weighted normalized symmetric difference*, those individuals can still be evaluated and be unlikely selected to the following generations, ensuring more diversity to the evolution process.

Due to the fork-and-join characteristic of the evolution stream defined by a genetic algorithm, the response time of each generation and therefore for the entire evolution is limited by execution time of the slowest individual. Previous works [21] have identified that the execution time distribution of randomly generated individuals follows the characteristic of a power law, with a few small values dominating the distribution, complemented by a long tail representing the slowest individuals (see Figure 28). Unfortunately, like in all unusual extreme events modelled by power laws, those infrequent occurrences are responsible for the worst damage to the calibration phase capacity to provide shorter response time.

Then, to decrease the overall calibration time, it is crucial to be able to consistently decrease the random individual's execution time. The violin plots in Figure 29 compare the FARSITE execution time for a 1-hour-deadline-driven strategy and the adaptive-evaluation for the real "*La Jonquera*" forest fire scenario. Considering these two execution time distributions, it would be expected that the new strategy helps to calibrate the input data approximately three times faster than any strategy based solely on a deadline-driven approach set to 1 hour per generation. In next section we evaluate the adaptive strategy applied to this real fire scenario.

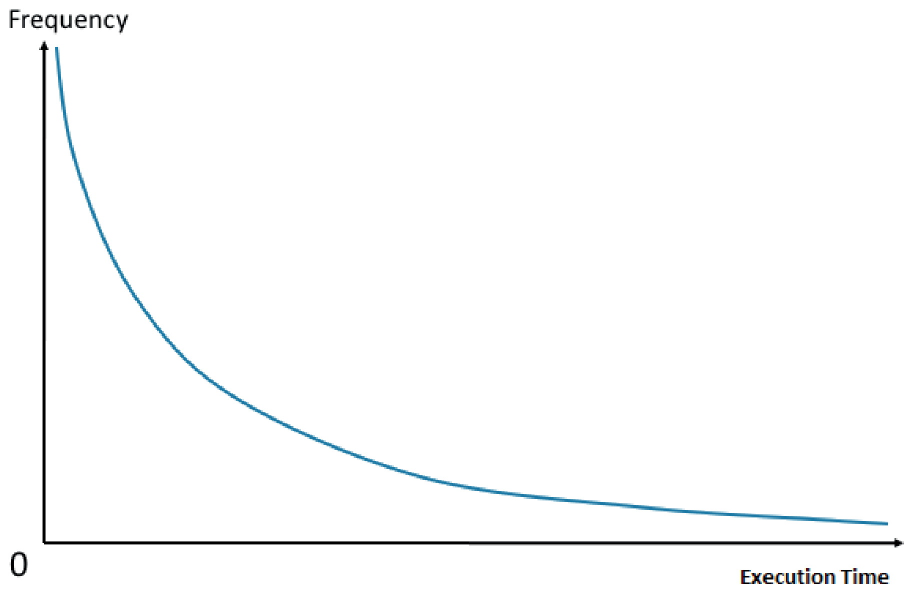


*Figure 28: Power Law's characteristic of long-running executions.*

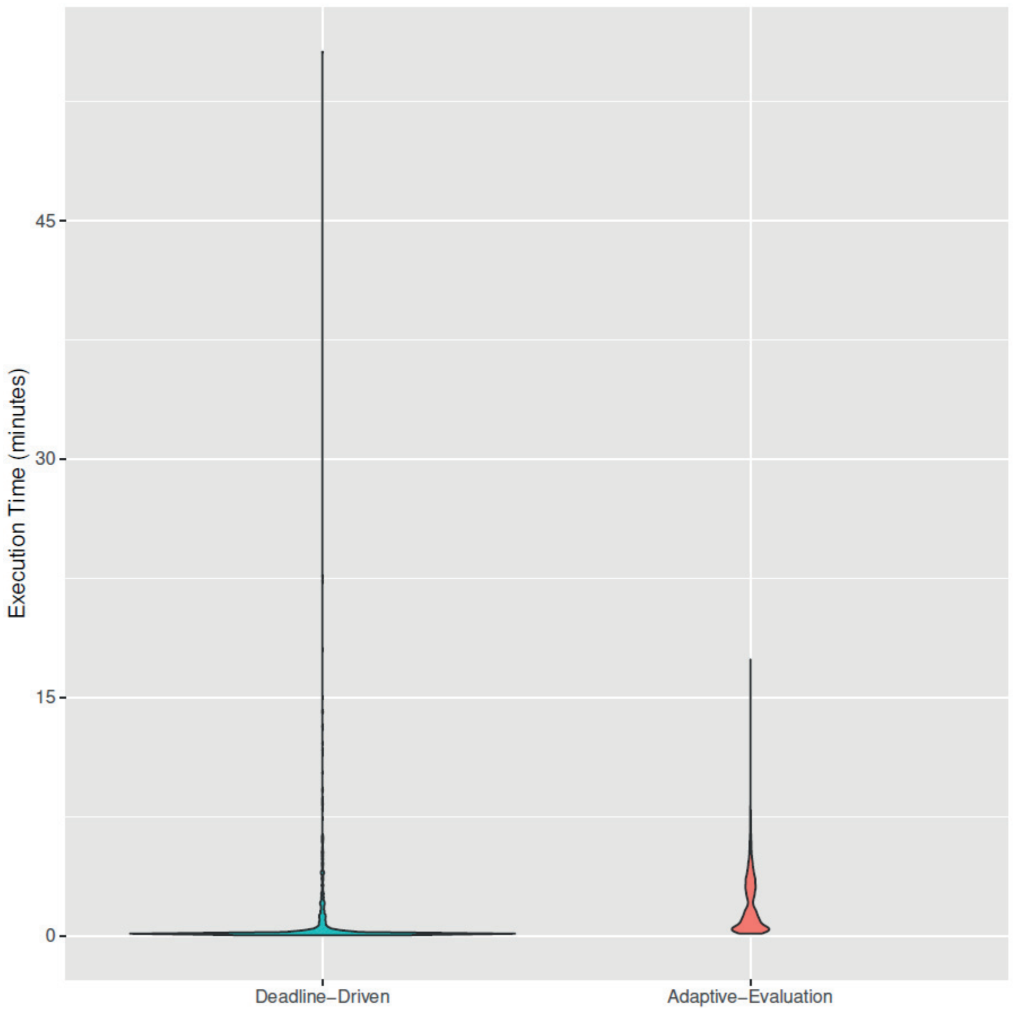


*Figure 29: Runtime distribution for different evaluation strategies.*

### 3.2.2. Definition of a strict-deadline policy per generation

The stream of generations resulting from applying the genetic algorithm in the calibration phase of the *two-stage prediction* scheme is essentially a *periodic fork-join model*. In such model, execution branches off in parallel at designated points in the program, to join at a subsequent point, resume the sequential execution, then repeat if necessary. The problem lies in the fact that an entire generation is as fast as its slowest individual. The execution time of each simulation varies drastically depending on the input parameters used, provoking a significant load imbalance.

Artés et al. [20] investigated the influence of deadlines for each generation. Three different strategies have been designed considering individuals that exceed the generation execution time limit. Due to its convoluted nature and the demand of excessive tweaking on the genetic algorithm philosophy, the strategies have not been embraced by the community.

The main disadvantage was that an individual must necessarily be finished before evaluation. Such limitation was overcome by the *Early Adaptive-Evaluation* strategy proposed in the current thesis. Building on Artés' findings and applying the *Early-Adaptive evaluation* to the fitness function calculation, we can now consider early terminated individuals even for strictly imposed deadlines. We can then take advantage of the early termination evaluation technique to consider all the elements, even the killed ones, keeping the variety in the population and the possibility to select the right elements to apply the genetic operations throughout the evolution.

In an attempt to decrease the overall calibration time, we decided to set more strict deadlines per generation than the 1-hour used by Artés. The deadlines range from 30 seconds to the default value of one hour per generation. In the chapter of experimental results, we evaluate the impact of these superimposed deadlines against the well-known *La Jonquera's* fire scenario.

## 3.2.3. A goodness of fit function to compare predicted and real fire spread

The work described in the earlier section aimed at decreasing the overall calibration time when using the two-stage forest fire prediction framework. To keep improving the framework, the next natural step is to advance in the prediction phase. The work described in this current session proposes a new metric to compare predicted and real fire spread with the intention of improving such prediction quality.

While dealing with metaheuristic based evolutionary algorithms (like in genetic algorithms), proper fitness function selection needs to be the more accurate possible as it is the responsible for conducting the algorithm evolution and allowing exploration of promising search space area. As presented in Equation (1), the framework currently uses a variation of the well-known *symmetric difference* between *Real* and *Predicted* fire spreads.

$$SymmetricDifference = \bigcup(Real, Pred) - \bigcap(Real, Pred) \tag{5}$$

The variation used as a fitness function is the normalized symmetric difference (*NSD*), as it normalizes the symmetric difference by the area of the *Real* fire spread.

$$NSD = \frac{\cup(Real, Pred) - \cap(Real, Pred)}{Real} \tag{6}$$

After the normalization, the symmetric difference metric is interpreted as the fire spread prediction error. Another way to interpret the symmetric difference is in terms of ***Hits***, ***Misses,*** and ***False Alarms*** (see Figure 19).

$$NSD = \frac{Misses + FalseAlarms}{Misses + Hits} \tag{7}$$

When applied to a real fire scenario, the normalized symmetric difference presents a drawback, in which *False Alarms* are too much penalized in comparison to the misses. Unfortunately, such characteristics is not desirable when comparing predicted and real fire spreads because wildfires in a real scenario does not spread freely as human forces fight hard to control them besides the presence of many barriers in the terrain, like highways, rivers, and other lack of vegetation acts as firebreaks.

To see this undesired behavior of normalized symmetric difference, Figure 30 shows four scenarios related to the *La Jonquera's* fire:

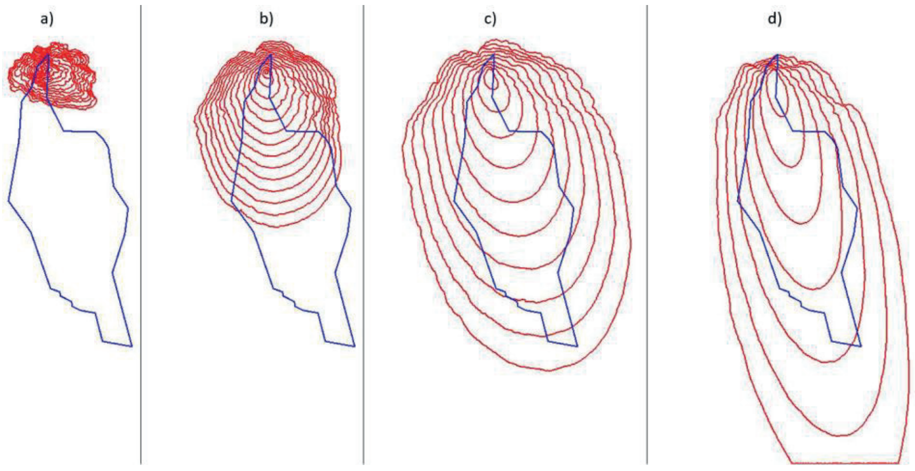


*Figure 30: Four representative scenarios of a fire spread evolution in La Jonquera's fire.*

If we apply the normalized symmetric difference for these four scenarios and rank their fitness, we end up with the following values in Table 7:

| Rank | NSD | Scenario |
|---|---|---|
| **1º** | 1.0448 | b) |
| **2º** | 1.1654 | a) |
| **3º** | 2.0775 | c) |
| **4º** | 2.3804 | d) |

*Table 7: Ranking of representative fire scenarios regarding the NSD.*

Surprisingly, when considering normalized symmetric difference, the scenario b) is the top ranked and yet more unexpected the scenario a) is the second most suitable. Scenario c), one that encompass the entire real fire perimeter, comes as the third followed by scenario d). If we analyze the normalized symmetric difference closely, we can see that *False Alarms* are the main reason for a greater error. When dealing with a prediction that contains many misses, these are compensated by their occurrence in the denominator as well. Such characteristic occurs in the scenarios a) and b). Scenarios c) and d) manage to get 100% of *Hits*, but in expense of having many misses considering the shape of the real fire spread perimeter.

