Universitat Autònoma de Barcelona

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

Methodology for Time Response and Quality Assessment in Natural Hazards Evolution Prediction

Thesis submitted by **Andrés Cencerrado Barraqué** in fulfillment of the requirements for the doctoral degree from Universitat Autònoma de Barcelona, advised by Dra. Ana Cortés Fité.

Barcelona, May 4th, 2012

Methodology for Time Response and Quality Assessment in Natural Hazards Evolution Prediction

Thesis submitted by **Andrés Cencerrado Barraqué** in fulfillment of the requirements for the doctoral degree from Universitat Autònoma de Barcelona. This work has been developed under RD 1393/2007 in the High Performance Computing doctoral program and presented to the Computer Architecture & Operating Systems Department at the Escola d'Enginyeria of Universitat Autònoma de Barcelona. This thesis was advised by Dra. Ana Cortés Fité.

Bellaterra, Barcelona (Spain). May 2012.

Thesis advisor:

Ana Cortés Fité

Abstract

This thesis describes a methodology for time response and quality assessment in natural hazards evolution prediction. This work has been focused on the specific case of forest fires as an important and worrisome catastrophe, but it can easily be extrapolated to all other kinds of natural hazards.

There exist many prediction frameworks based on the use of simulators of the evolution of the hazard. Given the increasing computing capabilities allowed by new computing advances such as multicore and manycore architectures, and even distributed-computing paradigms, such as Grid and Cloud Computing, the need arises to be able to properly exploit the computational power they offer.

This goal is fulfilled by introducing the capability to assess in advance how the present constraints at the time of attending to an ongoing forest fire will affect the results obtained from them, both in terms of quality (accuracy) obtained and time needed to make a decision, and therefore being able to select the most suitable configuration of both the prediction strategy and computational resources to be used.

As a consequence, the framework derived from the application of this methodology is not supposed to be a new Decision Support System (DSS) for fire departments and Civil Protection agencies, but a tool from which most of forest fire (and other kinds of natural hazards) DSSs could benefit notably.

The problem has been tackled by means of characterizing the behavior of these two factors during the prediction process. For this purpose, a two-stage prediction framework is presented and considered as a suitable and powerful strategy to enhance the quality of the predictions.

This methodology involves dealing with Artificial Intelligence techniques, such as Genetic Algorithms and Decision Trees and also relies on a strong statistical study from training databases, composed of the results of thousands of different simulations.

The results obtained in this long-term research work are fully satisfactory, and give rise to several new challenges. Moreover, the flexibility offered by the methodology allows it to be

applied to other kinds of emergency contexts, which turns it into an outstanding and very useful tool in fighting against these catastrophes.

Resumen

En esta tesis doctoral se describe una metodología para la evaluación del tiempo de respuesta y la calidad en la predicción de la evolución de emergencias medioambientales. El trabajo se ha centrado en el caso específico de los incendios forestales, como uno de los desastres naturales más importantes y devastadores, pero es fácilmente extrapolable a otro tipo de emergencias medioambientales.

Existen muchos entornos de predicción que se basan en el uso de simuladores de la evolución del fenómeno catastrófico. Dado el creciente poder en cuanto a capacidad de cómputo que nos ofrecen los nuevos avances computacionales, como las arquitecturas *multicore* y *manycore*, e incluso los paradigmas de cómputo distribuido, como *Grid* o *Cloud Computing*, surge la necesidad de ser capaces de explotar acertadamente el poder computacional que éstos nos ofrecen.

Tal objetivo se alcanza proporcionando la capacidad de evaluar, de antemano, cómo las restricciones existentes a la hora de atender un incendio forestal activo afectarán a los resultados que se obtendrán, tanto en términos de calidad (precisión) obtenida, y tiempo necesario para tomar una decisión, y por consiguiente, tener la capacidad de escoger la configuración más adecuada tanto de la estrategia de predicción, como de los recursos computacionales.

Como consecuencia, el sistema que deriva de la aplicación de esta metodología no está diseñado para ser un Sistema de Soporte a las Decisiones (DSS), pero sí una herramienta de la que la mayoría de DSSs para incendios forestales se pueden beneficiar notablemente.

El problema se ha tratado por medio de la caracterización del comportamiento de estos dos factores durante el proceso de predicción. Para ello, un método de predicción de dos etapas es presentado y utilizado como base de trabajo, dado el notable aumento de calidad que proporciona en las predicciones.

Esta metodología implica lidiar con técnicas propias del campo de la Inteligencia Artificial, como son los Algoritmos Genéticos y los Árboles de Decisión, y a su vez se apoya en un intenso estudio estadístico de bases de datos de entrenamiento, compuestas por los resultados de miles de distintas simulaciones.

Los resultados obtenidos en este trabajo de investigación a largo plazo son completamente satisfactorios, y abren camino a nuevos retos. Además, la flexibilidad que ofrece la metodología permite aplicarla en cualquier otro contexto de emergencia, lo que la convierte en una destacable y muy útil herramienta para luchar contra estas catástrofes.

Resum

En aquesta tesi doctoral es descriu una metodologia per a l'evaluació del temps de resposta i la qualitat en la predicció de l'evolució d'emergències mediambientals. El treball s'ha centrat en el cas específic dels incendis forestals, com un dels desastres naturals més importants i devastadors, però és facilment extrapol.lable a altre tipus d'emèrgencies mediambientals.

Existeixen molts entorns de predicció que es basen en l'ús de simuladors de l'evolució del fenòmen catastròfic. Donat el creixent poder quant a capacitat de cómput que ens ofereixen els nous progressos computacionals, com les arquitectures *multicore* i *manycore*, i inclús els paradigmes de cómput distribuit, com *Grid* o *Cloud Computing*, sorgeix la necessitat d'explotar encertadament el poder computacional que aquests ens ofereixen.

Aquest objectiu s'assoleix proporcionant la capacitat d'avaluar, per endavant, com les restriccions existents en el moment d'atendre un incendi forestal actiu afectaran als resultats que s'obtindran, en termes de qualitat (precisió) obtinguda, i temps necessari per prendre una decisió, i en conseqüència, tenir la capacitat de escollir la configuració més adient tant de l'estratègia de predicció, com dels recursos computacionals.

Com a conseqüència, el sistema que deriva de l'aplicació d'aquesta metodologia no està dissenyat per ser un Sistema de Suport a les Decisions (DSS), però sí una eina de la que la majoria de DSSs per incendis forestals es poden beneficiar notablement.

El problema s'ha tractat per mitjà de la caracterització del comportament d'aquests dos factors durant el procés de predicció. Per això, es presenta un mètode de predicció de dues etapes i s'utilitza com a base de treball, donat el notable augment de qualitat que proporciona en les prediccions.

Aquesta metodologia implica haver de treballar amb tècniques pròpies del camp de la Intel.ligència Artificial, com són els Algorismes Genètics i els Arbres de Decisió, i també es recolza en un intens estudi estadístic de les bases de dades d'entrenament, compostes pels resultats de milers de simulacions.

Els resultats obtinguts en aquest treball d'investigació de llarga durada són completament satisfactoris, i obren camí a nous reptes. A més, la flexibilitat que ofereix aquesta metodologia permet aplicar-la en qualsevol altre context d'emergència, el qual la converteix en una destacable i molt útil eina per lluitar contra aquestes catàstrofes.

Agradecimientos

Recuerdo que en mi primera clase de instituto la profesora nos dijo que uno entraba ahí siendo niño, y salía hecho un hombre. No me lo creí, y al finalizar esa etapa, constaté que en absoluto tenía razón. Al empezar la carrera en la universidad, un profesor volvió a decir lo mismo. En esa ocasión fui más ingenuo, y di crédito a esas palabras. De nuevo, al finalizar mis estudios, y a pesar de todo lo vivido, me di cuenta de que no sentía que hubiera dejado de ser un niño. Me hizo especial gracia cuando al empezar el doctorado también se lo oí mencionar a alguien.

Tal y como me siento al culminar esta importante etapa, lo lógico sería decir que estaban absolutamente equivocados. Paradójicamente, no considero que sea así. Mirando atrás y sopesando todas las experiencias por las que he pasado, me doy cuenta de que lo que ocurre es que he tenido la tremenda suerte de verme siempre rodeado de personas que han evitado que me endurezca como lo que cualquiera considera un *hombre*.

La vida me ha regalado valiosas personas que siempre han estado presentes, como mi familia. También ha puesto en camino valiosas personas que, una vez han aparecido, no han vuelto a desaparecer, ni quiero que lo hagan. También me ha obsequiado con la suerte de reencontrarme con otras que conectan distintas etapas, así como de contínuamente sorprenderme al conocer otras nuevas y maravillosas, y siempre con la fortuna de tener siempre bien claro quién de entre ellas no quiero que dejen nunca de formar parte de mí.

Mi agradecimiento más sincero es para todas esas personas que con su confianza, su cariño, su apoyo, su alegría y su amor han permitido que a estas alturas me siga sintiendo alegre como un niño, confiado como un niño, ingenuo como un niño, querido como un niño. Sin todos ellos, no sería el que soy.

No me gusta mencionar nombres propios porque temo olvidar alguno, y porque incluiría una a una todas las personas con las que he compartido cualquier momento bueno, pero siento la necesidad de destacar a aquellos que más me han ayudado en esta etapa:

Mama, Dani, Chiqui: no podría imaginar otra familia mejor para mí, y sé que siempre os tendré ahí. Os amo.

Carlos, Núria, Laura, Uku: mi segunda familia, la que mejor me conoce, la que me permite ser yo mismo sin más, por quien más me siento querido sin ser, aunque parezca mentira, de la misma sangre. Os llevo dentro.

Uri: no hay palabras para describir esta época memorable que ahora acabamos. Lo has sido durante casi mi vida entera, y siempre serás también, parte de mí.

Albert, Alba, Edgar, Gerard y demás "derivados" de la época del Balmes: al fin y al cabo, si siempre seguimos juntos, será porque nos queremos demasiado, estoy convencido. Os quiero.

Ester: qué suerte conocerte, quién nos lo iba a decir. Gracias por todo, sabes que siempre podrás contar conmigo.

Marta: te lo he repetido mil veces, y lo haré las veces que haga falta, no sé qué he podido hacer para encontrarme a alguien tan valioso como tú. Lo que me has ayudado, apoyado, y lo muchísimo que me has facilitado la vida en estos últimos años hace que siempre considere que te he dado tan poco a cambio... Muchas gracias por todo, por ser así de buena persona. Puedes estar segura de que lo recordaré y lo valoraré siempre.

Anna y Tomàs: para mi sois un espejo en el que me quiero ver reflejado. Espero conseguirlo.

Mónica, Darío, Hisham, Kerstin, Carlos (otra vez) y Tomás: mis pirómanos, sin vosotros no hubiera sido igual de valiosa esta experiencia.

Por último, recordaré a la persona más importante en mi vida, porque siempre le deberé a él toda mi parte buena, y todo lo bueno que consiga. Papa, tú me enseñaste a valorar lo importante de la vida, y las cosas por las que merece derrochar mi esfuerzo, sea cuanto sea. Te sigo teniendo dentro muy vivo, y parece mentira el tiempo que ha pasado desde que te fuiste. Todo esto culmina en una fecha demasiado especial, y como también me enseñaste a no creer en las casualidades, sé que ahí estarás apoyándome. Te lo agradezco todo, te quiero, y volveremos a vernos.

A Dios doy gracias por teneros a todos vosotros.

Contents

1 Introduction

1.1 Natural Hazards, a Permanent Threat

Since the very beginning of Computational Science as a research discipline, many efforts have been oriented towards facing and solving complex problems which present great challenges and serious threats to the social welfare state.

From *natural science* areas such as medicine, biology, physics, and chemistry to more technical ones, including every engineering field, many benefits and solutions have been historically obtained, both in terms of preventing and eradicating the problems in question.

Natural hazards, however, represent a permanent thread whose consequences may be catastrophic. For this reason, as will be detailed in Chapter 2, many research activities have been focused both on the prediction of their occurrence, and their evolution once they have taken place, in order to minimise the effects as much as possible.

Indeed, the number of damages caused by natural catastrophes have been increasing significantly over the last decades [1]. In the following subsection, some statistics about the consequences they entail are given.

Hazard	**Hazard type**	**Major impacts**
Storms	Hydrometeorological	Economic losses, human fatalities
Extreme temperature events	Hydrometeorological	Human fatalities
Forest Fires	Hydrometeorological	Human fatalities, ecosystem degradation
Water scarcity and droughts	Hydrometeorological	Economic losses, ecosystem degradation
Floods	Hydrometeorological	Economic losses, human fatalities
Snow avalanches	Geophysical	Human fatalities, economic losses
Landslides	Geophysical	Human fatalities, economic losses
Earthquakes/volcanoes	Geophysical	Human fatalities, economic losses
Oil Spills	Technological	Pollution of ecosystems
Industrial accidents	Technological	Pollution of ecosystems
Toxic spills	Technological	Pollution of ecosystems

Tab. 1.1: Typification of hazards and their major impacts, according to EEA.

Hazard type	Recorded events	Number of fatalities	Number of people affected (million people)	Overall losses (billion EUR)	Insured losses (billion EUR)
Storm	155	729	3.803	44.338	20.532
Extreme temperature	101	77551	0.005	9.962	0.186
Forest Fires	35	191	0.163	6.917	0.097
Drought	8	0	0	4.940	0.000
Flood	213	1126	3.145	52.173	12.331
Snow avalanche	8	130	0.01	0.742	0.198
Landslide	9	212	0.007	0.551	0.206
Earthquake	46	18864	3.978	29.205	2.189
Volcano	1	0	0	0.004	0.000
Total	**576**	**98803**	**11.112**	**148.831**	**35.739**

Tab. 1.2: Disasters caused by natural hazards in Europe in 1998-2009, as recorded in EM-DAT.

1.1.1 Natural Hazards Losses. Some Statistics.

According to the European Environment Agency (EEA), over the last years Europe has experienced an increasing number of natural disasters caused by a combination of changes in its physical, technological and human/social systems [41]. Table 1.1 shows a typification of hazards and their major impacts.

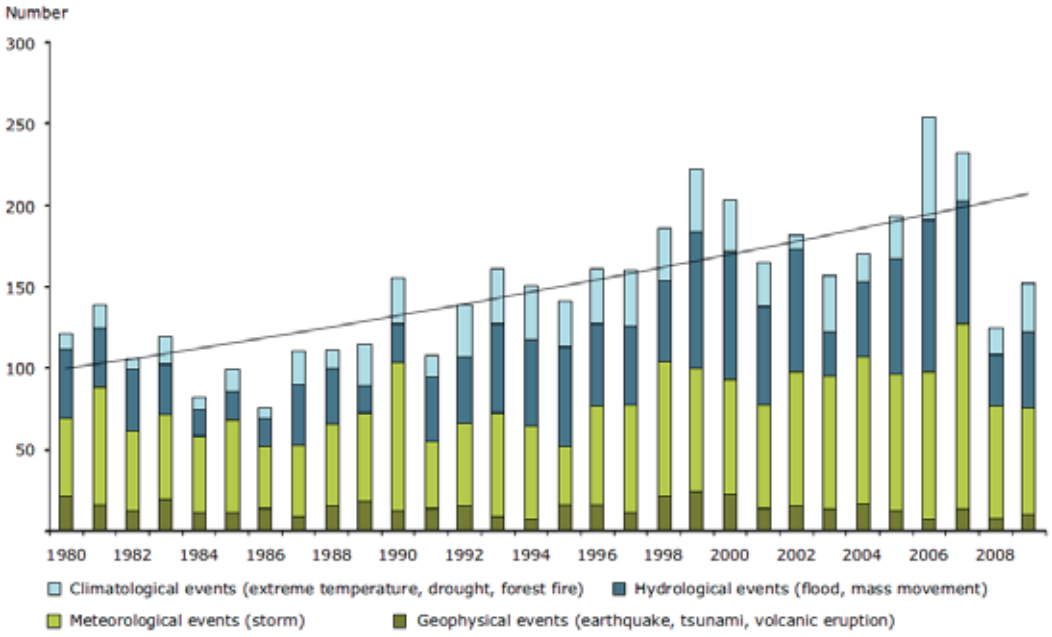

Fig. 1.1: Disasters due to natural hazards in EEA member countries, 1980-2009.

Based on the data obtained from the EM-DAT database [24] maintained by the Centre for Research on Epidemiology of Disasters (CRED) for the period 1998-2009, 576 disasters were due to natural hazards, causing nearly 100 000 fatalities, and close to EUR 150 billion

in overall losses. During this period, more than 11 million people (out of 590 million, approximately, in the EEA member countries) were somehow affected by disasters caused by natural hazards (see Table 1.2).

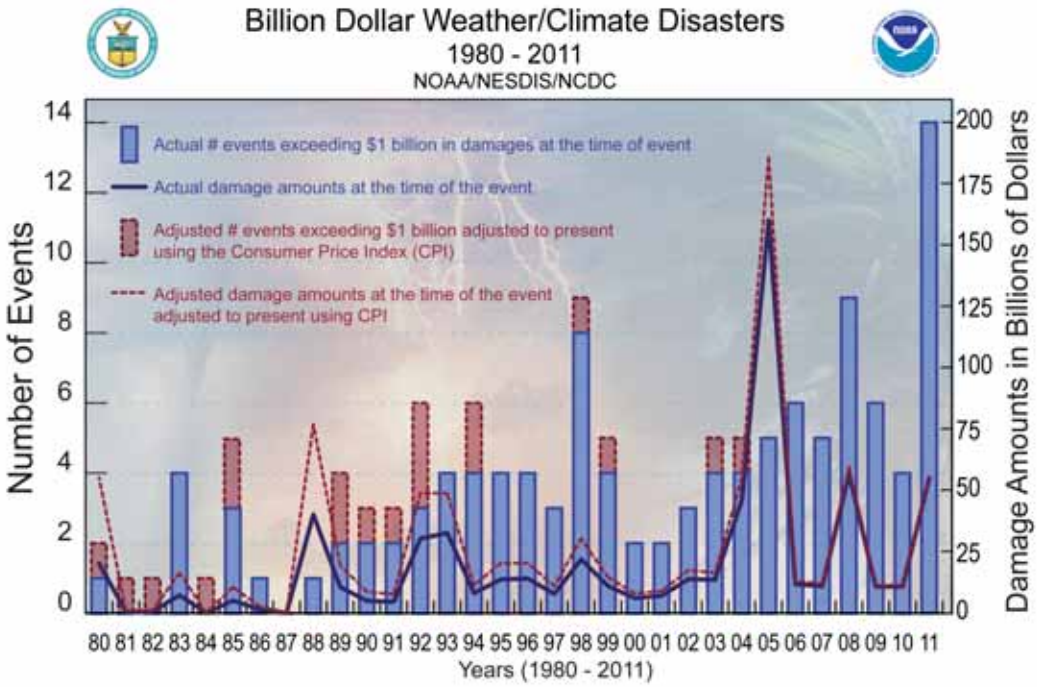

Fig. 1.2: Time series graph showing number of events and dollar costs by year.

Furthermore, according to NatCatSERVICE [46], the number of disasters in Europe has been showing an upward trend since 1980, largely due to the continuous increase of meteorological and hydrological events. Figure 1.1 depicts this information.

NatCatSERVICE, which includes events below the threshold used in EM-DAT, estimates overall losses of more than EUR 195 billion and insured losses of more than EUR 60 billion.

This situation not only affects Europe, but also the rest of the planet. The US National Oceanic and Atmospheric Administration (NOAA [47]) describes in [38] the events that have had the greatest economic impact since 1980. Figures 1.2 and 1.3 summarize this information for the period 1980-2011.

1.2 The Case of Forest Fires

As stated in [41], forest fires are a recurrent phenomenon in Europe and on other continents. Fires are a natural disturbance, which are essential for the regeneration of certain tree species and ecosystem dynamics. In addition, fire has been used in the environmental context for many purposes, including shrub removal in the forest and straw burning in agriculture.

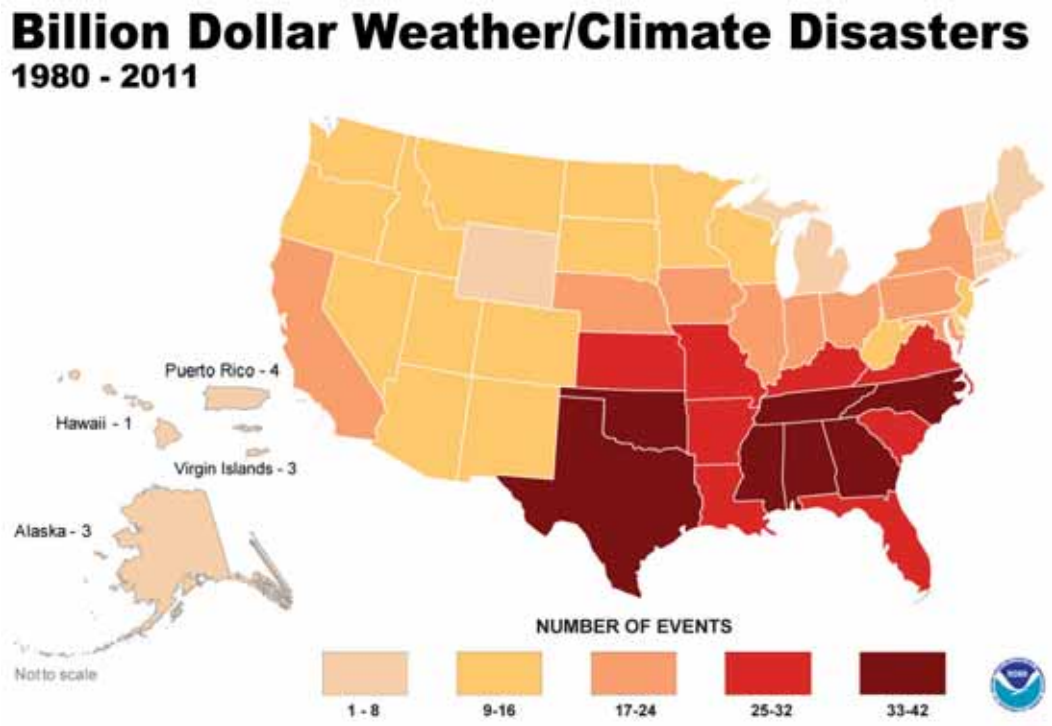

Fig. 1.3: US map showing spatial distribution of events by state.

However, fire events show increased intensity and produce more serious impacts. The reasons for this change are manifold, such as fundamental changes in land-use and demography. In the Mediterranean Region, for instance, the abandonment of traditional forest management practices and the suppression of fires for decades led to an accumulation of fuels in the forests, leading to more intense fires. Once these fires are ignited under high fire danger conditions that facilitate fire spread, they cannot be stopped. Despite the significant number of fire fighting resources used to extinguish them, large fire episodes lasting several days occurred recently in Portugal (2003, 2005), Spain (2006) and Greece (2007). Although fires also take place in central and northern Europe, the conditions of fire spread in these areas are often mild, allowing for fire control and extinction without major damages.

Over 95% of the fires are caused by humans, either deliberately (e.g. malicious arson) or by negligence/accident. Important causes of this latter category include escaped fires associated with agricultural practices, such as straw or shrub burning, forest debris burning or pasture renewals, engine sparks, careless use of recreational fires, electric power lines, railways and cigarettes.

Despite the human causality, fire ignition and the subsequent spread are mainly driven by the presence of fuels and the meteorological conditions that determine the dryness of the fuels. Fire ignition and spread are both enhanced by cumulated drought, high temperature, low relative humidity and the presence of wind. These factors are the essential components for the computation of what are known as fire danger indices, used to predict the degree of danger of fire occurrence and spread [15].

The European Forest Fire Information System (EFFIS) [23] provides detailed analysis of fire events for the European region. According to this data, an average of 70000 fires take place every year, burning more than half a million hectares of the forested areas in Europe. In critical years, e.g. 2007, this figure can increase to one million hectares.

Fire activity and fire effects are concentrated in the European Mediterranean Region. About 70% of fires occur in this region, and they are responsible for 85% of the total burnt area of Europe. Although fire frequency shows three peaks during the year, i.e. winter fires in the mountain regions, spring fires related to agricultural practices, and summer fires closely related to high temperatures and summer drought, most fire damage occurs in the summer period, that is, during July, August and September.

Table 1.3 shows the number of fatalities in the EU Member States during the period 2000-2009. These official figures were extracted from the EFFIS reports produced by the relevant fire services in the member states. Figure 1.4 illustrates the trend of forest fires in 1998-2009 for the European Mediterranean Region.

Country	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Portugal	3	1	4	21	2	18	11	6	3	7
Spain	6	0		9	5	17		1		12
France	9	4		10	2	6			0	0
Italy	2	3	5	7	2	3	1	23	3	4
Greece	10	4	0	1	2	0		80		0
Total	**30**	**12**	**9**	**48**	**13**	**44**	**12**	**110**	**6**	**23**

Tab. 1.3: Number of fatalities caused by forest fires in EU Member States (source: EFFIS).

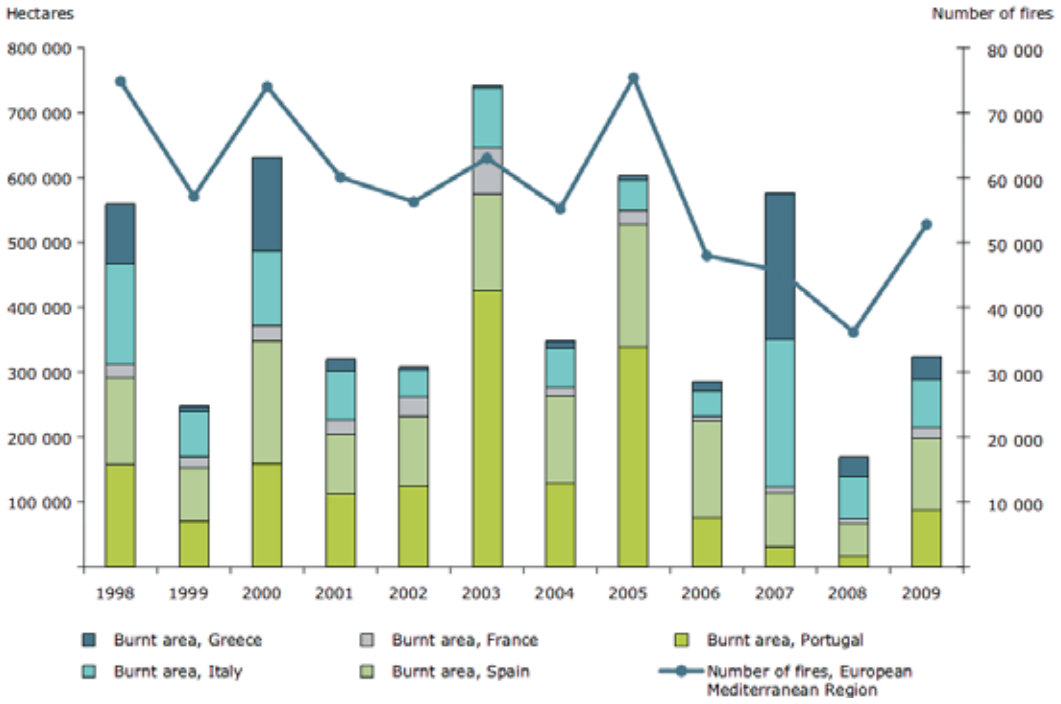

Fig. 1.4: Number of fires and burnt area in southern Europe (source: EFFIS).

