Universidad Carlos III de Madrid

Ph.D. Thesis

High Performance Computing Techniques Applied to the Design of Complex Railway Infrastructures

Author: Carlos Gómez Carrasco
Advisors: Jesús Carretero Pérez
Félix García Carballeira

Computer Science and Engineering Department
Leganés, October 2016

Universidad Carlos III de Madrid

Tesis Doctoral

Técnicas de altas prestaciones aplicadas al diseño de infraestructuras ferroviarias complejas

Autor: Carlos Gómez Carrasco
Directores: Jesús Carretero Pérez
Félix García Carballeira

Departamento de Informática
Leganés, Octubre 2016

TESIS DOCTORAL

Técnicas de altas prestaciones aplicadas al diseño de infraestructuras ferroviarias complejas

AUTOR:	**Carlos Gómez Carrasco**
DIRECTORES:	**Félix García Carballeira**
	Jesús Carretero Pérez

TRIBUNAL CALIFICADOR

PRESIDENTE:

VOCAL:

SECRETARIO:

CALIFICACION:

Leganés, a de de 2016

A mis padres y mi abuelo Luis.

Cada "tic-tac" es un segundo de la vida que pasa
huye, y no se repite. Y hay en ella tanta intensidad,
tanto interés que el problema es solo saber vivir.
Que cada uno lo resuelva como pueda.

Frida Kahlo

El abajo firmante llora bajo la lluvia
mira su teléfono y sueña con un aforismo
que ilumine el camino,
su nombre con almohadilla, un pájaro
que trine al cautivo sin ballestero.

Ismael Serrano

Agradecimientos

Es muy difícil agradecer un trabajo como este debido a que no se trata simplemente del fin del doctorado, con este libro se cierra una de las etapas más importantes de mi vida. A lo largo de todo este tiempo he pasado por buenos y malos momentos, y durante algún tiempo pensé que este momento no llegaría nunca.

Podría dedicar a esta sección páginas y páginas, y es por ello que seguramente me deje sin mencionar a gente que ha formado parte de mi vida en todo este trayecto. Espero que nadie se sienta ofendido, si has formado parte de este trabajo, GRACIAS. Ahora vamos a los agradecimientos más personales.

A mis padres, sin ellos nada de esto habría sido posible. Gracias por aguantarme y apoyarme todos estos años.

A mi abuelo Luis, he llegado a ser la persona que soy hoy gracias a él y a todos los momentos que hemos pasado a lo largo de mi vida.

A mis tíos Luisa y Rufino, por todos los fines de semana que me han aguantado a lo largo de mi vida y últimamente por ser los canguros de Zeus.

A "Los Colegas", Raúl y Alfredo, hemos pasado muchas cosas a lo largo de este tiempo y me habéis insistido y apoyado en todo este tiempo. Muchas gracias por todos esos momentos increíbles.

A Alberto, gracias a ti he llegado hasta aquí. Hemos trotado y pasado por todas las fases de la universidad y has sido un apoyo increíble. Un Abrazo tío y HAY GOOOL!!!

Al equipo SIRTE al completo, y muy especialmente a Alberto Cascajo, parece que al final hemos conseguido cerrar esto (o no).

Al grupo del laboratorio, Javi, Estefanía, Silvina y Saúl. Muchas gracias por acogerme en el lab y aguantar mis chorradas.

Al grupo de torturados por la tesis, Rafa y Fran. Muchas gracias por todo, hemos compartido unos momentos increíbles a lo largo de estos años.

A David, hemos pasado muchos y muy buenos momentos, paseos perrunos y charlas por gtalk. Muchas gracias por escucharme.

A Gabriel, las charlas contigo han sido de lo más inspiradoras. Algún día iré para allá y nos tomaremos esas pintas que nos debemos golfo.

A todo el grupo de ARCOS, a lo largo de estos años hemos pasado juntos muchas cosas y he aprendido mucho con vosotros.

A la gente de los grupos "BBQ", "3 guys,1 girl and d observer", "Los gilipollas de la uc3m" y "Mujeres, Mojitos, Mojacar". A lo largo de todos estos años me habéis alegrado y distraído muchos días a lo largo de estos años.

A Jesús y Félix, no sólo por ser mis directores de tesis, a lo largo de estos años han sido mis profesores, jefes, directores y amigos. Ha sido un placer trabajar con vosotros todos estos años.

Carlos Gómez Carrasco

Abstract

In this work we will focus on overhead air switches design problem. The design of railway infrastructures is an important problem in the railway world, non-optimal designs cause limitations in the train speed and, most important, malfunctions and breakages. Most railway companies have regulations for the design of these elements. Those regulations have been defined by the experience, but, as far as we know, there are no computerized software tools that assist with the task of designing and testing optimal solutions for overhead switches. The aim of this thesis is the design, implementation, and evaluation of a simulator that that facilitates the exploration of all possible solutions space, looking for the set of optimal solutions in the shortest time and at the lowest possible cost.

Simulators are frequently used in the world of rail infrastructure. Many of them only focus on simulated scenarios predefined by the users, analyzing the feasibility or otherwise of the proposed design. Throughout this thesis, we will propose a framework to design a complete simulator that be able to propose, simulate and evaluate multiple solutions. This framework is based on four pillars: compromise between simulation accuracy and complexity, automatic generation of possible solutions (automatic exploration of the solution space), consideration of all the actors involved in the design process (standards, additional restrictions, etc.), and finally, the expert's knowledge and integration of optimization metrics.

Once we defined the framework different deployment proposes are presented, one to be run in a single node, and one in a distributed system. In the first paradigm, one thread per CPU available in the system is launched. All the simulators are designed around this paradigm of parallelism. The second simulation approach will be designed to be deploy in a cluster with several nodes, MPI will be used for that purpose. Finally, after the implementation of each of the approaches, we will proceed to evaluate the performance of each of them, carrying out a comparison of time and cost. Two examples of real scenarios will be used.

Resumen

El diseño de agujas aéreas es una problema bastante complejo y crítico dentro del proceso de diseño de sistemas ferroviarios. Un diseño no óptimo puede provocar limitaciones en el servicio, como menor velocidad de tránsito, y lo que es más importante, puede ser la causa principal de accidentes y averías. La mayoría de las compañías ferroviarias disponen de regulaciones para el diseño correcto de estas agujas aéreas. Todas estas regulaciones han sido definidas bajo décadas de experiencia, pero hasta donde sé, no existen aplicaciones software que ayuden en la tarea de diseñar y probar soluciones óptimas. Es en este punto donde se centra el objetivo de la tesis, el diseño, implementación y evaluación de un simulador capaz de explorar todo el posible espacio de soluciones buscando el conjunto de soluciones óptimas en el menor tiempo y con el menor coste posible.

Los simuladores son utilizados frecuentemente en el mundo de la infraestructura ferroviaria. Muchos de ellos sólo se centran en la simulación de escenarios preestablecidos por el usuario, analizando la viabilidad o no del diseño propuesto. A lo largo de esta tesis, se propondrá un framework que permita al simulador final ser capaz de proponer, simular y evaluar múltiples soluciones. El framework se basa en 4 pilares fundamentales, compromiso entre precisión en la simulación y la complejidad del simulador; generación automática de posibles soluciones (exploración automática del espacio de soluciones), consideración de todos los agentes que intervienen en el proceso de diseño (normativa, restricciones adicionales, etc.) y por último, el conocimiento del experto y la integración de métricas de optimización.

Una vez definido el framework se presentarán varias opciones de implementación del simulador, en la primera de ellas se diseñará e implementará una versión con hilos pura. Se lanzará un hilo por cada CPU disponible en el sistema. Todo el simulador se diseñará en torno a este paradigma de paralelismo. En un segundo simulador, se aplicará un paradigma mucho más pensado para su despliegue en un cluster y no en un único nodo (como el paradigma inicial), para ello se empleará MPI. Con esta versión se podrá adaptar el simulador al cluster en el que se va a ejecutar. Por último, se va a emplear un paradigma basado en cloud computing. Para ello, según las necesidades del escenario a simular, se emplearván más o menos máquinas virtuales.

Finalmente, tras la implementación de cada uno de los simuladores, se procederá a evaluar el rendimiento de cada uno de ellos, realizando para ello una comparativa de tiempo y coste. Se empleará para ello dos ejemplos de escenarios reales.

Contents

List of Figures

List of Tables

List of Algorithms

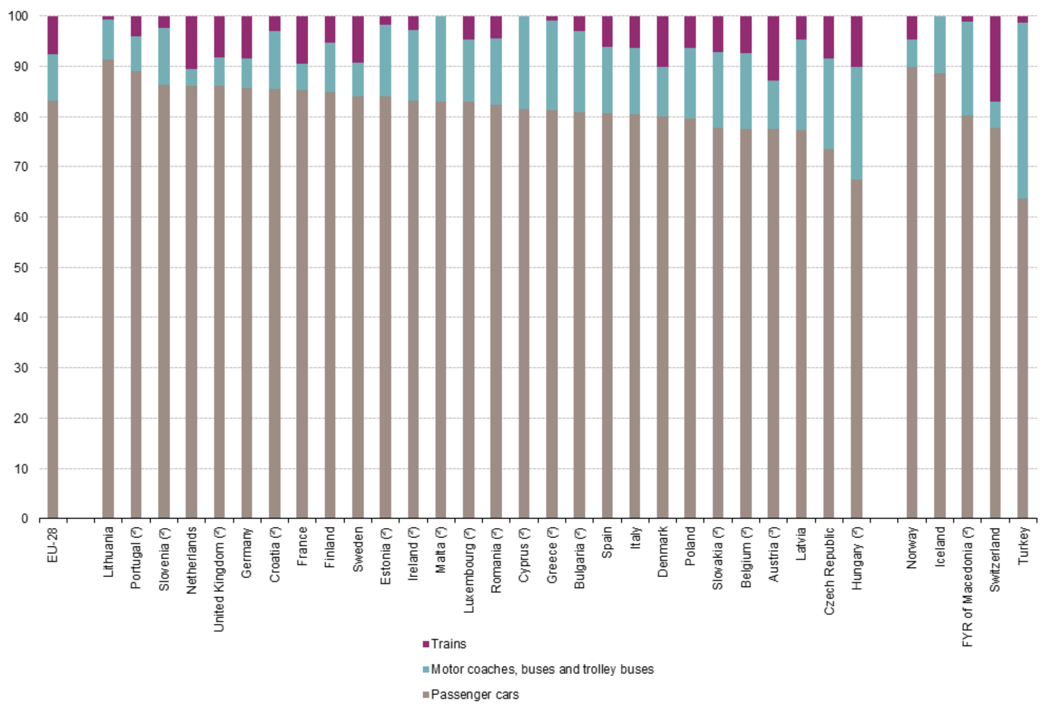
Chapter 1

Introduction

Since the mid-twentieth century, railway companies tend to deploy overhead lines (catenary) as a mechanism for supplying energy to electric locomotives, in detriment of other supply forms [Gri56, Ros71]. The use of the catenary has significant advantages over other kinds of traction. First, this approach has less impact on environment. Second it performs a higher power-to-weight ratio. Finally it is safer in terms of accidental contacts (humans or animals [Cle72, Har08]). However, overhead line systems have a complex infrastructure and require significant capital expenditure regarding the installation costs [Bro27, CW53, JOPLGC06]. Moreover, the higher the speed of the train, the greater the costs of implementation and safety measures [DI93].

As we can see in Figure 1.1, the percentage of railway passenger is lower than air passenger or road passenger. However if we look at the transport rush hours in major cities such as Madrid, Barcelona, etc., it can be seen as the train, metro or tram occupy a much higher position in that comparison. This information is obtained from Eurostat.

Since design and deployment tasks of overhead lines are a complex engineering problem, many elements must be considered. The route and camber of tracks, the pantograph geometry, the contact wires and messenger wires, the catenary support poles and portal frames, and the cantilevers to hold the wires, must be laid out so as to guarantee a correct electricity supply to the train (i.e., to ensure that the pantograph never loses contact with any contact wire). Other parameters, such as wind influence, pantograph, and contact wires wear, are also considered in the general problem. The focus of this thesis will be on the design of the overhead switches. The pantograph is an apparatus mounted on the roof of an electric train, tram or electric bus[1] to collect power through contact with an overhead catenary wire.



(¹) Excluding powered two-wheelers. Cyprus, Malta and Iceland: railways not applicable.
(²) Includes estimates or provisional data.
(³) The railway in Liechtenstein is owned and operated by the Austrian ÖBB and included in their statistics.
Source: Eurostat (online data code: tran_hv_psmod)

Figure 1.1: Modal split of inland passenger transport, 2013.

1.1 Motivation

The design of complex railway portals is a very complex process that requires knowledge and experience to ensure that the designed superstructure will not only be economic and regulation compliant, but will also cater for all current load cases, cable routing and supports, etc. Usually, most companies have a small database of standardized portals that are used for all situations. Traditional Warren and Vierendeel designs are still the most cost effective solutions to long span portals design, thus most of the solutions are based in those structures.

In addition, switches are critical elements in railway networks [RG95], as they allow the trains to change their trajectory from one outgoing track to another incoming track. Although the functionality of a switch does not change, there may be several kinds of switches with specific characteristics and geometries for different track speeds, different configurations for the straight track, etc. These heterogeneity results in the complex design task, since many mechanical aspects have their influence on the track change process. Wrong designs may lead to train derailments and hazardous breakdowns, indeed.

When considering switches on electrified tracks, the overhead line air switches mechanism comes into play. It allows the pantograph to lose the contact with the outgoing catenary of the straight track and to get contact with the incoming catenary of the diverging track. Therefore, the pantograph has to change progressively the catenary being rubbed against, always respecting the requirement of continuous friction with the contact wire so as to avoid electricity supply notches. Overhead line air switches tends to be considered a crucial issue in catenary-based railway systems. They are common points of failure due to wrong designs, thus requiring a regular maintenance. Those failures have an impact on security, reliability, quality of service, and economical costs.

1.2 Objectives

The major objective of this thesis is to propose a methodology that facilitates the design based parametric simulations in HPC environments. This methodology will be applied to develop a simulator that allows to look a set of optimal configurations in railway infrastructure.

Additionally, the thesis targets the following objectives:

O1 Designing a framework for railway simulations in HPC. It will be necessary to design a generator of possible solutions that work with an initial design and a set of variable parameters. In order to decide which solutions are best and which are worse must be designed a set of evaluation rules.

O2 Designing an ontology that facilitate the design based parametric simulations To design a correct infrastructure is required rules to guide the selection of elements. In the railway world a large number of valid designs would be possible with the same initial scenario.

O3 Building the infrastructure simulator. Having defined the mathematical model and ontology design, it will design and implement a simulator to find a valid solution.

O4 Building the overhead air switch simulator. Having defined the mathematical model for a catenary simulator, it will design and implement a simulator to find a set of valid solutions.

1.3 Document Structure

The thesis is structured as follows.

- Chapter 2 *State of the art* contains the state of the art related to this thesis. This chapter provides an overview of the current situation in the field of railway simulators, the principles of high performance computing are described. Chapter is structured as follows, first the principles of high performance computing are detailed, dividing the section between distributed computing and parallel computing paradigms. Then a small sample of main simulators in the field of railways are summarized. Finally an overview of the world of railroads today is exposed.

- Chapter 3 *Railway Simulation Framework* describes the generic structure of a generic railway simulator.
- Chapter 4 *Railway Simulator* describes the design problem and the lack of simulation. This is one of the main chapters of the thesis. The design of railway infrastructures is a very complex task and requires careful because the trains run at high speeds and an accident can be very dangerous. The basic principles of design and calculation are set, and the ontology used in the design and simulation of switches and structures is presented.Finally, describes in detail the principles and considerations that have been followed for the design and calculation to provide a complete design of structures. The necessary changes that have been made on the general framework described.
- Chapter 5 *Overhead Air Switches Design.* This chapter describes the restrictions and considerations that have been taken throughout the design process. The generation and evaluation algorithms of new possible solutions are described. Finally the evaluation of this simulator is presented, two versions of the simulator (mpi, and shared memory) will be developed for this evaluation.
- Chapter 6 *Experimental Results* reports performance results. The evaluation has been defined in two distinct phases, first, each simulator has been evaluated separately. Secondly several test scenarios have been defined and evaluated and tested with the binding of both simulators.
- Chapter 7 *Conclusions* contains a summary of this thesis, and future plans.

Chapter 2

State of the Art

This chapter presents a State of the Art in Electric Railways and High Performance Computing, which is one of the main trends used nowadays in the simulation. First, current technologies and paradigms used in HPC are presented and described. Finally, a detailed analysis of electric railway and simulators in this area are also provided.

2.1 High-Performance Computing

Many differences exist among HPC, High Throughput Computing (HTC) and Many Task Computing (MTC). HTC is characterized by tasks that similarly to HPC, require large amounts of computing power, but run for significantly longer periods of time. HPC applications are measured in terms of FLOPS (floating point operations per second).

There is not a standard definition of HPC, which depends on factors like the environment in which is employed. A general definition for this term could be:

> *Computing model that uses all available resources and techniques for solving complex problems with the lowest consumption of resources or in the shortest possible time.*

Given this definition, we can find HPC environments in Workstations, laptops, smart-phones, Supercomputers, Clusters, Grids, Clouds, and any combination of the above.

2.1.1 Parallel Platforms

Parallel platforms can be roughly classified according to the level at which the hardware supports parallelism, with multi-processor computers having multiple processing elements within a single machine, while clusters, grids, and clouds use multiple computers to work on the same task.

Cluster

A computer cluster is a set of independently operational computers, which are integrated by means of an interconnection network, and supported by user-accessible software for organizing and controlling concurrent computing tasks that may cooperate on a common application program or work-load [Ste02]. The typical architecture of a computer cluster is shown in Figure 2.1 [BB99].

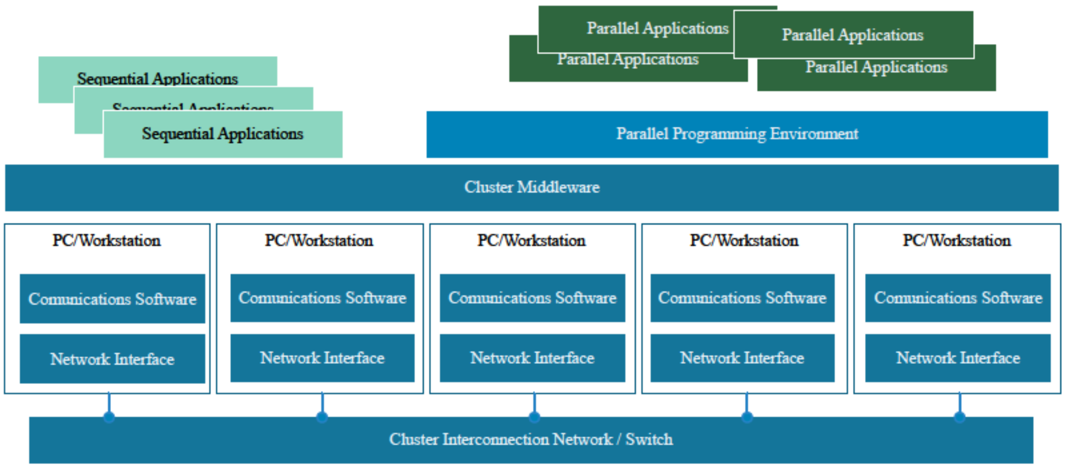


Figure 2.1: Cluster Architecture.

Computer clusters are classified into many categories based on various factors [BB99]:

- **Application Target.** Computational science or mission-critical applications:
  - High Performance Cluster (HP)
  - High Availability Cluster (HA)
- **Node Ownership.** Owned by an individual or dedicated as a cluster node.
  - Dedicated Clusters
  - Nondedicated Clusters
- **Node Hardware.** PC, Workstation, or SMP.
  - Cluster of PC's (CoPs) or Piles of PCs (PoPs)

  - Cluster of Workstations (COWs)
  - Cluster of SMPs (CLUMPs)

- **Node Operating System.** Linux, NT, Solaris, etc.

- **Node Configuration.** Heterogeneous or Homogeneous clusters.

- **Levels of Clustering.** Based on location of nodes and their count. According to this category we can find:

  - Group Cluster (Number of nodes: 2-99): Nodes are connected by SANs like Myrinet and the are either stacked into a frame or exist within a center.
  - Departmental Cluster (Number of nodes: 10s to 100s).
  - Organizational Cluster (Number of nodes: many 100s).
  - National Metacomputers (Number of nodes: many departmental/organizational clusters).
  - International Metacomputers (Number of nodes: 1000s to many millions).

Grid

The term "the Grid" [FKT01, FKNT02, Fos04] was coined in the mid-1990s to denote a proposed distributed computing infrastructure for advanced science and engineering. Much progress has since been made in terms of infrastructures and on its extension and application to commercial computing problems. While the term "Grid" has also been occasions conflated to embrace everything from advanced networking and computing clusters to artificial intelligence, there has also emerged a good understanding of the problems that Grid technologies address, and at least a first set of applications for which they are suited.

Grid concepts and technologies were originally proposed to enable resource sharing within scientific collaborations, first within early gigabit/sec testbeds [SC92] and then on increasingly larger scales [BJB+00, BCF+98, JGN99, SWDC97]. Applications in this context included distributed computing for computationally demanding data analyzes (pooling of compute power and storage), the federation of diverse distributed datasets, collaborative visualization of large scientific datasets (pooling of expertise), and coupling of scientific instruments with remote computers and archives (increasing functionality as well as availability).

Cloud Computing

We briefly describe the cloud service models and technologies to provide some foundation for the discussion [Ter15, AFG+10]. As in the case of Cluster Computing, there are several definitions of cloud computing. Some examples are:

A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers [BYV08].

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. [MG10].

Therefore, the essential characteristics can be summarized into the rapid elasticity, resource pooling, bread network access, on-demand self-service, and measured service [MG10]. Cloud computing technologies and service models are attractive to scientific computing users due to the ability to obtain on-demand access to resources that can replace or supplement their existing systems, as well as, the ability to control the software environment. Scientific computing users and resource providers servicing these users are considering the impact of these new models and technologies.

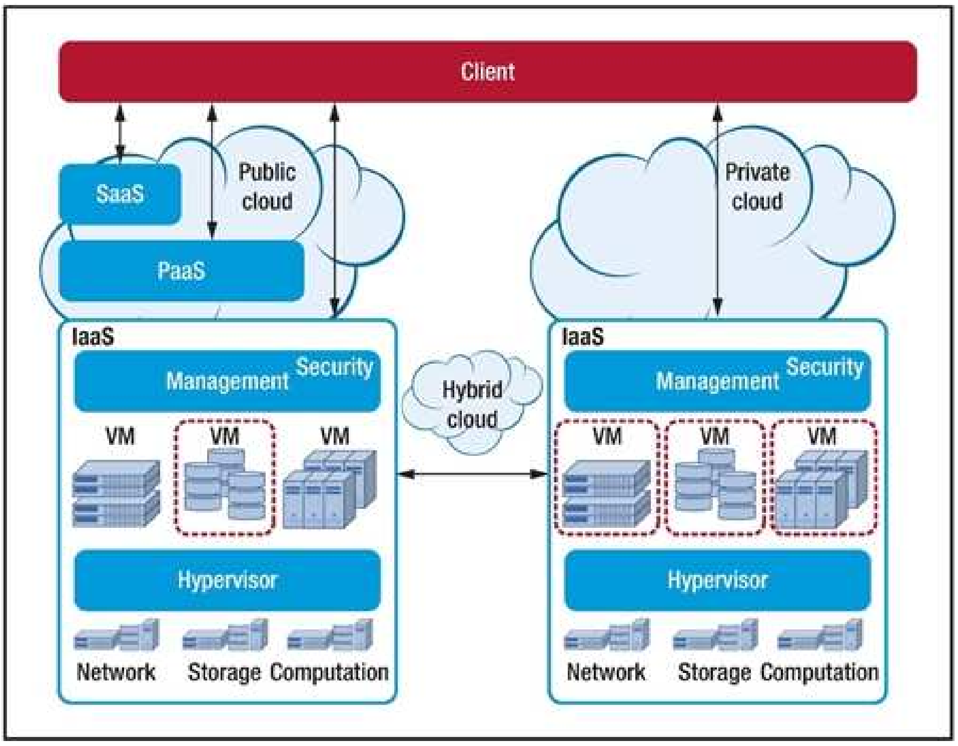


Figure 2.2: The three cloud models [SGH15].

There are three principal services models regarding cloud computing: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service

(SaaS), and the principal deployment models (Private cloud, Public cloud, and Hybrid cloud) which are shown in Figure 2.2 [SGH15].

The NIST (National Institute of Standards and Technology) defines the different service models as follows [Cay13]:

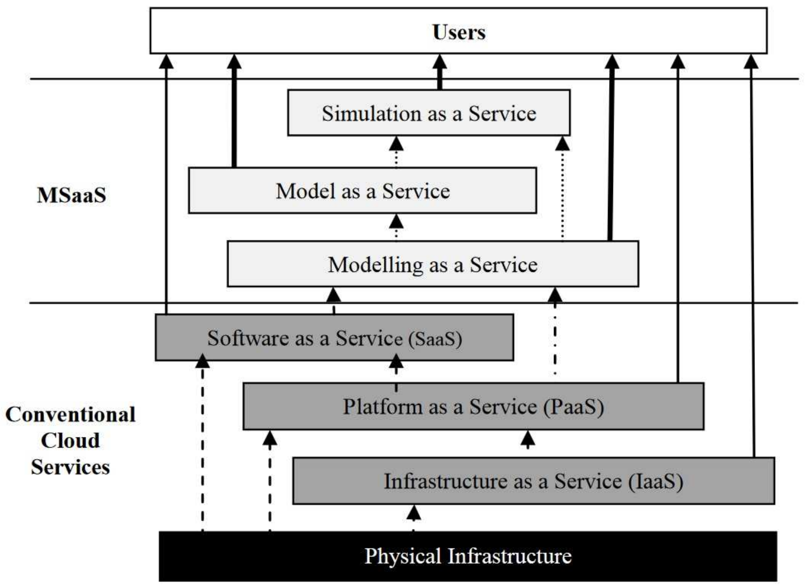

Users
Simulation as a Service
MSaaS
Model as a Service
Modelling as a Service
Software as a Service (SaaS)
Conventional Cloud Services
Platform as a Service (PaaS)
Infrastructure as a Service (IaaS)
Physical Infrastructure

Figure 2.3: Inter-relations between cloud services [Cay13].

IaaS Infrastructure as a Service [SGH15, VGM$^+$16, MG10] *"The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls)."*.

PaaS Platform as a Service [XFPM14, MG10] . *"The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment. "*

SaaS Software as a Service [LZJ15, MG10] *"The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser, or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network,* *servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings."*.

In addition, in recent years, new models have been added, such as HaaS (Hardware as a Service)[SHK12], XaaS (Everything as a Service)[DFZ$^+$15], and MSaaS (Modelling and Simulation as a Service)[Cay13]. Figure 2.3 shows the inter-relations between classic models and new models to give a user MSaaS.

2.1.2 Parallel Programming Models

Parallel computing paradigms rely in the fact that a computing task can be divided into smaller tasks that can be executed in parallel, leading to a better performance than the obtained with the execution of the original task. There exist many parallel models that can be applied in different levels of the hierarchy and architectures. In this section, we will enumerate the most important ones related to this thesis: Shared Memory Model, Distributed Memory, and Distributed Shared Memory.

Shared Memory

Figure 2.4 describes an example of shared memory architecture. There are different options for programming in shared memory systems:

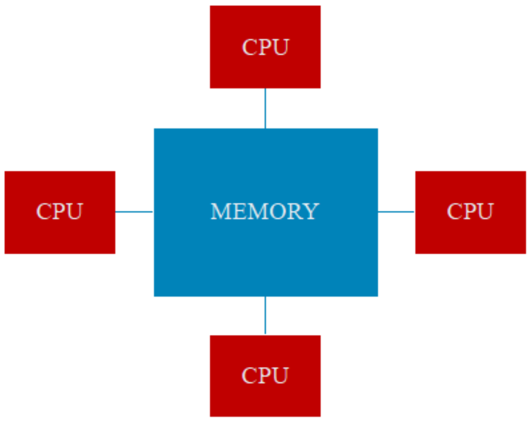


Figure 2.4: Shared Memory Architecture (UMA).

Threads The thread programming model consists of splitting of the problem into multiple concurrent tasks. In function of the number of the computing elements, these tasks can be run on a single processor or on separete processors [HM93]. In Unix systems, threads are implemented using the POSIX Thread Library (pthreads) [oEE, But97]. In C++, we can use library routines like Intel Threading Building Blocks for C++ (Intel TBB) [tbb11].Threads provide the ability to share memory and offer very fine-grained synchronization with other sibling threads. These low level features can provide very fast and flexible approaches to parallel execution. Threaded applications are ideal for multi-core designs because the processors share local memory.

OpenMP *Open Multi-Processing* is an extension to the programming languages, such as C and Fortran, based on compiler directives or programs [Boa13]. OpenMP provides automatic parallelization of loops by adding compiler directives to the loop declaration. The compiler interprets these directives and divides the loop task between several threads.

CUDA Framework developed by Nvidia with the aim of exploiting processing power of its GPUs [NVI11]. CUDA provides C and C++ extensions. While Nvidia is the most popular General-Purpose Computing on Graphics Processing Units (GPGPU) in supercomputing, AMD and Intel have developed its own alternatives to Nvida GPUS: AMD APP and Intel LEO [SGC+16], respectively.

OpenCL OpenCL is the open counterpart of proprietary GPU programming platforms [TNI+10]. OpenCL provides a programming language and API which creates data-level parallelism thus allowing to offload compute tasks onto both CPU and GPU devices. Currently, many GPU developers support OpenCL on its devices.

OpenACC OpenACC [Ope11] is a collection of compiler directives, similar to those utilized in OpenMP. OpenACC offloads compute task onto GPGPU devices. As OpenMP, OpenACC allows to parallelize applications using these hardware devices with a minimum impact on the source code. Besides, OpenACC provides portability across multiple GPGPUs from different providers.

Distributed Memory

Figure 2.5 shows an example of Distributed Memory architecture. Equally that shared memory, there are several alternatives for programming in distributed memory platforms:

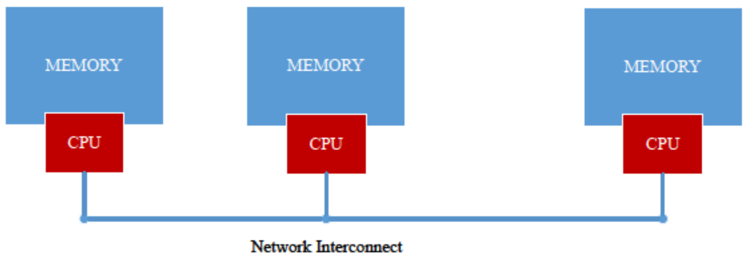


Figure 2.5: Distributed Memory Architecture.

MPI The *Message Passing Interface* is a specification standard, which describes syntax and semantics when multiple processes want to cooperate by means of exchanging messages [For12]. MPI is a standard defined to develop HPC applications which are going to be executed across multiple compute nodes. Today, there are many implementations of MPI. Some of these as follows: OpenMPI [GFB+04] and MPICH [GL96b, GL96a].

The principal advantages of using message passing include:

- **Portability**. MPI is implemented on most parallel platforms.
- **Universality**. The model makes minimal assumptions about underlying parallel hardware.
- **Simplicity**. The model supports explicit control of memory references for easier debugging.

Figure 2.6 shows the execution model of a basic MPI program.

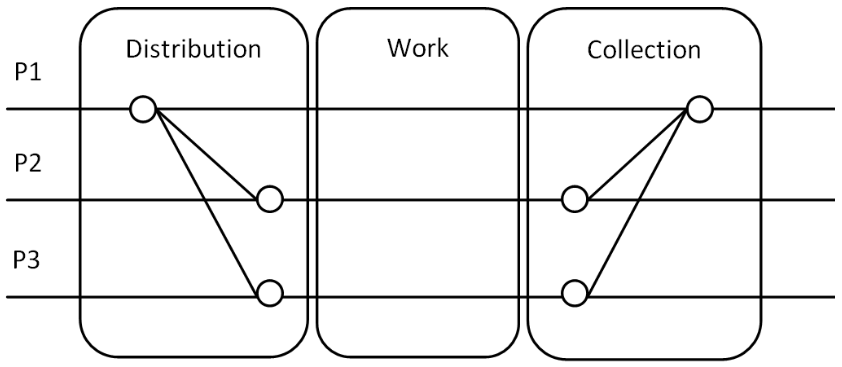


Figure 2.6: MPI execution model.

Distributed Shared Memory

The most efficient, and widely used, programming paradigm on distributed memory systems in message passing. A problem with this paradigm is that it is complex and difficult to program compared to shared memory programming systems. Shared memory systems offer a simple and general programming model, but they suffer from scalability. An alternate cost-effective solution is to build a distributed shared memory system, which exhibits simple and general programming model and scalability [BB99]. Figure 2.7 shows an example of distributed shared memory architecture. The characteristics of software implemented for distributed shared memory systems are:

- Usually built as a separate layer on top of the communications interface.
- They take full advantage of the application characteristics.
- Virtual pages, objects, and language types are units of sharing.

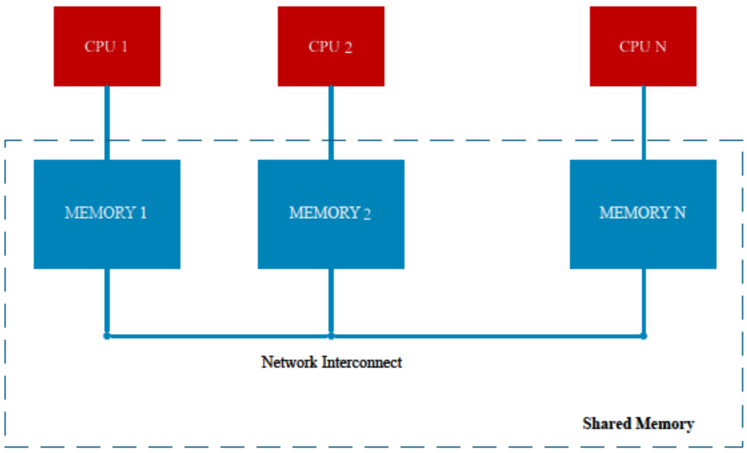


Figure 2.7: Distributed Shared Memory Architecture.

