OPTIMIZATION AND PARALLELIZATION METHODS FOR THE DESIGN OF NEXT-GENERATION RADIO NETWORKS

Lucas Benedičič

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, September 2013

Evaluation Board:

Assist. Prof. Dr. Jurij Šilc, Chairman, Jožef Stefan Institute, Ljubljana, Slovenia.

Assist. Prof. Dr. Aleš Švigelj, Jožef Stefan Institute, Ljubljana, Slovenia.

Prof. Dr. Marian Vajteršic, Department of Computer Science, University of Salzburg, Austria.

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Lucas Benedičič

OPTIMIZATION AND PARALLELIZATION METHODS FOR THE DESIGN OF NEXT-GENERATION RADIO NETWORKS

Doctoral Dissertation

OPTIMIZACIJSKE IN VZPOREDNE METODE ZA NAČRTOVANJE RADIJSKIH OMREŽIJ NASLEDNJE GENERACIJE

Doktorska disertacija

Supervisor: Assist. Prof. Peter Korošec

Co-supervisor: Assist. Prof. Tomaž Javornik

Ljubljana, Slovenia, September 2013

Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis. Dedicatoria de la thesis.

Contents

Abstract

First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading. First paragraph in current heading.

Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading.

Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading. Next paragraph in current heading.

Povzetek

Prvi odstavek. Prvi odstavek.

Drugi odstavek. Drugi odstavek.

Abbreviations

A

2G	=	Second Generation of mobile networks.
3G	=	Third Generation of mobile networks.
4G	=	Fourth Generation of mobile networks.
CPICH	=	Common Pilot power Channel.
GSM	=	Global System for Mobile communications.
HSDPA	=	High Speed Downlink Packet Access.
HSPA	=	High Speed Packet Access.
HSUPA	=	High Speed Uplink Packet Access.
LTE	=	Long Term Evolution.
RSCP	=	Received signal code power.
RSG	=	Regular square grid.
SHO	=	Soft handover.
UMTS	=	Universal Mobitel Telecommunications System.
WCDMA	=	Wideband Code Division Multiple Access.

S

γ^{cov}	=	Signal-to-interference ratio (SIR) coverage threshold.
γ^{cov}	=	Signal-to-interference ratio (SIR) coverage threshold.
γ^{sho}	=	SHO window.
A_{covered}	=	Area under service coverage of the mobile network.
A_{total}	=	Complete geographical area under optimization.
$as^{\max}$	=	Maximum number of neighbouring cells in the active set.
C	=	Set of antenna installations (cells) in a mobile network.

C = Set of antenna installations (cells) in a mobile network.

C_m = Subset of cells, $C_m \subset C$, that cover a mobile $m \in M$.

c_m^* = Best-serving cell of mobile $m \in M$.

c_m^* = Best-serving cell of mobile $m \in M$.

$cov(c, m)$ = Binary function to assert the coverage of a mobile $m \in M$ by from a cell $c \in C$.

$cov(x, y)$ = Returns 1 if the coordinate (x, y) is under mobile-network coverage.

cov_{cm} = Returns 1 if the mobile $m \in M$ is under radio coverage of cell $c \in C$.

f_{cov} = Objective function for the service-coverage optimization problem.

f_{sho} = Objective function of the SHO-balancing problem.

$L_{cm}^{\downarrow}$ = Downlink attenuation factor between cell $c \in C$ and mobile $m \in M$.

$L_{cm}^{\downarrow}$ = Downlink attenuation factor between cell $c \in C$ and mobile $m \in M$.

$L_{mc}^{\uparrow}$ = Uplink attenuation factor between mobile $m \in M$ and cell $c \in C$.

$L_{mc}^{\uparrow}$ = Uplink attenuation factor between mobile $m \in M$ and cell $c \in C$.

M = Set of mobile devices or users of a mobile network.

M = Set of mobile devices or users of a mobile network.

P_c = Set of candidate CPICH power settings for cell $c \in C$.

P_c = Set of candidate CPICH power settings for cell $c \in C$.

p_c = Pilot-power setting of cell $c \in C$.

p_c = Pilot-power setting of cell $c \in C$.

$p_m^{\uparrow}$ = Uplink transmit power of mobile $m \in M$.

$SHO_m^{\downarrow}$ = Cell set to which a mobile may maintain concurrent downlink connections, i.e. downlink SHO.

$SHO_m^{\uparrow}$ = Cell set to which a mobile may maintain concurrent uplink connections, i.e. uplink SHO.

$sir(c, m)$ = Signal-to-interference ratio (SIR) from cell $c \in C$ to mobile $m \in M$.

$sir(c, m)$ = Signal-to-interference ratio (SIR) from cell $c \in C$ to mobile $m \in M$.

1 Introduction

Many researchers believe the computer has become the third method to do research, behind theory and experimentation, for both science and engineering. Although there is no complete agreement on the position intended for scientific computing with respect to the other two methods, it is undeniable that computational methods are an essential tool in most disciplines, particularly in those related to decision making.

Nowadays, decision making is present practially everywhere. As scientistis, engineers and managers have to make decisions in more complex and competitive circumnstances every day, decision making involves dealing with rational and optimal approaches. According to Talbi [1], decision making consists in the following steps:

- formulate the problem,
- model the problem,
- optimize the problem, and
- implement a solution.

Formulating a decision problem means making an initial statement about it. Despite this first formulation may be imprecise, the objectives of the problem are outlined, together with internal and external factors that have some degree of influence over it. During the modelling of the problem, an abstract mathematical model is built for it. Sometimes this model is inspired by similar models in the literature, making it possible to tackle the problem with well-studied methods. After a model of the problem is available, the optimization step, i.e. generating "good" solutions for the problem, may begin. It is worth pointing out that the resulting solutions are given for the abstract model, and not for the original problem itself. Therefore, the performance of the obtained solution is indicative when the model is an accurate one [1]. In the last step, the obtained solution is practically tested by the decision maker and implemented if it is an "acceptable" or "good" one. In case of "bad" or "unacceptable" solutions, the decision-making process is repeated, possibly improving the model and/or the optimization algorithm. The process, as described here, is depicted in Figure 1.1.

Scientific computing, by means of computer-science methodology, makes possible the study of problems that are too complex to be treated analytically, or those that are very expensive or dangerous to be studied by direct experimentation. Real-world problems are typically very complex systems to be directly assessed by analytical models, and require a numerical simulation for their study. Computer simulations provide a resource to mimic the behavior of complex systems, by numerically evaluating a model and gathering its data to estimate their true characteristics [2].

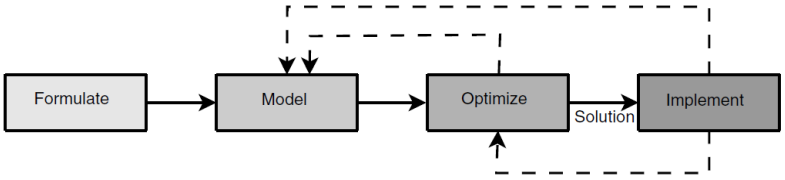


FIGURE 1.1 The classical process in decision making: formulate, model, solve, and implement. In practice, this process may be iterated to improve the optimization model or algorithm until an acceptable solution is found. Like life cycles in software engineering, the life cycle of optimization models and algorithms may be linear, spiral, or cascade.

Figure 1.1: The classical decision-making process, as presented in [1]. Multiple iterations of this process improve the optimization algorithm and/or model until an acceptable solution is found.

A model is a simplified representation of a studied problem, and one of its purposes is to predict the effects of variations within the system. A good model is a balance between realism and simplicity. The system simulation, on the other hand, is the operation of the model. Its configuration can also be changed, allowing multiple experimental executions, something that might not be possible with the real system it represents [3]. However, it is important to understand that the models used in scientific simulations and engineering never offer a perfect model of the system they represent, but only a subset of its composition and dynamics. For this reason, experimentation and expert observation will always be essential as reference points for understanding the studied phenomena. Consequently, problems categorized as of large size and of considerable complexity represent a challenge, because of the different involved disciplines and the degree of difficulty of their modeling. Radio networks, in particular those of the third and fourth generations, fall under this characterization.

Radio networks represent one of the most fast-growing technology markets since the introduction of the Global System for Mobile communications (GSM) [4] as the second generation (2G) mobile networks more than twenty years ago. Its successor, the Universal Mobile Telecommunications System (UMTS) [5] marks an evolution from 2G, representing a milestone for the third generation of mobile radio networks (3G). In recent years, the first commercial networks implementing the Long Term Evolution (LTE), also known as fourth generation (4G) LTE, have also appeared [ref???]. The always increasing demand for more bandwidth has been one of the main forces behind the standardization and later implementation of systems delivering higher speed data services in order to improve the user's experience.

This evolution, first from 2G to 3G and later from 3G to 4G, has introduced not only the technology needed to increase both data-transfer capacity and voice quality, but also a greater complexity in terms of radio-network planning, deployment, and configuration. This fact has attracted the attention of the research community into areas such as the design and optimization of radio-networks.

In a traditional or manual approach, during the design phase of a radio network, a software tool would execute the analysis, while the human would make the change decisions. Therefore, a radio engineer configures the network parameters manually and the software tool analyzes the given configuration. If the obtained results are not acceptable, the analysis process has to be repeated several times, until the goal is achieved. We will refer to this process as manual radio-network optimization.

Advances in the last few years have improved the manual optimization process by introducing different problem-solving approaches that increase the role the computer has during the optimization of radio networks, consequently enlarging the scope of problems and instance sizes that may be subject to automated optimization. Still, there are some important aspects that restrict the utilization of these methods not only in real-world enviroments, but also when doing research in the area of optimization of radio networks:

- It is usually the case that the selected method is a compromise between solution quality and computational-time complexity. The proposed state-of-the-art approach for the evaluation of radio networks is the Monte-Carlo snapshot analysis. However, real-world environments, where radio-network design is carried out, require the evaluation of networks comprising up to thousands of base stations in a reasonable amount of time. Moreover, for applications involving radio-network optimization, multiple thousand evaluations are required to find a good solution, in which case also snapshot simulations are too time-consuming to be employed. Therefore, for such applications and environments, methods with improved time efficiency are required.

- A considerable number of publications in the field of radio-network optimization, of which we cite just a few for reference purposes [6–11], base their simulations on platforms for which it is not possible to reproduce the experiments, either because they have used proprietary software or because the data is not available. This fact reduces the possibilities for comparing different approaches among each other, and significantly contrasts with other research areas, such as evolutionary computing or artificial intellingence, which have a set of open and well-known benchmarks for the community to use. Thus, an open and standardized framework would allow researchers to compare different methods and results in a simple and objective manner.

- Available commercial tools for radio-network evaluation have major drawbacks with respect to the size of networks that can be considered, simulation time, and especially the accuracy of the modeled system. Yet, if precise and fast methods were commercially available, the would lack the level of flexibility required by the scientific community. Consequently, an essential attribute of the framework is to be open source, so that anyone could extend it to meet some specific requirements. In the long term, this process should also extend the set of built-in functionality.

- Particularly for path-loss predictions, created with empirical mathematical models, the inaccuracy of the input data directly deteriorates the precision of the calculated results. Moreover, since the physical properties that influence the propagation of radio signals are not constant and every environment introduces its own deviations, the calculated coverage may be considered as not more than an approximation. Therefore, there is a need for a method to adapt state-of-the-art mathematical models for radio propagations so that the calculated path-loss predictions are as accurate as possible, despite the various sources of error noted before.

This thesis focuses on providing methods and tools to ??? the pointed drawbacks. Short summary of the introduced novelties??? It is important to note that some methods proposed in this thesis have been particularly designed for problems emerging in radio planning of 3G networks. Despite this, they may be adapted to other standards, e.g. GSM or LTE, without lose of generality.

1.1 Scope

In this thesis, we intend to contribute methods and tools that will provide solutions to the disadvantages pointed out in the previous section. The introduced methods and tools have a close correlation with the simulation and optimization of radio networks, especially those of the 3G and 4G. We will introduce the steps necessary to mitigate the problem of experimental reproducibility found in most published works, by simplifying setup, execution, and sharing of experimental results. With the development of the framework, we will evaluate and assess the possibilities it offers as a support system for simulation and optimization problems of radio networks. By including parallel programming techniques for computer clusters and GPUs, we intend to go beyond the classical methodology provided by previous works, taking advantage of the inherent parallelism of some optimization techniques. We will also be looking into the application of many advances in HPC that should provide the framework with the computing power needed to improve the simulation process, thus enhancing its scalability to support real-world 3G radio networks. Finally, since we believe that this work will only become truly productive through the cooperation and long-term development of the scientific and engineering community, we will be releasing the source code, algorithms, documentation, and data to the public domain. This way, anyone will be able to use and to extend the framework for their own needs. Encouraging cooperation and sharing of experimentation-related tools and data should be a common goal from which everyone will benefit.

1.2 Hypothesis and aims

- The great proportion of published works about optimization problems for 3G radio networks is very difficult to reproduce, if possible at all.
- A common and open framework, designed for simulation of 3G radio networks, should mitigate the experimental reproducibility problem by simplifying sharing of experimental results.
- Parallelization techniques improve the scalability and time requirement of the framework, thus making it possible to process real-world data sets.
- Using hardware, specialized for parallel execution (e.g. GPU), improves the time requirement of parallel algorithms using threads or message-passing mechanisms.

Aims

- Analyze the state-of-the-art of optimization problems for 3G radio networks in general, and UMTS in particular.

- Identify common obstacles in optimization problems for 3G radio networks that prevent them from being reproducible by other members of the research community.

- Design and implement framework tools to provide a common open environment that will enable the scientific community to easily share and reproduce different experimental conditions for maintenance and optimization of 3G radio networks.

- Integrate parallelization techniques to provide framework scalability, so that it will be able to deal with large real-world data sets.

- Evaluate the framework design by reproducing optimization environments and introducing new algorithms for tackling previously unsolved instances of optimization problems in 3G radio networks.

- Evaluate and analyze all experimental results and simulations in the context of real networks, using real-world data.

1.3 Methodology

The dissertation will use the following methodology to prove the hypothesis stated in the previous sections.

First, we will survey existing optimization problems and techniques for 3G radio networks in general, and UMTS in particular. The focus of the survey will be on optimization problems that do not provide the means required by experimental reproducibility and require the use of a snapshot-based model of the radio network.

Second, we will design and develop a parallel framework for radio network simulations. The development of the framework will focus on joining optimization and simulation, so that it will cover a wider range of use cases, namely as a maintenance and optimization tool for 3G radio networks. Additionally, we will make an intensive study of the mathematical models currently used, seeking to assess their reliability when used in simulations of 3G radio networks. Since the algorithms and simulations used in the optimization of 3G radio networks consume large amounts of data and require high computing power (due to the large number of simulations they need to run), we will improve the performance of the framework even more by adapting its most time-consuming parts for execution on GPUs.

Finally, we will evaluate the benefits of the above-mentioned framework by conducting reproducible experiments that will support the hypothesis outlined under ???, by tackling similar and bigger problem instances when solving both well-known and new optimization problems for 3G radio networks.

1.4 Contributions

The expected contributions of this dissertation to the fields of telecommunications and computer sciences include the following:

- State-of-the-art overview of optimization methods for 3G radio networks.

- Design and development of a framework that provides an open environment for radio network simulations, implemented for execution on computer clusters and GPUs. The framework will allow the scientific community to share a common domain to run the simulations needed by modern optimization methods, since most currently available simulation tools are proprietary and therefore unsuitable for experimental reproducibility.
- Improvement of quality and speed of renowned mathematical models, used for radio propagation predictions, by applying parameter optimization and parallelization techniques. The expected speed improvement should be of at least one order of magnitude.
- Proposal of a new algorithm, based on autonomous agents, to solve the service coverage problem. The solved problem instances should be of bigger size than ever solved in the literature and reach equal or better quality thereof. This should make our approach applicable for large real-world problem instances and data sets.
- Identification of a new optimization problem in 3G radio networks that deals with soft-handover alignment of downlink and uplink areas. By solving this problem, we should avoid abnormal network functioning in areas where there is soft-handover capability in the uplink, but none in the downlink. So far this problem has been solved manually by radio experts.
- Empirical comparison of the proposed metaheuristic algorithm against the existing state-of-the-art optimization algorithms on the soft-handover alignment problem.

1.5 Organization

The introduction provided in this chapter pretends to delimiter the context within which the dissertation will address.

The rest if this dissertation is organized as follows.

Chapter 2 present an overview of some well-known optimization problems that occur during deployment and configuration of mobile networks. A description of each optimization problem is given, followed by a short survey of recently proposed optimization methods. It closes by giving a conclusion regarding mobile networks and why they are a rich source of optimization problems.

Chapter 6 deals with the automatic tuning of parameters of the mathematical models, used for radio propagation predictions. Namely, based on field measurements, the configurable parameters of the mathematical models used by the framework may be automatically tuned to minize the deviation from the prediction to the actual state of the network.

In Chapter 5, a static network simulator is used to find downlink and uplink SHO areas. By introducing a penalty-based objective function and some hard constraints, we formally define the problem of balancing SHO areas in UMTS networks. The state-of-the-art mathematical model used and the penalty scores of the objective function are set according to the configuration and layout of a real mobile network, deployed in Slovenia by Telekom Slovenije, d.d.. The balancing problem is then tackled by three

optimization algorithms, each of them belonging to a different category of metaheuristics. We report and analyze the optimization results, as well as the performance of each of the optimization algorithms used.

Chapter **??** considers the problem of minimizing the total amount of pilot power subject to a full coverage constraint. Our optimization approach, based on parallel autonomous agents, gives very good solutions to the problem within an acceptable amount of time. The parallel implementation takes full advantage of GPU hardware in order to achieve impressive speed-up. We report the results of our experiments for three UMTS networks of different sizes based on a real network currently deployed in Slovenia.

2 Background and motivation

Short introduction to the content of this chapter.

2.1 Optimization models

Optimization may be informally defined as the procedure of finding better solutions to a given problem, which usually model some physical phenomenon. In our every day life, we are constantly solving small optimization problems, like choosing the shortest route to a friend's house, or organizing the appointments in our agenda. In general, these problems are small enough for us to find a good solution without extra help, but as they become larger and more complex, the aid of computers for their resolution is unavoidable.

Complex multidimensional optimization problems are popular in engineering, economics, physics and other scientific fields. When solving an optimization problem, the objective is to find a "good" solution in a "reasonable" computational time. In this respect, the field of mathematical optimization has received a lot of attention by the scientific community during the last decades. However, both "good" and "reasonable" are problem, application and context-specific concepts, in which the biggest challenge of selecting an appropiate optimization approach usually lays.

Mathematical optimization involves the process of finding solutions from a group of possible decisions, which may be defined as:

$$\min f(\vec{x}) \qquad \vec{x} \in \Omega \subseteq \mathbb{R}^n, \tag{2.1}$$

where $\vec{x} = (x_1, \ldots, x_n)$ is a vector representing the decision variables, $f(\vec{x})$ is the objective function measuring the quality of the decisions and Ω is the set of feasible solutions of the problem, also known as search space. Note that the objective function f makes it possible to define a total order relation between any pair of solutions in the search space Ω.

The search space Ω may also be expressed as a solution to a system of equalities or inequialities, e.g.:

$$\begin{aligned} g(x_1, \ldots, x_n) &\leq 0 \\ h(x_1, \ldots, x_n) &= 0 \end{aligned} \tag{2.2}$$

Optimization problems involving the maximization of the objective function also fall into this category, since:

$$\max f(\vec{x}) = -\min(-f(\vec{x})) \tag{2.3}$$

A point $\vec{x}^*$ is considered to be an unrestricted local minimum of a function f if it holds a better value than all its neightbours, i.e. there exits $\epsilon > 0$ so that:

$$f(\vec{x}^*) \leq f(\vec{x}) \qquad \forall \vec{x} \in \mathbb{R}^n \qquad | \vec{x} - \vec{x}^* | < \epsilon \tag{2.4}$$

Similarly, a point $\vec{x}^*$ is considered to be an unrestricted global minimum of a function f if it holds a better value than all others, i.e.:

$$f(\vec{x}^*) \leq f(\vec{x}) \qquad \forall \vec{x} \in \mathbb{R}^n \tag{2.5}$$

The concepts of local and global minimum are considered strict if the inequalities of 2.4 and 2.5 are strict. Likewise, the definition of local and global maximum is given by the existing relation between a minimization and a maximization problem, as specified in 2.3, i.e. a point $\vec{x}^*$ is a local or global maximum of a function f if and only if $\vec{x}^*$ is a local or global minimum of function $-f$, respectively.

2.1.1 Gradient-based methods

Gradient-based methods are among the oldest and most studied optimization approaches. They are based on the derivative of the optimized function, using the first and even the second derivate of a function f. The name gradient follows from the derivative of multidimensional functions, $\nabla f(\vec{x})$, which is simply a vector where each element is the slope of $\vec{x}$ in that dimension, i.e. $< \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} >$ [12].

The principle behind gradient-based methods is rather simple. Starting from an arbitrary value for x, we iteratevily substract (or add) a small positive value to it, e.g. for gradient descent:

$$x \leftarrow x - \alpha f'(x), \tag{2.6}$$

where α is a small positive value. Consequently, a positive slope will make x decrease, whereas a negative slope will make it increase. Figure 2.1 shows an example of this behaviour. Therefore, x will gradually move down the function until it finds its minimum, where $f'(x)$ is zero, causing it to stop.

However, gradient methods have certaing drawbacks that make them unsuitable for tackling a wide range of optimization problems. Take, for example, the time they take to converge. As gradient descent approaches a function minimum, it will skip this point and land on the other side. In the next step, something similar will happen, but this time from the other side of the minimum point, thus slowly approaching to the target in a "zig-zag" way. This behavior is directly related to the slope of the function at the given point, i.e. a steepest slope translates into a larger jump, and may be allivieatted by adjusting the value of α. However, some functions (or regions of functions) may require smaller values, while for others a bigger value would be more appropiate. Newton's method improves this by taking the second derivative of the function into account, i.e.:

$$x \leftarrow x - \alpha \frac{f'(x)}{f''(x)}, \tag{2.7}$$

thus adjusting the value of α as it converges towards a point with zero slope [12].

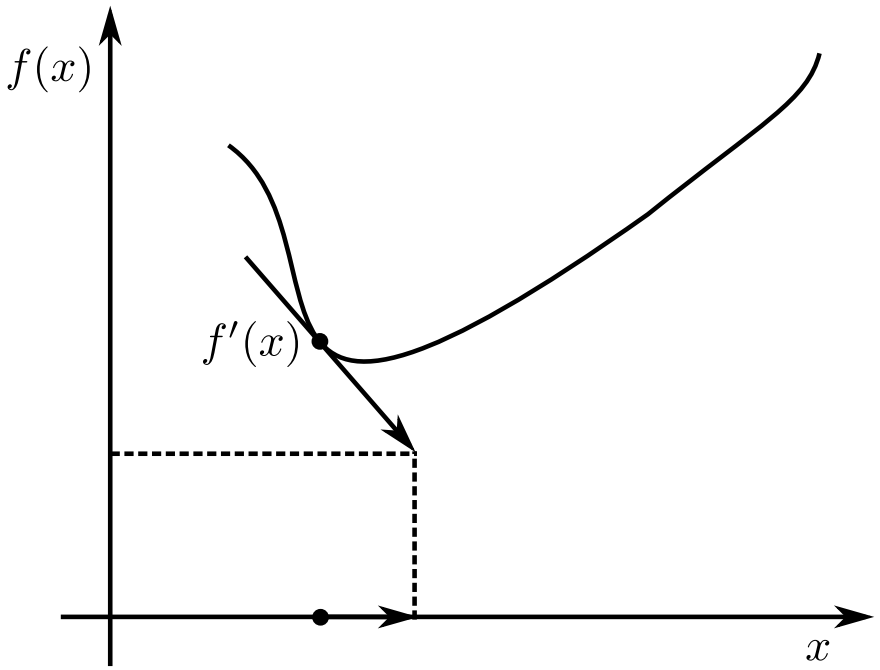


Figure 2.1: Gradient descent with a negative slope, i.e. x is increasing.

Another issue is how other points are handled. Beside maxima and minima points, some functions also contain saddle points (known as inflection points in one-dimensional functions). Clearly, the first derivative of a saddle point is zero, meaning gradient descent will stop looking for the minimum, even though it hasn't found it (see Figure 2.2). Newton's method, on the other hand, does not help either. Moreover, in this case, we would be even dividing by zero! These observations clearly show how gradient methods get caught in local optima. We define local optima of a function as the optima (or minima in our case) of a local region. Similarly, global optima are defined as the optima of the whole domain of a function. It follows that gradient methods, as gradient descent or Newton's method, are local optimization algorithms [12].