1.2.1 Forest Fire Impacts

The major damage caused by forest fires is the loss of human life. The analysis of human casualties and fire accidents is very complex. However, recently a study of major fire accidents has been published under the umbrella of EFFIS [60]. According to its authors, fire entrapment is the major threat posed by a forest fire, and this is usually a result of interaction between human behavior and fire behavior, both of which require detailed attention and understanding.

Economic losses due to forest fires are difficult to quantify in a harmonised manner for the entire European territory. If we use a value of EUR 3 000 per ha, derived through extensive consultation with stakeholders in the field [30], to estimate the economic loss due to forest fires, the average loss is about EUR 1.5 billion every year. Considering the fact that additional indirect damage to local economies and the loss of human lives and property are not taken into account, this is a conservative estimate.

In addition to the impacts referred to above, extreme fire events produce substantial emissions [7] that have harmful effects on populations in nearby cities and villages. In the case of large fire events, such as the ones in Portugal in 2003 and Greece 2007, these emissions constituted a significant percentage of the total CO_2 emissions in these countries.

Over 3.5 million ha of forest areas were burnt by forest fires in Europe during the period of 2003 to 2009, affecting natural areas and ecosystems. These fires, which occurred mainly in the Mediterranean Region, led to land degradation and desertification processes. Impacts on ecosystems, such as the Natura 2000 areas, are reported and evaluated yearly by EFFIS [55]. Over half a million ha of these protected ecosystems were burnt in the reporting period. The highest impacts were recorded in Portugal in 2003 and 2005, with nearly 150000 ha burnt, and in 2007, with over 100 000 ha burnt in Greece, Italy and Spain. Impacts of forest fires on ecosystems are widespread. High intensity fires remove the existing vegetation cover and leave bare ground exposed to further processes of soil erosion, or even landslides. In areas where the fire return period is short (i.e. fires occur frequently on the same site) or vegetation regeneration is hampered by the lack of precipitation, fires may lead to desertification processes.

This data demonstrates the great importance of forest fire as a disturbing phenomena for human life and turns it into an important matter and a suitable case to focus on in this work.

1.3 Natural Hazards Management

Generally speaking, disasters normally occur when hazards meet vulnerability [67], and the potential for a hazard to become a disaster mainly depends on a society's capacity to address the underlying risk factors, to reduce the vulnerability of a community and to be ready to respond in case of emergency. The matter of adequately managing these kinds of catastrophes and minimizing casualties and other losses in crisis situations is an issue which Civil Protection Agencies (CP) are in charge of.

In the specific case of forest fires, many countries around the world have had to consider how best to organise the resources they have for fighting wildfires. Aerial assets -aeroplanes, helicopters and the associated equipment and supporting infrastructure- present a particular challenge as they are only in active use for part of any given year, are often specialised in design and tend to be costly to purchase and to operate. Sharing such resources within a country and between countries is a logical way of reconciling the twin objectives of providing an aerial means of attack against wildfires and prudent financial management.

In the European Union, during the 1980s and 1990s, there was some exchange of expertise on firefighting, but little in the way of formal cooperation. The Community Civil Protection Mechanism (now known as the European Union Civil Protection Mechanism) was established in 2001 and further strengthened in 2007. It provided a new capacity for coordination for Europe. It now plays a central role in the EU forest fire risk prevention and forest firefighting coordination at the EU level. There are currently 31 countries participating in the Mechanism - the 27 Member States of the European Union (EU) together with Iceland, Liechtenstein, Norway and Croatia. The Mechanism, which is managed by the European Commission, has tools to cope with wildfires in three phases of the disaster management cycle. The main responsibilities and tools allocated to the European Commission are outlined here under the headings of monitoring and prevention, preparedness and response.

Disaster risk reduction and management in Europe has shifted from a response-oriented approach towards an Integrated Risk Management (IRM) approach that includes prevention, preparedness, response and recovery. Measures addressing the reduction of risks have better ensured the safety of the population, infrastructure and the environment; for example, in the case of avalanches, where IRM has already reached an advanced level and incorporates technical measures developed and implemented over the last five decades. Nevertheless, more effort is needed to implement an integrated risk management approach throughout Europe that would address all hazards, and an important aspect in this sense is the implementation of suitable *Decision Support* tools.

For these kinds of decision support tools to be effective, it is important to be able to assess in advance how the present constraints at the time of attending to an ongoing emergency will affect the results obtained from them, both in terms of quality (accuracy) obtained and time needed to obtain a decision. This will be the main focus of this thesis.

1.4 Computational Science and High Performance Computing

Traditionally, scientific research has been categorised in the following areas: theoretical and observational or experimental science. However, it is often not feasible or too expensive, dangerous or immoral to study certain scientific phenomena using the second approach. In searching for an effective way to better understand the world around us, over the last 30 years, the area of computational science has revolutionised the way science can be applied to solve large and complex problems. It is now widely accepted and a crucial third pillar in tackling scientific investigation and engineering design [56].

Computational science - often also referred to as Computational Science and Engineering (CS&E) or *Scientific Computing* - is the application of computational and numerical techniques to science. Our wish to gain understanding of real-world processes such as weather and climate, air flow around a plane or the design and control of vehicles, combined with the domain expertise from physics, biology, medicine and others, leads to mathematical algorithms and models. These can be implemented in a computer understandable language and, as a result, we will be able to run simulations of real-world occurrences instead of preparing time-consuming and cost-intensive experiments.

Knowing this, Computational Science can be described as a methodology that studies various phenomena by means of modelling and simulation. It is an interdisciplinary field at the intersection of three domains: mathematics, computer science, and social and natural sciences and engineering as shown in Figure 1.5. Computational Science focuses on the appropriate use of a computational architecture to apply an algorithm to solve a scientific problem [56]. By integrating knowledge from the three mentioned domains, Computational Science deals with the development of problem-solving numerical algorithms and robust scientific tools. In doing so, scientific computing combines domain expertise, mathematical modelling, numerical analysis, algorithm development, software implementation, program execution, validation and the visualisation of results.

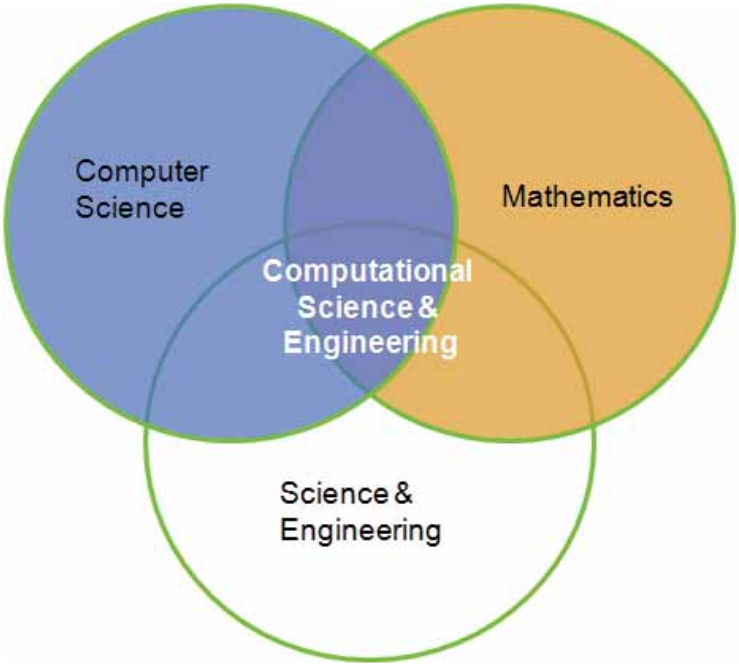

Fig. 1.5: Representation of Computational Science and Engineering as discipline merging the fields of Computer Science, Mathematics, Engineering and Sciences.

Computational science is typically applied in the following problem domains [64]:

- Numerical simulations to reconstruct and understand known events or to predict future or unobserved situations.

- Model fitting and data analysis to appropriately tune models or solve equations to reflect observations, subject to model constraints.

- Computational optimisation to optimise known scenarios.

Fundamental problems in science, especially those having a high impact on society, politics, and economics, are also referred to as "Grand Challenge" problems, e.g. genome sequencing, global climate modelling, speech and language studies, pharmaceutical design and many others. For the most part, these can only be solved by applying computational science and are typically executed on supercomputers or distributed computing platforms. The emergence of computational science would not have been possible without the power of modern computers and the advance of high performance computing resources. These allow for the massive required amounts of calculations and demonstrate that computational science greatly profits from the improvements achieved in computer hardware.

But beyond all advances in computer architecture, it is of paramount importance to provide computational algorithms and methods, which are able to use the available infrastructure most efficiently to solve real-world problems. This is why on the software side new sophisticated algorithms and concepts have been developed, too. Parallel computation paradigms and tools are essential for discovering, understanding, and designing solutions to these "Grand Challenge" problems. Further progress in computer architecture and applications will be necessary to satisfy the demand for ever higher levels of detail in scientific models and simulations.

This thesis is concerned with the grand challenge of forest fire spread prediction and the design and development, based on the afore-mentioned CS&E principles, of a methodology for time response and quality prediction assessment, which can complement and enhance Decision Support Systems for end-users, such as Civil Protection Agencies or Fire Brigades. More details can be found in the following Section 1.5, which describes the overall contribution of this work.

1.4.1 Urgent High Performance Computing

The topic of this thesis is framed within a real need which involves serious consequences. It is obvious that natural hazard management entails the need to make urgent and crucial decisions. On the other hand, most of these decisions, as in the case we are dealing with, are based on solutions provided by CS&E/HPC techniques, which require important computational capabilities. So, in the field of natural hazard management, we need the support of *Urgent High Performance Computing* solutions.

In many cases, there is no other choice but to rely on distributed-computing solutions, such as Grid or Cloud Computing [13, 69]. In addition, it is well-known that the availability and time response in such environments may become an important drawback for our needs.

Therefore, Urgent Computing mechanisms are necessary to properly tackle the natural hazard management problem.

As P. Beckman states in [8], for some simulations, insights gained through supercomputer computation have an immediate application. Consider, for example, an HPC application that could quickly calculate the exact location and magnitude of tsunamis immediately after an

undersea earthquake. Since the evacuation of local residents is both costly and potentially dangerous, promptly beginning an orderly evacuation in only those areas directly threatened could save lives. In connection to the specific topic of forest fire spread prediction, imagine a parallel wildfire simulation that coupled weather, terrain, and fuel models and could accurately predict the path of a wildfire days in advance. Firefighters could cut firebreaks exactly where they would be most effective. For these urgent computations, late results are useless results. As the HPC community builds increasingly realistic models, applications are emerging that need on-demand computation. Looking into the future, we might imagine event-driven and data-driven HPC applications running on-demand to predict everything from where to look for a lost boater after a storm to tracking a toxic plume after an industrial or transportation accident.

It is straightforward to imagine building a supercomputer specifically for these emerging urgent computations. Even if such a system led the Top 500 list, however, it would not be as powerful as the combined computational might of the world's five largest computers. Aggregating the country's largest resources to solve a critical, national-scale computational challenge could provide an order of magnitude more powerful than attempting to rely on a prebuilt system for on-demand computation.

Furthermore, costly public infrastructure, idle except during an emergency, is inefficient. A better approach, when practical, is to temporarily use public resources during times of crisis. For example, rather than building a nationwide set of radio towers and transmitters to disseminate emergency information, the government could require that large TV and radio stations to participate in the Emergency Alert System. When public broadcasts are needed, most often in the form of localized severe weather, broadcasts would be automatically interrupted, and critical information is shared with the public.

Nowadays, many research efforts are oriented towards exploring and implementing Urgent Computing solutions, which could be suitable, depending on each specific case. Some examples are [43, 5, 33, 40].

As detailed above, a framework for urgent computing must manage users, resources, elevated priority policies, and sessions. In this sense, SPRUCE (Special PRiority and Urgent Computing Environment) [9, 57] is an outstanding prototype framework for urgent computing. The primary responsibility of the framework is to authorize certain users to access certain resources at an elevated priority during an urgent computing event.

One of the primary design goals of SPRUCE is that during an urgent computing event, the ability to submit elevated priority jobs should be straightforward, efficient and easily transferrable [58]. To that end, SPRUCE uses simple right-of-way tokens as an authorization mechanism.

1.5 Contribution

The benefits of this work are aimed at providing a valuable help to the Civil Protection Agencies mentioned in Section 1.3. As stated above, much effort is needed to implement

integrated risk management approaches that would address all hazards, and the case of forest fires is not an exception. Although the work described in this thesis has been directly applied to the forest fire case, it can also be easily extrapolated to any other natural catastrophe.

In previous works [10, 20, 51], a two-stage prediction scheme (which will be detailed in Chapter 2.3.2) was introduced to enhance classical spread prediction results by enabling input parameters calibration. This prediction framework highly improves the quality of the predictions in forest fire spread simulations. However, it consists of a complex schema which introduces several adjustment techniques. Depending on the specific case we are dealing with in a certain moment, it could be very hard to know, or even to estimate, how much time will be necessary to spend on such processes, as well as the ideal amount and type of computational resources to be used.

The contribution of this work is concerned with this issue. Formally speaking, our main goal would be stated as follows:

"*The establishment of a methodology for the quality and response time assessment of environmental emergency evolution prediction under real-time restrictions*"

This, as will be discussed, implies carrying out a proper characterization of the whole prediction system in order to be able to determine, before running the prediction scheme, which configuration of the systems (both Hardware and Software) best fulfills the existing constraints, in terms of both the response time and the quality of the prediction.

This objective entails a very ambitious project. In the current *state-of-art*, there exist some works which deal with this matter in the sense of exploiting many (and distributed) computing infrastructures so as to be able to meet strict deadlines when an emergency takes place. An outstanding example is the VENUS-C project [59].

However, to the author's knowledge, there is no other work which considers this problem from the specific point of view of assessing the trade-off between quality and urgency, taking into consideration of the existing limitations regarding time response, computational resources availability, and quality requirements. Obviously, the correct application of the proposed methodology will lead us to many advantages in the complementation of decision support tools.

In the following chapters, the details concerning the methodology design, development, validation and experimentation will be given. The next section provides the reader with an outline of how this work is organized.

1.6 Organization

This work is composed of six chapters:

- **Chapter 1. Introduction:** The present chapter, which introduces this research work.

- **Chapter 2. Decision Support Systems for Forest Fires:** Gives a state-of-the-art background about all the issues the problem we are dealing with encompasses. From existing fire simulators to current DSSs, their details are listed, as well as the existing necessities which motivate this work.

- **Chapter 3. Methodology for Prediction Scheme Characterization:** In this chapter, we describe the proposed methodology to characterize each element of the proposed prediction strategy, with the aim of assessing, in advance, the efficiency of our prediction method regarding the quality of the final prediction, given certain time restrictions. Furthermore, a technical description of the tools that have been used is given.

- **Chapter 4. Methodology Validation:** The conducted validation of each step of the proposed methodology is presented in this chapter.

- **Chapter 5. Methodology Applicability. Experimental Evaluation.:** This chapter presents all the issues related to the experimental part of this research work, from the description of the application used as a study case to the results obtained, detailing how the theoretical background is applied in practice, as well as what the set of experiments consisted of. Implementation details of the experiments are also given.

- **Chapter 6. Conclusions and Future Work:** Presents the main conclusions inferred from carrying out this research work, as well as the research lines that remain open and, therefore, on which future work will be based.

2 Decision Support Systems for Forest Fires

The prediction of forest fire propagation is a key issue in order to minimize forest fire effects and avoid human fatalities. During a real emergency, control centres require reliable predictions about forest fire propagation to take adequate measures to fight against them.

These predictions are based on physical or empirical models that describe fire propagation considering the actual conditions during the emergency. These models must be solved by analytical or numerical methods and such solutions must be implemented in some simulation tool in such a way that the prediction of fire behaviour can be provided much faster than real time. These simulation tools can be integrated in Decision Support Systems to help control centre coordinators react in a real emergency.

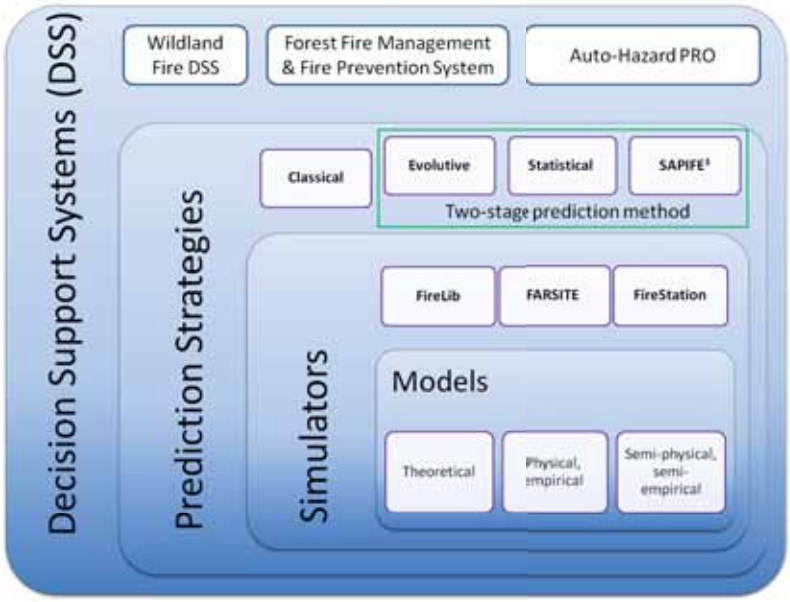

Fig. 2.1: Hierarchical relationship between components of the computational-based tools of a Decision Support System for forest fires management.

Forest fire-related decision-making processes encompass different conceptual phases based on different criteria. Such decisions are usually made taking different tactical, strategic, and even political aspects into account.

Regarding the computational-based tools which most current DSSs include, they present a hierarchical relationship between their components as the one depicted in Figure 2.1. In this chapter, an overview of the *state-of-art* of each one of these components is given, and the present weaknesses and necessities of the current DSSs are discussed in conclusion.

2.1 Fire Models

Fire models are useful for all aspects and during all phases of fire management, e.g. fire break analysis, real-time forecasting, fire fighter training, and illustration of fire behaviour to the general public. Although, in theory, models of limitless complexity can be constructed, which are capable of representing arbitrary wildland fire events, model parameterisation becomes more and more difficult with increased complexity. Especially for real-time spread prediction, computational efficiency is at a premium and, in that case, it becomes computationally prohibitive to simulate fire at the highest levels of complexity.

Fire spread models have a long history; the first models date back to the late 1940s. Many research projects around the world since then have sought a way to simulate fire behaviour using site-specific data. They attempt to describe and translate various environmental conditions into equations and inequalities to characterise the spatial and temporal evolution of fires. The most significant output parameters are rate of spread, fire line intensity, and fuel consumption.

Surface fire spread modelling cannot yet be considered as definitely resolved and remains an ongoing active research field. In spite of the abundant number of existing models, only a few of them are successfully used in practical applications. Since fire spread models evolved from a research to an operational tool, their use is now constantly growing.

Subsequently, an overview of existing surface fire spread modelling approaches and the well-established Rothermel model is presented as well.

Physical Models

Also referred to as theoretical or analytical, physics-based models attempt to represent both the physics and the chemistry of fire. This class of models predicts fire spread based on the physics of combustion. Physical principles, such as conservation of energy, mass and momentum, are used to derive a formula for the rate of spread and other quantities of interest. Several physical models are available, but generally require large amounts of detailed input data and computing power and they are of limited operational application.

Empirical Models

Also called statistical models, these models are based on observation and experiment and do not contain a physical basis. Fire data from laboratories, field-based experiments, or historical wildland fire studies is analysed and put into statistical correlations. These models are only applicable to systems in which conditions are similar to those used in formulating and testing the models. As mentioned in [49], they function fairly reliably in low wind, at terrain situations.

Furthermore, there exist hybrids, namely semi-empirical models, which close the gap between the two mentioned modelling techniques and try to combine the advantages of both physical and empirical modelling. The most popular representative of this class is the Rothermel model.

The Rothermel Wildland Fire Spread Model

The fire spread model developed by Richard C. Rothermel in 1972 [52] for surface fires is among the most widely used fire prediction models. It is based on physical laws and enhanced with empirical factors. Therefore, a clear classification into one of the previously mentioned modelling categories becomes difficult. Rothermel's model can be rather seen as a hybrid between the physical and statistical modelling approach. It is thus also referred to as physical-statistical or semi-empirical model.

The Rothermel model consists of a non-linear system of equations and is based on the principle of the conservation of energy. The 17 input parameters are grouped into four main categories: fuel type, fuel moisture, topography and weather. According to this grouping, the totality of required input parameters is composed of fuel bed depth, fuel load, surface-area-to-volume ratio, heat content, mineral content, silica content, particle density, percentage of dead fuel, moisture of extinction, moisture content of live fuel, moisture content of dead fuel after 1, 10 and 100 hours, aspect, slope, wind speed at half-flame height, and wind direction. The majority of these parameters are relatively easily observable, which might be one of the main reasons for the model's popularity. The main output parameters are the rate of spread, direction of maximum spread, and effective wind speed. Output parameters of minor importance include fire line intensity, heat release per unit area, and flame length.

The Rothermel model is possibly still the most popular fire spread model and time could not impair its functionality as explained in [62]. "What it lacks in complexity, it makes up in reliability and ease of use". It is implemented by the majority of simulation systems.

2.2 Forest Fire Simulators

In the field of physical systems modeling, specifically forest fire behavior modeling, there exist several fire propagation simulators, based on some of the above mentioned models, whose main objective is to try to predict the fire evolution. In the following subsections, an overview of the main features of the ones used in this research work is given.

2.2.1 FireLib

FireLib [26] is a simple, yet efficient open source library developed in C implementing the Behave algorithm. In fact, it is a library that encapsulates the fire behaviour algorithm BEHAVE [6], but it is modified for interactive applications that operate with cell-based growth simulations. It applies the Rothermel spread model [52, 53] and the simulation

technique used is cellular automata. FireLib is able to predict the rate of spread (ROS), spread intensity, flame length, and scorch height of free-burning surface fires in two dimensions.

To calculate ROS, every cell during a running simulation gets labelled with the time at which the fire arrives at the middle of the cell. The simulation system is intended for programmers and scientists who are in need of a highly optimized application programming interface to develop or investigate fire behaviour growth simulators [28]. The free availability of source code and its pure core functionality without additional modules were the main reasons why this simulation system was chosen for some of the experimental work in this research.

A compact C library eased the programming of interfaces for the integration of parameter calibration techniques, though the core simulator was treated as a black box and left unchanged.

FireLib has no GIS support included. The terrain, including slope and aspect, has to be defined by the user and can be spatially heterogeneous. The 13 standard fuel models are accepted and the possibility to define different, e.g. non-propagating, fuel types is given. During a running simulation, the simulator iterates over each of the defined cells of the terrain. At every cell from which fire propagation is theoretically possible (sufficient fire intensity, unburnt neighbouring cells left, etc.), fire behaviour is calculated for all neighbouring cells using a pipeline pattern, explained in the FirreLib manual [26]).

2.2.2 FireStation

Firestation [37] is a software system aimed at the simulation of fire spread over complex topography. The software implements a semi-empirical model for fire rate of spread, which takes as input local terrain slope, parameters describing fuel properties, as well as the wind speed and direction. Fire shape is described with recourse to an ellipsetype model. The whole system is developed under a graphical interface (see Figure 2.2), aiming at an easier use and output readability so as to facilitate its application under operational conditions.

FireStation falls into the BEHAVE [6] fire prediction model [52, 53] coupled with a gridcell approach. In addition, FireStation includes two different models for the simulation of the wind field. Both of these models predict wind velocity and direction based on local observations from meteorological stations. This constitutes a major contribution for fire simulation quality. Another important feature of FireStation is the use of the system for the analysis of fire danger indexes on a broad spatial scale.

FireStation software was developed within the environment of the CAD application Microstation, from Bentley Company. The decision to develope the software within this environment was made on the grounds that Microstation provides a user-friendly interface and developing tools that proved to fulfill the needs for the development of the system. The underlying software is written in MDL, a specific C language of Microstation that has built-in subroutines for the design of window-based interfaces, generation of visualization elements in the 3D space, on top of the usual mathematical capabilities of the C language.

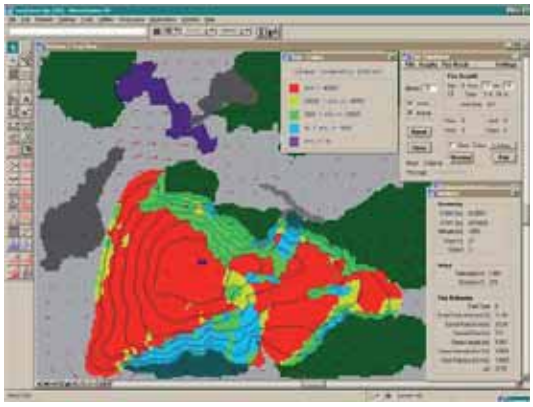

Fig. 2.2: Screenshot of FireStation software after fire growth simulation

The wind models are self-contained Fortran codes, which run as external programs. The graphical environment of Microstation is three-dimensional. Thus the visualization process can employ, not only the normal top-view, but any other view perspectives as well, for map display and other visualization procedures. This is very important for interpretation purposes, since different visualization angles may provide a much clearer and more precise interpretation of the data.

The use of FireStation in supporting fuel management decisions allowed for the definition of critical areas subjected to potential extreme fire behavior, and, in that way, the optimization of resource/treatment allocation in a given area.

2.2.3 FARSITE

In [27], FARSITE (Fire Area Simulator) is described as a 2-dimensional fire behaviour and fire growth simulator, which incorporates both spatial and temporal information on topography, fuels, and weather. It was developed by Finney in the 1990s [25] and has been updated and enhanced many times since then. FARSITE has been divulged worldwide, is very well established and is frequently used by biologists, land managers and ecologists, in addition to fire managers and fire fighters. There exists a version, which runs with a graphical user interface on the Windows operating system, and another command-line version for UNIX-based operating systems. FARSITE is also based on the Rothermel spread model, but implements wave propagation as a simulation technique.

Besides standard surface fires, phenomena such as crown fires, spotting, and post-frontal combustion can be simulated. Additionally, modules for aerial and ground suppression actions and the calculation of fuel moisture are included. Further special features include the support of single or multi-point ignition scenarios, the simulation of partial sections of the fire front, and the user specification of spatial and temporal resolution of the computations.

Initially, FARSITE was created for complete and comparable analyses of different fire scenarios in planning and long-range projection of active prescribed fires. Simulations are started for long-range weather scenarios to simulate fire gaming and asking multiple what-if questions and to suggest possible outcomes of fire growth over several weeks. FARSITE is

now also applied more and more to short-range predictions of large wildfires to support strategic firefighting decisions.

FARSITE is a deterministic modelling system, i.e. simulation outcomes have a direct connection to input values. The latter are required by the fire simulator in the form of geographic information system (GIS) data in either raster or vector form to obtain the necessary spatial landscape and fuel information. Weather data are generally provided as a stream or a table of values over time. The 13 standard fire behaviour fuel models are supported and users can again define custom fuels. The integration of real landscapes allows prevention and extinction strategies to be designed in a localised way [21]. The produced real-time on-screen graphics (fire perimeter etc.) are compatible with PC and workstation applications and GIS software and portable for later analysis and display.