2.1.3 Performance Metrics

When we develop a parallel solution from a serial problem, it is necessary to discuss how fast it is. Two common measurement metrics will be used in this thesis: speedup and efficiency.

Speedup

Speedup is defined for each number of processors n as the ratio of the elapsed time when executing a program on a single processor (the single processor execution time) to the execution time when n processors are available [EZL89]

$$S(n) = \frac{T_1}{T_n} \tag{2.1}$$

Given T_1 as the execution time of the serial version of an application and T_n as the execution time of the parallel program accelerated by n processes.

Efficiency

Efficiency is defined as the average utilization of the n allocated processors. Ignoring I/O, the efficiency of a single processor system is 1. Speedup in this case is of course 1 [EZL89]. The relation between efficiency and speedup is defined in Equation 2.2.

$$E(n) = \frac{S(n)}{n} \tag{2.2}$$

Where $S(n)$ represents the speedup with n processes and n represents the number of processes.

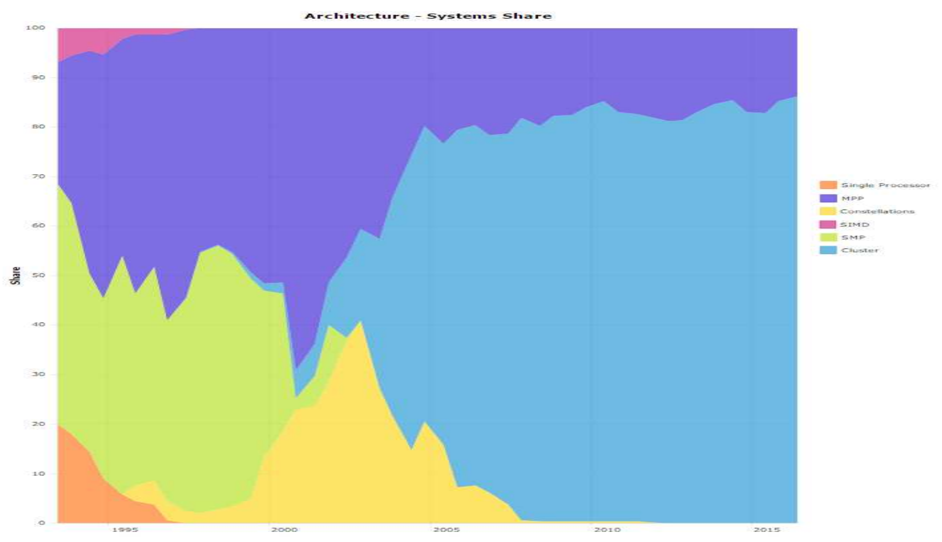


Figure 2.8: Evolution of architecture in HPC.

2.1.4 Supercomputers and Petascale Systems

The association of several supercomputers results in HPC clusters and grids. These partnerships can be built on the same local network or across multiple distributed systems. They can be homogeneous and heterogeneous, gathering CPU and GPU nodes in the same cluster [KES+09, FQKYS04].

Current leading systems in the Top500 rank [DMS+97], which enumerates the 500 most powerful supercomputers, are GPU-based and capable of reporting over one *petaflop* under the standardised Linpack benchmark [DL11]. Some examples of the so-called petascale infrastructure are shown in Table 2.1, which includes the top five rated systems in the Top500 ranking of June 2016 [DMS+97].

Table 2.1: Top five positions in the Top500 ranking of June of 2016.

System	**Cores**	**Tflop/s**	**KW**	**Location**
Sunway TaihuLight	10,649,600	93,014.6	15,371	China
Tianhe-2	3,120,000	33,862.7	17,808	China
Titan	560,640	17,590.0	8,209	USA
Sequoia	1,572,864	17,173.2	7,890	USA
K-Computer	705,024	10,510.0	12,660	Japan

This classification and measurement is not the only one, increasingly researchers and end users seeking a measure that indicates the quality of the systems in real situations, such as data analysis, 3D simulations, and in turn to minimize the energy cost required.

Another important ranking is Graph500 [MWBA10]. Its goal is to evaluate the performance of HPC systems to approach the complex data-intensive applications, measuring traversed edges per second (*TEPS*). Current leaders in June

2016 Graph500 [Top14] ranking positions are shown in Table 2.2. This rank includes shared-memory, distributed memory and, cloud benchmarks.

Table 2.2: Top five positions in the Graph500 ranking of June of 2016.

System	Cores	Performance (GTEPS)	Location
K computer	663,552	38,621.4	Japan
Sunway TaihuLight	10,599,680	23,755.7	China
Sequoia	1,572,864	23,751	USA
Mira	786,432	14,982	USA
JUQUEEN	262,144	5,848	Germany

Nowadays, sustainability and energy efficiency is key in the development and evaluation of HPC infrastructures. Following the Top500 sense, the Green500 list [FC07] is dedicated to rank supercomputers, but in terms of their efficiency, which is measured in performance-per-Watt.

Table 2.3 shows that current leading positions in the rank do not match any of the Top500 systems [Gre14]. and their overall power consumption is significantly smaller than the shown by the latter. This indicates that there is still a lot of research to be done in order to reduce the gap between performance and efficiency, especially considering that supercomputers will keep increasing their target performance to reach the Exascale goal [BWTWc13].

Table 2.3: Top five positions in the Green500 ranking of November of 2015.

System	Performance (Mflops/W)	Power (kW)	Location
RIKEN	7,031.4	50.3	Japan
GSIC Center	5,331.5	51.1	Japan
GSI	5,272.1	57.2	Germany
IMP	4,778.5	65	China
Stanford	4,112.1	190	USA

Exascale systems will become the next generation of supercomputers, capable of performing with at least one exaflop. Scientific simulations will likely benefit from the upcoming Exascale infrastructures [ABC$^+$10], however many challenges must be overcome [BBC$^+$08, GL09] including, processing speed [Cou13], data locality, and power consumption. Among them, energy efficiency seems to be one of the most limiting factor [Hem10].

Nowadays, cheaper and lower power alternatives are on research to overcome such difficulties. For instance, low-end processors are being considered to build large scale supercomputers. Besides, multiple efforts are currently under way in order to reduce energy consumption without reducing compute power, from scaling dynamically the number of compute cores [FL05], to deploy power-aware techniques in the I/O subsystem [LBIC13].

2.2 Electric Railway

Rail transport has become since its inception in the most important way of proving massive transport. Even to this day, rail transport remains one of the methods most of the used to transport both people and goods. There are two major types of railway traction systems, diesel and electric traction. It is shown that electric traction [Cle72], and in particular the use of the catenary has several advantages over diesel and tractions, as it performs a higher power-to-weight ratio. This resuls in a faster acceleration and a higher practical limit of power. Electric locomotives do not depend on crude oil like fuel, so railway electrification systems have less impact on the environment. As stated in [Ake11], specific greenhouse emissions concerning propulsion and fuel production in electric trains will be lower than emissions from other kinds of traction. If we focus our sights on urban transport, railway and its alternative options (train, metro, tram) are considered the best option for several reasons:

- **Pollution.** Although rail transport powered by electricity also depends on fossil fuels, they are also being used in many countries, nuclear and clean energy such as wind, geothermal, etc. Furthermore, this demonstrates that the use of fossil fuels for electricity generation involves less consumption than other alternative transport systems.

- **Safety.** Two factors are responsible of the low rates of accident. First it has its own platform of movement, without interaction the with other means of transport or persons, and absolute control of the movement by technological means. It is true that in case of system failure specific accidents can occur with high numbers of casualties, but fortunately these accidents are rare in the total number of displaced persons. Hence, the International Association of Public Transport claiming that traveling by train in Europe is between 20 and 25 times safer than road.

Electric traction can be classified by three main parameters: voltage, current, and contact system. First of all, the permissible range of voltages allowed for the standardized values is as stated in standards EN-50163. Second, direct current and alternating current are the possible choices in current systems. Figure 2.9 shows the distribution of the electrification systems in Europe. Finally, there are two major alternatives in the contact systems, third rail and overhead lines. Overhead lines have an additional advantage over other ground-level located systems for supplying energy in the rail: the former mechanism is both safer in terms of accidental contacts of people and animals, and have fewer voltage restrictions than the latter one, owing to the elevation over the ground. These facts allow railway companies to use powerful locomotives and to increase traffic over the tracks (see [Har08, MC07]).

In spite of the advantages of using overhead lines, their deployment along the railway tracks is a very complex design task. This complexity can be analysed from four different perspectives.

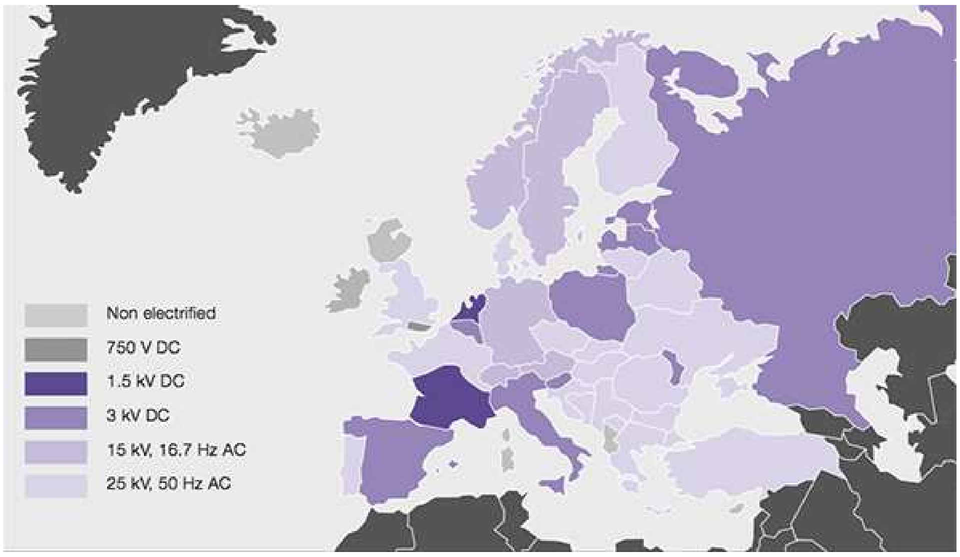


Figure 2.9: Railway electrification systems of Europe.

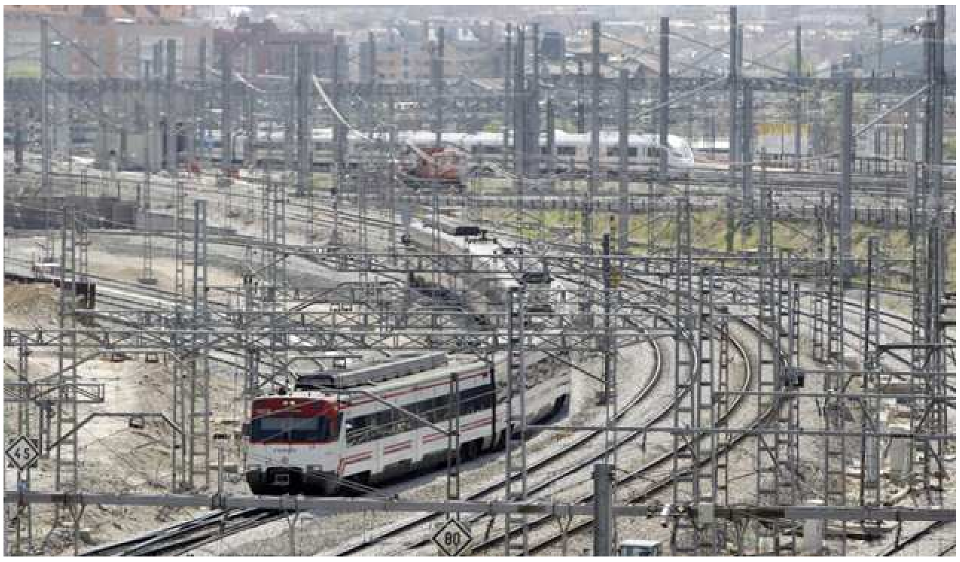


Figure 2.10: Portal frames.

First, many elements have to be considered so as to electrify a track stretch of several kilometres in length. The overhead contact line (also called catenary), is assembled considering a range of spans of about 60 m in length, normally between 15 and 20 [MC07]. If each of them is supported by a pair of poles, more than 30 poles per km in two-way standard tracks are needed. At railway stations, the number of tracks is increased and the space is limited, so poles are replaced with portal frames, allowing simultaneous support for multiple co-located catenaries through a single structure. As an example, Figure 2.10 illustrates the high number of portal frames in a railway station.

Second, the deployment process involves several complex and critical tasks: placing structures along the track stretches. This may include a ground projection of the elements and an analysis of geographical, climatic, and ground conditions.Designing support elements, such as poles and portal frames, in order to withstand the main catenary infrastructure components (wires and cantilevers). Moreover, these supporting elements must deal with extreme conditions, such as strong winds and ice overload.

Third, there are many experts that take part in the design process [DBV09]. Every previously mentioned task requires knowledge from different fields, such as topography, architecture, structural calculus, and drawing. Moreover, technical security and legal normative have to be considered during all the process. Therefore, every task is assigned to a different expert of each field. From this point of view, the complexity of the design process lies in the variety of knowledge sources and it becomes worse due to the difficulty and slowness of communication among all the experts.

Fourthly, as the experts taking part in the process can belong to different companies, the railway company must deal with several outsourced enterprises within a rail work project. This fact results in a hard communication among them, because every company has its own organization, interoperability protocols, and interfaces [PM08]. Moreover, when concerning a cross country project, several railway companies get involved in it, so compatibility issues must be take into account [MT08].

The complexity and the duration of the design process have widespread repercussions in the costs of deploying overhead lines on a rail work project. As discused in [CDR08], a very large proportion of the electrification cost in a railway construction project is spent on the deployment of overhead equipment, such as wires, cantilevers, poles, foundations, and portal frames. H. Brown [Bro27] describes the high number of elements involved in overhead infrastructure installations. Currently, railway companies are still deploying such elements.

Every supporting element has its own specific characteristics and constraints, such as regard the track route, the number of catenaries to be held, and the construction regulations. As there may be several hundreds of one-of-a-kind catenary support structures per project, it is needed a substantial effort for the railway companies and for the others experts part [TCY00]. Working together, they must carry out a laborious process in order to obtain a valid design for each supporting element, including railway inventory management, CAD drawing, structural design and calculus, verification of standards and regulations, and cost analysis.

Railway companies and engineers have been trying to enhance and to regulate their design methods for electrification infrastructure for many years now. Several normatives have been adopted for such aim [UIC81, AEN09]. Computers have played an important role on this effort, as railway and engineering companies have developed several tools to make possible an automation of the tasks, thus reducing the time and costs invested in the design process [VOL$^+$11].

Concerning structural calculus, Finite Element Methods (FEM) [Ode91], presented by [Cou43], is mainly used to carry out the task of calculating support structures, obtaining efforts and displacements at any point. Direct Stiffness Method (DSM), defined by [Tur59], is the most common implementation of FEM. Both FEM and DSM have been included in a lot of software tools in order to automate structural calculus, avoiding errors due to hand-calculation. For example, S. Kanagasundaram and B. Karihaloo [KK90] provides a Fortran 77 code to design minimum-weight frames considering multiple loads. O. Ural [Ura73] proposes an integrated system approach to design housing projects in an urban area and the author enumerates some software tools that perform structural analysis through either DSM or FEM. Regarding structural design, tools like AutoCAD or 3D CAD [Vee06] are used to model railway infrastructure as well.

We have identified some flaws in the previously cited works. First, they lack the vision of an integrated approach, being necessary to use independent tools for the experts part to accomplish designing and calculating a structure tasks. For instance, AutoCAD is used for structural design and CESPLA [Cel03] is used for structural calculus. Second, those tools including that integrated approach [CYP10, Con12], tend to not include railway domain knowledge in the process, which may lead to inadequate designs that are failure-prone or difficult to maintain. Third, those tools that do include specific knowledge about railway domain are shortly versatile, as they do not cover the whole process or all the possible elements. For instance, CALPE [ABA$^+$08] allows to design overhead wires, but it does not consider the structures for supporting wires. In the work presented in [KPSS09], the authors do include the design and the calculation of overhead lines, only considering poles as support structures, leaving out portal frames, that are frequently used at railway stations.

2.3 Railway Simulators

Application areas for simulation are numerous and diverse. For example, designing an analyzing manufacturing systems, determining hardware and software requirements, designing and optimizing transportation systems, and analyzing financial or economic systems. Law and Kelton [LKK91] categorize systems into two types, discrete and continuous systems. A discrete system is one for which the state variables change instantaneously at separated points in time. Conversely, a continuous system is one for which the state variables change continuously with respect to time.

Simulators are excellent tools to face new engineering problems, testing different prototypes to develop optimal designs in an easy, fast and economic manner. Simulators have been widely used in railways since the past century [BCM$^+$98, GSH98]. However they have traditionally adopted the role of solvers or testers, calculating the physical, chemical or mathematical equations associated to a particular engineering problem (e.g. FEM, CFD), which usually are set by the user. One of the major functions of the simulators in the railway sector has always been the traffic simulation These simulators are employed to evaluate and establish timetables and traffic on a particular route, with these schedules the power consumption can be calculated.

We state that new generation simulators should be capable of, starting from a range of possible parameters, proposing and evaluating new designs. Additionally,

Table 2.4: Assessment of the issues covered by representative simulators in the field of railway infrastructure design.

Simulator	Field	Issues	Analysis
Nejlaoui et al.	Railway dynamics	Issue 1	Uncharted
		Issue 2	Yes, it uses artificial intelligence techniques to generate different solutions
		Issue 3	Yes, safety and comfort criteria have been considered in the evaluation
		Issue 4	Yes, it uses artificial intelligence techniques to find the optimal solution
SiCat Master	Overhead contact line design	Issue 1	Uncharted
		Issue 2	No, SiCat Master generates one single design of the span
		Issue 3	Yes, the railway company inventory and the normative EN-50318 have been considered
		Issue 4	Limited, SiCat Master uses algorithms for longitudinal span length optimization
SiCat Dynamic	Pantograph-catenary interaction	Issue 1	Uncharted
		Issue 2	No, it simulates only the user-provided solution
		Issue 3	Yes, the railway company inventory and the normative EN-50318 have been considered
		Issue 4	No, SiCat does not generate different solutions and does not evaluate optimization metrics
Calpe	Pantograph-catenary interaction	Issue 1	Less than 1 hour, to simulate 33 candidate solutions
		Issue 2	Yes, the user provides maximum and minimum values of some parameters.
		Issue 3	No, it neither references any normative nor refers to constructive pieces or company inventory
		Issue 4	No, the tool does not look for an optimal solution, and does not relay on the expert's knowledge
Chang Han Bae	Power provisioning	Issue 1	Uncharted
		Issue 2	Yes, it evaluates the bulk of space solution
		Issue 3	Yes, the international standard IE60146 has been considered
		Issue 4	No, optimization metrics, heuristics, or expert knowledge are not used

they should consider all issues that can affect to the final solution as part of its scope. Examples of such aspects are physical optimizations, normative, cost analysis, etc. This approach can be resume in for main issues:

1.- Trade-off between accuracy and complexity. The simulator must evaluate a possible solution in the lowest time. The results obtained must be applicable to real world.

2.- Automatic generation and simulation of possible solutions. A simulator must be capable of proposing and evaluating new solutions, exploring the search space.

3.- Other actors taking part in the design process (e.g. legislation and normative) must be taken into account to incorporate them into the simulator insofar as they influence the validity of the solutions.

4.- Expert's domain knowledge, useful to find the best solutions, must be also integrated into the simulator, as well as optimization metrics.

These issues have a great impact on the complexity and usability of the simulators, and should be considered carefully. For the evaluating of the issues described previously, a representative group of simulators in the field of railway infrastructure design have been chosen. These can be classified in four categories, depending of the action area: railway dynamics, pantograph-catenary interaction, overhead contact line design, and power provisioning. We analyze these simulators stating whether each issue is fulfilled or not. A summary of this analysis is presented in Table 2.4.

In railway dynamics, a simulator to optimize the design of rail vehicle in a short radius curved scenario has been chosen [NHAR13]. The authors propose a combination between a Monte Carlo simulation and genetic programming. The evaluation of the different solutions is performed according to safety and comfort criteria. The first issue can not be analysed because in this approach, the time required to simulate and evaluate a solution is not indicated.

In the field of overhead contact line design, Sicat Master [KPSS09] has been selected. Sicat Master aims to design and calculate catenary spans, including wire positions, pole locations, and cantilever designs. The railway company inventory is taken into account during the design step and the normative EN-50318 [AEN03] is considered in the calculus step. Due to this, the application satisfies the issue 3. Although SiCat Master uses algorithms for longitudinal span length optimization, it lacks multiple solutions.

In the field of pantograph-catenary interaction two simulators have been chosen. These simulators evaluate the mechanical behaviour of the catenary when the pantograph of the train is running along. The first simulator is SiCat Dynamic, which is compatible with the suite SiCat Master. SiCat Dynamic evaluates dynamic behaviour of the catenary designs obtained with SiCat Master. The second issue is poorly covered because the tool just simulates the solution introduced by the user (inherited of Sicat Master) and it does not generate new solutions in an automatic fashion. However, as SiCat Master, the inventory and normative EN-50318 are considered in the simulation process. The second simulator is Calpe [ABA+08]. This simulator starts from maximum and minimum values of several parameters and iterates over them generating the solutions, so issue 2 is covered. However, the simulator provides a set of valid solutions, but does not evaluate which is the best solution. So issue 4 is not covered. The paper does not refer to any other part in the design process, neither normative, nor constructive pieces, nor company inventory, etc.

In the field of power provisioning, the simulator described in [Bae09] has been chosen. In this work, the author proposes an algorithm to study the installation locations and capacity of regenerative inverters in electric railways. To determinate the capacity of regenerative inverters, the international standard IE60146 [IEC09] has been considered. In order to find a solution, the author evaluates the bulk of solution space. Optimization metrics, heuristics, or expert knowledge are not considered.

From a global point of view, the simulator that best fits the enhanced structure is [NHAR13]. Issues 2, 3, and 4 are covered, thus allowing a high degree of productivity. Issue 1 can not be analysed because time requirements are not mentioned. However, the use of genetic algorithms implies a higher number of candidate solutions, so the time required to evaluate one single candidate should be necessary small.

Although individually lack on fulfil some issues, Sicat Master and Sicat Dynamic are part of a full suite of programs. From the point of view of the whole design process of designing and calculating railway catenary, both simulators working together are which best cover the issue 3, taking into account railway normative, inventory, economic aspects, outputs to the construction phase, etc.

2.4 Summary

In this chapter we have presented a complete vision of high performance computing, including parallel platforms, parallel programming models, and performance metrics.

An overview of electric railway has been introduced. Finally, a recapitulation of the principal railway simulators has been presented.

Chapter 3

Railway Simulation Framework

3.1 Introduction

In this chapter we propose a generic simulation framework focused on the railway sector. This simulation framework aims to present our proposal for enhance functionality and productivity of simulators in the field of railway infrastructure design.

This framework is based on four pillars: compromise between simulation accuracy and complexity; Automatic generation of possible solutions or infrastructure scenarios (automatic exploration of the solution space); Consideration of all the actors involved in the design process (standards, additional restrictions, etc.); And finally, the expert's knowledge and integration of optimization metrics.

3.2 Railway Simulator Structure

Railway infrastructures are considered critical systems, with requirements of efficiency, security and safety, and hence they should be optimized. Nevertheless, performing a high number of experiments with real systems (tracks, locomotives, electric installations, etc.) is unfeasible in terms of time and cost. The main goal of a simulator, in the field of railway infrastructure design, is to simulate candidate solutions for the infrastructure (typically experimental designs or prototypes) to evaluate if they are acceptable or not, or to provide a degree of fitness. This procedure is composed of several tasks. First, a candidate solution must be selected, either being provided by the user or being generated by the simulator itself. Then, the simulation is carried out and the results are analysed. The candidate solution is scored and a decision to accept is taken. This procedure is repeated across multiple fields in this area. In railway dynamics, rail-vehicle interaction is analysed, aiming to get new designs of rails and bogies which may reduce wear and breakdowns. In overhead contact line designs, structural behavior [NHAR13] of poles and portal frames are evaluated, checking their feasibility. In the field of energy provisioning, a proposal of electric installation locations may be simulated checking whether energy is available to all planned trains [Abr].

Although this general structure is present in most railway infrastructure simulators, it may not be sufficient to grant an acceptable degree of productivity and should be enhanced. Moreover, a simulator in railway infrastructure design should not be restricted to evaluate solutions provided by the user, but also it should find acceptable solutions by itself, with a high degree of fitness and in a reasonable amount of time.

3.3 Railway Catenary Infrastructure Design Process

The design and calculation of the railway catenary infrastructure is a very complex process, as discussed in [KPSS09]. It involves several stages that need to be accomplished in order to obtain a valid solution. Every stage of the process requires specific knowledge from different fields, so that different experts have to take part in it. These experts usually belong to different outsourced companies, which must deal with the railway company in order to fulfill the requirements such as costs, quality, security and technical aspects, and legal issues.

In this chapter, we present the stages of this design process with further detail, based on three sources: railway company experts, the design planning process described in [KPSS09], and previous works [CPGC$^+$03]. Figure 3.1 shows all the steps as they are carried out currently, and the experts involved in each one. As may be seen, several rapports have to be established between the different experts.

The main steps for a complete design are described bellow:

- The manager of a rail work project demands the design of the railway catenary infrastructure. He must define several requirements such as the ground features, the height of the catenary points, and how and where the catenaries are held. This definition is sent to the design engineer.

- The design engineer provides a possible design solution for every structure within the project. Each one must be valid from a geometrical point of view. At this stage, many elements belonging to the railway inventory (foundations, poles, lintels, cantilevers, wires, etc.) have to be considered in order to generate possible combinations that fit the requirements specified by the project manager. When consulting the inventory through the railway company, the aim is to provide minimum cost design solutions. Costs are defined by weight, type of material, and manufacturing efforts. All the combinations that can be proposed make the process more complex and more expensive given that specific knowledge for the design experts part is needed.

- At this point, a rapport between the railway company and the design outsourced company is established. The solutions provided by the design engineer

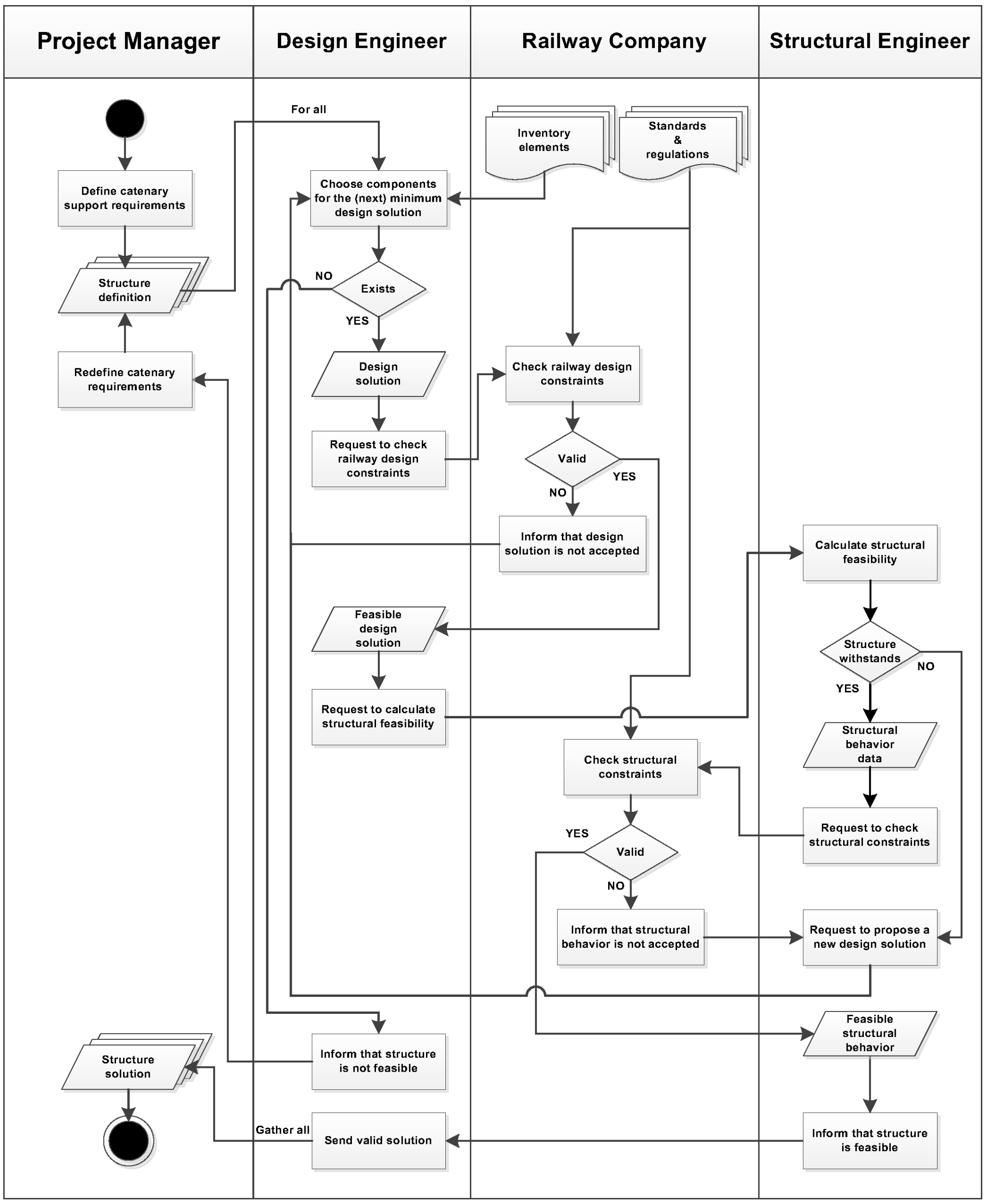

Project Manager
Design Engineer
Railway Company
Structural Engineer
For all
Inventory elements
Standards & regulations
Define catenary support requirements
Choose components for the (next) minimum design solution
Structure definition
NO
Exists
YES
Design solution
Check railway design constraints
Redefine catenary requirements
Request to check railway design constraints
Valid
YES
NO
Inform that design solution is not accepted
Calculate structural feasibility
Feasible design solution
Structure withstands
NO
YES
Request to calculate structural feasibility
Structural behavior data
Check structural constraints
Request to check structural constraints
YES
Valid
NO
Inform that structural behavior is not accepted
Request to propose a new design solution
Feasible structural behavior
Structure solution
Inform that structure is not feasible
Gather all
Send valid solution
Inform that structure is feasible

Figure 3.1: The process of designing and calculating railway catenary infrastructure and its actors.

must be checked. For example existing railway regulations and security and technical aspects must be fulfilled. Therefore, if a design solution for a structure is incorrect, it is discarded and another one must be adopted following the premise of minimum cost design.

- Once every valid design solution, compounded of specific components from the railway inventory, has been provided, the railway catenary infrastructure is analysed from a structural perspective by the structural engineer. Efforts and displacements are calculated in order to check the resistance of the materials, i.e., the structural feasibility of the whole structure. The task of calculating a railway catenary support structure usually relies on Direct Stiffness Method (DSM) to obtain its structural behaviour. Therefore, a model consisting of a set of bars interconnected at nodes is necessary, where all the loads affecting the structure are included.

- At this stage, a rapport between the railway company and the outsourced company responsible for the structural calculus is necessary. On the one hand, the former one is in charge of defining the different load cases that affect structural calculus, such as ice, wind, snow, variations of temperature, etc. On the other hand, the calculus must take into account, considering the structural normative adopted by the railway company. Thereby, if the calculation of a structure is incorrect, it is discarded and, as previously mentioned, a new minimum cost design solution should be adopted by the design engineer.

- When the design process finishes, the project manager can order the construction of the infrastructure.

3.4 Railway Simulator Framework Enhancements

To achieve those targets in this thesis we propose to enhance existing simulators in order to increase the output of simulators by covering more capabilities than the main procedure described before.

Our enhancements are focused on four main issues addressed below. The sources of this approach are: railway company experts, railway infrastructure design, and planning processes described in [KPSS09].

First of all, a trade-off between accuracy and complexity is required when designing a simulator. Productivity issues in railway industry require not to expend so much time when evaluating a single solution, as the design process may require to evaluate a lot of candidate solutions. There is a relation between the accuracy of the model and the complexity of the simulation. On the one hand, accurate models are usually hard to simulate and require more and more operations so the more accurate is the model, the more complex it is, and the more time is needed to reach the solution. On the other hand, accurate models are likely to reproduce real results. Optimal balance between accuracy and simulation may be different in different design processes. However we state that an efficient simulator should simulate and evaluate a single candidate solution in the lowest possible time. We state that an acceptable threshold to be productive is to simulate and evaluate a candidate solution in less than one hour.

In the second place, automatic generation and simulation of solutions falls outside the scope of most of the simulators. Therefore, users must feed the simulator providing new possible solutions, which leads to productivity losses. Moreover, the capacity of finding good (maybe optimal) solutions is tied to the user and her own ability to explore the problem's search space. We state that an efficient simulator should evaluate and simulate a set of solutions with a minimal user involvement. To achieve that: a) the user should provide the simulation parameters as a set of possible values (e.g. [minimum, maximum, increment]) and the simulator should use them to generate candidate solutions; b) the simulator should be able to generate new solutions starting from an initial database (e.g. an inventory or catalogue).