But maybe the biggest concern with gradient-based methods is they assume the function under optimization is derivable. This assumption holds only when optimizing a well-formed mathematical function. Unfortunately, this is generally not the case, since in most cases the gradient is not computable because the function is not known. The only available approach in such situations is creating inputs to the function in order to assess their quality. Metaheuristics are good candidates for this class of problems to solve moderate and large instances.

2.1.2 Linear and non-linear programming

It was in the early 40s of the twentieth century, through the work of teams formed by mathematicians, economists and physicists, that the basis were established for the resolution of problems with a set of techniques known as linear and non-linear programming. Their initial goal was to solve different kinds of logistic problems during the second world war.

In a linear programming optimization problem, both the objective function f and a given set of constraints are linear functions. The constraints impose restrictions

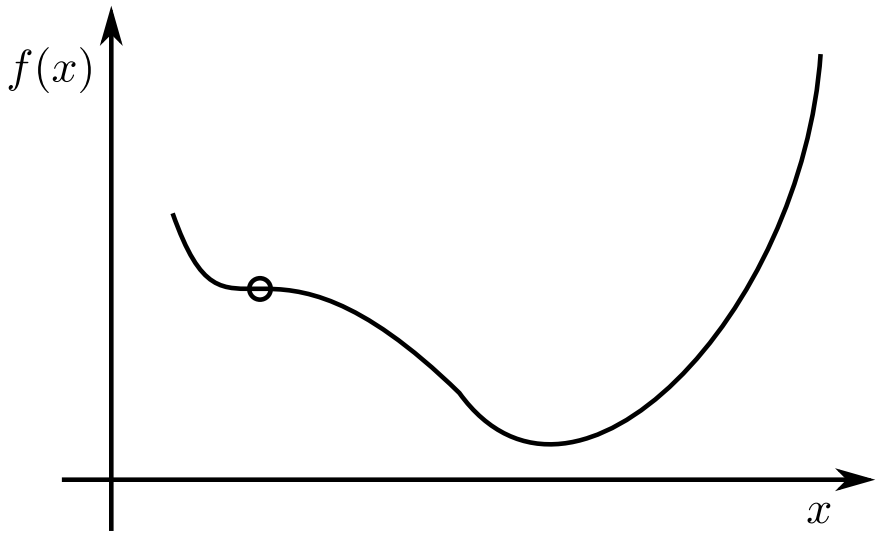


Figure 2.2: A saddle point or point of inflection, where the derivative is zero.

over $\vec{x}$, i.e. they must meet certain requirements as, for example, fullfil a limited availability of resources. An problem may be formulated as follows, e.g.:

$$\min f(\vec{x}) = c \cdot \vec{x} \tag{2.8}$$

subject to

$$\begin{aligned} A \cdot \vec{x} &\leq b \\ \vec{x} &\geq \vec{0} \end{aligned} \tag{2.9}$$

In the example above, the inequalities defined in 2.9 are the constraints to the linear program defined in 2.8.

For solving continous linear optimization problems, efficient exact algorithms exist, such as the simplex method [13] or the interior-points method [14]. Indeed, linear programming is one of the most satisfactory models of solving optimization problems, since the feasible region of the problem is a convex set and the objective function is a convex function. It follows that the global optimum is a node of the polytope representing the feasible region [1]. See Figure 2.3 for a linear-programming example with several constraints.

Non-linear programming models, on the other hand, consider problems where the objective function f and/or the constraints are non-linear [15]. However, non-linear continuous problems are more difficult to solve. Despite several existing technique to linearize such models, they often not only introduce extra variables and constraints, but also some degree of approximation [16]. Moreover, some problem properties such as high dimensionality, parameter interaction, and multi-modality make these approaches ineffective.

Generally, when dealing with real-world problems, the availability of analytical optimization models, such as required by gradient methods or (non-)linear programming, is not guaranteed. Indeed, for some applications, only simulations or physical models are the available means for objective-function evaluation [17]. Once again,

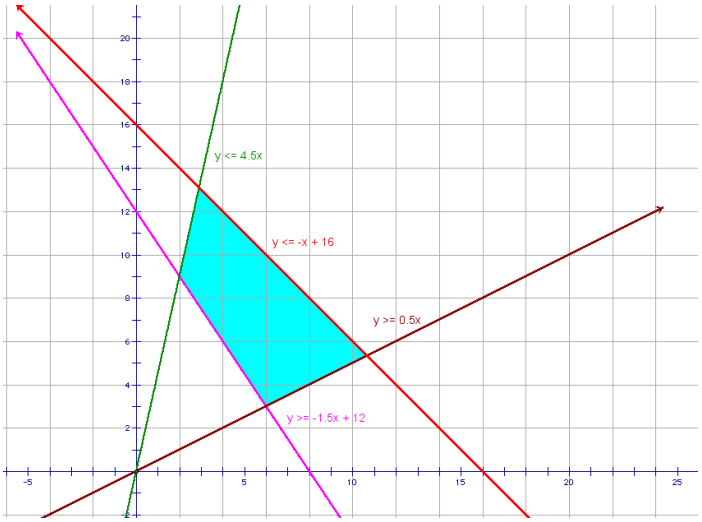


Figure 2.3: Graphical representation of a linear-programming example with several constraints. The greyed area is the polytope representing the region of feasible solutions.

metaheuristics appear as good candidates to solve different instance sizes of this class of problems.

2.1.3 Metaheuristics

Metaheuristics, a term proposed by Glover in [18], represent a group of approximation algorithms designed to combine basic heuristic principles with advanced high-level guidance methods, targeted at improving the efficiency of a search process. These techniques are meant to find good solutions to a given problem, for which the mathematical function is not available or its search space is big enough for an exhaustive search to be unfeasible [19].

From the theorical point of view, metaheuristics represent a subset of stochastic optimization, since they use some degree of randomness to find optimal (or as optimal as possible) solutions to hard problems. They are the most general of these kinds of algorithms, and are applied to a wide range of problems [12]. Unlike the exact optimization methods introduced in the previous sections, metaheuristics do not guarantee the optimality of the obtained solutions [1]. Moreover, they do not define how close the obtained solutions are from the optimal ones, as approximation algorithms do.

The characterization given by Blum and Roli [20] provides a clear overview of the fundamental properties associated with metaheuristics:

- metaheuristics are strategies that “guide” the search process;
- their goal is to efficiently explore the search space in order to find optimal or near-optimal solutions;
- they build upon techniques which range from simple local search procedures to complex learning processes;
- they are approximate and usually non-deterministic;

- they may incorporate mechanisms to avoid getting trapped in confined areas of the search space;
- their basic concepts permit an abstract-level description, which is not problem-specific;
- they may make use of domain-specific knowledge in the form of heuristics that are controlled by the upper level strategy;
- advanced metaheuristics use search experience (implemented as some form of memory) to guide the search process.

The strategies used by metaheuristics should provide a dynamic balance between the exploitation of the accumulated search experience (commonly called intensification) and the exploration of the search space (commonly called diversification) [20]. This balance provides the necessary means to quickly identify promising regions, and early discarding those which have already been explored or don't provide solutions of better quality. Promising regions within the search space, which are identified by the obtained "good" solutions, are thoughfully explored during the intensification phase, hoping to find better solutions. On the other hand, during the diversification phase, not yet visited regions are explored, making sure the search space as a whole is evenly explored, thus avoid confining the search to a reduced number of regions. In this context, the ultimate search algorithm in terms of diversification is random search. Random search generates a random solution in the search space at each iteration, without using memory [1]. In terms of intensification, iterative local search is a representative algorithm. The steepest local search algorithm selects, at each iteration, the best neighboring solution that improves the current one [1].

Metaheuristics are applicable where state-of-the-art exact algorithms cannot tackle the given instances within the required time, either because of the size or the structure of the given problem instances. The meaning of "required time" within this context directly depends on the target optimization problem itself. A feasible or acceptable time may vary from some seconds to several months, again, depending on the target optimization problem, e.g. real-time decisions against structural-design problems.

Based on the characterization given by Talbi [21], let us summarize the essential properties of optimization problems that justify the use of metaheuristics:

- Very large problem instances. Even though exact polynomial-time algorithms might be known for solving the target problem, they are too expensive due to the size of instances.
- Problems with hard real-time constraints, where a "good solution" has to be found online. Metaheuristics appear as an alternative to exact algorithms in order to reduce the search time.
- A difficult problem of moderate size, which input instances have an intrincate structure.
- Optimization problems with time-consuming objective function(s) and/or constraints. Indeed, various real-world optimization problems are characterized by the huge computational cost of the objective functions. Several radio-network design problems fall into this category.

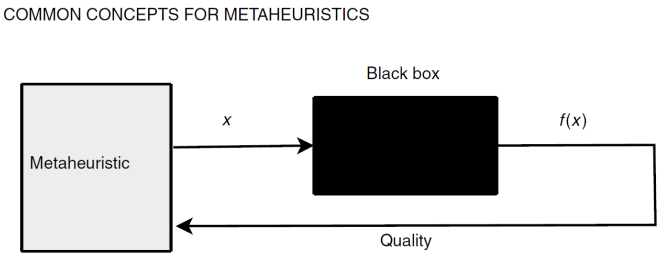


FIGURE 1.14 Black box scenario for the objective function.

Figure 2.4: A metaheuristic optimization process using a black box for objective-function evaluation.

- Problems that cannot be solved with exhaustive search due to the non-analytical models on which they are based. These problems are defined by a black-box evaluation of the objective function (see next section).

The influence of these conditions may increase in the presence of non-deterministic optimization models, e.g. problems with complex Monte Carlo simulations [22].

Undoubtedly, metaheuristics are rapidly gaining popularity as optimization problems are increasing in both size and complexity. Indeed, as the computing power of commodity hardware increases, the possibility of building models of greater complexity is available for developing more accurate models of real-world problems in engineering and science.

2.1.4 Black-box optimization

A problem complexity is equivalent to the complexity of the best algorithm solving that problem [21]. If there exists a polynomial-time algorithm to solve a problem, we say the problem is easy or tractable. Similarly, if a problem is difficult or intractable, there is no known polynomial-time algorithm to solve it.

Many optimization problems cannot be formulated with a clear analytical mathematical notation. In such cases, the objective function may become a black box [23]. This is one of the main advantages when using metaheuristics, i.e. there is no need of a complete knowledge of the targeted model. Indeed, in a black box optimization, no analytical formulation of the objective exists [21], as Figure 2.4 shows.

More specifically, we say a function $f(\vec{x})$, $\vec{x} \in \mathbb{R}^n$, is a black-box function if and only if [21]:

- the domain $\vec{x}$ is known,
- it is possible to get the value of f for each $\vec{x}$ based on simulation, and
- there is no other information available for function f.

Typically, the experiments associated with these kind of problems are very expensive in terms of time and cost, since a simulation must be forced to evaluate the solution. Generally speaking, the most time-consuming part of a metaheuristic optimization process is the evaluation of the objective function [1]. This is especially true when dealing with real-world problems of areas such as structural design [24], molecular docking [25] and, the field on which this thesis focuses, radio-network design [26]. A

possible subtitution for lenghty evaluations is to reduce their complexity by approximating the objective function, thus replacing the it with an approximation during the optimization process. This approach is known as meta-modeling [1]. However, when dealing with approximations, some degree of solution quality is inevitably sacrified. As we will show in the following chapters, there is a very fine balance between the number of evaluations and the quality of the achieved solutions. Consequently, reducing the time spent in objective-function evaluation should favorably influence the solution quality achieved by a preferred metaheuristic algorithm. A major portion of this thesis is dedicated to improve this specific aspect on the area of radio-network optimization.

In practice, black-box evaluation of the objective function presents an inherent problem. In the context of research works related to radio-network optimization, there is an increasingly habit of not providing the black-box used to evaluate a given approach, and that leads to the results provided. A quick review of the state-of-the-art in radio-network optimization indicates that this fact has become increasingly regular in several published works [refs???]. This fact creates a barrier to one of the most important phases of scientific methodology: experimental reproducibility [27].

There are several reasons why the situation has reached this point. It is a known fact that proprierary software, providing good computational models for radio-network simulation, is a very expensive tool for science. Even neglecting the economical aspect, but considering the great variety of software packages and license combinations, it is practically impossible for a research laboratory to have at its disposal the whole pallete of commercially-available solutions. Moreover, genuine users of these applications are generally not allowed to mention the formats, protocols, or algorithms used by the proprietary software, since their disclosure is explicitly forbidden by the commercial licenses.

If we turn our attention to non-commercial and open-source packages for radio-network simulation [refs!???], we would quickly run into mainly two difficulties: poor documentation and low scalability. The scalability issues shown by some projects [refs!???] retrain the packages to be used in real-world environments, where big problem instances are the rule. Despite this, it is a huge merit and acknowledgement to the authors of this packages, not only for providing it to the scientific community, including their source code, but fundamentaly because providing an environment in which different kinds of simulations are completely reproducible. Regarding the lack of documentation, it represents a big hurdle when extending the base code, which becomes a difficult taskj without the help of the original authors of the package, who have a deep knowlegge of the code-base. This knowledge is required in order to effectivelly expand the functionality of the open-source tool in question. This is especially true when dealing with complex simulation frameworks, as the ones used for radio networks.

2.1.5 Metaheuristic algorithms

Related literature groups metaheuristic algorithms due to their behaviour. For example, the following is a list of fundamentally different optimization algorithms, namely:

- differential evolution, from the family of evolutionary algorithms;

- differential ant-stigmergy algorithm, from the family of swarm-intelligence algorithms; and
- simulated annealing, from the group of classic metaheuristic algorithms, targeted at combinatorial optimization problems.

Each of these algorithms shall minimize an objective-function value by adopting essentially disparate approaches, hence the diversity of applying algorithms belonging to different families to solve the same optimization problem. In this way we want to find out whether any of the presented approaches is better suited for solving our problem.

In the following sections we give a short introduction about their functioning and controlling parameters.

2.1.6 Differential evolution

Differential evolution (DE) [28] is a simple and powerful evolutionary algorithm proposed for global optimization. A wide range of optimization problems have been solved by applying DE [29]. The algorithm exhibits a parallel direct search method, which utilizes D-dimensional parameter vectors. The balancing problem is expressed in each component of a vector X of the population, which maps to the CPICH power of one cell under optimization:

$$X_{aG} = \{x_1, x_2, \ldots x_i, \ldots, x_D\}, \tag{2.10}$$

where $x_i \in P_i$ represents a candidate CPICH power setting of cell i, and G indicates the generation of an individual a in the population. Since there are $|N|$ cells in the mobile network, it follows that $D = |N|$.

In each generation, DE produces new parameter vectors by adding the weighted difference between two population vectors to a third one [28]. The resulting vector is retained if it yields a lower objective function value than a predetermined population member; otherwise, the old vector is kept.

There are different variants of DE. We have chosen the most popular one to solve our optimization problem, called *DE/rand/1/bin*. The nomenclature used to name this variant indicates the way the algorithm works:

- *DE* denotes the differential evolution algorithm,
- *rand* indicates that the individuals selected to compute the mutation values are randomly chosen,
- 1 specifies the number of pairs of selected solutions used to calculate the weighted difference vector, and
- *bin* means that a binomial recombination operator is used.

We considered four parameters to control the search process of DE: the population size, the maximum number of generations for the algorithm to run, the crossover constant, and the mutation scaling factor.

An extensive description of DE and its variants may be found in [30].

2.1.7 Differential ant-stigmergy algorithm

Based on the metaheuristic Ant-Colony Optimization (ACO) [31], the differential ant-stigmergy algorithm (DASA) [32] provides a framework to successfully cope with high-dimensional numerical optimization problems. It creates a fine-grained discrete form of the search space, representing it as a graph. This graph is then used as the walking paths for the ants, which iteratively improve the temporary best solution.

The mapping between the balancing problem and DASA is similar to the one depicted in Equation (2.10):

$$X_a = \{x_1, x_2, \ldots x_i, \ldots, x_D\} \tag{2.11}$$

In this case, each ant, a, creates its own solution vector, X_a, during the minimization process. At the end of every iteration, and after all the ants have created solutions, they are evaluated to establish if any of them is better than the best solution found so far.

There are six parameters that control the way DASA explores the search space: the number of ants, the discrete base, the pheromone dispersion factor, the global scale-increasing factor, the global scale-decreasing factor, and the maximum parameter precision.

For a more in-depth explanation about these parameters and the DASA algorithm itself, we refer the reader to [32].

2.1.8 Simulated annealing

As the third optimization algorithm to tackle the balancing problem we have chosen simulated annealing (SA) [33], a classic metaheuristic algorithm often used when the search space is discrete. SA has proved to be a solid optimization algorithm, capable of giving high-quality solutions to a wide scope of optimization problems [34].

At each time step during the process, the system under optimization is in a given *state*. The objective function maps a system state to a value known as the *energy* of the system in that state. A *move* in the search space represents a change in the state of the system. After making a move, the system may exhibit lower or higher energy, depending on the results of the objective function. When dealing with minimization problems, a better state always describes lower energy than the previous one.

SA incorporates the notion of *temperature*, by which the probability of moving the current state of the system into a worst one is lowered as the temperature decreases. Exploration of the search space is thus induced at higher temperature, whereas exploitation appears at lower temperature, when only improving moves are accepted.

Table 2.1 shows the pseudo-code of a move in the search space of possible CPICH power settings, resulting in a new state of the system.

At the first step, a cell, i', is randomly selected from the set of all cells in the network, N. In step 2, a change of +0.01 dB or -0.01 dB is applied with 50% probability to $p_{i'}$. The current CPICH power of cell i' is expressed in dBm. The randomly generated CPICH power setting, $p_{i'}^{\text{NEW}}$, is checked for validity in step 3, i.e. it must be an element of the set $P_{i'}$. If $p_{i'}^{\text{NEW}}$ is not a valid CPICH power, step 2 is executed again, generating another random CPICH power. Finally, in step 4, the CPICH power of cell i is replaced by $p_{i'}^{\text{NEW}}$.

Table 2.1: Pseudo-code: a move in the search space of SA.

Step	
1	$i' = random\,cell(N)$
	do
2	$if\, rand() < 0.5\, then\, p_{i'}^{\mathrm{NEW}} = p_{i'} + 0.01$
	$else\, p_{i'}^{\mathrm{NEW}} = p_{i'} - 0.01$
3	**while** $p_{i'}^{\mathrm{NEW}} \notin P_{i'}$
4	$p_{i'} = p_{i'}^{\mathrm{NEW}}$

It is important to note that, as long as $|P_{i'}| > 1$, the algorithm shown in Table 2.1 shall never be trapped in an endless loop. On the other hand, if $|P_{i'}| < 2$, there are no candidate CPICH powers for cell i' and thus no possibility of optimization by means of CPICH power adjustment.

Notice also that the acceptance of a move in the search space is left to SA and its stochastic components.

2.2 Optimization of radio networks

Once a radio network is launched, an important part of its operation and maintenance deals with monitoring its quality characteristics and making changes in the configuration of the deployed equipment and parameter values in order to improve its overall performance.

The evolution from 3G to 4G has introduced not only the technology needed to increase data capacity and voice quality, but also a greater complexity in terms of network planning, deployment, and configuration, which have rendered some of the traditionally used methods to be ineffective. In a traditional approach (i.e. manual), during the network planning and maintenance processes, a radio-network simulation software executes the analysis, while an engineer makes the change decisions. Therefore, a radio-planning engineer adapts the network parameters manually and the network-planning tool analyzes the given configuration. If the obtained results are not acceptable, the analysis process has to be repeated several times, until a given goal is achieved.

The complexity of radio-network has grown even faster than their throughput capacity, thus making it impossible to plan 4G radio networks with traditional methods. In this sense, an examination of colored coverage maps in conjunction with some statistical analysis are no longer appropriate tools for troubleshooting a network. Moreover, since real-world 4G radio networks are large and many of their configuration parameters are interdependent, an engineer is not able to cope with the level of complexity present in such systems. For that reason, the computer, along with specialized software, guides the engineer to the most appropriate configuration for the network. In the context of this work, we will refer to this process as radio-network optimization.

Radio-network optimization may be divided into two fundamental phases: analysis and decision [35]. The analysis phase consists on the examination of network performance, which mainly focuses on the definition and collection of several Key Performance Indicators (KPIs). KPIs are quantifiable measurements, agreed to beforehand, that reflect network quality factors. The second phase deals with the decision making,

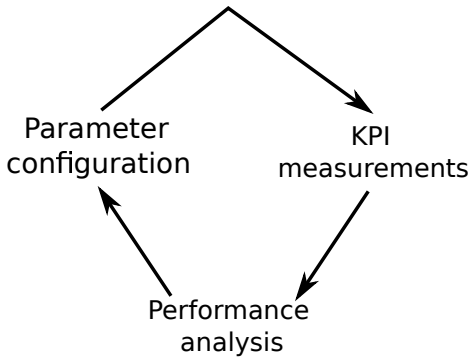


Figure 2.5: Typical optimization cycle for radio networks. This sequence is repeated until the achieved results are acceptable.

based on the analytical results collected in the previous phase, about the configuration of a particular configuration or parameter setting. This process, depicted in Figure 2.5, is repeated until the achieved results are acceptable.

A common limitation of several optimization methods appearing in the literature [ref!???] is their inability to meet the requirements needed by real-world radio networks, since their computational-time complexity make them unfeasible for practical use in industrial-sized scenarios. Particularly, this limitation is more common in implementations targeting traditional computer architectures of sequential execution [ref!???]. When analyzing real-world radio networks with thousands of transmitters it is necessary to reduce the execution time of the optimization processes, so that they are useful for practical use.

2.3 Survey of optimization problems for radio networks

Since radio networks are increasingly more sophisticated, the need for optimization methods, capable of copeing with greater complexity, is far from declining. It has been established that most radio-network optimization problems are NP-hard, since the computational time grows non-polynomially as the problem size increases [36–42]. Moreover, there are other reasons directly related with the evolution of already deployed networks that greatly increase the need for optimization methods, as described in [35]:

Network performance improvement more users receive service coverage with the same physical infrastructure, making parameter optimization the less expensive and only viable short-term approach.

Changes in users' profile the introduction of new and faster services puts additional stress on the infrastructure, requiring additional optimization efforts.

Changes in the propagation conditions the allocation of a different frequency band for LTE compared to UMTS requires deployment of 4G sites, which radio propagation behaves differently than in 3G networks, mostly in urban areas.

In this context, network operators define different optimization targets, depending on the addressed optimization problem. These targets are formed by an objective

function that maps possible configurations into a real value. This number represents a quality rating of the proposed solution and it is used to compare different solutions among each other, and ultimately select the best one. Unfortunately, there is no definitive objective function in the field of radio-network optimization [35]. However, it is possible to optimize for different targets such as service coverage, base station locations, interference levels, etc.

In this work, we will address network optimization methods that are performed "off-line", meaning that the optimization software is not an active functioning part of the radio network in operation. Statistical data about network operation is used as the input or feedback information for different optimization targets.

This paper gives an overview of well-known optimization problems in 3G mobile networks. At the beginning of each of the following sections, a description of an optimization problem is given, followed by a short survey of recently proposed optimization methods. Finally, a discussion about the introduced methods is given, before closing with some concluding remarks.

2.3.1 Optimizing base station locations

2.3.1.1 Problem formulation

Some references [43–45] formulate the base station location problem in terms of the minimum set covering problem (shown in Figure 2.6). The coverage problem is defined by considering the signal level in every test point from all base stations and requiring that at least one level is above a fixed threshold.

(a) (b)

Figure 2.6: The minimum set covering problem: (a) the problem input and (b) the solution.

A different formulation considers the site selection problem as a p-median problem, in which base station location is the only decision variable considered. To each of the candidates solutions, an installation cost is also associated. The p-median problem constitutes seeking p different locations each time, regardless of how distant the sites are. The problem is to select one candidate site from each region to install a base station such that the traffic capacity and the size of the covered area are maximized with the lowest installation cost.

2.3.1.2 Proposed solutions

Aydin et al. [46] propose a solution to the p-median problem based on three meta-heuristic algorithms; a genetic algorithm, simulated annealing, and tabu search. Their

experimental study focuses on performance comparison between the three algorithms.

A solution to the set covering problem is proposed by Hao et al. [43]. A simulated annealing implementation was developed to solve the formulated combinatorial problem. The results presented demonstrate the feasibility of the proposed approach. Tutschku [44] presents a specialized greedy algorithm to solve the same problem. This work is part of the implementation of a planning tool prototype. Mathar and Niessen [45] propose a solution based on integer linear programming, which they claim finds optimal solutions in most cases. They also introduce simulated annealing as an approximate optimization technique. This approach substitutes linear programming whenever an exact solution is out of reach because of the complexity of the problem.