When contrasted with real wildfire expansion, if the genetic algorithm keeps selecting individuals like a) instead of individuals like c) we would expect a slow convergence or even a misfit after the calibration phase. One obvious drawback of the normalized symmetric difference is noted if we consider a hypothetical prediction scenario where the predicted burned area is equals to zero (`Predicted=0`). For this case, hits are also equals to zero (`Hits=0`) and false alarms are equals to zero as well (`Misses=0`). Misses are equals to the real fire spread are and, as a result:

$$NSD = \frac{Misses + FalseAlarms}{Misses + Hits} = \frac{Real + 0}{Real + 0} = \frac{Real}{Real} = 1 \qquad (8)$$

Considering this result, the fitness for this hypothetical scenario would surpass the four ones presented in Figure 30. Considering that for the *La Jonquera*'s fire the best fit individuals usually have fitness evaluation placed between 0.8 and 0.9, **the occurrence of misfit individuals throughout the evolution stream might end up undermining the entire evolution process (calibration phase) and consequently the final prediction.**

To cope with this limitation, we propose the use of a new fitness function based on a Goodness of Fit (*GoF*) approach. The metric is calculated as follows:

$$GoF = \frac{Hits}{Hits + Misses} \times \frac{Hits}{Hits + FalseAlarms} \qquad (9)$$

Originally, the *GoF* is a method that unambiguously shows the degree of spatial concordance between two or more categorical maps [71]. While adapting its use to the two-stage prediction framework, the intention is to evaluate the degree of fit among the predicted fire spread and real fire spread. GoF is resolution-independent, and it is suitable for quantifying the spatial uncertainty of map features, especially polygon shapes.

Figure 31 generalizes the idea of a goodness of fit function in terms of polygons (map) comparison. The GoF metric is based on two values: (1) the proportion of the intersecting area to the total area from `Map 2`, and (2) the proportion of the intersecting area to the total area of the `Map 1`. The first term gives the proportion of *insideness* of the `Map 2` with the `Map 1`. The second term weights this insideness (or degree of fit) by the fractional share of the area from `Map 1` that is intersected. The GoF units are area squared over area squared, so that this GoF measure is unitless and can be expressed as a percentage. Notice that there is a direct relation between the GoF for map comparison and the one proposed for fire spread prediction, where `C`=`Hits`, `B`=`Misses` and `A`=`False Alarms`.

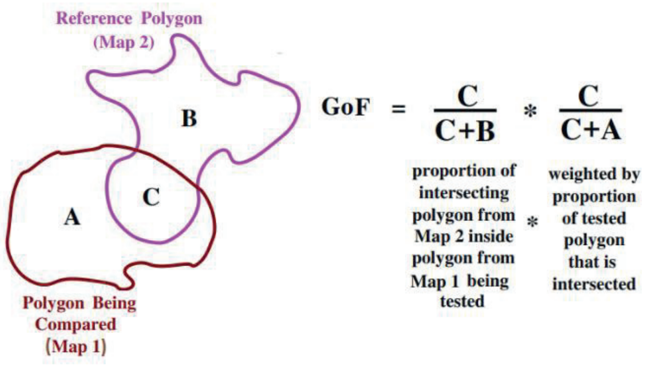

Reference Polygon (Map 2)
B
C
A
Polygon Being Compared (Map 1)
GoF = C / C+B * C / C+A
proportion of intersecting polygon from Map 2 inside polygon from Map 1 being tested
*
weighted by proportion of tested polygon that is intersected

*Figure 31: Goodness-of-Fit function for generic map comparison.*

Figure 32 depicts the **GoF** measure as the degree of spatial overlap (*insideness*) of two polygons increases. When spatial overlap is maximized, the goodness of fit is high, and an identity between the map categories is suggested. On the other hand, when there is little *insideness*, goodness of fit is low, and identity is unlikely. An ideal GoF model will be especially responsive to incremental increases at high overlap, since this extra sensitivity will discriminate excellent fit from good fit, while distinguishing both from poor fits. This latter characteristic is vital when selecting a **GoF** as a fitness function for an evolutionary algorithm.

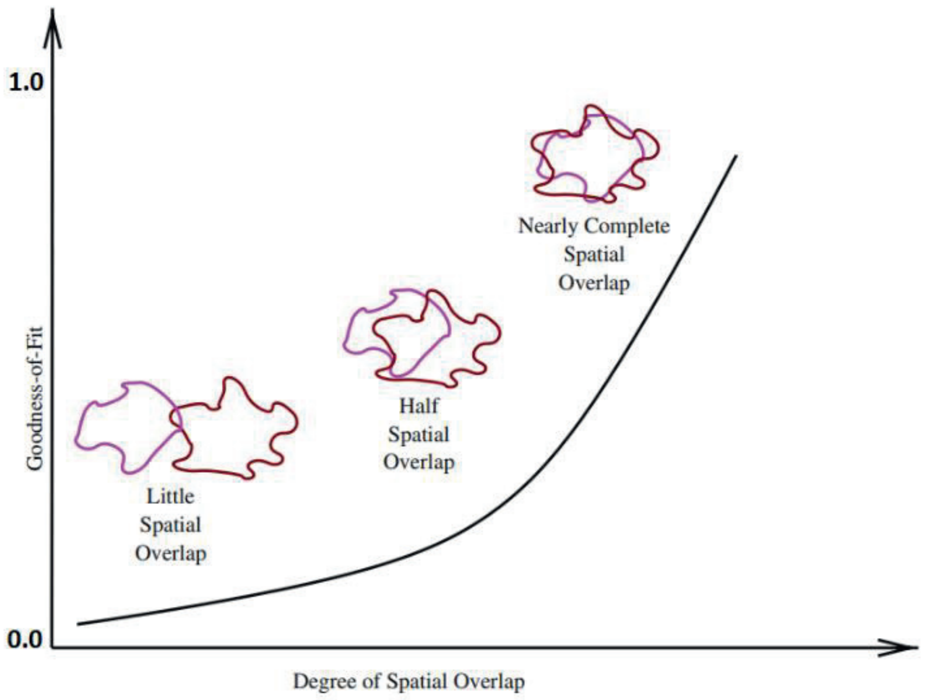


*Figure 32: Goodness-of-Fit measure as the degree of spatial overlap of two polygons increases.*

In opposition to the normalized symmetric difference (**NSD**), the greatest the metric the better the fitness. The following picture summarizes the behavior for both metrics when applied to a canonical scenario. The canonical scenario is based on a hypothetical fire ignited in a single point and that spreads evenly in a succession of concentrical circles. Each circle represents the fire expansion and the radius for the real fire spread is equals to 1 (r=1.0). The horizontal axis represents the radius for the predictions.

After analyzing the graphic, we can notice that ***GoF*** metric succeeds when the objective is to marginalize the results with much missing (values where prediction radius is less than 0.5) while does not penalize too much the results with false alarms (r > 1.0). Another positive characteristic of the GoF is its bounded nature, comprises in the range [0.0 ≤ GoF ≤ 1.0], giving more accurate results.

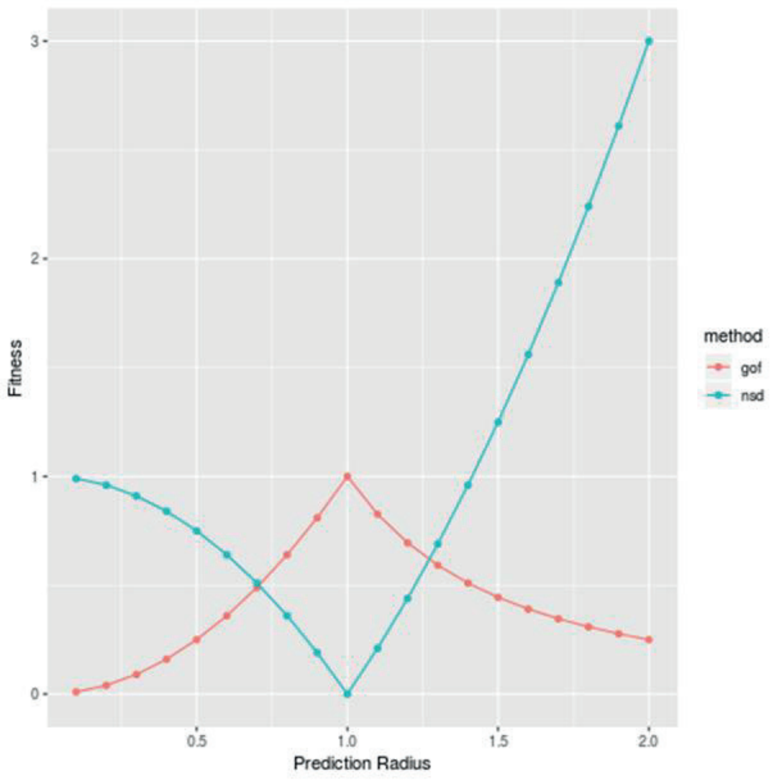


*Figure 33: Comparison between Goodness of Fit and Normalized Symmetric Difference functions.*

Applying the *Goodness of Fit* and the normalized symmetric difference to the real fire spread scenario *Arkadia*, we can see in Figure 34 a representative result showing the **NSD** and **GoF** metric comparison for two fire spread predictions, one with ***less false alarms*** (but with ***more misses***) and the second with ***more false alarms*** (but with ***less misses***). The two executions come from the same initial population.