Thirdly, there are many stakeholders taking part in the design process which usually fall out of the scope of the simulation models. These parts can influence, or even determine, the final acceptance of the candidate solutions [NHC13]. For instance, the set of possible solutions when looking for a valid design of a railway portal frame can be limited by the availability of constructive pieces in the company's inventory. Once found, a portal frame could not be in compliance with legal normative in certain countries [BE09b]. All issues that have to be considered throughout the design process, but fall out the scope of the simulation model should be also taken into account when simulators generate and evaluate candidate solutions. This category includes provider specifications, client requirements, technical security, and legal normative. Different ways to include such restrictions out of the simulation model are: a) restrictions to generate candidate solutions: the simulator only generates candidate solutions that fulfill with these initial restrictions; b) restrictions to evaluate a candidate solution, so that the simulator evaluates these restrictions as well as any others conditioned by the simulation model.

Finally, expert's domain knowledge is a fundamental part in the engineering design process [Ade03]. Expert's knowledge defines heuristics that allow to speed up the search process and to achieve the best solutions in the problem's search space. Therefore it should be included as a part of the simulation, particularly in those simulators that include automatic generation of candidate solutions (described as the second issue). In a similar way to other participants in the design process, expert's knowledge can be included when generating candidate solutions in the form of decision rules. Those rules guide the search process to generate better candidate solutions. They can also be included to evaluate a candidate solution in the form of optimization metrics, which can be used to score the solution and to compare it with others, thus choosing the best one.

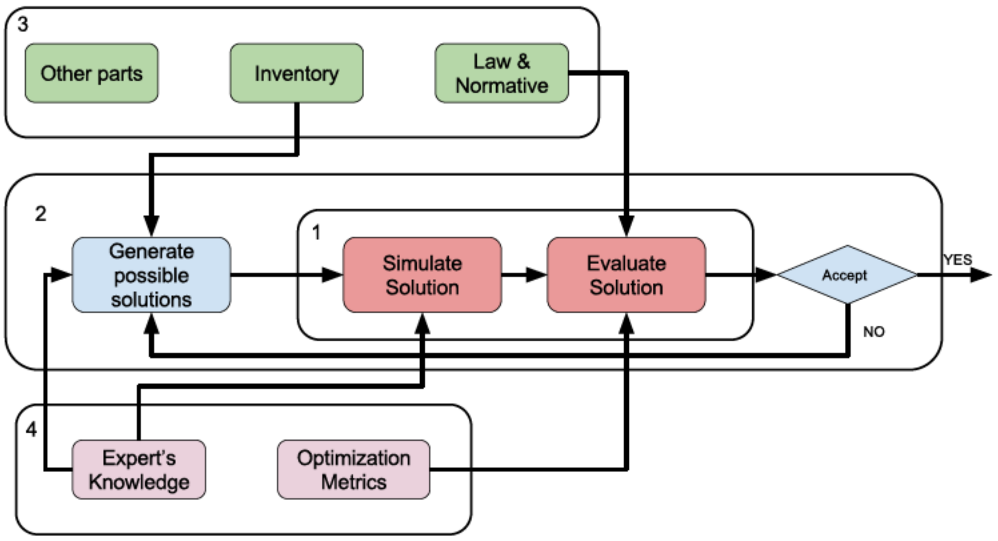


Figure 3.2: Structure of the simulation framework proposed.

3.5 Generic Simulator Framework

To include the former enhancements in railway simulators, we propose here a generic simulator framework. Figure 3.2 represents the layers that compose our simulator framework. The framework architecture is layered following the four issues previously mentioned.

The core procedure for simulating and evaluating candidate solutions compounds layer 1. Time invested in performing these two tasks must be the lowest possible. Issue 2 is covered by the layer 2, which contains the task for generating automatically new solutions to be evaluated. This task could be fed from other elements in layers 3 and 4 (e.g. requirements related to an inventory of constructive pieces, or expert's domain knowledge, applied to generate better candidate solutions). Layer 3 is composed by those restrictions that are not included in the simulation procedure (layer 1), but that have an impact on the solution. Those restrictions can be applied either when generating or when evaluating a candidate solution. Examples of such restrictions are availability of constructive pieces in company's inventory when proposing a design, or compliance with legal normative when evaluating the proposed design. Finally, layer 4 represents the expert's domain knowledge, which allows to obtain better solutions. Decision rules used to generate better candidate solutions, or optimization metrics used to choose the best one are included in this layer. This approach improves the efficiency of the simulators by giving them the ability of searching for the best solutions in the problem space. Obtained solutions will be fully-integrated with the different actors of the design process and therefore they are more suitable to be implemented in the real world.

3.5.1 Railway Infrastructure Design Simulation

In the previous section, we have introduced the railway infrastructure design process. An automation of this process would reduce the time invested in achieving a valid and optimal solution. The Algorithm 3.1 shows our proposal to automate and optimize the design process. First we must determine if there are some overhead air switch (see line 2). The overhead air switches are the most critial aspect in the catenary design process. If there are some switch under the structure, we simulate and evaluate the overhead air switch first (see lines 5 to 7). This step modifies the design of the catenary and the structure characteristics. At the end, we simulate all the structures of the global project (see line 9).

Algorithm 3.1 ResolveProject.

```
Input: $S, c$
Output: $S$
 1: for all $s_i \in S$ do
 2:     $hasOverhead \leftarrow evaluateCatenary(s_i)$
 3:     if $hasOverhead$ then
 4:         $overheadSwitches \leftarrow getOverheads(s_i)$
 5:         for all $o_i \in overheadSwitches$ do
 6:             $s_i \leftarrow simulateSwitch(s_i, o_i)$
 7:         end for
 8:     end if
 9:     $s_i \leftarrow simulateStructure(s_i, c)$
10: end for
```

3.6 Summary

In this chapter we have presented a generic simulation framework with the aim of enhancing functionality and productivity of simulators in the field of railway infrastructure design. This approach is focused on four main issues: trade-off between accuracy and complexity, automatic generation and simulation of possible solutions, taking into account other participants in the design process, and integrate expert's domain knowledge and optimization metrics. This structure improves the efficiency of the simulators by giving them the ability of searching for the best solutions in the problem space. Also, obtained solutions will be fully-integrated with the different actors of the design process.

In the following chapters, we will present RDIS to simulate railway structure, and OCLS to simulate overhead air switches.

Chapter 4

Railway Simulator

4.1 Introduction

The goal of this chapter is double: first of all, a railway infrastructure ontology for the railway portal design process is defined; secondly, we introduce a new complete simualtor (RDIS) to design and calculate railway catenary infrastructure. RDIS has three principals parts: track design, catenary design over the track, and infrastructure design. All these parts are tightly coupled.

4.2 RDIS Railway Simulator

Since there is a lot of expert knowledge involved in each stage of the process, the communication among the experts is usually slow. Besides, if a proposed solution for a single support structure is incorrect, some steps have to be repeated until either a valid one is found, or all possible solutions have been tested. This means that the process of designing and calculating the railway catenary infrastructure is desired to be shortened in time in order to increase its efficiency and to reduce costs. Regarding the input requirements such as the geometric feasibility and the structural strength, it is important to point out that there may be several valid solutions for a railway catenary infrastructure in compliance with current railway legislation and regulation. However, the minimum cost solution should be provided.

In order to automate the design process of a complex railway portal, we have to be able to fulfill all the steps from the catenary points specified by the railway electrification engineers (shown in Figure 4.1) to the final structure (shown in Figure 4.2) that is geometrically and structurally feasible and cost-effective.

The proposed architecture includes several features to cope with the complexity of railway infrastructure design problem that make it novelty:

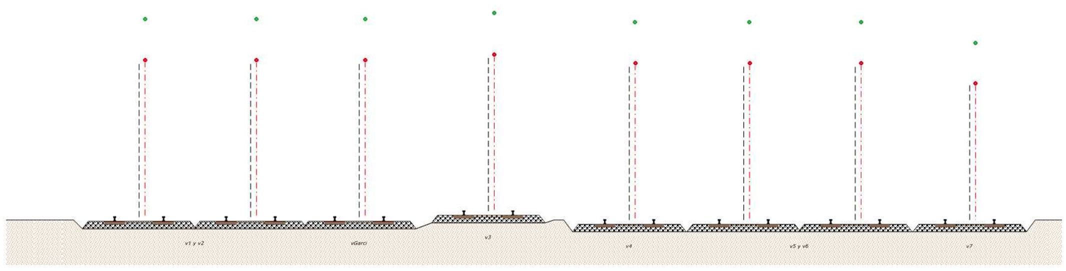


Figure 4.1: Initial data for railway portal design.

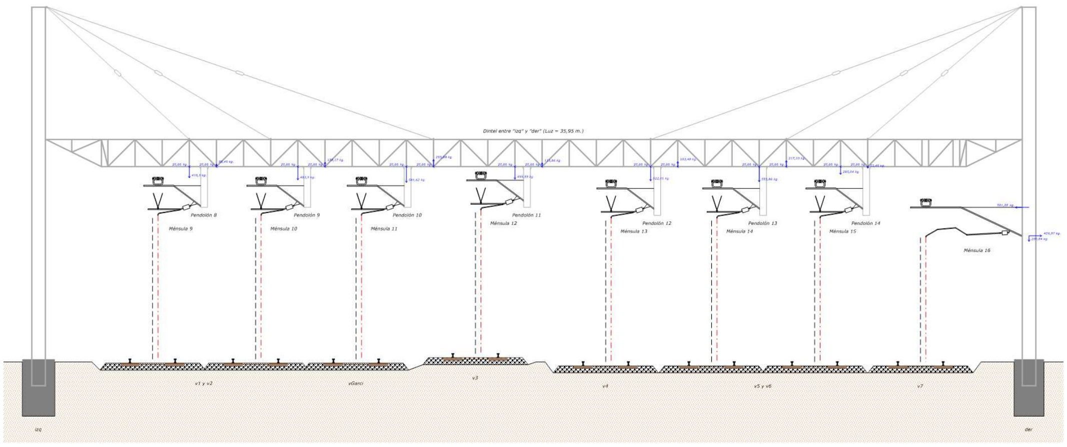


Figure 4.2: Result of the intelligent design for the former problem.

1.- A comprehensive database of standardized materials. Initially, those used in Spanish railways, but any material can be included using a graphic interface. This database includes catenaries and electrification system materials, and different railway elements such as switches and crossings.

2.- A powerful and intuitive graphic interface specifically adapted to the railway. This tool provides a computer-aided track design that allows users to design different elements of the tracks: curves, transition curves, switches, and crossings.

3.- A complete geometric and structural calculus library to make, not only graphic design, but also structural validation of systems, according to railway standards.

4.- A simulation system that, once a project is designed, can simulate train circulations along the infrastructure to test components as rails, switches, or the interaction between the pantograph and the contact wire.

5.- An ontology, knowledge rules-based system, and a rule engine that automatically chooses the best design options for every situation. The production rules include normative and the knowledge of railway company and railway design experts in several fields (i.e. tracks, electrification, signals). The knowledge base and the ontology can be expanded to other railway systems.

4.2.1 Ontology Definition

One of the most important topics of intelligent systems is the knowledge representation on the problem domain. Thus, an important design decision is to choose the techniques for representing the knowledge of the application to be developed. In our case, we use techniques that can capture knowledge about railway infrastructure in the best way.

Several knowledge representation techniques can be used: semantic networks, frames, uncertain reasoning, ontology and rules, etc. [GYO$^+$05, GHS05, Dut96, NEGED10, Sch00]. For complex systems, as railway infrastructure, simple mechanisms as semantic networks or frames are not appropriate, as they only let to use declarative statements that are true or false. The same applies to uncertainty reasoning [KC07], which provides solutions for situations whether a true or false cannot be got, leading to conflicts with railway regulations. Nowadays, the trend for representig railway infrastructure knowledge is to use ontologies and rules [MA11]. Ontologies allow to structure information integrating different unstructured information sources and providing an unified terminology. The definition of the ontology concept changes among the users of the AI field, but the definition of an ontology as a specification of conceptualization [Gru93] is generally accepted nowadays [NM01], but adding to each object of the ontology a set of properties that represent its features. However, the ontology itself is not enough. To design real-life structures, it must be integrated with a set of well-defined rules that can manage information and apply heuristics to represent regulations, structural constraints, and calculus, etc. [Joh02, RC03]. A hybrid approach has been adopted in RDIS: the ontology of the railway infrastructure and a rule-based representation technique, as the railway infrastructure can be captured using ontologies and rule-based representation techniques can be used to solve the process.

The first challenge to build a railway ontology for the simulator was the knowledge identification, which was achieved through an extensive literature review, including railway books [MC07] and journal papers [Hol86, MM99, SKS00, QJ10], study of existing ontologies, consultation to experts and railway company managers, regulation and legislative documents, and personal experience of the authors, were used as sources of knowledge. Along this process, concepts, attributes, and relations of the railway infrastructure ontology were established.

The second challenge was the knowledge specification of the railway infrastructure model. A two-step approach has been applied: semiformal modeling of the ontology using Unified Modeling Language (UML) to represent objects, classes, relations and properties; and representation using a Web Ontology Language (OWL) via the use of the ontology editor *Protege* [Sta11]. *Protege* is an open-source integrated and platform independent ontology editing tool. One major advantage is supporting the SWRLTab, a plug-ing to edit Semantic Web Rule Languages (SWRL) rules.

Figure 4.3 presents the UML semiformal model implemented in *Protege* for the ontology subset representing the railway portal domain. The figure shows the following main concepts: Portal, which is the complex structure joining several elements; Lintel, the structure supporting cantilevers and overhead wires; Mast, the poles supporting the lintel, and probably cantilevers; and Foundations, that are the structures supporting the masts. A portal is a complex system that may be formed by several of the former components, but as a minimum two foundations, two masts, and a lintel are needed for a portal. Multiple cantilevers and other elements are also included. Consultations with railway experts and design and structural engineers helped to refine and to confirm the conceptual models obtained.

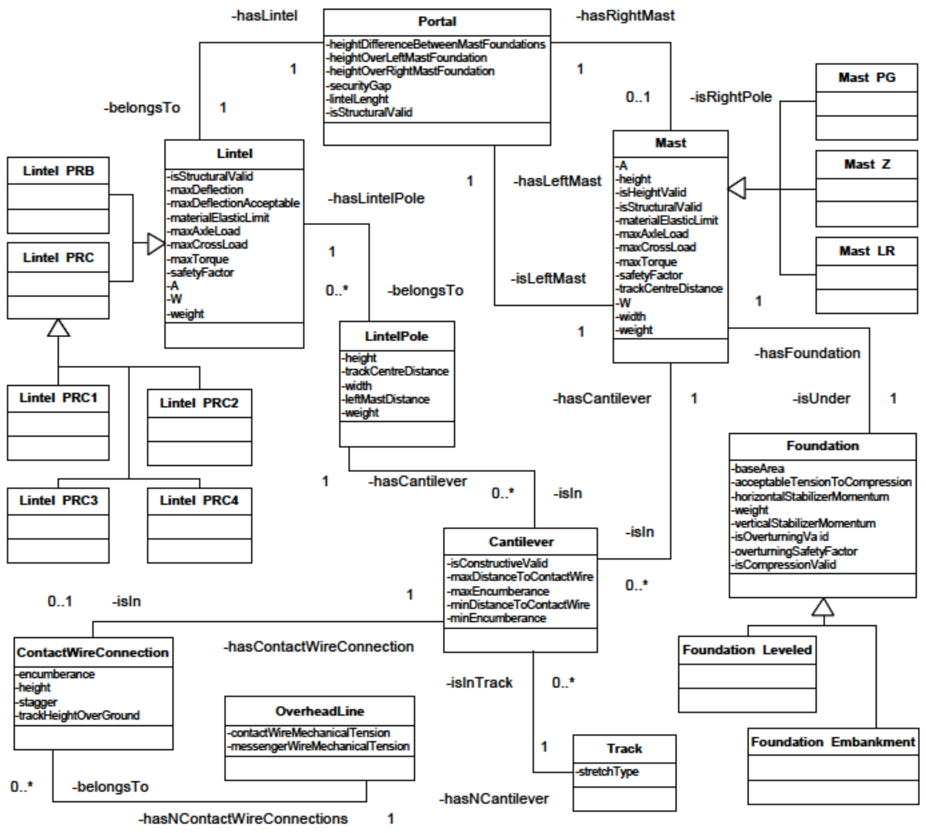
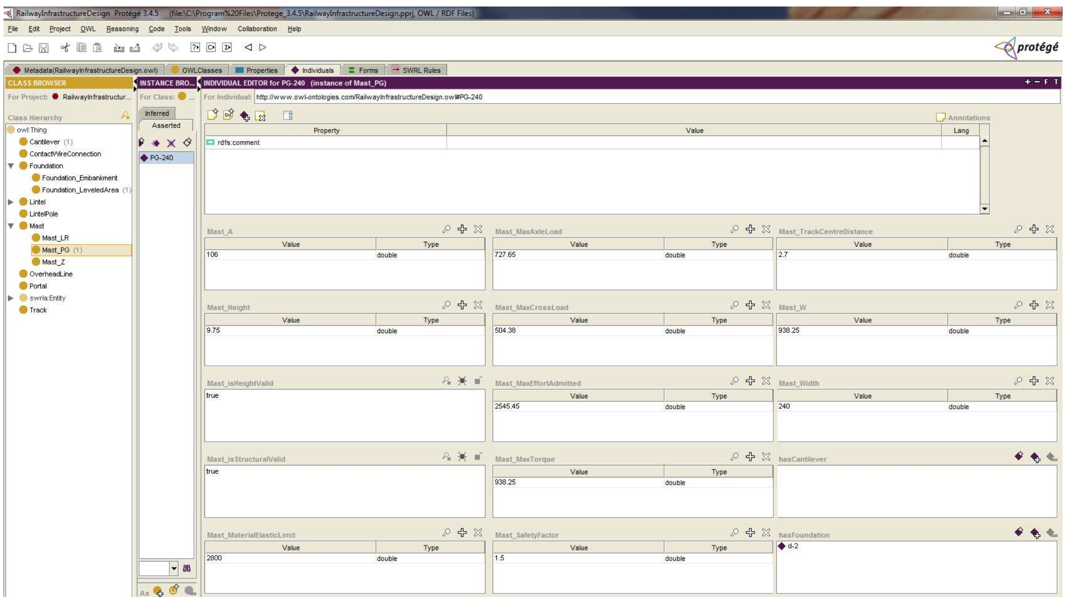


Figure 4.3: Semiformal modeling of the railway infrastructure ontology using UML for the railway portal design process.

Those components are stored in an object-oriented database whose elements belong to the *Protege* core components shown in Figure 4.4. These components have object classes, instances, relations, and attributes or properties. They are mapped to the Protg database as shown in the figure. Each component has several properties. The figure also shows some properties of an element (a mast) edited with *Protege*. Properties may be descriptive, constructive, structural, validity conditions, etc. For example, *Mast_ MaxTorque* is 938.25, and *Mast_ Height* is 9.75 meters for this element. As may be seen, an object has several properties. All of them are used by the rule system to get a solution to the design problem. Object properties remain unchanged in the translation process to *Protege*, as data properties changed to start with has (for example *hasCantilever*). This practice conforms to the ontology knowledge model in *Protege* and it is common in ontologies to facilitate readability. Axioms and annotation properties are also used in the Protg ontology to define operations among properties. Inheritance, for example, is defined as a belongsTo relation to define a taxonomical relationship among concepts and subconcepts, with subconcepts inheriting the concept properties. For example, *Lintel_ PRC* is a type of Lintel. The relation IsIn shows a constructive relation and it is related to the set of elements belonging to the domain of a railway portal. There are several IsIn relations depending on the object (cantilever, track,). We have other names, as IsUnder, for similar properties. Those names are used to have a more readable model and ontology.


RailwayInfrastructureDesign Protégé 3.4.5 (file:\C:\Program%20Files\Protege_3.4.5\RailwayInfrastructureDesign.pprj, OWL / RDF Files)
File Edit Project OWL Reasoning Code Tools Window Collaboration Help
protégé
Metadata(RailwayInfrastructureDesign.owl) OWLClasses Properties Individuals Forms SWRL Rules
CLASS BROWSER
For Project: RailwayInfrastructur...
Class Hierarchy
owl:Thing
Cantilever (1)
ContactWireConnection
Foundation
Foundation_Embankment
Foundation_LeveledArea (1)
Lintel
LintelPole
Mast
Mast_LR
Mast_PG (1)
Mast_Z
OverheadLine
Portal
swrla:Entity
Track
INSTANCE BRO...
For Class:
Inferred
Asserted
PG-240
INDIVIDUAL EDITOR for PG-240 (instance of Mast_PG)
For Individual: http://www.owl-ontologies.com/RailwayInfrastructureDesign.owl#PG-240
Annotations

Property	Value	Lang
rdfs:comment		

Mast_A

Value	Type
106	double

Mast_MaxAxleLoad

Value	Type
727.65	double

Mast_TrackCentreDistance

Value	Type
2.7	double

Mast_Height

Value	Type
9.75	double

Mast_MaxCrossLoad

Value	Type
504.38	double

Mast_W

Value	Type
938.25	double

Mast_isHeightValid: true

Mast_MaxEffortAdmitted

Value	Type
2545.45	double

Mast_Width

Value	Type
240	double

Mast_isStructuralValid: true

Mast_MaxTorque

Value	Type
938.25	double

hasCantilever

Mast_MaterialElasticLimit

Value	Type
2800	double

Mast_SafetyFactor

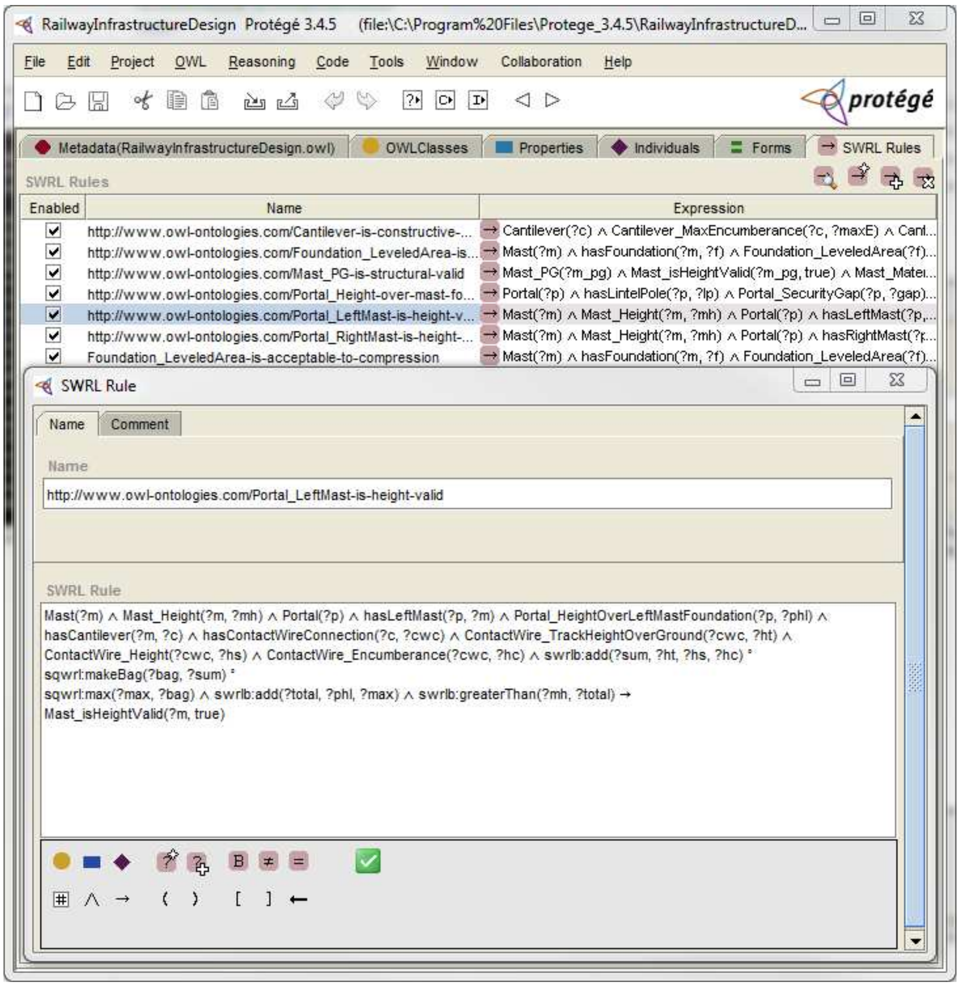
Value	Type
1.5	double

hasFoundation: d-2

Figure 4.4: A reduced view of the railway portal process ontology.

4.2.2 Rule System Development

To accomplish all the design process, our simulator uses production rules that guide the design process itself depending on the ontology objects and their properties (terrain quality, portal length, number of tracks under the portal, number of catenaries, weight and wires tension, track shape, etc.). A top-down approach is used to choose all the feasible elements applying the knowledge rules of the system. Figure 4.5 shows some rules and queries belonging to the railway portal design process. After choosing all the elements, constructive feasibility must be assessed. Thus, the simulator geometrically builds the railway portal to check whether all components can


RailwayInfrastructureDesign Protégé 3.4.5 (file:\C:\Program%20Files\Protege_3.4.5\RailwayInfrastructureD...
File Edit Project OWL Reasoning Code Tools Window Collaboration Help
protégé
Metadata(RailwayInfrastructureDesign.owl)
OWLClasses
Properties
Individuals
Forms
SWRL Rules
SWRL Rules
SWRL Rule
Name
Comment
Name
SWRL Rule

Enabled	Name	Expression
✓	http://www.owl-ontologies.com/Cantilever-is-constructive-...	Cantilever(?c) ∧ Cantilever_MaxEncumberance(?c, ?maxE) ∧ Cant...
✓	http://www.owl-ontologies.com/Foundation_LeveledArea-is...	Mast(?m) ∧ hasFoundation(?m, ?f) ∧ Foundation_LeveledArea(?f)...
✓	http://www.owl-ontologies.com/Mast_PG-is-structural-valid	Mast_PG(?m_pg) ∧ Mast_isHeightValid(?m_pg, true) ∧ Mast_Mate...
✓	http://www.owl-ontologies.com/Portal_Height-over-mast-fo...	Portal(?p) ∧ hasLintelPole(?p, ?lp) ∧ Portal_SecurityGap(?p, ?gap)...
✓	http://www.owl-ontologies.com/Portal_LeftMast-is-height-v...	Mast(?m) ∧ Mast_Height(?m, ?mh) ∧ Portal(?p) ∧ hasLeftMast(?p,...
✓	http://www.owl-ontologies.com/Portal_RightMast-is-height-...	Mast(?m) ∧ Mast_Height(?m, ?mh) ∧ Portal(?p) ∧ hasRightMast(?p...
✓	Foundation_LeveledArea-is-acceptable-to-compression	Mast(?m) ∧ hasFoundation(?m, ?f) ∧ Foundation_LeveledArea(?f)...

http://www.owl-ontologies.com/Portal_LeftMast-is-height-valid

```
Mast(?m) ∧ Mast_Height(?m, ?mh) ∧ Portal(?p) ∧ hasLeftMast(?p, ?m) ∧ Portal_HeightOverLeftMastFoundation(?p, ?phl) ∧
hasCantilever(?m, ?c) ∧ hasContactWireConnection(?c, ?cwc) ∧ ContactWire_TrackHeightOverGround(?cwc, ?ht) ∧
ContactWire_Height(?cwc, ?hs) ∧ ContactWire_Encumberance(?cwc, ?hc) ∧ swrlb:add(?sum, ?ht, ?hs, ?hc) °
sqwrl:makeBag(?bag, ?sum) °
sqwrl:max(?max, ?bag) ∧ swrlb:add(?total, ?phl, ?max) ∧ swrlb:greaterThan(?mh, ?total) →
Mast_isHeightValid(?m, true)
```

Figure 4.5: Example of rules and queries for railway portal process ontology.

fit the spatial restrictions of the infrastructure (track values and catenary points, basically). If they do, the structural feasibility of the portal is assessed by applying different computing methods like direct stiffness method, Sulzberger, etc. If something is wrong, a new composition is tested, and the process continues until getting a feasible configuration or an error. As this calculation process is CPU intensive, it is important to apply optimizations [Esc03]. RDIS methods have been optimized to compute the entire tested configurations in seconds or a few minutes. Several algorithm optimizations have been applied and parallel computing techniques have been used to provide a very fast answer to the designer, as show in the final section of this chapter.

Rules allow to go beyond the structural relations provided by the ontology. Protg provides SWRL, to extend weak forms of OWL systems with first-order rules [Wal07]. SWRL provides a Jess rule engine, knowledge base populated with rule instances, user interface SWRL-Tab, and the rule engine bridge SWRL-REB. Rule knowledge base is an extension of the ontology knowledge base and the queries on the ontology can be executed using Queries-Tab plugin. Queries on the rule base can be made using SWRLTab. The rule knowledge base contents information captured from railway experts, books, journals, regulations, and standards and it has been tested by experts of a railway company. The rules knowledge base was developed by constructing a decision tree analysis. The decision tree revealed a minimum of 136 possible outcomes. However, the system could increase to thousands of rules by adding more complexity, details, and variety to RDIS. The decision tree was very useful to accomplish process optimization to get outcomes as fast as possible, and to compare the decisions of RDIS to those taken by the experts.

All rules in the knowledge base follows the format:

$$A_1, A_2, \ldots, A_n \rightarrow B \tag{4.1}$$

Where A_i and B are atomic formulas depicting conditions (A_i) and the resulting action (B) executed when the conditions are fulfilled. Figure 4.5 shows a rule of RDIS, specifically rule number 5, to choose a mast with enough height to maintain the lintel over wire support structures. The rule is written using the SWRL syntax, which only provides the conjunction symbol ($\wedge$) and the implication symbol ($\rightarrow$), rule variables, and syntax for components, classes, property atoms, and data valued atoms. The rule variables are represented by the interrogation symbol (?) (e.g. ?m). The class atoms are constructed from an OWL class name followed by one variable or component name (e.g. `Mast(?m)`). As may been seen, deciding if the mast is tall enough for the portal depends on all the elements of the portal, like foundations, contact wire height, catenary, etc. The final result is equivalent to an if then sentence with several conditions:

If « Mast height is taller than
the maximum height of contact wire points +
the height of the cantilever used to hold the contact wire +
the height of the mast foundation »

Then « Mast_IsHeightValid is true for that portal »

This is only one of the premises to be tested before choosing a mast and it is a step of the portal design phase. After that, structural constraints must be tested, which also have several conditions to be satisfied. For example:

If « Mast_MaxTorque is higher than current mast torque $\wedge$
Mast_MaxEffort is higher than current mast effort $\wedge$
Mast_MaxAxleLoad is higher than current axle load »

Then « Mast_IsStructuralValid is true for that portal »

Some of the former conditions result from complex calculus and they must be computed dynamically for each possible solution, as they do not depend on the properties of the object itself, but on the forces and restrictions resulting in the design. For example, to test the current mast torque, the following equation is applied:

$$fct * \frac{M_{zk}}{W_{yUPN}} + \frac{N_k}{A_{UPN}} < \frac{\sigma_e}{\gamma_{m0}} \tag{4.2}$$

being N axial efforts, A the area of the mast, W the rigidity of the mast, and σ, γ, and fct several safety coefficients defined by the regulations.

Several masts from the ontology could satisfy the former conditions, thus a set of valid solutions is obtained to build the portal. The same situation is possible for all the components like foundations, cantilevers, etc. leading to a combinatory explosion of possibilities that must be considered.

4.3 Generating and Calculating Portal Frames

In the previous section, we have presented the process proposed to design and calculate railway catenary infrastructures. An automation of this process would reduce the time invested in achieving a valid solution. Regarding the elements of the railway inventory that are used in the process, the aim is to find:

- A valid design for all the cantilevers, poles and lintels, catering for the geometric configuration of the catenaries held by the structure.
- A feasible bar assembly of poles and lintels after calculating their structural behaviour through DSM.
- A valid calculus for all the hypothesis.
- A valid choice for every foundation, considering its overturning and subsidence resistance.

Three main problems arise from this design and calculation process.

First, the designed structures are very heterogeneous, i.e., they have their own characteristics and constraints, with regard to the track route, the catenaries to be held, the hypothesis of load cases that are used, or the construction regulations. This problem prevents design and structural engineers from developing a single common solution.

Second, applying DSM is a time-consuming task due to it requires a large number of operations. In our proposal, we have to apply one DSM per hypothesis, $H = h_1, h_2, ..., h_n$, where h is a hypothesis defined for the user. On the one hand, the catenary support structure must be modelled as a set of bars interconnected at nodes. The way to obtain efforts and displacements at any point of the structure consists of resolving a set of equations, generated from the stiffness matrices of the bars, and the loads affecting the structure. On the other hand, feasibility verification formulas with safety coefficients specified by the railway company must be also considered in order to resolve the whole structure. Finally, the more complex the structure, the more operations must be performed.

Third, depending on the inventory size which the system is linked to, the set of combinations where a feasible solution can be found may be large. Let I the inventory that contains the constructive elements used to assemble a structure. $I = \{L \mid P \mid F \mid C\}$, where $L = \{l_1, l_2, ..., l_m\}$ is the set of lintels included in the inventory, $P = \{p_1, p_2, ..., p_n\}$ is the set of poles included in the inventory, $F = \{f_1, f_2, ..., f_p\}$ is the set of foundations included in the inventory, and $C = \{c_1, c_2, ..., c_r\}$ is the set of cantilevers included in the inventory. Let W the planned project. W contains a number of structures, $W = \{s_1, s_2, ..., s_s\}$. A single structure is defined by $s_i = \{n_{il}, n_{ip}, n_{ic}\}$, where n_{il} is the number of lintels in the structure, n_{ip} is the number of poles in the structure, and also the number of foundations, and n_{ic} is the number of cantilevers that support the overhead lines attached to the structure. The number of possible assemblies for all the cantilevers, poles, and lintels in s_i, is

$$N_{a_i} = \|C\|^{n_{ic}} \cdot \|P\|^{n_{ip}} \cdot \|L\|^{n_{il}} \tag{4.3}$$

For the first assembly in N_{a_i} that is considered to be feasible during the process, the number of possible choices for all the foundations in s_i, is

$$N_{f_i} = \|F\|^{n_{ip}} \tag{4.4}$$

For example, having 6 lintels, 23 poles, 25 foundations, and 7 cantilevers catalogued in the inventory, the number of possible assemblies for a standard portal frame (a lintel, two poles, and 4 cantilevers) is 7,620,774. The number of choices for the foundations once a feasible assembly is found is 625.