Although GIS files provide accurate input information, inexperienced users have to spend significant time carrying out the digital cartography before working with FARSITE systematically. Moreover, users should be familiar with fuels, weather, wildfire situations and the associated terminology before making fire and land management decisions based on simulation outcomes. In [21] Pastor et al. claim that FARSITE has not been thoroughly validated. It is, therefore, very diffcult to detect the origin of inaccuracies, which may be due to data input or to the underlying mathematical modelling. Finally, FARSITE is not totally suited for studying very large forest fires because it lacks a dynamic wind model on complex landscapes and achieves only poor precision in crown fire models. However, FARSITE can be considered one of the most useful simulation systems for both forest fire prevention and extinction decision-making.

2.3 Prediction Strategies

The prediction of forest fire propagation usually relies on a method consisting of the use of a simulator, which may be any one of the types described above. This section describes the *classical* way of predicting the forest fire spread, as well as the *two-stage* prediction scheme, which was introduced to enhance classical spread prediction results by enabling input parameters calibration.

2.3.1 Classical Prediction

The traditional way of predicting forest fire behaviour, summarised in Figure 2.3, takes the initial state of the fire front as input as well as the input parameters given for that time instant. These values are entered into any existing fire simulator, which then returns the prediction for the state of the fire front at a later time instant.

Depending on the complexity of the chosen simulation software as well as the size and resolution of the region affected by fire, the classical prediction consumes comparatively reasonable computing resources. By comparing the simulation results with the advanced real fire at the same instant, the prediction error may be assessed.

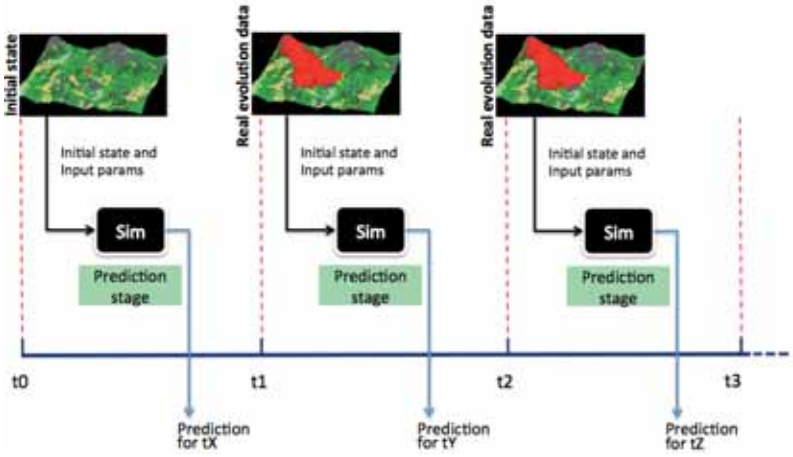

Fig. 2.3: Classical prediction schema

It has to be noted that the forecasted fire front tends to differ from the real fire line to a greater or lesser extent. As the prediction error accumulates gradually as the prediction time advances, deviations between real phenomenon behaviour and forecasted fire spread become even more significant.

One reason for this incidence is that the classic calculation of the simulated fire is based upon one single set of input parameters afflicted with many inadequacies, as will be explained in the following subsection. To improve parameter quality and enable real-time estimation and calibration of model input parameters in each time step during an ongoing prediction, a two-stage prediction scheme was proposed by Abdalhaq in [2] and is presented next.

2.3.2 Two-stage Prediction Method

The simulators presented in the previous section need certain input data, which define the characteristics of the environment where the fire is taking place in order to evaluate its future propagation. This data usually consists of the current fire front, terrain topography, vegetation type, and meteorological data such as humidity, wind direction and wind speed. Some of this data could be retrieved in advance and with notable accuracy, such as, for example, the topography of the area and the predominant vegetation types.

However, there is some data that turns out to be very difficult to reliably obtain. For instance, getting an accurate fire perimeter is very complicated because of the difficulties involved in getting, in real time, images or data about this matter. Both live and dead fuel moistures are examples of data which cannot be retrieved with reliability at the moment of the emergency. Another kind of data sensitive to imprecisions is that of meteorological data, which is often distorted by the fire itself. However, this circumstance is not only related to forest fires, but also happens in any system with a dynamic state evolution over time (e.g. floods [39], thunderstorms [3, 61], etc.). These restrictions concerning uncertainty in the input parameters, added to the fact that these inputs are set up only at the very beginning of the simulation process, become an important drawback, because, as the simulation time goes on, variables previously initialized could change dramatically, misleading simulation results. In order to overcome these restrictions, we need a system capable of properly estimating the

values of the input parameters needed by the underlying simulator so that the results we obtain correspond to reality.

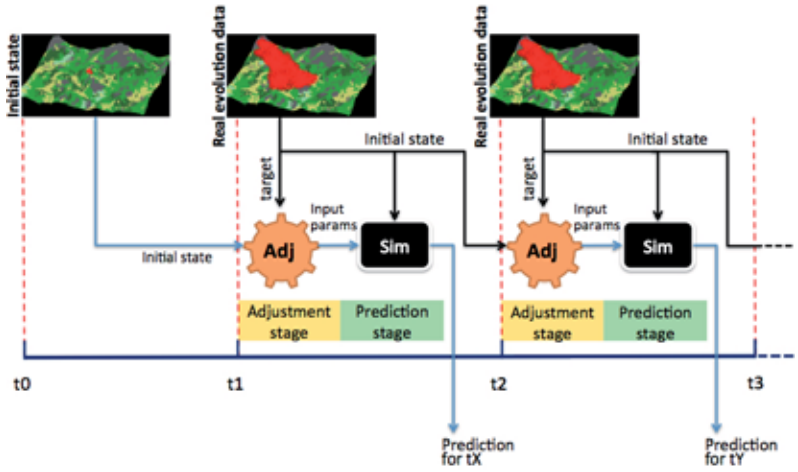

Fig. 2.4: Two-stage Prediction Method

The classic way of predicting forest fire behavior, which is summarized in Figure 2.3, takes the initial state of the fire front as input, as well as the input parameters given for a certain time instant. The simulator then returns the fire spread prediction for a later time instant.

Comparing the simulation result with the advanced real state, the forecasted fire front tends to differ to a greater or lesser extent from the real fire line. One reason for this behavior is that the classic calculation of the simulated fire is based upon one single set of input parameters afflicted with the previously explained insufficiencies. To overcome this drawback, a simulator independent data-driven prediction scheme was proposed to optimize model input parameters [2]. Introducing a previous calibration step as shown in Figure 2.4, the set of input parameters is optimized before every prediction step. The proposed solution comes from reversing the problem: how to find a parameter configuration such that, given this configuration as input, the fire simulator would produce predictions that match the actual fire behavior. Having detected the simulator input that best describes current environmental conditions, the same set of parameters, could also be used to best describe the immediate future, assuming that meteorological conditions remain constant during the next prediction interval. Then, the prediction becomes the result of a series of automatically adjusted input configurations.

Previous works have proposed several calibration techniques, which made the problem of fire spread prediction to fit the *Dynamic Data-Driven Application Systems* (DDDAS) paradigm [18, 19], rather than the classic prediction scheme such as [11, 51, 20]. Since the two-stage method for forest fire spread prediction described in Figure 2.4 constitutes a simulator-independent prediction method, the same technique could be extrapolated to any kind of natural disaster by only exchanging the underlying simulator. Figure 2.5 shows a general scheme for a two-stage prediction method for natural hazard management. In the following section, we shall describe a methodology to perform the prediction time assessment under this prediction framework.

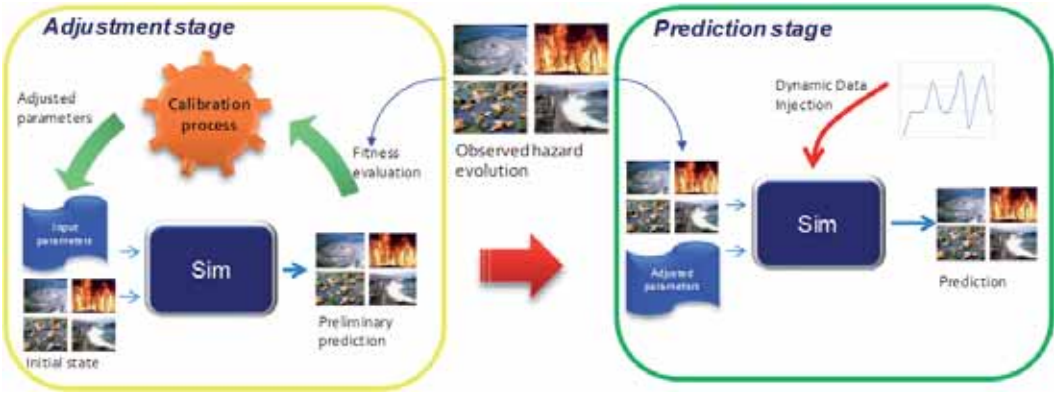

Fig. 2.5: General two-stage DDDAS for natural hazard prediction evolution

2.4 Decision Support Systems

Over the past thirty years, the development of Decision Support Systems to fight against Forest Fires has been a recurrent research and engineering topic. In this section, we describe some prominent examples. As will be seen, the purposes and scopes of these systems may differ from one to another. In Section 2.5, we shall discuss the main shortcomings that these systems currently present, which constitute the key point of this thesis.

2.4.1 US Wildland Fire Decision Support System (WFDSS)

Developed by the USDA Forest Service, WFDSS is the most noticeable Decision Support System for wildland fire management in the United States [48].

As a web-based system, WFDSS provides risk-informed decision logic and display capacity at multiple management levels. The components of risk analysis include access to spatial data, fire modeling and computing resources, and economic valuation. Detailed risk assessments from WFDSS are readily accessible online to field users, analysts, and policy and decision makers at local, regional, and national levels. The process is linear, scalable, and responsive to changes in the fire situation (weather, values at risk, and fire fighting capacity, etc.).

The design of WFDSS improved upon existing documentation systems and made the wildland fire decision process accessible, consistent, flexible, and geospatial. Accessibility to thousands of users is ensured through a web-based system, where authorized users require only an internet connection and logon for access to the system, alleviating the need for updating desktop programs and acquiring large datasets.

Incident information, analyses, and text are entered while outputs produced in WFDSS improve the ability of fire managers to quickly focus on pertinent information. Moreover, maps and other spatial information of values, assets, and the fire environment also contribute to the synthesis of information (see Figure 2.6).

When a wildland fire starts, a cyclical process of assessment, risk-characterization, analysis, and deliberation begins in order to make a risk-informed decision. During the initial fire

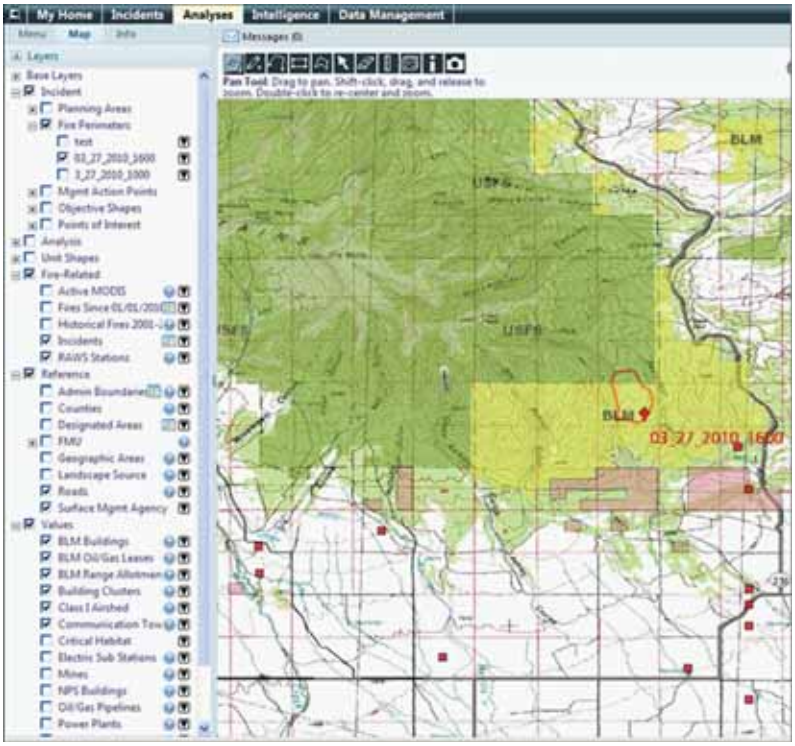

Fig. 2.6: Geospatial data provided by WFDSS including information about fire location and size, reference information, and values (source: [48]).

response, fire dispatchers and managers use WFDSS to assess the basic fire situation with a variety of assessment tools. Current weather observations, fire danger, and fuel moisture are retrieved from weather stations and US National Weather Service forecasts. Models and tools for analyzing fire behavior, economic, and air quality impacts are included in WFDSS. Fire behavior modeling systems are used to determine fire size probabilities, make fire progression forecasts, and predict fire behavior characteristics such as rate of spread, crown or surface fire occurrence, and fire intensity.

In conclusion, the developers of WFDSS assert that all these features turn it into an unique DSS when compared to all previous fire decision analysis and documentation systems in the United States.

2.4.2 FOMFIS

FOMFIS is an acronym for FOrest Fire Management and FIre prevention System [14]. It was a two-year project partially funded by the European Comission, completed early in 1999.

The project aimed to define, design and implement a computer based system that would give support to the planning process of the activities and resources distribution for the preventive operations belonging to the forest fire defence services.

The main goal of the FOMFIS project was to integrate a set of technological solutions using the same information system platform, thus allowing forest fire service personnel to accom-

plish their off-line planning duties, mainly pre-suppression activities, quickly, accurately and cost effectively. The FOMFIS system was conceived and built as a modular system running under the same user interface integrating remote sensing, statistical analysis, stochastic generation, knowledgebased simulation systems, simulation models and spatial analysis tools.

It deals with several areas of forest fire research, namely:

- Forest fuel mapping.
- Socio-economic risk analysis.
- Forest fire behaviour and fire fighting simulation.
- Probabilistic planning.

Despite the fact that it never advanced to an operational DSS owing to, among other reasons, its demanding software and hardware requirements, these unique and innovative characteristics make it worth mentioning in this section.

The FOMFIS prototype is an off-line system. Its main focus is to allow fire managers to best determine the level of resources they need, their allocation, and other management actions that must be performed in order to achieve a desirable fire protection level within a prespecified budget.

2.4.3 Auto-Hazard Pro

The Automated Fire and Flood Hazard Protection System (Auto-Hazard Pro) [32], is one of the most recent projects related to DSS for Forest Fires. Specifically, it is a DSS for forest fire protection in the Euro-Mediterranean region that includes weather data management, a geographical data viewer, a priori danger forecasting and fire propagation modeling, automatic fire detection, and optimal resource dispatching.

Funded by the European Union, this research project has been designed to improve the level of technological development in forest fire risk management in Europe and help authorities take appropriate action to protect both man-made and natural environments. Its main objective was to integrate real-time and on-line wildland fire hazard management approaches into a geographic information system (GIS) platform. The specific evolution reported in this article is the development of a decision support system (DSS) that incorporates proactive planning, weather data management, a geographical data viewer, a priori risk forecasting and fire propagation modeling, automatic fire detection, optimal resource dispatching governed by the pertinent principles, and emergency management of real-time fire episodes.

The AHP DSS is divided into five main modules that represent different kinds of information to the user and perform different tasks, providing support during the whole fire management process. These modules are: the fire weather module, the fire detection module, the fire danger rating module, the fire propagation simulation module, and the resource dispatching

module. The final application provides capabilities for the simultaneous visualization of different pieces of information (fire danger indices, available resources, and active fires), fire alarm information management, and resource information management.

As regards the fire propagation module, it allows the user of this DSS to estimate the growth of a fire in a fixed amount of time and under a set of customized meteorological and other fire environmental factors. A "wizard" interface guides the user through the steps in which the necessary information for the simulation is defined. Within this, a computer model called "Fire Spread Engine" (FSE) is used, and it estimates the fire front expansion on surface forest fuels, using spatial data about topography, moisture content, wind vector field, and fuel type. The FSE code is an improved version of a code that was used in the previously mentioned FOMFIS system.

By including these features, the authors and developers report positive responses from the operational users who have tested the system, emphasizing some key points, such as:

- A relatively simple and user-friendly interface.
- A manageable level of complexity.
- Usefulness of the functionalities (pre-incident analysis, detection, and dispatching support).
- Efficient communication of alerts (SMS, etc.) according to predefined rules.

2.5 Decision Support Systems: Weaknesses and Necessities

Despite the huge research work invested over the past years in the development of suitable Decision Support Systems, there are still many issues which have to be tackled for these kinds of systems to be made fully operational.

In [68], an overview of the state of the art concerning DSS for forest fires, as well as the existing problems in this sense, is explained. The authors claim that one of the main reasons is that very few users in a DSS scenario are equally capable or are equally interested in "Decision", "Support" and "System". These three terms establish three different focuses of interest: "Decision" is related to non-technical functional and analytical aspects of DSS and to criteria for selecting applications; "Support" focuses on implementation and understanding of the way real people (forest managers) operate and how to help them; "System" directly emphasizes skills of design and development technology. In addition, during the last years/decades, the three components of the term have lost balance to the detriment of "Decision" and "Support" in favour of technology. So, there exists a noticeable gap between the users involved in the development and use of DSSs.

This fact becomes clear when examining certain kinds of data and procedures. For instance, Figure 2.7 depicts the flow of actions that are carried out in the case of the afore-mentioned US Wildland Fire Decision Support System. As can be seen, questions related to fire behavior modeling are considered as *Optional decision content*.

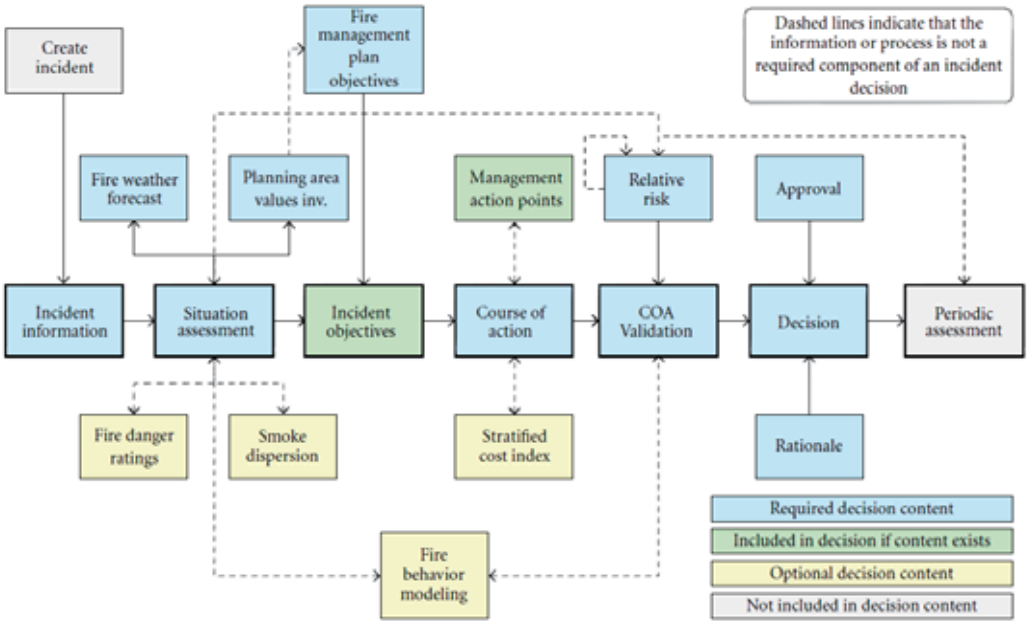

Fig. 2.7: WFDSS conceptual model. The highlighted horizontal flow depicts the major phases necessary to make a decision in WFDSS (source: [48]).

Furthermore, in [68] the results of a questionnaire given to 18 different participants of an international workshop on "Improving Dispatch for forest fires", held in Greece, are also reported. The results obtained for the question "*The dispatcher, in regard to forest fires, makes dispatching decisions with the help of (select all that apply)*" stand out. These results can be observed in Figure 2.8. As can be seen, the use of modern systems for decision support that take advantage of scientific advances (GIS, computerized DSS, fire simulation...) is uncommon, especially when compared to the dispatcher's own experience (16 out of 18 replies).

This *lack of confidence* in computational solutions at the time of making decisions during a forest fire is often related to the lack of flexibility that DSSs provide to dispatchers, in spite of the immense opportunities the new technological advances could bring. Usually, the spread simulation depends on a single system/simulator, whose performance, in terms of response time and the quality of the results, is fixed and commonly unknown beforehand.

Many of these systems present heavy data needs and software and hardware requirements which make them impractical for adoption by operationals, since the decisions to be made have to be transparent and communicated in a compressed time frame.

Moreover, in the case of various alternatives, regarding the specific settings of the DSS and their eventual impact in those terms, the final user of such systems may be unaware of the benefits they could offer.

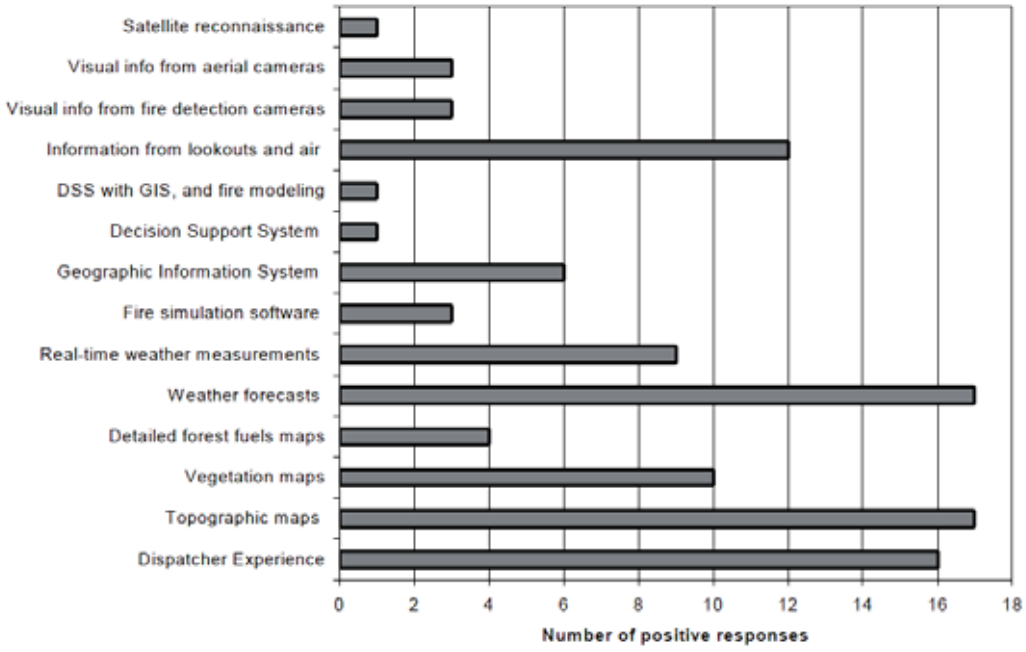

Fig. 2.8: Sources of information in support of dispatching decisions (source: [68]).

The main goal of the methodology developed in this research work is to deal directly with this problem, so that this gap between the DSS implementation and the final users (decision dispatchers) is at least partially filled. This goal is fulfilled by giving the capability to assess in advance how the present constraints at the time of attending an ongoing forest fire will affect the results obtained from them, both in terms of the quality (accuracy) obtained, and the time needed to obtain a decision, and therefore, to be able to select the most suitable configuration of both the prediction strategy and computational resources to be used.

Therefore, the framework derived from the application of this methodology is not supposed to be a new DSS, but rather a tool from which most of forest fire (and other kinds of natural hazards) DSSs could benefit notably.

The following section details the developed methodology for the prediction strategy characterization, assuming the use of the two-stage prediction method described in Section 2.3.2.

3 Methodology for Prediction Scheme Characterization

As stated in Chapter 1, when a natural emergency takes place, the accuracy of the prediction and time response become crucial. As has been studied in the previous chapter many research efforts are currently oriented towards designing integrated platforms, which rely on High Performance Computing solutions, to help deal with this kind of phenomena. These *computational-aided* environments combine physical models, simulators, powerful computational resources and data acquisition techniques in order to assist the personnel in charge of making the appropriate decisions to fight against the ongoing disaster.

In order to be useful, any evolution prediction of an ongoing hazard must be delivered as fast as possible in order not to be outdated. Consequently, we come up with the dicotomy *urgency-accuracy*. In the area of emergency management, these two concepts are closely related. A quick response for an ongoing emergency obviously depends on the computational power of the available resources (the faster they are, the sooner a prediction may be given). In addition, when relying on the two-stage prediction method, accuracy also depends on the amount of resources we have access to perform different simulations, since the adjustment strategies for parameter optimization may be carried out in a parallel way, taking advantage of the simultaneous execution of different simulations.

In this chapter, we describe the proposed methodology to characterize each element of the prediction process proposed in Chapter 2.3.2, with the aim of assessing, in advance, the efficiency of our prediction method regarding the quality of the final prediction, given certain time restrictions. As we will see, the application of this methodology will allow us to optimize the use of the available resources at the moment of attending to an ongoing emergency with time constraints and computational resource restrictions.

3.1 Problem Assumptions and Requirements

This research work is oriented towards the development of an intelligent system which allows important needs to be faced in the context of forest fire spread prediction. From the point of view of attending to an ongoing emergency, it is necessary to be able to assess in advance how many computational resources are needed, at least, to attend it. This is due to the existing strict deadlines for giving a response. Moreover, it is also necessary to accurately assess the quality that simulators will give us, since this kind of emergency may threaten urban areas and even human lives.

This constitutes a very ambitious project, so in order to bound the problem, we work under certain assumptions:

- We consider only scenarios in which the computational resources are dedicated. Currently, we are working on adapting tools that allow urgent execution of tasks in distributed-computing environments, e.g. SPRUCE [9].

- We rely on the two-stage prediction method for natural hazard management, described in Chapter 2.3.2, in which the adjustment strategies work in an iterative way.

In the two-stage prediction strategy explained in Chapter 2.3.2, the adjustment process plays the main role. Previous have studies demonstrated that the quality of the prediction is directly correlated to the quality obtained at the end of the adjustment process [20, 11]. Thus, it is absolutely necessary to have a good characterization of the adjustment process in order to be able to evaluate the adjustment quality we can reach under certain conditions.

As has been previously mentioned, we focus on iterative adjustment techniques, which lead us to the desired solution progressively, i.e. the more iterations we are able to perform, the better the solution we will find. Obviously, this fact has a direct impact on the time incurred in the adjustment process. Nevertheless, on the other hand, it has to be taken into account that typically, at each iteration, a new set of scenarios is generated, and each of them has to be run and subsequently evaluated. Therefore, if we are able to run them in a parallel way, it may also allow us to save computational time.

In summary, in order to reach a good trade-off between quality and urgency, one must consider three main interrelated issues:

- The quality of the prediction is directly related to the quality of the adjustment, and the quality of the adjustment depends on how many times the adjustment technique is iterated, as well as on the number of scenarios tested per iteration.