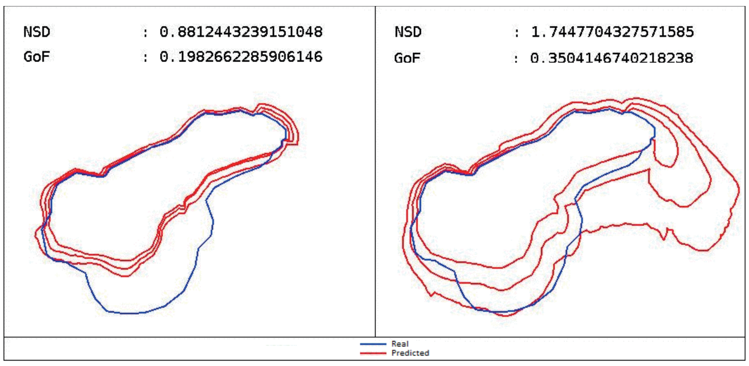


*Figure 34: Goodness of Fit and Normalized Symmetric Difference for Arkadia's scenario.*

The current two-stage framework has been adapted to use the new fitness function and after a preliminary set of experiments showed robust convergency in the calibration phase. Next step should compare the prediction quality of the new fitness function and evaluate the hypothesis that ***better prediction results can be obtained with a better fitness function***.

# Chapter 4: Experimental Results

We have worked with three historical fires that represent different and increasing levels of complexity in relation to the simulation challenges. *Arkadia* is a *mild* scenario, *La Jonquera* is *moderate*, while *Camp Fire* is an *extreme* wildfire case. The enhancements to the two-stage prediction framework presented in this work have been validated against fire scenarios in the Mediterranean region at Spain (*La Jonquera*) and Greece (*Arkadia*).

Theses scenarios have been thoroughly explored, studied, and compared in previous works by our research team [20] [33] [34]. In this thesis, we have also set up an experimental study to validate the CuCo platform against the *Camp Fire*, a deadliest and destructive wildfire that occurred in California - USA in the year 2018. The following sections present each one of these scenarios, detailing the experimental results obtained from each o them.

## 4.1. Arkadia's Fire: a mild case

This fire took place in the Arkadia, a regional unit in the *Peloponnese* region, in Greece, during the summer season of 2011. Fortunately, there were no casualties, as the fire burned the forest zone. Figure 35 shows its location and the fire perimeters used in the experiments.

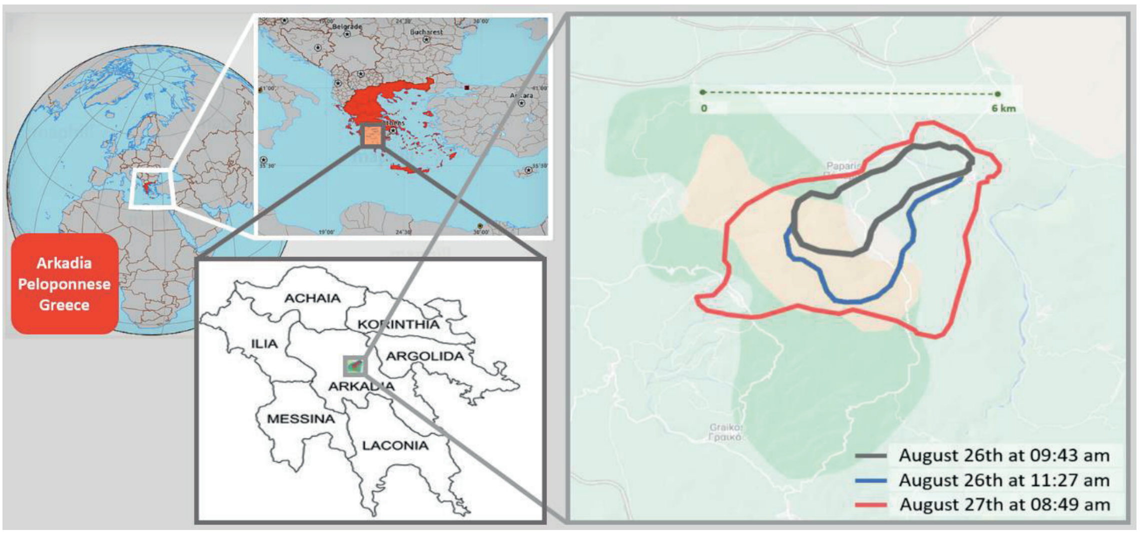


*Figure 35: Arkadia's Fire (Peloponnese - Greece), and its calibration perimeter.*

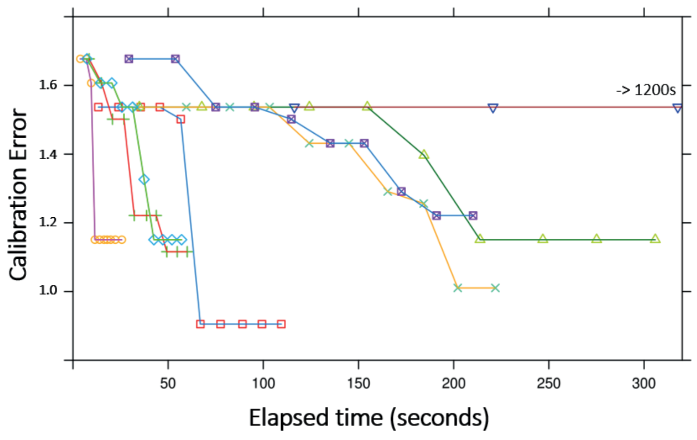
Arkadia's fire scenario was used to assess the new proposed strategies and to validate the earlier version of the cloud-based implementation of the two-stage prediction method. It is not a time-consuming scenario, what turns it into a perfect fit for proof-of-concept validations. To have a better understanding of the ***Two-Stage Prediction Framework***, we have conducted a set of parameters sweep experiments in some key parameters that impact the quality of prediction or the time to calculate it. The first intention was to assess the relative quality of prediction after adjusting parameters related to the computing infrastructure being used and the spatial resolution of FARSITE model simulation. We evaluated the performance of the framework while executing on a very small virtual cluster and on a larger one. We also compared it using conventional and high-end virtual instance types. In a FARSITE simulation, the spatial resolution factors, the distance resolution is the maximum projected spread distance from any perimeter point, whereas the perimeter resolution determines the maximum distance between points used to define the fire perimeter. Table 8 shows the parameters used in the configuration of this experiment.

| | Property | Value |
|---|---|---|
| **Arkadia's fire** | Area | 17.6 km² (1.760 ha) |
| **(Greece, August 2011)** | Perimeter | 18.6 km |
| **Landscape map** | Dimension | 15.6 km x 11.8 km |
| | Resolution | 100m x 100m |
| | File Size (MiB) | 0.188 |
| **Genetic algorithm** | Number of generations | 10 |
| | Population size | 128 |
| | Crossover probability | 0.1 to 0.9 |
| | Mutation probability | 0.1 to 0.9 |
| **Fire simulation** | Perimeter Resolution | 30m, 100m |
| | Distance Resolution | 30m, 100m |
| **Infrastructure** | Cluster Size | 2 and 8 |
| | Instance Type | c1.mediun, c1.xlarge |

*Table 8: Set up configuration for Arkadia's scenario.*

After selecting the best individual for each group of executions, we obtained the scenario displayed in Figure 36. As we can see, the judicious choice of a parameter configuration impacts considerably both the adjustment error (quality) and the time necessary to perform the prediction. From this initial experiment, as expected, we concluded that a better infrastructure and a higher spatial resolution do really improve the quality of the adjustment. We can also notice that small errors are obtained in exchange for a larger prediction time. This kind of trade-off is a key concern when dealing with real time events and must be taken carefully in consideration.

From this exploratory experiment, we extract important configuration values to be used in the other scenarios: crossover and mutation probabilities, and perimeter and distance resolutions.



| | | Configuration | Crossover Probability | Mutation Probability | Cluster Size | Instance Type | Perimeter Resolution (m) | Distance Resolution (m) | Time to Adjust (s) | Final Error |
|---|---|---|---|---|---|---|---|---|---|---|
| □ | — | 1 | 0.9 | 0.1 | 8 | xl | 30 | 30 | 60 | 0.910 |
| ○ | — | 2 | 0.9 | 0.2 | 8 | xl | 100 | 100 | 25 | 1.155 |
| △ | — | 3 | 0.9 | 0.4 | 8 | m | 30 | 30 | 305 | 1.157 |
| + | — | 4 | 0.2 | 0.2 | 8 | m | 100 | 100 | 65 | 1.102 |
| × | — | 5 | 0.1 | 0.1 | 2 | xl | 30 | 30 | 240 | 1.021 |
| ◇ | — | 6 | 0.7 | 0.3 | 2 | xl | 100 | 100 | 60 | 1.155 |
| ▽ | — | 7 | 0.5 | 0.3 | 2 | m | 30 | 30 | 1200 | 1.131 |
| ⊠ | — | 8 | 0.1 | 0.2 | 2 | m | 100 | 100 | 220 | 1.203 |

*Figure 36: Decreasing error when populations evolve.*

## 4.2. La Jonquera’s Fire: a moderate case

The Mediterranean area is one of the European regions most affected by forest fires during high-risk seasons. The selected case study corresponds to a region within the Mediterranean coast that is frequently affected by wildfires. We used a wildfire that occurred in La Jonquera (North-East of Catalonia, Spain – see Figure 37) in July 2012 that devastated over 14,000 ha (140 km$^2$) and caused the death of two people. This wildfire scenario has been thoroughly studied and compared in previous works [20] [33] [72].

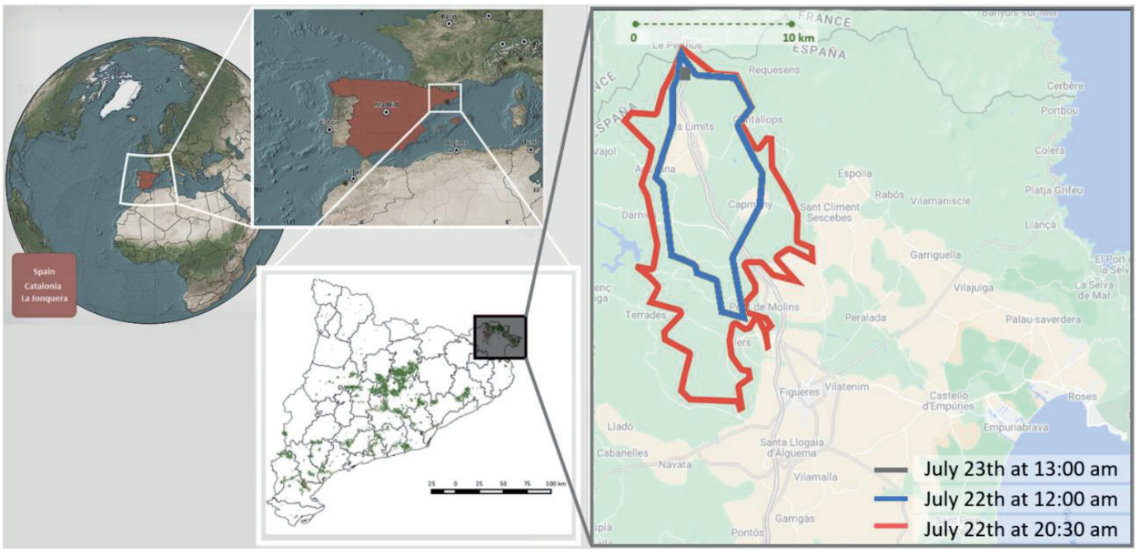


*Figure 37: La Jonquera’s Fire (Catalonia - Spain), and its calibration perimeter.*

The opportunity to compare the proposed strategies with data from these previous works was the main reason to choose this scenario to evaluate the enhancements in the two-stage prediction framework. In this section we will use this scenario to evaluate the *Early Adaptive-Evaluation* strategy and the *Strict Deadline* policy.