Let $t_{d_{avg}}$ the average time in obtaining a valid design for a single assembly[1], $t_{c_{avg}}$ the average time in performing the structural calculus of that assembly, and $t_{f_{avg}}$ the average time in choosing a valid foundation[2]. Maximum time spent on computing all structures in the project W, is

$$t_{tot} = \sum_{i=1}^{\|W\|} \left(t_{d_{avg}} \cdot N_{a_i}\right) + \left(t_{c_{avg}} \cdot N_{a_i}\right) + \left(t_{f_{avg}} \cdot N_{f_i}\right) \tag{4.5}$$

Considering these three issues, a hand-design and a hand-calculation of the structure might result in human mistakes, besides being unfeasible in terms of time and effort invested.

We provide an algorithm, named *ResolveStructure*, that is in charge of automating the process. From the inventory I and the structure s_i, the algorithm carries out the step sequence described in Algorithm 4.1. As may be seen, the main actions are:

1.- A specific assembly design is set for a structure s_i by using the components existing in C, L, and P, that belong to the inventory I. For example, let a

[1] $t_{d_{avg}} \ll t_{c_{avg}}$
[2] $t_{f_{avg}} \ll t_{c_{avg}}$

Algorithm 4.1 ResolveStructure.

```
Input: $s_i, L, P, F, C, H$
Output: $s_i, feasibleSolution$
1: $feasibleSolution \leftarrow false$
2: $moreOptions \leftarrow true$
3: $hypothesis \leftarrow GetMoreRestrictiveHypothesis(H)$
4: while $!feasibleSolution \& moreOptions$ do
5:   if $(CheckAssemblyDesign(s_i.assembly))$ then
6:     $barsSet \leftarrow \emptyset$;
7:     for all $lintel \in s_i.assembly$ do
8:       $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
9:       $barsSet \leftarrow [barsSet, lintelBars]$
10:    end for
11:    for all $pole \in s_i.assembly$ do
12:      $posteBars \leftarrow GeneratePoleBarModel(pole)$
13:      $barsSet \leftarrow [barsXplane, posteBars]$
14:    end for
15:    for all $hypothesis \in H$ do
16:      $feasibleSolution \leftarrow RegulatedCalculus(barSet, hypothesis) \& feasibleSolution$
17:    end for
18:  end if
19: end while
20: for all $foundation \in s_i$ do
21:   $foundationFeasibility \leftarrow FALSE$
22:   $s_i.foundation.value \leftarrow GetNextFoundation(s_i, F)$
23:   while $!foundationFeasibility and s_i.foundation.value \neq \emptyset)$ do
24:     $subsidence \leftarrow CheckSubsidence(s_i.foundation.value)$
25:     $foundationFeasibility \leftarrow CheckOverturning(s_i.foundation.value)$
26:     $foundationFeasibility \leftarrow foundationFeasibility \& subsidence$
27:   end while
28:   $feasibleSolution \leftarrow foundationFeasibility and feasibleSolution$
29: end for
```

20-metres-span portal frame. A possible design to be calculated might be a truss lintel and two I-beam-poles with 106 cm^2 cross-section each. This step is executed every time a new design solution is requested (see line 2 in Algorithm 4.1).

2.- The assembly design is checked with railway regulations. Since the elements in C, L, and P, are mutually combinable, the railway companies are required to filter the assemblies that can be designed, but are not considered to be suitable in terms of best practices. Therefore, if the assembly is not accepted, a request for a new design solution is ordered (see line 3 in Algorithm 4.1).

3.- The set of bars representing the assembled structure is generated from the elements of the inventory that were selected in step 1. This inventory of those elements, as well as their constituent profiles. Every profile has its own properties (dimensions, section, weight per metre, thickness, moment of inertia, section modulus, and Young's elasticity modulus), that determine the behavior of the different bars that are generated in the model. The inventory also contains other properties of the elements, like the length, or the kind of joint to the rest of the elements within the structure (see lines 4 to 12 in Algorithm 4.1).

4.- The calculation of efforts and displacements at any point of the bars is performed by applying DSM to the generated model (see line 3 in Algorithm 4.2). We apply one DSM per hypothesis (see lines 13 to 15 in Algorithm 4.1).

5.- The structure is verified by applying the structural feasibility conditions proposed by the railway companies. This step consists of analyzing how the calculated efforts affect the resistance of the bars from a structural point of view. All the bars belonging to the model must be checked, so that it may be known where the structure collapses. The deflection of lintels must also be analyzed to verify whether they fit the railway company normative or not. If the assembly design is not feasible, the process goes to step 1 again, i.e., as many assemblies as necessary will be proposed until finding the feasible one (see lines 6 to 12 in Algorithm 4.2).

Algorithm 4.2 RegulatedCalculus.

```
Input: $barsSet, hypothesis$
Output: $feasibleCalculus$
1:  $conditions \leftarrow GetConditions(hypothesis)$
2:  $barSet \leftarrow ApplyConditions(barSet, conditions)$
3:  $DSM(barsSet)$
4:  $barsFeasibility \leftarrow TRUE$
5:  $lintelFeasibility \leftarrow TRUE$
6:  for all $bar \in barsSet$ do
7:      $barsFeasibility \leftarrow CheckStructuralFeasibility(bar) \& barsFeasibility$
8:  end for
9:  for all $lintel \in s_i.assembly$ do
10:     $lintelFeasibility \leftarrow CheckDeflection(lintel) \& lintelFeasibility$
11: end for
12: $feasibleCalculus \leftarrow barsFeasibility \& lintelsFeasibility$
```

6.- When a feasible assembly design is found, a specific and valid component of F is assigned to every foundation and anchor foundation. This step is executed once per each structure s_i because in DSM the poles are considered to be fixed in their base, i.e., the structural calculus may be carried out regardless of the foundations. So, once the forces generated on the base of the poles are known by applying DSM, foundations are analyzed through Sulzberger method [Sul45] in order to check their overturning and subsidence resistance (see lines 18 to 24 in Algorithm 4.1).

The algorithm finishes when either a valid assembly (regarding the design and structural feasibility) is found or when all the combinations have been tested and none of them resulted in a feasible solution.

Table 4.1: Selection order example of various poles with different heights and steel profiles.

Steel profile	Height (m)	Weight (kg)	Inertia moment (cm^4)	Order
HEB 240	9.75	811	11 259	1
HEB 400	11.65	1 805	57 680	2
HEB 550	14.00	2 796	136 691	3

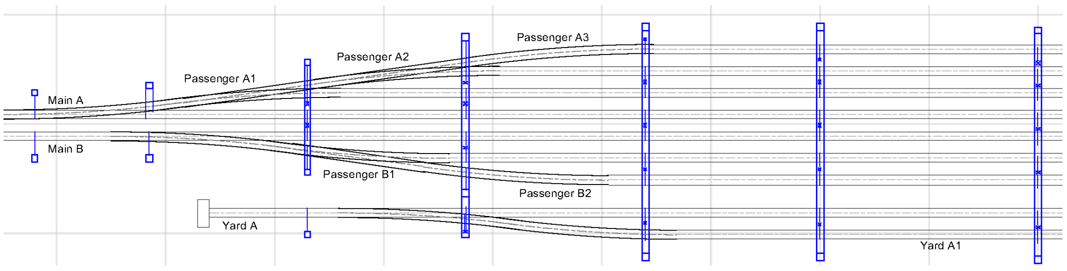


Figure 4.6: Planned portal frames and poles within a railway project.

The proposed algorithms are computationally complex mainly due to two factors. First, generating a feasible solution entails the combination with repetition of the inventory components per each element existing in the structure (lintels, poles, foundations, and cantilevers). The criteria used to choose these component combinations are based, not only on height restrictions of the catenaries situated under the portal frame, but also on a cost-optimization approach in terms of minimum weight design. According to this approach, the components are selected from the lowest to the highest weight until the whole structure is feasible. Table 4.1 represents an example of different poles, showing their steel profile, height, weight, moment of inertia, and selection order in Algorithms 4.1 and 4.2. The same criteria is analogously used for the rest of existing railway elements (lintels, cantilevers, and foundations). The second factor is that DSM consists of resolving a set of equations through matrix algebra. Hence, a high use of memory and processing resources is needed [FSSC11].

4.4 Tool Description

In the previous section, the process of designing and calculating a single catenary support structure has been presented. Nevertheless, when designing a real rail work project, such as planning the infrastructure of a railway station, or allocating all the poles along a 50 km track stretch, a high number of these structures may be needed. Figure 4.6 shows an example of a planned project containing rail tracks, portal frames, and single poles. All things considered, if all the structures within a project are desired to be calculated, the computational effort would increase enormously.

Our proposal automates the process of designing and calculating railway catenary support structures. RDIS is oriented to the computer-aided design of railway infrastructures. Through a user-friendly interface, users are able to design real rail work projects in detail, defining and modifying the elements that are planned (length

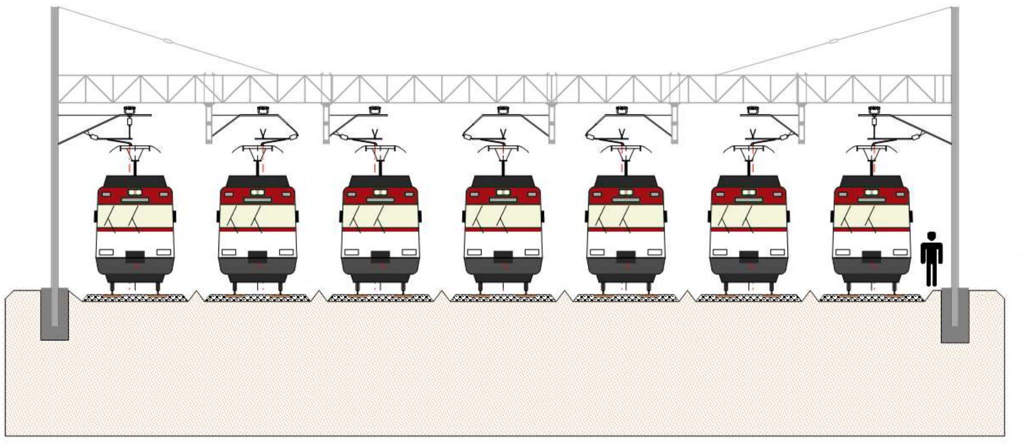


Figure 4.7: Assembled portal designed and calculated by the tool.

and direction of track stretches, type and mechanical tension of overhead wires, catenary height, cantilevers, poles and portal frames and their location along the tracks, etc.). Then, our tool gathers all the information related to each catenary support structure in the project and it is able to perform its design and calculation, allowing for structural constraints and normative regulations. Structural calculus is carried out through DSM, which is implemented within our tool. Moreover, it works with the railway inventory, so the components and materials of its stock list are included and considered.

Since RDIS is desired to be interactive, users are always informed about the results obtained, whether a feasible solution is achieved or not. On the one hand, if a feasible solution is achieved, the assembled structure is presented showing the following information:

- A CAD drawing, including the elements that compound the structure. Figure 4.7 shows an example of a truss portal frame that leans on two beam poles. The user is able to identify the specific components per element that were used in the solution obtained. Besides, since real components are used and a well-designed structure with real measurements is provided.
- Numerical results of the calculation process are also presented. Users can access detailed information at any point of the modelled bars: axial and shear stresses, bending moments, and displacements. Their maximum and minimum values are also obtained and located at specific points in the assembled structure, so that users can analyze its structural behaviour. Besides, different diagrams are also used to represent graphically these numerical values, as may be seen in Figures. 4.8, 4.9, 4.10, and 4.11.
- Overturning moments and compression forces of foundations where poles are embedded in.

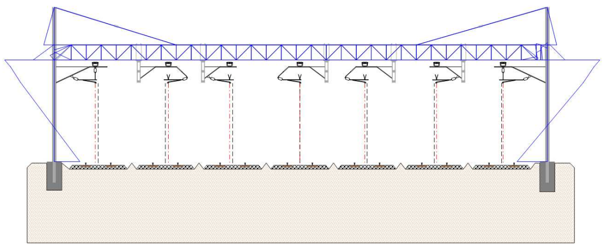


Figure 4.8: Portal frame bending moment obtained by the tool.

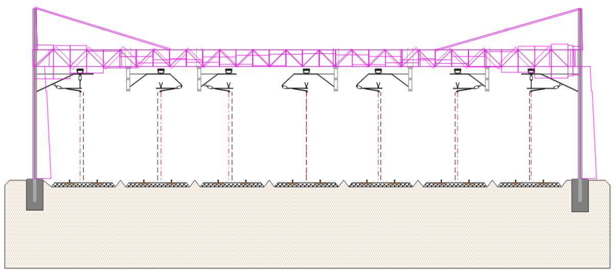


Figure 4.9: Portal frame axial moment obtained by the tool.

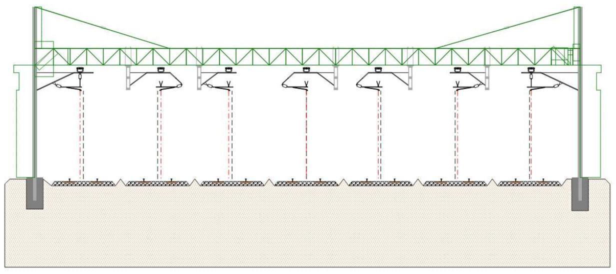


Figure 4.10: Portal frame cutting effort obtained by the tool.

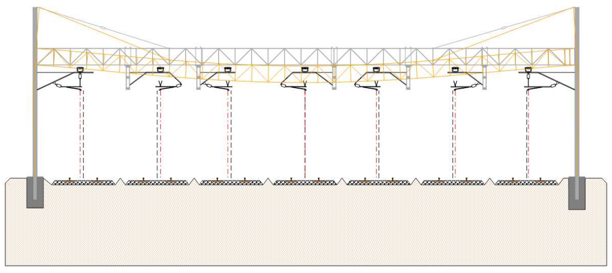


Figure 4.11: Portal frame deflection (magnified) calculated by the tool.

- Tension of the wires used to hold the lintels under a feasible deflection. Their locations along the lintel are also provided.

On the other hand, if RDIS is not able to find a feasible solution, the user is informed about the reasons why the structure is not valid: lintel over-deflection, bar collapse, foundation subsidence, etc. RDIS also shows graphic information, as will for a feasible solution. Invalid elements are identified with a different colour, so that users can analyze where and why the structure is collapsing.

4.5 Enhancements

To achieve this, we have divided the problem in two parts. Firstly we tackle the several hypotheses per planned structure. Secondly, we propose an optimization to calculate a real project with thousands planned structures. All the improvements proposed will be evaluated in the Chapter 6.

4.5.1 Single Structure

In the Algorithm 4.1 we could see the process for calculating a structure. The algorithm takes all possibles solutions, one by one from minimum to maximum cost, and calculates for all hypothesis. In the worst case, the total number of calculations, or *RegulatedCalculus* calls, can become equal to

$$C_{max} = N_{ai} \cdot N_H \tag{4.6}$$

where N_{ai} is the number of possible sets, previously described in Equation 4.3, and N_H is the number of hypotheses to satisfy. In the example described above, with 6 lintels, 23 poles, 25 bases, overhangs 7, and 13 different hypotheses, the maximum number of calculations is 99,070,062.

We confront the challenge of improving the calculus of a single structure from two perspectives. First, we propose a parallel approach. Each hypothesis will be calculated in a different thread. Second, we suggest a heuristic approach in collaboration with the expert's knowledge.

Parallel approach

Figure 4.12 shows the proposed architecture. In this approach we propose divide the calculus of each hypotheses into threads. The hypotheses calculate the same structure and all their characteristics must be shared as we can see in the Algorithm 4.3, lines 14 to 17. Between lines 4 to 13, we build the barsSet to all the hypotheses. The principal advantage of this solution is that we only have to check a common structure.

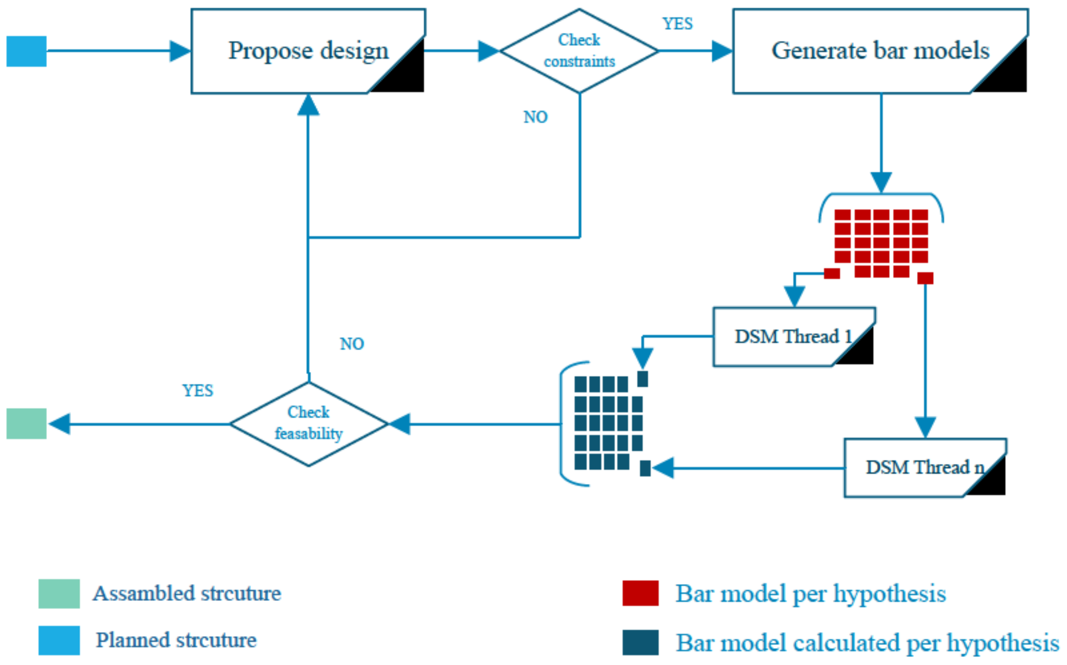


Figure 4.12: Parallel calculus of a structure.

Algorithm 4.3 ResolveStructureParallel.

```
Input: $s_i, L, P, F, C, H$
Output: $s_i, feasibleSolution$
 1: $feasibleSolution \leftarrow FALSE$
 2: $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
 3: while $!feasibleSolution \& (s_i.assembly \neq \emptyset)$ do
 4:   if $(CheckAssemblyDesign(s_i.assembly))$ then
 5:     $barsSet \leftarrow \emptyset$;
 6:     for all $lintel \in s_i.assembly$ do
 7:       $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
 8:       $barsSet \leftarrow [barsSet, lintelBars]$
 9:     end for
10:     for all $pole \in s_i.assembly$ do
11:       $posteBars \leftarrow GeneratePoleBarModel(pole)$
12:       $barsSet \leftarrow [barsXplane, posteBars]$
13:     end for
14:     for all $hypothesis \in H$ do
15:       $WaitForIdleCore()$
16:       $StartThread\{RegulatedCalculusParallel(barsSet, hypothesis, feasibleSolution)$
17:     end for
18:     if $!feasibleSolution$ then
19:       $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
20:     end if
21:   end if
22: end while
```

Heuristic approach

After assessments of sequential approach, we identify a major problem as the order of calculation of the hypotheses greatly affects the resolution time of the structure. Unnecessary calculations, if the design process begins with a less restrictive hypothesis, it is possible that the results are not valid for the most restrictive hypothesis; all the above calculation should be discarded, and it needs to start with a new option from the beginning. In order to solve this problem, the Algorithm 4.4 is presented.

As may be seen, major actions are:

1.- The most restrictive hypothesis is obtained in line 3. This hypothesis is selected according to the criteria established by the expert, and the description of the hypothesis given by the user.

2.- The calculation and design for this hypothesis is performed as if it were the only one in the system (see lines 7 to 23 in Algorithm 4.4).

3.- When a viable solution for the more restrictive hypothesis is obtained, the remaining hypotheses are calculated as in the sequential algorithm (see lines 25 to 38 in Algorithm 4.4).

Heuristic parallel approach

Finally, we have decided to combine the two principal approaches, parallel and heuristic. The Algorithm 4.5 describes the sequence to calculate with this approach.

The major contributions of this algorithm is when a feasible solution for the most restrictive hypothesis is obtained, the remaining hypotheses are calculated in different threads (see lines 35 to 38 in Algorithm 4.5).

Algorithm 4.4 ResolveStructureHeuristic.

```
Input: $s_i, L, P, F, C, H$
Output: $s_i, feasibleSolution$
 1: $feasibleSolution \leftarrow false$
 2: $moreOptions \leftarrow true$
 3: $hypothesis \leftarrow GetMoreRestrictiveHypothesis(H)$
 4: while $!feasibleSolution \& moreOptions$ do
 5:    $feasibleFirstHypothesis \leftarrow false$
 6:    $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
 7:    while $!feasibleFirstHypothesis \& (s_i.assembly \neq \emptyset)$ do
 8:       if $(CheckAssemblyDesign(s_i.assembly))$ then
 9:          $barsSet \leftarrow \emptyset;$
10:          for all $lintel \in s_i.assembly$ do
11:             $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
12:             $barsSet \leftarrow [barsSet, lintelBars]$
13:          end for
14:          for all $pole \in s_i.assembly$ do
15:             $posteBars \leftarrow GeneratePoleBarModel(pole)$
16:             $barsSet \leftarrow [barsXplane, posteBars]$
17:          end for
18:          $feasibleFirstHypothesis \leftarrow RegulatedCalculus(barSet, hypothesis)$
19:          if $!feasibleFirstHypothesis$ then
20:             $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
21:          end if
22:       end if
23:    end while
24:    if $feasibleFirstHypothesis$ then
25:       $hypothesis \leftarrow GetNextHypothesis(H)$
26:       while $hypothesis$ do
27:          $barsSet \leftarrow \emptyset;$
28:          for all $lintel \in s_i.assembly$ do
29:             $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
30:             $barsSet \leftarrow [barsSet, lintelBars]$
31:          end for
32:          for all $pole \in s_i.assembly$ do
33:             $posteBars \leftarrow GeneratePoleBarModel(pole)$
34:             $barsSet \leftarrow [barsXplane, posteBars]$
35:          end for
36:          $feasibleSolution \leftarrow RegulatedCalculus(barSet, hypothesis)$
37:          $hypothesis \leftarrow GetNextHypothesis(H)$
38:       end while
39:    else
40:       if $s_i.assembly = \emptyset$ then
41:          $moreOptions \leftarrow true$
42:       end if
43:    end if
44: end while
```

Algorithm 4.5 ResolveStructureHeuristicParallel.

```
Input: $s_i, L, P, F, C, H$
Output: $s_i, feasibleSolution$
 1: $feasibleSolution \leftarrow FALSE$
 2: $moreOptions \leftarrow TRUE$
 3: while $!feasibleSolution \& moreOptions$ do
 4:    $feasibleFirstHypothesis \leftarrow false$
 5:    $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
 6:    while $!feasibleFirstHypothesis \& (s_i.assembly \neq \emptyset)$ do
 7:       if $(CheckAssemblyDesign(s_i.assembly))$ then
 8:          $barsSet \leftarrow \emptyset$;
 9:          for all $lintel \in s_i.assembly$ do
10:             $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
11:             $barsSet \leftarrow [barsSet, lintelBars]$
12:          end for
13:          for all $pole \in s_i.assembly$ do
14:             $posteBars \leftarrow GeneratePoleBarModel(pole)$
15:             $barsSet \leftarrow [barsXplane, posteBars]$
16:          end for
17:          $hypothesis \leftarrow GetMoreRestrictiveHypothesis(H)$
18:          $feasibleFirstHypothesis \leftarrow RegulatedCalculus(barSet, hypothesis)$
19:          if $!feasibleFirstHypothesis$ then
20:             $s_i.assembly \leftarrow GetNextMinimumCostAssemblyDesign(s_i, L, P, C)$
21:          end if
22:       end if
23:    end while
24:    if $feasibleFirstHypothesis$ then
25:       $barsSet \leftarrow \emptyset$;
26:       for all $lintel \in s_i.assembly$ do
27:          $lintelBars \leftarrow GenerateLintelBarModel(lintel)$
28:          $barsSet \leftarrow [barsSet, lintelBars]$
29:       end for
30:       for all $pole \in s_i.assembly$ do
31:          $posteBars \leftarrow GeneratePoleBarModel(pole)$
32:          $barsSet \leftarrow [barsXplane, posteBars]$
33:       end for
34:       $hypothesis \leftarrow GetNextHypothesis(H)$
35:       while $hypothesis$ do
36:          $WaitForIdleCore()$
37:          $StartThreadRegulatedCalculusParallel(barSet, hypothesis, feasibleSolution)$
38:          $hypothesis \leftarrow GetNextHypothesis(H)$
39:       end while
40:    else
41:       if $s_i.assembly = \emptyset$ then
42:          $moreOptions \leftarrow TRUE$
43:       end if
44:    end if
45: end while
```

4.5.2 Several Structures

There may be several hundreds of railway catenary structures per project with heterogeneous design features. A new extension proposed for the *ResolveStructure* algorithm allows engineers to design and calculate as many single structures as there are within a project. Algorithm 4.6 shows this extension.

Algorithm 4.6 ResolveNStructures.

```
Input: $W, L, P, F, C, H$
1: for all $(s_i \in W)$ do
2:     $ResolveStructure(s_i, L, P, F, C, H)$
3: end for
```

Concerning the computational complexity, the higher the number of catenary support structures within a project, the greater the computational effort for a computer. In order to cope with this issue, we propose a new approach that exploits the ability of current computers to run concurrent tasks. The number of cores in computers has been increasing, so different design processes might be fulfilled in different threads at the same time, optimizing the overall performance.

Figure 4.13 shows how several design processes can be performed simultaneously. We start from the assumption that the initial set of planned structures has it own features and these features are independent of each other. If so, the process of resolving a single catenary support structure can be assigned to a calculating task. At the end of the process, every assembled solution is obtained by its assigned task. Due to this task independence, a coarse-grained parallelism approach has been adopted, i.e., a thread is in charge of resolving a structure. This design allows to

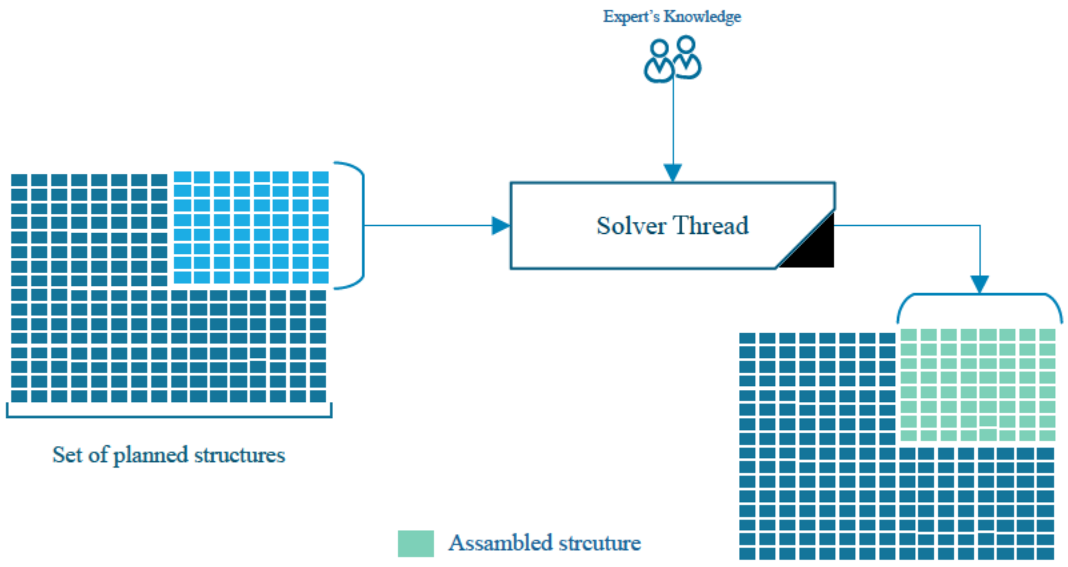


Figure 4.13: Parallel design of a structure.

dispatch each thread to a core when using multi-core computing, that the tasks can run concurrently. Threads are dispatched in an orderly fashion, so as not to saturate the processor. Initially, the algorithm starts with a determined number of tasks, corresponding with the number of available processor cores. Then, it waits for the termination of an executing task to dispatch the next one to the core which has just become available. In this way, the maximum number of executed concurrent tasks is equal to the number of processor cores, allowing to exploit the maximum degree of parallelism without saturating the computer. The railway inventory is the only element shared among all the tasks. Inventory data are fetched from disk to memory at the beginning of the algorithm, avoiding disk use overhead. During the process, these data are accessed by the threads in a read-only manner, so simultaneous accesses can be performed with no integrity problems. The use of threads modifies the pseudo code shown in Algorithm 4.6 to the stated in Algorithm 4.7.

Algorithm 4.7 ResolveNStructures.

```
Input: $W, L, P, F, C, H$
1: for all ($s_i \in W$) do
2:     $WaitForIdleCore()$
3:     $StartThread\{ResolveStructure(s_i, L, P, F, C, H)\}$
4: end for
```

4.6 Summary

We have presented an intelligent computer-aided design tool, named RDIS, that helps railway infrastructure designers to create and calculate safer and more efficient complex structures for railway electrification systems, especially overhead wire support structures such as cantilevers and frame portals. The final goal of that tool is to help designers to build infrastructure for the electric railway transport that are compliant with the existing normative and safe from the circulation and structural point of view to avoid dangerous situations that arise currently due to mistakes in design. RDIS follows a holistic approach to design, including facilities to automatically build from scratch detailed structures, including platforms, tracks, complex railway portals, catenaries and electrification systems. Our tool delivers an analysis of three-dimensional structures, detailed component calculations, election of minimum cost and minimum weight designs, and constructive plans adjusted to the millimeter of the optimal solution. The final result is the project documentation that can be provided to the building company.

The novelty of this research is the holistic approach of including all railway systems, with the intention of creating a knowledge system to incorporate automatically experts knowledge by using a rules engine, and the possibility of making automatically infrastructure design choosing optimal solutions, which will enhance the system efficiency and safety.

A dependency between RDIS and the position of the catenary has been detected. After several studies and after consultation with experts, we have come to the conclusion that the major problem stated in the catenary design was the design of overhead air switches. In this situation, two or more differnt catenaries are needed to garantee the power supply to both tracks. A bad design could result in an accident or incident with an important economic impact due to stop of service temporarily.

Chapter 5

Overhead Air Switches Design

5.1 Introduction

This chapter introduces the adaptation of the general framework described in the previous chapter to the design of overhead air switches (OCLS). In the design of overhead air switches we must consider the following aspects. The different types of switches, tangent and cross air switches, regulation normative, the different types of catenaries and scenarios, and metrics to find the optimal solution or set.

Finally, two versions of OCLS are introduced.

5.2 Overhead Air Switches

Catenaries are deployed along several spans of different lengths.They are composed of a messenger wire holding a contact wire that supplies the electric power to the pantograph of the train. Both wires are hung at a specific tension and are attached to each other at regular intervals by drop wires. These droppers are responsible for maintaining the contact wire hung at a constant height with a slight deflection. Hence an uniform contact between the pantograph and the wire as the train travels along the track is possible, avoiding any notches due to the pantograph thrust force. In addition, contact wires must be zigzagged slightly to the left and to the right of track axis so that the pantograph wears evenly its friction surface. This stagger is a critical issue to be analyzed in the problem presented.

In this chapter we focus on the critical study case of railway switches, where a train travelling along the straight track has to change to the diverging track. In this situation, two different catenaries are needed to guarantee the electricity supply to both tracks. Therefore, there will be overlapped spans along the switch stretch length. There are two major models of overhead air switches, tangent, and cross.

Tangent Air Switch

Figure 5.1 shows the configuration of a tangent air switch, where both contact wires do not cross at any point. As may be seen, the diverging track elevation span allows to lower the contact wire height, so that the pantograph can progressively change the rubbing wire while moving forward. The beginning of this change takes place at a characteristic point Cp where the heights of both elevation spans match (see Figure 5.2). The pantograph interacts with the contact wires of two different catenaries. Next, in the switching span, both catenaries are gradually separated. This allows the pantograph to lose the contact with the outgoing catenary and to get contact with the incoming catenary.

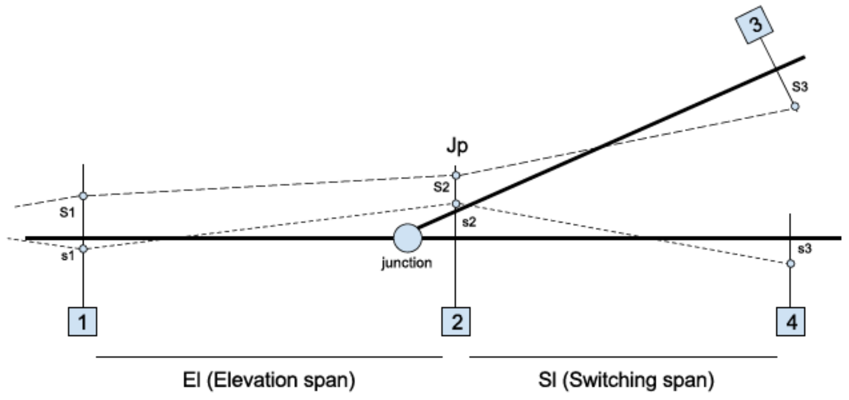


Figure 5.1: Ground plan of the configuration of a tangent air switch.