Amaldi et al. [**?**] offer a discussion about the computational results of two different heuristics: greedy search and tabu search. The problem formulation is based on a set of candidate sites where the base stations can be installed, an estimation of the traffic distribution and a propagation description of the area to be covered. Some years later, the same authors [**?**] extended the problem formulation by also considering base station configuration and hardware characteristics. In both works, they propose a mixed integer programming model with which they aim to maximize the trade-off between total traffic covered and total installation costs. The only difference between the models is the constraint definition of the linear program, where the constraints in [**?**] are a subset of the constraints in [**?**].

Finally, Whitaker et al. [47] focus on providing the required service coverage at the lowest possible financial cost. Their framework supports the use of any multiple objective optimization algorithm which seeks to approximate a Pareto front. The performance of four different algorithms is explored, namely SEAMO, SPEA2, NSGA-II and PESA.

2.3.2 Optimizing antenna parameters

There are many antenna parameters that control the coverage and interference in the network, since the antenna shapes the emitted energy. Two important parameters are the azimuth angle and the elevation angle (or tilt) of the antenna. The antenna azimuth (shown in Figure 2.7) is the direction in which the main beam of the horizontal pattern points [48]. The antenna tilt (shown in Figure 2.8) is defined as the angle of the main beam of the antenna relative to the horizontal plane [48]. Both of these parameters have a great influence on network quality, although antenna tilt requires less effort to implement, since most modern radio networks already support remote electrical tilt. The adjustment of these two parameters optimize some important aspects of the network, namely:

- path loss between the base station and the mobile phone, since less power is required for a connection, hence more power is available for traffic; and
- interference between neighboring cells, which leads to an overall capacity increase.

2.3.2.1 Proposed solutions

Karner [49] proposes an "ad-hoc" strategy for adjusting antenna azimuth and downtilt by analyzing the structure of the network. The objective of this optimization is to

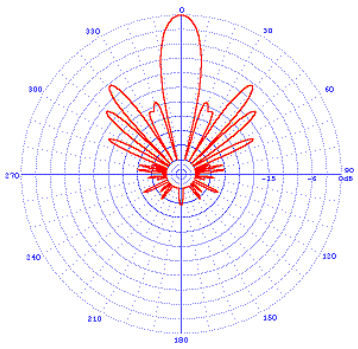


Figure 2.7: A typical antenna azimuth pattern.

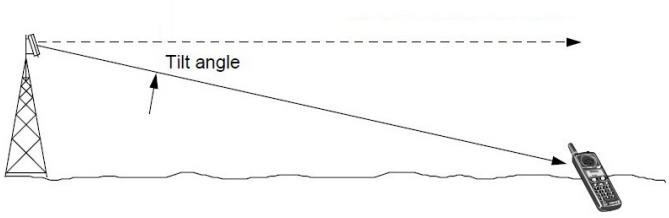


Figure 2.8: The antenna tilt angle with the horizontal plane.

improve the results presented in [50] by increasing the number of served users in the target area. In a similar line of work, Jakl [51] included in his doctoral thesis an antenna azimuth optimization algorithm, based on attempts of avoiding coverage holes by properly adjusting the azimuth settings.

Siomina and Yuan [52] propose a framework for automated optimization of antenna azimuth and tilt, including both mechanical and electrical tilt. The implementation introduces a simulated annealing algorithm that searches the solution space of possible antenna configurations. The goal of the optimization is targeted to address power sharing among cell channels and ultimately improve High-Speed Downlink Packet Access (HSDPA) [53] throughput.

Zhang et al. [54] present a method which is composed of two optimization loops: the inner one and the outer one. The inner loop concentrates on frequency planning while the outer loop focuses on finding the optimal setting of antenna azimuth and tilt for the current solution delivered by the inner loop. Although frequency planning is not directly related to 3G network optimization, we found this approach interesting enough to make a reference to it. The inner loop could be easily replaced with some other optimization target, e.g. common pilot channel power setting.

2.3.3 Optimizing common pilot channel power

The Common Pilot Channel (CPICH) is used as reference for handover [55], cell selection [56], and cell reselection [57]. Whenever a mobile phone is switched on, it tries to register with the cell providing the highest received CPICH level. It also defines the effective coverage area of the cell: according to the the CPICH power level, the cell coverage area will enlarge or shrink. Consequently, by appropriately adjusting the CPICH power at the base stations, the number of served users per cell can be balanced among neighboring cells. This procedure is called load balancing and it reduces interference, stabilizes network operation, and facilitates radio resource

management [48].

2.3.3.1 Proposed solutions

Chen and Yuan [?] optimize CPICH transmit power starting from a uniform allocation (i.e. all cells are set with the same transmit power level). Their objective is to enhance HSDPA performance at cell edges while preserving control of R99 soft handover [58]. The solution approach is based on a linear-integer mathematical model. A very similar problem and proposed solution is presented by Siomina and Yuan [59]. The problem definition slightly differs from [?] as there is no reference to soft handover control. The solution is also implemented as a linear program.

Olmos et al. [60] apply simulated annealing to the optimization of CPICH transmit powers in order to force mobile phones to transmit to the best available cell in a service area. Their results show decreased cell load with a consequently increased network capacity.

Chen and Yuan [?] present a two-phase optimization algorithm for large-scale mobile networks. The algorithm uses the total CPICH power consumption as a minimization objective. The authors claim that the tabu-search-based algorithm can compute near optimal solutions within a few seconds, even for large networks.

2.3.4 Optimizing CPICH and antenna parameters

Both the antenna tilt and azimuth directly affect the direction and range in which the cell broadcasts its CPICH. Consequently, optimal CPICH power is highly dependent on how the antenna tilt and azimuth are configured at base stations. Ideal configuration of antenna tilt and azimuth network-wise, with the objective of optimizing the CPICH power consumption, is a challenging task. For this reason, many authors have considered optimization methods that address all three parameters.

2.3.4.1 Proposed solutions

Varbrand et al. [61] approach the optimization of service coverage (see section 2.3.5) by considering all three configuration parameters, namely: CPICH transmit power, antenna tilt, and antenna azimuth. Their simulated annealing algorithm searches the solution space of possible configurations in order to find improvements in network performance and total transmitted power. Interestingly, the algorithm is efficient enough to optimize large networks without using excessive computing resources.

Siomina [62] combines optimization of base station antenna tilts and CPICH power to reduce the total interference level and to improve network capacity. The introduced algorithm optimizes the antenna downtilt setting so that the total CPICH power in the network is minimized.

Neubauer et al. [50] present two optimization algorithms for finding an optimal setting of antenna tilt and CPICH power of the base stations. The first algorithm builds on a rule-based approach, while the second one extends it by incorporating simulated annealing. An evaluation of both techniques shows that the second algorithm returns better results.

Jakl et al. [63] proposed a problem-specific genetic algorithm to tackle the optimization of antenna tilts and CPICH transmit powers. The goal of the optimization is

to increase network capacity. The implementation involves a deterministic fitness selection scheme, a problem specific recombination operator and an improved mutation operator. After the initial identification of the best individuals, a local optimization technique is used to improve their fitness.

2.3.5 Optimizing coverage

Coverage is maybe the most common optimization objective considered in 3G network optimization. The objective function for coverage optimization may be defined as follows:

$$f_{\text{cov}} = \frac{A_{\text{covered}}}{A_{\text{total}}}$$

where A_{covered} represents the area covered by the network and A_{total} represents the total area under optimization. Thus, the expression f_{cov} represents the portion of the total area that is actually under network coverage. This value ranges from 0 (no coverage) to 1 (total coverage).

The area being optimized is usually divided in squares (or pixels) of a certain size, creating a regular square grid (RSG) of a certain resolution. A pixel is considered covered if the signal to noise ratio is above a given threshold [64]. It is also common to use a test function $cov(x, y)$, which returns 1 if the pixel located at (x, y) is covered, and 0 otherwise.

2.3.5.1 Proposed solutions

Siomina and Yuan [65] consider the problem of minimizing pilot power subject to the coverage constraint. Their approach consists of mathematical programming models and methods, based on a linear-integer mathematical formulation of the problem. A special numerical analysis studies the trade-off between service coverage and pilot power consumption for different test networks.

Capone et al. [**?**] investigate mathematical programming models for supporting decisions on where to install new base stations and how to select their configuration (antenna height and tilt, sector orientations, maximum emission power, pilot signal, etc.) so to find a trade-off between maximizing coverage and minimizing costs. The overall model takes into account signal-quality constraints in both uplink and downlink directions, as well as the power control mechanism and the CPICH signal.

Valkealahti et al. [66] propose a method for automatic setting of CPICH power. The control algorithm applies total transmission power measurements from the base station, neighboring cells and mobile terminals to determine the pilot qualification. The CPICH power is then periodically updated based on a group of heuristic rules in order to improve coverage and load balance. A similar approach is described by Parkinnen et al. [67] where some network performance parameters are combined with service coverage in a cost function. The pilot power of a cell is periodically updated with a gradient descent method that minimizes the afore mentioned function.

2.3.6 Optimizing alignment of soft-handover areas

We have defined this new problem. About the connecting of this 3G-specific problem with LTE ...???

2.3.7 Discussion

In the field of 3G network optimization, comparing the efficiency of different optimization methods has proven to be a very difficult task. There are several reasons for this, among which we have identified the following:

- The networks on which the experiments are carried out are not standardized. Thus, it is problematic to set up an environment in which the presented results may be reproduced.

- There are no "open" implementations of 3G mobile network simulators. On the contrary, there is a vast variety of proprietary (or "closed") network simulators that only increase the ambiguity regarding the experimental environment used for a certain network layout and configuration. Moreover, some of the "closed" network simulators do not even account on the built-in path loss estimation methods used.

- Only a small part of the referenced optimization methods perform their experiments on real-life networks, that have been already deployed and are currently running. This fact creates a gap between the research field and the industry (i.e. the application area) and it is counterproductive for both the researchers and the network operators, since they don't benefit from mutual collaboration.

We believe that the creation of a standardized framework would be a great benefit in the field of 3G network optimization, as it would allow researchers to compare different methods, results and solutions in an easy, fast and objective manner. An effort in this direction is the MOMENTUM project [68]. It was created with the objective of setting up a standardized experimental environment that would facilitate research cooperation and would ultimately benefit from an improvement on the research field of 3G networks. The project includes complete data about the terrain, properties of the service area and hardware used in three different mobile networks. It offers the researcher detailed information about real-life networks based in Berlin, Lisbon and The Hague. As part of the package, there is also a Java application programming interface (API) that eases the implementation of some conventional tasks such as data parsing. The main focus of project is on data, thus there is no network simulator included. In any case, the researcher benefits from several included traffic snapshots that help assessing different network configuration settings. There have been no updates in the MOMENTUM project since 2005, when the last data corrections were introduced [68].

In the context of this survey and after reviewing many different papers on 3G network optimization, we can say that the MOMENTUM project has not been widely adopted by the scientific community. Moreover, several of the reviewed works do not offer detailed instructions on how to resemble the experimental environment. Consequently, it is very complicated (if not even impossible) to reproduce the presented

Table 2.2: A comparison among the presented optimization methods.

Algorithm	Running time	Typical application
Local search	Shorter	Solution quality improvement.
Tabu search	Shorter	Solution quality improvement.
Simulated annealing	Longer	Initial search of the solution space.
Genetic algorithm	Longer	Initial search of the solution space and solution quality improve
Linear programming	(formulation dependent)	Coverage network planning.

results. Therefore, the selection of optimization methods, based on the results presented by their corresponding authors, is virtually meaningless, since the results are not comparable with each other. For this reason, we have decided to concentrate the following discussion on the optimization methods used, leaving the results aside.

2.3.8 Discussion about the presented methods

Regarding the optimizations methods presented in previous chapters, three distinctive groups emerge: genetic algorithms, linear programming and other search methods.

Genetic algorithms These algorithms work on a population of solutions that allows a more comprehensive search for optimal solutions. As a direct consequence, an increase in running time is commonly observed. The implementation effort of genetic algorithms is to some degree higher than for simpler search methods (e.g. local search), but their inherent structure greatly simplifies possible parallel implementations and execution.

Linear programming Linear optimization problems are widely used in different optimization areas and there are many good software packages to solve such problems. Consequently, if a problem can be modeled as a continuous linear problem, there is usually no difficulty in finding optimality. In the context of this survey, linear programming has proven useful for coverage optimization in early network planning stages.

Other search methods Other search methods[1] usually represent a compromise between running time and quality of results. They rely on evaluating a great number of alternative configurations. The number of parameters taken into account, as well as the evaluation precision, directly influence their running time. These methods don't excel in full simulation scenarios. On the other hand, some search methods (e.g. tabu search) have powerful mechanisms to escape local minima.

A short comparison of the optimization methods presented in this survey is shown in Table 2.2. The comparison variables arise from the context in which the methods were presented.

[1]Namely, local search, simulated annealing and tabu search in the context of this survey.

Summary

The variety of optimization problems that have been introduced in the previous sections differ in many aspects like implementation, running time and solution quality. Picking the right method for a given situation depends on the optimization task and the desired results. Since computation time is usually an important restriction, simpler and faster methods may be preferable.

Beside the convenience of a survey such as the one presented in this work, it is very important to develop a feeling for the properties, advantages and drawbacks of the respective methods. Moreover, radio expert's recommendations regarding solution interpretation and feedback from everyday network operation, are an essential input for creating quality optimization methods. In this sense, and based on our own experience, the expert's advice is irreplaceable and a most valuable contribution to the research work.

Also, as it may be observed in some of the presented references, it is often advisable to combine different methods. Therefore, a simple optimization method may find a subset of reasonable parameter configuration, whereas a more complex method could be applied afterward, to refine the search. Sometimes it may also be useful to apply a simple search method at the end to find better solutions in the vicinity of a current promising one.

2.4 Principles of mobile radio networks

2.4.1 Quality of service

QoS...???

2.4.2 Handover and soft-handover

In mobile networks, handover is one of the main features that allows user's mobility [48]. The concept behind the handover operation is simple: when a user moves from the coverage area of a cell to the coverage area of a neighboring cell, the system creates a new connection with the latter cell and disconnects the user from the former one, while keeping the current connection active. Soft-handover (SHO), on the other hand, is a possibility available in mobile networks using the Wideband Code Division Multiple Access (WCDMA) technology, which the UMTS employs. SHO enhances handover functionality by allowing a user to potentially operate on multiple radio links in parallel. Since different users are separated by unique spreading codes, the detection of single user's signal is implemented by despreading with the same code sequence used in the transmitter [48].

Every mobile terminal constantly monitors the common pilot power channel (CPICH) of the connected cell and its neighbors. The information about these measurements is sent to the network by the user terminal (i.e. mobile). The SHO condition depends on the relative received signal quality from different cells and the SHO window, which triggers the addition of a cell to the user's active set. Depending on radio propagation characteristics and different transceiver capabilities, the radio transmission can gain more than 3 dB out of a SHO situation [48]. From this point of view, SHO is a method

to reduce interference and improve radio quality, particularly at the cell border where radio coverage is of inferior quality. In UMTS Release 99 [69], SHO is specified to work from the network towards the user (i.e. downlink), and from the user towards the network (i.e. uplink).

With the introduction of High Speed Packet Access (HSPA) as an improvement of the performance existing in WCDMA protocols, the role SHO plays in mobile network configuration and functioning slightly changed. The key difference is that High Speed Downlink Packet Access (HSDPA) does not support SHO, whereas the High Speed Uplink Packet Access (HSUPA) does. This particular distinction is discussed in Chapter 5, since it has some key implications in the balanced distribution of SHO areas, and thus in the quality of HSPA services [70].

2.4.3 Pilot signal and power

The CPICH transmit power is typically between 5% to 10% of the total downlink transmit power of the base station [71], but there is no standardized method to find a CPICH power setting.

The CPICH transmits in the downlink of a UMTS cell system. The transmit power is usually between 5% and 10% of the total power available at the base station [72]. The capacity of a cell is limited by the amount of available power at the base station and the interference level at the mobile terminal. The coverage area of any cell is controlled by changing its pilot power, which consequently modifies the service area of the network.

The CPICH transmit power is common to many different planning and optimization problems in UMTS networks [73].

2.5 Principles of parallel systems/implementations

2.5.1 OpenCL

We have chosen the Open Computing Language (OpenCL) [74] as the implementation platform of our optimization system on GPU.

OpenCL is an open parallel computing API designed to enable GPUs and other coprocessors to work together with the CPU, providing additional computing power. As a standard, OpenCL 1.0 was released in 2008, by The Khronos Group, an independent standards consortium [75]. For additional information about the OpenCL standard and API, we refer the reader to the numerous guides available online.

Our choice in using OpenCL was greatly influenced by the fact that its bitcode runs on a variety of hardware, including multicore CPUs and GPUs from different vendors. This provides a complete framework capable of comparing execution speedup on different hardware without the need of changing the implementation.

One unfortunate consequence of the vendor variety is that NVIDIA's CUDA [76] and OpenCL documentation present disparate naming conventions for some key components. For the sake of consistency, in Table 2.3, we present a short "translation dictionary" between them. In the remaining of this work, we will stick to the naming convention used in the CUDA documentation.

Table 2.3: *Terminology translation between OpenCL and CUDA [77].*

OpenCL	**CUDA**
Grid	Grid
Work group	Block
Work item	Thread
__kernel	__global__
__global	__device__
__local	__shared__
__private	__local__
image*n*d_t	texture<type,*n*,...>
barrier(L|M|F)	__syncthreads( )
get_local_id(0|1|2)	threadIdx.x|y|z
get_group_id(0|1|2)	blockIdx.x|y|z
get_global_id(0|1|2)	*(not implemented)*

Despite the use of OpenCL as the target platform for our implementation, the details described in the next sections may be equally applied on CUDA.

3 Framework design and implementation

There is a constant growing demand for hardware resources, longer-processing times and more memory to follow the evolution of 3G radio networks [78–80]. Fortunately, high-performance computer systems are increasingly accessible; something made possible because of the emergence of computer clusters and commodity hardware, capable of true parallel processing, e.g. multi-core CPUs [81] and GPUs [82]. Moreover, the highly parallel structure present on GPUs makes them more effective than CPUs for execution of algorithms where large blocks of data need to be processed in parallel. Commodity GPUs have evolved from being a graphic accelerator into a general-purpose processor. They can achieve higher performance at lower power consumption and lower costs when compared to conventional CPUs. Additionally, the implementation of the framework will benefit from valuable advances in computer science and High Performance Computing (HPC), in order to perform faster and more reliable simulations [81, 82].

More than 20 years have passed since the world's first GSM mobile call was made in Finland. Still, the coverage planning of the radio network remains a key problem that all mobile operators have to deal with. Moreover, it has proven to be a fundamental issue not only in GSM networks, but also in modern standards such as the third generation (3G) UMTS and the fourth generation (4G) LTE Advanced [**?**,**?**,65,83]. In radio networks is generally the case that the radio stations are installed at fixed locations. For this reason, one of the primary objectives of mobile-network planning is to efficiently use the allocated frequency band to assure that the whole of the geographic area of interest can be satisfactorily reached with the radio stations of the network. To this end, radio-coverage prediction tools are of great importance as it allows the network engineers to test different network configurations before physically implementing the changes. Nevertheless, radio-coverage prediction is a complex task due to the wide range of various combinations of hardware and configuration parameters which have to be analyzed in the context of different environments. The complexity of the problem means that radio-coverage prediction can be a computationally-intensive and time-consuming task, hence the importance of fast and accurate prediction tools.

Although different mathematical models have been proposed for radio propagation modeling, none of them excels in a network-wide scenario [**?**]. A combination of different models and parameters is generally needed in order to calculate radio-propagation

predictions for particular environments. Moreover, since the number of deployed cells (transmitters) keeps growing with the adoption of modern standards [**?**], there is a clear need for a radio propagation tool that is able to cope with larger work loads in a feasible amount of time.

Despite various options of commercial tools specialized in radio-propagation modeling, the common thread among them is the restricted nature of its usage, mostly dominated by black-box implementations. This fact induces lack of adaptability, sometimes even combined with cumbersome user interfaces that are not suitable for big batch jobs, involving thousands of transmitters. Moreover, the evolution of any commercial tool is strictly bounded to its vendor, forcing the user to adapt its workflow to it, when the opposite situation should be preferred.

To tackle the afore-mentioned issues, we present a high-performance parallel radio-prediction tool for the open source Geographic Resources Analysis Support System (GRASS). For its design, we have focused on scalability, clean design and open nature of the tool, inspired by the GRASS geographic information system (GIS). These facts make it an ideal candidate for calculating radio-predictions of big problem instances, i.e. real mobile networks containing thousands of transmitters. This is also true for the scientific research community, since our design may be used as a template for parallelization of computationally-expensive tasks within the GRASS environment.

3.0.2 Parallel computation on computer clusters

In consideration of the high computational-intensity of predicting the radio-coverage of a real mobile network, the use of a computer cluster is required, i.e. a group of interconnected computers that work together as a single system. To reach high levels of parallel performance and scalability, this work discusses in detail the key steps of parallel decomposition of the radio-coverage prediction problem for real networks and the distribution of the computational load among the computing nodes that belong to the cluster.

Such computer clusters typically consist of several commodity PCs connected through a high-speed local network with a distributed file system, like NFS [**?**]. One such system is the DEGIMA cluster [**?**] at the Nagasaki Advanced Computing Center of the Nagasaki University. This system ranked in the TOP 500 list of supercomputers until June 2012[1], and in June 2011 held the third place of the Green 500 list[2] as one of the most energy-efficient supercomputers in the world.

3.0.3 Objectives

The main goal of this work is to develop a radio prediction tool to be used in large real-world network environments, such as the ones currently deployed by several mobile operators around the world. To achieve this, we have developed a high-performance parallel radio prediction tool (PRATO) for radio networks. Therefore, our focus is on the performance and scalability of PRATO, while other more dynamic aspects of radio networks are not considered. Among these aspects are code distributions, details of (soft) handover, and dynamics related to radio resource management.

[1] http://www.top500.org
[2] http://www.green500.org

The performance evaluation of PRATO in a distributed computing environment is a major objective of this work. Furthermore, by presenting a detailed description of the design and implementation of the parallel version of PRATO, we intend to provide guidelines on how to achieve high efficiency levels of task parallelization in GRASS GIS. Additionally, we introduce techniques to overcome several obstacles encountered during our research as well as in related work, which significantly improve the quality and performance of the presented implementation, e.g. the inability to use GRASS in a threaded environment, lowering overhead of I/O operations, saving simulation results asynchronously and independently from GRASS, and improving load balancing with a new message-passing technique.

The paper is organized as follows. Section 3.1 gives a description of the radio prediction tool, including the propagation model and GRASS GIS. Section 3.2 concentrates on the design principles and implementation details of the radio propagation tool, for the serial and parallel versions. Section 3.3 discusses the experimental results and their analysis. Finally, Section 6.6 gives an overview of relevant publications, describing how they relate to our work, before drawing some conclusions.

3.1 Description of the radio coverage prediction tool

PRATO is a high-performance radio-prediction tool for GSM (2G), UMTS (3G) and LTE (4G) radio networks. It is implemented as a module for the GRASS Geographical Information System (for details of GRASS see Section 3.1.2). It can be used for planning the different phases of a new radio-network installation, as well as a support tool for maintenance activities related to network troubleshooting or upgrading.

As a reference implementation, we have used the publicly available radio coverage prediction tool, developed by Hrovat et al. [84]. The authors of this work have developed a modular radio coverage tool that performs separate calculations for radio-signal path loss and antenna radiation patterns, also taking into account different configuration parameters, such as antenna tilting, azimuth and height. The output result, saved as a raster map, is the maximum signal level over the target area, in which each point represents the received signal from the best serving cell (transmitter). This work implements some well-known radio propagation models, e.g. Okumura-Hata and COST 231, the later is explained in more detail in Section 3.1.1. Regarding the accuracy of the predicted values, the authors report comparable results to those of a state-of-the-art commercial tool. Therefore, we use the implementation developed by [84] as the reference implementation for PRATO. Furthermore, to ensure that our implementation is completely compliant with the afore-mentioned reference, we have designed a comparison test that consists of running both the reference and PRATO with the same input parameters. The test results from PRATO and the reference implementation are identical.