- The amount of computing resources determines the amount of simulations that can be simultaneously executed per iteration at the adjustment stage.

- The response time in providing a prediction is a critical point and seriously limits the number of iterations that can be executed at the adjustment stage.

In a real emergency, the response time in providing the result of the prediction is fixed by the decision control center. In the same way, the quality of the prediction is also a parameter fixed by the decision control center. Under these constraints, it is necessary to determine the computing resources required to fulfill the time and quality constraints. This leads to the necessity of deploying a way to set up in advance:

- The prediction scheme settings, in particular, the adjustment policy's specific parameters, for a required quality of the prediction. This is especially relevant when the ongoing hazard may threaten urban areas and even human lives.

- The computational resources needed to deliver a required quality of the prediction, given certain time constraints.

To accomplish this goal, it is necessary to characterize the adjustment strategy in such a way that it is possible to determine beforehand the number of iterations and the number of scenarios per iteration that should be executed to ensure a certain prediction quality. Since each scenario implies one execution of the simulation kernel, it is also necessary to characterize this simulation kernel to estimate the time required to run each simulation.

Having characterized the adjustment technique and the simulation kernel, then it is possible to determine the necessary computing resources to execute a certain number of iterations with a certain number of simulations per iteration, with each simulation having an estimated execution time. Thus, our methodology allows us to determine the required computing resources to reach a certain prediction quality in a given time.

In order to properly tackle this objective, the details of the proposed methodology for the characterization of the two-stage prediction method are given in the subsequent section, where the manner in which we face these challenges as well as the principles we based our strategy on to tackle them are summarized. Furthermore, the details concerning each step of the proposed framework for the whole prediction process characterization are reported.

3.2 Two-stage Prediction Method Characterization

Based on the assumptions and requirements detailed above, we have designed a methodology to characterize the two-stage prediction process. It is worth mentioning that, despite the fact that this work is focused on the specific case of forest fires, an implicit requirement is to be able to extend the proposed solutions to any other kind of natural hazard, respecting the premises previously described. So, for this methodology to be as flexible as possible, we have to independently analyze the behavior of the afore-mentioned processes in terms of the main variables we must deal with: the time spent and the quality of the results. Table 3.1 summarizes the dependencies for each case.

	Simulation	**Adjustment**	**Final Prediction**
Time	-Dependent on inputs -Dependent on computational resources	-Dependent on simulator time -Dependent on computational resources -Dependent on the configuration of the adjustment method	-Simulator time
Quality	-Dependent on inputs	-Dependent on adjustment time	-Dependent on adjustment quality and real-time eventualities

Tab. 3.1: Dependencies between each factor belonging to the 2-stage prediction framework.

As one can see, there exists a series of dependencies from the prediction quality to the simulation time.

As regards the time needed for the final prediction process, it consists of a single simulation of the winning input setting at the adjustment stage, and the quality of this simulation is directly correlated to the quality obtained at the end of the adjustment process [11].

Since the developed adjustment methods are all iterative, the quality obtained at the end of the adjustment stage presents a single dependence: the available time to perform it. Regarding the necessary time, this process presents a couple of dependencies: obviously, the time incurred in each simulation will determine the overall adjustment time, but it also depends on the specific configuration of the adjustment method itself, as in many cases this configuration can take greater advantage of the available resources, since adjustment strategies may be run in a parallel way. Thus, the time incurred in the adjustment process will also depend on the available computational resources.

Finally, each simulation, in terms of time needed, is dependent on the input parameters that describe the scenario being simulated and the underlying computational resources where it is run. The quality of each simulation will depend on the input parameters, since it is determined by the similarity between the simulation produced using those input parameters and the actual evolution of the fire.

Therefore, the characterization must be done from the simulation process to the prediction process, so that by characterizing the time incurred in the simulation process in terms of its inputs and the computational resources, we can reach the final prediction assessment.

3.2.1 Genetic Algorithm as Adjustment Technique

In this particular work, we focus on Genetic Algorithms (GA) as a suitable adjustment technique, since it has been proved to be a very powerful technique [2, 20, 11]. As will be studied in this chapter, GA provide very good solutions in an acceptable amount of time.

By its own nature, GA represents a technique whose quality of results directly depends on the times it is iterated as well as its specific configuration settings, and also allows to be benefited from HPC techniques, since it can take great advantage of parallel computation. For these reasons, the methodology discussed in this thesis has been applied using Genetic Algorithms as an adjustment technique for the two-stage prediction method.

Specifically, the individuals used in the case of forest fire spread prediction are defined as a sequence of different genes, namely: wind speed and wind direction, moisture content of the live fuel, moisture content of the dead fuel (at three different times), and type of vegetation (out of the 13 standard Northern Forest Fire Laboratory fuel models [4]). Topographic data is assumed to be constant and invariable, so it is not considered in the evolutive process.

Details concerning how Genetic Algorithms work is given in the subsequent section, but basically, GA uses three operations to obtain the consecutive generations: selection, crossover and mutation. Selection operation selects good quality parents to create children that will inherit their parents' good characteristics (by crossover operation). In order to guarantee natural diversity of individual characteristics, mutation phenomenon can occur for each

child characteristic (under a very slight probability). Selection can include elitism where the best j individuals (j > 0) are included in the new generation directly.

Since simulated fires can be represented as a grid of cells map, indicating which cells were burned as a consequence of the simulated fire, the quality of a specific individual is determined by means of the fitness function expressed by Equation 3.1. This equation calculates the differences in the number of cells burned, both missing or in excess, between the simulated and the real fire. Formally, this formula corresponds to the *symmetric difference* between the actual spread and the simulated spread, divided by the actual spread, so as to express a proportion. $\cup Cell$ is the union of the number of cells burned in the real fire and the cells burned in the simulation, $\cap Cell$ is the intersection between the number of cells burned in the real fire and in the simulation, $RCell$ are the cells burned in the real fire and $ICell$ are the cells burned at the starting time.

$$E = \frac{(\cup Cell - ICell) - (\cap Cell - ICell)}{RCell - ICell} \tag{3.1}$$

Subsequently, the details of Genetic Algorithms are given.

3.2.2 Genetic Algorithms: Theoretical Basis

Concisely stated, a genetic algorithm (or GA for short) is a programming technique that mimics biological evolution as a problem-solving strategy [42, 66]. Given a specific problem to solve, the input to the GA is a set of potential solutions to that problem, encoded in some fashion, and a metric called a fitness function that allows each candidate to be quantitatively evaluated. These candidates may be solutions already known to work, with the aim of the GA being to improve them, but more often they are generated at random.

The GA then evaluates each candidate according to the fitness function. In a pool of randomly generated candidates, of course, most will not work at all, and these will be deleted. However, purely by chance, a few may hold promise - they may show activity, even if only weak and imperfect activity, toward solving the problem.

These promising candidates are kept and allowed to reproduce. Multiple copies are made of them, but the copies are not perfect; random changes are introduced during the copying process. These digital offspring then go on to the next generation, forming a new pool of candidate solutions, and are subjected to a second round of fitness evaluation. Those candidate solutions which were worsened, or made no better, by the changes to their code are again deleted; but again, purely by chance, the random variations introduced into the population may have improved some individuals, making them into better, more complete or more efficient solutions to the problem at hand. Again these winning individuals are selected and copied over into the next generation with random changes, and the process repeats. The expectation is that the average fitness of the population will increase each round, and so by repeating this process for several rounds (some problems may need on the order of hundreds or thousands), very good solutions to the problem can be discovered.

Genetic algorithms have proven to be an enormously powerful and successful problem-solving strategy, dramatically demonstrating the power of evolutionary principles. Genetic algorithms have been used in a wide variety of fields to evolve solutions to problems as difficult as or more difficult than those faced by human designers. Moreover, the solutions they come up with are often more efficient, more elegant, or more complex than anything comparable to what a human engineer would produce.

Methods of representation

Before a genetic algorithm can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form that a computer can process. One common approach is to encode solutions as binary strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution. Another, similar approach is to encode solutions as arrays of integers or decimal numbers, with each position again representing some particular aspect of the solution. This approach allows for greater precision and complexity than the comparatively restricted method of using binary numbers only and often "is intuitively closer to the problem space" [29].

As outstanding applications of this technique, is worth noting the work of Steffen Schulze-Kremer, who wrote a genetic algorithm to predict the three-dimensional structure of a protein based on the sequence of amino acids that go into it [45]. Schulze-Kremer's GA used real-valued numbers to represent the so-called "torsion angles" between the peptide bonds that connect amino acids. (A protein is made up of a sequence of basic building blocks called amino acids, which are joined together like the links in a chain. Once all the amino acids are linked, the protein folds up into a complex three-dimensional shape based on which amino acids attract each other and which ones repel each other. The shape of a protein determines its function.) Genetic algorithms for training neural networks often use this method of encoding also.

A third approach is to represent individuals in a GA as strings of letters, where each letter again stands for a specific aspect of the solution. One example of this technique is Hiroaki Kitano's "grammatical encoding" approach, where a GA was put to the task of evolving a simple set of rules called a context-free grammar that was in turn used to generate neural networks for a variety of problems [45].

In our specific context, individuals are represented by arrays of decimal numbers, where each number corresponds to the value of a specific variable (mostly environmental variables).

The virtue of all three of these methods is that they make it easy to define operators that cause the random changes in the selected candidates: flip a 0 to a 1 or vice versa, add or subtract from the value of a number by a randomly chosen amount, or change one letter to another.

Methods of selection

There are many different techniques which a genetic algorithm can use to select the individuals to be copied over into the next generation, but listed below are some of the most common methods. Some of these methods are mutually exclusive, but others can be, and often are, used in combination.

Elitist selection: The fittest members of each generation are guaranteed to be selected. (Most GAs do not use pure elitism, but instead use a modified form where the single best, or a few of the best, individuals from each generation are copied into the next generation just in case nothing better turns up.)

Fitness-proportionate selection: Fitter individuals are more likely, but not certain, to be selected.

Roulette-wheel selection: A form of fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitors' fitness. (Conceptually, this can be represented as a game of roulette - each individual gets a slice of the wheel, but fitter ones get larger slices than less fit ones. The wheel is then spun, and whichever individual "owns" the section on which it lands each time is chosen.)

Scaling selection: As the average fitness of the population increases, the strength of the selective pressure also increases and the fitness function becomes more discriminating. This method can be helpful in making the best selection later on when all individuals have relatively high fitness and only small differences in fitness distinguish them from each other.

Tournament selection: Subgroups of individuals are chosen from the larger population, and members of each subgroup compete against each other. Only one individual from each subgroup is chosen to reproduce.

Rank selection: Each individual in the population is assigned a numerical rank based on fitness, and selection is based on this ranking rather than absolute differences in fitness. The advantage of this method is that it can prevent very fit individuals from gaining dominance early at the expense of less fit ones, which would reduce the population's genetic diversity and might hinder attempts to find an acceptable solution.

Generational selection: The offspring of the individuals selected from each generation become the entire next generation. No individuals are retained between generations.

Steady-state selection: The offspring of the individuals selected from each generation go back into the pre-existing gene pool, replacing some of the less fit members of the previous generation. Some individuals are retained between generations.

Hierarchical selection: Individuals go through multiple rounds of selection each generation. Lower-level evaluations are faster and less discriminating, while those that survive to higher

levels are evaluated more rigorously. The advantage of this method is that it reduces overall computation time by using faster, less selective evaluation to weed out the majority of individuals that show little or no promise, and only subjecting those who survive this initial test to more rigorous and more computationally expensive fitness evaluation.

Methods of change

Once selection has chosen fit individuals, they must be randomly altered in hopes of improving their fitness for the next generation. There are two basic strategies to accomplish this. The first and simplest is called mutation. Just as mutation in living things changes one gene to another, so mutation in a genetic algorithm causes small alterations at single points in an individual's code.

The second method is called crossover, and entails choosing two individuals to swap segments of their code, producing artificial offspring that are combinations of their parents. This process is intended to simulate the analogous process of recombination that occurs to chromosomes during sexual reproduction. Common forms of crossover include single-point crossover, in which a point of exchange is set at a random location in the two individuals' genomes, and one individual contributes all its code from before that point and the other contributes all its code from after that point to produce an offspring, and uniform crossover, in which the value at any given location in the offspring's genome is either the value of one parent's genome at that location or the value of the other parent's genome at that location, chosen with 50/50 probability.

3.3 Methodology for Simulator Kernel Characterization

It has been demonstrated that the adjustment techniques detailed in Chapter 2.3.2 contribute to the improvement of the quality of the predictions. However, the time incurred in exploring the huge search space involved in that problem is worthwhile. This fact turns out to be a great disadvantage, taking into account that, in these kinds of urgent situations, a successful prediction is not only determined by the accuracy of the results: it is also necessary to seriously consider the time restrictions. When a natural catastrophe is taking place, it is necessary to make urgent decisions to effectively fight against it.

As stated above, the fact of having well characterized each simulator we deal with, in terms of execution time, becomes crucial in validating the proposed methodology. This matter may be tackled by means of taking the strategy of carrying out large sets of executions of the underlying simulator and then analyzing its behavior from the obtained results. However, this fact may not be trivial in certain cases. While it is easy to detect that the application presents a high sensitivity to certain input parameters, even in an intuitive way, some of them produce a behavior of the simulator that turns out to be hard to predict. Figures 3.1 and 3.2 show examples (using FireLib simulator) of each case, respectively. In the former, one can observe that the dimension of the map being simulated has a direct influence on the execution time (as was bound to happen), whereas, in the latter, it can be observed that the

relationship between execution time and wind direction is not so clear[1], and it becomes even odder when combining variations in wind direction with variations with vegetation type.

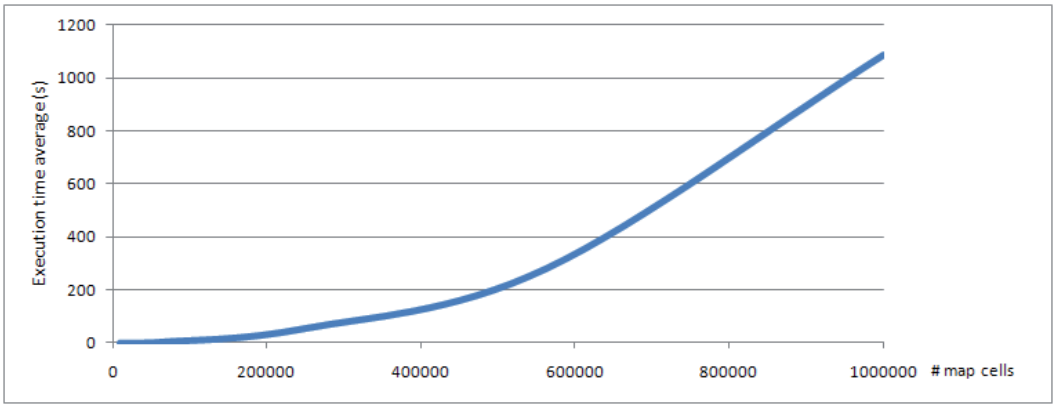

Fig. 3.1: Execution time as a function of the number of cells.

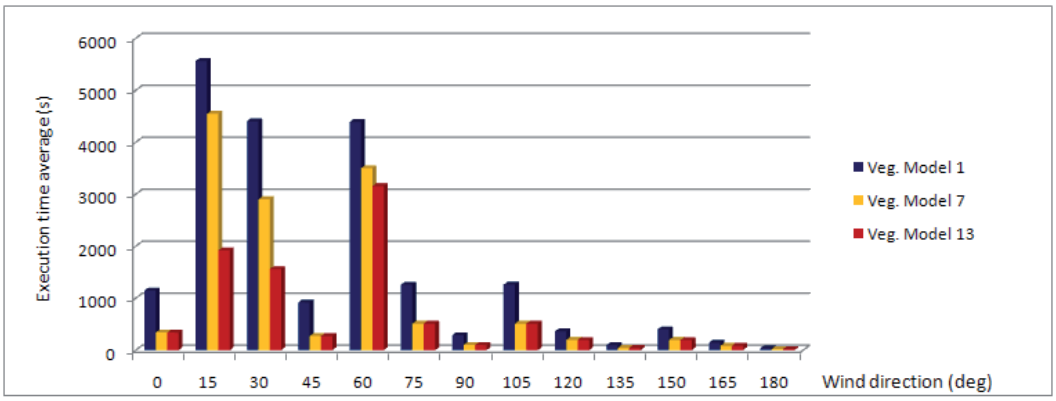

Fig. 3.2: Variations in execution time according to variations in wind direction and vegetation type.

This fact highlights the need to rely on complex criteria in order to successfully classify the input data sets according to the execution time they will cause.

The experimentation carried out for the specific case of forest fire simulation revealed that the execution time of a single simulation on the same map and simulating the same time can vary from seconds to several minutes or even hours. Such long simulations produce prohibitive times and should be not executed. So, the parameter configurations implying such long simulation times must be detected beforehand and must be discarded from the adjustment process. Therefore, it is necessary to apply some techniques that, given a set of input parameters, provide the expected simulation time in order to determine the amount of simulations that can be executed given certain real time constraints.

Consequently, we need to be able to anticipate how much execution time certain input settings will produce, without the necessity of running that simulation. For this purpose, what we propose is a solution which allows us to tackle the problem of, given a certain configuration of the scenario to be simulated, rapidly assessing whether we have enough resources

[1] this *anomaly* is reported in [26])

and available time to perform the simulation, regardless of whether the component(s) of the scenario are the ones which provoke the simulation to take longer or not.

A suitable way of facing these kinds of problems consists of applying different strategies, going from data-mining to machine learning approaches, to a knowledge database in order to draw right inferences.

To be operative during a real hazard, this execution-time estimation of a given scenario must be inferred as quickly as possible, keeping the cost of carrying out this operation to a minimum, in terms of time needed.

For this reason, we rely on the field of Artificial Intelligence to be able to automatically learn from stored knowledge, so as to provide *smart* decisions. In particular, we use Decision Trees to extract this knowledge from a certain database. In the next subsection, we describe how Decision Trees work and, subsequently, the different steps included in the proposed methodology are described.

3.3.1 Decision Trees: Theoretical Basis

Decision tree learning, used in statistics, data mining and machine learning, uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels [65].

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather, the resulting classification tree can be an input for decision making.

Decision tree learning is a method commonly used in data mining. The goal is to create a model that predicts the value of a target variable based on several input variables. Each interior node corresponds to one of the input variables; there are edges to children for each of the possible values of that input variable. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf.

A tree can be "learned" by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all have the same value of the target variable or when splitting no longer adds value to the predictions.

In data mining, decision trees can also be described as the combination of mathematical and computational techniques to aid the description, categorisation and generalisation of a given set of data. Data comes in records of the form:

$$(x, Y) = (x_1, x_1, x_1, \dots, x_k, Y)$$

The dependent variable, Y, is the target variable that we are trying to understand, classify or generalise. The vector x is composed of the input variables, x_1, x_2, x_3, etc., that are used for that task.

Decision trees used in data mining are of two main types:

- **Classification tree analysis** is when the predicted outcome is the class to which the data belongs.
- **Regression tree analysis** is when the predicted outcome can be considered a real number (e.g. the price of a house or a patient's length of stay in a hospital).

The term Classification And Regression Tree (CART) analysis is an umbrella term used to refer to both of the above procedures, first introduced by Breiman et al. in [12]. Trees used for regression and trees used for classification have some similarities - but also some differences, such as the procedure used to determine where to split.

There are many specific decision-tree algorithms. Notable ones are ID3 algorithm and C4.5 algorithm, which are subsequently described.

ID3 Algorithm

The ID3 (*Iterative Dichotomiser 3*) algorithm can be summarized as follows:

1. Take all unused attributes and count their entropy concerning test samples.
2. Choose attribute for which entropy is minimum (or, equivalently, information gain is maximum).
3. Make node containing that attribute.

The algorithm is as follows:

```
function ID3(Examples, Target-Attribute, Attributes)
   Create a root node for the tree
   if all examples are positive then
      return the single-node tree Root, with label equals to +
   end if
   if all examples are negative then
      return the single-node tree Root, with label equals to -
   end if
   if number of predicting attributes is empty then
      return the single node tree Root with label equals to the most
      common value of the target attribute in the examples.
   else
      A = The Attribute that best classifies examples
```

```
        Decision Tree attribute for Root = A
        for each possible value v_i of A do
           Add a new tree branch below Root, corresponding to the test A = v_i
           Examples[v_i] = the subset of examples that have the value v_i for A
           if Examples[v_i] is empty then
              below this new branch add a leaf node with label equals to
              the most common target value in the examples
           else
              below this new branch add the subtree ID3 (Examples[v_i], Target-Attribute,
              Attributes-A)
           end if
        end for
     end if
     return Root
  end function
```

This algorithm usually produces small trees, but it does not always produce the smallest possible tree. The optimization step makes use of information entropy:

$$E(S) = -\sum_{j=1}^{n} f_s(j) log_2(f_s(j))$$

Where:

- $E(S)$ is the information entropy of the set S.
- n is the number of different values of the attribute in S (entropy is computed for one chosen attribute)
- $f_s(j)$ is the frequency (proportion) of the value j in the set S

An entropy of 0 identifies a perfectly classified set. Entropy is used to determine which node to split next in the algorithm. The higher the entropy, the higher the potential to improve the classification here.

Gain is computed to estimate the gain produced by a split over an attribute:

$$G(S, A) = E(S) - \sum_{j=1}^{m} f_s(A_i) E(S_{A_i})$$

Where:

- $G(S, A)$ is the gain of the set S after a split over the A attribute.
- $E(S)$ is the information entropy of the set S.
- m is the number of different values of the attribute A in S.

- $f_s(A_i)$ is the frequency (proportion) of the items possessing A_i as value for A in S.
- A_i is the i^th possible value of A.
- S_{A_i} is a subset of S containing all items where the value of A is A_i.

Gain quantifies the entropy improvement by splitting over an attribute: higher is better.

C4.5 Algorithm

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set of already classified samples. Each sample is a vector where attributes or features of the sample are represented. The training data is augmented with a vector where the class to which each sample belongs is represented. At each node of the tree, C4.5 chooses one attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. Its criterion is the normalized information gain (difference in entropy) that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists. This algorithm has a few base cases.

All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class. None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class. Again, C4.5 creates a decision node higher up the tree using the expected value.

In pseudocode, the general algorithm for building decision trees is [34]:

1. Check for base cases
2. For each attribute *a* find the normalized information gain from splitting on *a*
3. Let *a-best* be the attribute with the highest normalized information gain
4. Create a decision node that splits on *a-best*
5. Recurse on the sublists obtained by splitting on *a-best*, and add those nodes as children of node

C4.5 made a number of improvements to ID3. Some of these are:

- Handling both continuous and discrete attributes - In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it [50].

- Handling training data with missing attribute values - C4.5 allows attribute values to be marked as ? for missing. Missing attribute values are simply not used in gain and entropy calculations.

- Handling attributes with differing costs.

- Pruning trees after creation - C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes.

As will be described in Chapter 4.1, in this work we use an open source Java implementation of the C4.5 algorithm in the Weka [31] data mining tool in order to estimate which time interval the duration of a certain simulation will belong to, given certain input parameters.

3.3.2 Kernel Characterization Methodology Step by Step

As has been previously stated, the simulator kernel characterization is fulfilled by means of carrying out large sets of executions (on the order of tens of thousands) counting on different initial scenarios (different input data sets), and then, applying knowledge-extraction techniques from the info they provide, i.e. we record the execution times from the experiment, and then we establish a classification of the input parameters according to the elapsed times they produced. Specifically, we follow the subsequent sequence of steps:

1. **Training database building:** Currently, we work with training databases composed of 12000 up to almost 40000 different scenarios. The distribution of each input parameter corresponds to the ones specified in Table 3.2. These probability distributions and their associated parameters, as regards wind speed and direction, were the ones used in [16]. The vegetation models correspond to the 13 standard Northern Forest Fire Laboratory (NFFL) fuel models [4].

2. **Determination of *execution time classes*:** The whole training database is executed, and every pair of scenario - execution time is recorded. After this, the histogram of execution times is analyzed. Identifying the local minimums of the histogram, the upper and lower boundaries of each *execution time class* are determined.

3. **Decision Tree building:** Once we have determined how many classes we will consider, then we are ready to build the Decision Tree. For this purpose, we rely on the aforementioned C4.5 algorithm, specifically, the J48 open source Java implemented in the Weka [31] data mining tool.

These steps are made *offline*, i.e. they are all already carried out at the moment of the fire occurrence. It is worth mentioning that this process must be done for each specific computational platform where we will run the simulations, and also for each topographic area we want to have characterized (those maps of interest, especially because of their historical fire recurrence), since these factors present a high impact on the duration of the simulations.

Input	Distribution	μ,σ	Min,Max
Vegetation model	Uniform	—	1,13
Wind Speed	Normal	12.83,6.25	—
Wind Direction	Normal	56.6,13.04	—
Dead fuel moisture	Uniform	—	0,1
Live fuel moisture	Uniform	—	0,4

Tab. 3.2: Input parameters distributions description.

Once this methodology has been followed, only a final step remains: the application of the built Decision Tree with the scenario describing the ongoing fire, in order 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).

3.3.3 Urgent approach with unknown resources appearance

The methodology for the simulator kernel characterization proposed above lead us to obtain very satisfactory results, as will be studied in Chapter 4.3 and 5.

However, a previous training process is needed, which, depending on the performance of the computational platform in question, may last a long time.

Based on the reasoning presented in Chapter 1.4.1, it is plausible to consider an approach consisting of, when possible, temporarily using public resources or resources lent by institutions/organizations during times of crisis.

In these cases, the fire would be in progress, so performing the whole training process would be prohibitive. A straightforward way, as yet untested, to benefit from these new resources would be to rely on the linear correlation between the execution times of an already characterized platform and the new available ones. This way, it would be necessary to execute a subset of the training database. Figures 3.3, 3.4 and 3.5 depict correlations between different computational platforms for a subset of 1500 simulations.

Technical descriptions of each platform are:

- DELL: Dell PowerEdge M710. Intel Xeon E5430, 2.66GHz, 2x6MB L2 cache memory, 16 GB RAM Fully Buffered DIMM 667 MHz.
- IBM: IBM x3650. Dual-Core Intel Xeon CPU 5150, 2.66GHz, 4MB L2 cache memory, 8 GB Fully Buffered DIMM 667 MHz.

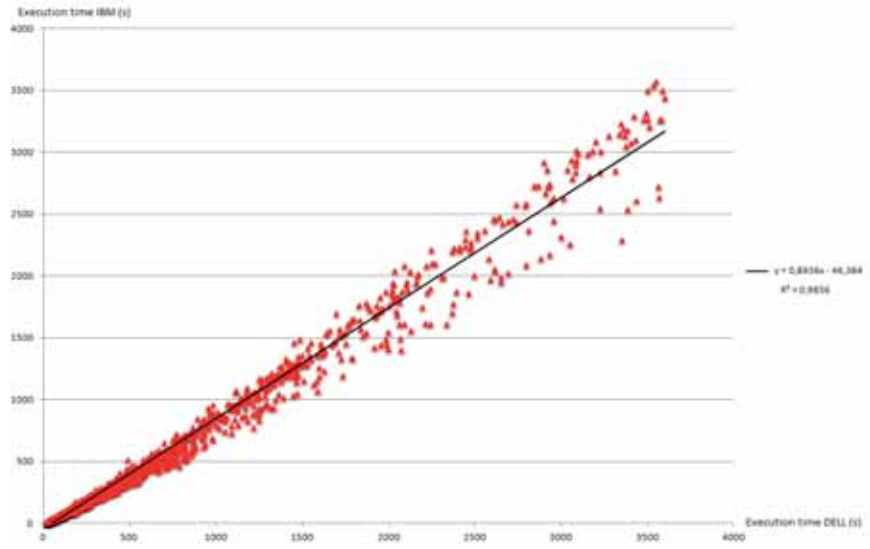

Fig. 3.3: Correlation between execution times running in DELL and IBM platforms.