The period of time for the calibration was set to match exactly the observed evolution that goes from July 22, 2012, at 12:00 to July 22, 2012, at 20:30, as depicted in Figure 37. The genetic algorithm has been configured to evolve for 10 generations, each one with population size set-up to 100 individuals. Probabilities of crossover and mutation used in the executions are 0.9 and 0.1, respectively. Each one of the individuals is simulated, and the resulting simulated fire perimeter is compared to the actual fire spread using the *weighted normalized symmetric difference* between the real burned area and the predicted one, as described earlier. Table 9 summarizes the configuration set up.

| | Property | Value |
|---|---|---|
| **La Jonquera's fire** | Area | 140 km² (14.000 ha) |
| **(Spain - July 2012)** | Perimeter | 82.3 km |
| **Landscape map** | Dimension | 48.3 km x 33.7 km |
| | Resolution | 30m x 30m |
| | File Size (MiB) | 18.100 |
| **Genetic algorithm** | Number of generations | 10 |
| | Population size | 100 |
| | Crossover probability | 0.9 |
| | Mutation probability | 0.1 |
| **Fire simulation** | Perimeter Resolution | 30m |
| | Distance Resolution | 30m |
| **Infrastructure** | Cluster Size | 4 |
| | Instance Type | c5.2xlarge |

*Table 9: Set up configuration for La Jonquera's scenario.*

Regarding the configuration for the *early adaptive-evaluation strategy*, each monitoring agent is launched with a prediction error threshold set to 1.5. This implies that the maximum ongoing tolerable error above which the monitoring agent must early terminate the individual is 1.5. In other words: when the predicted area is one and a half times the size of the one defined by the real fire perimeter, the corresponding simulation will be terminated.

To have maximum parallelism, we make sure that there are enough pods/cores available to run all individuals in parallel and then avoid queuing of FARSITE tasks. In the present case, we are not concerned about what instances has been chosen to host the pods, as long as we can obtain the maximum level of parallelism.

### 4.2.1. Early adaptive-evaluation and deadline-driven strategies comparison

Figure 38 shows an overview of a single calibration execution using the adaptive-evaluation strategy and the deadline-driven approach set to 1 hour per generation for a population of one hundred individuals where each one of the individuals have been executed sequentially in a single core.

Confronting the two figures, we can notice that the overall calibration time is reduced from above 8 hours to 1 hour on average, representing an improvement of impressive **85%**. When compared to the current best data-driven Hybrid MPI-OpenMP Parallel approach, the *Time Aware Core allocation* (TAC) [20] [33], the calibration time is reduced from above 3 hours to the same 1 hour on average, resulting in a **60%** time-to-result improvement. The *TAC* approach, which applies classification techniques to detect in advance those individuals that will last longer and allocate more cores to them, was implemented with strict time restrictions: a time lapse of 4 hours was considered to carry out the calibration stage and the final prediction. This time window was split into two intervals: 3 hours for the calibration, and 1 hour for the simulation that will result in the final prediction.

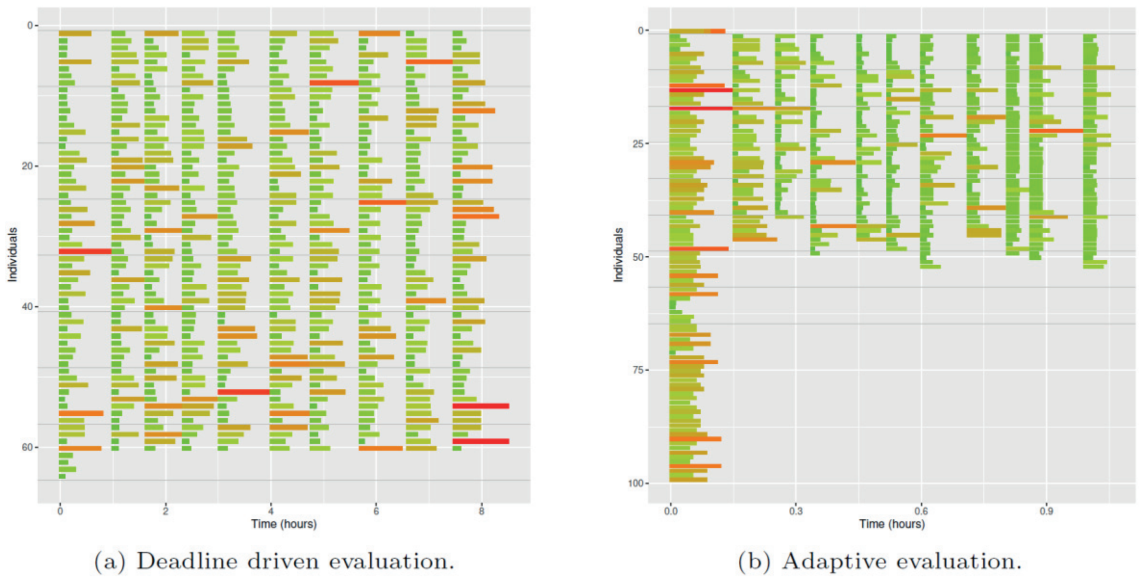


*Figure 38: Traces comparing Deadline-driven and Adaptive-evaluation strategies.*

As a performance gain summary, Figure 39 shows the overall comparison between the three calibration strategies. In relation to the adaptive evaluation metric, the error bars represent the minimum and the maximum values obtained for 20 different calibrations whereas the intermediate point is the average value. For the other scenarios, the bar height represents the average value as reported by Artés et al [20]. As expected, the adaptive evaluation shows a better performance when compared to the other two strategies, representing a major improvement even when compared to the *TAC* strategy (more sophisticated).

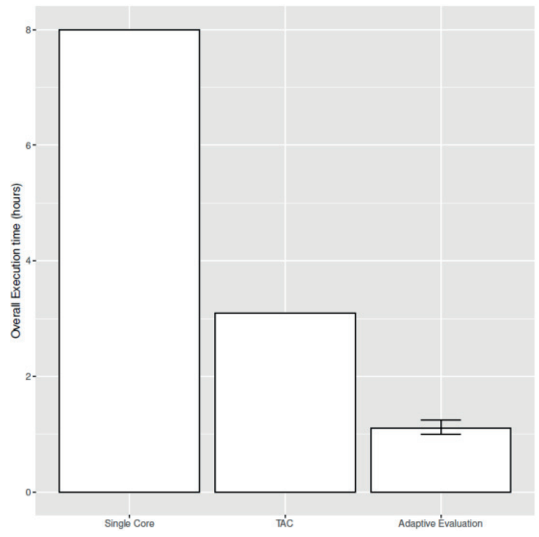


*Figure 39: Comparison between different evaluation strategies.*

Runtime values from Artés work have been normalized according to the CPU factor of the *AWS Computed Optimized instances*. For example, as of May 2023, the bare metal used by Amazon to host the instances of the compute optimized instances is the *Intel Xeon Platinum series 8000*, while Artès used two *PowerEdge C6145* nodes, each node with *4 AMD OpteronTM6376 of 16 cores with 128GB of DDR3 1600 MHz*.

### 4.2.2. Strict-deadline policy per generation

We define a set of deadline values to investigate its impact on the overall calibration quality and to identify at which point quality starts to deteriorate. The values used were 30, 60, 120, 180, 240, 300, 600, 900, 1200, 1800 and 3600 seconds. In the related works, for the same real fire scenario, the reference deadline value used was 3600 seconds (1 hour). For that default value, in the worst-case scenario, the entire calibration would take 10 hours to adjust the input values to be used in the prediction phase. Fortunately, as a matter of fact, on the average case it takes a little bit more than 1 hour.

After performing the calibration for all the generation deadlines, we have obtained the values presented in Figure 40. The average accuracy loss is calculated considering the best result of the simulation as a baseline (the one with the highest deadline per generation). In Figure 40, each data point shows the average for 10 runs, and each error bar represents the corresponding standard deviation. As we can see in the data, there is a clear turning point beyond which a more lenient deadline does not pay off. Being the strict deadline per generation a parameter informed by the wildfire analyst, it is up to him or her to define it carefully considering the trade-off between the calibration accuracy and the timeliness of the wildfire prediction.

In conclusion, a strict deadline of 5 minutes could be chosen for each generation, and it would compromise only about 3.87% of the prediction quality in the calibration. It can be a good bet if we consider that we are guaranteed that the calibration phase will not last more than 50 minutes in the worst-case scenario.