3.1.1 Propagation modeling

The COST-231 Walfisch-Ikegami radio-propagation model was introduced as an extension of the well-known COST Hata model [**?**, **?**], designed for frequencies above 2000 MHz. The suitability of this model comes from the fact that it distinguishes

between line-of-sight (LOS) and non-line-of-sight (NLOS) conditions. Equation (3.1) describes the path loss when there is LOS between the transmitter and the receiver.

$$PL_{\mathrm{LOS}}(d) = 42.64 + 26\log(d) + 20\log(F), \tag{3.1}$$

where d is the distance (in kilometers) from the transmitter to the receiver point, and F is the frequency, expressed in MHz.

On the other hand, in NLOS conditions, the path loss is calculated as

$$PL_{\mathrm{NLOS}}(d) = L_0 + L_{\mathrm{RTS}} + L_{\mathrm{MSD}}, \tag{3.2}$$

where L_0 is the attenuation in free space, L_{RTS} represents the diffraction from roof top to street, and L_{MSD} represents the diffraction loss due to multiple obstacles.

In this work, as well as in the reference implementation [84], the terrain profile is used for LOS determination. The wave-guide effect in streets of big cities is not taken into account, because the building data is not available. In order to compensate the missing data, we include a correction factor, based on the land usage (clutter data). This technique is also adopted by other propagation models for radio networks, like the artificial neural networks macro-cell model developed by Neskovic et al. [**?**]. Consequently, both Equations (3.1) and (3.2) have an extra term for signal loss due to clutter (L_{CLUT}), thus redefining the LOS and NLOS path losses as

$$PL_{\mathrm{LOS}}(d) = 42.64 + 26\log(d) + 20\log(F) + L_{\mathrm{CLUT}} \tag{3.3}$$

and

$$PL_{\mathrm{NLOS}}(d) = L_0 + L_{\mathrm{RTS}} + L_{\mathrm{MSD}} + L_{\mathrm{CLUT}}. \tag{3.4}$$

3.1.2 GRASS Geographical Information System

As the software environment for PRATO we have chosen GRASS (Geographic Resources Analysis Support System) [**?**], which is a free and open-source software project that implements a Geographical Information System (GIS). This GIS software was originally developed at the US Army Construction Engineering Research Laboratories and is a full-featured system with a wide range of analytical, data-management, and visualization capabilities. Currently, the development of GRASS GIS is supported by a growing community of volunteer developers.

The use of GRASS GIS as an environment for PRATO presents many advantages. First, the current development of GRASS is primarily Linux-based. Since the field of high performance computing is dominated by Linux and UNIX systems, an environment with Linux support is critical for this work. Software licensing is another important consideration for choosing GRASS, since it is licensed under the GNU Public License [**?**] and imposes the availability of the source code. This allows us to make potential modifications to the system, thus adapting it for the parallel computation environment. Moreover, being an open system, GRASS provided us with a great deal of useful built-in functionality, capable of operating with raster and vector topological data that can be stored in an internal format or a relational database. For additional information about the GRASS, we refer the reader to the numerous guides and tutorials available online.

3.2 Design and implementation

3.2.1 Design of the serial version

This section describes the different functions contained in the serial version of PRATO, which is implemented as a GRASS module. Their connections and data flow are depicted in Figure 3.1 on page 36, where the parallelograms in the flow diagram represent input/output (I/O) operations.

Our design follows a similar internal organization as the radio planning tool developed by Hrovat et al. [84], but with some essential differences. Specifically, we have decided to avoid the modular design to prevent the overhead of I/O operations for communicating data between the components of the modular architecture. Instead, we have chosen a monolithic design, in which all the steps for generating the radio coverage prediction are calculated inside one GRASS module. Regarding the way results are saved, our approach employs a direct connection to an external database server, instead of the slow built-in GRASS database drivers. To explicitly avoid tight coupling with a specific database vendor, the generated output is formatted in plain text, which is then forwarded to the database server. Any further processing is achieved by issuing a query over the database tables that contain the partial results for each of the processed transmitters.

3.2.1.1 Read input parameters

All input data are read in the first step (see "Read input data" in Figure 3.1 on page 36), e.g. digital elevation model, clutter data, transmitter configurations, and other service-dependent settings. Their format differs based on the data they contain, namely:

- GRASS raster files are used for the digital elevation model and clutter data, whereas
- a text file is used for the transmitter configurations and other simulation-dependent options.

Since the module accepts a considerable amount of input parameters, they are read from a text-based initialization (INI) file. This is far more practical than passing them as command-line parameters, which would make them error-prune and difficult to read. Besides, the INI file may contain configuration parameters for many transmitters. The user selects which one(s) to use at run-time by passing a command-line option.

3.2.1.2 Isotropic path-loss calculation

The first step here is to calculate which receiver points, r, are within the specified transmission radius (see "transmission radius" in Figure 3.1 on page 36). For these points, the LOS and NLOS conditions are calculated, with respect to the transmitter (see "Calculate LOS/NLOS" in Figure 3.1 on page 36). The following step consists of calculating the path loss for an isotropic source (or omni antenna). This calculation is performed by applying the COST-231 path-loss model, which was previously introduced in Section 3.1.1, to each of the points within the transmission radius around the

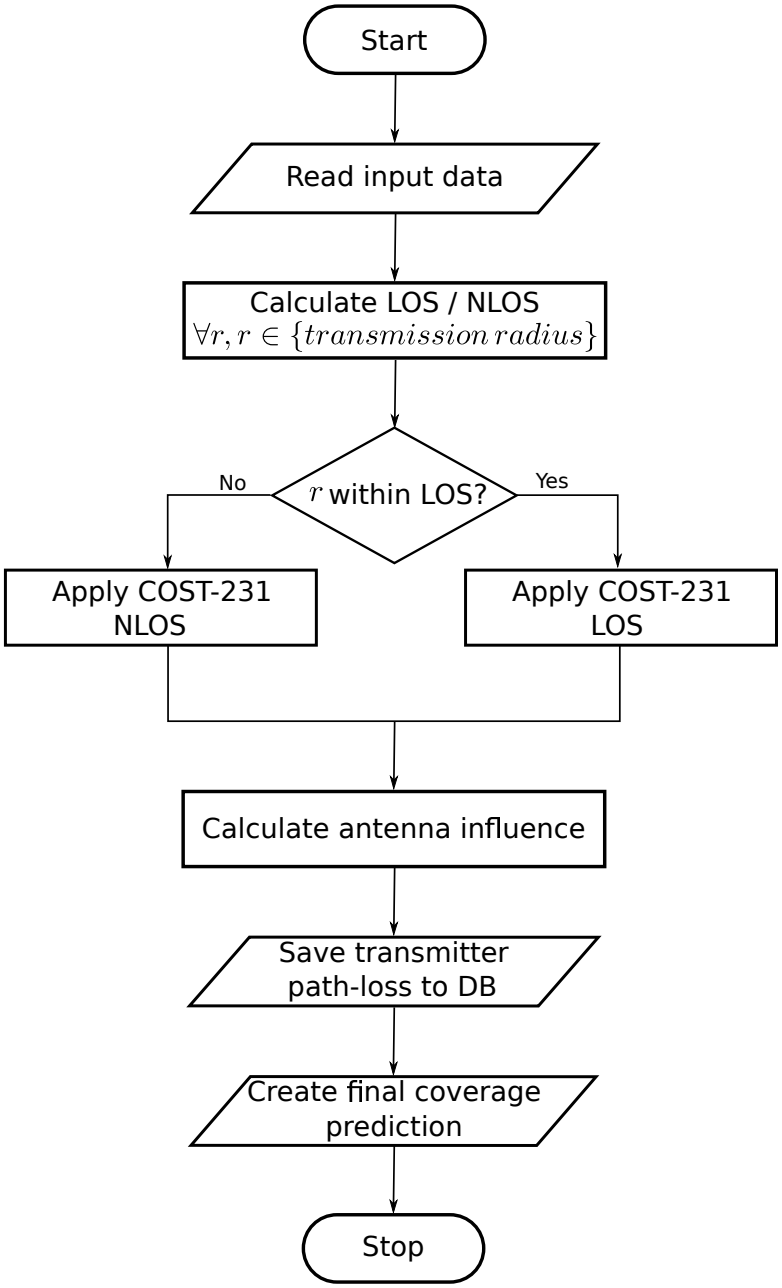


Figure 3.1: Flow diagram of the serial version.

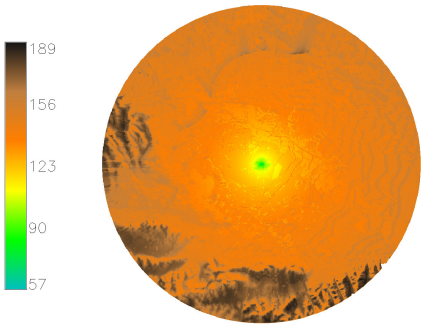


Figure 3.2: Example of raster map, showing the result of a path-loss calculation from an isotropic source.

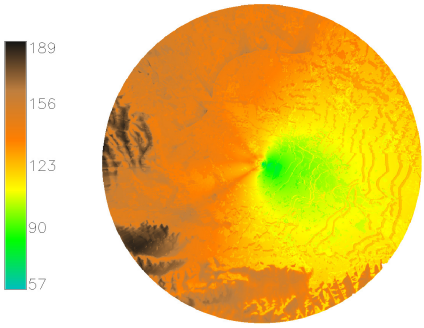


Figure 3.3: Example of raster map, showing the antenna influence over the isotropic path-loss result.

transmitter. Depending on whether the receiver point r is in LOS or NLOS, either Equation (3.3) or Equation (3.4) is respectively applied (see "Apply COST-231, LOS" or "Apply COST-231, NLOS" in Figure 3.1 on page 36).

Figure 3.2 on page 36 shows a portion of a raster map with an example result of the isotropic path-loss calculation. The color scale is given in dB, indicating the signal loss from the isotropic source, located in the center. Also, the hilly terrain is clearly distinguished due to LOS and NLOS conditions from the signal source.

3.2.1.3 Antenna diagram influence

This step considers the antenna radiation diagram of the current transmitter and its influence over the isotropic path-loss calculation (see "Calculate antenna influence" in Figure 3.1 on page 36). Working on the in-memory results generated by the previous step, the radiation diagram of the antenna is taken into account, including beam direction, electrical and mechanical tilt. Figure 3.3 on page 36 shows a portion of a raster map, where this calculation step has been applied to the results from Figure 3.2 on page 36. Notice the distortion of the signal propagation that the antenna has introduced.

3.2.1.4 Transmitter path-loss prediction

In this step, the coverage prediction of the transmitter is saved in its own database table (see "Save transmitter path-loss to DB" in Figure 3.1 on page 36), thus considerably enhancing the write performance during the result-dumping phase, which involves saving the path-loss results. This is accomplished by connecting the standard output of the developed module with the standard input of a database client. Naturally, the generated plain text should be understood by the database server itself.

3.2.1.5 Coverage prediction

The final radio coverage prediction, containing an aggregation of the partial path-loss predictions of the involved transmitters, is created in this step (see "Create final coverage prediction" in Figure 3.1 on page 36). The received signal strength from each of the transmitters is calculated as the difference between its transmit power and path loss for the receiver's corresponding position. This is done for each point in the target area by executing an SQL query over the tables containing the path-loss predictions of each of the processed transmitters.

Finally, the output raster is generated, using the GRASS built-in modules *v.in.ascii* and *v.to.rast*, which create a raster map using the results of the above-mentioned query as input. The raster map contains the maximum received signal strength for each individual point, as shown in Figure 3.4 on page 38. In this case, the color scale is given in dBm, indicating the received signal strength from the transmitters.

3.2.2 Multi-paradigm parallel programming

The implementation methodology adopted for PRATO follows a multi-paradigm parallel programming approach in order to fully use the resources of a computing cluster. To effectively use a shared memory multi-processor, PRATO uses POSIX threads to

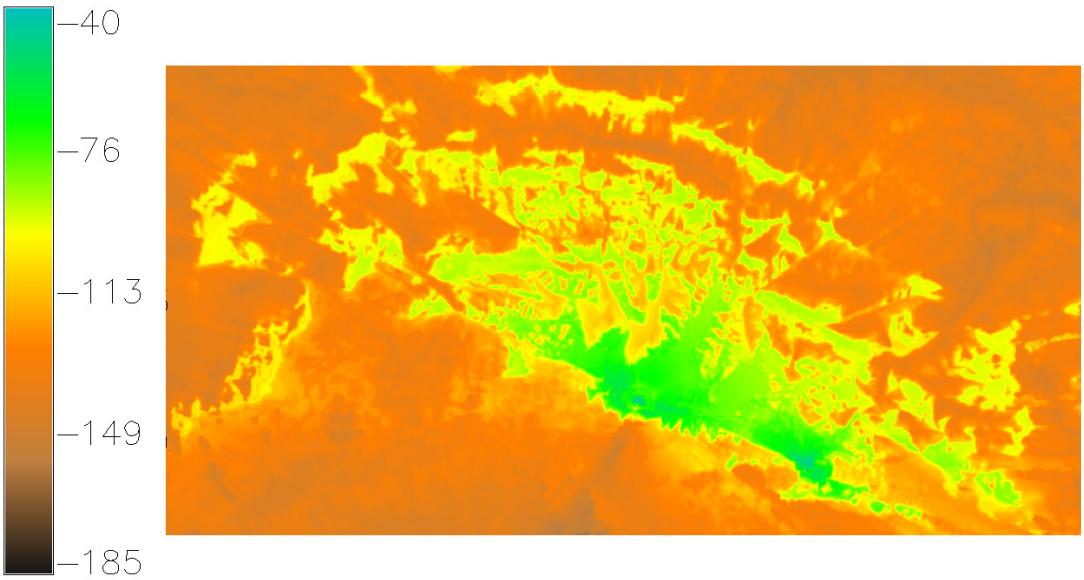


Figure 3.4: Example of raster map, displaying the final coverage prediction of several transmitters. The color scale is given in dBm, indicating the received signal strength.

implement parallelism [?]. In a nutshell, POSIX thread is a POSIX standard for creating and manipulating light-weight processes or threads. By using POSIX threads, multiple threads can exist within the same process while sharing its resources. For instance, an application using POSIX threads can execute multiple threads in parallel by using the cores of a multi-core processor, or use the system resources more effectively, thus avoiding process execution-halt due to I/O latency by using one thread for computing while a second thread waits for an I/O operation to complete.

To use the computing resources of a distributed memory system, such as a cluster of processors, PRATO uses the Message Passing Interface (MPI) [?]. MPI is a message-passing standard which defines syntax and semantics designed to function on a wide variety of parallel computers. MPI enables multiple processes running on different processors of a computer cluster to communicate with each other. MPI was designed for high performance on both massively parallel machines and on workstation clusters. It has been developed by a broadly based committee of vendors, developers, and users.

In order to make the text more clear and to differentiate between the programming paradigms used from here on, we will refer to a POSIX thread simply as a 'thread' and a MPI proccess as a 'process'.

3.2.3 Design of the parallel version

Keeping our focus on the performance of PRATO, we are introducing a new distributed implementation to overcome computational-time constraints that prevented the reference implementation from tackling big problem instances [84].

Some authors have already published their work on implementing parallel versions of GRASS modules for solving different time-consuming tasks [?, ?, ?]. However, one major drawback of GRASS as a parallelization environment is that it is not thread-safe, meaning that concurrent changes to a data set have undefined behavior. To overcome this problem, we present a technique that saves the simulation results asyn-

chronously and independently from the GRASS environment, e.g. into an external database system. This database system works also as an input source, serving data to GRASS, whether it is used to aggregate the partial results of the path-loss prediction or to visualize them. We also introduce a methodology that allows the parallel implementation to be almost completely GRASS independent. This means that a GRASS installation is needed on only one of the nodes, i.e. the master node of the target computer cluster. Also, a message-passing technique is proposed to distribute the work-load among nodes hosting the worker processes. Using this technique, computing nodes featuring more capable hardware receive more work than those with weaker configurations, thus ensuring a better utilization of the available computing resources despite hardware diversity.

3.2.3.1 Master process

As it has been suggested before, the parallel version of PRATO follows a master-worker model. The master process, for which the flow diagram is given in Figure 3.5 on page 40, is the only component that should be run from within the GRASS environment. As soon as the master process starts, the input parameters are read. This step corresponds to "Read input data" in Figure 3.5 on page 40, and it is done in a similar way as in the serial version. In the next step, the master process dynamically initiates the worker processes using the available computing nodes (see "Dynamic worker-process spawning" in Figure 3.5 on page 40), based on the amount of transmitters for which the coverage prediction should be calculated. In other words, this means that master process never starts more worker processes than there are transmitters to be processed. However, most often is the number of transmitters larger than the amount of available computing nodes. Therefore, the master process can assign several transmitters to each of the worker processes. For distributing the work among the worker processes, the master process proceeds to decompose the loaded raster data into arrays of basic-data-type elements, e.g. floats or doubles, before dispatching them to the multiple worker processes (see "Input data broadcasting" in Figure 3.5 on page 40). The decomposition of the data applies to the digital-elevation and the clutter data only. In the next step, the master process starts a message-driven processing loop (see "Processing loop" in Figure 3.5 on page 40), which main task is to assign and distribute the configuration data of different transmitters among idle worker processes.

The flow diagram shown in Figure 3.6 on page 41 depicts in more detail the steps inside the "Processing loop" step of the master process. In the processing loop, the master process starts by checking the available worker processes, which will calculate the radio coverage prediction for the next transmitter. It is worth pointing out that this step also serves as a stopping condition for the processing loop itself (see "Any worker still on?" in Figure 3.6 on page 41). The active worker processes inform the master process they are ready to compute by sending an idle message (see "Wait for idle worker" in Figure 3.6 on page 41). The master process then announces the idle worker process it is about to receive new data for the next calculation, and it dispatches the complete configuration of the transmitter to be processed (see "Send keep-alive message" and "Send transmitter data" steps, respectively, in Figure 3.6 on page 41). This is only done in case there are transmitters for which the coverage prediction has yet to be calculated (see "Any transmitters left?" in Figure 3.6 on page

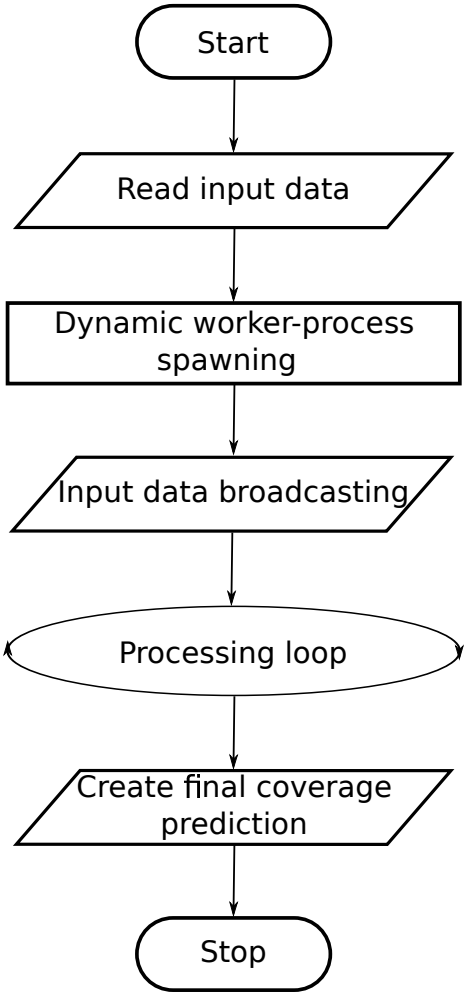


Figure 3.5: Flow diagram of the master process.

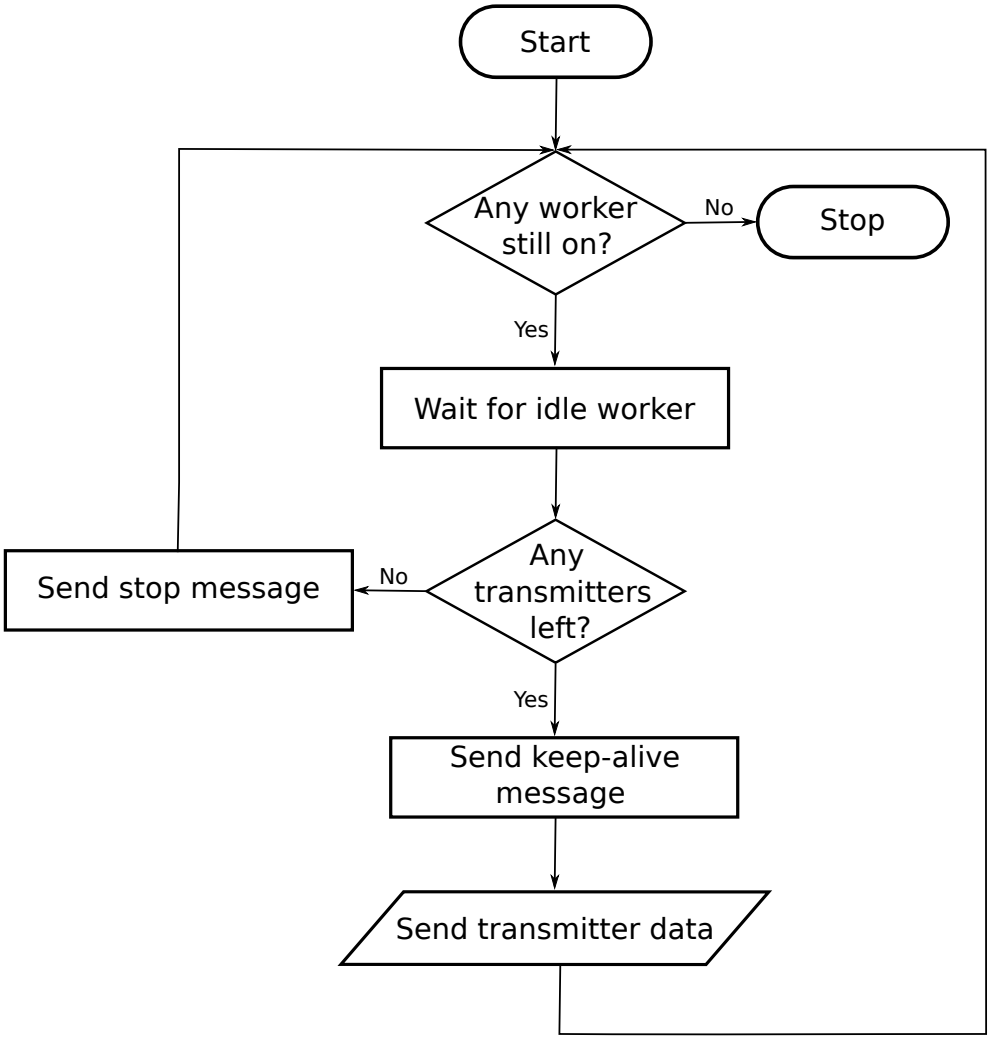


Figure 3.6: Flow diagram of the "Processing loop" step of the master process.

41). The processing loop of the master process continues to distribute transmitter data among worker processes, which asynchronously become idle as they finish the coverage-prediction calculations for the transmitters they have been assigned by the master process. When there are no more transmitters left, all the worker processes announcing they are idle will receive a shutdown message from the master process, indicating them to stop running (see "Send stop message" in Figure 3.6 on page 41). The master process will keep doing this until all worker processes have finished (see "Any worker still on?" in Figure 3.6 on page 41), thus fulfilling the stopping condition of the processing loop.

Finally, the last step of the master process is devoted to creating the final output of the calculation, e.g. a raster map (see "Create final coverage prediction" in Figure 3.5 on page 40). The final coverage prediction of all transmitters is an aggregation from the individual path-loss results created by each of the worker processes during the "Processing loop" phase in Figure 3.5 on page 40, which provides the source data for the final raster map. The aggregation of the individual transmitter path-loss results is accomplished in a similar way as in the serial version.

3.2.3.2 Worker processes

An essential characteristic of the worker processes is that they are completely independent from GRASS, i.e. they do not have to run within the GRASS environment nor use any of the GRASS libraries to work. This aspect significantly simplifies the deployment phase to run PRATO on a computer cluster, since no GRASS installation is needed on the computing nodes hosting the worker processes.

The computations of the worker processes, for which the flow diagram is given in Figure 3.7 on page 43, are initialized by data that are received from the master process at initialization time (see "Receive broadcasted data" in Figure 3.7 on page 43). It is important to note that the received data contain the transmitter and terrain-profile information which is common to all the coverage-prediction calculations, therefore making each worker process capable of processing any given transmitter.

The reason for the worker processes to be independent from GRASS arises from the design of GRASS itself. Specifically, the existing GRASS library, distributed with the GRASS GIS package, is not thread-safe, because GRASS was designed as a system of small stand-alone modules and not as a library for multi-threaded programs [**?**]. Because of this limitation, it is not an option for a parallel implementation to create separate threads for each worker process, since this would mean worker processes should wait for each other to finish, before accessing the target data. Consequently, the scalability of such implementation would be very limited.