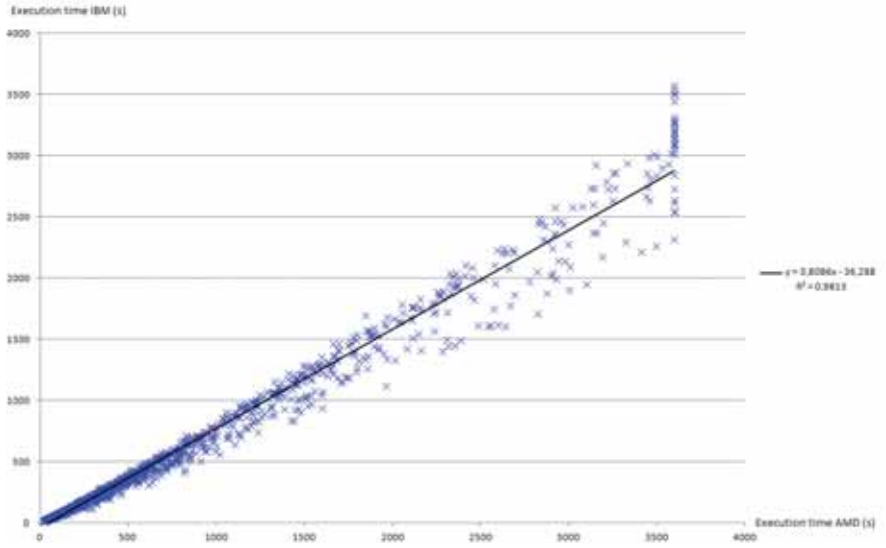

Fig. 3.4: Correlation between execution times running in AMD and IBM platforms.

- AMD: AMD Opteron 6174, 2.2GHz, 6MB L2 cache memory, 64GB Fully Buffered DIMM 667 MHz.

As can be observed, good linear correlations exist in all three cases, even comparing different computer architectures. However, it is easy to note that the longer lasting the simulations are, the worse the correlation becomes between different platforms and the execution times present. Therefore, it is necessary to carry out a deeper study before putting this aspect into practice.

Given such good linear correlation coefficients, it would be suitable to base our estimations on a linear regression so as to obtain the estimated simulation times for the new (and unknown) computational resources, and then extrapolate these times to the corresponding

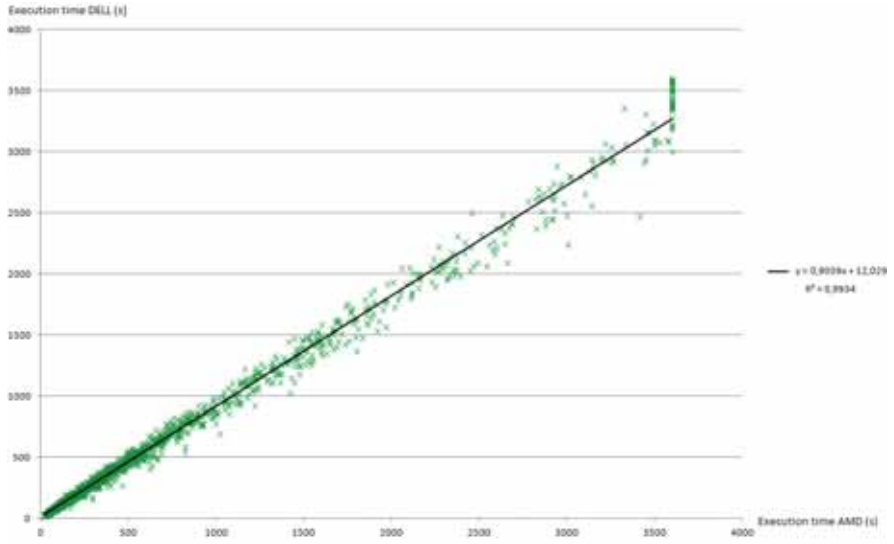

Fig. 3.5: Correlation between execution times running in AMD and DELL platforms.

classes of the decision trees. This is an interesting matter to deal with in future work, as will be discussed in Chapter 6.2.

3.4 Methodology for Genetic Algorithm Characterization

Accurate prediction of the evolution of natural hazards is a crucial point in helping the control centers in the decision process. In our prediction framework, it highly depends on the adjustment process performance. As stated in Section 3.2.1, in this work we focus on Genetic Algorithms (GA) as a suitable adjustment technique, since it has been proved to be a very powerful technique [20, 11].

The characterization of Genetic Algorithms as an adjustment technique for the 2-stage prediction method must allow us to properly estimate the adjustment quality we may obtain, given certain restrictions, both in terms of timing and resource availability when the adjustment stage is done.

Adjustment quality stands for the difference between the simulation result once the adjustment process is completely carried out and the real state of the hazard.

In general terms, this issue is addressed by means of performing a statistical analysis to determine, for each calibration strategy, those features that affect the quality of the results. Then, a study of the particular impact of these factors on the convergence of each method is done, to infer criteria in order to estimate the achievable quality of results under certain restrictions.

As presented in Section 3.2.2, GA works in an iterative way. It starts with an initial population of individuals which will be evolved through several iterations in order to guide them to

better search space areas. Operators such as *elitism*, *selection*, *crossover* and *mutation* are applied to every population to obtain a new one superior to than the previous one. Each individual from a population is ranked according to a predefined fitness function. The fitness function in the case of forest fire spread prediction is the difference between real and simulated fire spread. The iterative nature of GA leads to an eventually near-optimal solution in the adjustment stage after a certain number of GA iterations. For this reason, it is mandatory to analyze the GA convergence for the particular case of forest fire spread prediction, as well as to be able to extract a general characterization of its behaviour.

In this section, we will also present the study carried out to fulfill the need of being able to select in advance the best settings for the adjustment method, Genetic Algorithm (GA) in this case, given a certain prediction quality constraint. On the one hand, this technique allows us to obtain different degrees of quality in the solutions obtained, which allows us to be able to adapt to eventual restrictions (deadlines, available resources, etc.). On the other hand, this flexibility makes it harder to characterize in order to choose the correct configuration of the method for each case.

Parameters such as number of generations, individuals per population, elitism factor, mutation probability, and so on, affect the quality of the winner individual, i.e. the final solution we will deliver at the end of the adjustment process.

For the characterization of GA, we have carried out massive executions to obtain a proper statistical analysis. In the first instance, the characterization reported in this section was performed using FARSITE [25] as the fire spread simulator. This experiment uses the GIS data from the benchmark provided by FARSITE (the *Ashley project*, see Figure 3.6). Based on this benchmark, we set a reference fire with a duration five hours. The simulations carried out in this study take these first five hours of spread as the adjustment time interval, and every initial simulation setting (i.e. every individual in the GA) is configured according to the probability distributions and their associated parameters shown in Table 3.2 for each type of input parameter (i.e. each gene of each individual). As regards wind speed and wind direction, these probabilities correspond to the ones used in [16]. Vegetation models correspond to the 13 standard Northern Forest Fire Laboratory (NFFL) fuel models [4].

The final fire front of the reference fire is taken at the end of each simulation in order to calculate the difference between it and the simulated one. We call this difference *adjustment error*, and it is calculated by means of the formula exposed in Equation 3.1.

As regards the computational platform, all the experiments carried out in this work were done on a cluster of 8 x Dell PowerEdge M600 nodes, each of which counting on 2xQuad-Core Intel Xeon E5430, 2.66GHz, 2x6MB L2 cache memory (2x2) and 16 GB RAM Fully Buffered DIMMs 667MHz, running Linux version 2.6.16.

3.4.1 GA Convergence Analysis

In order to perform a proper statistical analysis, the study of the GA convergence has been carried out considering different settings, keeping track of the results obtained from the

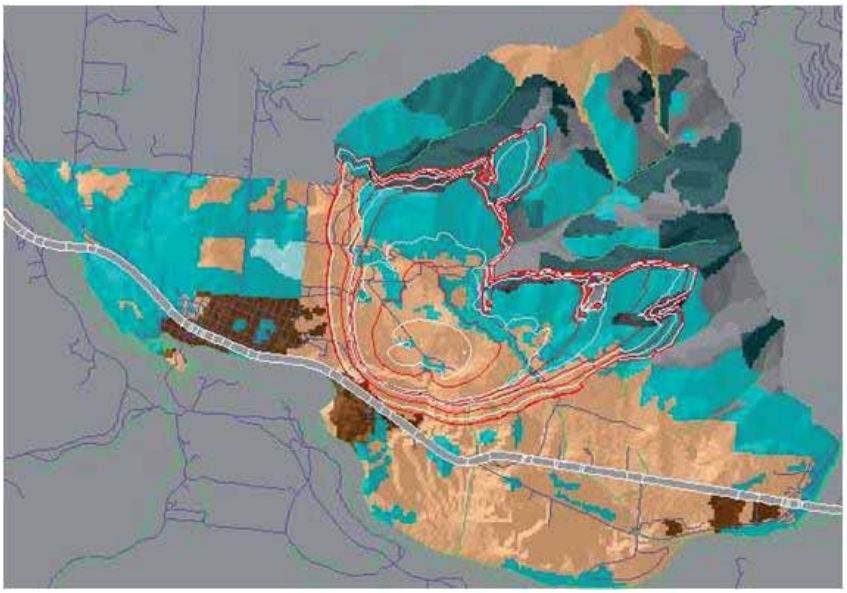

Fig. 3.6: Topographic area represented in FARSITE's *Ashley project*.

evolution of 50 different populations in each case. The different GA parameter configurations used in this were the following ones:

- Population size: 10, 25, 50 and 100 individuals.
- Mutation probability: values 0.01, 0.1 and 0.25.

In every case, the crossover probability was set to 0.2 and the elitism factor to 0.1. The adjustment error (i.e. the error produced by the best individual) was recorded at each step (generation) of the algorithm. Results are summarized in Table 3.3 and depicted in Figures 3.7, 3.8, 3.9.

Individuals	**Mutation**	**G1**		**G2**		**G3**		**G4**		**G5**	
		μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
10	0.01	1.28	1.01	1.27	0.86	1.06	0.52	0.96	0.4	0.93	0.36
	0.1	1.67	1.05	1.27	0.95	1.02	0.59	0.82	0.32	0.81	0.50
	0.25	1.65	1.43	1.06	0.44	0.97	0.42	0.94	0.41	0.93	0.41
25	0.01	0.82	0.33	0.81	0.37	0.74	0.40	0.67	0.43	0.65	0.41
	0.1	0.90	0.42	0.74	0.31	0.68	0.39	0.56	0.26	0.49	0.26
	0.25	0.86	0.34	0.62	0.30	0.46	0.21	0.42	0.18	0.34	0.18
50	0.01	0.51	0.44	0.40	0.23	0.37	0.26	0.35	0.21	0.28	0.23
	0.1	0.53	0.50	0.41	0.26	0.38	0.24	0.33	0.23	0.28	0.20
	0.25	0.67	0.47	0.55	0.30	0.40	0.25	0.31	0.18	0.22	0.13
100	0.01	0.44	0.20	0.38	0.19	0.35	0.18	0.28	0.15	0.22	0.12
	0.1	0.43	0.19	0.40	0.19	0.35	0.17	0.27	0.17	0.22	0.12
	0.25	0.62	0.30	0.48	0.20	0.35	0.18	0.28	0.15	0.22	0.12

Tab. 3.3: Average adjustment error values and standard deviations for each generation. Populations composed of 10, 25, 50 and 100 individuals, and mutation probabilities 0.01, 0.1 and 0.25. Each value obtained from sets composed of 50 different populations.

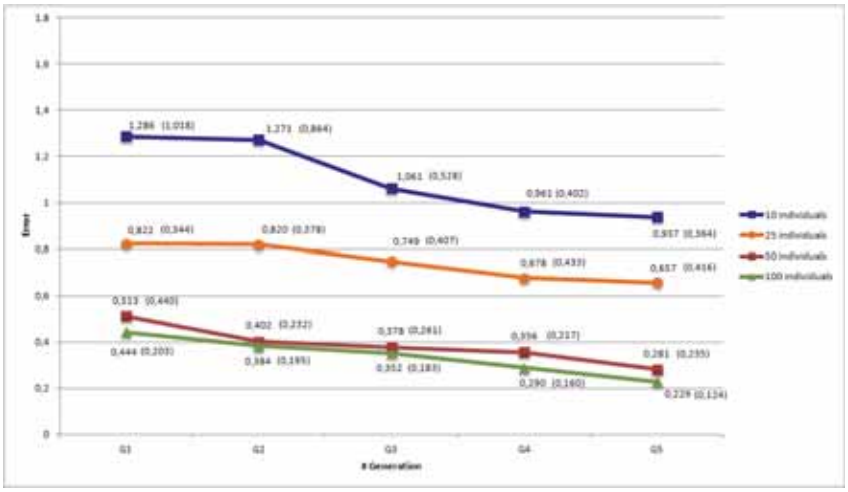

Fig. 3.7: Average adjustment error values. Mutation probability set to 0.01. Values in parenthesis represent standard deviations.

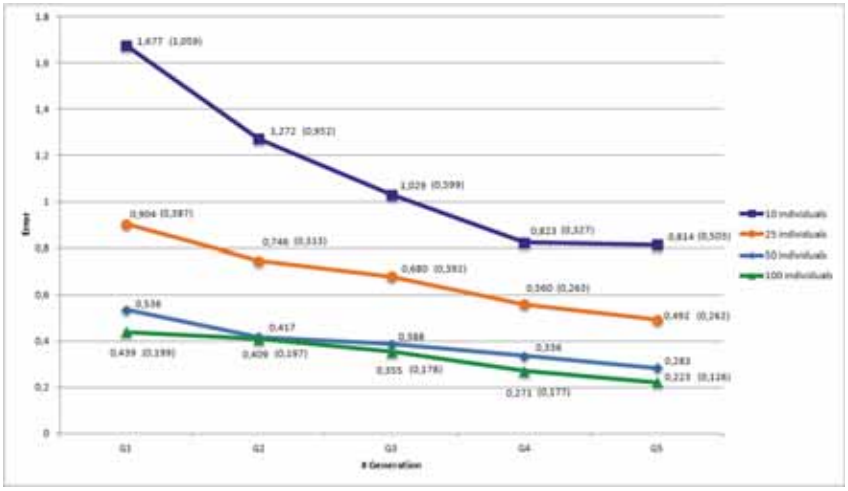

Fig. 3.8: Average adjustment error values. Mutation probability set to 0.1. Values in parenthesis represent standard deviations.

Analyzing the obtained results, a preliminary conclusion is that, as expected, the size of the populations has a direct and high impact on the convergence speed. Populations composed of 50 and 100 individuals present noticeably lesser errors in the earlier generations of the evolutionary process in every case.

Regarding the variations on mutation probabilities, one can observe that the smaller the size of the population is, the more sensitive to these changes it is. This fact is due to the increase of the possibility of exploring other search areas in the solution space, which turns out to be very beneficial in those cases where, because of the size of the population, the variety of the solutions proposed (the individuals of each population) is poor.

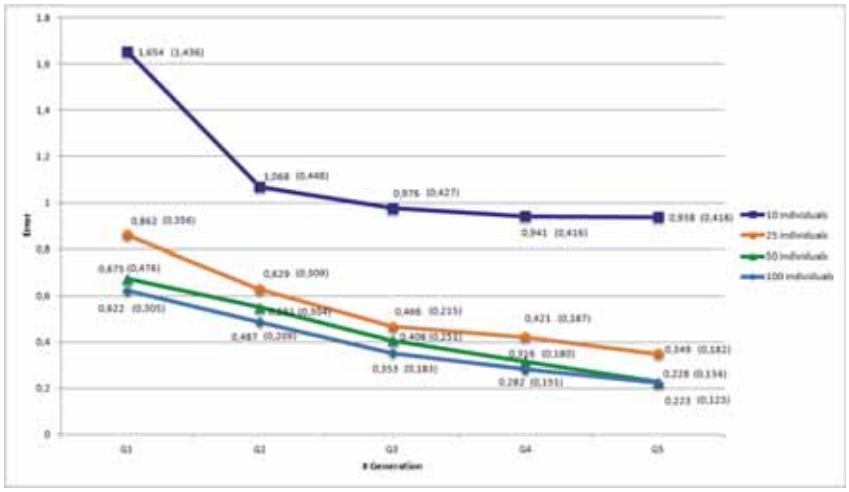

Fig. 3.9: Average adjustment error values. Mutation probability set to 0.25. Values in parenthesis represent standard deviations.

3.4.2 GA Statistical Study

From the obtained results, a statistical study carrying out the Kolmogorov-Smirnov, Anderson-Darling, and Chi-squared tests allowed us to determine that the probability distribution which best fits the obtained data is the *Logistic distribution*, which resembles the normal distribution but presents higher kurtosis. Its probabilty density function is the following:

$$pdf(x;\mu,s) = \frac{e^{-(x-\mu)/s}}{s(1+e^{-(x-\mu)/s})^2} \tag{3.2}$$

In this equation, x is the random variable (which corresponds to the obtained adjustment error), μ is the *location* factor, which is analogous to the mean value in a normal distribution, and s is the *scale* factor, which is proportional to the standard deviation, both of which are needed to define such probability distribution. Although the probability distribution of the data is the same throughout the whole evolution process, these factors vary depending on the iteration of the GA we are evaluating. So, Figure 3.10 depicts the different probabilty density functions for each generation, for the particular case of populations composed of 100 individuals and mutation probability set to 0.1.

By means of these probability density functions, we are able to guarantee, with different degrees of certainty, the maximum adjustment error we will obtain given a certain configuration of the GA. In addition, since the number of evolved generations has a direct impact on both the available resources and the time needed to perform the adjustment process, it is worth highlighting that we are able to give this guarantee taking into account the number of generations we are able to execute.

For instance, Table 3.4 shows the different maximum adjustment errors (considering the adjustment time interval [0 hours - 5 hours]) for which we have different *degrees of guarantee*, depending on the number of generations the GA iterates, for the specific case of populations composed of 100 individuals and mutation probability set to 0.1. Here, *guarantee degree*

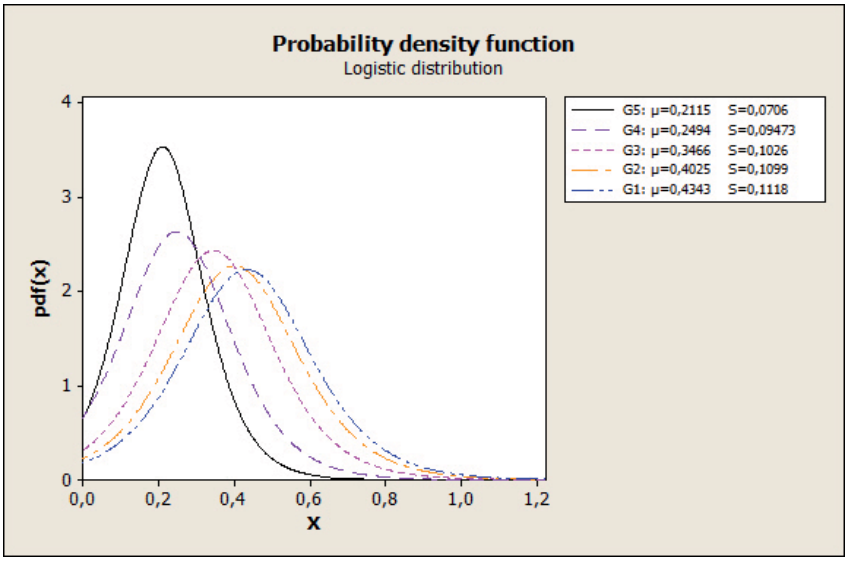

Fig. 3.10: Probability density functions for the obtained errors at each generation of the evolution process

stands for the probability of obtaining an adjustment error lesser than or equal to the specified value, on the basis of the above presented probabilty density function (Equation 3.2).

Guarantee degree	G1	G2	G3	G4	G5
95%	0.763	0.726	0.649	0.528	0.419
90%	0.680	0.644	0.572	0.458	0.367
85%	0.628	0.593	0.525	0.414	0.334
80%	0.589	0.555	0.489	0.381	0.309
75%	0.557	0.523	0.459	0.353	0.289
70%	0.529	0.496	0.434	0.330	0.271
65%	0.504	0.471	0.410	0.308	0.255
60%	0.480	0.447	0.388	0.288	0.240
55%	0.457	0.425	0.367	0.268	0.226
50%	0.434	0.403	0.347	0.249	0.211

Tab. 3.4: Maximum adjustment errors and degrees of guarantee, depending on the number of GA generations. Populations of 100 individuals, mutation probability set to 0.1.

Figure 3.11 also depicts this information, from a guarantee degree of 95% down to 75%. As can be easily understood, the lesser the error requested is, the lesser the degree of guarantee, for the same number of iterations of the GA.

Table 3.5 shows the different maximum adjustment errors (considering the adjustment time interval [0 hours - 5 hours]) that can be guaranteed in 90% of the adjustment processes, depending on the number of individuals per population and the number of iterations of the GA. Again, mutation probability was set to 0.1.

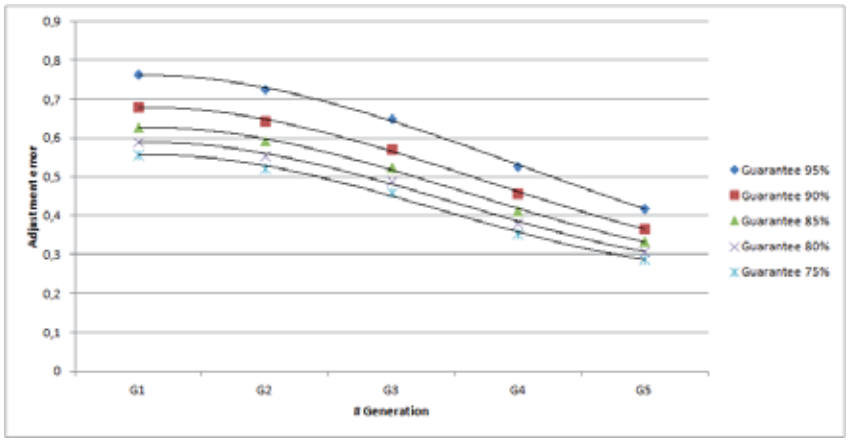

Fig. 3.11: Maximum adjustment errors and degrees of guarantee. Populations of 100 individuals, mutation probability set to 0.1.

Individuals per population	G1	G2	G3	G4	G5
10	2.96	2.43	1.75	1.22	1.12
25	1.39	1.13	1.05	0.87	0.80
50	1.15	0.74	0.68	0.62	0.54
100	0.68	0.64	0.57	0.46	0.37

Tab. 3.5: Maximum adjustment errors for a 90% degree of guarantee, depending on the number of GA generations and the number of individuals per population. Mutation probability set to 0.1.

Considering a real situation, where the quality of the prediction is a parameter fixed by the decision control centre in charge of making the appropriate decisions about how to fight against the ongoing fire, this information turns out to be very important, since we are able to give a certain guarantee of quality in the final prediction, taking into account how many evolution steps (i.e. how many generations) we can perform. This, as previously stated, will be determined by the available computational resources and time to deliver a prediction. Moreover, it is also possible to fix the quality of the adjustment and then determine the guarantee degree of reaching such error in a given number of iterations.

3.4.3 GA Methodology Step by Step

Taking into account the convergence analysis and statistical studies reported above, our proposed methodology for the GA characterization can be summarized in the following steps:

1. **GA key settings identification:** It is necessary to assess and determine those settings that could affect the quality of the results when using GA as the adjustment technique. So far, the analyzed factors in this thesis are the size of the populations, the number of generations and the mutation probability.

2. **Study of the impact on convergence:** Depending on the specific case (mainly, the topographic area), the influence of certain GA settings upon the quality of the results

may be decreased in favour of others, and vice-versa. Therefore, it is necessary to study the particular impact of these factors on the convergence for each case.

3. **Statistical analysis:** At this point, a statistical analysis is performed. This consists of identifying which probability distribution best fits the results obtained, in terms of the quality of the adjustment obtained. Once it is determined, by means of the corresponding probability density function, we are able to establish a guarantee degree in our estimations, as well as determine which configuration of the GA is most suitable. All the statistical analysis reported in this work was done using EasyFit 5.5 [22], and Minitab 15 [44] as data analysis tools.

As in the methodology for the simulator kernel characterization, described in Section 3.3.2, these steps are made *offline*, which means that they are all already carried out at the moment of the fire occurrence. However, in this case, the results and applicability of this methodology are independent of the computational resources that are being used.

Having completed these three processes, we are ready to infer criteria to determine the achievable adjustment quality under certain restrictions, and are therefore able to advise on the best specific settings for the GA.

In the next chapter, we discuss the details of the conducted validation carried out to confirm this study.

4 Methodology Validation

4.1 Simulator Kernel Characterization Validation

As discussed in Chapter 3.3, every simulator presents different sensitivity to certain input parameters, and our methodology has to be flexible regarding the underlying simulator; therefore, it is necessary to characterize this process in a way that we can rapidly assess the execution time of a future simulation independently from the simulator we will use.

This fact highlights the need to rely on complex criteria in order to successfully classify the input data sets according to the execution time they will cause. Consequently, we rely on the Artificial Intelligence field to reach such an objective. Decision trees have come to be a suitable solution, since the time needed to determine an estimation for the execution time is almost negligible (on every study case reported in this document, this process lasted between 0.2 and 0.5 seconds). Moreover, as will subsequently be analyzed, it provides a very high hit ratio in its classifications. The decision trees used in this research were generated by the C4.5 algorithm [50], specifically, the J48 open source Java implementation of the C4.5 algorithm in the Weka [31] data mining tool.

This characterization is fulfilled by means of carrying out large sets of executions (on the order of thousands) counting on different initial scenarios (different input data sets), and then, applying knowledge-extraction techniques from the info they provide. We record the execution times from the experiment, i.e. we build a training database, and then we establish a classification of the input parameters according to the elapsed times they produced. At this moment, we are capable of building the decision tree to determine classification criteria and, therefore, given a new set of input parameters, estimate how long the execution will last.

This learning process is carried out *offline*, i.e. the classification rules are established prior to the hazard occurrence. Therefore, at the moment of urgency management, we only have to apply the classification technique, which, as stated above, involves a negligible cost of computational time.

The validation of this part of the methodology was carried out using the three different simulators introduced in Chapter 2.3.2. Subsequently, we detail the different scenarios and present the results obtained for each case.