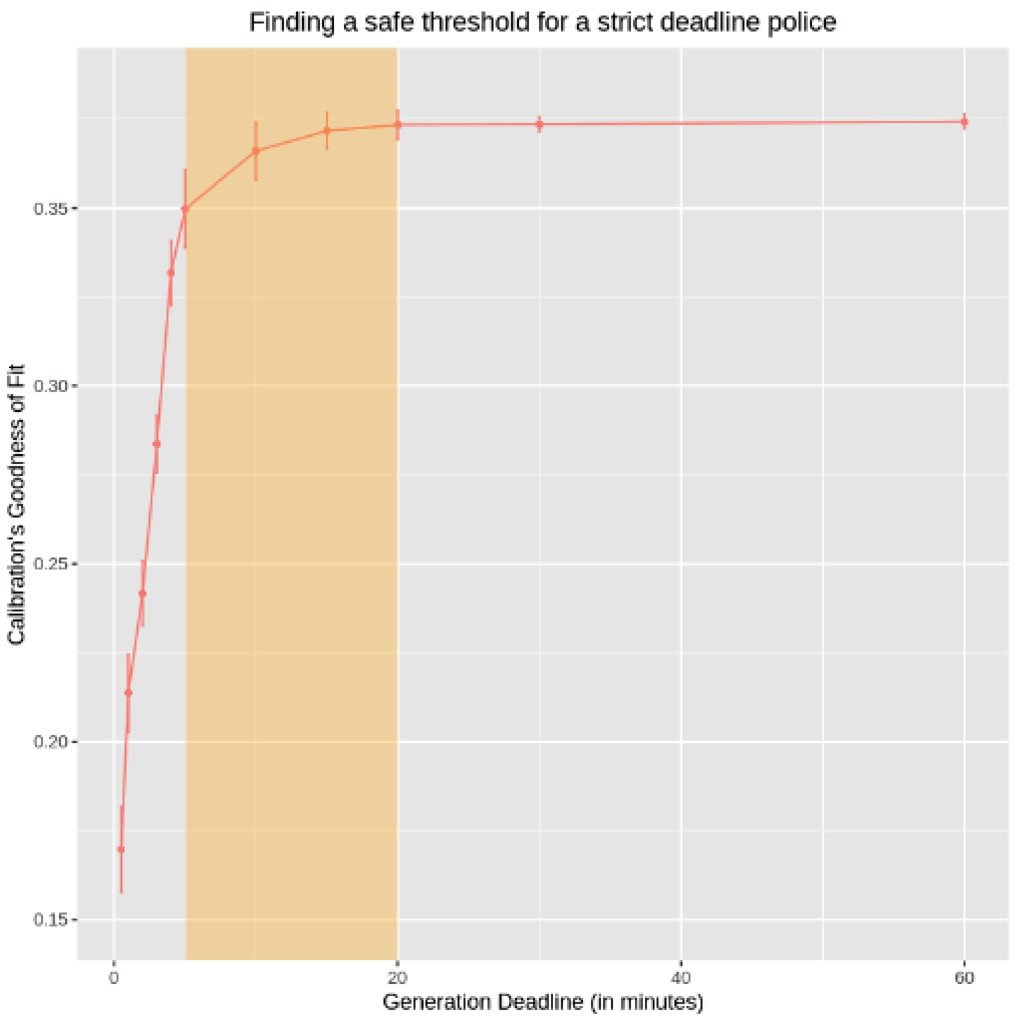


*Figure 40: Calibration time Vs generation deadlines.*

## 4.3. Camp Fire: an extreme case

We have also set up an experimental study to validate the CuCo platform against the Camp Fire, a deadliest and destructive wildfire that occurred in California - USA in the year 2018. The East part of the USA is one of the regions most affected by forest fires during high-risk seasons. In particular, the state of California is frequently affected by wildfires. Regarding the two-stage prediction methodology, its calibration is particularly challenging due to the unusual weather conditions that cause the fire to spread faster than expected. It is a resource-demanding scenario, represents an urge for timely predictions, and gives much less time to act.

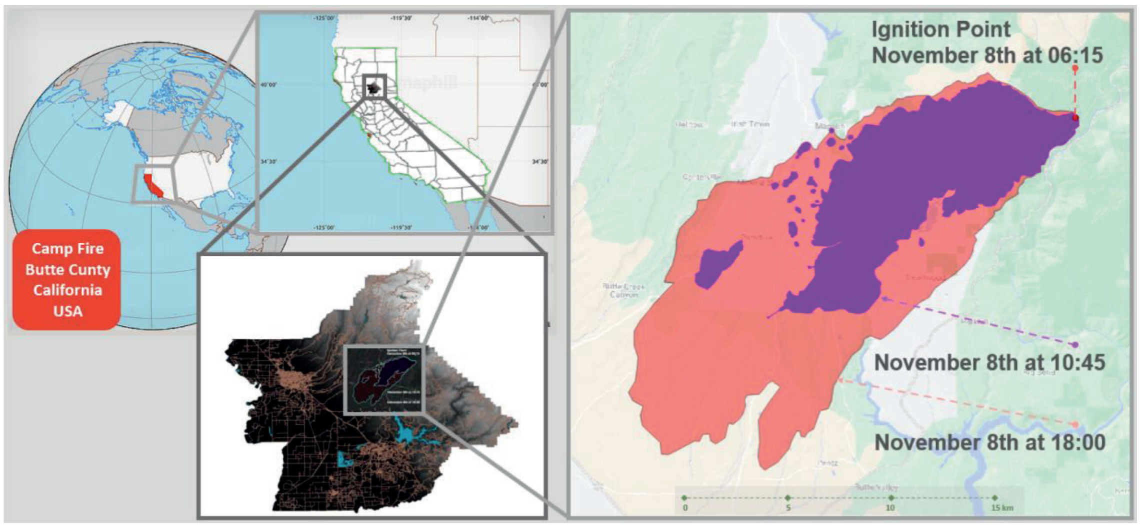


*Figure 41: Camp Fire (California - USA), and its calibration perimeter.*

The Camp Fire was the deadliest and most destructive wildfire in California's history. The fire started on Thursday, November 8, 2018, at 6:15 a.m., in Northern California's Butte County (see Figure 14). It was ignited by a faulty electric transmission line, originated above several communities. An east wind drove the fire downhill through developed areas, and sustained winds of 40 to 48 km/h, with gusts ranging from 65 to 80 km/h, drove rapid fire spread. Drought was a major factor for the fire intensity and spread. Relative humidity across the area was generally below 20 percent. Exhibiting spotting behavior, extreme fire spread, and strong fireline intensity, it run over rural communities, later also forming an urban firestorm in a foothill town. The fire advanced nearly 24 km in the first 12 hours, burning over 22,000 hectares. Only after the arrival of the first winter rainstorm of the season, the fire reached 100 percent containment after seventeen days on November 25.

In the following sections we analyze each iteration of the whole prediction scheme separately, in order to better understand the results obtained when applying the two-stage strategy. Although the fire spread over 17 days in total, we consider only the perimeters in the propagation occurred in the first 12 hours, which was the most critical ones. The propagation results have been compared using the function stated in Equation 9, which evaluates the goodness-of-fit of a fire spread prediction.

| | Property | Value |
|---|---|---|
| **Camp fire** | Area | 620 km² (62,052 hectares) |
| **(USA, 2018)** | Perimeter | 287,92 km |
| **Landscape map** | Dimension | 38.850 km x 36.180 km |
| | Resolution | 30m x 30m |
| | File Size (MiB) | 14.900 |
| **Genetic algorithm** | Number of generations | 10 |
| | Population size | 64 |
| | Crossover probability | 0.9 |
| | Mutation probability | 0.1 |
| **Fire simulation** | Perimeter Resolution | 100m |
| | Distance Resolution | 100m |
| **Infrastructure** | Cluster Size | 4 |
| | Instance Type | c5.2xlarge |

*Table 10: Set up configuration for Camp Fires' scenario.*

### 4.3.1. Calibration window

The time window selected for the Calibration goes from 6:15 a.m. to 10:45 a.m., covering from the ignition point until the first complete perimeter extracted from a satellite image. That is, the calibration stage covers the initial 4,5 hours of the fire event, while the corresponding prediction stage covers the following 7,25 hours (from 10:45 to 18:00).

Figure 42 shows Landsat 8 captures at about 10:45 a.m. PST on Thursday, Nov. 8, just around four hours after fire ignition. In its corresponding short-wave infrared band, the images are processed to show green vegetation in the near infrared (NIR), and dead vegetation and the fire in red colors using shortwave infrared (SWIR), with the burn front clearly visible. It shows the full extent of the actively burning area, and the red patches are fires that leap in front of the primary burn front (spot fires). The fire was growing at a rate of approximately 5 km per hour.

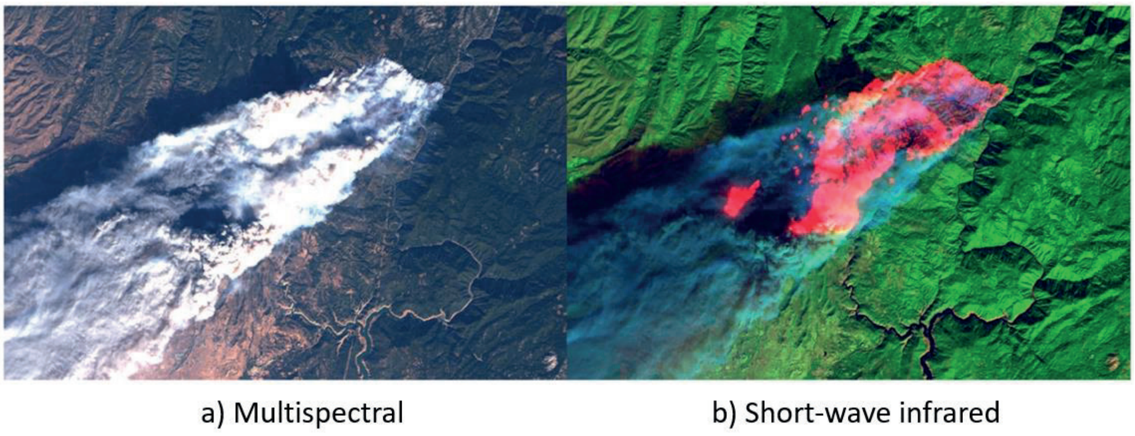


*Figure 42: Landsat 8 multispectral and infrared satellite images.*

We use actual values of elevation, slope and aspect of the terrain, fuels (vegetation types) and canopy cover. The genetic algorithm gets configured to evolve for 10 generations, each one with a population size set to 64 individuals. The probabilities of crossover and mutation used in the executions are 0.8 and 0.1, respectively. Those values are used empirically and has given good calibration results. Each one of the individuals gets simulated, and the resulting fire perimeter is compared to the actual fire spread using the weighted GoF function.

First of all, we need to be sure that there are enough available pods/cores to run all individuals in parallel and then avoid queuing *FARSITE* tasks. In the present case, we are not concerned about what instances get actually chosen to host the pods, as long as we can obtain the maximum level of parallelism at the same performance level. Although we could have by-passed the *Optimized Resource Allocation Planner* directly informing the cluster configuration of preference, we can obtain the same result setting the parameters of the utility function. Setting $\mathrm{b}_1 = \frac{9}{10}$ and $\mathrm{b}_2 = \frac{1}{10}$, i.e., the user is much more concerned with the runtime than with the cost. In this case, the chosen instance is again the c5.2xlarge, but this time for the configuration with 8 nodes.

The results obtained for this calibration-prediction stage are shown in Figure 43. As it can be observed, although not capturing the spotting behavior of the fire, it resembles the main trend of the actual fire shape, resulting in a somewhat good calibration. The parameters corresponding to this best individual can then be immediately fed into the *prediction* stage.

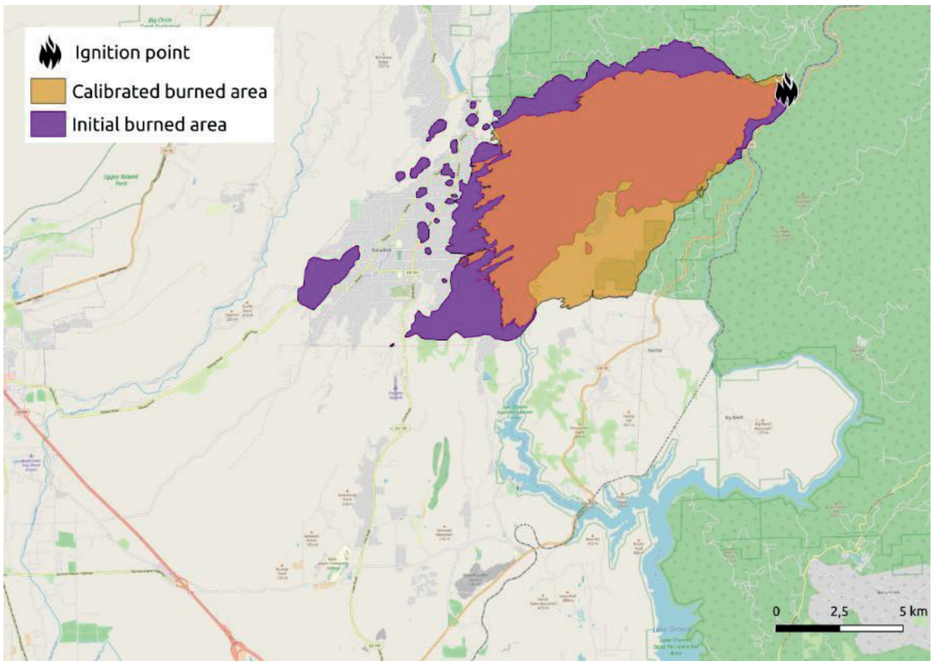


*Figure 43: Result of calibration stage from 6:15 to 10:45.*

### 4.3.2. Prediction window

*Prediction* stage goes from 10:45 to 18:00 and uses the adjusted input data resulting from the *calibration* stage. This simulation is challenging as its reference fire affects both rural communities, urban constructions, and forest fuel. Wildfires in these conditions do not spread exactly as modelled in the fire propagation reference patterns. Besides, the expansion of the high number of spot fires increases the prediction time.