Because concurrent access to data within GRASS by multiple processes yields undefined behavior, i.e. it is not thread-safe, the results generated by the worker processes cannot be directly saved into the GRASS data set. One possible solution would be to save the transmitter path-loss prediction result through the master process, thus avoiding concurrent access. However, sending intermediate results back to the master process from the workers would represent a major bottleneck for the scalability of the parallel version, since the results generated by a parallel computation would have to be serially processed by the master process alone. Instead, our approach allows each of the worker processes to output its results into an external database server, following an asynchronous and decoupled design. Each of the transmitter path-loss prediction results are saved in separate tables, following a similar design as the serial version. Moreover, worker processes do this from an independent thread, which runs concurrently with the calculation of the next transmitter received from the master process. When compared to the serial version, the overlap between calculation and communication achieved by the use of an auxiliary thread completely hides the latency created by the result dumping task, and makes better use of the system resources.

After the broadcasted data are received by all the worker processes, each worker process proceeds to inform the master process that it is ready (in an idle state) to receive the transmitter-configuration data that defines which transmitter path-loss prediction to perform (see "Send idle message" in Figure 3.7 on page 43). If the master process does not instruct to stop processing (see "Has stop message arrived?" in Figure 3.7 on page 43), the worker process collects the transmitter configuration sent (see "Receive transmitter data" in Figure 3.7 on page 43). However, in case a stop message is received, the worker process will wait for result-dumping threads to finish (see "Wait for result-dump threads" in Figure 3.7 on page 43) before shutting down. The coverage calculation itself follows a similar design as the serial version (see "Coverage calculation" in Figure 3.7 on page 43) and it is executed for the received

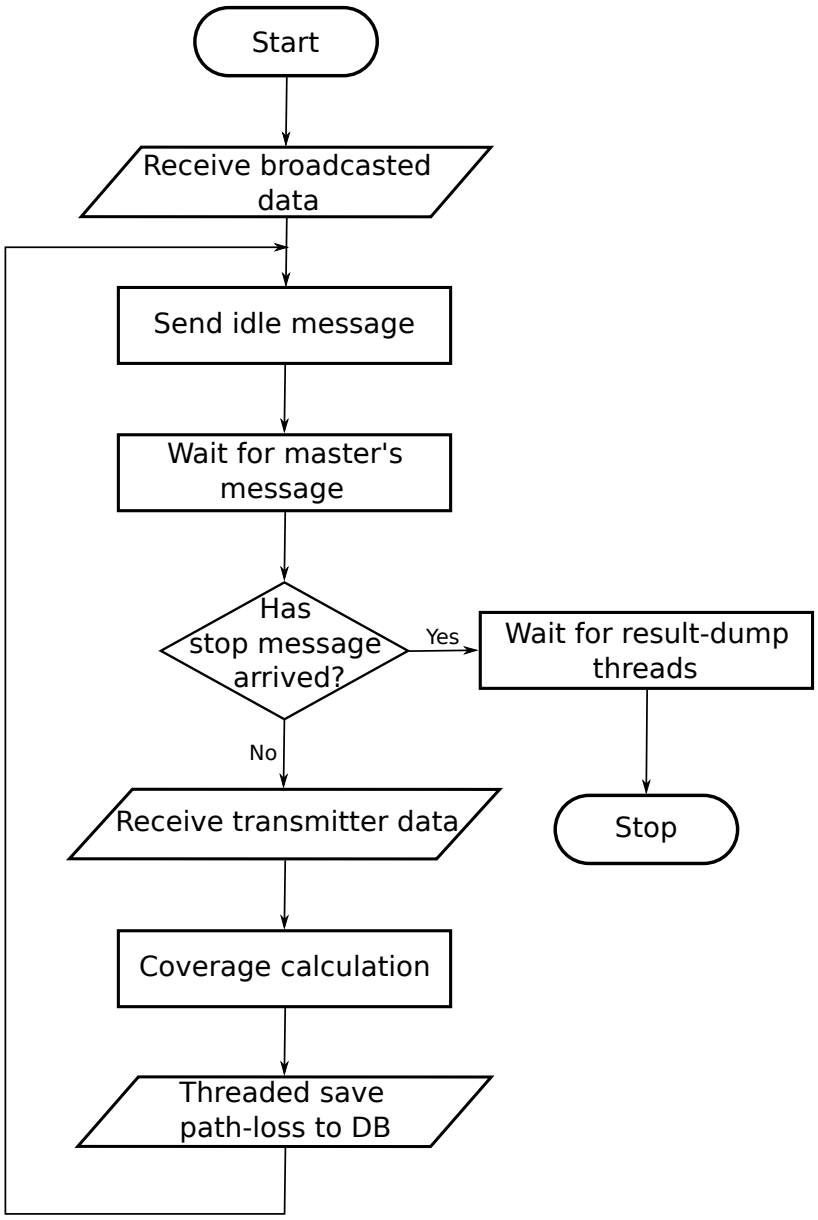


Figure 3.7: Flow diagram of a worker process.

transmitter.

As it was mentioned before, the worker process launches an independent thread to save the path-loss prediction of the target transmitter to a database table (see "Threaded save path-loss to DB" in Figure 3.7 on page 43). It is important to note that there is no possibility of data inconsistency due to the saving task being executed inside a thread, since path-loss data from different workers belong to different transmitters and are mutually exclusive. For this reason, there is no need for any concurrent I/O overlapping [**?**].

3.2.3.3 Master-worker communication

The selected message-passing technique introduced in this work might seem too elaborated, but important reasons lay behind each of the messages passed between master and worker processes. These decisions are supported by the experimental results,

introduced in Section 3.3.

The first reason to implement the message-passing technique is to support heterogeneous computing environments. In particular, our approach focuses on taking full advantage of the hardware of each computing node, thus explicitly avoiding the possible bottlenecks introduced by the slowest computing node in the cluster. In other words, computing nodes that deliver better performance get more calculations assigned to the worker processes they host. The main advantages of this technique are simplicity and negligible overhead, which contrast with more elaborated approaches for parallel-task allocation in heterogenous clusters [**?**].

A second reason for selecting a message-passing technique is related to the flexibility for load balancing, which is of great importance on heterogeneous cluster. This can be seen in Figure 3.7 on page 43 where the master process, before delivering the transmitter-configuration data, sends a message to the worker process indicating that it is about to receive more work. This a priori meaningless message has a key role in correctly supporting computer clusters. In general, there are many different ways a parallel program can be executed, because the steps from the different processes can be interleaved in various ways and a process can make non-deterministic choices [**?**], which may lead to situations such as race conditions [**?**] and deadlocks. A deadlock occurs whenever two or more running processes are waiting for each other to finish, and thus neither ever does. To prevent the parallel version of PRATO from deadlocking, message sending and receiving should be paired, being equal number of send and receive messages on the master and worker sides [**?**].

Figure 3.8 on page 45 depicts a diagram of the master-worker message passing, from which the transmitter-data transmission has been excluded for clarity. Note how each idle message sent from the worker process is paired with an answer from the master process, whether it is a keep-alive or a stop message.

3.3 Simulations

This section presents the simulations and analysis of the parallel version of PRATO. Our aim is to provide an exhaustive analysis of the performance and scalability of the parallel implementation in order to determine if the objectives of this work are fulfilled. The most common usage case for PRATO is to perform a radio-coverage prediction for multiple transmitters, therefore, a straight forward parallel decomposition is to divide a given problem instance by transmitter, for which each coverage prediction is calculated by a separate worker process.

The following simulations were carried out on 34 computing nodes of the DEGIMA cluster. DEGIMA is a computer cluster located at the Nagasaki Advanced Computing Center (NACC), in the University of Nagasaki, Japan. The computing nodes are connected by a LAN, over a Gigabit Ethernet interconnect, and share a NFS partition, from which all input and intermediate files are accessed.

Each computing node of DEGIMA features one of two possible configurations, namely:

- Intel Core i5-2500T quad-core processor CPU, clocked at 2.30 GHz, with 16 GB of RAM; and

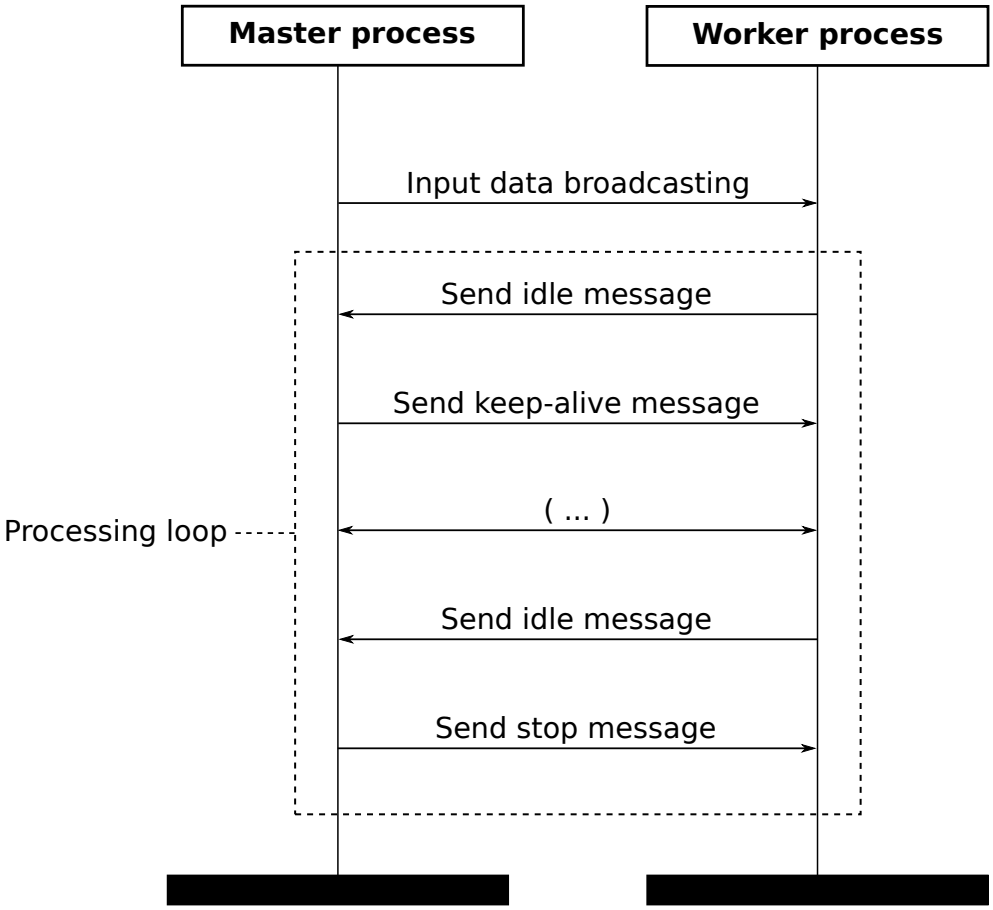


Figure 3.8: Communication diagram, showing message passing between master and one worker process.

- Intel Core i7-2600K quad-core processor CPU, clocked at 3.40 GHz, also with 16 GB of RAM.

During the simulation runs, the nodes equipped with the Intel i5 CPU host the worker processes, whereas the master process and the PostgreSQL database server (version 9.1.4) run each on a different computing node, featuring an Intel i7 CPU. The database server is the only node not writing or reading data from the common NFS partition. Instead, all I/O is done on the local file system, which is mounted on a 8 GB RAM disk.

All nodes are equipped with a Linux 64-bit operating system (Fedora distribution). As the message passing implementation we use OpenMPI, version 1.6.1, which has been manually compiled with the distribution-supplied gcc compiler, version 4.4.4.

3.3.1 Test networks

To test the parallel performance of PRATO, we have prepared different problem instances that emulate real radio networks of different sizes. In order to create synthetic test data-sets with an arbitrary number of transmitters we use the data of a group of 10 transmitters, which we randomly replicate and distribute over the whole target area. The configuration parameters of these 10 transmitters were taken from the UMTS network deployed in Slovenia by Telekom Slovenije, d.d. The path-loss predictions are calculated using the COST-231. The digital elevation model has an area of 20,270 km^2, with a resolution of 25 m^2, the same as the clutter data, which contains different levels of signal loss based on the land usage. For all the points within a transmission radius of 20 km around each transmitter, we assume that the receiver is positioned 1.5 m above the ground, and the frequency is set to 2040 MHz.

3.3.2 Weak scalability

This set of simulations is meant to analyze the scalability of the parallel implementation in cases where the workload assigned to each process (one MPI process per processor core) remains constant as we increase the number of processor cores and the total size of the problem, i.e. the number of transmitters deployed over the target area is directly proportional to the number of processor cores and worker processes. We do this by assigning a constant number of transmitters per core while increasing the number of cores hosting the worker processes. Consequently, we tackle larger radio-network instances as we increase the number of cores. Here we test for the following numbers of transmitters per worker/core: $\{5, 10, 20, 40, 80\}$, and increase the number of workers per core from 1 to 128 in powers of 2.

Problems particularly well-suited for parallel computing exhibit computational costs that are linearly dependent on the problem size. This property, also referred to as algorithmic scalability, means that proportionally increasing both the problem size and the number of cores results in a roughly constant time to solution. Therefore, with this set of experiments, we would like to investigate how well-suited the coverage-prediction problem is for parallel computing environments.

Table 3.1: Wall-clock times (in seconds) of the simulation results for weak scalability.

	Number of cores							
TX/core	1	2	4	8	16	32	64	128
5	92	99	118	122	123	124	125	126
10	140	152	171	175	177	179	180	182
20	244	260	278	282	284	285	287	290
40	451	470	491	497	500	502	504	509
80	865	892	920	925	928	931	937	948

3.3.2.1 Results and discussion

The results collected after the simulations for the weak-scalability experiments are shown in Table 3.1. All measurements express wall-clock times in seconds for each problem instance, defined as number of transmitters per core (TX/core). Wall-clock time represents real time that elapses from the start of the master process to its end, including time that passes waiting for resources to become available. They are plotted in Figure 3.9 on page 48, where the wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

The time measurements observed from the weak-scalability results show that the wall-clock times do not grow rapidly, especially when the number of cores is more than 8. Moreover, these times are almost constant for bigger problem instances, revealing that the achieved level of scalability gets close-to-linear as the amount of transmitters-per-core increases. Certainly, the parallel version of PRATO scales especially well when challenged with a big number of transmitters (10240 for the biggest instance) over 128 cores. This fact shows PRATO would be able to calculate the radio coverage prediction for real networks in a feasible amount of time, since many operational radio networks have already deployed a comparable number of transmitters, e.g. the 3G network within the Greater London Authority area, in the UK [**?**].

Not being able to achieve perfect weak scalability is due to a number of factors. Specifically, the overhead time of the serial sections of the master process grow proportionally with the number of cores, although the total contribution of this overhead remains low for large problem sizes. Moreover, the communication overhead grows linearly with the number of cores used.

To confirm these arguments, we analyze the times of each of the steps taken by the master process relative to the total processing time. To this end, we have created plots for three problem instances 5, 20 and 80 transmitters per core, which are shown in Figure 3.10 on page 49. The relative-processing-time plots follow the formula

$$RT = \frac{t_{\text{rd}} + t_{\text{ps}} + t_{\text{db}} + t_{\text{pl}} + t_{\text{cp}}}{t_{\text{total}}}, \tag{3.5}$$

where t_{rd} is the "Read input data" wall-clock time, t_{ps} is the wall-clock time of the "Dynamic worker-process spawning" step, t_{db} is the wall-clock time of the "Input data broadcasting" step, t_{pl} is the wall-clock time of the "Processing loop" step, t_{cp} is the wall-clock time of the "Create final coverage prediction" step, and t_{total} is the total wall-clock processing time. For a reference of the different steps taking part of the

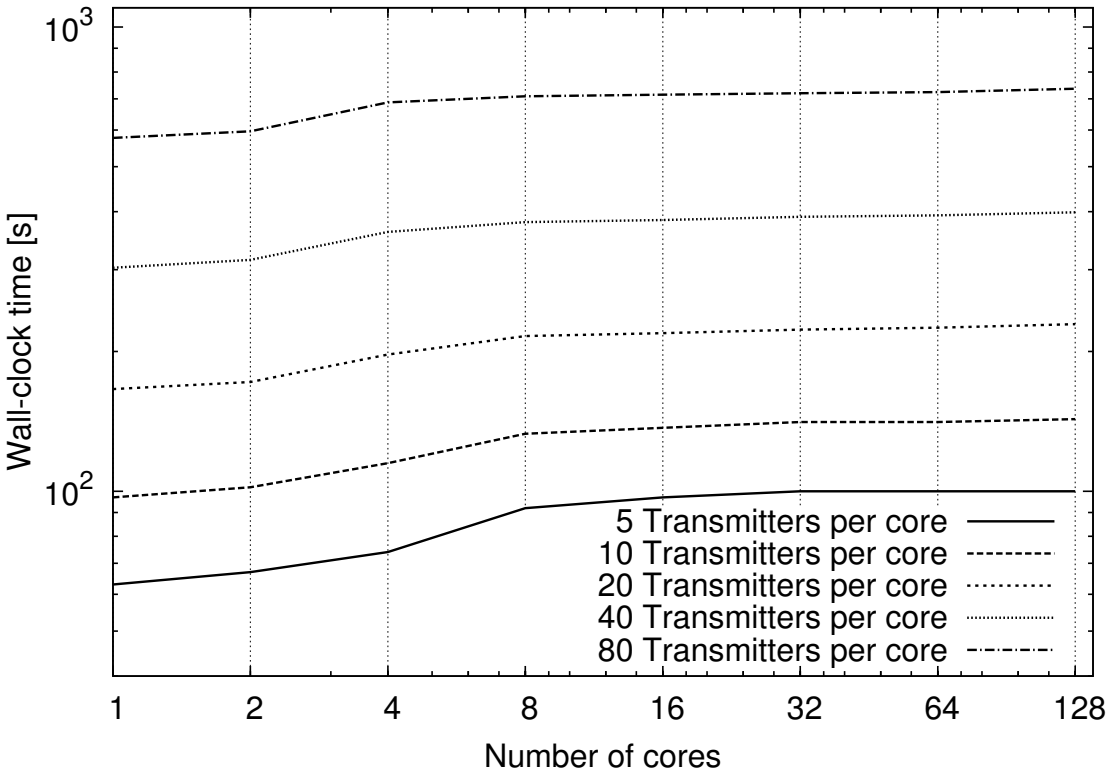


Figure 3.9: Measured wall-clock time for weak-scalability experiments as shown in Table 3.1. Experiments performed assigned one MPI worker process per available core. The wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

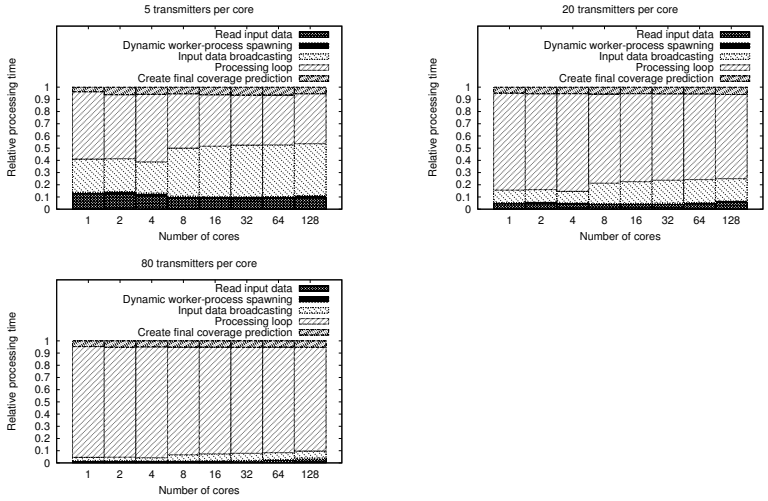


Figure 3.10: Relative times for the weak-scalability experiments. The relative-processing time axes are expressed in linear scale, whereas the axes representing the number of cores are expressed in base-2 logarithmic scale.

master process, see Figure 3.5 on page 40.

From the relative-times plots, we see that, as we increase the number of nodes, the largest fraction of the run-time is spent on the parallel processing of transmitters, which scales notably well for larger problem instances. The plotted relative times show that there is no dependency between the relative processing times and the number of cores used, confirming the good weak-scalability properties noted before. Additionally, in all three plots we may observe a "jump" in the relative time for the "Input data broadcasting" step that takes place when comparing the result from 4 to 8 cores, i.e. from one to two computing nodes, as each node hosts "1 worker per core" or a total of "4 workers per node". This "jump" is due to the use of network communication when more than one computing node participates in the parallel processing. In addition, we may also conclude that the network infrastructure has not been saturated with the data-passing load, since the relative times for input-data broadcasting do not grow exponentially from 8 cores onward. Regarding the "Create final coverage prediction" step, we may see that as we increase the number of cores the relative times grow proportionally for all three problem sizes.

3.3.3 Strong scalability

This set of simulations is meant to analyze the impact of increasing the number of computing cores for a given problem size, i.e. the number of transmitters deployed over the target area does not change, while only the number of cores used is increased. Here we test for the following number of transmitters: $\{64, 128, 256, 512, 1024, 2048, 4096\}$, and increase the number of workers per core from 1 to 128 in powers of 2 for each problem size.

3.3.3.1 Results and discussion

The results of the time measurements collected after the simulations for the strong-scalability experiments are shown in Table 3.2. All times are expressed in seconds.

Table 3.2: Wall-clock times (in seconds) of the simulation results for strong scalability.

	Number of transmitters						
No. cores	64	128	256	512	1024	2048	4096
1	714	1392	2740	5437	10830	21562	43217
2	386	734	1419	2791	5535	10996	21987
4	232	408	751	1432	2811	5549	11042
8	155	242	409	754	1441	2817	5549
16	113	156	244	414	759	1447	2821
32	92	114	159	245	414	760	1449
64	82	94	115	159	245	420	764
128	-	83	94	116	159	248	423

These wall-clock time measurements are plotted in Figure 3.11 on page 51, where the time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

The time measurements show that small problem sizes per core have a relatively large proportion of serial work and communication overhead. Therefore, the performance deteriorates as the number of transmitters per core approaches one. It can be observed in Figure 3.11 on page 51 that as we increase the number of transmitters used to solve a given problem size, the slope of the curve generated by the progression of wall-clock times tends to a flat line, i.e. as we increase the number of transmitters there is no reduction in compute time. This idea is more clearly noted in the test with smaller problem instances, e.g. 64, 128 and 256 transmitters. In contrast, for the problems with a number of transmitters larger than 512, the relative contribution of the non-parallel steps to the wall-clock time is smaller, and a larger portion of the time is spent on computing the transmitters coverage in parallel (see Section 3.2.3 for details on the steps of PRATO algorithm). A more detailed discussion of the reasons for the loss of parallel efficiency will be presented towards the end of this section.