4.1.1 Firelib Simulator

The simulated scenario in this case was the following one:

- Domain: an artificial 1001x1001 cells map was used (cell width and height: 100 feet).
- Simulation duration: FireLib simulations end once the fire reaches one edge of the map.
- Ignition point: the ignition point in this case was the central cell of the map.

Once the distribution of each input parameter was established (see Table 3.2), a set of 38750 different input settings was generated and simulated.

As regards the computational platform, all the experiments were carried out on a cluster of 8 x Dell PowerEdge M600 nodes, each of which counting on 2xQuad-Core Intel Xeon E5430, 2.66GHz, 2x6MB L2 cache memory (2x2) and 16 GB RAM Fully Buffered DIMMs 667MHz, running Linux version 2.6.16.

As one can see in Figure 4.1, the variance in simulation time is very noticeable. The great majority of the executions are located under the 2500 seconds threshold, but there were several executions that lasted more than 30000 seconds, and some even more than 50000 seconds.

From the point of view of emergency prediction, it is crucial to have the question of execution time under control, so we may deal with cases that drastically slow down the prediction process. An elapsed time prediction for a simulator execution with an error on the order of thousands of seconds would be prohibitive; so, from cases like this one, the need arises to be able to predict how the simulator is going to behave and, therefore, the need to use an efficient classification technique.

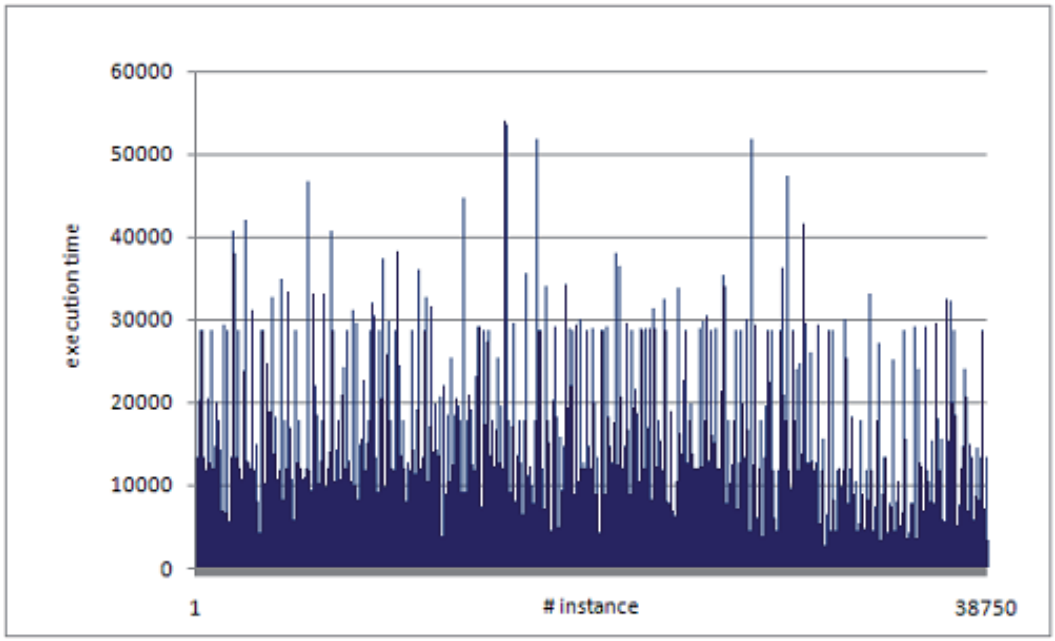

Fig. 4.1: Execution times using fireLib.

The number of classes, and the execution time intervals they represent, were determined by taking into account where our work is framed, i.e. the intervals chosen for each class are

those that in a real emergency situation would matter (it makes no sense, for example, to classify by intervals of 10 seconds when predicting forest fire spread). They are:

- Class A: ET $\leq$ 900 seconds.
- Class B: 900 seconds $<$ ET $\leq$ 1800 seconds.
- Class C: 1800 seconds $<$ ET $\leq$ 3600 seconds.
- Class D: 3600 seconds $<$ ET $\leq$ 7200 seconds.
- Class E: 7200 seconds $<$ ET.

		Predicted Class				
		A	B	C	D	E
Real Class	A	669	14	4	2	0
	B	17	72	9	4	0
	C	2	12	72	12	4
	D	5	6	14	24	5
	E	0	3	2	12	36

Tab. 4.1: Correspondence between real and predicted classes.

Where *ET* stands for execution time.

In order to validate our method, a set of 1000 new instances were generated (according to the distributions shown in Table 3.2) to be used as a test set. The results of the application of decision trees to the test set are summarized in Table 4.1. Here, one of the main aspects to highlight is the prominence of the main diagonal, which means that perfect matches are predominant over the whole set of predictions. Furthermore, one can notice that the values decrease as one moves away from the main diagonal. Indeed, the worst possible cases (predict A when the real class is E, and vice-versa) never happened.

Figure 4.2 shows the absolute values of the number of predictions that totally hit the real class, as well as the absolute values where the prediction had an accuracy determined by the distance between classes. A *Distance X* accuracy means that there are X-1 classes between the predicted class and the real class.

The most noticeable aspect when analyzing this graphic is that, if we consider *Distance 1* as a good prediction accuracy, then the results obtained present a 96.8% satisfactory classifications.

4.1.2 Firestation Simulator

The simulated scenario in this case was the following one:

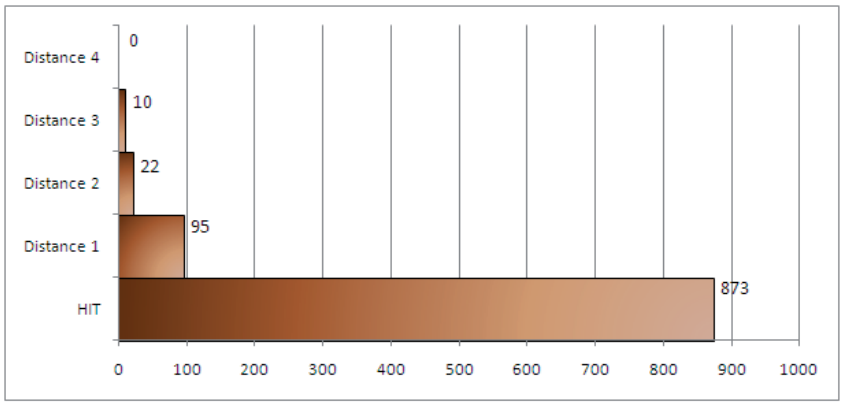

Fig. 4.2: Classification accuracy using FireLib.

- Domain: an artificial 2893x2860 cell map was used (cell width and height: 25 meters).
- Simulation duration: simulations end once the fire burns 10^6 cells.
- Ignition point: the ignition point in this case was the central cell of the map.

Again, a validation test analogous to the one previously presented was carried out, but in this case, the training database was composed of 13000 different input settings.

The number of classes, and the execution time intervals they represent, were as follows:

- Class A: ET $\leq$ 900 seconds.
- Class B: 900 seconds $<$ ET $\leq$ 1800 seconds.
- Class C: 1800 seconds $<$ ET $\leq$ 3600 seconds.
- Class D: 3600 seconds $<$ ET.

Where *ET* stands for execution time.

Again, a set of 1000 new instances were generated (according to the distributions shown in Table 3.2) to be used as a test set. The results of the application of decision trees to the test set are summarized in Table 4.2. As one can observe, the main diagonal remains prominent, as expected.

		Predicted Class			
		A	B	C	D
Real Class	A	216	30	1	1
	B	26	655	10	9
	C	0	13	25	4
	D	0	2	6	2

Tab. 4.2: Correspondence between real and predicted classes using Firestation.

Figure 4.3 shows the absolute values of the number of predictions that totally hit the real class, as well as the absolute values where the prediction had an accuracy determined by the distance between classes.

In this case, if we keep considering *Distance 1* as a good prediction accuracy, then the results obtained present a 98.7% satisfactory classifications.

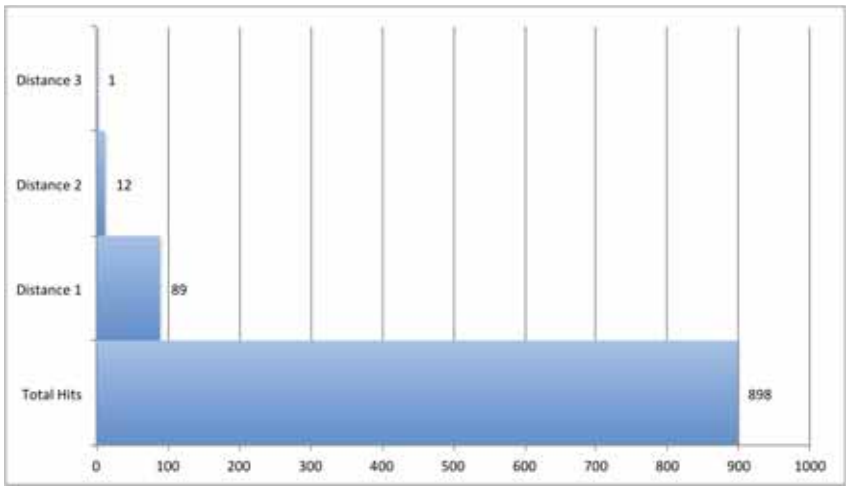

Fig. 4.3: Classification accuracy using FireStation.

4.1.3 FARSITE Simulator

The simulated scenario in this case was as follows:

- Domain: GIS data from the benchmark provided by FARSITE (the *Ashley project*).
- Simulation duration: simulation of 30 hours of fire spread.
- Ignition point: the ignition point in this case was the central cell of the map.

Again, a validation test analogous to the previous ones was carried out, but, in this case, the training database was composed of 12000 different input settings.

In this particular case, the number of classes, and the execution time intervals they represent, were established taking into account the information the histogram of execution times gives us.

Figure 4.4 shows the boundaries for the execution time intervals we will use when proceeding to classify a given unknown set of input parameters. For this purpose, we bound different classes by analyzing the different local minimums the histogram presents, so that we minimize the classification errors due to values too close to the boundaries.

Thus, the defined classes are the following, where *ET* stands for execution time:

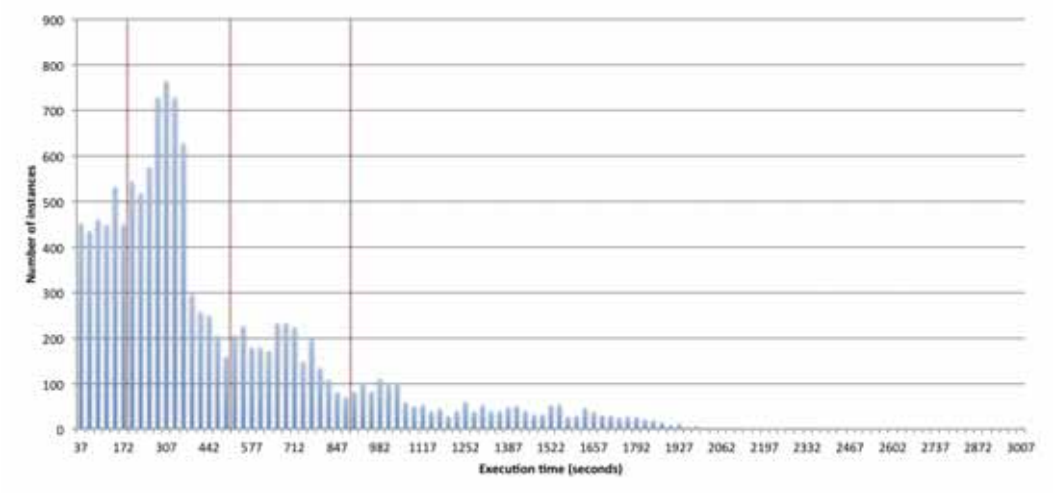

Fig. 4.4: Histogram of execution times using FARSITE. Vertical dotted lines indicate the defined classification boundaries.

Tab. 4.3: Correspondence between real and predicted classes using FARSITE.

		Predicted Class			
		A	B	C	D
Real Class	A	341	11	0	1
	B	10	358	6	1
	C	0	6	108	3
	D	0	0	5	150

- Class A: ET $\leq$ 175 seconds.
- Class B: 175 seconds $<$ ET $\leq$ 500 seconds.
- Class C: 500 seconds $<$ ET $\leq$ 875 seconds.
- Class D: 875 seconds $<$ ET $\leq$ 3600 seconds.

The results of applying decision trees to the test set are summarized in Table 4.3. Here, one of the main aspects to highlight is the prominence of the main diagonal, which means that perfect matches are predominant over the whole set of predictions. Furthermore, one can notice that the values decrease as one moves away from the main diagonal.

Figure 4.5 shows the absolute values of the number of predictions that totally hit the real class, as well as the absolute values where the prediction had an accuracy determined by the distance between classes. A *Distance X* accuracy means that there are X-1 classes between the predicted class and the real class.

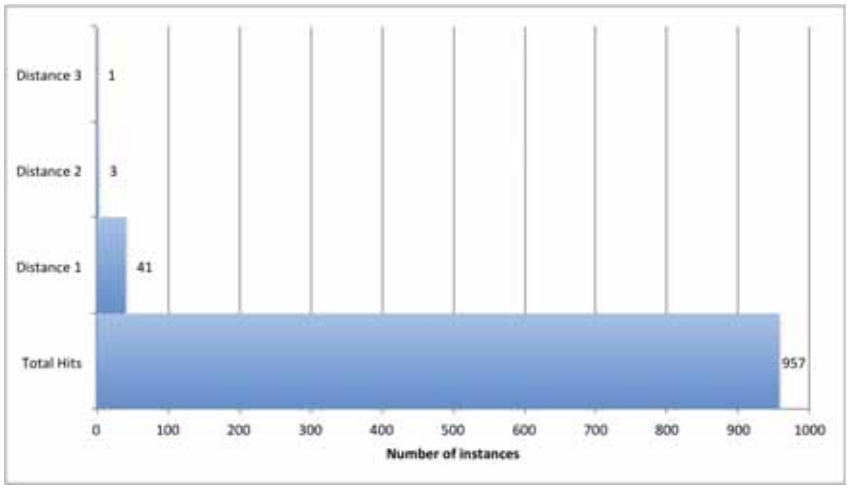

Fig. 4.5: Classification accuracy.

As can be observed when analyzing this graphic, we obtained a 95.7% correct classifications, and, if we consider *Distance 1* as a good prediction accuracy, then the results obtained present a 99.8% satisfactory classifications.

Obviously, this represents a very good result, which allows us to predict in advance how long a simulation will last, with a very high degree of certainty. It is worth mentioning that, in order to be prudent, given a certain input setting for a simulation, we assign the upper-bound value of its class to the expected execution time it will produce.

4.2 Genetic Algorithm Characterization Validation

Once the afore-mentioned statistical analysis has been carried out, we present a validation test by means of which we can prove that this characterization methodology is suitable for the problem that we are tackling.

For this purpose, we carried out the adjustment process, considering the adjustment time interval [0 hours - 5 hours], for five new random populations (p0 - p4) in the case of a new, different fire. Specifically, as described in Section 3.4, we used FARSITE as the fire propagation simulator, as well as the same GIS data, but we completely changed the conditions of the reference fire, so its spread varied significantly.

As regards the GA configuration, the five new populations were composed of 100 individuals, both elitism and mutation factor were set to 10%. Again, we performed five-generation evolution process.

Table 4.4 shows the obtained errors for each population. In order to contrast this data with the guaranteed errors exposed in Table 3.4, let us consider three cases of different guarantee degrees in our estimations before the adjustment process was carried out:

Population	G1	G2	G3	G4	G5
p0	0.601	0.376	0.376	0.376	0.376
p1	0.993	0.450	0.450	0.125	0.096
p2	0.433	0.433	0.433	0.433	0.309
p3	0.823	0.332	0.105	0.105	0.105
p4	0.894	0.343	0.343	0.323	0.323

Tab. 4.4: Adjustment errors for populations p0-p4 for the calibration interval (0hours - 5hours)

- Estimation with 95% degree of guarantee: in this case, all of our estimations were right, with the exception of the errors obtained at generation 1 of populations p1, p3 and p4, which present very high adjustment errors. This fact was expected, and it is understandable, since because of the random features of genetic algorithms, only one step of evolution is not enough at all to determine any kind of significant approach to a suitable solution, so it is not enough to give an estimation with any degree of guarantee.

- Estimation with 85% degree of guarantee: again, we failed in the estimations of populations p1, p3 and p4 at generation 1. However, for the rest of the generations, we only failed in one estimation at generation 4 (population p2), and one in generation 5 (population p0), which is acceptable taking into account this guarantee degree.

- Estimation with 75% degree of guarantee: in this case, we failed in populations p0, p1, p3 and p4 at the end of generation 1. At the end of generation 4, we failed in our estimations in the cases of populations p0 and p2. At the end of generation 5, we failed in three cases: populations p0, p2 and p4. In this specific case of guarantee degree we have obtained worse results, since we expected a 25% probability of fail, that is, one or two fails for a set of five populations. However, the difference between the guaranteed error and the ones obtained for those populations is very small, which leads us to consider that in a greater set of populations, the guarantee degree in our estimations would fit the results obtained.

These three different cases validate our proposal, so we are able to establish, in advance, adjustment error boundaries in our prediction framework. This turns out to be very important for the final predictions, since the adjustment error and final prediction error in our two-stage prediction method are highly correlated [10]. In addition, we introduce a degree of certainty to our future predictions, which is very valuable at the time of making decisions.

Furthermore, taking into account the capability to introduce a degree of guarantee in our estimations, based on the probability density function, then we may tackle the problem from another point of view: given a fixed adjustment quality required by the decision control center, we can determine the guarantee degree of reaching such error (or lesser) in a given number of iterations.

Figure 4.6 shows the guarantee degree for errors of 0.33 and 0.5 considering populations of 50 and 100 individuals.

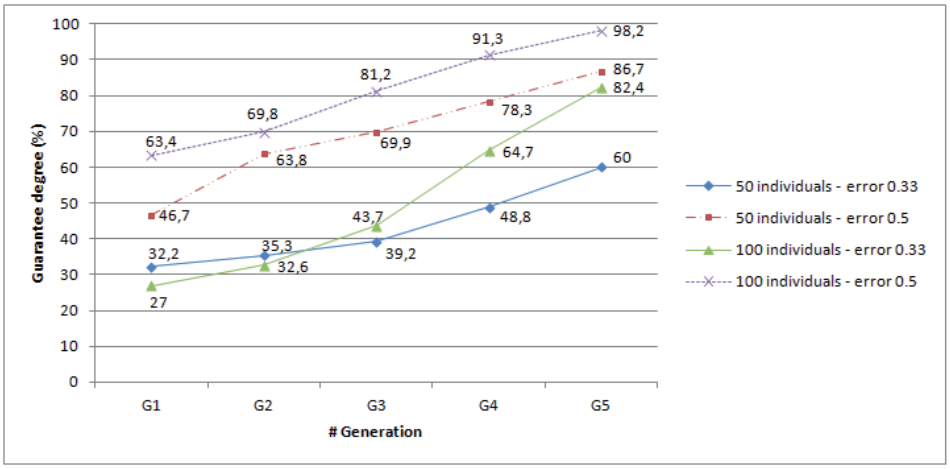

Fig. 4.6: Guarantee degrees for adjustment errors 0.33 and 0.5 (populations composed of fifty and one hundred individuals, mutation probability set to 0.1).

This information turns out to be very significant, since we are able to establish error thresholds taking into account how many individuals we can simulate in parallel, and how many evolution steps (i.e. how many iterations) we can perform. This information will determine the computational resources and time necessary to deliver a prediction.

Counting on this information, we are able to recreate a real situation where different quality and time restrictions are imposed. As mentioned above, in the presence of restrictions, we have to effectively deal with the specific configurations of the GA used as the adjustment technique, since the number of iterations (generations) have a direct impact on both the quality of the results and the execution time. In the same way, the number of individuals affects the convergence speed and the amount of computational resources required.

In this sense, an experiment consisting of the evolution of 10 populations of 50 individuals, and 10 populations of 100 individuals was carried out.

The fire simulator used in this case was FARSITE, and the classification of the execution times is the same one described in Section 3.3, which is:

- Class A: ET $\leq$ 175 seconds.
- Class B: 175 seconds $<$ ET $\leq$ 500 seconds.
- Class C: 500 seconds $<$ ET $\leq$ 875 seconds.
- Class D: 875 seconds $<$ ET $\leq$ 3600 seconds.

Moreover, in this experiment, only individuals whose execution time had been previously classified, at the most as *Class C*, were allowed to be executed. A further study of how this action affects the quality of the adjustment is reported in Chapter 5). Table 4.5 shows the obtained values, in terms of the quality of adjustment and the time spent to achieve it.

Taking into account the probability density function expressed by Equation 3.2, let us consider two different cases, regarding adjustment quality requirements:

- Case A: an adjustment error of 0.33 is required. In this case, the only suitable configuration for the GA would be a population of 100 individuals. As can be seen in Figure 4.6, the probability density function tells us that we can achieve this quality with a guarantee degree of 82.4% at the fifth generation, and with 64.7% at the fourth generation.

- Case B: an adjustment error of 0.5 is required. In this case, a configuration of 100 individuals would be appropriate, so we can achieve it at the fourth generation with a 91.3% degree of guarantee. A configuration with 50 individuals would also be suitable, since we can reach such quality at the fifth generation with a guarantee degree of 86.7%.

As can be observed in Table 4.5, populations composed of 100 individuals fulfill the requirements in both cases. Considering *Case A*, at the end of the fifth generation, only two populations exceeded the error of 0.33 (populations p2 and p3). At the end of the fourth generation, three populations exceeded that error (populations p2, p3 and p7). These results comply with the established degrees of guarantee (82.4% and 64.7%, respectively). *Case B* is also fulfilled at the fourth generation, since only one population (p7) presented an error higher than 0.5, being 91.3% the corresponding degree of guarantee.

As for populations composed of 50 individuals, for this specific experimental set, *Case B* (error 0.5) is not satisfied in three cases (p0, p2 and p6). Given the previously mentioned guarantee degree of 86.7%, one might expect only one or two negative cases, but taking into account the size of this experimental sample, and the fact that the error produced by population p0 (0.512) is very close to our target, the results are acceptable.

Individuals	Population	G1		G2		G3		G4		G5	
		ϵ	time	ϵ	time	ϵ	time	ϵ	time	ϵ	time
50	p0	0.831	711 s	0.698	714 s	0.698	715 s	0.512	738 s	0.512	768 s
	p1	0.431	759 s	0.431	778 s	0.431	813 s	0.431	458 s	0.427	575 s
	p2	0.763	560 s	0.763	730 s	0.564	609 s	0.563	669 s	0.563	573 s
	p3	0.602	813 s	0.602	718 s	0.598	537 s	0.598	800 s	0.496	692 s
	p4	0.602	794 s	0.602	530 s	0.602	668 s	0.602	694 s	0.412	558 s
	p5	0.627	801 s	0.572	791 s	0.572	784 s	0.572	866 s	0.422	718 s
	p6	0.843	862 s	0.843	761 s	0.624	776 s	0.624	755 s	0.624	834 s
	p7	0.544	856 s	0.541	858 s	0.541	580 s	0.489	605 s	0.449	707 s
	p8	0.503	809 s	0.503	705 s	0.312	772 s	0.312	681 s	0.311	623 s
	p9	0.829	790 s	0.828	792 s	0.026	805 s	0.026	844 s	0.026	780 s
100	p0	0.848	710 s	0.148	799 s	0.148	865 s	0.148	799 s	0.146	640 s
	p1	0.333	870 s	0.330	717 s	0.330	813 s	0.330	818 s	0.330	814 s
	p2	0.861	835 s	0.344	742 s	0.344	806 s	0.342	809 s	0.342	779 s
	p3	0.865	787 s	0.520	867 s	0.393	773 s	0.393	805 s	0.393	764 s
	p4	0.166	855 s	0.166	866 s	0.166	861 s	0.164	728 s	0.161	786 s
	p5	0.993	863 s	0.450	868 s	0.450	828 s	0.125	672 s	0.095	740 s
	p6	0.823	860 s	0.332	860 s	0.105	708 s	0.105	866 s	0.105	828 s
	p7	0.588	839 s	0.558	802 s	0.558	780 s	0.558	770 s	0.317	767 s
	p8	0.825	734 s	0.287	728 s	0.287	864 s	0.235	781 s	0.235	818 s
	p9	0.830	865 s	0.322	809 s	0.322	681 s	0.322	646 s	0.315	863 s

Tab. 4.5: Adjustment errors obtained for 10 different populations, and elapsed times for each generation. Populations composed of 50 and 100 individuals. Mutation probability set to 0.1.

4.3 Discussion

The results reported in this chapter satisfactorily validate the methodology for the prediction scheme characterization proposed in Chapter 3. In both cases of *Simulator Kernel Characterization* and *Genetic Algorithm Characterization*, the accuracy of our estimations indicate that our proposed methodology is suitable and feasible. Some aspects to be discussed arise from this validation process.

The capability of executing in parallel as many simulations as possible not only has a direct impact on the quality of the results, but also on the time needed to deliver them. In Table 3.5, one can easily see that populations composed of 100 individuals present errors hardly achievable by populations of ten individuals, even extending the adjustment process to more than five generations. This fact happens because of the existence of local minimums in the search space. It is also noticeable that the evolution of populations of 100 individuals are always more favourable than populations of 50 individuals, even performing one less iteration of the GA. Nevertheless, in order to benefit from a 100-individual configuration of GA, it is compulsory to be able to simulate them in a parallel way; otherwise, this configuration would make the time needed to give the prediction shoot up. Hence, this fact highlights the need to count on efficient Urgent Computing solutions so that we can tackle this problem in a feasible way.

We need to be able to anticipate input settings which may lead the simulation to last too long, without the need of executing them. K. Leyton-Brown et alter describe in [36] a similar idea about capping certain input configurations of a black-box algorithm in order to be able to identify hard regions of the feature space of the *Winner Determination Problem*

These results not only validate our proposals, but also emphasize the importance the amount of available resources. It is obvious that the more individuals populations have, the faster the convergence to a good solution is. This, in terms of execution time, becomes a key point to be considered when dealing with emergencies.

Let us suppose that, in addition to the previously mentioned quality requirements, a strict deadline of one hour for carrying out the adjustment process is imposed. As mentioned above, *Class C* individuals last, at the latest, 875 seconds. This deadline would prevent us from iterating more than four generations, considering we have the capacity to execute all the individuals in a parallel way. Obviously, this capacity depends on the number of computational resources we have access to at the moment of dealing with the ongoing fire.

Thus, in the case of having access to 100 computational resources (i.e. being able to carry out 100 simulations in parallel), both *Case A* and *B* are satisfied with good degrees of guarantee. However, a limitation of 50 computational resources would make our degree of guarantee significantly decrease to only 48.8% in *Case A* (error 0.33, see Figure 4.6) and 78.3% in *Case B* (error 0.5). This important drop in the degree of guarantee regarding the quality of the adjustment would be prohibitive, especially in cases where human lives are threatened.

These results highlight the need to be able to count on as many individuals (i.e. as many simulations per iteration) as possible in order to perform a prediction with guarantees. In practice, this fact implies the ability to have access to more computational resources at the moment of dealing with the emergency. These results clearly highlight the need of Urgent Computing solutions in order for our two-stage prediction method to be effective.