Notwithstanding these caveats, the prediction managed to capture the trends of the actual fire spread, thanks to the incorporation of wind field model. Figure 44 shows the resulting prediction perimeter, contrasted with the actual fire spread at the same time. The resulting prediction overestimates the fire spread, but it clearly describes the three main spread direction of the fire. Overestimate the fire spread prediction is not really a big concern, as in the real case the wildfire does not spread freely due to human intervention.

Curiously, an important fire front on the north-west part was not detected. This was probably due to that part being the one affected by an urban firestorm in the foothill town, difficult to modelling as it propagated mainly by new spot fires over a *Wildland–Urban interface* (*WUI*).

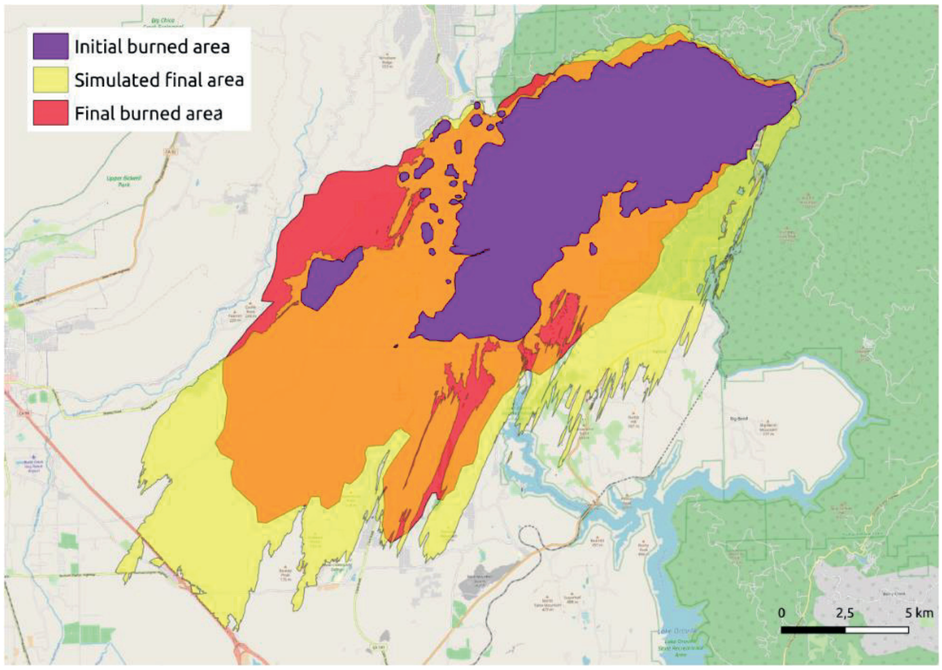


*Figure 44: Result of prediction stage from 10:45 to 18:00.*

The execution of the two-stage prediction scheme has been repeated 10 times and each data point in Figure 45 a) and b) shows the average of these runs. The error bar represents the corresponding standard deviation. Figure 45 a) shows the execution time for the calibration and prediction stages, while Figure 45 b) shows the average *goodness-of-fit* (i.e., prediction quality, see Equation 9) also for both stages.

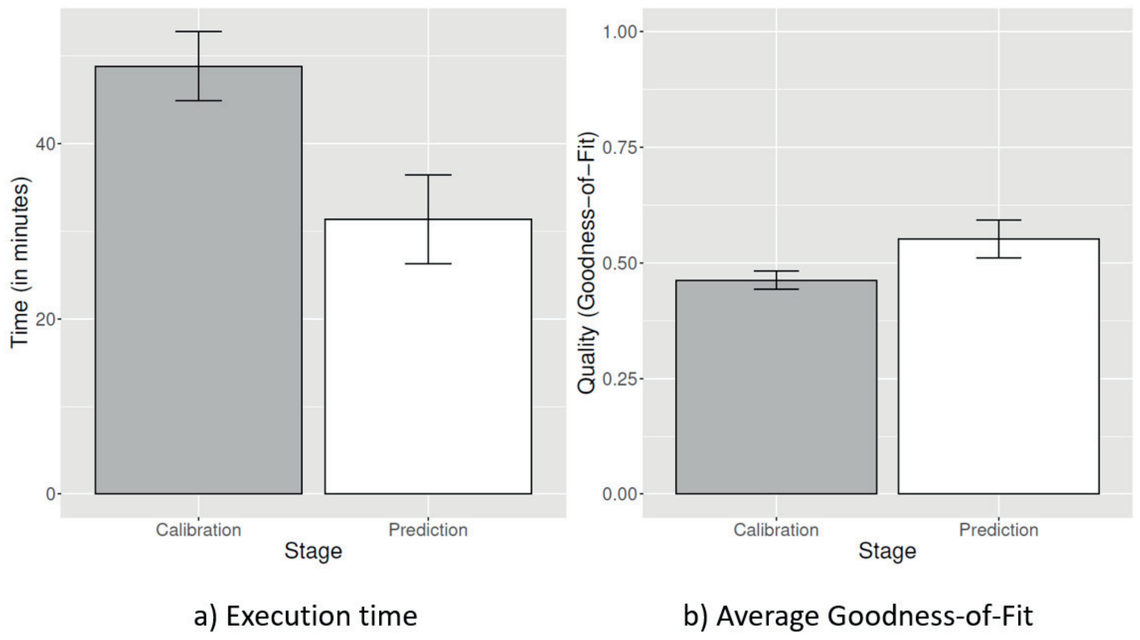


*Figure 45: Execution time and quality of prediction for calibration and prediction stages.*

As we can see in Figure 46, the prediction cost is just a minor fraction of the overall two-stage cost. Due to the high-parallel and compute-intensive nature of the genetic algorithm calibration, its execution consumes 95% of the two-stage prediction budget. Although the prediction stage execution time is around 60% of the calibration stage, we can see that when it comes to the cost, its comparison with the calibration is fairly negligible.

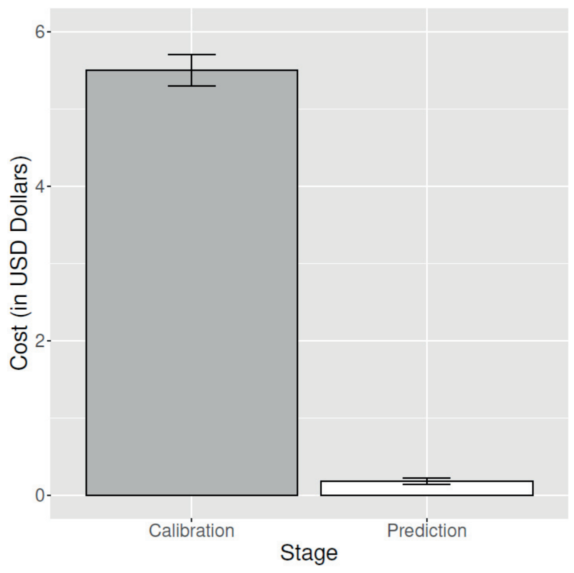


*Figure 46: Cloud cost for calibration and prediction stages.*

Although we have engineered the solution to be cost-effective, i.e., able to balance between cost and time-to-response, in an operational event, the fire analyst may want to allocate the most powerful resource, no matter how much it cost. In this case, the *Optimized Resource Allocation Planner* can be by-passed and the *Resource Provisioner* allocates the cluster configuration informed by the analyst.

For example, if the fire analyst believes that having more virtual machines the workload can be evenly spread between them, resulting in an improved performance, he or she can then choose a configuration with 4 instances of type c5.18xlarge, with 288 *vCPUs* in total (4 X 72 = 288). We would then have the results showed in Figure 47, where we can surprisingly see indeed a significant reduction in the time needed for the calibration stage.

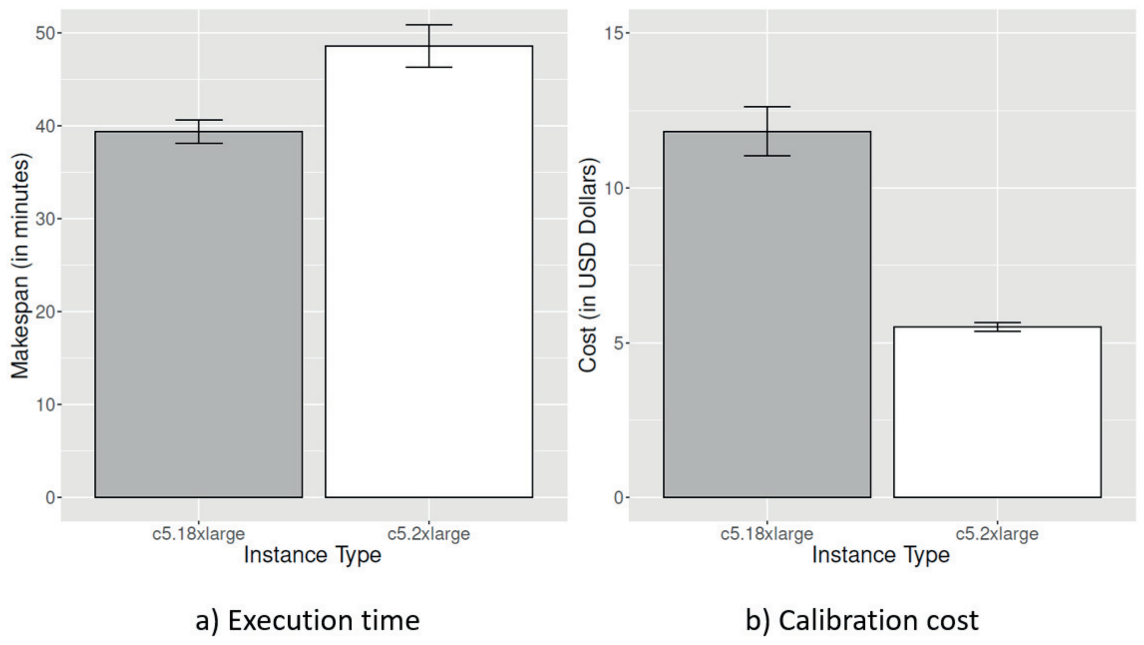


*Figure 47: Comparison of calibration time and cost between clusters of two selected instance types.*

This result cannot be explained by considering the number of cores (or *vCPUs*) alone, as a virtual cluster with 8 instances of type *c5.2xlarge* has exactly 64 vCPUs (8 X 8 = 64), sufficient to run all the individual of a given generation in parallel. The reason must be the underlying resources, as the cloud provider usually allocated the most powerful instances in high-end physical machines. On the other hand, such calibration cost double in comparison with the cluster suggested by the *Optimized Resource Allocation Planner*.