In order to observe how well the application scales when compared against a base case, we have also measured the performance of the parallel implementation in terms of the speedup, which is defined as

$$S(NP) = \frac{execution\,time\,for\,base\,case}{execution\,time\,for\,NP\,cores}, \tag{3.6}$$

where NP is the number of cores executing the worker processes. As the base case for comparisons we have chosen the parallel implementation running on only one core and decided against using the serial implementation. We consider that the serial implementation is not a good base comparison for the parallel results as it does not reuse resources between each transmitter coverage calculation and it does not overlap I/O operations with transmitter computations. In practice, this means that several concatenated runs of the serial version would be considerably slower than the parallel but single worker implementation, because the serial implementation is not able to use all of the memory bandwidth and computing resources simultaneously. Therefore such comparison would be entirely biased towards the parallel implementation, showing super-linear scaling and speedups which would not be real, as the parallel version is

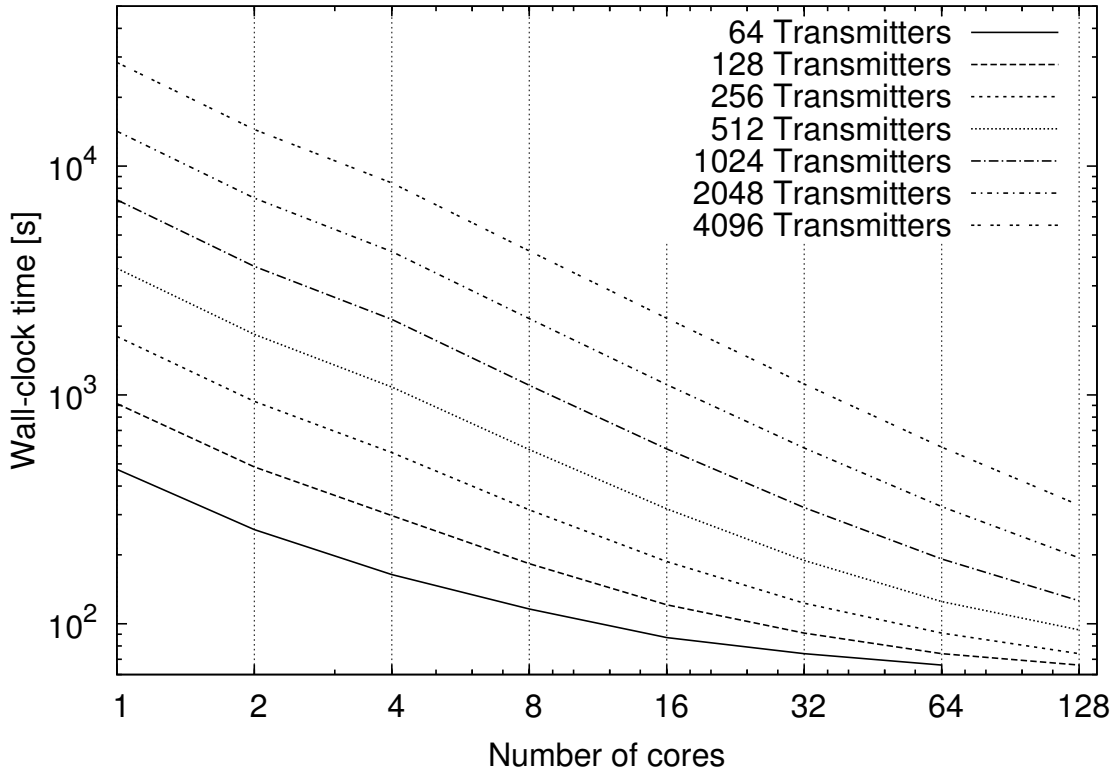


Figure 3.11: Measured wall-clock time for strong-scalability experiments as shown in Table 3.2. Experiments performed assigned one MPI worker process per available core. The wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

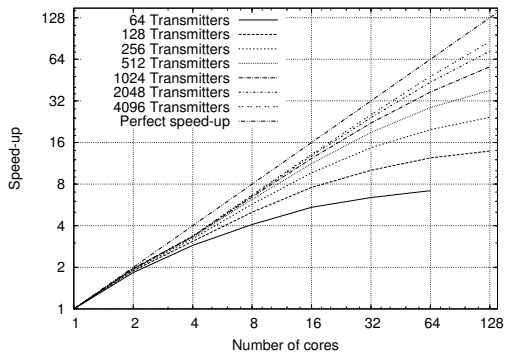


Figure 3.12: Measured speedup for strong-scalability experiments. The speedup axis is expressed in base-2 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

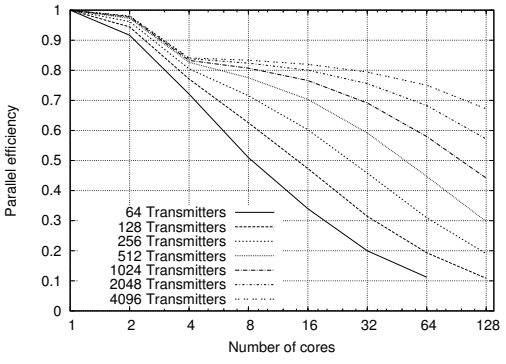


Figure 3.13: Measured parallel efficiency for strong-scalability experiments. The parallel-efficiency axis is expressed in linear scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

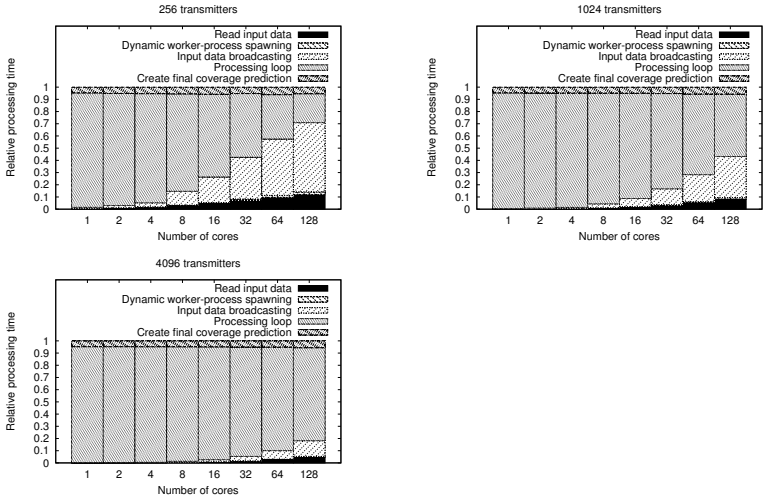


Figure 3.14: Relative times for the strong-scalability experiments. The relative-processing time axes are expressed in linear scale, whereas the axes representing the number of cores are expressed in base-2 logarithmic scale.

better equipped to make use of the system resources by means of multiple threads.

Using the speedup metric, linear scaling is achieved when the obtained speedup is equal to the total number of processors used. However, it should be noted that perfect speedup is almost never achieved, due to the existence of serial stages within an algorithm and communication overheads of the parallel implementation [**?**].

Figure 3.12 on page 51 shows the speedup of the parallel implementation for up to 128 cores (running one worker process per node), and compares seven different problem sizes with 64, 128, 256, 512, 1024, 2048 and 4096 transmitters deployed over the target area. The number of transmitters used in these problem sizes are comparable to several operational radio networks that have already been deployed in England, e.g. Bedfordshire County with 69 UMTS base stations, Cheshire County with 132 UMTS base stations, Hampshire County with 227 UMTS base stations, West Midlands with 414 UMTS base stations, and Greater London Authority with 1086 UMTS base stations [**?**]. Moreover, consider that it is common for a single UMTS base station to host multiple transmitters.

We can see that the significant reductions in wall-clock time for large problem sizes shown in Figure 3.11 on page 51 are directly correlated with the speedup factors shown in Figure 3.12 on page 51.

To study how well PRATO utilizes the available computing resources we consider the parallel efficiency of the implementation, i.e. how well the parallel implementation makes use of the available processor cores. The definition of parallel efficiency is as follows:

$$E(NP) = \frac{S(NP)}{NP}, \tag{3.7}$$

where $S(NP)$ is the speedup as defined in Equation (3.6), and NP is the number of cores executing worker processes. Figure 3.13 on page 51 shows the parallel efficiency of the parallel implementation for different problem sizes as we increase the number of processing cores.

The ideal case for a parallel application would be to utilize all available resources,

in which case the parallel efficiency would be constantly equal to one as we increase the core count. From the plot in Figure 3.13 on page 51, we may observe that the efficiency is less than one, hence the computational resources are under utilized. In accordance to the previous analysis, the under utilization of the computing resources is more significant for the smaller problem sizes, where number of assigned transmitters per core approaches one. This is due to the increased relative influence introduced by serial and communication overheads, without which the parallel implementation would not be feasible. On the other hand, the relative time contribution of the serial and communication overheads is significantly reduced as the work-load per core increases. Unsurprisingly, these results confirm what it has previously been suggested during the weak-scaling analysis, i.e. it is not worth parallelizing small problem instances over a large number of nodes, since the time reduction due to computations that make use of the extra parallel resources is surpassed by the extra parallel initialization and communication overhead.

Similarly as in the weak-scaling test, we study the relative contribution of each of the steps of the master process as we increase the number of cores used for a fixed problem size. In this case, we have created plots for three problem instances, namely 256, 1024 and 4096 transmitters, which are shown in Figure 3.14 on page 52. The relative times shown are calculated using the formula depicted in Equation (3.5).

We may observe the non-parallel steps comprising "Read input data", "Dynamic worker-process spawning", "Input data broadcasting" and "Final coverage prediction" contribute with a larger portion of time as we increase the number of cores, because the total wall-clock processing time decreases. Additionally, the low parallel efficiency for small problem sizes, particularly for 256 transmitters (left-most plot in Figure 3.14 on page 52), is validated as we see the relative small proportion of the radio-coverage calculation ("Processing loop") compared to the serial steps of the process.

3.3.4 Load balancing

In this section, we analyze the level of utilization of the computing resources available at the computing nodes hosting the worker processes. Computing-resource utilization is achieved by partitioning the computational workload and data across all processors. Efficient workload distribution strategies should be based on the processor speed, memory hierarchy and communication network [**?**].

The parallel implementation of PRATO performs load-balancing using point-to-point messages (see Section 3.2.3.3) between master and worker processes. When a worker process issues an idle message (see "Send idle message" in Figure 3.8 on page 45), the worker process will block until the message arrives to the master process. A similar situation occurs when the master process signals a worker back, whether to indicate it to shutdown or to continue working. Since the process-to-core mapping is one-to-one, blocking messages typically waste processor cycles on a computing node [**?**]. Specifically, we would like to verify the penalties that such synchronization technique has on the scalability of the parallel implementation.

We evaluate the load empirically [**?**] by using the following metric as an indicator of the load balancing among processes:

$$LB(NP) = \frac{minimum\,execution\,time\,among\,NP\,cores}{processing\,loop\,time\,of\,master\,process}, \tag{3.8}$$

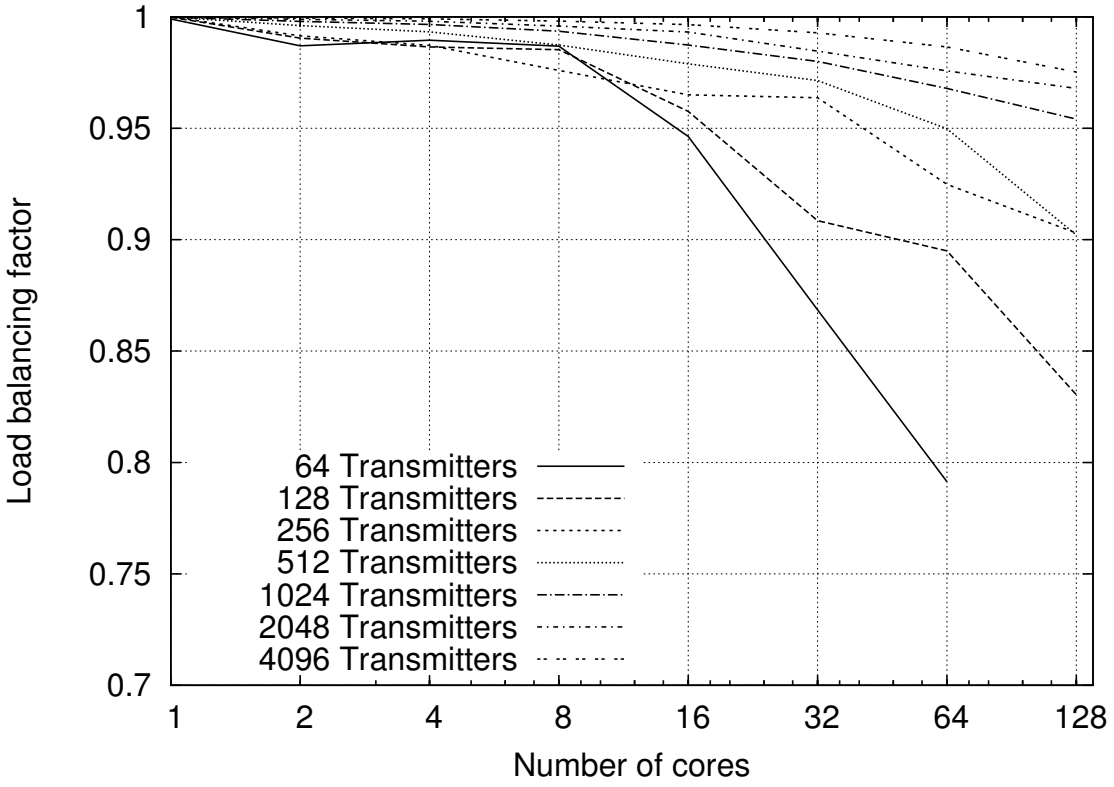


Figure 3.15: Load balancing among worker processes.

where NP is the number of cores executing worker processes. Taking the processing-loop time of the master process ensures we measure the overhead of the message passing during the time while the coverage prediction is being executed by the workers. This means that the time measurement is performed excluding the serial parts of the process, i.e. after the common data have been broadcasted to all worker processes ("Input data broadcasting" in Figure 3.5 on page 40), until the beginning of the last step ("Create final coverage prediction" in Figure 3.5 on page 40).

High performance is achieved when all cores complete their work within the same time, hence showing a load-balancing factor of one. On the other hand, lower values indicate disparity between the run times of the various worker processes sharing the parallel task, thus reflecting load imbalance.

3.3.4.1 Results and discussion

For this set of experiments, we have chosen the same problem sizes as for strong scalability in Section 3.3.3, where the coverage predictions are calculated up-to 128 cores, running on 32 computing nodes.

From the plot shown in Figure 3.15 on page 54, it is clear that the influence of the message-passing overhead over the processing time is inversely proportional to the amount of work each worker process receives. Additionally, for the biggest problem instances (1024, 2048 and 4096 transmitters), parallel-process execution times are within 95% of a perfect load-balancing factor, and within 90% for problem sizes with 256 and 512 transmitters, showing a very good performance of the dynamic task assignment, driven by our message-passing technique. For problem instances of 64

and 128 transmitters, the parallel-process times are within 80% of the perfect load balancing, showing that, as the number of transmitters per core approaches to one, latencies introduced by several hardware and OS-specific factors (e.g. TurboBoost, process affinity, etc.) are influential over the total process time. Particularly, message-passing is not able to compensate these latencies as it is executed only once per worker process.

It is worth pointing out that the very good load-balancing factors shown here are not only merit of the message-passing technique. The result dumping of partial path-loss predictions, performed by the worker processes in a separate thread into an external database server, prevents data synchronization from occurring at each iteration of the parallel process, consequently improving the load-balancing factors significantly.

3.4 Related work

As it has been mentioned before, the reference implementation for PRATO is the work done by Hrovat et al. [84]. The reported results show a comparable quality to those of a professional radio-planning tool. Since the results of the conducted comparison tests showed identical results between PRATO and this work, we may conclude that PRATO reaches solutions of comparable quality to those of a professional tool. However, a performance comparison with this work has not been performed, because it only deals with serial implementations.

A different example of a GIS-based open-source radio planning tool, called Q-Rap, has been presented in [**?**]. Developed by the University of Pretoria and the Meraka Institute of South Africa, the software was made publicly available in May 2010. Its design is geared towards an end-user tool with a graphical user interface, not appropriate for big batch jobs involving thousands of transmitters, or even parallel job execution. It is implemented as a plug-in for the Quantum GIS (QGIS) open source system [**?**].

The task-parallelization problem within the GRASS environment has been addressed by several authors in different works. Campos et al. [**?**] present a collection of GRASS modules for watershed analysis. Their work concentrates on different ways of slicing raster maps to take advantage of a potential MPI implementation, but there are no guidelines for work replication. Moreover, the hardware specification, on which the experiments have been run, is missing, making it very difficult to build upon this work.

On the field of high-performance computing, Akhter et al. [**?**] have presented implementation examples of a GRASS raster module, used to process vegetation indexes for satellite images, for MPI and Ninf-G environments. The main drawback with their methodology is the compulsory use of GRASS libraries in all the computing nodes that take part in the parallel calculation, making them more difficult to setup. Moreover, the authors explicitly acknowledge a limitation in the performance of their MPI implementation for big processing jobs. The restriction appears due to the computing nodes being fixed to a specific range, since the input data are equally distributed among worker processes, creating an obstacle for load balancing in heterogeneous environments. It is worth pointing out that in the parallel implementation of PRATO we specifically address this problem with our message-passing technique.

Similarly, Huang et al. [**?**] use the parallel inverse distance weighting interpolation algorithm as a parallel-pattern example. Although it is not explicitly noted, it can be concluded that the computing nodes make use of the GRASS environment, again making them more difficult to setup. Moreover, since the amount of work is evenly distributed among all processes (including the master one), their approach would also show decreased efficiency in heterogeneous environments.

3.5 Summary

We have presented the design and implementation of PRATO, a parallel radio-coverage prediction tool for GRASS GIS. Extensive simulations were performed in the DEGIMA computer cluster of the Nagasaki Advanced Computing Center. The results have been analyzed to determine the level of scalability of the implementation, as well as the impact of the introduced patterns for parallel algorithm design within GRASS GIS.

The conducted analysis shows that PRATO is able to calculate the radio-coverage prediction of real-world mobile networks in a reduced amount of time with a high scalability level. The promising results also show the great potential of our approach to parallelize other time-consuming tasks for GRASS GIS, although this point still has to be fully demonstrated. Particularly, the gathered results suggest that our approach would be also beneficial in the area of mobile network optimization, where thousands of simulations take part of the evaluation step during an optimization process. Still, further research is needed on how this method may be fully exploited.

Nevertheless, as PRATO is a free and open-source software project, it can be readily modified and extended to support, for example, other propagation models and post-processing algorithms. This characteristic defines a clear advantage when compared to commercial and closed-source tools.

4 Experimental evaluation: the service-coverage problem

The high-performance of PRATO, the radio-coverage simulation framework presented in Chapter 3, allows dealing with big problem instances in a reduced amount of time. Additionally, it enables tackling optimization problems that, because of their size, are out-of-reach of traditional approaches, mainly due to the computational-time complexity of their objective-function evaluation.

In this chapter, the challenge is to exploit PRATO for solving one the of classic optimization problems of radio networks: the service-coverage problem. Considering the minimization of the total amount of pilot power subject to a full coverage constraint, a novel optimization approach is introduced. The presented method, based on parallel autonomous agents, gives very good solutions to the problem in an acceptable amount of time. The parallel implementation takes full advantage of GPU hardware in order to achieve considerable speed-up. The interpretation of the experimental results, considering six real-world, radio networks of different sizes, analyzes solution-quality and performance aspects.

The content of this chapter extends the research work published by the author in [26] and [85]. The rest of this chapter is organized as follows. Section 4.1 gives a description of the coverage problem and its motivation from the mobile operator's perspective. In Section 4.2, a short overview of related research works is given, before formally introducing the key elements of the service-coverage problem in Section **??**. The parallel-agent approach, as well as the strategies used for result comparison, are presented in Section 4.4. A detailed description of the tested implementations is given in Section 4.5, followed by the simulations and their analyses in Section 4.6.

4.1 Motivation

Solving the service-coverage problem for radio networks has received a great deal of attention in the past years. Its complexity demands the confluence of different skills in areas such as propagation of radio signals, telecommunications and information systems, among others.

Even several decades after the launch of the first commercial GSM network, service-coverage planning remains a key problem that all mobile operators have to deal with. Its intricacy arises from the wide range of different combinations of configuration parameters and their evaluation-time complexity. One crucial parameter, which is mainly subject of adjustment, is the transmit power of the pilot signal (see Section 2.4.3).

Regardless of the mobile technology used, e.g., GSM, UMTS or LTE, reducing

pilot-power usage is related to issues regarding human exposure to the electromagnetic fields generated by base-station antennas [86]. During the past few years, public opinion has been extremely sensitive regarding this issue, and thus many countries have already imposed safety standards to limit the electromagnetic field levels produced by antennas in a given range.

From the UMTS perspective, minimizing pilot-power usage leaves more power available for increased network capacity. This is especially important if the traffic and other channels are configured relative to CPICH [72]. Moreover, as the users' demand for internet access and data services increases [87], so does the pressure on existing network infrastructure, making parameter optimization the only viable solution in the short-term [73].

The idea of using autonomous agents for optimization is not new. It has proven to be a solid optimization approach for solving different types of problems, not only within the area of mobile networks [86, 88], but also in other fields [83, 89]. Nevertheless, we have not observed any similar optimization method for solving the service coverage problem in mobile networks [26]. Moreover, this is, to the best of our knowledge, the first work to experiment with optimization of a real UMTS network on a fully-enabled GPU environment.

Our optimization approach is based on a state-of-the-art mathematical model, that has been previously used to solve a comparable problem [90]. The large computational-time complexity when dealing with big problem instances is tackled using a parallel implementation of the agent-based algorithm for on GPU. This minimizes the overhead when deploying a larger number of agents working in parallel over the service area, only limited by the amount of memory available.

The algorithm on subsets of a real radio network deployed in Slovenia. All network-related data were provided by the engineers of the Radio Network department at Telekom Slovenije, d.d. The results show that the solution quality is greatly improved, compared to classic coverage-planning techniques.

4.2 Related work

There are several approaches in the literature that address solving the service-coverage problem in radio networks [73, 91]. Some of them even claim to achieve near-optimal solutions [65]. As a matter of fact, most formulations are only useful for small network instances and often fail when challenged with larger and real-world networks.

A genetic-algorithm approach for solving the service-coverage problem for GSM networks is presented in [92]. The proposed solution is based on the physical distribution of base-station antennas in order to maximize coverage. The simulations were performed on a test network with 40 candidate sites for base-station antennas.

In [65], Siomina and Yuan considered the problem of minimizing the total amount of pilot power for UMTS networks subject to a full coverage constraint. They tackled the problem with an iterative linear-programming approach, reporting very good results for some test networks, containing from 15 to 65 base stations. The authors noted that bigger problem instances could not be solved because of hardware constraints on the target platform.

As for LTE networks, the service-coverage problem is presented in [93]. The authors presented an algorithm, based on reinforcement learning, to tackle three aspects

of the coverage problem, i.e., coverage holes, weak coverage and pilot pollution. The experimental simulations, performed on 3 base stations, used different antenna-tilt configurations as the proposed solutions. The service-coverage problem as presented in this chapter corresponds to achieving full coverage of the target area, i.e., without coverage holes.

4.3 Radio-network model

Extending the representation of a radio-network model from [94], this section addresses the definitions of all the elements included in the mathematical model used for the simulations.

Our goal here is to analyze the state of the network in a given situation, e.g. a 'snapshot' at an arbitrary instance. A snapshot consists of a set of mobiles (or users) having individual properties, such as location, and equipment type. The static approach inherently ignores dynamic effects that influence the system, like fast power control [94].

For additional information regarding mathematical models of comparable problems, see [73].

4.3.1 Basic elements

Consider a UMTS network with a set of antenna installations (cells), C. A RSG of a given resolution represents a geographical area, A_{total}, within which a set of mobiles, M, is spatially distributed over the pixels of A_{total}. Further, $L^{\downarrow}_{cm}$ is defined as the downlink attenuation factor between cell $c \in C$ and mobile $m \in M$. Similarly, $L^{\uparrow}_{mc}$ represents the uplink attenuation factor between mobile m and cell c. The attenuation factor values are calculated by performing signal propagation predictions for every pair (c, m), $c \in C$, $m \in M$, using the radio-propagation model introduced in ...???. These predictions already include losses and gains from cabling, hardware, and user equipment.

The amount of power allocated to the pilot signal of cell c is denoted as p_c, and it can adopt any value from the sorted set of available pilot power levels, $P_c = \{p_c^1, p_c^2, ..., p_c^k\}$, where p_c^k is the maximum power.

Based on the introduced elements, the received pilot power from cell c to mobile m is $L^{\downarrow}_{cm} p_c$.

4.3.2 Coverage

A mobile m within the area A_{covered} is under service coverage, meaning that, at least, one cell c covers it. Cell coverage is provided to a mobile m from a cell c if its signal-to-interference ratio, $sir(c, m)$, at the RSG pixel where m is located, is not lower than a given threshold, γ^{cov}, i.e.,

$$sir(c, m) = \frac{L^{\downarrow}_{cm} p_c}{\tau_0 + \sum_{i \in C} L^{\downarrow}_{im} p_i} \geq \gamma^{\text{cov}} \tag{4.1}$$

where τ_0 is the thermal noise. For convenience, a binary function is defined to determine the coverage of a mobile m by a cell c. So, for any pair (c, m), $c \in C$, $m \in M$,

the coverage of mobile m by cell c is defined as

$$cov(c,m) = \begin{cases} 1 & if\ sir(c,m) \geq \gamma^{\text{cov}} \\ 0 & otherwise \end{cases}. \tag{4.2}$$

A set, denoted as C_m, $C_m \subset C$, contains all the cells covering a mobile m. From this set, the cell with the highest $sir(c,m)$ is refered to as the best server, and denoted as c_m^*.
Notice that the described radio-network model is easily adaptable for different mobile technologies, e.g., GSM, UMTS and LTE. For example, if solving the service-coverage problem for UMTS, it would be reasonable to assume that all cells in the network operate at maximum power, and adapt Equation (4.1) accordingly. This is, from the interference point of view, the worst-case scenario [65, 90]. This assumption guarantees, that even under heavy user traffic, full coverage of the service area is maintained due to the cell-breathing principle [72].