The point separating the two mentioned spans may be situated at any specific kilometric point along the switch and it is called junction point (Jp). Its value indicates a distance from the straight track axis to the diverging track one. Since this distance increases as moving forward along the switch, it is a critical design decision to choose a suitable position for the junction point: the shorter the distance between the two track axis is, the nearer both catenaries are each other, and vice versa. The junction point must belong to an interval defined by railway regulations. In Spain, for non high-speed tracks, the distance must be between 80 and 100 cm.

The configuration of a tangent air switch poses a set of restrictions to be ensured in the simulation of the train trajectory on the railway switch:

- When the train travels over the straight track, the pantograph should only rub the contact wire of this track. This avoids an excessive wear and tear of the diverging track contact wire, that may result in breakdowns and economical costs.
- Other possible flaw points to be analyzed may occur when the train is travelling along the diverging track. Firstly, as the entire pantograph surface is not

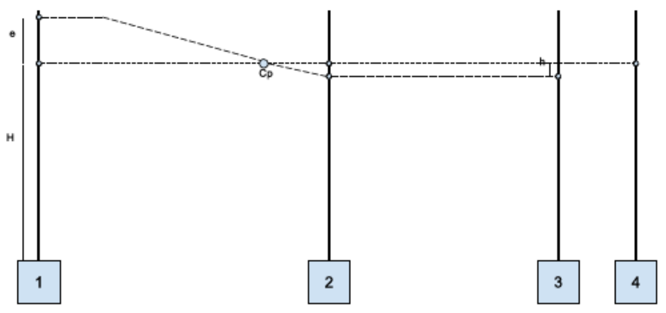


Figure 5.2: Elevation view of the configuration of a overhead air switch.

suitable for making contact with the contact wire, the pantograph should start rubbing it over its central part, called friction surface. Secondly, the beginning of the rub should be smooth and progressive. A hazardous breakdown of the catenary or the pantograph may occur otherwise. Thirdly, contact wires of both tracks must not cross, so as to avoid snagging the pantograph on the contact wire of the diverging track.

- Regardless of the track to be simulated, it is essential that the pantograph is always rubbing one of the contact wires, so that there is no electricity notches affecting the train movement.

Reliability and quality of service of railway systems will be improved by making optimal overhead knuckle junction designs. As a result, an economical cost reduction will also be possible. In the following sections, we propose a simulator model in order to achieve these optimal designs.

Cross Air Switch

Figure 5.3 shows the configuration of a cross air switch, where both contact wires cross at the switching span. The beginning of the change takes place at a characteristic point Cp where the heights of both elevation spans match (see Figure 5.4) and both wires are crossed. Around this point on, the pantograph will interact with the contact wires of two different catenaries. Next both catenaries are gradually separated. This allows the pantograph to lose the contact with the outgoing catenary and to get contact with the incoming catenary.

The point where the two mentioned spam are separated may be situated at any specific kilometric point along the switch and it is called junction point (Jp). Its value indicates a distance from the straight track axis to the diverging track one. Since this distance increases as moving forward along the switch, it is a critical design decision to choose a suitable position for the junction point. This point must belong to an interval defined by railway regulations. In Spain, for non high-speed tracks, the distance must be between 40 and 70 cm.

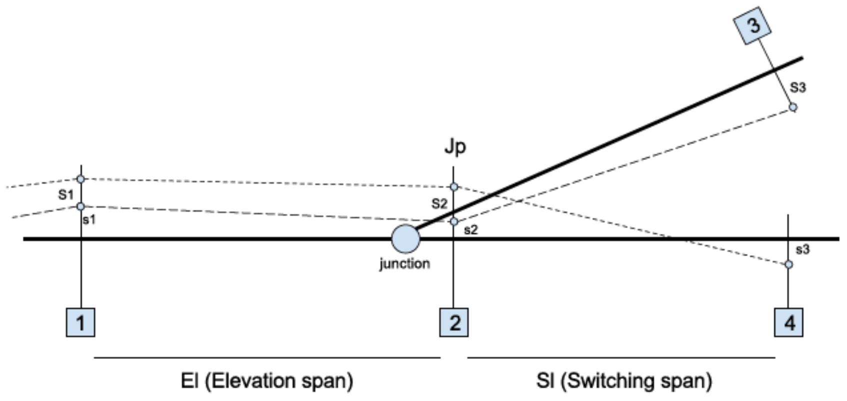


Figure 5.3: Ground plan of the configuration of a cross air switch.

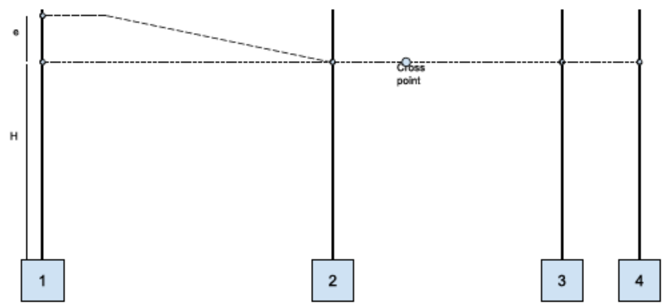


Figure 5.4: Elevation view of the configuration of a cross air switch.

5.3 Railway Switch Simulation Algorithm

This section presents OCLS, including input and output data analyzed in this thesis. By considering a set of parameters, the pantograph-catenary interaction along the switch is simulated. Thereby, an analysis of the problems that may occur in an overhead knuckle junction is possible.

A skeleton of the main steps of OCLS is shown in Algorithm 5.1. As may be seen, the simulator follows a 4-stage scheme in order to find an optimal solution to our overhead air switch design problem:

1.- **Definition of the problem to be simulated.** This problem is typically bounded to a single switch with an overhead air switch. However even with a single problem, there are many parameters to be defined, and some restrictions to be applied according to railway regulation, physic constraint of the components, etc. All these parameters are stored in a simulation configuration file that is used as input for the simulation.

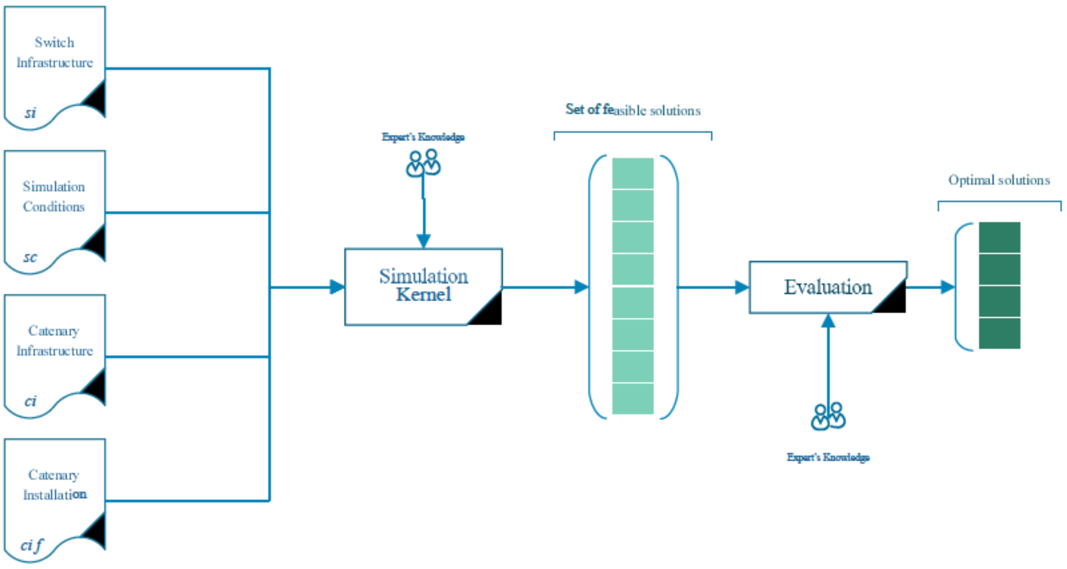


Figure 5.5: Simulator architecture.

2.- **Computation of all the possible solutions for the defined problem from the simulation configuration file data.** The simulator must generate (see Algorithm 5.1, line 2) and process (see Algorithm 5.1, line 5) all the possible solutions. For each processed solution, a log file is recorded including all useful output parameters (described in Section 5.4). The equations and algorithm proposed are described in Section 5.5. Even with the restrictions imposed in the scenario to be simulated, the range of solutions may reach to hundreds of thousands, thus being important to optimize the execution time using parallel programming techniques, as we propose in this thesis.

3.- **Finding the set of feasible solutions.** The former stage may provide two kind of solutions: feasible and unfeasible. Unfeasible solutions are those that do not satisfy railway regulations, that are not able to provide power to the train in some point, or that may have some parameters out of certain limits defined by railway experts. In order to reduce the size of the problem so as to get an optimal solution by decreasing the execution time, a procedure is executed to detect unfeasible solutions, thus being discarded in this step (see Algorithm 5.1, lines 9-16). A set of feasible solutions that satisfies all the former constraints is created. But even with this filter, the cardinality of this set may still range hundreds of thousands. The procedure to find an feasible solution is described in detail in subsection 5.6.1.

4.- **Finding and optimal solution.** This step of the simulator is in charge of gathering all data logged for the space of feasible solutions and uses them to achieve the optimal solution set from a technical point of view. We define several metrics to compare the merits of each solution and to find the optimal set. For each solution, the value of each metric is computed and stored into a vector. Then, a weighted equation is used to get the solution score (Algorithm 5.1 line 20). The optimal solution will be the highest scored. All the process is also described in detail in subsection 5.6.2.

Algorithm 5.1 Overhead air switch simulation algorithm.

```
Input: $si$ {switchInfrastructure}
Input: $ci$ {catenaryInfrastructure}
Input: $cif$ {catenaryInstallationFeatures}
Input: $sc$ {simulationConditions}
Output: $optimumDesign$
    {Let $E$ the set of scenarios to be simulated}
 1: $E \leftarrow \emptyset$
 2: $E \leftarrow BuildSimulationScenarios(si, ci, cif, sc)$
    {Let $S$ the set of possible solutions}
 3: $S \leftarrow \emptyset$
 4: for all $(e_i \in E)$ do
 5:    in parallel
 6:    $S \leftarrow [S, simulateScenario(e_i)]$
 7: end for
    {Let $F$ the set of feasible solutions}
 8: $F \leftarrow \emptyset$
 9: for all $(s_i \in S)$ do
10:    in parallel
11:    if $isFeasible(s_i)$ then
12:       $F \leftarrow [F, s_i]$
13:    else
14:       $discard(s_i)$
15:    end if
16: end for
    {Let $O$ the vector of optimality of the scenarios}
17: $O \leftarrow \emptyset$
18: for all $(f_i \in F)$ do
19:    in parallel
20:    $O \leftarrow [O, computeScenarioOptimality(f_i)]$
21: end for
22: $optimumDesign \leftarrow getOptimumDesign(O)$ {Gets the best scored solution}
23: return $optimumDesign$
```

5.4 Analysis of the Simulation Space

The simulation algorithm proposed has several data inputs, that may be gathered into four groups: switch infrastructure, simulation conditions, catenary geometry, and catenary installation features. Their details are stated below.

- *Switch infrastructure data (si).* Data related to railway infrastructure are shown

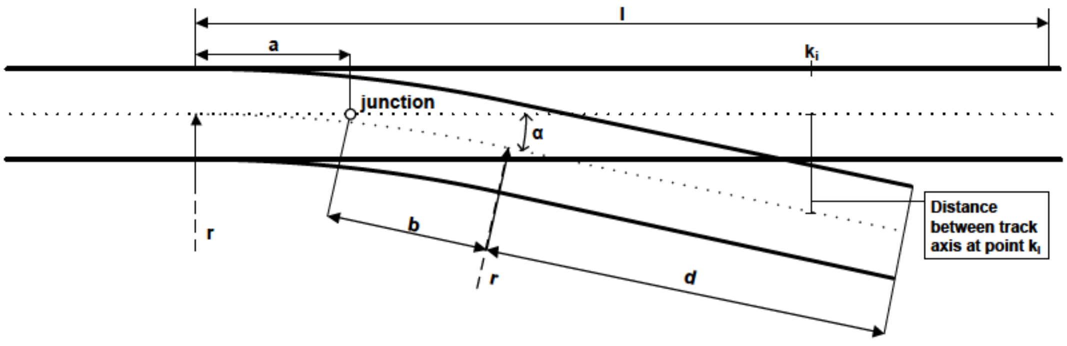


Figure 5.6: Scheme of a standard railway switch.

in Figure 5.6, which includes the scheme of a standard railway switch. Basing on these data and trigonometry equations, the angle at any point of the switch axis can be obtained, allowing the simulation of the pantograph position when travelling along the switch. A concrete switch is defined by the following variables:

- α. Tangential angle of the switch.
- a. Length from the switch starting point to its junction. As may be seen in Figure 5.6, the switch junction is the point where the tangential angle is applied.
- r. Radius of the switch curve stretch.
- b. Length from the switch junction to the ending point of its curve stretch.
- d. Straight length from the ending point of the switch curve stretch to the switch ending point.
- l. Total length of the switch. It is measured along the straight track from the switch starting point to the intersection between the straight track and a line at right angles to it from the switch ending point.

- *Simulation conditions data (sc).* In order to simulate the trajectory of the train along the railway switch, the following variables must be considered so as to get a reliable approach to the reality. Ts. Train speed, Ws. Wind speed, and Wd. Wind direction.

- *Catenary infrastructure geometry data (ci).* This group of variables contains the parameters related to the modeling of the ground plan and elevation of the catenary, i.e., its geometry. However, catenary geometry can support multiple configurations that are allowed for the given design problem. For example, one solution may be feasible whether the junction point is 90 or 120. These different configurations define the multiple solutions for the design problem. All of them must be simulated in order to check its feasibility, i.e., whether it is a valid catenary configuration, and its optimality, i.e., whether it is the best catenary configuration. Therefore, several catenary parameters are defined as an interval of test values, specified by a maximum, minimum, and a delta variation.

Catenary infrastructure geometry parameters are stated below, including those that are interval parameters:

- El. Elevation span length of the catenaries of both the straight and the diverging track.
- Sl. Switching span length of the catenaries of both the straight and the diverging track.
- $Jp = \{Jp_{min}, Jp_{max}, \Delta Jp\}$. Junction point where the main pole of the system is located. Its value is the distance at right angles from the straight track to the diverging one.
- $Cp = \{Cp_{min}, Cp_{max}, \Delta Cp\}$. Characteristic point where the heights of both catenaries match. It is always located at any point of the elevation span.
- $e = \{e_{min}, e_{max}, \Delta e\}$. Elevation of the diverging track catenary at its starting point.
- $h = \{h_{min}, h_{max}, \Delta h\}$. Elevation of the diverging track catenary at its ending point.
- $s1, s2, s3; s = \{s_{min}, s_{max}, \Delta s\}$. Staggers at the three main supporting points of the straight track catenary. Due to the catenary zigzagging, $s1 = s3 = -s2$.
- $S1, S2, S3; S = \{S_{min}, S_{max}, \Delta S\}$. Staggers at the three main supporting points of the diverging track catenary. As mentioned above, $S1 = S3 = -S2$.
- H. Height of the straight track contact wire.

• *Catenary installation mechanical features (cif)*. In addition to geometry data, it is necessary to consider some extra parameters in order to simulate a realistic behavior of the catenary installation features and pantograph-catenary interaction.

  - K_{max}. Catenary stiffness at the supporting point, i.e., where the catenary is stiffer.
  - K_{min}. Catenary stiffness in the middle of the span, where the catenary is more flexible.
  - D_{max}. Contact wire deflection in the middle of the span. It refers to the gap under the height set for this wire.
  - d. Distance to the first dropper. No contact wire deflection is applied along this distance.
  - N. Number of contact wires belonging to the catenary. There may be one or two contact wires.

- *T*,*A*. Tension and section area respectively of both the contact and the messenger wire. These parameters are responsible for the behavior of the wires under wind forces.

- *Ccs*. Type of current collection system, i.e., whether the contact wire is powered by AC or DC. It is needed to calculate the force that the pantograph exerts on the contact wire. ETI regulation [ETI08] is used for this aim.

- *epl*. Effective pantograph length, i.e., the length of the pantograph friction surface, that may rub with the contact wire.

All input parameters considered, let E the set of possible solutions to be simulated, $E = \{e_1, e_2, \ldots, e_n\}$. Each element e_j belonging to E is defined as a quadrupla: $e_j = \{si, ci_j, sc, cif\}$, where $ci_1, ci_2, \ldots, ci_n$ are all the possible combinations of catenary geometry infrastrucure data (ci_j) ranging the interval parameters.

In the Figure 5.7 the analysis of possible solutions are showed and the number of possible solutions can be derived from Equation 5.1. Table 5.1 shows an example of analysis.

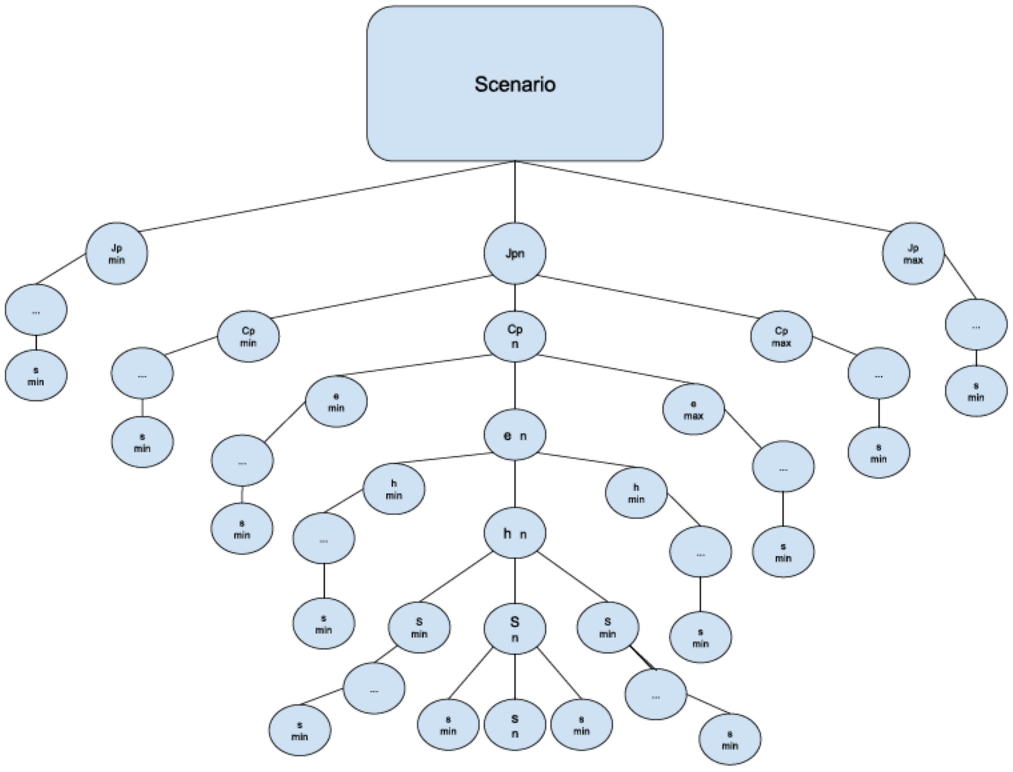


Figure 5.7: Expansion of possible solutions.

$$\begin{aligned} N &= \left(\frac{Jp_{max} - Jp_{min}}{\Delta Jp} + 1\right) \cdot \left(\frac{h_{max} - h_{min}}{\Delta h} + 1\right) \cdot \left(\frac{Cp_{max} - Cp_{min}}{\Delta Cp} + 1\right) \\ &\cdot \left(\frac{e_{max} - e_{min}}{\Delta e} + 1\right) \cdot \left(\frac{s_{max} - s_{min}}{\Delta s} + 1\right) \cdot \left(\frac{S_{max} - S_{min}}{\Delta S} + 1\right) \end{aligned} \tag{5.1}$$

Table 5.1: Possible solutions generated.

	Medium	**Extreme**
$(Jp_{max}, Jp_{min}, \Delta_{Jp})$	(80, 100, 5)	(80, 100,1)
$(h_{max}, h_{min}, \Delta_h)$	(10, 50, 10)	(10, 50, 1)
$(Cp_{max}, Cp_{min}, \Delta_{Cp})$	(10, 40, 5)	(10, 40, 1)
$(e_{max}, e_{min}, \Delta_e)$	(600, 300, 100)	(600, 300, 1)
$(s_{max}, s_{min}, \Delta_s)$	(250, 50, 50)	(250, 50, 1)
$(S_{max}, S_{min}, \Delta_S)$	(-250, -50, 50)	(-250, -50, 1)
Total	**17.500**	**324.581.270.391**

Output data

In order to achieve the desired design solution, we propose a simulator that provides the following output data for each simulation step of a scenario i.

- k_i. Kilometric point where an iteration of the simulation algorithm has been executed.
- Cs_i, Cd_i. Output data of the catenaries belonging to the straight track and the diverging track respectively. Both, Cs_i and Cd_i, include the following values:
  - p_i. Elevation applied to the contact wire by the pantograph. This is only possible when both are in contact with each other.
  - st_i. Stagger of the contact wire from the pantograph asix. This stagger may take a negative or positive value, depending on whether it is located at the left side of the pantograph axis or at the right one, respectively.
  - y_i. Height of the contact wire. y_i already includes the elevation e_i if the pantograph is rubbing this catenary.
  - d_i. Distance between the contact wire position in iteration i and its position in iteration *i-1*.
  - β_i. Angle between a horizontal line and the line joining the contact wire position in iteration i and its position in iteration *i-1*.
- Sd_i. Switching distance at this simulation kilometric point. It increases as moving forward along the switch.

- Cr_i. Identification of the catenary being rubbed by the pantograph. Both catenaries may have contact with the pantograph at the same time. If so, y_i of both Cs_i and Cd_i have the same values.

Having all these output data it will allow to reproduce the simulation results a posteriori showing relevant data as wire positions, alarms, etc. and can help to provide users relevant information [Kul94]. Moreover, output data will allow to analyze the progressive changes produced throughout the iterations in order to achieve an optimal solution (see subsection 5.6.2).

5.5 Simulation Kernel

In this section we present the kernel of the simulator, as it executes for each possible scenario $e_j \in E$ the mathematical model to compute all output parameters seen before. Table 5.2 shows the major intermediate calculus in the algorithm.

Table 5.2: Significant data calculated.

Symbol	Description
k	Significant data calculate
y	Contact wire position
st	Contact wire stagger
w	Transversal wind force
E	Catenary elasticty
sd	Switching distance

The algorithm *simulateScenario*(e) starts in the kilometer point (k_{jinit}) corresponding to the first pole of the switch infrastructure (see Figure 5.6), and it runs simulation steps along the straight (Cs_j) and diverging tracks (Cd_j) until the elevation and switching span lengths are covered (k_{jend}). The number of simulation steps depends on the length of track to simulate $(k_{jinit} - k_{jend})$ and the pantograph displacement (δ_j) for each step, as follows:

$$n = \frac{k_{jinit} - k_{jend}}{\delta_j} \tag{5.2}$$

where k_{jinit}, k_{jend}, and δ_j are configuration parameters for each simulation. However, in order to get valid accurated results so as to propose a design solution, the pantograph displacement δ_j should be smaller than a few millimeters. By default, all simulations are run using the maximum accuracy, which means 1 millimeter displacement per step. Considering that a typical switch installation may require to simulate 200 meters, the number of simulation steps for each scenario will range around 200,000 steps for the catenary of the straight track and the same for the diverging track. To log all the results, every step writes output data to a file, so a statistical study or a graphical reproduction of the simulation may be possible. The details of the algorithm *simulateScenario*(e_j) are described below for a single step

Algorithm 5.2 Procedure *simulateScenario*(e) to simulate all scenarios.

```
Input: $e$ {Scenario}
Output: $S$ {Scenario simulated}
 1: $n \leftarrow \frac{k_j init - k_j end}{\delta_j}$
 2: $k_i \leftarrow k_j init$
 3: while $k_i < k_j end$ do
 4:     $pos \leftarrow ComputePantographPosition(k_i)$
 5:     $Y \leftarrow ComputeContactWiresPosiotions(pos)$
 6:     for all $y \in Y$ do
 7:         $stagger \leftarrow ComputeStagger(y)$
 8:         $y \leftarrow CalculateLateralDisplacement(y)$
 9:         $stagger \leftarrow CalculateVerticalDisplacement(stagger)$
10:     end for
11:     $h_{initial} \leftarrow CalculateHeight(Y, pos)$
12:     $pos \leftarrow LateralDisplacempentPanto(pos, h_{initial})$
13:     $h_i \leftarrow CalculateHeight(Y, pos)$
14:     $e_i \leftarrow ETI(c_i f, T_s)$
15:     $applyElevationContact(Y, h_i)$
16:     write results
17:     $k_i \leftarrow k_i + \delta_j$
18: end while
```

i of the simulation scenario j. For the sake of simplicity, only main equations are stated.

First, the **pantograph position k_i is computed** for each simulation step i by increasing the former pantograph position in the track with the pantograph displacement defined (see Equation 5.3). For the first step, the initial position is set to the position of the first pole of the switch infrastructure k_{jinit}.

$$k_i = k_{i-1} + \delta_j \tag{5.3}$$

Second step is **computing contact wires position**. This step is in charge of calculating the position of the contact wires accurately for the pantograph position at this simulation step. For the straight track, the position of the contact wire is modeled depending on the positions of the first and the last droppers, and the maximum deflection defined as input data. The wire has a constant height H between the beggining of the span to the first dropper, and between the last dropper to the end of the span. However, this base value must be modified, because we must respect the deflection in the middle of the catenary span, between the first and the last droppers. Equation 5.4 is applied along the span to compute the base wire height for that point.

$$y_i(k_i) = \begin{cases} H & k_i < d \\ H - deflection(k_i) & d \leq k_i < l - d \\ H & k_i \geq l - d \end{cases} \tag{5.4}$$

where d is the distance from the beggining of the span to the first dropper, and

l is El or Sl, depending on whether the current span is the elevation span or the switching span.

Thus, the deflection of the contact wire due to the messenger wire tension must be considered in this step. Since there are droppers linking the messenger wire to the contact wire, the contact wire height never has a deflection larger than the limit defined D_{max}. Equation 5.5 is used to compute wire deflection at each point.

$$deflection(k_i) = \left(4 \cdot \frac{D_{max}}{(l-d)^2}\right) \cdot \left((l-d) \cdot (k_i - d) - (k_i - d)^2\right) \tag{5.5}$$

Equation 5.5 is derived from Equation 5.6, a more general expression of wires deflection used in railway domain.

$$y = \frac{p \cdot x^2}{2 \cdot T} \tag{5.6}$$

where p is the weight per meter of the wire, T is the wire tension, and x is the distance between the calculated point and the closest support.

Equations 5.4 and 5.5 are applied in every span except in the diverging track elevation span. Contact wire deflection of the latter span is not considered, and contact wire height is obtained by applying trigonometry in the scheme of the right side of Figure 3.1.

Next, we have to **compute wire stagger** for the wires being rubbed by the pantograph. The stagger of each contact wire st_i at this kilometric point is computed as the distance from the wire to the pantograph axis. This distance is measured at right angles to the track axis. This parameter is also very important in order to find an optimal solution.

Fourth, we must **apply environmental conditions**. Depending on the simulation conditions defined by the input parameters, wire stagger could be modified due to several environmental aspects.

Equation 5.7 is used in the proposed simulator to include the transversal wind force. The equation follows the standard EN 50119 [BE09b] to compute the horizontal displacement of the contact wire due to that wind in standard conditions (15 degrees and 600 meters over the sea).

$$W_c = Pv_{ContWire} + Pv_{MesWire} \tag{5.7}$$

where

$$Pv = q_k \cdot G_c \cdot dWire \quad being \quad q_k = \frac{1}{2} G_q \cdot G_t \cdot \rho W s^2 \tag{5.8}$$

G_q reflects the wind burst, with a value of 2.05, as defined by the standard ENV 1991-2-4:1995 (see page 42 in [BE09b]), G_t is a terrain factor, Ws is the wind speed, ρ is a factor equal to $1.225\frac{kg}{m^3}$, and $dWire$ is the diameter of the wire, obtained from its section area A.

Equation 5.9 is applied to calculate wire contact horizontal displacement:

$$w_i(k_i) = \left(\frac{W_c}{T}\right) \cdot \left(\frac{k_i^2}{2}\right) \tag{5.9}$$

where W_c is the resulting wind force and T is the tension due to the catenary. The result is a quadratic curve, similar to the wire deflection.

Fifth step consists of **determining pantograph height** e_i. Once calculated the contact wire positions, the pantograph height must be computed for this simulation step as follows. Since there are two catenaries, for the main and the diverging tracks, the pantograph height will be the minimum height of the wires that are within the projection of its friction surface, i.e., the rubbing contact wire will be the lower one. If both wires are out of the friction surface projection, then the pantograph height will be considered as ∞ to indicate an error.

The result of this step allows to divide the simulation space in three areas that are very important to find an optimal design solution in subsection 5.6.2:

1.- Cs pantograph interaction. All the simulation steps where the pantograph makes contact with the contact wire of the straight track.

2.- Cd pantograph interaction. All the simulation steps where the pantograph makes contact with the contact wire of the diverging track.

3.- $Cs \wedge Cd$ pantograph interaction. All the simulation steps where the pantograph makes contact with both contact wires of the straight and the diverging track.

The sixth step consists of **modifying contact wire elevation and angle** due to the pantograph interaction. Some parameters needed, such as elasticity in the center of the catenary spans and in the cantilevers, are received as input parameters in the catenary installation features cif. Train speed Ts, contained in sc, has also to be considered. In order to know the elevation, the pantograph pressure over the wires must be computed, as shown in Equation 5.10 that follows ETI regulation [ETI08].

$$F_m = \begin{cases} 0.00097 \cdot Ts^2 + 70 & \text{Ccs is A.C.} \\ 0.00097 \cdot Ts^2 + 110 & \text{Ccs is D.C. 3.0 kV} \\ 0.00228 \cdot Ts^2 + 90 & \text{Ccs is D.C. 1.5 kV} \end{cases} \tag{5.10}$$

Next, the elasticity is computed for the catenary point using Equation 5.11, where the denominator is the stiffness at that point. This equation is detailed in [BBLS00, KCP+07, WB99].

$$E(k_i) = \frac{1}{K_0\left(1 - \alpha \cos\left(\frac{2\pi k_i}{l}\right)\right)} \tag{5.11}$$

where

$$K_0 = \frac{K_{max} + K_{min}}{2} \quad and \quad \alpha = \frac{K_{max} - K_{min}}{K_{max} + K_{min}} \tag{5.12}$$

The elevation of the contact wire due to the pantograph is obtained using Equation 5.13.

$$e(k_i) = E(k_i) \cdot F_m \tag{5.13}$$

After determining the elevation produced by the pantograph, the definitive contact wire height must be computed as expressed in Equation 5.14.

$$y_i(k_i) = y_i(k_i) + e(k_i) \tag{5.14}$$

Thus, the contact wire position at a kilometric point k_i can be defined as the following equation:

$$Wp_i(k_i) = (st_i(k_i) + w_i(k_i), y_i(k_i)) \tag{5.15}$$

Seventh step is **computing switching distance** in k_i. Since this distance, measured from the straight track axis to the diverging track one, increases as moving forward along the switch, its value is a crucial issue to find an optimal solution.

Eighth step is **applying the lateral displacements** depending of the position in the track and the height.

Last step consists of **storing the results to file**. Once computed all the significant parameters, the target output data of the simulation step are written to the simulation scenario log file. Output data were defined in Section 5.4.

5.6 Choosing Optimal Solution Set

After computing all possible scenarios, we have to select the optimal solution. The selection of the choosing optimal set has been divided in two phases. In the first phase, all the possible solutions are classified between feasible solutions and unfeasible solutions during simulation. Next, the feasible set is analyzed to discover the optimal solution set.

5.6.1 Set of Feasible Solutions

The final goal of our simulator is to propose an optimal design solution for the overhead air switch design problem. Currently, most of the railway regulations propose a junction point value of 90 to install the junction pole in the tangent air switch and 55 in the cross air switch. As we said before, better solutions might exist in the interval $(80, 100)$ for tangent, and $(40, 75)$ for cross. We propose to analyze the results of the former simulations of each scenario and compute metrics to propose an optimal solution.

However, not all the executed simulation scenarios for a single installation generate feasible solutions as there may be constructive errors or output data not allowed by the railway regulations. As we have to cope with hundreds of thousands of possible

Algorithm 5.3 Procedure $isFeasible(s)$ to filter out unfeasible solutions.

```
Input: $s$ {Scenario simulation solution}
Output: $feasible$ {Boolean result indicating feasible or unfeasible solution}
 1: $feasible \leftarrow TRUE$
 2: $j \leftarrow 0$
 3: $s_j \leftarrow 0$ {Structure to log results from a step of the simulation}
 4: read $s_j$ from $s.logfile$
 5: while $(s_j \neq 0) \wedge feasible$ do
 6:   if $(p_j = \infty) \vee (|straight.st_j| > \frac{epl}{2}) \vee (|diverging.st_j| > \frac{epl}{2}) \vee intersect(straight.Wp_j, diverging.Wp_j)$ then
 7:     $feasible \leftarrow FALSE$
 8:     write $s$ and failure cause to $unfeasibles.logfile$
 9:   end if
10:   $j \leftarrow j + 1$
11:   read $s_j$ from $s.logfile$
12: end while
13: return $feasible$
```

scenarios, the first operation accomplished by the simulator is to discard unfeasible solutions from all the scenarios existing in E in order to reduce the optimization problem size. Unfeasible solutions are due to three major causes:

1.- The contact wires do not interact with the pantograph. That means that pantograph height e_i is set to ∞ at any point of the scenario.