In the end, it is a decision justifiable in the presence of the trade-off between cost and performance while facing a natural hazard. In the day-to-day activities, involving planning wildfire prevention and suppression, where thousands of wildfire simulations need to be run, the decision needs to be well informed and cost-effective.

# Chapter 5: Conclusions

Forest fire is a significant natural hazard that every year cause important damages in areas with high fire risk indexes. Wildfire models and their implementation can provide estimations of fire behavior, but their results are affected by elevated level of data uncertainty. Besides, input parameters are difficult to know or even to estimate so there is a need of strategies to minimize this uncertainty and to provide better predictions. When applied into a real case scenario in production, there is also an extra key challenging factor: the response time. In such situation, late results are useless. They represent a relevant application area in which the use of high-performance computing resources helps to provide more accurate results.

Many research efforts have been focused on the proposal and development of different models to better predict how an ongoing fire will behave. Such forecast is a complex problem by itself, considering the plethora of factors capable of influence these natural phenomena. Apart from that, input data uncertainty represents an additional challenge to devise an accurate model. Consequently, any approach that focuses on accelerating forest fire spread prediction without losing accuracy is more than welcome. This kind of natural hazard is still difficult to be modeled and to be accurately simulated. The research activities described in this thesis aimed to investigate the feasibility of accurately predict a forest fire spread under a strict deadline constraint in a cost-effective way. It relies on the assumption that through cloud computing enable technology it is possible to achieve more accurate prediction results in less time even for larger fire occurrences when compared to traditional *HPC*-based solutions.

In this work, we present a cloud-based solution for the data uncertainty problem in forest fire spread prediction. We build up our work on an established two-stage fire spread prediction model. Our solution uses a *goodness-of-fit* function for the genetic algorithm in the adjustment stage, improving the genetic algorithm convergence and decreasing the response time for the *calibration*. We use an early adaptive-evaluation technique to evaluate the individuals, based on a periodic monitoring of their prediction error, avoiding the waste of computing time running undoubtedly unfit individuals. This strategy, at the same time, simplified the implementation and improved the response time of the critical calibration stage.

The *main objective* of this thesis was to define a *performance-efficient* and *cost-effective cloud-based* solution built upon a *proven methodology for forest fire spread prediction* under data uncertainties. Therefore, from an initial hypothesis that a solution based on cloud offerings may improve the accuracy of fire propagation prediction results, a set of activities was conducted to devise and evaluate an adequate solution to the problem at hand. This primary contribution was published in the *HiPC 2021 Conference*:

> E. Fraga, A. Cortés, T. Margalef and P. Hernández, "***Cloud-Based Urgent Computing for Forest Fire Spread Prediction under Data Uncertainties***," in 2021 IEEE 28th International Conference on High Performance Computing, Data, and Analytics (HiPC), Bengaluru (Bangalore), 2021. [73]

Apart from the technological contribution of using a new computing paradigm, as one of the secondary objectives, we have proposed a new strategy in the core of the ***Two-Stage Prediction Framework***, namely the *Early Adaptive-Evaluation strategy* this contribution was published in the *ICCS 2020 Conference*:

> E. Fraga, A. Cortés, A. Cencerrado, T. Margalef and P. Hernández, "***Early Adaptive Evaluation Scheme for Data-Driven Calibration in Forest Fire Spread Prediction***," in Computational Science - ICCS 2020, Amsterdam, 2020. [74]

At the same time, the strategies of a new *Goodness-of-Fit* function and the *Strict-Deadline Policy* increased the convergence and improve the response time of the critical adjustment stage, being a solid step forward to have an efficient and easy to deploy version of the *Two-Stage Prediction Framework*. These contributions were published in the *2022 e-Science Conference*:

> E. Fraga, A. Cortés, T. Margalef and P. Hernández, **"Efficient Cloud-Based Forest Fire Spread Input Data Calibration,"** in 18th IEEE International Conference on e-Science (eScience'22), Salt Lake City, SL, 2022. [75]

We have also set up an experimental study to validate the platform against the *Camp Fire*, a deadliest and destructive wildfire that occurred in California - USA in the year 2018. The results showed that the final prediction has successfully captured the main fire growth trend for the first 12 hours of fire, using the first 4 hours of data to calibrate and then predict the fire spread for the next 8 hours. The results are currently under review in the Journal *Environmental Modelling & Software*:

> E. Fraga, A. Cortés, T. Margalef, P. Hernández and C. Carrillo, "***Cloud-Based Urgent Computing for Forest Fire Spread Prediction***," in Environmental Modelling & Software (EMS), 2023. [*under review*].

# References

[1] J. S. Miguel-Ayanz, J. M. Moreno e A. Camia, “Analysis of large fires in European Mediterranean landscapes: Lessons learned and perspectives,” *Forest Ecology and Management,* vol. 294, pp. 11-22, 2013.

[2] P. Costa, M. Castellnou, A. Larrañaga, M. Miralles e D. Kraus, Prevention of Large Wildfires using the Fire Types Concept, United Nations Office for Disaster Risk Reduction, 2011.

[3] P. Dennison, S. Brewer, J. Arnold e M. Moritz, “Large wildfire trends in the western United States, 1984-2011,” *Geophysical Research Letters,* vol. 41, April 2014.

[4] J. Pausas, J. Llovet, A. Rodrigo e R. Vallejo, “Are wildfires a disaster in the Mediterranean basin? - A review.,” *International Journal of Wildland Fire,* vol. 17, January 2008.

[5] J. Pausas e S. Muñoz, “Fire regime changes in the Western Mediterranean Basin: From fuel-limited to drought-driven fire regime,” *Climatic Change,* vol. 110, pp. 215-226, January 2012.

[6] M. V. Moreno, M. Conedera, E. Chuvieco e G. B. Pezzatti, “Fire regime changes and major driving forces in Spain from 1968 to 2010,” *Environmental Science & Policy,* vol. 37, pp. 11-22, 2014.

[7] J.-L. Rossi, T. Marcelli, J. Chatelon, D. Morvan e A. Simeoni, “Fuelbreaks: a part of wildfire prevention,” United Nations Office for Disaster Risk Reduction, 2019, p. 25.

[8] S. H. Leong e D. Kranzlmüller, “Towards a General Definition of Urgent Computing,” *Procedia Computer Science,* vol. 51, pp. 2337-2346, 2015.

[9] R. Buyya, High Performance Cluster Computing, Prentice Hall PTR, 1999.

[10] S. H. Leong, A. Frank e D. Kranzlmüller, “Leveraging e-Infrastructures for Urgent Computing,” *Procedia Computer Science,* vol. 18, pp. 2177-2186, 2013.

[11] K. V. Knyazkov, D. A. Nasonov, T. N. Tchurov e A. V. Boukhanovsky, “Interactive Workflow-based Infrastructure for Urgent Computing,” *Procedia Computer Science,* vol. 18, pp. 2223-2232, 2013.

[12] S. V. Kovalchuk, P. A. Smirnov, S. V. Maryin, T. N. Tchurov e V. A. Karbovskiy, “Deadline-driven Resource Management within Urgent Computing Cyberinfrastructure,” *Procedia Computer Science,* vol. 18, pp. 2203-2212, 2013.

[13] R. Brzoza-Woch, M. Konieczny, B. Kwolek, P. Nawrocki, T. Szydło e K. Zieliński, “Holistic

Approach to Urgent Computing for Flood Decision Support," *Procedia Computer Science,* vol. 51, pp. 2387-2396, 2015.

[14] K. K. Yoshimoto, D. J. Choi, R. L. Moore, A. Majumdar e E. Hocks, "Implementations of Urgent Computing on Production HPC Systems," *Procedia Computer Science,* vol. 9, pp. 1687-1693, 2012.

[15] S. H. Leong e D. Kranzlmüller, "Urgent Computing - A General Makespan Robustness Model for Ensembles of Forecasts," *Procedia Computer Science,* vol. 80, pp. 2062-2073, 2016.

[16] M. P. Thompson e D. E. Calkin, "Uncertainty and risk in wildland fire management: A review," *Journal of Environmental Management,* vol. 92, pp. 1895-1909, 2011.

[17] A. Benali, A. R. Ervilha, A. C. Sá, P. M. Fernandes, R. M. Pinto, R. M. Trigo e J. M. Pereira, "Deciphering the impact of uncertainty on the accuracy of large wildfire spread simulations," *Science of The Total Environment,* vol. 1, pp. 73-85, 2016.

[18] L. Cai, H. S. He, Y. Liang, Z. Wu e C. Huang, "Analysis of the uncertainty of fuel model parameters in wildland fire modelling of a boreal forest in north-east China," *International Journal of Wildland Fire,* vol. 28, p. 205, 2019.

[19] A. G. Carlyle, S. L. Harrell e P. M. Smith, "Cost-Effective HPC: The Community or the Cloud?," em *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, 2010.

[20] T. Artés, A. Cencerrado, A. Cortés e T. Margalef, "Relieving the Effects of Uncertainty in Forest Fire Spread Prediction by Hybrid MPI-OpenMP Parallel Strategies," *Procedia Computer Science,* vol. 18, pp. 2278-2287, 2013.

[21] A. Cencerrado, A. Cortés e T. Margalef, "Genetic Algorithm Characterization for the Quality Assessment of Forest Fire Spread Prediction," *Procedia Computer Science,* vol. 9, pp. 312-320, 2012.

[22] G. Bianchini, P. Caymes-Scutari e M. Méndez-Garabetti, "Evolutionary-Statistical System: A parallel method for improving forest fire spread prediction," *Journal of Computational Science,* vol. 6, pp. 58-66, 2015.

[23] I. Altintas, J. Block, R. de Callafon, D. Crawl, C. Cowart, A. Gupta, M. Nguyen, H.-W. Braun, J. Schulze, M. Gollner, A. Trouve e L. Smarr, "Towards an Integrated Cyberinfrastructure for Scalable Data-driven Monitoring, Dynamic Prediction and Resilience of Wildfires," *Procedia Computer Science,* vol. 51, pp. 1633-1642, 2015.

[24] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica e M. Zaharia, "A View of Cloud Computing," *Communications of the ACM,* vol. 53, p. 50–58, April 2010.

[25] M. A. S. Netto, R. N. Calheiros, E. R. Rodrigues, R. L. F. Cunha e R. Buyya, “HPC Cloud for Scientific and Business Applications: Taxonomy, Vision, and Research Challenges,” *ACM Computing Surveys,* vol. 51, January 2018.