4.3.3 Problem complexity

It has been proved that the problem of pilot power optimization for full coverage of the service area is NP-hard, since it can be reduced to the set-covering problem [95]. Consequently, as long as $P \neq NP$, it is unfeasible that a polynomial-time algorithm exists, which is able to find an exact solution to this problem.

4.3.4 Optimization objective and constraints

In the problem of optimization of pilot powers for service coverage, the objective is to find a set of pilot power settings for all cells in the network, such that the total pilot power used is minimized, and a given service coverage criteria is fulfilled. In other words, solving the service-coverage problem corresponds to finding the pilot power levels p_c, for all cells $c \in C$, such that coverage of at least b mobiles is guaranteed, while the total amount of pilot power used is minimized. Since we are considering full coverage, we denote $b = |M|$. Consequently, the optimization objective is defined as follows

$$P^* = \min \sum_{c \in C} p_c, \tag{4.3}$$

subject to

$$\frac{\sum_{m \in M} cov(c_m^*, m)}{b} = 1. \tag{4.4}$$

4.4 Optimization approaches

Since the problem instances we will be analyzing are part of a real mobile network deployed in Slovenia by Mobitel Telecommunication Services, Inc., there are no references in the literature of other optimization techniques dealing with exactly the same data set. For this reason, we will introduce two different strategies for setting the pilot power, that shall enable us to compare the experimentation results. The first strategy

is the attenuation-based pilot power, presented in [91], in which a pixel of the service area is always covered by the cell with the maximum attenuation-factor value, i.e., the minimum path loss. The second strategy is our parallel-agent approach, based on ideas inspired by two-dimensional cellular automata [96] and metaheuristics [1]. A detailed description is given in Section 4.4.2.

Similar criteria for result comparison have also been used in [65].

4.4.1 Attenuation-based approach

The first heuristic for setting the pilot power of all cells in the network is known as attenuation-based, since it relies on the downlink-attenuation factor, $L^{\downarrow}_{cm}$. A mobile located on some pixel of the service area is always covered by the cell with the maximum $L^{\downarrow}_{cm}$. Whenever the maximum available power, p^k_c, is the same for all the cells in the network, this is equivalent to selecting the cell with the minimum required pilot power to cover a mobile m. Hence, under this assumption, we identify the cell c covering mobile m as

$$p^{\text{att}}_{cm} = \min p_c \forall c \in C \iff cov(c, m) = 1 \tag{4.5}$$

Picking the cells conforming to Equation (4.5) and setting the pilot powers accordingly, we achieve full coverage of the service area with a solution exhibiting a total pilot power of

$$P^{\text{att}} = \sum_{c \in C} \max p^{\text{att}}_{cm} \tag{4.6}$$

The procedure to find a cell c for every mobile m in the service area consists in sorting, in descending order, all mobiles by their attenuation-factor values, $L^{\downarrow}_{cm}$. The solution is thus established by the first b mobiles of the sorted sequence, taking the maximum pilot-power setting for a cell into account, i.e., p^{att}_{cm}.

4.4.2 Parallel-agent approach

In the parallel-agent approach, a set of autonomous worker agents explore the target geographical area, A_{total}, in order to optimize the pilot-power consumption. Each agent randomly moves over the A_{total} as it dictates different changes to the pilot power of the cells. PRATO, the radio-coverage framework presented in Chapter 3, performs the objective-function evaluations and radio-propagation predictions based on the proposed changes of each agent.

The moving process during the optimization is strictly random. However, several physical properties that are exclusive to the service-coverage problem are being exploited during the exploration of the search space. Additionally, whenever the current solution breaks any of the given constraints, the optimization process is guided back to the space of valid solutions, providing a mechanism for improving exploration and escaping from local optima.

Because the behaviour of the agents is independent between each other, a parallel implementation is fairly straight-forward to achieve. Figure 4.1 shows the architecture of the agent-optimization system. In this GPU-only architecture, agents work in a parallel and autonomous manner, while the evaluator reacts to their changes.

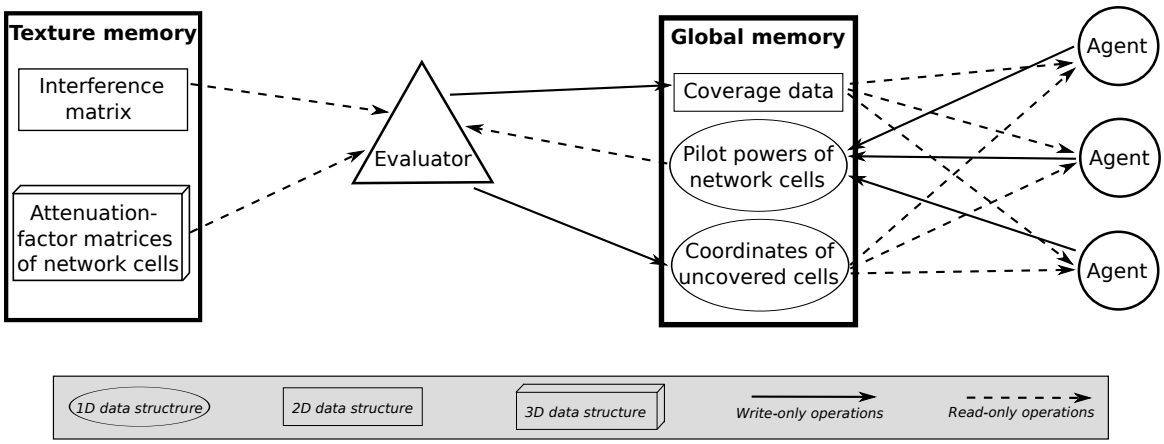


Figure 4.1: Architecture of the optimization system on GPU.

Algorithm 4.1 Pseudo-code representing the behaviour of an agent.

```
repeat
  if is_special_agent() and |A_covered| > 0 then
    l ← pick_random_location(A_covered)
  else
    l ← pick_random_location(A_total)
  end if
  move(l)
  if |C_m| = 0 then
    apply(SS_0)
  else
    if |C_m| ≥ 1 then
      apply(SS_1)
    end if
  end if
until stopping_criterion()
```

4.4.2.1 The agents

The agents apply the pilot-power changes only considering local information. Each of them encapsulates a set of steps that is consistently applied as it randomly moves through the service area of the network. Whenever an agent arrives at a new location, the set of cells covering it is calculated, e.g., C_m.

The step set an agent applies following this point is directly related to $|C_m|$, whereas its movement is determined by $|\overline{A_{\text{covered}}}|$, i.e., the cardinality of $\overline{A_{\text{covered}}}$, denoting the locations without service coverage.

The behavior of an agent is dictated by the pseudo-code shown in Algorithm 4.1. The first four steps are responsible for guiding its movements. The coordinates are randomly selected from two sets, A_{total} and $\overline{A_{\text{covered}}}$. Only "special" agents may move to a location without service coverage, and they apply the step set SS_0 for as long as the solution is not valid. The portion of "special" agents used for correcting a solution is a parameter of the optimization process. During the following steps of Algorithm 4.1, the agent applies step sets SS_0 and SS_1 based on the number of cells in C_m.

Algorithm 4.2 Pseudo-code representing step set SS_0, which is applied by the agents in areas with no service coverage.

repeat
  $c' \leftarrow next\_cell\_with\_max\_att(m)$
  $p_{c'} \leftarrow adjust\_power(c', inc\_rate)$
until $p_{c'} \in P_{c'}$

Algorithm 4.3 Pseudo-code representing step set SS_1, which is applied by the agents in areas with service coverage.

repeat
  $c' \leftarrow pick\_random\_cell(C_m)$
  $p_{c'} \leftarrow adjust\_power(c', dec\_rate)$
until $p_{c'} \in P_{c'}$

If the current location of the agent is not covered by any cell, i.e., $|C_m| = 0$, the step set SS_0 is applied (see Algorithm 4.2). At the beggining, the cell with the highest attenuation factor that may cover a mobile at this location, c', is selected. If several cells have the same $L^{\downarrow}_{cm}$ value, one of them is randomly chosen. Once c' is uniquely identified, the agent changes its pilot power by $inc\,rate$ dB.

The step set SS_1 listed in Algorithm 4.3 is applied if the location of the agent is under the coverage of one or more cells, i.e., $|C_m| \geq 1$. The first step randomly selects a cell from the set C_m, followed by a decrease of the pilot power of cell c'. This practice keeps the coverage constraint valid over A_{total}, although it might potentially break it on other areas. Ideally, every pixel of the geographical area has to be covered by exactly one network cell, although this is just a representation of a perfect solution that is unreachable because of irregularities in network topology and terrain.

In both step sets, SS_0 and SS_1, the agent makes sure that the new pilot power setting, $p_{c'}$, is an element of $P_{c'}$. If this is not the case, cell c' is discarded and another cell is repeatedly selected at the beggining of both step sets, until this condition is satisfied.

The values $inc\,rate$ and $dec\,rate$ are configurable parameters that should be set before starting the optimization process. They indicate the relative adjustment (expressed in dB) of the pilot power of cell c'. On the one hand, lowering the pilot power of a cell decreases the interference it creates within its coverage area and those of their neighbours. Since the $sir(c, m)$ value increases with lower interference, the coverage of m may be achieved by a neighbor cell with the same or lower pilot power. On the other hand, increasing the pilot power of the cell with the maximum attenuation factor improves coverage by evenly distributing the power among different network cells. This cell is, on average, the nearest one to the location of a mobile m.

4.4.2.2 The evaluator

The evaluator represents a central component of the optimization system. It reacts to the pilot-power changes by recalculating the objective-function value. Recall that the objective-function evaluation involves the radio-coverage prediction of the service area and the calculation of the total pilot power used by the cells in the target network.

After a short initialization, during which the attenuation-factor matrices of all the

cells and the interference matrix are calculated, the evaluator computes the coverage of the service area based on the pilot powers supplied as the initial solution. Initial solutions are randomly generated from valid pilot-power settings that conform to the full coverage constraint.

The evaluator also maintains a special part of the memory (see "Coordinates of uncovered cells" in Figure 4.1) that is intended for registering uncovered areas, i.e., $\overline{A_{\text{covered}}}$. If Equation (4.4) does not hold, the "special" agents randomly select a location from this portion of memory so that a valid solution may be reached again.

It is worth mentioning that the evaluator itself has no influence in the optimization process from a quality point-of-view. Its task is to provide feedback and updated information to the agents that move through the service area. From a performance point-of-view, the importance of the evaluator is significant, as it will be shown in the following sections.

4.5 Implementation

The evaluation of the objective function was completely implemented on the GPU using OpenCL (see Section 2.5.1). The reason behind this decision is the impact objective-function evaluation has on the performance of the optimization system as a whole, as discussed in Section 2.1.4. Agents' implementation is also based on the GPU, which drastically reduces the number of data transfers between CPU and GPU, since all problem elements are available on the GPU during the optimization process. In this sense, the first hardware restriction we have to overcome is the amount of memory on the GPU device. Consequently, careful memory utilization and organization are critical to successfully accommodate all involved problem elements on the GPU, the memory of which is significantly smaller than the RAM memory available in desktop computers.

4.5.1 Objective-function evaluation on GPU

The GPU implementation of the objective-function evaluator is based on work done by Hrovat et al. [84]. The evaluator implementation is one of the biggest challenges we faced. The great amount of data needed to evaluate agents' changes of pilot power was the first restriction we run into, as there was not enough memory on the GPU for all of them. The solution is to change their internal representation, namely:

- one path-loss matrix for each of the cells of the network being optimized,
- one interference matrix for the whole service area.

Still, we had to consider different data representation schemes for the problem elements, so to avoid memory limitations on the GPU device.

The path-loss matrices are potentially as big as the underlying data used to calculate the radio propagation predictions over the service area of the network. In our case, we are dealing with an area of more than 78,000 km^2, including a digital elevation model of 100 m^2 resolution. This grid includes the whole country and some of the neighboring ones, ensuring data availability even at the country borders. Having more than 100 transmitters containing real numbers as matrix elements, requires

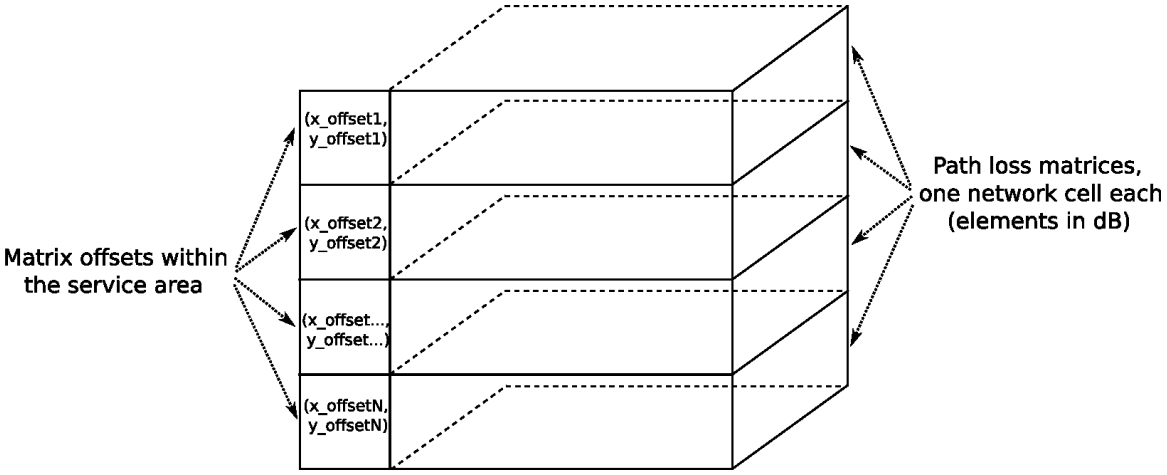


Figure 4.2: Memory organization of the path-loss matrices for network cells.

more memory than is currently available on most modern GPU hardware. For this reason, we decided to change the path-loss matrix elements unit from the linear scale to decibels (dB), coding them as *unsigned char*. After consultation with experts on the radio-telecommunications field, the decision was that the additional error introduced by using integer decibels instead of real numbers is negligible, since the digital elevation model, which has a resolution of 100 $m2$ per pixel, presents a bigger rounding problem. The path-loss between a network cell and any point on the service area should, consequently, never exceed -255 dB. This scale is large enough for problem representation, since service discovery by the mobile terminal is still successful with a RSCP of around -115 dB [72]. We also include a calculation radius around each network cell to reduce the amount of memory needed even more. For example, take a 60 km calculation radius around each cell, which is enough for the coverage calculation purposes over the 2 GHz spectrum of UMTS [72], yet it drastically lowers the memory requirement on the GPU. To correctly locate the path-loss matrices within the service area, we supply the offset of the upper-left corner of each of them. The memory layout for the path-loss matrices is shown in Figure 4.2. Because their content is constant throughout the optimization process, they are kept in read-only texture memory to take advantage of the fast access time.

Additional speed-up is achieved by incrementally recalculating the service area coverage as the agents' changes arrive. Specifically, the network cells containing new settings that have not yet been evaluated, have their pilot powers saved in negative form, serving as a flag to indicate that coverage re-evaluation is needed. Since pilot powers are, by definition, positive numbers, there is no possibility for confusion.

4.5.2 Parallel agents on GPU

With the objective-function evaluation running on the GPU, a new performance bottleneck appeared. The limitation factor in this case was the CPU-to-GPU data transfers in each iteration of the optimization process.

The GPU kernel of the agents is launched as one thread block that contains one thread per deployed agent. The thread block is organized in a one-dimensional grid. The initial location of each agent is randomly generated using the current system time as a random seed. Since OpenCL provides no function for random-number generation,

a simplified version of Marsaglia's generator [97] was implemented.

The analysis each agent does about the received signals at the current location is saved into shared memory of the thread block. It contains the network cell and its pilot-power setting. Since both numbers are of type *short*, there is enough space in a 16 KB shared-memory block to allocate 4,096 agents. The last step involves saving the new pilot powers into global memory. This step is performed by only one of the threads within the thread block in order to avoid memory-access conflicts. Updated pilot powers are saved in negative form to indicate that coverage re-calculation is needed for these cells. In case there are several updated pilot powers for one network cell, the median is calculated and applied as the new pilot power.

Even though coalesced access is not achieved by the GPU kernel of the agents, its sole implementation provided enhanced performance. This performance gain appears because of the lower number of data transfers between the CPU and the GPU, since most data is available in global memory. Moreover, the GPU kernel also produces the truly parallel behavior of the agents, as they all apply the pilot-power changes at the same time.

4.6 Simulations

4.6.1 Test networks

The test networks, Net_1, Net_2 and Net_3 are subsets of the real UMTS network deployed by Telekom Slovenije, d.d. The path-loss predictions were calculated using the ...??? model presented in Section ...???. A DEM of 100 m^2 resolution was used as the terrain-profile data. The requirements for the coverage threshold, γ^{cov}, were provided by experts of the Radio Network department of Telekom Slovenije, d.d.

Net_1 is deployed over a densely populated urban area. For this reason, the value of γ^{cov} is lower here, since network capacity is the dominating factor, whereas coverage is flexible because of a larger cell density, i.e., more base stations per surface unit. Net_2 represents a network deployed over a rural area, meaning that network capacity can be reduced at the cost of better coverage, since user density is lower. The last network, Net_3, represents a suburban area with a densely populated, but relatively small, downtown center, where a compromise between network capacity and coverage has to be achieved.

The second group of test networks, including Net_4, Net_5 and Net_6, is part of the publicly available MOMENTUM project [68]. Test network Net_4 represents the city of Berlin (Germany), Net_5 represents the city of The Hague (Netherlands), and Net_6 is one of the networks optimized in [65], representing a reduced version of Net_4. All networks include information about site locations, path-loss predictions and realistic antennas, which are part of the scenarios provided by the MOMENTUM project.

Network configurations, that represent what could be an initial-network setup by common planning standards [72], were produced using the attenuation-based approach. Such configurations can be easily calculated by a network planner. Table 4.1 lists the number of base stations and cells per test network, as well as the size of the geographical area. Different network-parameter values used during the simulations are shown in Table 4.2.

Table 4.1: Sizes of the test networks used, in terms of equipment and geographical area.

	Number of base stations	Number of cells	Surface (km^2)	Resolution (m^2)
Net_1	26	77	100.00	25
Net_2	8	23	306.25	25
Net_3	45	129	405.00	25
Net_4	65	193	56.25	50
Net_5	12	36	16.00	50
Net_6	50	148	56.25	50

Table 4.2: Network parameters of the test networks used.

	p_c^k	τ_0	γ^{cov}
Net_1	15.00 W	$1.55{\cdot}10^{-14}$ W	0.010
Net_2	19.95 W	$1.55{\cdot}10^{-14}$ W	0.020
Net_3	15.00 W	$1.55{\cdot}10^{-14}$ W	0.015
Net_4	19.95 W	$1.55{\cdot}10^{-14}$ W	0.010
Net_5	19.95 W	$1.55{\cdot}10^{-14}$ W	0.010
Net_6	19.95 W	$1.55{\cdot}10^{-14}$ W	0.010

4.6.2 Parameter settings of the parallel-agent approach

The parameter settings for the optimization algorithm were determined after some experimentation with the test networks. The parameter settings for each test networks are listed in Table 4.3.

Using a higher *inc_rate* than *dec_rate* reflects the behavior of the agents when full coverage of the service area is not guaranteed. In practice, areas without service coverage usually appear as irregular islands. The stopping criteria were set by limiting the total number of pilot-power changes an agent is allowed to make. The value was set to 10,000 even though for some of the test networks the best solutions were found in the first quarter of the experiment.

4.6.3 Experimental environment

All experiments were performed on a multi-core Intel i7 2.67 GHz desktop computer with 6 GB of RAM running a 64-bit Linux operating system. The GPU hardware

Table 4.3: Parameter settings of the parallel-agent approach for each test network.

	Agents	*inc_rate* (dB)	*dec_rate* (dB)	Pilot-power changes
Net_1	16	0.2	-0.1	10,000
Net_2	16	0.2	-0.1	10,000
Net_3	16	0.2	-0.1	10,000
Net_4	6	1.0	-0.1	10,000
Net_5	2	1.0	-0.1	10,000
Net_6	6	1.0	-0.1	10,000

Table 4.4: Optimization results of all test networks after applying different approaches for solving the service-coverage problem. All values are expressed in Watts.

	Attenuation-based		Parallel agents	
	Total power	Average pilot power	Total power	Average pilot power
Net_1	419.292	5.445	137.064	1.780
Net_2	78.297	3.404	33.344	1.450
Net_3	1,014.113	7.861	582.954	4.519
Net_4	179.876	0.932	145.715	0.755
Net_5	73.872	2.052	34.884	0.969
Net_6	147.014	0.993	112.332	0.759

used was an ATI HD5570 with 1 GB of DDR3 RAM. The implementation language used was C, combined with OpenCL and OpenMPI extensions.

4.7 Results

The results achieved by the parallel-agent approach, listed in Table 4.4, improved the optimization objective significantly. They show that the pilot-power usage was reduced in all networks while the service area was kept under full coverage. Moreover, the parallel-agent solution for Net_1 improved the attenuation-based setting by more than 300%. As for Net_2, the observed improvement is around 232%, while the improvement for Net_3 is more than 170%.

As mentioned in Section 4.1, the interpretation of these results depends on the mobile technology used. For GSM, ...???. As for UMTS, the capacity of the network has been significantly increased in all three problem instances. Therefore, a greater number of users should be able to access services provided by the mobile network, since coverage is assured. Moreover, an increased speed in data services should be observed [72]. Regarding LTE, ...???

The last test network, Net_6, is the same as in [65]. When comparing these results to those of [65], an improvement of almost 3% can be observed in the solution provided by the parallel-agent approach.

4.7.1 Convergence analysis

The graphs shown in Figures 4.3, 4.4 and 4.5 depict the convergence of the parallel-agent approach after ten independent runs for test networks Net_1, Net_2 and Net_3, respectively. Only feasible solutions were plotted, i.e., solutions that meet the full-coverage constraint. Unfeasible solutions were marked with a value of inferior quality than the worst solution found: 428 for Net_1, 129 for Net_2, and 1,435 for Net_3.

From the graphs of Net_1 (see Figure 4.3) and Net_2 (see Figure 4.4) a good initial convergence can be observed. This is followed by a steady improvement of the intermediate solutions. In Net_1, an additional solution improvement is noticed towards the end of the optimization process. This fact suggests that longer runs would potentially find better solutions in this case. In contrast, the solution improvement of Net_2 shows

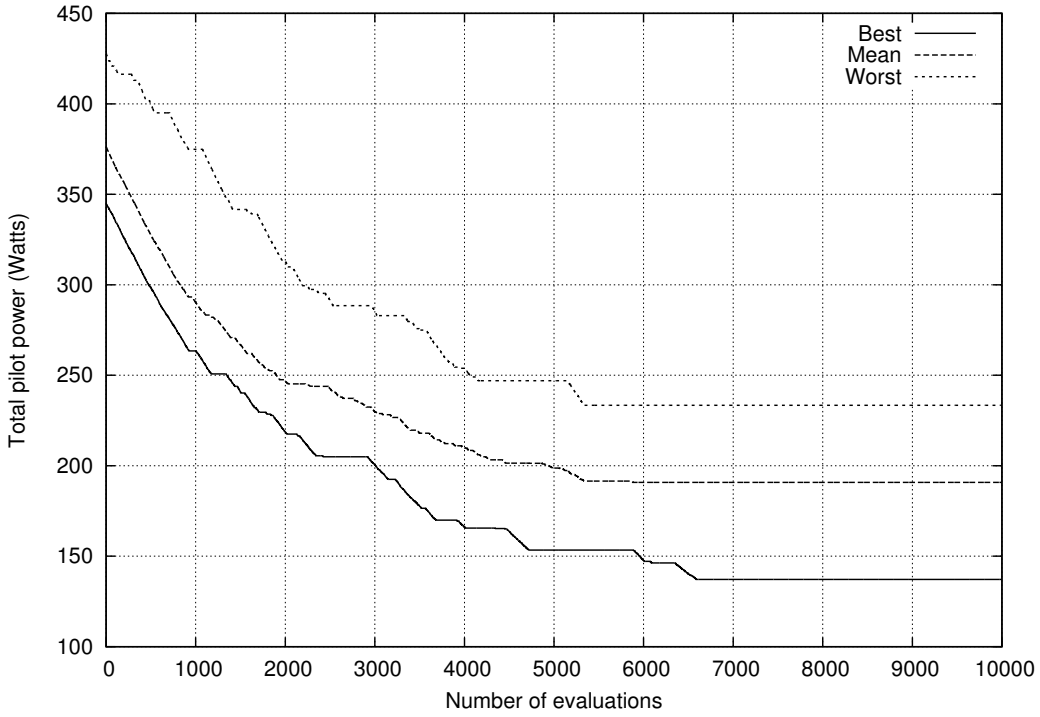


Figure 4.3: Convergence profile of the parallel-agent approach for the test network Net_1.

a flat profile towards the end, suggesting that the stopping criterion is appropiate for this instance. From the graph of Net_3 (see Figure 4.5), a slower initial convergence, followed by a steady improvement of intermediate solutions and no significant solution enhancement towards the end, can be observed. This convergence profile suggests that this problem instance presents a more difficult optimization case than for Net_1 and Net_2. Indeed, this is the largest test network in terms of surface area. However, further investigation is needed to confirm this hypothesis. Nevertheless, the parallel-agent approach improved the pilot-power usage of this test network by almost 75%.