5 Experimental Evaluation

The methodology described in Chapter 3 has been designed to be able to respond to an ongoing hazard in an efficient way. Efficiency, in the case of natural hazard management, stands for delivering accurate predictions respecting strict deadlines. For this purpose, it is absolutely necessary to optimize the use of the available computational resources, at the moment of attending to the emergency.

By means of the application of the proposed methodology, we are able to tackle this problem from different points of view.

In this chapter, a series of experiments aimed at demonstrating the different benefits of the application of this methodology is described and analyzed.

5.1 Time Constraints vs Quality of the Prediction

In this section, we report an experimental study to demonstrate the benefits of the ability to discard in advance those initial settings for the simulation whose execution times would cause the adjustment technique to last longer than the initial pre-set deadline for the prediction. Furthermore, we demonstrate that the application of the previously described classification technique does not have an impact on the quality of the prediction results.

The fire simulator used in this case was FARSITE, and the classification of the execution times is the same one described in Chapter 3.3, which is:

- Class A: ET $\leq$ 175 seconds.
- Class B: 175 seconds $<$ ET $\leq$ 500 seconds.
- Class C: 500 seconds $<$ ET $\leq$ 875 seconds.
- Class D: 875 seconds $<$ ET $\leq$ 3600 seconds.

In this study, ten experiments have been carried out, starting from ten different initial random populations of fifty individuals and evolving them over five generations. The elitism factor was set to 5 (10%), the mutation probability to 1%, and the crossover factor to 20%. Then, we analyze the results obtained from the calibration step, considering the adjustment time interval [0 hours - 5 hours].

Population	Calibration error	#Generations with Class D	Execution time
0	0.31238	0	2278.15 s
1	0.120206	0	6095.8 s
2	0.203242	2	7966.77 s
3	0.127323	4	10412.32 s
4	0.13543	2	5342.65 s
5	0.022934	0	4292.8 s
6	0.071767	1	8585.69 s
7	0.178331	2	9355.31 s
8	0.1724	0	4244.5 s
9	0.209174	0	6098.34 s

Tab. 5.1: Results obtained in the calibration interval [0 hours - 5 hours]

It is worth emphasizing that the computational resource used in this work provides enough computing elements to be able to execute every individual of a given generation in a different node, i.e. all individuals of each generation start their corresponding simulation at the same time, being processed in parallel. This fact implies that the time incurred in processing each generation depends on the individual which produces the slowest simulation.

In this experiment, we also establish a timeout of one hour, so simulations that reached this threshold were discarded from the study.

Analysis of the results.

Table 5.1 summarizes the results obtained from this experiment. The values in the second column correspond to the error of the best individual after five generations for each particular population. Since the underlying fire simulator produces a raster file indicating the *time of arrival* of the fire for each cell of the simulated map, the quality error is calculated by means of the Equation 3.1.

As expected, different initial populations lead to different qualities of results. Nevertheless, since our techniques are supposed to be applied in an urgent situation, it is worth examining the time spent on each evolution process. For this purpose, we perform a *post-mortem* classification of the individuals involved in that process.

As stated above, the time spent in each generation depends on the individual that produces the slowest simulation in that particular generation. Therefore, in order to evaluate the elapsed time for each evolution process, we focus on analyzing, for each population, how many generations have individuals that notably delay its evolution, i.e. for each population, how many generations have individuals classified as D. Table 5.1 summarizes this information in the third column.

Other interesting results that should be pointed out from this experiment consist of the lack of relation, for the same configuration of the Genetic Algorithm, between the time incurred

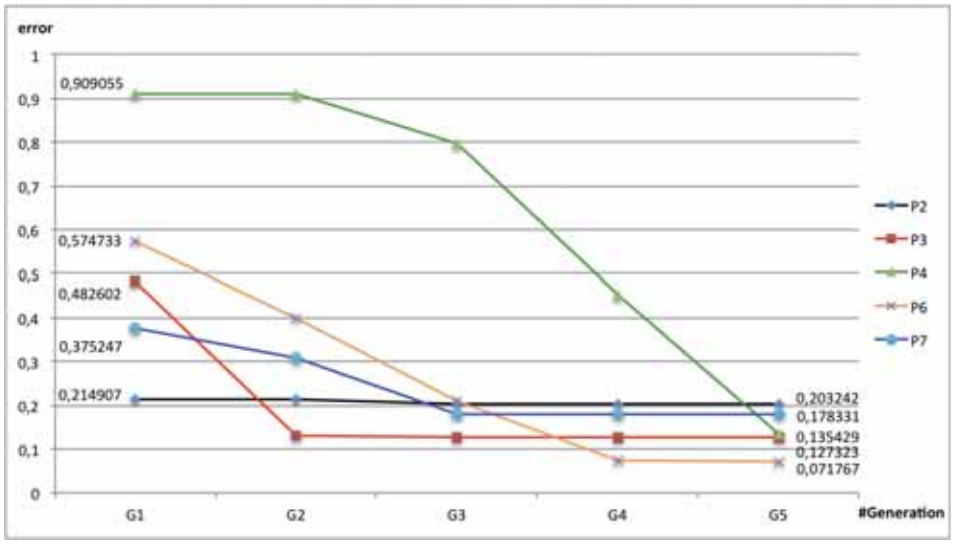

Fig. 5.1: Convergence curve not discarding Class D individuals

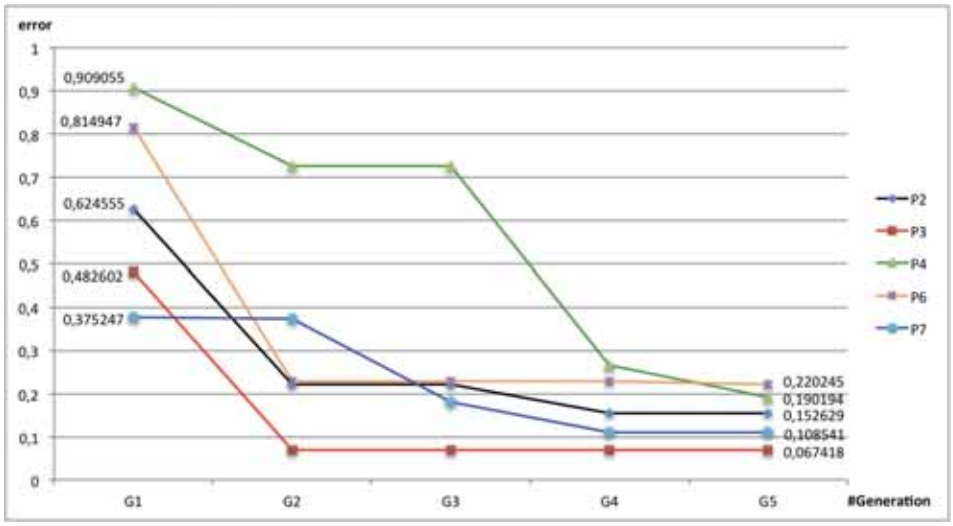

Fig. 5.2: Convergence curve discarding Class D individuals

Population	Calibration error	Execution time
2	0.152629	4926.66 s
3	0.063892	6988.91 s
4	0.190194	3803.98 s
6	0.220245	5389.55 s
7	0.108541	6936.61 s

Tab. 5.2: Results obtained in the calibration interval [0 hours - 5 hours], discarding Class D individuals

during the calibration step and the quality of results obtained. Excepting population 4[1], every population presenting Class D individuals produced an evolution time significantly higher than those which do not.

This fact becomes clear when examining the cases of population 5 and population 3. The former is the second fastest with a very low calibration error, and the latter also produced a good calibration quality, but lasted more than double the time. As one can see, the error obtained in population 5 is approximately six times less than the one obtained in population 3, and the execution time of the latter was approximately two and a half times the time produced by the former. This, in absolute terms, means a difference of more than 100 minutes (almost two hours).

This fact suggests that discarding slow individuals from the evolution process, and replacing them with other faster ones, does not necessarily mean losing results in quality. In order to assess this hypothesis, we repeated the evolution of each population presenting Class D members with exactly the same conditions (including the same initial population), but replacing Class D individuals at the moment they are detected with other randomly-generated individuals (classified as not Class D).

Table 5.2 shows the results of these evolutions, in terms of quality of the calibration and execution time.

As expected, the fact of discarding slow individuals from the evolution process does not represent a loss of quality, whereas it saves a considerable amount of time, which may turn out to be crucial, taking into account the risks involved. Figures 5.1 and 5.2 show the convergence of these populations both discarding and not discarding Class D individuals. Although they may present different behaviors during the process, the quality of the final result is similar in each case, as is the average error, as can be observed in Figure 5.3.

The main conclusion of this experiment is that if we apply the classification strategy previous to the submission of the individuals to the computing platform, we are able to detect in advance combinations of input parameters that will make the adjustment process increase its duration prohibitively. Therefore, we can remove them from the process, and this elimination will not necessarily affect the accuracy of the results.

[1]This is due to the fact that, in this particular case, the execution times of Class D individual were very close to the Class C - Class D boundary.

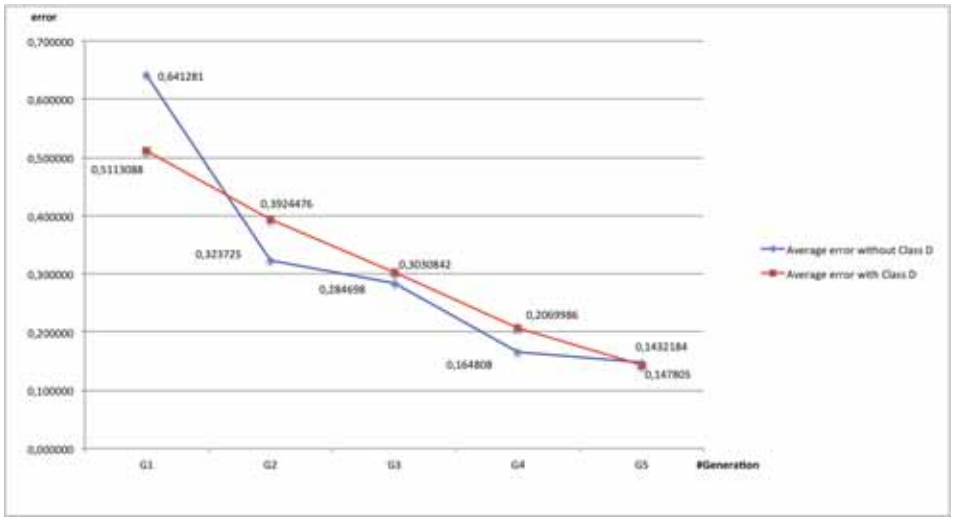

Fig. 5.3: Average error values for convergence both discarding and not discarding Class D individuals

Furthermore, for this experimental study, each generation was processed in a parallel way, so one can realize the huge gain that can be obtained from the application of the proposed methodology for prediction time and quality enhancement assessment.

5.2 Characterization Based on *Cap de Creus* Landscape

Since the proposed methodology was succesfully validated (see Chapter 4), we focus on real areas prone to forest fires every year. For this purpose, we examined the most problematic areas in Catalonia. With the help of the *Centre de Recerca Ecològica i Aplicacions Forestals* [17] of the Universitat Autònoma de Barcelona, we can access very valuable data. Figure 5.4 shows the ocurrences of fires bigger than 30 hectares in the period 1975-2010 [35, 54]. As can be observed, the most recurrent area corresponds to the north-east cape (*El Cap de Creus*). This area is zoomed in on Figure 5.5, and has an approximate real extension of 900 squared kilometers. In this case, different colors indicate different vegetation types.

Based on this new landscape, a new set of experiments was carried out in order to test the proposed methodology. For this purpose, we followed the steps detailed both in Sections 3.3.2 and 3.4.3.

5.2.1 Simulator Kernel Characterization

The simulated scenario in this case was the same as in Section 4.1.3, but replacing the landscape (see Figure 5.5). The training database was also identical (12000 different input settings), as well as the validation set (1000 instances).

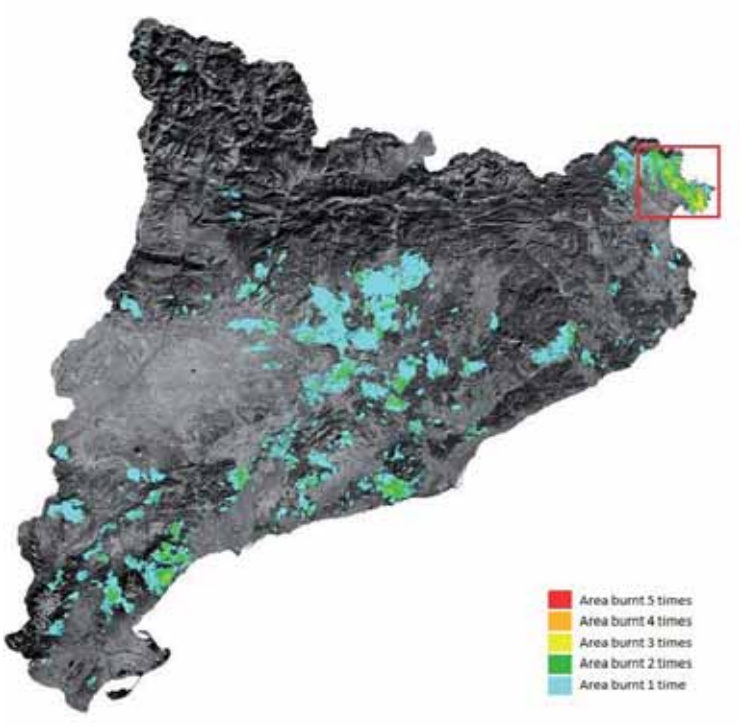

Fig. 5.4: Fire ocurrences in Catalonia in the period 1975-2010 (Fire sizes bigger or equal to 30 ha).

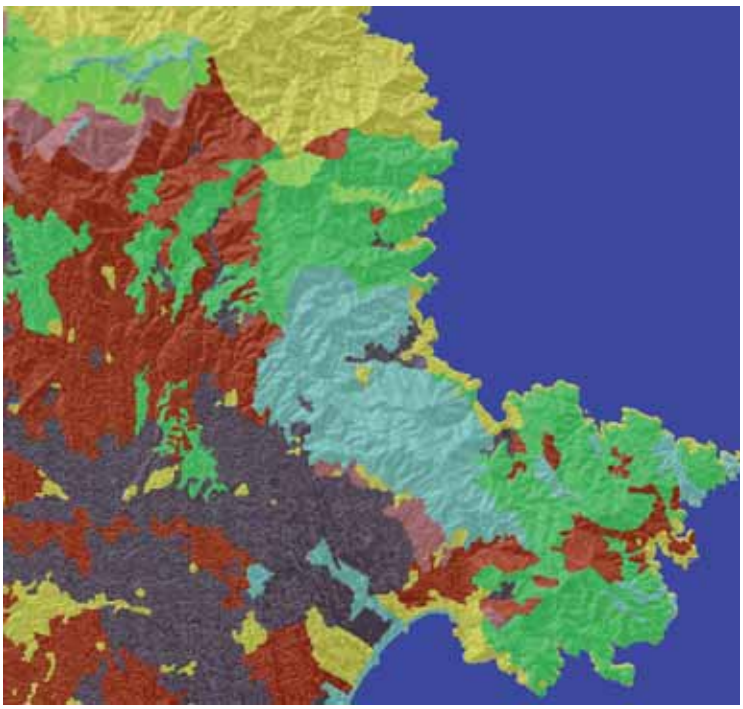

Fig. 5.5: *Cap de Creus*, Catalonia (image from FARSITE simulator). Different colors indicate different type of vegetation.

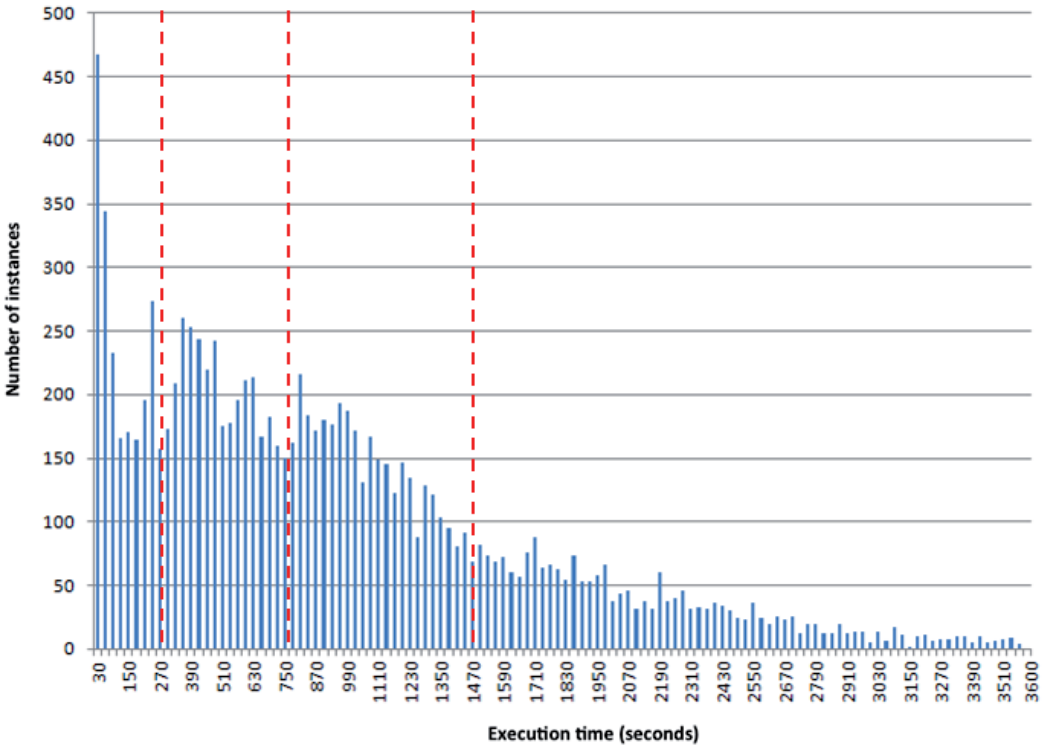

Fig. 5.6: Histogram of execution times using FARSITE in the case of *Cap de Creus* landscape. Vertical dotted lines indicate the defined classification boundaries.

Tab. 5.3: Correspondence between real and predicted classes using FARSITE in the case of *Cap de Creus* landscape.

		Predicted Class			
		A	B	C	D
Real Class	A	247	2	0	0
	B	0	263	8	0
	C	0	5	234	12
	D	0	0	9	220

Figure 5.6 shows the corresponding histogram of execution times. Again, we bound different classes by analyzing the different local minimums the histogram presents, so that we minimize the classification errors due to values too close to the boundaries.

Thus, the defined classes are the following, where *ET* stands for execution time:

- Class A: ET $\leq$ 270 seconds.
- Class B: 270 seconds $<$ ET $\leq$ 750 seconds.
- Class C: 750 seconds $<$ ET $\leq$ 1470 seconds.
- Class D: 1470 seconds $<$ ET $\leq$ 3600 seconds.

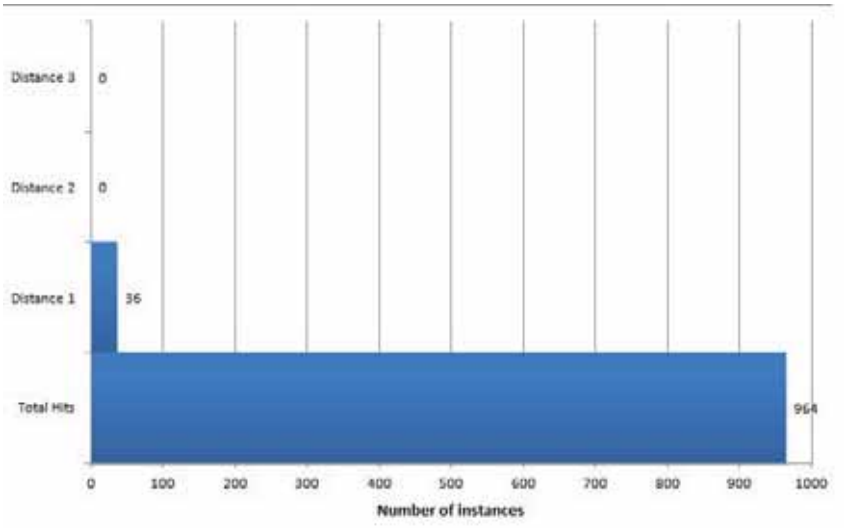

Fig. 5.7: Classification accuracy in the case of *Cap de Creus* landscape.

The results of applying decision trees to the test set are summarized in Table 5.3. It is worth noting the great prominence of the main diagonal, which means that perfect matches are absolutely predominant over the whole set of predictions.

Figure 5.7 shows the absolute values of the number of predictions that totally hit the real class, as well as the absolute values where the prediction had an accuracy determined by the distance between classes.

As can be observed when analyzing this graphic, we obtained a 96.4% correct classifications, and, if we consider *Distance 1* as a good prediction accuracy, then every classification was satisfactory.

5.2.2 Genetic Algorithm Characterization

As regards the Genetic Algorithm characterization, in this specific case, the analysis of the convergence as well as the statistical study were performed in the following terms:

- Populations composed of 25 and 100 individuals.
- Populations evolved over 10 generations.
- Mutation probability fixed to 10%.
- Crossover probability fixed to 25%.
- Initial fire consisting of a single initial ignition point.
- Adjustment time interval: 0h - 6h (from the ignition to 6 hours after).

Guarantee degree	G2	G3	G4	G5	G6	G7	G8	G9	G10
95%	2.06	1.72	1.58	1.51	1.36	0.888	0.79	0.752	0.585
90%	1.27	1.08	0.987	0.933	0.848	0.582	0.525	0.497	0.398
85%	0.92	0.782	0.719	0.673	0.615	0.438	0.399	0.357	0.307
80%	0.711	0.607	0.559	0.519	0.476	0.349	0.32	0.301	0.25
75%	0.57	0.488	0.45	0.415	0.383	0.288	0.265	0.248	0.209
70%	0.467	0.402	0.371	0.34	0.314	0.242	0.224	0.209	0.178
65%	0.388	0.335	0.31	0.283	0.262	0.206	0.192	0.179	0.154
60%	0.326	0.283	0.261	0.237	0.22	0.177	0.165	0.154	0.134
55%	0.275	0.239	0.222	0.2	0.186	0.152	0.143	0.133	0.117
50%	0.233	0.203	0.188	0.169	0.158	0.132	0.124	0.115	0.102

Tab. 5.4: Achievable error with different degrees of guarantee. Populations composed of 25 individuals.

Guarantee degree	G2	G3	G4	G5	G6	G7	G8	G9	G10
95%	0.552	0.325	0.289	0.211	0.164	0.164	0.138	0.121	0.113
90%	0.399	0.251	0.226	0.172	0.136	0.134	0.114	0.101	0.095
85%	0.32	0.212	0.191	0.149	0.12	0.116	0.1	0.09	0.085
80%	0.269	0.185	0.168	0.133	0.109	0.104	0.09	0.082	0.077
75%	0.232	0.164	0.15	0.121	0.099	0.095	0.083	0.076	0.071
70%	0.203	0.148	0.135	0.111	0.092	0.087	0.077	0.07	0.066
65%	0.179	0.134	0.123	0.103	0.086	0.08	0.071	0.066	0.062
60%	0.159	0.122	0.113	0.095	0.08	0.074	0.067	0.062	0.058
55%	0.142	0.112	0.103	0.089	0.075	0.069	0.062	0.058	0.055
50%	0.127	0.102	0.095	0.082	0.071	0.064	0.058	0.055	0.052

Tab. 5.5: Achievable error with different degrees of guarantee. Populations composed of 100 individuals.

Using these configurations for the GA, we carried out the evolution process for 50 different populations (in both cases of size equals to 25 individuals and size equal to 100).

Considering the quality of the adjustment, in this case the probability distribution which best fits the results obtained was the LogNormal distribution. By means of its probability density function, we determined the different degrees of guarantee of achieving a certain adjustment error, depending on the number of generations over which the GA is iterated. This information is summarized in Tables 5.4 and 5.5. In these tables, columns corresponding to the first generations are omitted, since they provide no useful information (initial populations are generated randomly). Figures 5.8 and 5.9 also depict this information.

In the subsequent subsection, we detail an experiment which is supposed to recreate a hypothetical real situation.

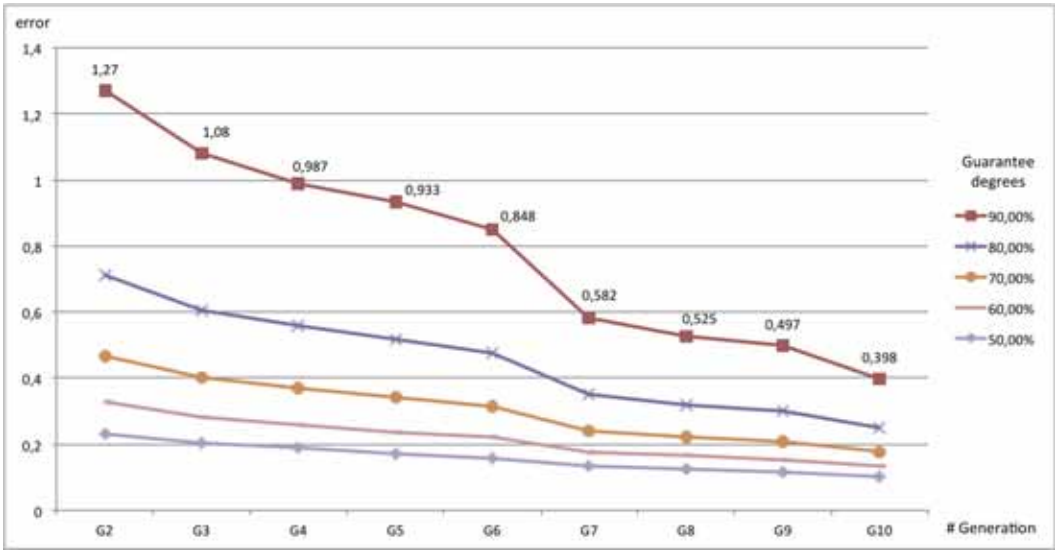

Fig. 5.8: Achievable error with different degrees of guarantee. Populations composed of 25 individuals.

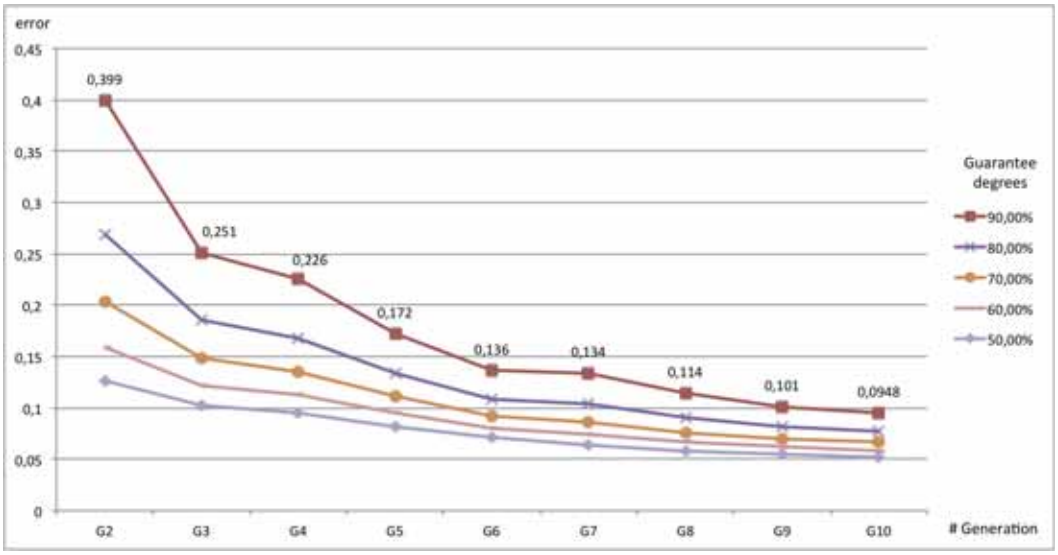

Fig. 5.9: Achievable error with different degrees of guarantee. Populations composed of 100 individuals.