2.- The stagger st_i of any of the wires interacting with the pantograph is larger than the maximum stagger allowed by the railway regulations. By default, the maximum is half of the pantograph friction surface.

3.- Contact wires of straight and diverging tracks intersect. This fact can be deduced from the output data as we have the contact wire position Wp_i for each wire and at every simulation step i. (Only in tangent air switches)

Algorithm 5.3 shows the proposed procedure defined to identify unfeasible solutions. This procedure is carried out to analyze the log file of every solution provided by the simulation of all the scenarios. After the execution of this step of the simulator, unfeasible solutions are discarded and the set of feasible solutions F is created.

5.6.2 Finding Optimal Solutions

Once the set of feasible solutions F has been created, the simulator can advance to the next phase: obtaining optimal solutions. Thus, for each feasible scenario simulated $e_i \in F$, several metrics must be calculated to choose an optimal solution. However, the issue of getting such a solution for the overhead air switch design problem is not defined by any regulation, i.e., it is still an open research topic. We have closely cooperated with railway experts to define the metrics to be used in both types.

According to pantograph-catenary interaction, the simulation space is divided into three zones:

- **STI**. Straight track pantograph-catenary interaction.
- **DTI**. Diverging track pantograph-catenary interaction.
- **BTI**. Both tracks pantograph-catenary interaction.

BTI zone is the critical area to find the optimal solution, but the transition from the other two zones to BTI zone is also important because the contact wire must respect a certain angle when starts rubbing with the pantograph.

An optimal solution for tangent air switch is the one that satisfies the following conditions:

1.- Maximizing the average distance between contact wires of straight and diverging tracks, defined as metric *M1*. It forces the wires to be as far apart as possible. This will avoid potential problems due to high electrical voltages flowing through the wires.

2.- Minimizing the variance of stagger of the diverging track contact wire, defined as metric *M2*. This metric is intended to avoid too many sudden changes of position of the contact wire on the pantograph.

3.- Minimizing the average symmetry between contact wires of straight and diverging tracks, defined as metric *M3*. It measures the difference between the stagger of both contact wires, which are sought to be as symmetrical as possible to the axis of the pantograph, thus avoiding a pantograph tilt towards one of the sides.

4.- Minimizing the input angle of the diverging track contact wire in the pantograph along the transitions STI-BTI, defined as metric *M4*. This angle is intended to be as low as possible, thus avoiding a sharp blow on the pantograph. By smoothing the entry of the contact wire in the pantograph, damages and premature wear of the wire can be decreased.

5.- Minimizing the output angle of the straight track contact wire out of the pantograph along the transitions BTI-DTI, defined as metric *M5*. This angle is desired to be as low as possible, thus avoiding a sharp blow on the pantograph. This metric is particularly important when simulating a train the other way around, i.e., from the diverging track to the straight track.

6.- Minimizing the average of stagger of the diverging track contact wire, defined as metric *M6*. This metric ensures that the diverging track contact wire is as focused as possible to the axis of the pantograph, thus avoiding the approximation of the thread to the edges of the pantograph and ensures that the contact wire is always going to enter a valid area.

Algorithm 5.4 shows the simulator procedures in charge of computing the metrics for a single feasible scenario simulated e_i. In the algorithm, we can observe that the final value of each scenario is the sum of all the metrics. Since the values of the

Algorithm 5.4 Procedure *computeScenarioOptimality*(f) to calculate optimal solution.

```
Input: $f$ {Scenario simulation solution (feasible)}
Output: $o$ {Solution score}
1: $o \leftarrow 0$
2: $M1 \leftarrow computeMetricM1(f)$
3: $M2 \leftarrow computeMetricM2(f)$
4: $M3 \leftarrow computeMetricM3(f)$
5: $M4 \leftarrow computeMetricM4(f)$
6: $M5 \leftarrow computeMetricM5(f)$
7: $M6 \leftarrow computeMetricM6(f)$
8: $o \leftarrow (W_{M1} \cdot M1) + (W_{M2} \cdot M2) + (W_{M3} \cdot M3) + (W_{M4} \cdot M4) + (W_{M5} \cdot M5) + (W_{M6} \cdot M6)$
9: return $o$
```

metrics are very different, they have to be normalized. As seen before, the metric *M1* is maximized, but the overall function, that includes all the metrics, must be minimized. To resolve this conflict, we change the sign of the *M1* metric value so as to normalize the result. In order to compute the overall function, we use a specific weight to confer greater or lesser importance on each metric.

At the end, the optimal scenario is the one that minimizes the overall function value. This algorithm is only applied to the set of feasible solutions. Since there are metrics inversely correlated, it is impossible to find a scenario having the best value per metric, being possible to have a scenario better in some metrics and worse in other ones. According to this fact, the algorithm finds the best scenario considering an overall function of all metrics.

5.7 Matching OCLS to Generic Framework

The simulation model is composed of the simulation and evaluation components. A generation engine produces different scenarios to be tested, dispatching simulations concurrently to any CPU available. Data proceeding from other actors and expert knowledge feed both generation and evaluation engines, in order to reduce the amount of generated scenarios, filter the number of feasible scenarios, and calculate the degree of goodness in order to obtain the best ones.

In this section we show that our simulator framework also matches OCLS.

5.7.1 Layer 1: Trade-off Between Accuracy and Complexity

Time spent on simulating one single scenario is a critical issue with regard to simulator's usability and productivity. Furthermore, the more accurate a simulation is, the longer the time required to carry on with it will be. When simulating overhead air switches, step size determines the detail level of the simulation. In order to obtain results as real as possible, simulations have to be carried out by millimetre-steps. By this way, pantograph and wires output data are more accurate. On the other hand, more steps imply more computing resources, performing more calculations in order to obtain a larger amount of output data.

For each simulation step, the processor has to solve the equations shown before. Modern CPU cores can perform millions of operations per seconds, which implies no more than a millisecond spent on solving simulation steps and writing output data to files. The problem is that the largest railway switches may have a length up to 1000 metres. This implies that a scenario may require about one second to be simulated (using steps of one millimetre). This amount of time is acceptable when dealing with just one single scenario, but when dealing with thousands or millions of scenarios (see further sections) high-performance techniques are necessary to run different simulations concurrently, taking advantage of multi-core or multi-processor systems.

5.7.2 Layer 2: Generation and Evaluation of Possible Solutions

As previously mentioned, we state that an efficient simulator should evaluate and simulate a set of solutions with a minimal user involvement. New generation simulators should be capable of, starting from a range of possible parameters, proposing and evaluating new designs. The proposed framework aims this objective through introducing a new component in the simulator.

This component is a scenario generator, which wraps the simulation model (simulation and evaluation of one single scenario) generating different solutions to be evaluated. It generates new scenarios through variations on the input data, allowing experimentation with different simulation parameters, different components, or different domain restrictions. Those scenarios are provided to the simulation engine, which carries on with the simulation as described in the previous section.

Generating and evaluating multiple scenarios automatically allows the simulator to test different solutions, thus providing a faster way of exploring the solution space. Rather than obtaining one single solution, by this way a set of feasible solutions is obtained, and the user can select the best one. Moreover, as we describe in subsection 5.7.4, enhancing the simulator with optimization metrics or some expert's knowledge brings the opportunity of performing an automatic guided search of the solution space.

In order to cope with this issue, our simulator implements a new module, accountable for generating multiple scenarios. In order to result this, we change the input data definition. With regard to the simulation model, input parameters are transformed from scalar values (e.g. train speed $Ts = 220$ km/h) to an interval of test values defined by the user, who specifies a maximum, a minimum, and a delta variation. Let P an interval parameter, $P = \{P_j / P_{min} \leq P_j \leq P_{max}; P_j = P_{min} + j \cdot \Delta P; j \in \mathbb{N}^*\}$. By this way we define a complete set different values, and different simulations each one using a different value of this parameter have to be performed.

Of course, introducing variations in several parameters at the same time increases the number of scenarios exponentially, since we have to perform combinations with elements of the two (or more) sets. An advantage of this explosion is that the solution space is rapidly explored, evaluating a huge number of solutions from the problem space. As a drawback, a large amount of computing power is required to perform a large number of simulations, so the trade-off between accuracy and complexity previously mentioned has great significance. In this particular case, the number of combinations of catenary configuration parameters can reach over one million. Each one of these combinations is a possible solution of the design problem. In order to carry out the search efficiently, multiple scenarios can be simulated concurrently, dispatching simulation kernels performing different scenarios to different CPUs. In Chapter 6, an illustrative example will be evaluated, indicating input parameters variation, number of scenarios generated, evaluated, and time consumed in simulation.

5.7.3 Layer 3: Other Actors

The amount of scenarios outputted from the previous layer would be unmanageable by the user if no more filtering is applied apart from the physical domain restrictions. In order to increase the functionality and productivity of the simulator, we have to take into account the different stakeholders which take part in the design process. Different determining factors may fall into this category: legislation and normative, cost limitations, provider or client restrictions, available stock, and so on. There are two ways of considering such participants. The first is enhancing the set of evaluation rules, checking not only physical domain restrictions, but also specific restrictions from different sources. The second is restricting values of the input parameters, limiting the generation of new scenarios to only those which may comply with those restrictions.

Our simulator may take into account different normative currently in force. So additional evaluation rules have been implemented in order to check if a solution is feasible or not, counting:

- Normative EN-50119 [BE09b]. This normative stipulates different restrictions with regard to overhead line deployment (minimum and maximum height, droppers configuration, etc.). Moreover, it stipulates different restrictions about the way the pantograph makes contact with the wires.
- Normative EN-15273 [BE09a]. This normative stipulates maximum width and headroom in railway lines, stating that certain area around the rails have to be free of obstacles. This restriction has effect in the way the overhead lines are deployed.

5.7.4 Layer 4: Expert's Knowledge

Even if additional restrictions from other actors are considered to filter the number of feasible solutions, the resulting set might be too large to be useful. Besides, the user doesn't know what solutions are better than the others. Expert's knowledge can be applied in order to discriminate, from the set of feasible solutions, what are the best ones. In order to accomplish this, first of all we have to declare what optimization metrics are going to be followed, i.e. the criteria that determines if a solution is better than other. Then, that criteria can be applied by two ways: the first is an enhancing of the set of evaluation rules with a new set of rules which don't check the feasibility of the solution, but score the solution following the proposed criteria; the second is modifying again the generation of new scenarios trying to seek those scenarios that best fit with the proposed criteria, in the same way as MOEAs try to reach the optimal solution. The first approach may lead to an exhaustive search in the solutions space, but as drawback, all solutions must be simulated. The second approach saves time by driving a guided search, but a number of solutions can remain "untested".

With regard to our overhead air switch simulator, several optimization metrics have been chosen and described before. The first metric is maximized, but the overall function, that includes all the metrics, must be minimized. To resolve this conflict, we change the sign of the first metric value so as to normalize the result. In order to compute the overall function, we use a specific weight to confer greater or lesser importance on each metric. At the end, the optimal scenario is the one that minimizes the overall function value. Since there are metrics inversely correlated, it is impossible to find a scenario having the best value per metric, being possible to have a scenario better in some metrics and worse in other ones. According to this fact, the framework finds best scenarios considering an overall function of all metrics, following a Pareto front.

5.8 Shared Memory Implementation

This version of the simulator is composed of three principal steps: generation, simulation, and evaluation. In the first step (Figure 5.8), the Generator reads the initial scenario and all the parameters of the simulation. Next, all the possible solutions are generated in function of the local inventory and the expert's knowledge. The inventory can change depending of the country. A set of possible solutions are generated as a result of this step. Each possible solution is represented in a small file that contains all the data needed to simulate in the next step. We can generate a massive number of this files. An extreme and a medium example are showed in Table 5.1.

Figure 5.9 represents the second step of the simulator. First, a set of threads is created to simulate and the possible solutions are distributed fairly, the number of threads is configurable and depends on the workstation. Each thread executes Algorithm 5.2 and Algorithm 5.3 per possible solution. If we detect troubles with the generated solution, the simulation stops immediately and the possible solution is labeled as unfeasible solution. All unfeasible solutions and their simulation files are erased from the system. If a simulation finishes correctly, the thread writes the name of the feasible solution to a common index, and it keeps the simulation file to be evaluated later.

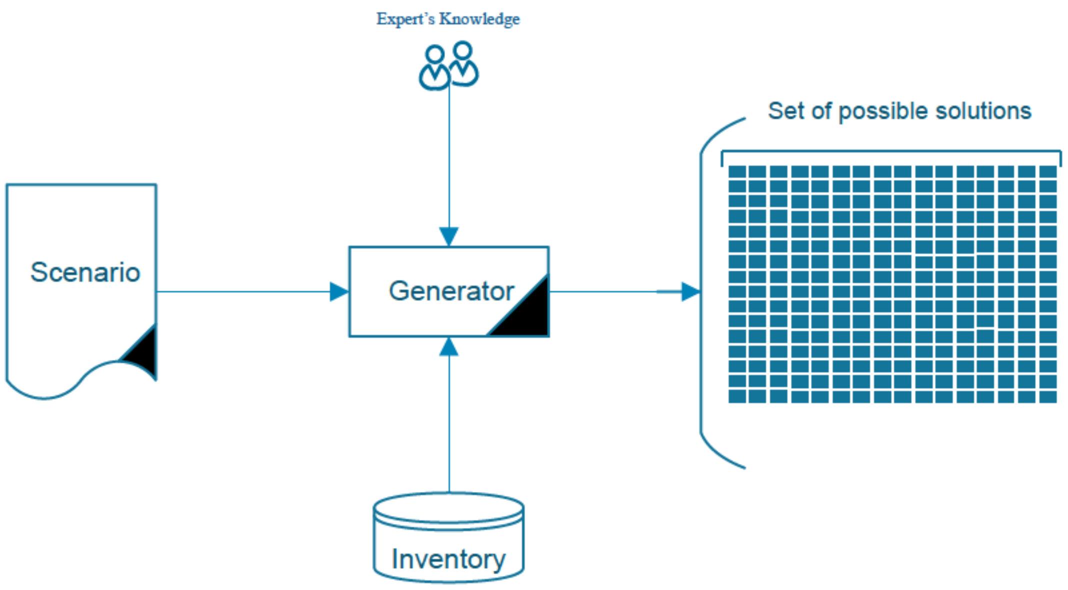


Figure 5.8: Generation of all possible solutions.

Finally, the evaluation step is shown in Figure 5.10. This step is run sequentially, the object of evaluation takes one by one all the feasible solution and their simulation files. The number of selected optimal solutions depends on the user, following a Pareto front.

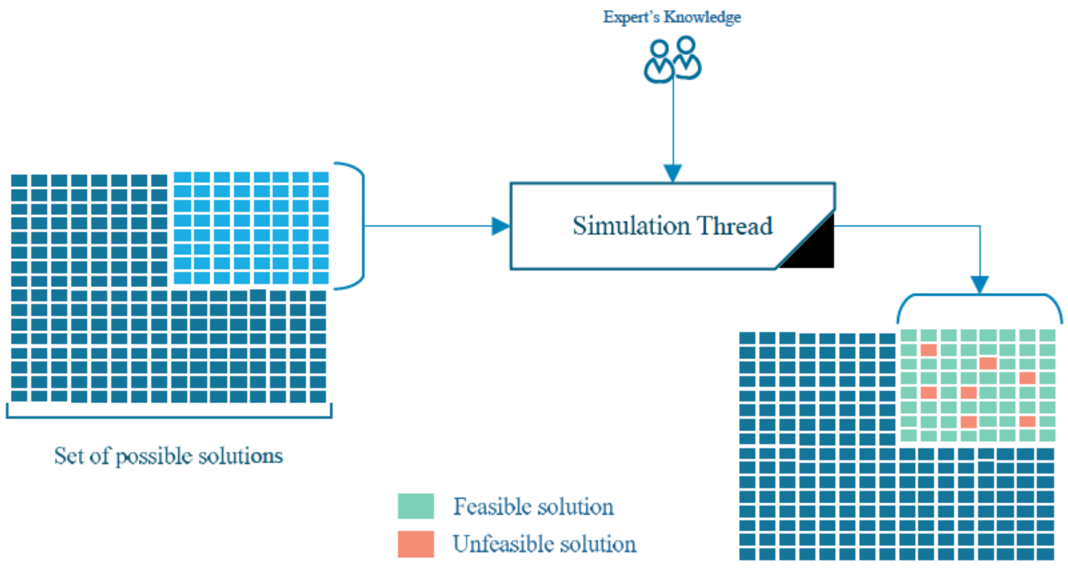


Figure 5.9: Simulation of all possible solutions.

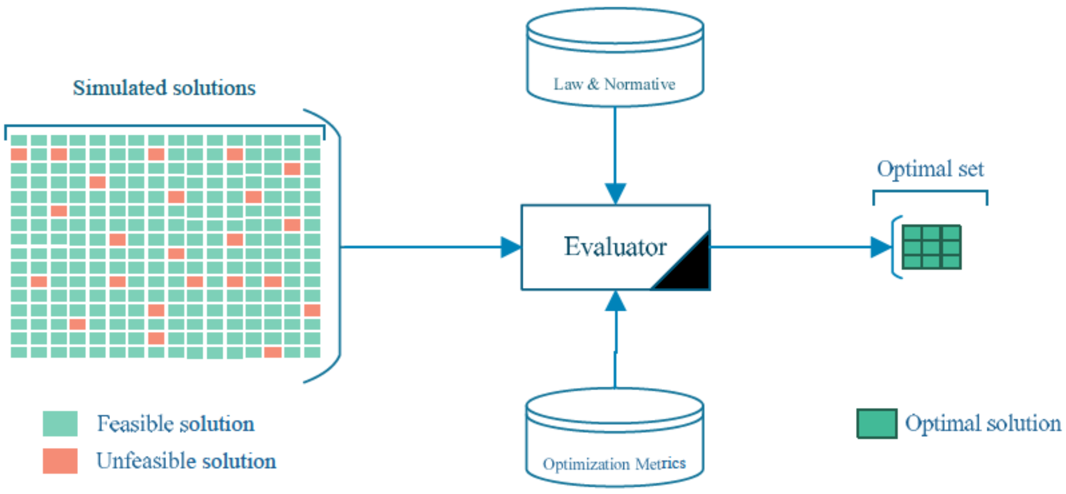


Figure 5.10: Evaluation of all feasible solutions.

5.9 Distributed Memory Implementation

In this section we detail the two extra phases needed to balance the problem between all the MPI processes only. First of all, we must to create an intermediate scenario per MPI process, these scenarios are equals that the initial scenario, but the first parameter has been calculated to generate a part of the complete spectre. The new interval is expressed in Equations 5.16 and 5.17, when n is the number of the MPI process.

$$[Jp_{min} + n\delta, Jp_{min} + (n+1)\delta] \tag{5.16}$$

$$\delta = \frac{(Jp_{max} - Jp_{min})}{MPI_processes} \tag{5.17}$$

Figure 5.11 shows the process of generation and distribution of all the intermediate scenarios. Each MPI process executes an instance of the shared memory simulator with the intermediate scenario as initial scenario a with as many threads as cores. After that, we obtain an optimal set per process and a second phase of evaluation is needed. The final step is shown in Figure 5.12

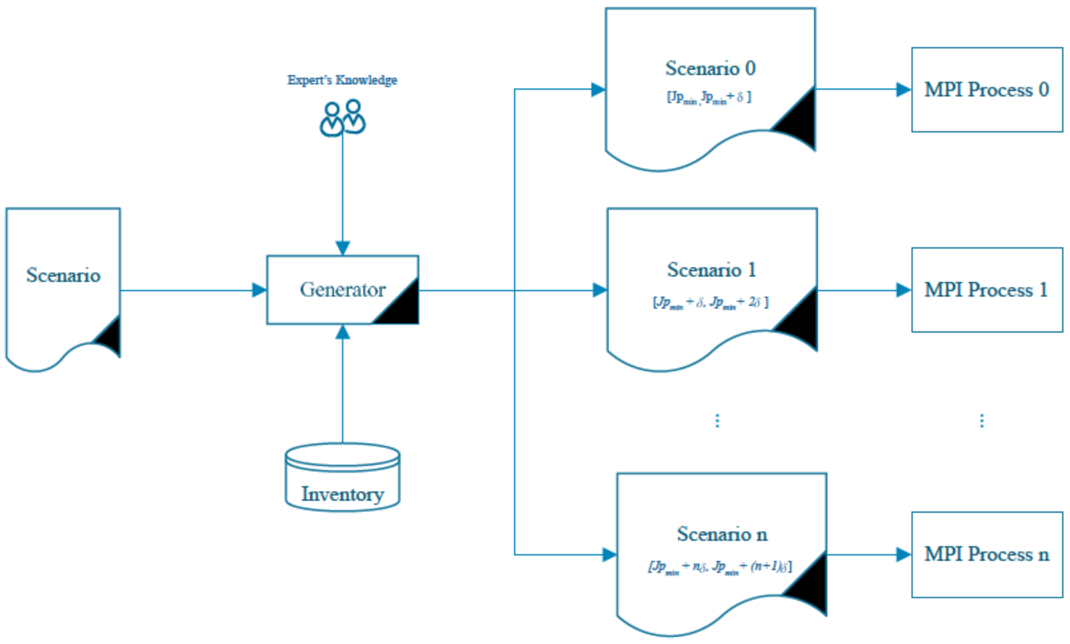

Expert's Knowledge
Scenario
Generator
Inventory
Scenario 0
$[Jp_{min}, Jp_{min} + \delta]$
Scenario 1
$[Jp_{min} + \delta, Jp_{min} + 2\delta]$
Scenario n
$[Jp_{min} + n\delta, Jp_{min} + (n+1)\delta]$
MPI Process 0
MPI Process 1
MPI Process n

Figure 5.11: Generation of all intermediate scenarios.

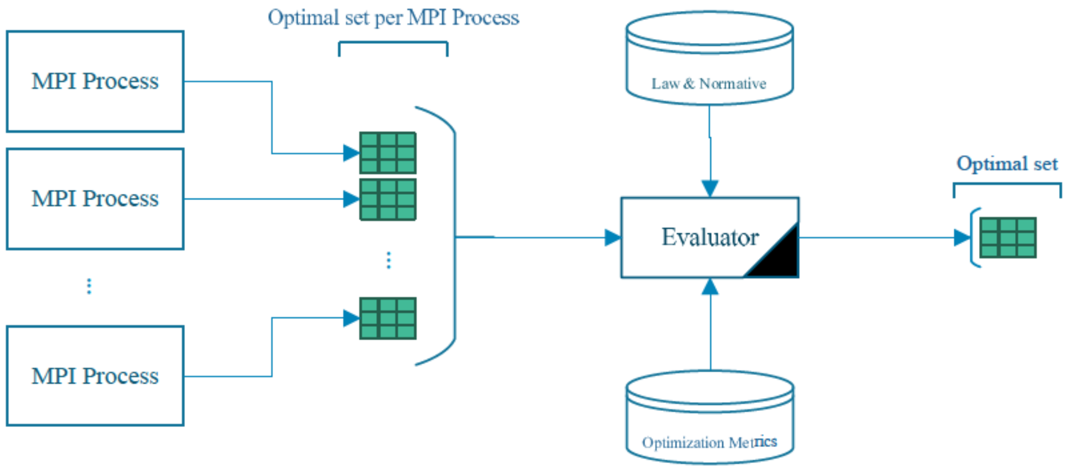


Figure 5.12: Evaluation of optimal sets.

5.10 Summary

In this chapter we have presented a simulator tool that helps railway infrastructure designers with the task of designing and calculating safer and more efficient complex overhead air switches for railway electrification systems, OCLS. This task is a crucial and complex problem since non-optimal overhead air switches designs cause limitations in train speed and, most important, malfunctions and breakages. Those failures are serious because they are very expensive to be repaired and cause long traffic interruptions. The final goal of our simulator is to help designers to find and optimal solution compliant with the existing normative and safe from a circulation and a structural point of view. Dangerous situations and speed limitations that arise currently due to design mistakes, can be avoided. The novelty of this research is the possibility of making automatically overhead air switches designs, choosing optimal solutions, which will enhance the system efficiency and safety. Even when most railway companies have regulations for the design of overhead air switches, as far as we know, there are no computerized software tools to help with the task of designing and testing optimal solutions.

The simulator starts from a model based on the description of the problem that includes all the significant elements that may affect the design process, in order to find an optimal solution in terms of reliability and safety. The obtained designs will be the more reliable to face failures, such as excessive wire and pantograph wears, wrong geometry configurations of the catenary, and electricity supply notches. The simulator also allows to evaluate current designs and to prove their possible flaws.

Chapter 6

Experimental Results

6.1 Introduction

The goal of this chapter is to demonstrate the feasibility of the framework and simulator proposed in this Ph.D. work. We have carried out an experimental evaluation of the simulators proposed in this thesis. The chapter is divided into two sections corresponding to the experimental evaluation of the simulators presented in previous chapters.

The first section presents a complete evaluation of the simulator RDIS described in Chapter 4. First of all, we present a real portal frame and we propose a design to evaluate the framework and the ontology introduced in Chapter 4. Then we evaluate the performance of the simulator taking into acount the performance of a simple portal frame with several hypothesis. Finally, we present the speedup obtained with a different number of portal frames, from 4 to 2,048.

In the case of the simulator OCLS, we present the performance results for both shared and distributed memory platforms. The results are obtained employing five study cases with real data: four non high-speed tangent switch and one high-speed.

6.2 RDIS Evaluation

The evaluation of RDIS is divided into two parts. First, the performance improvement obtained with the different optimizations are measured. On the one hand, we analyze the improvement between sequential and heuristic proposed version. We will select a set of structures from low to high level difficulty to study the effect of complexity in the calculation. We selected an existing railway portal frame in order to calculate the speedup between sequential, parallel, heuristic, and heuristic parallel versions.

Second, we verify the scalability of the proposed algorithm. To achieve this, a massively number of portal frames will be calculated at the same time, as in a real railway project. They must be calculated independently of each other. A single common solution can not be developed because the portal frames are different from each other.

The RDIS simulator version employed has been implemented in C# and compiled with .Net Framework 4.5 and for 64-bit processors. The experiments have been carried out in the following platforms:

Workstation 1 (WS1). Intel Core i5 760 2800 MHz, 4 Cores, 4 Threads, and 24 GB of RAM.

Workstation 2 (WS2). Intel Core i7 920 2660 MHz, 4 Cores, 8 Threads, and 12 GB of RAM.

6.2.1 Evaluation of Enhancements

We have designed a single portal frame to evaluate the performance of RDIS when making multiple calculation of hypothesis for each structure. We have followed the norm EN-50119 [BE09b], where six hypotheses must be evaluated per structure:

- **Minimum Temperature**. It should take into account the permanent loads and the traction forces of the conductor to the minimum temperature and the target room temperature. Two hypothesis, minimun and target temperature.

- **Wind**. Permanent loads, traction forces of the conductor must be increased by the action of wind. Figure 6.1 shows an example of application of the wind loads.

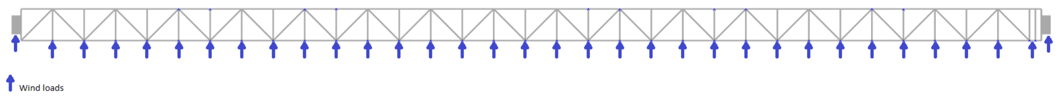


Figure 6.1: Wind loads in the portal frame.

- **Frost**. Permanent loads, traction forces of the conductor must be increased by the action of frost.

- **Wind and Frost**. Permanent loads, traction forces of the conductor must be increased by the action of wind and frost.

- **Maintenance**. It should take into account a force of 1.000kg in the middle of the lintel (see Figure 6.2).

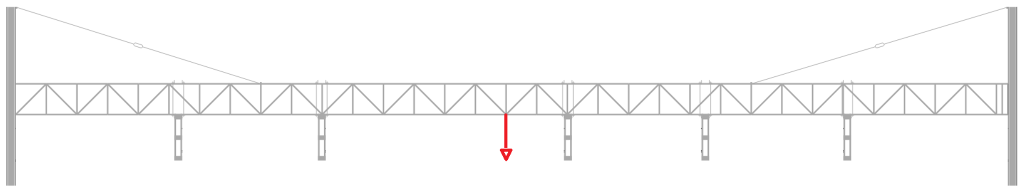


Figure 6.2: Maintenance.

- **Accident**. It should take into account a force equivalent to break a catenary in the worst case (see Figure 6.3).

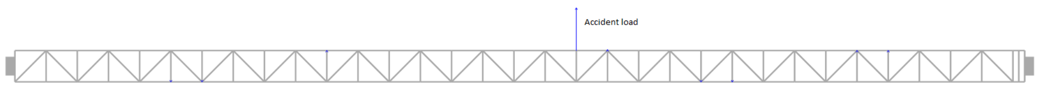


Figure 6.3: Accident.

Figure 6.4 represents all the views of the portal frame design proposed by RDIS. It has seven catenaries and it is 32 meters long, with a PRC lintel and a PG pole.

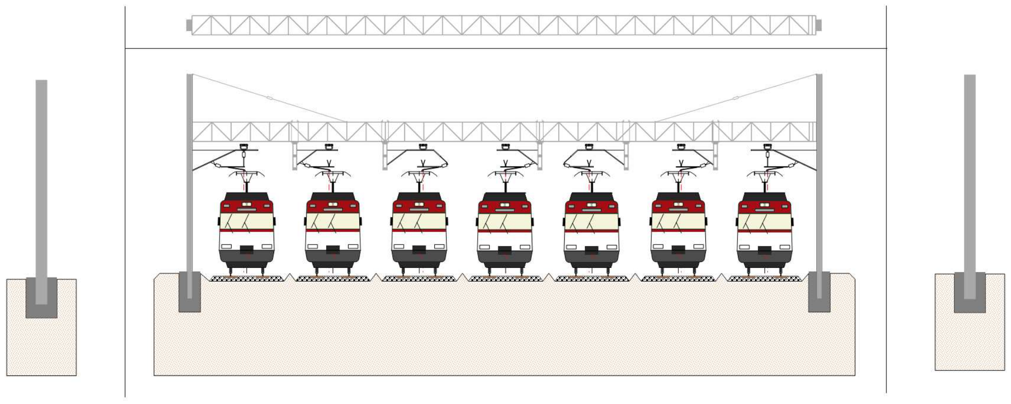


Figure 6.4: Portal Frame for the evaluation.

Figure 6.5 shows the comparison between sequential and sequential heuristic-optimized version of the proposed algorithm. As can be seen, an improvement of 80% is obtained at WS1 and 76% in WS2. The most restrictive hypothesis in the evaluated normative, wind and frost, is computed first as a heuristic rule, thus reducing the total number of calculations by almost 30%. This is because we do not try to calculate the least restrictive hypothesis until the most restrictive hypothesis have been calculated.

As a result, the sequential version carry out 114 calculations, while heuristic approach execute 78 only to design the structure. This improvement is independent of the platform, as we can see in Figure 6.5.

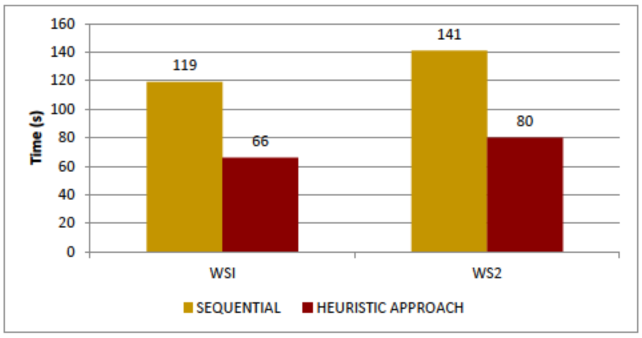


Figure 6.5: Runtimes for sequential and heuristic approach.

Figure 6.6 represents the comparison between sequential and parallel approach, without heuristic optimizations. The experiments have been carried out with 4 threads in Workstation 1, and with 8 threads in Workstation 2. As we can see, we obtain a better improvement than the sequential heuristic approach. In this approach, we have an speedup of 2.33 in Workstation 1 and 2.07 in Workstation 2. Here, we do not employ the heuristic approach but a a brute-force for parallel computation of the hypothesis is applied. We calculate 4 or 8 hypothesis at the same time, respectively.

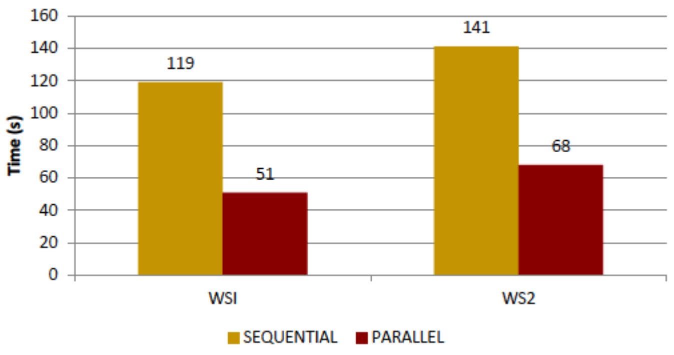


Figure 6.6: Runtimes for sequential and brute-force parallel approach.

A comparison between sequential and heuristic-optimized parallel approach is shown in Figure 6.7. In this case, first the most restrictive hypothesis is computed sequentially and then all the other hypothesis are computed in parallel using the preliminary structure. Blocks of 4 hypotheses are done in the workstation 1, while for workstation 2 these calculations are carried out in blocks of 8 simultaneously. As we can observe, the results are not much better than brute-force parallel approach for

Workstation 1, while performance in Workstation 2 is enhanced because parallelism is higher in Intel I7 processor.