[26] S. Jamalian e H. Rajaei, “ASETS: A SDN Empowered Task Scheduling System for HPCaaS on the Cloud,” em *2015 IEEE International Conference on Cloud Engineering*, 2015.

[27] A. Gupta e D. Milojicic, “Evaluation of HPC Applications on Cloud,” em *2011 Sixth Open Cirrus Summit*, 2011.

[28] H. Kim, Y. el-Khamra, S. Jha e M. Parashar, “An Autonomic Approach to Integrated HPC Grid and Cloud Usage,” em *2009 Fifth IEEE International Conference on e-Science*, 2009.

[29] S. Jamalian e H. Rajaei, “Data-Intensive HPC Tasks Scheduling with SDN to Enable HPC-as-a-Service,” em *2015 IEEE 8th International Conference on Cloud Computing*, 2015.

[30] T. Srivas, R. A. de Callafon, D. Crawl e I. Altintas, “Data Assimilation of Wildfires with Fuel Adjustment Factors in farsite using Ensemble Kalman Filtering,” *Procedia Computer Science,* vol. 108, pp. 1572-1581, 2017.

[31] B. Abdalhaq, A. Cortés, T. Margalef e E. Luque, “Enhancing wildland fire prediction on cluster systems applying evolutionary optimization techniques,” *Future Generation Computer Systems,* vol. 21, pp. 61-67, 2005.

[32] M. Méndez-Garabetti, G. Bianchini, M. L. Tardivo e P. Caymes-Scutari, “Comparative Analysis of Performance and Quality of Prediction Between ESS and ESS-IM,” *Electronic Notes in Theoretical Computer Science,* vol. 314, pp. 45-60, 2015.

[33] T. Artés, A. Cortés e T. Margalef, “Large Forest Fire Spread Prediction: Data and Computational Science,” *Procedia Computer Science,* vol. 80, pp. 909-918, 2016.

[34] C. Carrillo, A. Cortés, M. Tomàs, A. Espinosa e A. Cencerrado, “Relevance of Error Function in Input Parameter Calibration in a Coupled Wind Field Model-Forest Fire Spread Simulator,” em *2018 International Conference on High Performance Computing Simulation*, Orleans, France, July 16-20, 2018.

[35] S. Garg, J. Aryal, H. Wang, T. Shah, G. Kecskemeti e R. Ranjan, “Cloud computing based bushfire prediction for cyber–physical emergency applications,” *Future Generation Computer Systems,* vol. 79, pp. 354-363, 2018.

[36] J. Ramirez, S. Monedero e D. Buckley, “New approaches in fire simulations analysis with Wildfire Analyst,” em *5th International Wildland fore conference*, 2011.

[37] S. Monedero, J. Ramirez e A. Cardil, “Predicting fire spread and behaviour on the fireline. Wildfire analyst pocket: A mobile app for wildland fire prediction,” *Ecological Modelling,* vol. 392, pp. 103-107, 2019.

[38] K. Kalabokidis, N. Athanasis, C. Vasilakos e P. Palaiologou, “Porting of a Wildfire Risk and Fire Spread Application into a Cloud Computing Environment,” *International Journal of Geographical Information Science,* vol. 28, p. 541–552, March 2014.

[39] C. Miller, J. Hilton, A. Sullivan e M. Prakash, “SPARK -- A Bushfire Spread Prediction Tool,” *Environmental Software Systems. Infrastructures, Services and Applications,* pp. 262--271, 2015.

[40] B. Arca, T. Ghisu, M. Casula, M. Salis e P. Duce, “A web-based wildfire simulator for operational applications,” *International journal of wildland fire,* vol. 28, pp. 99-112, 2019.

[41] U. Oliveira, B. Soares-Filho, H. Rodrigues, D. Figueira, L. Gomes, W. Leles, C. Berlinck, F. Morelli, M. Bustamante, J. Ometto e H. Miranda, “A near real-time web-system for predicting fire spread across the Cerrado biome,” *Scientific Reports,* vol. 13, mar 2023.

[42] S. J. Pyne, P. L. Andrews e R. D. Laven, Introduction to wildland fire, 2nd ed., New York, NY: John Wiley and Sons, Inc, 1996, p. 769.

[43] R. C. Rothermel, A Mathematical Model for Predicting Fire Spread in Wildland Fuels, Intermountain Forest & Range Experiment Station, Forest Service, U.S. Department of Agriculture, 1972.

[44] L. F. DeBano, D. G. Neary e P. F. Ffolliott, Fire's effects on ecosystems, 1st ed., John Wiley & Sons, 1998.

[45] S. Rozen e A. Hagooly, *Bromochlorodifluoromethane,* John Wiley & Sons, Ltd, 2005.

[46] H. E. Anderson, “Aids to Determining Fuel Models for Estimating Fire Behavior,” U.S. Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station, 1982.

[47] J. K. Balch, B. A. Bradley, J. T. Abatzoglou, R. C. Nagy, E. J. Fusco e A. L. Mahood, “Human-started wildfires expand the fire niche across the United States,” *Proceedings of the National Academy of Sciences,* vol. 114, nº 11, pp. 2946-2951, 2017.

[48] A. Ganteaume, A. Camia, J. Marielle, J. San-Miguel-Ayanz, M. Long-Fournel e C. Lampin, “A Review of the Main Driving Factors of Forest Fire Ignition Over Europe,” vol. 51, pp. 651--662, 3 oct 2012.

[49] NWCG, “S-190 Introduction to Wildland Fire Behavior,” National Wildfire Coordinating Group, 2023.

[50] D. Leavell, C. Berger, S. Fitzgerald e B. Parker, “Fire Science Core Curriculum,” Oregon State University, OSU Extension Catalog, 2017.

[51] G. Xanthopoulos e M. Athanasiou, “Crown Fire,” em *Encyclopedia of Wildfires and Wildland-Urban Interface ({WUI}) Fires*, S. L. Manzello, Ed., Springer International

Publishing, 2019, pp. 1--15.

[52] G. Pagnini, “Fire Spotting Effects in Wildland Fire Propagation,” em *Advances in Differential Equations and Applications*, Springer International Publishing, 2014, pp. 203--214.

[53] M. A. Finney, FARSITE, Fire Area Simulator - Model Development and Evaluation, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, 1998.

[54] G. D. Richards, “An elliptical growth model of forest fire fronts and its numerical solution,” *International journal for numerical methods in engineering,* vol. 30, nº 6, pp. 1163--1179, 1990.

[55] C. E. Van Wagner, “Prediction of crown fire behavior in conifer stands,” em *10th conference on fire and forest meteorology*, Ottawa, Ontario, 1989.

[56] J. H. Scott e R. E. Burgan, “Standard Fire Behavior Fuel Models: A Comprehensive Set for Use with Rothermel's Surface Fire Spread Model,” USDA - United States Department of Agriculture, Rocky Mountain Research Station, 2005.

[57] J. J. Keetch e G. M. Byram, “A Drought Index for Forest Fire Control,” U.S. Department of Agriculture, Forest Service, Southeastern Forest Experiment Station, Asheville, NC, 1968.

[58] F. Albini, “Estimating wildfire behavior and effects,” USDA Forest Service, Intermountain Forest and Range Experiment Station, 1976.

[59] J. Forthofer, K. Shannon e B. Butler, “Simulating diurnally driven slope winds with WindNinja,” em *Proceedings of 8th Eighth Symposium on Fire and Forest Meteorology*, Kalispell, MT. Boston, MA, 2009.

[60] L. Giglio, J. Descloitres, C. O. Justice e Y. J. Kaufman, “An Enhanced Contextual Fire Detection Algorithm for MODIS,” *Remote Sensing of Environment,* vol. 87, pp. 273-282, 10 2003.

[61] M. J. Wooster, G. J. Roberts, L. Giglio, D. P. Roy, P. H. Freeborn, L. Boschetti, C. Justice, C. Ichoku, W. Schroeder, D. Davies, A. M. Smith, A. Setzer, I. Csiszar, T. Strydom e P. Frost, “Satellite remote sensing of active fires: History and current status, applications and future requirements,” *Remote Sensing of Environment,* vol. 267, p. 112694, 2021.

[62] NASA, “MODIS (Moderate Resolution Imaging Spectroradiometer),” NASA (National Aeronautics and Space Administration), 1999. [Online]. Available: https://modis.gsfc.nasa.gov/about/. [Acesso em 01 05 2023].

[63] NASA/USGS, “Landsat - Earth Resources Technology Satellite,” NASA/USGS (National Aeronautics and Space Administration/United States Geological Survey), 1972. [Online]. Available: https://landsat.gsfc.nasa.gov/. [Acesso em 01 05 2023].

[64] ESA, “Copernicus - Earth Observation component of the European Union’s space programme,” ESA (European Space Agency), 2014. [Online]. Available: https://www.copernicus.eu/en. [Acesso em 01 05 2023].

[65] R. D. Stratton, “Guidebook on LANDFIRE fuels data acquisition, critique, modification, maintenance, and model calibration,” U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, Fort Collins, CO, 2009.

[66] W. Walsh, G. Tesauro, J. Kephart e R. Das, “Utility functions in autonomic systems,” em *International Conference on Autonomic Computing, 2004. Proceedings.*, 2004.

[67] AWS, “Amazon Web Services,” 2006. [Online]. Available: https://aws.amazon.com/. [Acesso em 01 05 2023].

[68] FWI, “Fire Weather Index System,” 2008. [Online]. Available: https://www.nwcg.gov/publications/pms437/cffdrs/fire-weather-index-system. [Acesso em 01 05 2023].

[69] D. Haines, “A lower atmospheric severity index for wildland fire,” *National Weather Digest,* vol. 13, pp. 23-27, 1988.

[70] J. H. Friedman, “Multivariate Adaptive Regression Splines,” *The Annals of Statistics,* vol. 19, p. 1 – 67, 1991.

[71] W. W. Hargrove, F. M. Hoffman e P. Hessburg, “Mapcurves: A Quantitative Method for Comparing Categorical Maps,” *Journal of Geographical Systems,* vol. 8, January 2006.

[72] C. Brun, T. Artes, A. Cencerrado, T. Margalef e A. Cortés, “A High Performance Computing Framework for Continental-Scale Forest Fire Spread Prediction,” *Procedia Computer Science,* vol. 108, pp. 1712-1721, 2017.

[73] E. Fraga, A. Cortés, T. Margalef e P. Hernández, “Cloud-Based Urgent Computing for Forest Fire Spread Prediction under Data Uncertainties,” em *2021 IEEE 28th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, Bangalore, 2021.

[74] E. Fraga, A. Cortés, A. Cencerrado, T. Margalef e P. Hernández, “Early Adaptive Evaluation Scheme for Data-Driven Calibration in Forest Fire Spread Prediction,” em *Computational Science -- ICCS 2020*, Amsterdam, 2020.

[75] E. Fraga, A. Cortés, T. Margalef e P. Hernández, “Efficient Cloud-Based Forest Fire Spread Input Data Calibration,” em *18th IEEE International Conference on e-Science (eScience'22)*, Salt Lake City, SL, 2022.