5.2.3 Real Emergency Recreation

In order to prove the correctness of the above presented characterizations, in this subsection we consider an hypothetical situation based on the *Cap de Creus* landscape, where we have to meet the following restrictions regarding both the quality of the adjustment and the time available to perform it:

- Response time: one hour for the adjustment process.
- Quality of adjustment: a maximum error of approximately 0.5 is required.

In addition, let us consider that we count with the capability to execute 25 simulations at the same time (in a parallel way). Given these restrictions, and taking into account the previously discussed classification (Section 5.2.1), one can realize that if we discard individuals belonging to classes C and D in the whole adjustment process, we can evolve a population composed of 25 individuals over 5 generations in one hour, two minutes and thirty seconds, which would be appropriate.

By analyzing the data in Table 5.4, we assume that we could bring an error approximately equal to 0.5 with an 80% degree of guarantee (the error indicated is 0.519).

Based on this inference, we carried out an experiment consisting of the evolution of 30 populations composed of 25 individuals, where every individual belonging to classes C or D was automatically replaced with other one belonging to class A or B. At the fifth generation, we should obtain an error equal to or lesser than 0.519 in approximately the 80% cases.

Table 5.6 shows the results obtained from these evolutions. These evolution processes were extended up to 10 generations for further analysis purposes.

As can be seen, only 2 populations (p19 and p22) out of 30 exceeded an error of 0.5, which goes beyond the assumed 80% degree of guarantee. Moreover, by discarding individuals belonging to classes C and D, the response time restriction was also met.

Indeed, we can observe that in this experimental set only populations p19 and p22, in some generations, exceeded the errors corresponding to 90% degree of guarantee (see Table 5.4), which turns out to be an absolutely satisfactory result.

Population	G2	G3	G4	G5	G6	G7	G8	G9	G10
p0	0.447	0.414	0.265	0.265	0.124	0.124	0.08	0.08	0.072
p1	0.673	0.673	0.673	0.614	0.164	0.164	0.164	0.164	0.164
p2	0.251	0.251	0.251	0.18	0.025	0.025	0.025	0.025	0.025
p3	0.445	0.067	0.067	0.067	0.067	0.067	0.067	0.067	0.067
p4	0.235	0.125	0.099	0.035	0.035	0.035	0.019	0.019	0.005
p5	0.601	0.455	0.45	0.427	0.139	0.139	0.139	0.139	0.139
p6	0.417	0.417	0.417	0.246	0.246	0.219	0.12	0.12	0.105
p7	0.518	0.055	0.055	0.055	0.055	0.038	0.038	0.033	0.018
p8	0.716	0.411	0.122	0.061	0.044	0.044	0.044	0.044	0.044
p9	0.567	0.552	0.494	0.491	0.49	0.49	0.488	0.488	0.488
p10	0.742	0.694	0.59	0.353	0.351	0.351	0.351	0.351	0.351
p11	0.453	0.147	0.147	0.125	0.125	0.095	0.095	0.058	0.054
p12	0.31	0.31	0.31	0.31	0.241	0.234	0.097	0.096	0.096
p13	0.565	0.174	0.174	0.122	0.122	0.122	0.115	0.115	0.111
p14	0.341	0.339	0.223	0.223	0.217	0.217	0.217	0.117	0.117
p15	0.388	0.257	0.117	0.061	0.061	0.061	0.061	0.038	0.038
p16	0.75	0.414	0.272	0.272	0.248	0.248	0.245	0.179	0.038
p17	0.603	0.218	0.218	0.218	0.212	0.169	0.169	0.15	0.134
p18	0.507	0.48	0.342	0.224	0.22	0.197	0.197	0.197	0.034
p19	1.221	1.201	1.201	0.995	0.995	0.995	0.728	0.624	0.444
p20	0.448	0.448	0.448	0.23	0.23	0.17	0.17	0.17	0.092
p21	0.758	0.758	0.407	0.407	0.407	0.126	0.126	0.126	0.087
p22	1.01	1.01	1.01	1.01	0.876	0.612	0.612	0.312	0.259
p23	0.108	0.108	0.108	0.108	0.108	0.105	0.105	0.071	0.071
p24	0.214	0.111	0.111	0.111	0.104	0.104	0.104	0.086	0.086
p25	0.303	0.175	0.172	0.172	0.109	0.109	0.109	0.091	0.091
p26	0.637	0.549	0.444	0.042	0.039	0.037	0.037	0.037	0.032
p27	0.338	0.338	0.099	0.099	0.099	0.099	0.099	0.092	0.092
p28	0.75	0.51	0.155	0.155	0.066	0.066	0.066	0.066	0.016
p29	0.57	0.461	0.425	0.193	0.193	0.096	0.096	0.096	0.096

Tab. 5.6: Errors obtained from the evolution of 30 populations composed of 25 individuals in the case of *Cap de Creus* landscape. Every individual belonging to class C or D was replaced with other one belonging to class A or B.

6 Conclusions and Future Work

Natural hazard management is undoubtedly a relevant application area in which the Computational Science & Engineering and High Performance Computing fields can play a very important role.

As explained in Chapter 1.3, the matter of adequately managing these kinds of catastrophes and minimizing casualties and other losses in crisis situations is an issue which Civil Protection Agencies (CP) are in charge of. This thesis has been carried out bearing this fact in mind, and it has been focused on providing a valuable help to the Civil Protection Agencies. Concretely, it has been considered the specific case of dealing with an environvental emergency such as forest fires.

As has been discussed in Chapter 2, many solutions related to this issue have been developed, essentially involving fire spread simulation and prediction frameworks design. Nevertheless, in this kind of phenomena, it is common to have to deal with high degrees of uncertainty on the input parameters, which may lead us to important losses as regards the quality of the prediction. In this sense, in order to enhance the quality of the predictions while dealing with imprecise and uncertain input parameters, parameter calibration and estimation methods are applied.

The two-stage prediction scheme detailed in Chapter 2.3.2 [10, 20, 51] was introduced to enable parameter calibration and thus enhance the classical spread prediction result. This prediction framework highly improves the quality of the predictions in forest fire spread simulations but, due to the complexity of this schema, it could be very hard to know (or even to estimate) how much time will be necessary in such processes, as well as the ideal amount and type of computational resources to be used.

For this reason, we came up with the present project, which consists of determining in advance how a certain combination of a natural hazard simulator, computational resources, and adjustment strategy will perform in terms of execution time and prediction quality.

The contribution of this work is concerned with this issue. In Chapter 1.5, we formally set our main goal out as:

"*The establishment of a methodology for the quality and response time assessment of environmental emergency evolution prediction under real-time restrictions*"

On the one hand, since we are dealing with the area of natural hazards management, it is absolutely necessary to be able to assess in advance the quality of the predictions that will be delivered by means of our prediction framework. This is very important for the control centers to make the appropriate decisions in each case.

In this sense, we have presented our methodology to characterize a well-known Artificial Intelligence technique as an adjustment strategy, Genetic Algorithms, which has proven to be a powerful technique to perform the adjustment process in our two-stage prediction method. For this purpose, we have carried out a statistical study based on a huge set of simulations. Then, we have identified the probability distribution which corresponds to the results obtained, so that we can rely on its probability density function in order to establish certain degrees of guarantee in our adjustment errors estimations.

On the other hand, given the context of urgency where this work is framed, it is also absolutely necessary to take into account the time incurred for the prediction method, because the time needed to give a prediction becomes critical in these kinds of situations. For this purpose, our methodology also allows for the assessment the urgency-accuracy binomial in each particular case.

As is well known, the execution time of a particular simulator depends on the specific setting of the input parameters. However, as has been discussed, it becomes hard to predict how certain variations on certain input parameters would affect the execution time. The use of decision trees as a classification technique resulted in a very profitable strategy to tackle this problem, reaching accuracy results of up to 99.8% satisfactory classification prediction.

Moreover, we demonstrate that the capability to detect certain input settings (that would make simulations last too long) helps us save an important amount of time (which in the specific case of natural hazards management becomes crucial) without affecting the accuracy of the results. We also emphasize the need to count on efficient Urgent Computing techniques in order to be able to fulfill the requirements of the decision control centers, both in terms of quality and time restrictions.

In addition, this work has been carried out using different fire simulators (FireLib, FireStation, FARSITE) and different real topographic areas in order to highlight the flexibility of our proposed methodology, which may be extrapolated to any other scenario, and even to any other kind of natural hazard.

6.1 Conclusions

The development of this thesis has been carried out in an incremental way. In the first steps, the focus of the work was to identify which factors determine the behavior of the two-stage strategy in terms of quality.

At this point, and taking into account the fact that the quality of final predictions and the quality of the adjustment maintain a close relationship (see Chapter 3), it was easy to understand that for a proper characterization of the whole prediction system, it is compulsory to characterize the adjustment strategy we are using.

Bearing this premise in mind, we started the design of the methodology. The first and basic principles were presented in a couple of publications:

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Metodología para el diseño de modelos de estimación del comportamiento de aplicaciones HPC en entornos de tiempo real*. Actas del Congreso Español de Informática 2010.

Andrés Cencerrado, Roque Rodríguez, Ana Cortés, Tomás Margalef. *Enhancing Wildland fire spread prediction through dynamic data-injection techniques*. Proceedings of the International Conference on Forest Fire Research 2010 (CD-ROM).

Together with the design of the first steps of the methodology, the important question of how to deal with the important time restrictions the prediction systems are subjected to promptly arose. The preliminary approaches of how we planned to tackle this matter were presented in:

Andrés Cencerrado, Roque Rodríguez, Ana Cortés, Tomás Margalef. *Urgency versus Accuracy: Dynamic Data Driven Application System for Natural Hazard Management*. International Journal of Numerical Analysis and Modeling, Volume 9, Number 2, Pages 432-448. 2012.

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Prediction Time Assessment in a DDDAS for Natural Hazard management: Forest Fire Study Case*. Proceedings of the International Conference on Computational Science, ICCS 2011. Procedia Computer Science, Volume 4, 2011, Pages 1761-1770.

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Predicting Natural Hazards Evolution: How to Overcome the Impact of Input-parameter Uncertainty*, Proceedings of the 18th RCRA workshop (International Joint Conference on Artificial Intelligence 2011).

Subsequently, we decided to concentrate on the Genetic Algorithm as the adjustment strategy to be characterized because of the power and effectiveness it demonstrated in previous works such as [10, 11, 20]. The fruits of labor in this sense were:

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Genetic Algorithm Characterization for the Quality Assessment of Forest Fire Spread Prediction*. To be published on Procedia Computer Science (International Conference on Computational Science 2012).

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *On the Way of Applying Urgent Computing Solutions to Forest Fire Propagation Prediction*. To be published on Procedia Computer Science (International Conference on Computational Science 2012).

The last one also gets back to the problem of how to efficiently deal with time restrictions. This matter has been the focus of many of our efforts, and the obtained progress is currently under review in the following publication:

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Dealing with Time Constraints on Parameter Tuning on Natural Hazards Evolution Prediction* Artificial Intelligence Communications. Submitted (under 2nd revision).

The results obtained from the experiments presented in Chapter 5.2 are also under revision in:

Andrés Cencerrado, Ana Cortés, Tomás Margalef. *Time Response and Quality Assessment in Forest Fire Spread Prediction: a Case Study of the Cap de Creus - Spain* Environmental Modelling and Software. Submitted.

At present, we are working on a full description and experimentation of the whole methodology which, besides producing this PhD thesis document, we intend to include in further international publications.

6.2 Open Lines

As can be analyzed in Chapters 3 and 5, the results obtained in this long-term research work are fully satisfactory and give rise to several new challenges.

In Chapter 3.2.1, we set the Genetic Algorithm as the adjustment strategy, because of its effectiveness, as well as the implicit features, which turn it into a very challenging case. However, the two-stage prediction method described in Chapter 2.3.2 has been put to the test counting on alternative adjustment techniques, such as statistical analysis [10], case-based reasoning [63], or even merging statistical and evolutionary techniques [51].

The flexibility of the proposed methodology allows us to follow the same steps to determine which factors affect the convergence of the quality of the results. However, these alternative strategies entail a great test for the proposed methodology, so it has been planned to test it in the near future.

On the other hand, despite the fact that the most time-demanding processes within the methodology are designed to be performed in an *offline* fashion, it is necessary to repeat the whole process for each topographic area we might want to control. Currently, an intensive research study is being carried out in order to find suitable mechanisms to extrapolate the estimations for a specific area to unknown (or not analyzed following the methodology) ones, minimizing the eventual loss in accuracy.

Furthermore, these studies allow us to tackle the problem of attending to an emergency in different ways, by designing different policies to optimize the use of the available computational resources. It would be an important advantage to be able to dynamically group the fastest simulations in subsets of computational resources, allocating the slowest ones to other dedicated subsets, according to the specific needs of each case. It is obvious that the experience obtained from the development of our strategy to assess in advance how long each simulation will last will be highly valuable and opens up this new challenge with good expectations and a guaranteed background.

Bibliography

[1] *2007 disasters in numbers*. Centre for Research on the Epidemiology of Disasters. International Strategy for Disaster Reduction (UN/ISDR), 2008.

[2] B. Abdalhaq. „A methodology to enhance the Prediction of Forest Fire Propagation“. PhD thesis. Universitat Autònoma de Barcelona, 2004.

[3] S.D. Aberson. „Five-day tropical cyclone track forecasts in the North Atlantic basin“. In: *Weather and Forecasting* 13 (1998), pages 1005–1015.

[4] F.A. Albini. *Estimating wildfire behavior and effects*. Gen. Tech. Rep. INT-GTR-30. Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station. Ogden, UT (US), 1976.

[5] G. Allen, P. Bogden, T. Kosar, A. Kulshrestha, G. Namala, S. Tummala, and E. Seidel. „Cyberinfrastructure for Coastal Hazard Prediction“. In: *CTWatch Quarterly* 4(1) (2008), pages 17–26.

[6] P.L. Andrews. *BEHAVE: Fire Behavior prediction and modeling systems - Burn subsystem*. General Technical Report INT-194. Ogden TU (US), 1986.

[7] P. Barbosa, A. Camia, J. Kucera, G. Liberta, I. Palumbo, J. San Miguel Ayanz, and G. Schmuck. „Assessment of Forest Fire Impacts and Emissions in the European Union Based on the European Forest Fire Information System, Wildland Fires and Air Pollution“. In: *Developments in Environmental Science* 8(8) (2008), pages 197–208.

[8] P. Beckman. „Urgent Computing: Exploring Supercomputing's New Role“. In: *CTWatch Quarterly* 4(1) (2008), pages 3–4.

[9] P. Beckman, S. Nadella, N. Trebon, and I. Beschastnikh. „A System for Supporting Urgent High-Performance Computing“. In: *Grid-Based Problem Solving Environments* 239/2007 (2007), pages 295–311.

[10] G. Bianchini, A. Cortés, T. Margalef, and E. Luque. „Improved Prediction Methods for Wildfires Using High Performance Computing A Comparison“. In: *Lecture Notes in Computer Science*. Volume 3991. 2006, pages 539–546.

[11] G. Bianchini, M. Denham, A. Cortés, T. Margalef, and E. Luque. „Wildland Fire Growth Prediction Method Based on Multiple Overlapping Solution“. In: *Journal of Computational Science* 1(4) (2010). Edited by Elsevier Science, pages 229–237.

[12] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[13] R. Buyya and S. Venugopal. „A Gentle Introduction to Grid Computing and Technologies“. In: *Proceedings of the 12th WSEAS International Conference on Applied Mathematics*. 2007, pages 417–423.

[14] D. Caballero, G. Xanthopoulos, D. Kallidromitou G. Lyrintzis, M. Bonazountas, P. Papachristou, and O. Pacios. „FOMFIS: Forest Fire Management and Fire Prevention System“. In: *Proceedings of DELFI Intl. Symp. Forest Fires Needs and Innovations*. 1999, pages 93–98.

[15] A. Camia, G. Amatulli, and J. San-Miguel-Ayanz. *Paste and Future Trends of Forest Fire Danger in Europe*. EUR 23427 EN. Luxembourg (Luxembourg): OPOCE, 2008.

[16] R.E. Clark, A.S. Hope, S. Tarantola, D. Gatelli, P.E. Dennison, and M.A. Moritz. „Sensitivity Analysis of a Fire Spread Model in a Chaparral Landscape“. In: *Fire Ecology* 4(1) (2004), pages 1–13.

[17] *CREAF website*. http://www.creaf.uab.es/eng/index.htm (Accessed May 2012).

[18] F. Darema. „Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements“. In: *Lecture Notes in Computer Science*. Volume 3038. 2004, pages 662–669.

[19] F. Darema. „Grid Computing and Beyond: The Context of Dynamic Data Driven Applications Systems“. In: *Proceedings of the IEEE*. Volume 93(3). 2005, pages 692–697.

[20] M. Denham, A. Cortés, and T. Margalef. „Computational Steering Strategy to Calibrate Input Variables in a Dynamic Data Driven Genetic Algorithm for Forest Fire Spread Prediction“. In: *Lecture Notes in Computer Science*. Volume 5545(2). 2009, pages 479–488.

[21] Pastor E., Zárate L., E. Planas, and Arnaldos J. „Mathematical models and calculation systems for the study of wildland fire behaviour“. In: *Progress in Energy and Combustion Science* 29(2) (2003), pages 139–153.

[22] *EasyFit software - Mathwave website*. http://www.mathwave.com/ (Accessed May 2012).

[23] *EFFIS, European Forest Fires Information System EFFIS*. http://effis.jrc.ec.europa.eu (accessed April 2012). 2010.

[24] *EM-DAT, 2010. The International Disaster Database*. www.emdat.be (accessed April 2012). Centre for Research on Epidemiology of Disasters (CRED), 2010.

[25] M.A. Finney. *FARSITE: Fire Area Simulator-model development and evaluation*. Res. Pap. RMRS-RP-4. Ogden TU (US), 1998.

[26] *fireLib User Manual and Technical Reference*. http://www.fire.org/downloads/fireLib/1.0.4/doc.html.

[27] *Firemodels.org. U.S. Dept. of Agriculture*. http://www.firemodels.org (Accessed April 2012).

[28] *FIRE.ORG - Public Domain Software for the Wildland fire Community*. http://www.fire.org/.

[29] P. Fleming and R. Purshouse. „Evolutionary algorithms in control systems engineering: a survey“. In: *Control Engineering Practice* 10 (2002), pages 1223–1241.

[30] *FUEGO Project User's requirements. Final Report of the FUEGO Project*. http://www.dlr.de/iaa.symp/Portaldata/49/Resources/dokumente/archiv3/0601.pdf (accessed April 2012). 2000.

[31] G. Holmes, A. Donkin, and I.H. Witten. „Weka: A machine learning workbench“. In: *Proceedings of the Second Australia and New Zealand Conference on Intelligent Information Systems*. 1994, pages 357–361.

[32] K. Kalabokidis, G. Xanthopoulos, P. Moore, D. Caballero, G. Kallos, J. Llorens, O. Roussou, et al. „Decision support system for forest fire protection in the Euro-Mediterranean region“. In: *European Journal of Forest Research* 131(3) (2012), pages 597–608.

[33] B. Kirk, G. Palmer, C. Tang, and W. Wood. „Urgent Computing in Support of Space Shuttle Orbiter Reentry“. In: *CTWatch Quarterly* 4(1) (2008), pages 27–34.

[34] S.B. Kotsiantis. „Supervised Machine Learning: A Review of Classification Techniques“. In: *Informatica* 31(2007) (2007), pages 249–268.

[35] „La mejora del mapa diario de riesgo de incendio forestal en Cataluña“. In:

[36] K. Leyton-Brown, E. Nudelman, and Y. Shoham. „Empirical Hardness Models: Methodology and a Case Study on Combinatorial Auctions“. In: *Journal of the ACM (JACM)* 56(4) (2009), pages 1–52.

[37] A. Lopes, M. Cruz, and D. Viegas. „FireStation - An integrated software system for the numerical simulation of fire spread on complex toography“. In: *Environmental Modelling and Software* 17(3) (2002), pages 269–285.

[38] N . Lott and T . Ross. *Tracking and Evaluating U.S. Billion Dollar Weather Disasters, 1980-2005*. http://www1.ncdc.noaa.gov/pub/data/papers/200686ams1.2nlfree.pdf. National Climatic Data Center (NCDC), 2006.

[39] H. Madsen and F. Jakobsen. „Cyclone induced storm surge and flood forecasting in the northern Bay of Bengal“. In: *Coastal Engineering* 51(4) (2004), pages 277–296.

[40] S. Manos, S. Zasada, and P. Coveney. „Life or Death Decision-making: The Medical Case for Large-scale, On-demand Grid Computing“. In: *CTWatch Quarterly* 4(1) (2008), pages 35–45.

[41] *Mapping the impacts of natural hazards and technological accidents in Europe*. European Environment Agency Technical report 13/2010. 2010.

[42] A. Marczyk. *Genetic Algorithms and Evolutionary Computation*. http://www.talkorigins.org/faqs/genalg/genalg.html (accessed May 2012).

[43] S. Marru, D. Gannon, S. Nadella, P. Beckman, D. Weber, K. Brewster, and K. Droegemeier. „LEAD Cyberinfrastructure to Track Real-Time Storms Using SPRUCE Urgent Computing“. In: *CTWatch Quarterly* 4(1) (2008), pages 5–16.

[44] *Minitab website*. http://www.minitab.com/en-US/default.aspx (Accessed May 2012).

[45] M. Mitchell. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[46] *NatCatSERVICE*. http://www.munichre.com/en/reinsurance/business/non-life/georisks/natcatservice/default.aspx (accessed April 2012). Munich Re NatCatSERVICE, 2010.

[47] *NOAA website*. http://www.nesdis.noaa.gov (accessed April 2012).

[48] E. Noonan-Wright, T. Opperman, M. Finney, G. Zimmerman, R. Seli, L. Elenz, D. Calkin, et al. „Developing the US Wildland Fire Decision Support System“. In: *Journal of Combustion* 2011. 14p (2011).

[49] K. Outcalt. *Forest encyclopedia network. USDA Forest Service, Athens, GA*. http://www.forestencyclopedia.net (Accessed May 2012).

[50] J.R. Quinlan. „Improved use of continuous attributes in c4.5“. In: *Journal of Artificial Intelligence Research* 4 (1996), pages 77–90.

[51] R. Rodríguez, A. Cortés, and T. Margalef. „Injecting Dynamic Real-Time Data into a DDDAS for Forest Fire Behavior Prediction“. In: *Lecture Notes in Computer Science*. Volume 5545(2). 2009, pages 489–499.

[52] R.C. Rothermel. *A mathematical model for predicting fire spread in wildland fuels*. Res. Pap. INT-115. Ogden TU (US): USDA FS, 1972.

[53] R.C. Rothermel. „How to Predict the Spread and Intensity of Forest and Range Fires“. In: Gen. Tech. Rep. INT-143. Ogden TU: USDA FS, 1983, pages 1–5.

[54] R. Salvador and X. Pons R. Díaz-Delgado J. Valeriano. „Remote Sensing of Forest Fires“. In: *Proceedings of GIS PlaNET'98 International Conference and Exhibition on Geographic Information (CD-ROM)*. 1998.

[55] J. San-Miguel-Ayanz, A. Camia, G. Liberta, and R. Boca. *Analysis of Forest Fire Damages in Natura 2000 Sites during the 2007 Fire Season*. EUR 24086 EN. Luxembourg (Luxembourg): European Commission, 2009.

[56] *Shodor. Cserd: What is computational science?* http://www.shodor.org/cserd/Help/whatiscs (accessed April 2012). 1994.

[57] *Spruce: Urgent computing for supercomputers*. http://spruce.uchicago.edu.

[58] N. Trebon. „Enabling Urgent Computing within the Existing Distributed Computing Infrastructure“. PhD thesis. University of Chicago, US, 2011.

[59] *VENUS-C (Virtual multidisciplinary EnviroNments USing Cloud infrastructures) project website*. http://www.venus-c.eu/Pages/Home.aspx (Accessed May 2012).

[60] D. Viegas, A. Simeoni, G. Xanthopoulos, C. Rossa, L. Ribeiro, L. Pita, D. Stipanicev, et al. *Recent Forest Fire Related Accidents in Europe*. EUR 24121 EN. Luxembourg (Luxembourg): Publications Office of the European Union, 2009.

[61] H.C. Weber. „Hurricane Track Prediction Using a Statistical Ensemble of Numerical Models“. In: *Monthly Weather Review* 131 (2003), pages 749–770.

[62] G. Wells. „The Rothermel fire-spread model: Still running like a champ“. In: *JFSP Fire Science Digest* 2 (2008), pages 1–11.

[63] K. Wendt, A. Cortés, and T. Margalef. „Knowledge-guided Genetic Algorithm for input parameter optimisation in environmental modelling“. In: *Procedia Computer Science*. Volume 1(1). 2010, pages 1361–1369.

[64] *Wikipedia - Computational science. Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Computational_science (accessed May 2012).

[65] *Wikipedia - Decision Tree Learning. Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Decision_tree_learning (accessed May 2012).

[66] *Wikipedia - Genetic Algorithm. Wikipedia, the free encyclopedia*. http://en.wikipedia.org/wiki/Genetic_algorithm (accessed May 2012).

[67] B. Wisner, P. Blaikie, T. Cannon, and I. Davis. *At Risk - Natural hazards, people's vulnerability and disasters*. Routledge, Wiltshire, 2004. ISBN: 0-415-25216-4.

[68] G. Xanthopoulos, V. Varela, P. Fernandes, L. Ribeiro, and F. Guarneri. *Decision support systems and tools: a state of the art*. Deliverable D-06-02. EUFIRELAB, Euro-Mediterranean Wildland Fire Laboratory. 2004.

[69] L. Youseff, M. Butrico, and D. Da Silva. „Toward a Unified Ontology of Cloud Computing“. In: *Grid Computing Environments Workshop*. 2008, pages 1–10.

List of Figures

List of Tables

Colophon

This research has been supported by the MICINN-Spain under contract TIN2007-64974 and contract TIN2011-28689-C02-01.

This thesis was typeset with LaTeX 2_ε using the *Clean Thesis* style. The design of this style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.

Declaration

I hereby declare that I completed this work solely and only with the help of the references mentioned in the bibliography.

Bellaterra, Barcelona, May 4th, 2012

Andrés Cencerrado Barraqué