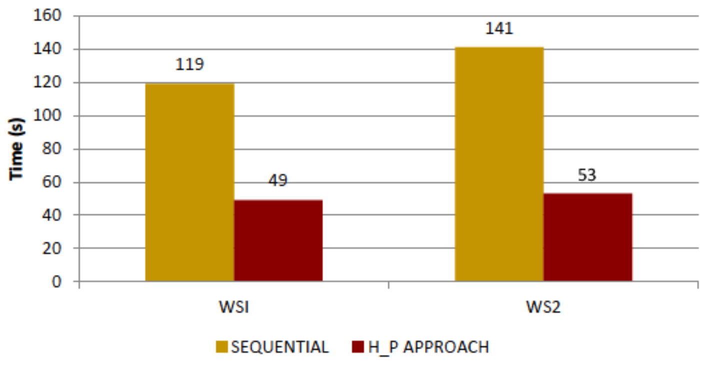


Figure 6.7: Runtimes for sequential and heuristic parallel approach.

To sum-up, Figure 6.9 shows the comparison between all the approaches. As we can see, the heuristic parallel approach is better in both workstations.

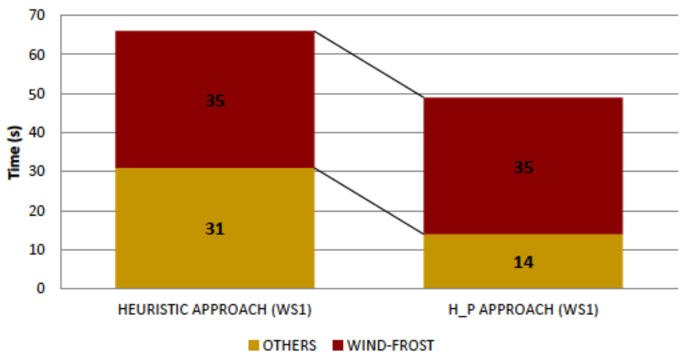


Figure 6.8: Runtimes for heusistic and heuristic parallel approach.

6.2.2 Evaluation of a Project with Many Independent Structures

We consider in thesis section a project with a set of 2 048 heterogeneous railway portal frames. Then, we perform experiments designing and calculating subsets of 64, 128, 512, 1 024 and 2 048 of these structures, and timing the process. Each experiment has been performed five times, collecting statistical values of average, standard deviation, and min-max. It is not relevant how the structures are defined and their catenary requirements, since we just want to measure computational efficiency.

Concerning the computational complexity of the proposed algorithm, it is determined by three different variables: the number of structures to be calculated, the

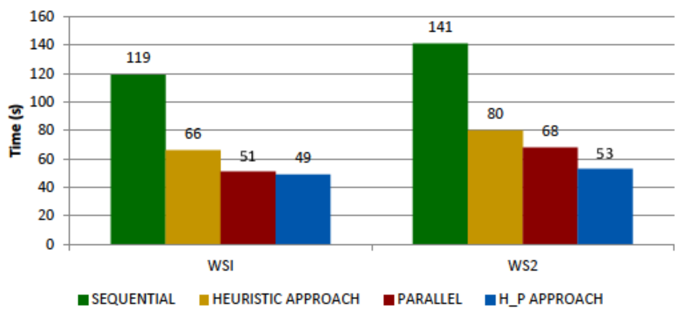


Figure 6.9: Runtimes for sequential, heuristic, parallel, and heuristic parallel approach.

inventory size, and the number of elements (poles, lintels, cantilevers, and foundations) per structure. In order to analyze the complexity of each single variable, the other two ones remain constant. The worst-case complexity of the algorithm is linear ($O(n)$) to the number of structures. Remaining constant the inventory size and the number of elements per structure, the total time in this case is calculated in Equation 6.1:

$$t_{tot} = \|W\| \cdot \left(\left(t_{d_{avg}} \cdot N_{a_i}\right) + \left(t_{c_{avg}} \cdot N_{a_i}\right) + \left(t_{f_{avg}} \cdot N_{f_i}\right)\right) \quad (6.1)$$

where $\|W\|$ is the number of structures, and Equation 6.2 is a constant value.

$$\left(\left(t_{d_{avg}} \cdot N_{a_i}\right) + \left(t_{c_{avg}} \cdot N_{a_i}\right) + \left(t_{f_{avg}} \cdot N_{f_i}\right)\right) \quad (6.2)$$

A linear complexity entails that the algorithm is scalable to the number of structures, and it could be possible to perform more calculations investing a proportional time. We have chosen a maximum of 2 048 structures because a standard rail work project may contain about 1 000 structures. Thus, only in exceptionally large projects we could find more than 2 000 structures and up to 4 000.

The results of the study case are showed in Figure 6.10, Figure 6.11, Table 6.1 and Table 6.2. Figure 6.10 compares sequential and parallel execution times for 4 to 2 048 structures in WS1, while Figure 6.11 shows the same for WS1.

As may be seen in both figures, there is a clear linear tendency in the computation time realted to the problem size. As Equation 4.3 and Equation 4.4 shows, the inventory size $\|C\|$, $\|P\|$, $\|L\|$, and $\|F\|$ is raised to the number of these elements per structure (n_{ic}, n_{ip}, n_{il}). Remaining constant the number of structures $\|W\|$, the worst-case complexity of the algorithm is polynomial ($O(n^a)$) to the inventory size, and exponential ($O(a^n)$) to the number of elements per structure.

Both figures also show that, regarding sequential calculus, WS1 is better that WS2 due to the higher clock frequency of WS1. Whereas workstation 2 has a larger number of cores, workstation 1 has a higher CPU speed, thus investing less time in the sequential calculus of every set of structures.

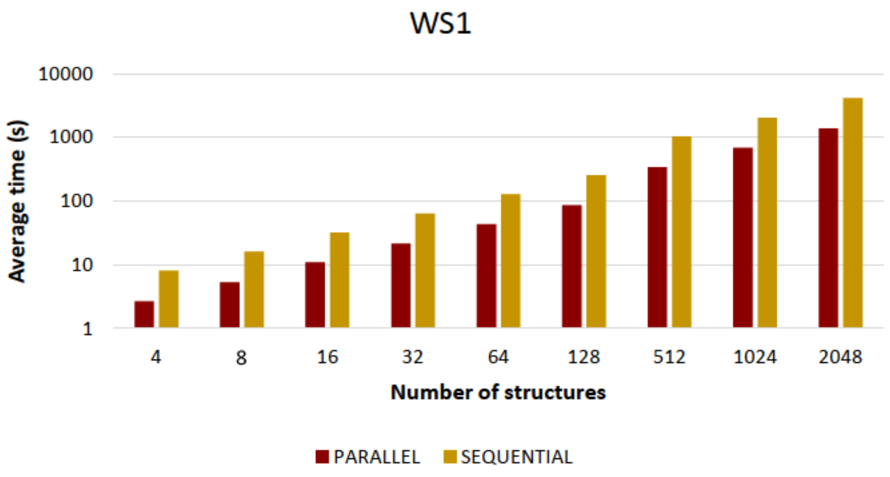


Figure 6.10: WS1: Average calculation times for 4 to 2048 structures.

Table 6.1: WS1: Average, standard deviation, and min-max calculation times for 4 to 2,048 structures.

Workstation 1	Parallel (s)			
Structures	Average	Std. dev.	Max.	min
64	43	0.938	44	41
128	86	0.522	86	85
512	341	1.9	343	339
1,024	681	3.1	685	678
2,048	1.373	1	1,374	1,372
Workstation 1	**Sequential (s)**			
Structures	Average	Std. dev.	Max.	min
64	128	0.1	128	128
128	255	0.12	255	255
512	1,027	1	1,029	1,026
1,024	2,025	2	2,027	2,022
2,048	4,100	3.9	4,107	4,096

They also show that parallel calculus outperforms sequential calculus, as there are not dependencies among the different structures and all the calculus can proceed in parallel, thus exploiting at maximum the number of cores of each CPU to execute several threads that may calculate different structures at the same time. To be precise, 4 and 8 threads can be run concurrently in workstation 1 and 2 respectively. However, surprisingly, the results of WS1 and WS2 are very similar in parallel computing. The reason is that WS2 provides hyperthreading, which is not useful for

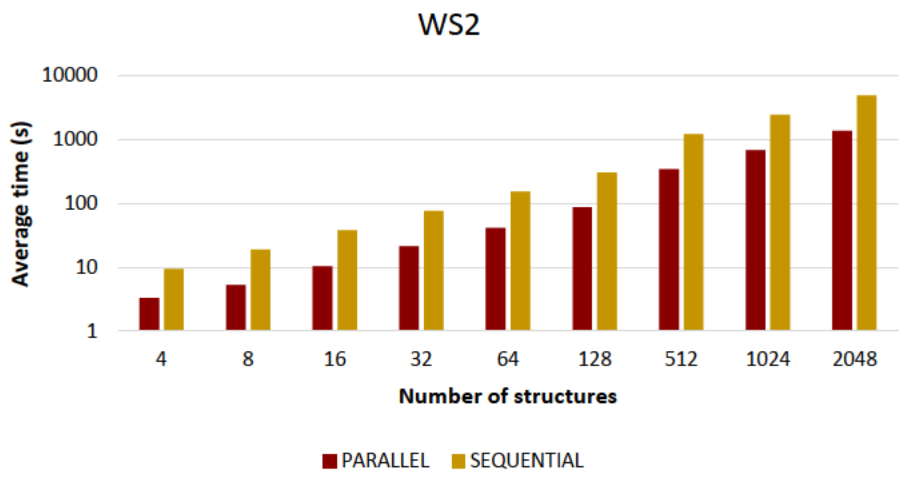


Figure 6.11: WS2: Average calculation times for 4 to 2048 structures.

this kind of calculus that are very CPU-intensive, thus coping the actual physical resources.

Table 6.2: WS2: Average, standard deviation, and min-max calculation times for 4 to 2,048 structures.

Workstation 2	Parallel (s)			
Structures	Average	Std. dev.	Max.	min
64	41	0.5	42	41
128	86	1.5	87	84
512	339	2.3	340	336
1,024	674	4.4	678	669
2,048	1,355	3.7	1,359	1,351
Workstation 2	**Sequential (s)**			
Structures	Average	Std. dev.	Max.	min
64	151	0.3	152	151
128	300	0.4	301	300
512	1,194	1.1	1,195	1,193
1,024	2,419	2.8	2,421	2,416
2,048	4,837	2.1	4,839	4,834

All in all, even considering the less powerful workstation and performing the tasks sequentially, 2,048 portal frames are designed and calculated in less than 5,000 seconds. An average of about 2.4 seconds per single structure is obtained in WS2, while in WS1 we obtain an average of about 1.95 seconds. When using parallel calculus, RDIS can provide a use case portal frame design in 600 seconds, which is an excellent enhancement compared with the manual methods used today that can take several days to design and compute a structure.

Looking at the numbers in detail, Table 6.1 and Table 6.2 show that standard deviation values are quite small as to the average ones, as the coefficient of variation is below $< 1\%$. Even, minimum and maximum values are near to the average and there are no outliers. These values show that the algorithm is stable, that there are no concurrency problems, and that the results are reproducible (which is critical for this kind of tools).

Figure 6.12 shows the speedup got in both workstations. In this experiment the maximum number of structures is computed varying the number of threads from 1 to 16. As may bee seen, WS2 provides a better speedup than WS1, but it is always below 4, which is the actual number of cores of the systems. The main difference between both systems is scalability, which is better in the WS2, due to the Intel I7 processor and the larger RAM memory.

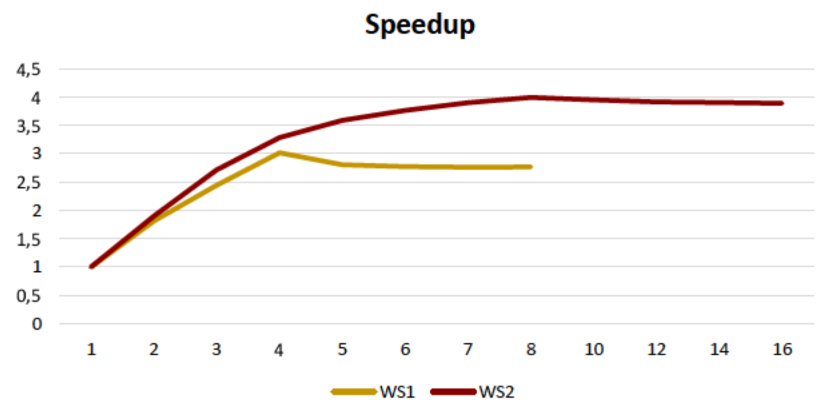


Figure 6.12: Speedup for 1 to 16 threads.

6.3 OCLS Evaluation

In this section we outline the experimental results obtained by running OCLS through several case studies. Each one of them has the input parameters presented in Table 6.3.

- Case studies numbered from 1 to 4, are referred to different types of non high-speed switches. On the one hand, they share the catenary infrastructure geometry data *(ci)*, the catenary installation mechanical features *(cif)* and the simulation conditions data *(sc)*. On the other hand, they differ on the switch infrastructure data *(si)*.

- Case study 5 is oriented to evaluate our algorithm over a high-speed switch, so it has its own input parameters.

Table 6.3: Input parameters and results of the study cases.

Parameters		Study case				
		1	2	3	4	5
si	α	0.11	0.09	0.075	0.09	0.042
	a	13662.92	13435.31	18692.47	22417.16	31468.61
	r	250000	300000	500000	500000	1500000
	b	13662.92	13435.31	18692.47	22417.16	61468.61
	d	6145	0	9165	0	16193
	l	33470	26870.63	46549.95	44834.33	79130.23
ci	El	40				55
	Sl	40				55
	$(Jp_{min}, Jp_{max}, \Delta Jp)$	(80, 100, 5)				(70, 110, 5)
	$(Cp_{min}, Cp_{max}, \Delta Cp)$	(10, 40, 5)				(10, 40, 5)
	$(e_{min}, e_{max}, \Delta e)$	(250, 600, 100)				(250, 600, 100)
	$(h_{min}, h_{max}, \Delta h)$	(10, 50, 10)				(10, 50, 10)
	$(s_{min}, s_{max}, \Delta s)$	(50, 250, 50)				(50, 300, 50)
	$(S_{min}, S_{max}, \Delta S)$	(-250, -50, 50)				(-300, -50, 50)
	H	5300				5300
cif	K_{max}	3.125				2
	K_{min}	1.538				1.66
	D	30				36
	d	5.79				5.79
	N	2				2
	T	1000				1530
	A	153				120
	Ccs	D.C. 3kV				A.C.
	epl	800				800
sc	Ts	60				100
	Ws	0				0
	Wd	-				-

We have executed experiments to analyze two kinds of results: optimal scenario obtained in every study case, and performance analysis for study case 5 in terms of computational efficiency. By way of illustration, the former analysis will be only described in detail for study case 1.

6.3.1 Optimal Switching Point Analysis

Table 6.4 shows the values of one of the optimal designs obtained by the simulator for the case study. These results indicate that the best value of the junction point for these non high-speed railway switches with the given catenary configuration is 80. In Spanish railways junction point 90 is considered to be the best value [MC07], though. Our research in this area concludes that there may be better overhead air switch designs with a lower junction point value. Those, as a conclusion of this thesis, we propose to place the junction point at 80 cm, rather than 90 cm, allowing to maximize the average distance between straight and diverging track contact wires (metric *M1*), thus avoiding potential problems due to electric voltages flowing through the wires.

Characteristic point Cp is another relevant outcome of these simulations. Currently, common Cp values used in Spain are about 25, while our simulator proposes lower values. A lower Cp value reduces the input angle of the diverging track con-

Table 6.4: Optimal switching point solution for the study cases.

Optimal solution	Study case				
	1	2	3	4	5
Jp	80	80	80	80	70
Cp	10	10	15	15	10
e	250	350	250	250	250
h	50	50	50	50	40
s	250	250	250	250	250
S	-150	-100	-50	-50	-50

tact wire on the pantograph along the transition from one catenary to another one, mitigating the impact damage and the stress over pantograph and wires.

Figure 6.13 plots the metric values and the aggregated score of a sampling set of 25 scenarios, including the optimal one, of 17,500 scenarios that were generated for study case 1. These values indicat a strong correlation between metrics *M1* and *M3*, and between metrics *M4* and *M6*. *M1* and *M6* metrics are inversely correlated. This suggests a trade-off between maximizing the distance between wires of both tracks through increasing their staggers, and ensuring the pantograph rubs the wire along its friction surface through decreasing the stagger of the diverging track contact wire.

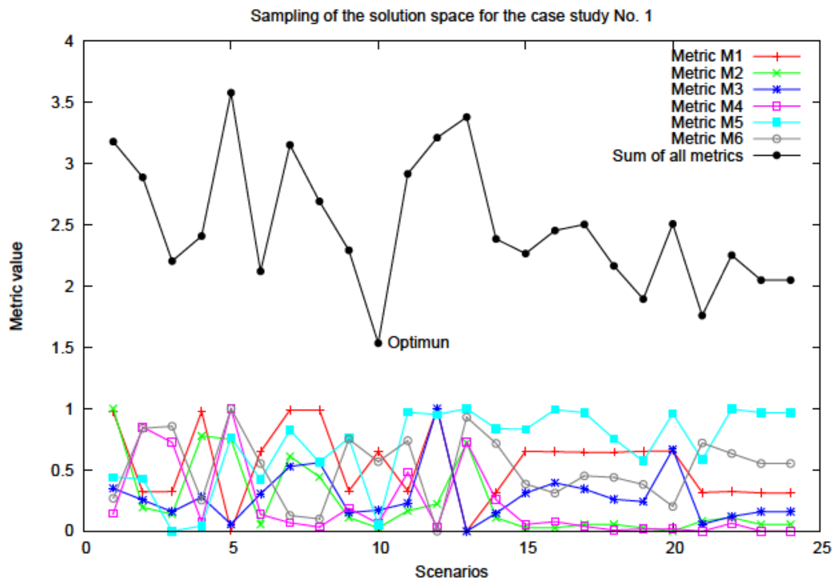


Figure 6.13: Optimal scenario.

Finally, Figures 6.14, 6.15, and 6.16 represent the transition of the pantograph between both catenaries in the optimal scenario obtained for study case 1, when the train is travelling along the diverging track of the switch. Figures 6.14 and 6.15 show elevation and ground plan views respectively, while Figure 6.16 is a three-dimensional view. These figures were made plotting the simulator output data (k_i, st_i, y_i, etc.), which indicate the position of the wires and the pantograph throughout

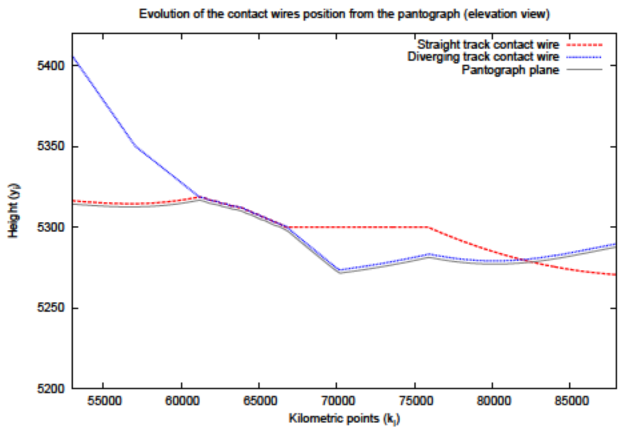


Figure 6.14: Elevation view.

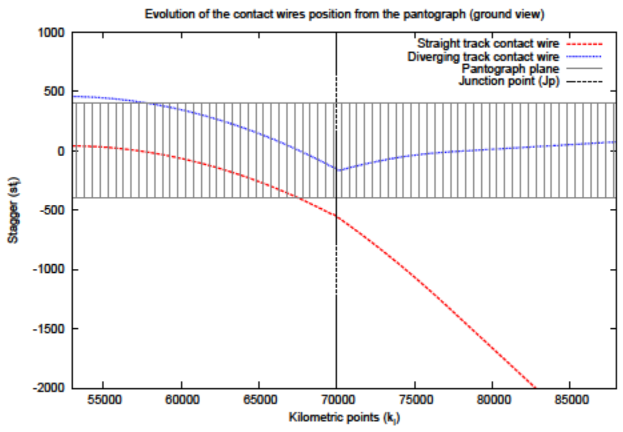


Figure 6.15: Ground plan view.

the simulation. All values are expressed in millimeters. These data allow further analysis of the scenario, including geometric issues or failure detection.

6.3.2 Performance Analysis in Shared Memory Systems

As said before, for performance analysis we focus on the performance measure of the switch described in the case study 1. All the experiments have been carried out simultaneously in the following two Linux workstations:

Workstation 1 (WS1) . Intel Core i5 760 2,800 MHz, 4 cores, and 24 GB of RAM.

Workstation 2 (WS2) . Intel Core i7 920 2,660 MHz, 4 cores with *HyperThreading*, and 12 GB of RAM.

Figure 6.17 presents the speedup for the simulation of the use case. Speedup shows the execution improvement provided by parallel simulation of scenarios when

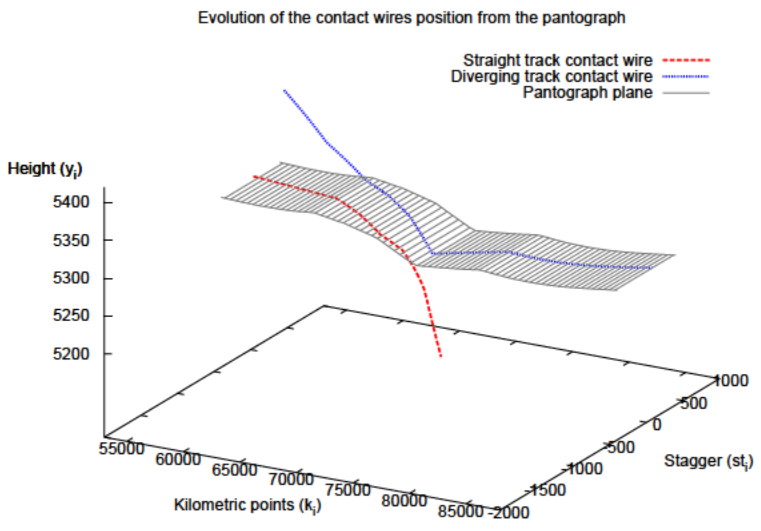


Figure 6.16: Pantograph rubbing against contact wires along a simulation.

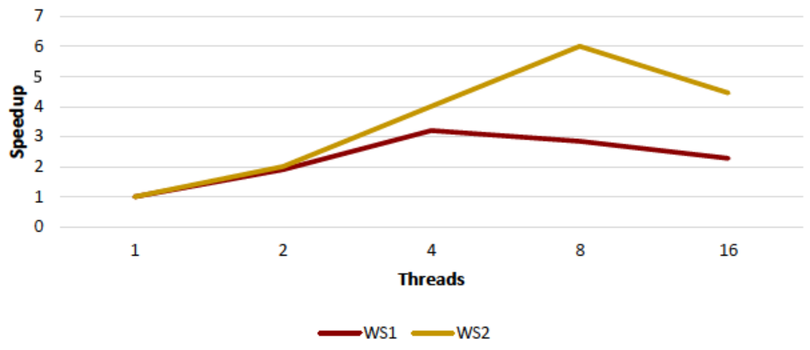


Figure 6.17: OCLS Speedup analysis.

using a multi-core computer. As may be seen, parallel simulation outperforms sequential simulation, as several threads may simulate different switch scenarios at the same time. To be precise, 8 threads can be run concurrently. Moreover, as the number of threads raises, speedup increases until the number of CPU cores is reached. When the number of threads used in simulation process goes beyond the number of cores, speedup starts decreasing, because there are not enough cores for all the running threads.

In Workstation 2 running 8 threads, the simulation spents 233 seconds as may be seen in Figure 6.18. During this time, the simulator has evaluated all the possible combinations of catenary geometry in order to find the optimal one. The number of

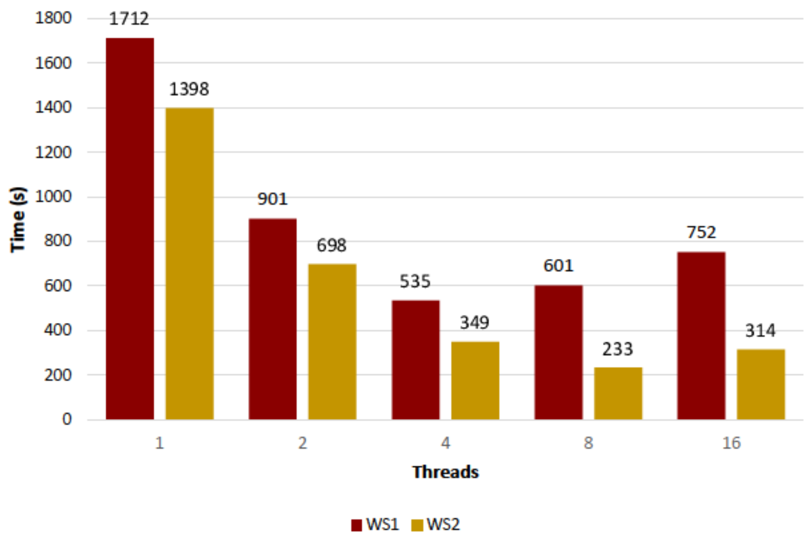


Figure 6.18: OCLS Simulation times.

combinations N can be derived from Equation 5.1. Replacing the variables with the values shown in table 6.3 for study case 5 results, we obtain a total amount of 45,360 scenarios evaluated.

6.3.3 Performance Analysis in Distributed Memory Systems

In this subsection, the performance of the distributed memory prototype of the simulator, based on MPI, is evaluated for the study case 1. To obtain the results, we vary the number of MPI process and the number of threads per MPI process. We have performed our experiments on two different clusters. The main features of the cluster used for evaluations are:

- **Cluster 1**: A cluster with 11 nodes with two Quad Core per node, installed in the labs at University Carlos III of Madrid. Each node has an Intel Xeon CPU E5405 with 8GB of RAM. The file systems installed in cluster 1 are local Linux files systems (EXT4) and remote storage with NFS.
- **Cluster 2**: A cluster with 2 nodes with four Six Core with *Hyperthreading* per node, installed in the labs at University Carlos III of Madrid. Each node has an Intel Xeon CPU E7-4807 with 128GB of RAM. The file systems installed in cluster 2 are local Linux files systems (EXT4) and remote storage with NFS.

Experiments have been divided in two groups per cluster:

1.- **Experiment1**. Executing OCLS with 2 to n MPI process and 1 thread per process, when n is the number of cores.

2.- **Experiment2**. Executing OCLS with 1 MPI process per node and 1 to n threads per process, when n is the number of cores.

Cluster 1

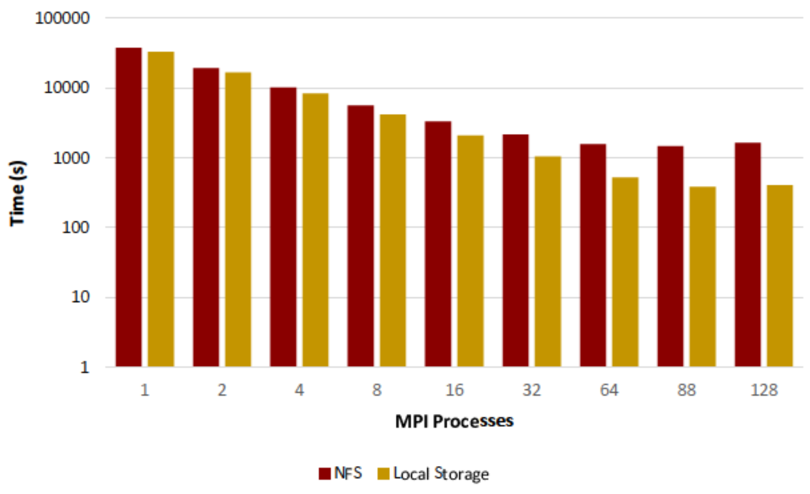


Figure 6.19: C1: Runtime for Cluster 1.

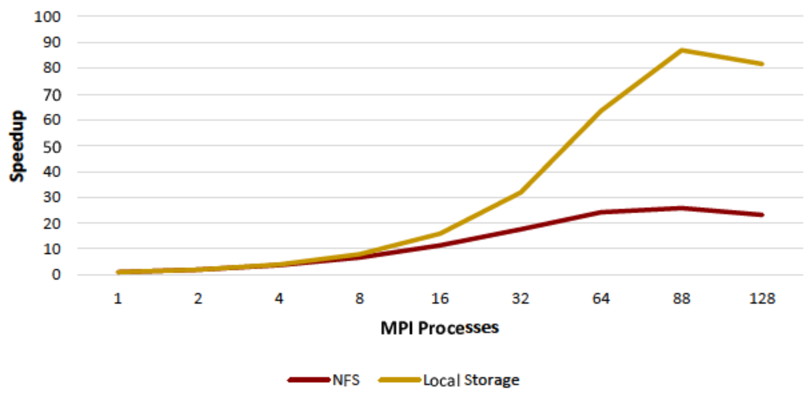


Figure 6.20: C1: Speedup for Cluster 1 for 1 to 128 MPI processes.

The results of the execution times for Experiment1 in the cluster 1 are shown in Figure 6.19. As may been seen the execution time varies from 33,000 seconds for a single process to less than 300 seconds for 128 processes running in parallel and using MPI for communication in the case of local storage. The NFS version is affected by the network, leading to a execution time of almost 1,400 seconds. Figure 6.20 shows

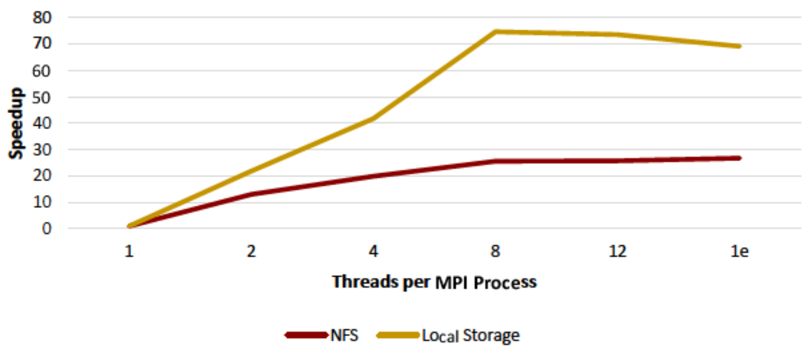


Figure 6.21: Speedup for Cluster 1 with 1 MPI process per node and varying local threads.

the speedup achieved, that is almost 4 times higher in the case of the local storage as compared to NFS when the number of processes increases.

Main reasons for this behavior are stated below:

- The algorithm is I/O intensive. The expected speedup is not obtained in NFS due to the algorithm writes one file per scenario as the results of every simulations are written in different files to be evaluated after. As an example, the study case generates more than 35 GiBytes of storage. Thus I/O bandwidth is critical for performance.

- Our cluster nodes only have one Gibit network interface per node. Thus, the network connection is a clear bottleneck when NFS is used, as the maximum storage bandwidth is limited by the network. This problem increases when more than one MPI process per node is allocated, as all of them have to share the network interface. This is the reason why NFS speedup is so low, compared to the local storage execution.

The results of speedup for Experiment 2 in Cluster 1 are shown in Figure 6.21. In this case, the speedup shows again that speedup grows when using the local file system until we arrive to $11 * 8$ threads, then speedup start to decrease slowly. The reason for that behaviour is that 8 is the optimum number of threads for the CPU in this cluster. After that, we do not have full parallelism, but concurrent execution of some threads that have to fight for a core. In the case of the NFS storage, like in the previous experiment, the network connection is a clear bottleneck. The speedup in this case is more stable from 8 to 16 threads because the network is a barrier that prevent the core saturation.

Cluster 2

The Figure 6.22 shows the execution times for Experiment 1 in the cluster 2. In this cluster, the execution time varies from 32000 seconds for a single process to less than 872 seconds for 48 processes running in parallel and using MPI for communication in the case of local storage. The NFS version is affected by the network like in the cluster 1, leading to a execution time of almost 3300 seconds. Figure 6.23 shows the speedup achieved. We obtain similar behaviour to the experiments in the cluster 1. We obtain better results with 48 processes due to we have 24 cores with *Hyperthreading* per node in cluster 2.

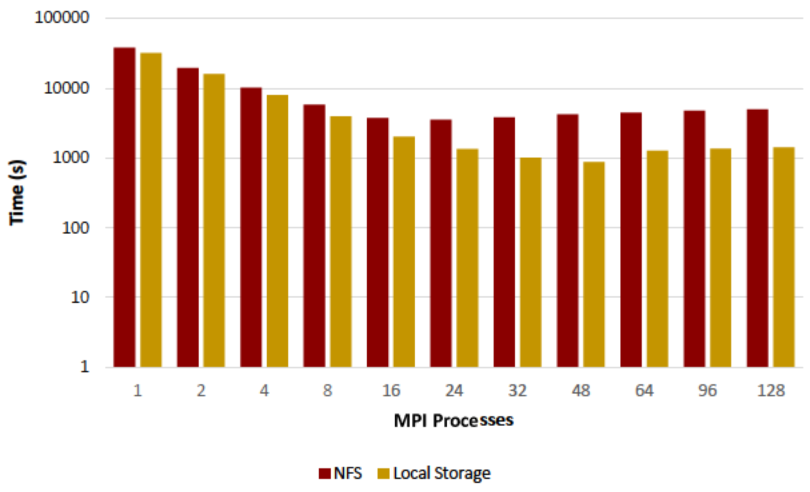


Figure 6.22: C2: Runtime for Cluster 2.

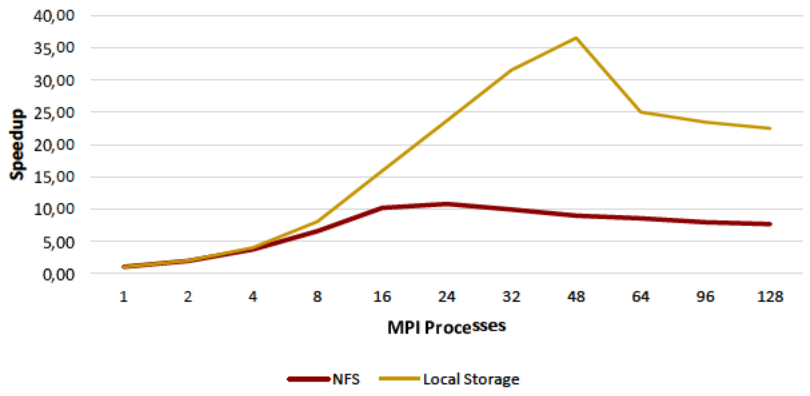


Figure 6.23: C2: Speedup for Cluster 1 for 1 to 128 MPI processes.

The results of speedup for Experiment 2 in Cluster 2 are shown in Figure 6.24. In this case, the speedup shows again that speedup grows when using the local file system until we arrive to $2 * 48$ threads, remember that we have 24 cores with *Hyperthreading* in cluster 2, then speedup start to decrease slowly. We obtain the most increase until 24 threads per MPI processes, the reason for that behaviour is that we have 24 real cores. After that, we use virtual nodes and the speedup increase slowly. In the case of the NFS storage, like in the previous experiment, the network connection is a clear bottleneck. The speedup in this case is more stable from 24 to 64 threads because the network is a barrier that prevent the core saturation.

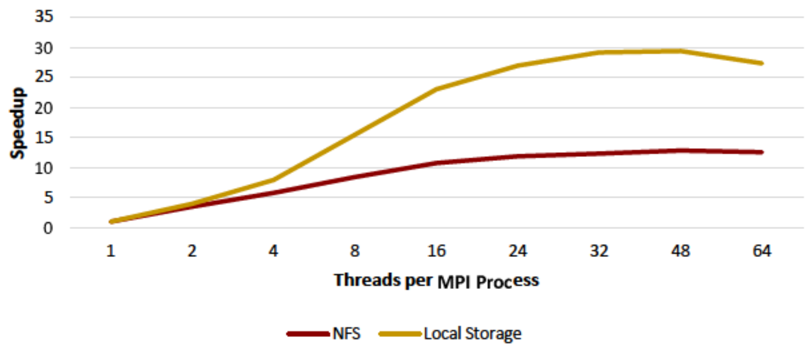


Figure 6.24: Speedup for Cluster 2 with 1 MPI process per node and varying local threads.

6.4 Summary

In this chapter, we have evaluated both simulators proposed in this Ph.D. thesis to assess their feasibility from the point of view of correctness and performance.

The computational performance and complexity of RDIS have been evaluated through an experiment, where up to 2,048 heterogeneous structures are computed in two workstations with different degrees of parallelism. Moreover, statistic values have been collected. Results show that the tool is able to design and calculate a railway catenary support structure in a few seconds.

The feasibility and applicability of OCLS have been tested through regulations and expert judgment, being both results very positive. As a result, the usage of the simulator presented in the Chapter 5 allows to reduce the time and effort of the overhead air switches design process, providing optimal solutions that increase the reliability of the systems. The validation and performance evaluations were made in this chapter through the simulation of case studies based on real switches, which are installed in the Spanish railways. Other input data, like catenary geometry and installation features, are also extracted from European normative and practices. Overhead air switch designs obtained are in harmony with catenary characteristics and specifications currently in use.

All in all, we can conclude that the simulators proposed allow to save time and money in the process design, while ensuring correctness of the solutions in relation to normative, which may result in safer infrastructure and less accidents.

Chapter 7

Conclusions

This chapter contains the summary, major contributions and major conclusion obtained from this thesis work. First, the main contributions of the thesis are reviewed. Second, the thesis results are presented. Finally, we analyze the possible future work of this Ph.D. thesis work.

The thesis has properly fulfilled all the primary objectives indicated in Section 1.2,

O1 *Designing a framework for railway simulations in HPC.* In Chapter 3 we have proposed a generic simulation framework with the aim of enhancing functionality and productivity of simulators in the field of railway infrastructure design. This structure improves the efficiency of the simulators by giving them the ability of searching for the best solutions in the problem space. In Chapter 4 and 5 we described some improvements of RDIS and OCLS in order to exploit the advantages of the HPC environment. Measures and evaluations are conducted in Chapter 6.

O2 *Designing an ontology that facilitate the design based parametric simulations.* In Section 4.2.1 we propose a knowledge representation on the domain of railway. A hybrid approach has been adopted: the railway infrastructure can be captured using ontologies and rule-based representation techniques can be used to solve the process.

O3 *Building the infrastructure simulator.* In Chapter 4 we have described a complete simulator to design and calculate complex railway infrastructure, RDIS. Two levels of parallelism have been introduced, for a multiple hypotheses, and for a several structures. All of them have been implemented in C# and evaluated in Chapter 6.

O4 *Building the overhead air switch simulator.* In Chapter 5 we have introduced an overhead air switch simulator, OCLS. The inputs, output and problem space is described. We apply a shared memory optimization with threads, and a distributed memory using MPI. Both have been implemented in C, and evaluated in Chapter 6.

7.1 Contributions

This thesis makes the following contributions:

C1 Generic Simulator Framework. The framework is focused on four main issues: trade-off between accuracy and complexity, automatic generation and simulation of possible solutions, taking into account other participants in the design process, and integrate expert's domain knowledge and optimization metrics. This structure improves the efficiency of the simulators by giving them the ability of searching for the best solutions in the problem space. Also, obtained solutions will be fully-integrated with the different actors of the design process.

C2 Railway Infrastructure Design Simulator (RDIS) can help to reduce design costs, by decreasing the design time and getting better solutions, by shortening the time for the infrastructure project to be available, and by reducing traffic interruptions due to failures. We have estimated that the design time for a portal could be reduced by two magnitude orders. The track and electrification infrastructure produced with RDIS includes only normalized equipment and strongly reduces the catalog of systems installed, which also generates cost reductions in maintenance and stock for components. As a result, maintenance teams may achieve a shorter answer and repair time to failures and they can be more skilled in the system to maintain.

C3 Overhead Contact Line Switch Simulator (OCLS), which impact may be important in the railway business, as it may allow to railway and engineering companies to make faster and safer designs, avoiding also expenses due to oversized systems or failures caused by erroneous designs.

7.2 Thesis Results

The work made during this Ph.D. Thesis has allowed to produce the following results in the form of publications, grants, participation in projects or derived bachelor thesis:

- **JCR-indexed Journal Articles**

  1.- Rubén Saa, Alberto García, Carlos Gómez, Jesús Carretero, Felix García-Carballeira. An ontology-driven decision support system for high-performance and cost-optimized design of complex railway portal frames. *Expert Systems With Applications.* August 2012. Elsevier 0957-4174. Impact Factor: 2.203 (2012). Q1

  2.- Carlos Gómez, Rubén Saa, Alberto García, Felix García-Carballeira, Jesús Carretero. A model to obtain optimal designs of railway overhead knuckle junctions using simulation. *Simulation Modelling Practice and Theory.* August 2012. Elsevier 1569-190X. Impact Factor: 0.969 (2012). Q2

  3.- Alberto García, Carlos Gómez, Rubén Saa, Felix García-Carballeira, Jesús Carretero. Optimizing the process of designing and calculating railway catenary support infrastructure using a high-productivity computational tool. *Transportation Research Part C: Emerging Technologies.* March 2013. Elsevier 0968-090X. Impact factor 1957 (2013). Q1

- **International Conferences.**

  1.- Rubén Saa, Alberto García, Carlos Gómez, Felix García-Carballeira, Jesús Carretero. A High-productivity Computational Tool to Model and Calculate Railway Catenary Support Structures, in *The 2012 International Conference of Computer Science and Engineering*, London, United Kingdom, July 2012. ISBN 978-988-19251, 2078-0966. Best Student Paper Award.

  2.- Alberto García, Carlos Gómez, Felix García-Carballeira, Jesús Carretero. Enhancing the structure of railway infrastructure simulators, in *OPT-i 2014. The 2014 International conference on Engineering and Applied Sciences Optimization*, Kos, Greece, June 2014.

  3.- Jesús Carretero, Carlos Gómez, Rubén Saa, Alberto García, Felix García-Carballeira. A Holistic approach to railway engineering design using a simulation framework, in *Simultech 2014*, Vienna, Austria, August 2014.

- **Grants**

  – Convocatoria PIF UC3M 01-11112 de Personal Investigador en Formación, PhD fully granted (4 years), 2011, Universidad Carlos III de Madrid

- **Participation in projects**
  - Administrador de Infraestructuras Ferroviarias (ADIF), Estudio y realización de programas de cálculo de pórticos rígidos de catenaria y de sistema de simulación de montaje de agujas aéreas. JM/RS 3.6/4100.0685-9/00100
  - Administrador de Infraestructuras Ferroviarias (ADIF), Proyecto para la Investigación sobre la aplicación de las TIC a la innovación de las diferentes infraestructuras correspondientes a las instalaciones de electrificación y suministro de energía. JM/RS 3.9/1500.0009/0-00000
  - Telice S.A., Colaboración en el diseño, desarrollo, modelización y validación de una herramienta para cálculo de catenaria.
  - Spanish Ministry of Education, Scalable Input/Output techniques for high-performance distributed and parallel computing environments, TIN2010-16497
  - Spanish Ministry of Economics and Competitiveness, Téecnicas de gestión escalable de datos para high-end computing systems, TIN2013-41350-P
  - European Union, COST Action IC1305, "Network for Sustainable Ultrascale Computing Platforms" (NESUS)
  - Spanish Ministry of Science and Tecnology, Computación de Altas Prestaciones sobre plataformas heterogéneas, TIN2014-53522-REDT

7.3 Future Work

The work described in this Ph.D. Thesis could be further extended in many different aspects, related both to the computing capabilities and the simulation features of the tool provided. Below, some or then are pointed out:

- Moving both simulators to the cloud to offer it as a service. Offering simulation services through the Web could be an important point to increase the impact of the solutions, spcially for small and medium enterprise that cannot afford their own developments. It would also make maintenance and coherence easier than now. To achieve that, both simulators should be moved to the cloud to provide the scalability needed to cope with the possible demand.
- There are still some enhancements that could be included in RDIS. The major challenge here is to obtain a better performance through strong parallelism. We have already achieved a good performance by adopting a coarse-grained

parallelism approach, i.e., a thread is in charge of resolving one structure. Implementing the tool as a multi-process distributed application (instead of a multi-core application), could enhance the system performance, allowing its execution in high-performance distributed computers. By this way, the number of structures being calculated concurrently may be increased several orders of magnitude: from 8 (common number of CPU cores in a computer) to hundreds or thousands (common number of CPU cores in a distributed supercomputer). Our work will also be focused on studying the use of new techniques of seeking for possible design assemblies, like the bisection method. The aim is to achieve a better performance in the design task. Another guideline could be to include different search criteria instead of the minimum weight design criteria currently used. Another future work will be focused on extending the railway design model to other railway components, like tracks and power.

- Separating the structural calculus inside the tool of the .NET virtual machine by implementing the algorithms in C / C ++ libraries. It should alleviate to reduce the maemory management problem in the .NET framework, which is not optimum currently and does not allow control by the application.

- OCLS is already a functional tool, but there are still many enhancements and functionalities that we would like to include. Using bio-inspired genetic algorithms to compute the optimal of the different metrics is our next goal. The major challenge here is to find a fitness function to evaluate the set of feasible solutions. Another field for future work is to provide a better visualization for the results of the simulator to allow the users to study the many parameters of the solution to make their own elections. One shortcoming of our work is that each railway company or authority has different infrastructure information, catalog, and regulations. Thus, another future work would be to adapt our simulator following the European initiatives for railway interoperability.

- One shortcoming of our work is the difficulty to get infrastructure information from the railway companies to feed databases. Usually they have their own catalog and inventory, and they are not very collaborative among companies. The European level initiative for railway interoperability could be an important step in this address. Achieving the collaboration of manufacturers of railway components could be another one. More effort will be devoted to both groups in a near future.

Bibliography

[ABA$^+$08] Angelines Alberto, JesÃČÂžs Benet, Enrique Arias, David Cebrian, TomÃČÂąs Rojo, and Fernando Cuartero. A high performance tool for the simulation of the dynamic pantographÃćâĆňâĂIJcatenary interaction. *Mathematics and Computers in Simulation*, 79(3):652 – 667, 2008.

[ABC$^+$10] Steve Ashby, P Beckman, J Chen, P Colella, B Collins, D Crawford, J Dongarra, D Kothe, R Lusk, P Messina, et al. The opportunities and challenges of exascale computing. *Summary Report of the Advanced Scientific Computing Advisory Committee (ASCAC) Subcommittee (November 2010)*, 2010.

[Abr]

[Ade03] H. Adeli. *Expert Systems in Construction and Structural Engineering.* Taylor & Francis, 2003.

[AEN03] AENOR. *EN 50318:2003. Railway applications. Current collection systems. Validation of simulation of the dynamic interaction between pantograph and overhead contact line.* AENOR, 2003.

[AEN09] AENOR. *EN 50119:2009. Railway applications. Fixed installations. Electric traction overhead contact lines.* Number ISBN: 978 0 580 56127 6. AENOR, 2009.

[AFG$^+$10] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[Ake11] J Akerman. The role of high-speed rail in mitigating climate change - the swedish case europabanan from a life cycle perspective. *Transportation Research Part D*, 16(3):208 – 217, 2011.

[Bae09] Chang Han Bae. A simulation study of installation locations and capacity of regenerative absorption inverters in {DC} 1500v electric railways system. *Simulation Modelling Practice and Theory*, 17(5):829 – 838, 2009.

[BB99] Mark Bakery and Rajkumar Buyyaz. Cluster computing at a glance. *High Performance Cluster Computing: Architectures and Systems*, 1:3–47, 1999.

[BBC$^+$08] Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Kerry Hill, Jon Hiller, et al. Exascale computing study: Technology challenges in achieving exascale systems. *Defense Advanced Research Projects Agency Information Processing Techniques Office (DARPA IPTO), Tech. Rep*, 15, 2008.

[BBLS00] Aldo Balestrino, Ottorino Bruno, Alberto Landi, and Luca Sani. Innovative solutions for overhead catenary-pantograph system: wire actuated control and observed contact force. *Vehicle System Dynamics*, 33(2):69–89, 2000.

[BCF$^+$98] Sharon Brunett, Karl Czajkowski, Steven Fitzgerald, Ian T. Foster, Andrew E. Johnson, Carl Kesselman, Jason Leigh, and Steven Tuecke. Application experiences with the globus toolkit. In *Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, HPDC '98, Chicago, Illinois, USA, July 28-31, 1998.*, pages 81–88, 1998.

[BCM$^+$98] D. Brunner, G. Cross, C. McGhee, J. Levis, and D. Whitney. Toward increased use of simulation in transportation. In *Simulation Conference Proceedings, 1998. Winter*, volume 2, pages 1169–1175 vol.2, 1998.

[BE09a] BS-EN-15273. Bs en 15273-3:2011 railway applications - gauges - part 3: Structure gauges. Technical report, -, 2009.

[BE09b] BS-EN-50119. Bs en 50119:2009 railway applications. fixed installations. electric traction overhead contact lines. Technical report, -, 2009.

[BJB$^+$00] Judy I. Beiriger, Wilbur R. Johnson, Hugh P. Bivens, Steven L. Humphreys, and Ronald Rhea. Constructing the ASCI computational grid. In *Proceedings of the Ninth IEEE International Symposium on High Performance Distributed Computing, HPDC'00, Pittsburgh, Pennsylvania, USA, August 1-4, 2000.*, pages 193–200, 2000.

[Boa13] OpenMP Architecture Review Board. Openmp application program interface version 4.0, Jul. 2013.

[Bro27] H. Brown. Catenary design for overhead contact systems. *Transactions of the American Institute of Electrical Engineers*, XLVI:1082–1107, January 1927.

[But97] David R Butenhof. *Programming with POSIX threads.* Addison-Wesley Professional, 1997.

[BWTWc13] Subramaniam Balaji, Saunders Winston, Scogland Tom, and Feng Wuchun. Trends in energy-efficient computing: A perspective from the green500. In *Green Computing Conference (IGCC), 2013 International*, pages 1–8. IEEE, 2013.

[BYV08] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on*, pages 5–13. Ieee, 2008.

[Cay13] Erdal Cayirci. Modeling and simulation as a cloud service: a survey. In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, pages 389–400. IEEE Press, 2013.

[CDR08] J. Campos and G. De Rus. Some stylized facts about high-speed rail: A review of hsr experiences around the world. *Transport Policy.*, 16(1):19 – 28, 2008.

[Cel03] J.T. Celigueta. *Curso de Análisis Estructural.* 2003.

[Cle72] C.J. Clemow. Planning for railway electrification. *Electrical Engineers, Proceedings of the Institution of*, 119(4):431 –440, april 1972.

[Con12] PROKON Software Consultants. *PROKON Structural Analysis and Design.* 2012.

[Cou43] R. Courant. Variational methods for the solution of problems of equilibrium and vibration. *Bull. Amer. Math. Soc.*, 49:1–23, 1943.

[Cou13] Rachel Courtland. The status of moore's law: It's complicated. *IEEE Spectrum, Oct*, 2013.

[CPGC$^+$03] Jesús Carretero, José M. Pérez, Félix García-Carballeira, Alejandro Calderón, Javier Fernández, Jose D. García, Antonio Lozano, Luis Cardona, Norberto Cotaina, and Pierre Prete. Applying rcm in large scale systems: a case study with railway networks. *Reliability Engineering &amp; System Safety*, 82(3):257 – 273, 2003.

[CW53] O.J. Crompton and G.A. Wallace. Economic aspects of overhead equipment for d.c. railway electrification. *Proceedings of the IEE - Part I: General*, 100(124):133 –145, july 1953.

[CYP10] CYPE. Cypecad: Structural calculus tool. CYPE Ingenieros, 2010.

[DBV09] H. De Bruijn and W. Veeneman. Decision-making for light rail. *Transportation Research Part A.*, 43:349 – 359, 2009.

[DFZ+15] Yucong Duan, Guohua Fu, Nianjun Zhou, Xiaobing Sun, Nanjangud C Narendra, and Bo Hu. Everything as a service (xaas) on the cloud: origins, current and future trends. In *2015 IEEE 8th International Conference on Cloud Computing*, pages 621–628. IEEE, 2015.

[DI93] G. De Rus and V. Inglada. Análisis coste-beneficio del tren de alta velocidad en espana. *Revista de Economía Aplicada*, 1:27–48, 1993.

[DL11] Jack Dongarra and Piotr Luszczek. Linpack benchmark. *Encyclopedia of Parallel Computing*, pages 1033–1036, 2011.

[DMS+97] Jack J Dongarra, Hans W Meuer, Erich Strohmaier, et al. Top500 supercomputer sites. *Supercomputer*, 13:89–111, 1997.

[Dut96] A. Dutta. Integrating ai and optimization for decision support: A survey. *Decision Support Systems*, 18(3):217–226, 1996.

[Esc03] H.A. Eschenauer. Optimization procedures in structural design. *Applied Mechanics Reviews*, 56(4):19–24, 2003.

[ETI08] ETI. C(2008) 807 - especificacion técnica de interoperabilidad del subsistema de energía del sistema ferroviario transeuropeo de alta velocidad. Technical report, -, 2008.

[EZL89] Derek L Eager, John Zahorjan, and Edward D Lazowska. Speedup versus efficiency in parallel systems. *IEEE Transactions on Computers*, 38(3):408–423, 1989.

[FC07] Wu-chun Feng and Kirk W Cameron. The green500 list: Encouraging sustainable supercomputing. *Computer*, 40(12):50–55, 2007.

[FKNT02] Ian T. Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. Grid services for distributed system integration. *IEEE Computer*, 35(6):37–46, 2002.

[FKT01] Ian T. Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *IJHPCA*, 15(3):200–222, 2001.

[FL05] Vincent W. Freeh and David K. Lowenthal. Using multiple energy gears in mpi programs on a power-scalable cluster. In *Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '05, pages 164–173, New York, NY, USA, 2005. ACM.

[For12] Message Passing Interface Forum. Mpi: A message-passing interface standard version 3.0, Sep. 2012. Chapter author for Collective Communication, Process Topologies, and One Sided Communications.

[Fos04] Ian T. Foster. The grid2003 production grid: Principles and practice. In *13th International Symposium on High-Performance Distributed Computing (HPDC-13 2004), 4-6 June 2004, Honolulu, Hawaii, USA*, pages 236–245, 2004.

[FQKYS04] Zhe Fan, Feng Qiu, Arie Kaufman, and Suzanne Yoakum-Stover. Gpu cluster for high performance computing. In *Proceedings of the 2004 ACM/IEEE Conference on Supercomputing*, SC '04, pages 47–, Washington, DC, USA, 2004. IEEE Computer Society.

[FSSC11] I. Farmaga, P. Shmigelskyi, P. Spiewak, and L. Ciupinski. Evaluation of computational complexity of finite element analysis. In *CAD Systems in Microelectronics (CADSM), 2011 11th International Conference The Experience of Designing and Application of*, pages 213 –214, feb. 2011.

[GFB$^+$04] Edgar Gabriel, Graham E. Fagg, George Bosilca, Thara Angskun, Jack J. Dongarra, Jeffrey M. Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, Ralph H. Castain, David J. Daniel, Richard L. Graham, and Timothy S. Woodall. Open MPI: Goals, concept, and design of a next generation MPI implementation. In *Proceedings, 11th European PVM/MPI Users' Group Meeting*, pages 97–104, Budapest, Hungary, September 2004.

[GHS05] H. Ghenniwa, M. Huhns, and W. Shen. emarketplaces for enterprise and cross enterprise integration. *Data and Knowledge Engineering*, 52(1):33–59, 2005.

[GL96a] William Gropp and Ewing Lusk. Sowing mpich: A case study in the dissemination of a portable environment for parallel scientific computing. *IJSA*, 11:11–2, 1996.

[GL96b] William Gropp and Ewing Lusk. User's guide for mpich, a portable implementation of mpi version 1.2.1, 1996.

[GL09] Al Geist and Robert Lucas. Major computer science challenges at exascale. *International Journal of High Performance Computing Applications*, 23(4):427–436, 2009.

[Gre14] Green500 Organization. Green500 list, November 2014.

[Gri56] J.W. Grieve. Electric traction. a review of progress. *Proceedings of the IEE - Part A: Power Engineering*, 103(9):229 –238, june 1956.

[Gru93] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. Technical report., Stanford Knowledge Systems Laboratory, 1993.

[GSH98] C.J. Goodman, L. K. Siu, and T.K. Ho. A review of simulation models for railway systems. In *Developments in Mass Transit Systems, 1998. International Conference on (Conf. Publ. No. 453)*, pages 80–85, 1998.

[GYO$^+$05] Warodom Geamsakul, Tetsuya Yoshida, Kouzou Ohara, Hiroshi Motoda, Hideto Yokoi, and Katsuhiko Takabayashi. Constructing a decision tree for graph-structured data and its applications. *Fundamenta Informaticae*, 66(1/2):131–160, 2005.

[Har08] D. Hartland. Electric contact systems - passing power to the trains. In *Electric Traction Systems, 2008 IET Professional Development course on*, pages 25 –33, nov. 2008.

[Hem10] Scott Hemmert. Green hpc: From nice to necessity. *Computing in Science Engineering*, 12(6):8–10, Nov 2010.

[HM93] Maurice Herlihy and J Eliot B Moss. *Transactional memory: Architectural support for lock-free data structures*, volume 21. ACM, 1993.

[Hol86] A. Holgate. *The art in structural design.* Oxford University Press., Oxftrod, UK, 1986.

[IEC09] IEC. *IEC60146-1, Semiconductor Converters, General Requirements and Line Commutated Converters. Part 1-1. Specifications of Basic Requirements.* Number ISBN: 978 0 580 56127 6. IEC, 2009.

[JGN99] William E. Johnston, Dennis Gannon, and Bill Nitzberg. Grids as production computing environments: The engineering aspects of nasa's information power grid. In *Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing, HPDC'99, Redondo Beach, California, USA, August 3-6, 1999.*, pages 197–204, 1999.

[Joh02] Johnson et al. A decision-tree-based symbolic rule induction system for text categorization. *IBM Systems Journal*, 41(3):428–437, 2002.

[JOPLGC06] J. R. Jimenez-Octavio, E. Pilo, O. Lopez-Garcia, and A. Carnicero. Coupled electromechanical cost optimization of high speed railway overheads. *ASME Conference Proceedings*, 2006(42037):231–240, 2006.

[KC07] S. Kendal and M. Creen. *An Introduction to Knowledge Engineering.* Springer Verlag Ltd., London, 2007.

[KCP$^+$07] Jin Woo Kim, Ho Chol Chae, Bum Seok Park, Seung Yeol Lee, Chang Soo Han, and Jin Hee Jang. State sensitivity analysis of the pantograph system for a high-speed rail vehicle considering span length and static uplift force. *Journal of sound and vibration*, 303(3):405–427, 2007.

[KES$^+$09] Volodymyr V Kindratenko, Jeremy J Enos, Guochun Shi, Michael T Showerman, Galen W Arnold, John E Stone, James C Phillips, and Wen-mei Hwu. Gpu clusters for high-performance computing. In *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, pages 1–8. IEEE, 2009.

[KK90] S. Kanagasundaram and B.L. Karihaloo. Optimum design of frames under multiple loads. *Computers &amp; Structures*, 36(3):443 – 489, 1990.

[KPSS09] Friedrich Kiessling, Rainer Puschmann, Axel Schmieder, and Egid Schneider. *Contact Lines for Electric Railways. Planning, Design, Implementation, Maintenance.* 2009.

[Kul94] Jasna Kuljis. User interfaces and discrete event simulation models. *Simulation Practice and Theory*, 1(5):207 – 221, 1994.

[LBIC13] Pablo Llopis, Javier Garcia Blas, Florin Isaila, and Jesus Carretero. Survey of energy-efficient and power-proportional storage systems. *The Computer Journal*, 2013.

[LKK91] Averill M Law, W David Kelton, and W David Kelton. *Simulation modeling and analysis*, volume 2. McGraw-Hill New York, 1991.

[LZJ15] Kai Liu, Deqing Zou, and Hai Jin. Uaas: Software update as a service for the iaas cloud. In *2015 IEEE International Conference on Services Computing, SCC 2015, New York City, NY, USA, June 27 - July 2, 2015*, pages 483–490, 2015.

[MA11] A. Mohan and G. Arumugan. Constructing railway ontology using web ontology languages and semantic web rule language. *Int. J. Comp. Tech. Appl.*, 2(2):314–321, 2011.

[MC07] J. Montesinos and M. Carmona. *Tecnologia de Catenaria.* ADIF, 2007.

[MG10] Peter Mell and Tim Grance. The nist definition of cloud computing. *Communications of the ACM*, 53(6):50, 2010.

[MM99] M. Mahendran and C. Moor. Three-dimensional modeling of steel portal frame buildings. *Journal of Structural Engineering*, 125(8):870–878, August 1999.

[MT08] S. Midya and R. Thottappillil. An overview of electromagnetic compatibility challenges in european rail traffic management system. *Transportation Research Part C.*, 16:515 – 534, 2008.

[MWBA10] Richard C Murphy, Kyle B Wheeler, Brian W Barrett, and James A Ang. Introducing the graph 500. *Cray UserâĂŹs Group (CUG)*, 2010.

[NEGED10] M. Nora, A. El-Gohary, and T. El-Diraby. Dynamic knowledge-based process integration portal for collaborative construction. *Journal of Construction Engineering and Management*, 136(3):316–328, March 2010.

[NHAR13] M. Nejlaoui, A. Houidi, Z. Affi, and L. Romdhane. Multiobjective robust design optimization of rail vehicle moving in short radius curved tracks based on the safety and comfort criteria. *Simulation Modelling Practice and Theory*, 30(0):21 – 34, 2013.

[NHC13] A. Naweed, G.R.J. Hockey, and S.D. Clarke. Designing simulator tools for rail research: The case study of a train driving microworld. *Applied Ergonomics*, 44(3):445 – 454, 2013.

[NM01] N. Noy and D. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical Report Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory, March 2001.

[NVI11] NVIDIA Corporation. *NVIDIA CUDA C Programming Guide*, June 2011.

[Ode91] J.T. Oden. Finite elements: An introduction. In *Finite Element Methods (Part 1)*, volume 2 of *Handbook of Numerical Analysis*, pages 3 – 15. Elsevier, 1991.

[oEE] Institute of Electrical and Electronics Engineers. Ieee standard for information technology-portable operating system interface (posix): approved september 15, 1993: Ieee standards board; approved april 14, 1994: American national standards institute. Inst. of Electrical and Electronics Engineers.

[Ope11] OpenACC Working Group. The openacc application programming interface, 2011.

[PM08] H. Panetto and A. Molina. Enterprise integration and interoperability in manufacturing systems: Trends and issues. *Computers in Industry.*, 59:641 – 646, 2008.

[QJ10] X. Quinghua and T. Jing. Research on extracting rule for structure optimal design of mechanical products based on date mining and knowledge discovery. In *2010 International Conference on E-Product E-Service and E-Entertainment (ICEEE)*, pages 1–5, 2010.

[RC03] S. Rocco and M. Claudio. A rule induction approach to improve monte carlo system reliability assessment. *Reliability Engineering and System Safety*, 82(1):85–92, 2003.

[RG95] M. Rodríguez and J.M García. *Desvíos ferroviarios.* RENFE, 1995.

[Ros71] Blair A. Ross. A survey of western european ac electrified railway supply substation and catenary system techniques and standards. *Industry and General Applications, IEEE Transactions on*, IGA-7(5):666 –672, sept. 1971.

[SC92] Larry Smarr and Charles E. Catlett. Metacomputing. *Commun. ACM*, 35(6):44–52, 1992.

[Sch00] Guus Schreiber. *Knowledge engineering and management. The common KADS methodology.* MIT Press, Boston, USA, 2000.

[SGC$^+$16] Avinash Sodani, Roger Gramunt, Jesús Corbal, Ho-Seop Kim, Krishna Vinod, Sundaram Chinthamani, Steven Hutsell, Rajat Agarwal, and Yen-Chen Liu. Knights landing: Second-generation intel xeon phi product. *IEEE Micro*, 36(2):34–46, 2016.

[SGH15] Nicolás Serrano, Gorka Gallardo, and Josune Hernantes. Infrastructure as a service and cloud technologies. *IEEE Software*, 32(2):30–36, 2015.

[SHK12] Alexander Stanik, Matthias Hovestadt, and Odej Kao. Hardware as a service (haas): Physical and virtual hardware on demand. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, CloudCom 2012, Taipei, Taiwan, December 3-6, 2012*, pages 149–154, 2012.

[SKS00] T. Smithberger, D. Kelly, and A.E. Shaw. *Transportation in the New Millennium: State of the Art and Future Directions.*, chapter Railroad Track Structure System Design. national-academies.org/trb. Transportation Research Board National Research Council, Washington, D.C., USA, 2000.

[Sta11] Stanford University. *The Protege Ontology Editor and Knowledge Acquisition System.* Stanford University, 2011.

[Ste02] Thomas Lawrence Sterling. *Beowulf cluster computing with Linux.* MIT press, 2002.

[Sul45] G. Sulzberger. Les fondations des supportes de lignes ÃĪlectriques aÃĪriennes et leur calcul. *Bulletin Association Suisse des Electriciens*, 10:–, 1945.

[SWDC97] Rick Stevens, Paul R. Woodward, Thomas A. DeFanti, and Charles E. Catlett. From the I-WAY to the national technology grid. *Commun. ACM*, 40(11):50–60, 1997.

[tbb11] TBB (intel threading building blocks). In *Encyclopedia of Parallel Computing*, page 2029. 2011.

[TCY00] Yiliu Tu, Xulin Chu, and Wenyu Yang. Computer-aided process planning in virtual one-of-a-kind production. *Computers in Industry*, 41(1):99 – 110, 2000.

[Ter15] Douglas Terry. Cloud storage services: A model of (in)consistency. In *2015 IEEE International Conference on Cloud Engineering, IC2E 2015, Tempe, AZ, USA, March 9-13, 2015*, page 235, 2015.

[TNI$^+$10] Ryoji Tsuchiyama, Takashi Nakamura, Takuro Iizuka, Akihiro Asahara, Satoshi Miki, and Satoru Tagawa. The opencl programming book. *Fixstars Corporation*, 63, 2010.

[Top14] Top500 Organization. Graph500 list, November 2014.

[Tur59] M.J. Turner. The direct stiffness method of structural analysis. In *Structural and Materials Panel Paper, AGARD Meeting.*, 1959.

[UIC81] UIC. *Electric Traction with Aerial Contact Line.* Union des Chemins de Fer., 1981.

[Ura73] Oktay Ural. Urban structures for lower-cost housing projects. *Computers &amp; Structures*, 3(2):413 – 417, 1973.

[Vee06] C. Veerhoek. 3d cad speeds rail design. *GeoConnection Int. Magazine.*, pages 22–24, April 2006.

[VGM$^+$16] Adriano Vogel, Dalvan Griebler, Carlos A. F. Maron, Claudio Schepke, and Luiz Gustavo Fernandes. Private iaas clouds: A comparative analysis of opennebula, cloudstack and openstack. In *24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2016, Heraklion, Crete, Greece, February 17-19, 2016*, pages 672–679, 2016.

[VOL$^+$11] S. Verstichel, F. Ongenae, L. Loeve, F. Vermeulen, P. Dings, B. Dhoedt, T. Dhaene, and F. De Turck. Efficient data integration in the railway domain through an ontology-based methodology. *Transportation Research Part C.*, 19:617 – 643, 2011.

[Wal07] C. Walton. *Agency and the semantic Web.* Oxford University Press, 2007.

[WB99] T. Wu and M.J. Bernnan. Dynamic stiffness of railway overhead wire system and its effect on pantograph-catenary system dynamics. *Journal of Sound and vibration*, vol. 219(3):483–502, 1999.

[XFPM14] Huanhuan Xiong, Frank Fowley, Claus Pahl, and Niall Moran. Scalable architectures for platform-as-a-service clouds: Performance and cost analysis. In *Software Architecture - 8th European Conference, ECSA 2014, Vienna, Austria, August 25-29, 2014. Proceedings*, pages 226–233, 2014.