4.7.2 Performance analysis

In this section, the performance analysis of the experimental results is presented. This analysis covers the running times during the optimization of the first three test networks, i.e., Net_1, Net_2 and Net_3. The running times were measured for each implementation, and the average times, calculated after ten independent runs, are given. The number of pilot-power changes per agent was limited to 100, while all other algorithm parameters were kept at the same values as in Section 4.6.2.

Table 4.5 lists the average wall-clock times in seconds for the different implementations and test networks. The implementations include: the CPU-MPI implementation that consists of objective-function evaluation on CPU and parallel agents over MPI, the GPU-MPI implementation that consists of objective-function evaluation on GPU and parallel agents over MPI, and the GPU-GPU implementation that consists of objective-function evaluation and parallel agents on the same GPU. The CPU-MPI implementation is the basis for the speed-up calculation of the other implementations.

Function evaluation on the GPU communicating with agents over MPI provides

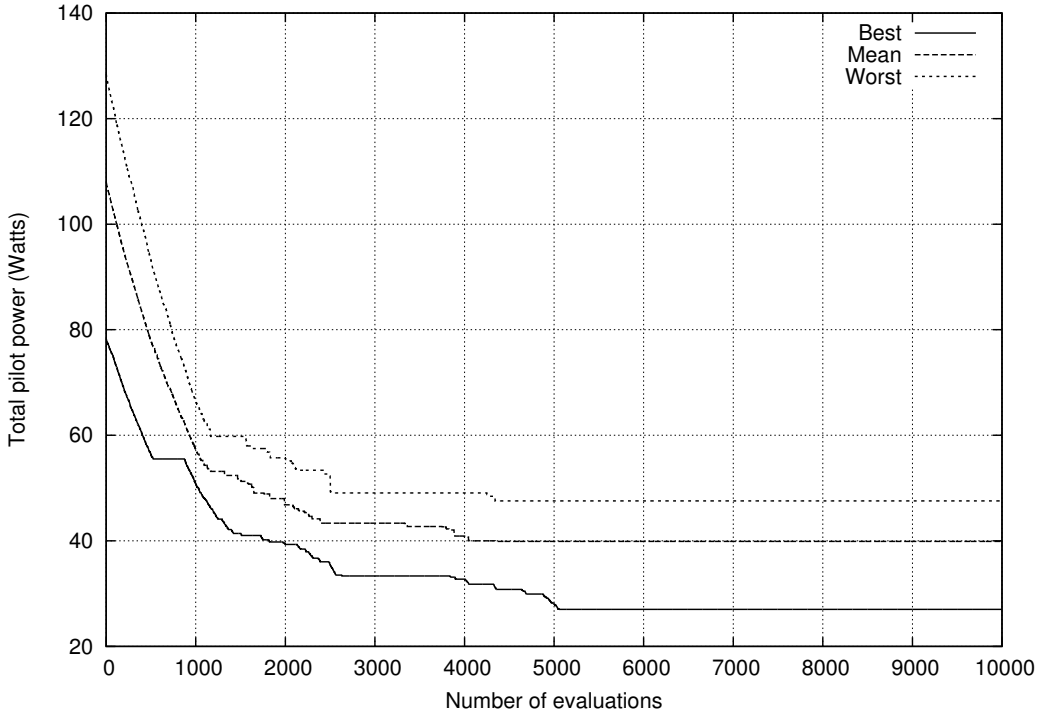


Figure 4.4: Convergence profile of the parallel-agent approach for the test network Net_2.

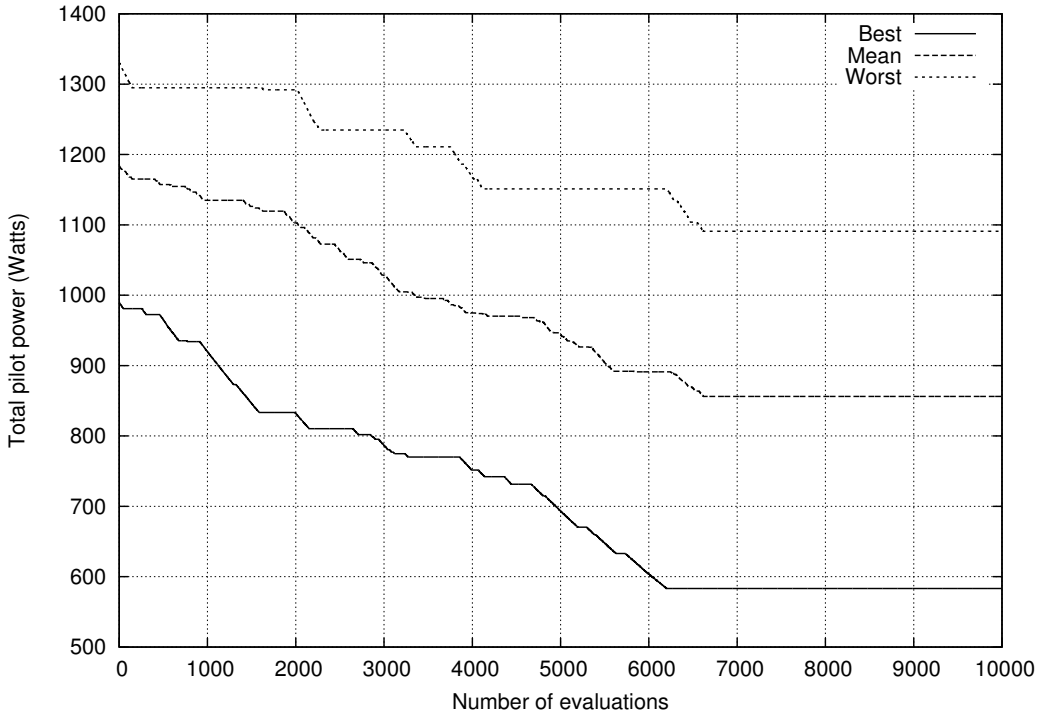


Figure 4.5: Convergence profile of the parallel-agent approach for the test network Net_3.

Table 4.5: Wall-clock times (in seconds) and speed-up factors for different implementations of the objective-function evaluation and the parallel agents.

	CPU-MPI	GPU-MPI		GPU-GPU	
	Avg. time	Avg. time	Speed-up	Avg. time	Speed-up
Net_1	105,455	346	305x	67	1574x
Net_2	33,700	195	173x	46	733x
Net_3	191,900	506	379x	117	927x

the second measured setup. The evaluator implementation takes advantage of shared memory for thread collaboration within a thread block and texture memory for constant elements, as is has been explained in Section 4.5.1. Still, the speed-up is considerable but improvable, since numerous data transfers between CPU and GPU are needed for the agents to access optimization-related information.

The last result set presents measurements for complete GPU implementation, including objective-function evaluation and agents on the same device. The substantial speed-up delivered by this combination highlights the great impact that CPU-to-GPU memory transfers have on overall system performance. This fact is supported by the speed-up between the second and third measured setups, which exhibit, on average, an improvement of more than 400%.

4.8 Summary

This chapter presented a novel optimization approach for solving the well-known service-coverage problem in radio networks. The problem addressed the full coverage of a geographical area using a minimum amount of pilot power. The newly introduced parallel-agent approach was successfully tested in six networks that represent real-world scenarios of deployed radio networks. The experimental results show that the parallel-agent approach is able to find better solutions than other common radio-planning methods [72]. Moreover, the algorithm successfully tackled larger networks, thus overcoming the obstacles of other state-of-the-art optimization methods regarding problem-instance size [65].

Compared to a different optimization approach in the literature [42], the solution-quality of the parallel-agent approach showed an improvement. The proposed solutions, calculated for the same problem instance as in [65], were improved at the cost of longer running time. It worth mentioning that it is feasible for the optimization algorithm to take a longer time to reach the solution, since design problems, as the service-coverage one, are usually solved offline.

Different implementations of the parallel-agent approach, combining a serial version on CPU, parallel processes over MPI and GPU kernels, were presented. In particular, GPU architectures enable the implementation of parallel heuristics in a natural way while substantially improving the computational-time performance. To the best of the author's knowledge, the GPU implementation of the parallel-agent approach as presented in this chapter, has not yet been described in the related literature.

5 Experimental evaluation: the SHO-balancing problem

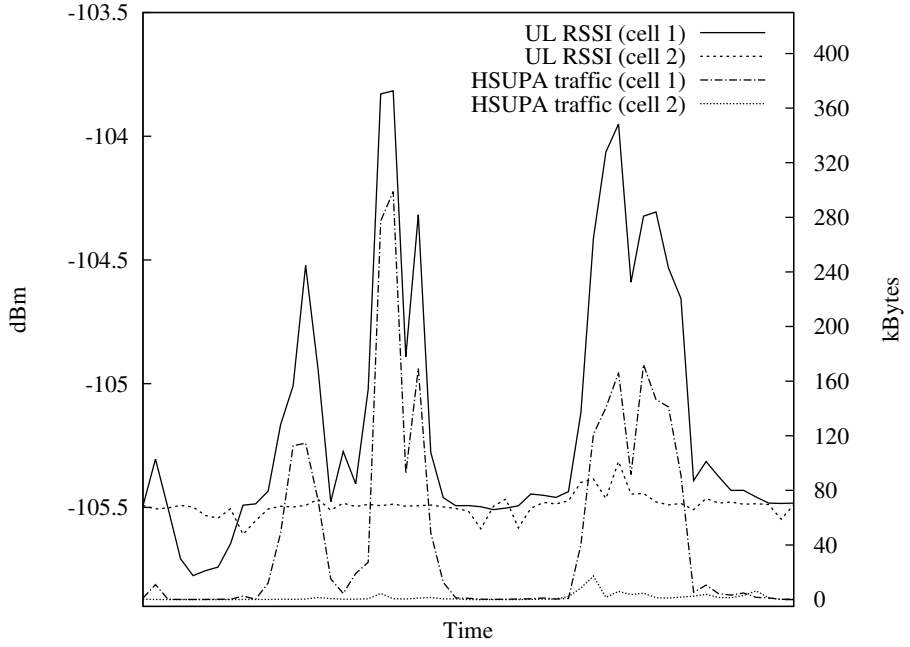
In Chapter 4, an application exploiting the advantages of faster evaluation methods has been introduced. Solving the service coverage problem for real-world networks capitalizes on the ability to tackle bigger problem instances, i.e. problems that, because of their size, were previously unsolvable in a feasible amount of time. This improved performance also allows solving optimization problems with a higher degree of complexity, usually represented by the evaluation of multi-dimensional non-convex objective functions.
This chapter focuses on solving a new optimization problem for 3G networks, dealing with downlink and uplink SHO areas (see Section 2.4.2). By introducing a penalty-based objective function and some hard constraints, the formal definition of the SHO-balancing problem in UMTS networks is given. The state-of-the-art mathematical model used and the penalty scores of the objective function are set according to the configuration and layout of a real mobile network, deployed in Slovenia by Telekom Slovenije, d.d. The balancing problem is then tackled by three optimization algorithms, each of them belonging to a different category of metaheuristics.

To the best of the author's knowledge, there is no reference in the literature to a simulation-based approach to find active downlink and uplink SHO areas. Additionally, there are no formal optimization methods known to the author that tackle the SHO balancing problem as described here. The approach described in this chapter extends the approach published by the author in [98].

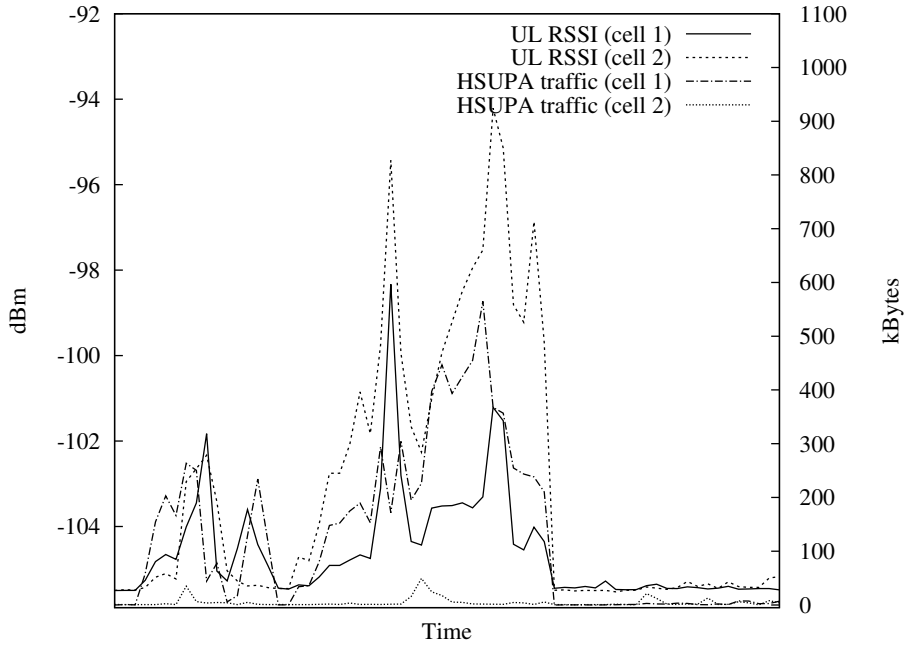
The remainder of this chapter is organized as follows. Section 5.1 decribes the motivation behind the SHO-balancing problem, whereas Section 5.2 gives an overview of other works related to CPICH-power and SHO optimization in UMTS. The static network model is presented in Section 5.3, where all the elements of the mathematical model and the objective function are defined. In Section 5.4 a short description of the optimization algorithms used is given, including their mapping to the SHO-balancing problem. The simulations, including their environment and parameter setup, are introduced in Section 5.5, followed by their results in Section 5.6.

5.1 Motivation

Despite several built-in mechanisms that allow the radio network to overcome different problems due to the lack of SHO during a HSDPA connection, some abnormal cases do arise, especially in those areas where there is SHO capability in the uplink, but none in the downlink. An example of such a case is depicted in Figure 5.1, which shows interference behavior during a HSPA connection in normal SHO conditions (a), and



(a)



(b)

Figure 5.1: HSUPA traffic and uplink interference with: (a) balanced downlink and uplink SHO conditions; (b) unbalanced downlink and uplink SHO conditions.

in unbalanced SHO conditions (b). Graph data are actual radio network statistics, taken from the mobile network deployed in Slovenia by Telekom Slovenije, d.d.. The graph on the left (a) shows a normal HSUPA-enabled service situation, in which the measured interference is proportional to the traffic being served. Note how the noise rises with the increased traffic on cell 1, while its neighbor (cell 2) has almost no interference nor traffic. Moreover, the graph profile for both traffic and noise of cell 1 are almost identical. The graph on the right (b) depicts a problematic situation, where the noise level does not only rise on the cell serving the HSUPA services (cell 1), but also on the neighboring one. Notice how the interference level rises on the cell that has almost no traffic (cell 2). It is clear that the source of this noise rise is generated by the active connection on cell 1, which shows an increase in HSUPA traffic. However, the noise level profile on cell 2 does not follow its traffic, as it did in the normal situation (a). This is due to cell 2 not being part of the active set. Such situations appear when the UL coverage is larger than the DL coverage. Interestingly enough, this seems to be an exceptional case, as Holma and Toskala write in [70] when describing soft handover in chapter 5:

"... There is no obvious reason why the serving E-DCH cell would not be the same as the serving HSDPA cell, and this is also required to be the case in the specifications."

Given the described context, the challenge is to achieve the correct balance or distribution of downlink and uplink SHO areas within a working UMTS network. Therefore, the network has to be fine-tuned to achieve a better SHO-area balancing, and thus avoiding the exceptional appearance of problematic situations as shown in Figure 5.1. This clearly implies that the mobile network configuration should not be excessively altered, since other aspects of the network are working well before starting the optimization process. Hence, the objective of the optimization problem is to find a CPICH power level configuration for all the cells in the target network, such that the balance of downlink and uplink SHO areas is improved and other network aspects are preserved. The optimization process takes into account different kinds of hardware (e.g. amplifiers, cables, and antennas), changing only the CPICH powers of the cells.

PRATO is used as the evaluation framework, as defined in Chapter 3. A state-of-the-art mathematical model [94] describes downlink and uplink SHO areas, considering the static nature of the evaluation. By introducing a penalty-based objective function and some hard constraints, a formal definition of the SHO-balancing problem in UMTS networks is given. The mathematical model and the penalty scores of the objective function are set according to the configuration and layout of a real mobile network, deployed in Slovenia by Telekom Slovenije, d.d. The SHO settings are also taken from actual network configuration, still they were adapted to closely model interference and other dynamics present in the network. The balancing problem is then tackled by three optimization algorithms, each of them belonging to a different category of metaheuristics. The optimization results, as well as the performance of each of the optimization algorithms used, are afterwards analyzed.

5.2 Related work

SHO optimization has received quite some attention from the scientific community in the last years. This mainly relates to the importance it has within deployed networks that provide high speed services such as video telephony [99] and Internet access by

means of HSPA [100].

Some authors tackle optimization problems at the planning stage of the network [101, 102], considering, among other variables, base-station locations and hardware. However, most mobile operators are unable to apply these contributions to a live network since the planning phase has long been concluded. Moreover, the great majority of the base stations have already been deployed and their hardware also installed. Therefore, from the mobile operator's point of view, mainly parameter and software optimization are the only tools available when it comes to QoS improvement (see Section 2.4.2) and network troubleshooting in the short term.

Optimizing SHO by means of CPICH is an established way of enhancing network capacity when high speed services like HSDPA and HSUPA coexist with legacy technologies [103]. The CPICH transmit power is typically between 5% to 10% of the total downlink transmit power of the base station [71], but there is no standardized method to find a CPICH power setting. A number of existing approaches to resolve this issue exist in the related literature (see [48, 104, 105]). The most effective ones are those based on optimization methods [71, 101, 106–108]. Such a wide spectrum of available procedures is directly related to the diverse criteria taken into account when assigning the CPICH power of a cell. The fundamental reason behind this fact is that the CPICH power is a common factor of various optimization problems in UMTS networks.

5.3 Radio-network model

Extending the representation of a radio-network model from [94], this section addresses the definitions of all the elements included in the mathematical model used for the simulations.

Our goal here is to analyze the state of the network in a given situation, e.g. a 'snapshot' at an arbitrary instance. A snapshot consists of a set of users (or mobiles) having individual properties, such as location, and equipment type. The static approach inherently ignores dynamic effects that influence the system, like fast power control [94].

The mathematical model links the SHO settings with CPICH power settings for each cell, the best-server pattern, and the network coverage.

For additional information regarding mathematical models of comparable problems, see [73].

5.3.1 Basic elements

Consider a UMTS network with a set of antenna installations (cells), C. A RSG of a given resolution represents the service area, A_{total}, within which a set of mobiles, M, is spatially distributed over the pixels of A_{total}. Further, $L^{\downarrow}_{cm}$ is defined as the downlink attenuation factor between cell $c \in C$ and mobile $m \in M$. Similarly, $L^{\uparrow}_{mc}$ represents the uplink attenuation factor between mobile m and cell c. The attenuation factor values are calculated by performing signal propagation predictions for every pair (c, m), $c \in C$, $m \in M$, using the radio-propagation model introduced in ...???. These predictions already include losses and gains from cabling, hardware, and user equipment.

The amount of power allocated to the pilot signal of cell c is denoted as p_c, where p_c may adopt any value from a finite set of possible pilot power levels, $P_c = \{p_c^1, p_c^2, ..., p_c^K\}$. Consequently, the received pilot power from cell c to mobile m is $L_{cm}^{\downarrow} p_c$.

Service coverage is guaranteed when a mobile has, at least, one cell covering it. Cell coverage is provided to a mobile m from a cell c if its signal-to-interference ratio, $sir(c, m)$, at the RSG pixel where m is located, is not lower than a given threshold, γ^{cov}, i.e.,

$$sir(c, m) = \frac{L_{cm}^{\downarrow} p_c}{\tau_0 + \sum_{i \in C} L_{im}^{\downarrow} p_i} \geq \gamma^{\text{cov}} \tag{5.1}$$

where τ_0 is the thermal noise.

The mathematical model links the SHO settings with CPICH power settings for each cell, the best-server pattern, and the network coverage.

By introducing a change step of 0.01 dB and bounding the CPICH power of a cell c to ± 2 dB, relative to the CPICH power setting the cell had before optimization, the number of elements in the set P_c is reduced. The purpose of this reduction is twofold. First, since the optimization targets a live network, there is no need for the algorithms to create complete new configurations, but just to fine-tune existing ones. Second, the problem complexity is lowered, because the size of the search space is smaller and discrete.

5.3.2 Coverage

A mobile m within the area A_{covered} is under network coverage if at least one cell c covers it. We define the downlink coverage by means of the received signal code power (RSCP) [48]. Following real network settings, and including a margin for interference derived from HSDPA [70], the RSCP threshold is set to -115 dBm ($3.16227766 \cdot 10^{-12}$ mW). So, for any pair (c, m), $c \in C$, $m \in M$, the coverage of mobile m by cell c is defined as

$$cov_{cm} = \begin{cases} 1 & if\ RSCP \geq -115\, dBm \\ 0 & otherwise \end{cases} . \tag{5.2}$$

In case m is covered by several cells, the one with the highest RSCP is refered to as the best server, an denoted as c_m^*.

5.3.3 SHO areas

To obtain a realistic outline of the areas where a mobile may potentially maintain connections to more than one cell, a static version of the active set, as defined in [94], is used. To this end, a SHO window, γ^{sho}, and a maximum active-set size, $as^{\max}$, are introduced. Both parameters are taken from the working configuration of the real network. It follows that the cells to which a mobile $m \in M$ may maintain concurrent downlink connections are part of the set

$$\begin{aligned} SHO_m^{\downarrow} &= \left\{ c | L_{c^*m}^{\downarrow} p_{c^*} - L_{cm}^{\downarrow} p_c \leq \gamma^{\text{sho}} \right\}, \\ |SHO_m^{\downarrow}| &\leq as^{\max}, \end{aligned} \tag{5.3}$$

where $c \in C$, $L_{c^*m}^{\downarrow}$ is the downlink attenuation factor of the best-serving cell, and p_{c^*} is its CPICH power. Since the number of elements in $SHO_m^{\downarrow}$ is at most $as^{\max}$, the weakest links are removed if there are several present. This method is well suited for configurations with no hysteresis, since dynamic effects are ignored in static models [94].

Additionally, in the uplink, the set of cells to which a mobile can potentially be in SHO is defined as

$$SHO_m^{\uparrow} = \left\{ c | L_{mc}^{\uparrow} p_m^{\uparrow} \geq 3.16227766 \cdot 10^{-12} mW \right\}, \tag{5.4}$$

where $L_{mc}^{\uparrow}$ is the uplink attenuation factor from mobile m to cell c, and $p_m^{\uparrow}$ is the uplink transmit power of mobile m.

The static nature of the model intentionally neglects mobility and interference by narrowing γ^{sho} down to 2 dB [94].

5.3.4 Optimization objective

Using the elements defined in Section 5.3, we have constructed an objective function in cooperation with a team of radio engineers of the Radio Network Department at Telekom Slovenije, d.d. The objective function is constructed as a weighted sum, containing different costs that penalize the occurrence of specific SHO conditions in downlink and uplink, which may potentially cause the aforementioned malfunctioning, introduced in Section 5.1.

A cost-based objective function is the most natural and straight-forward way of defining the optimization objective. Besides it is easily extendable to include other future situations, also defining the mutual importance of the different phenomena taken into account at the optimization phase.

Hence, the definition of the objective function for the SHO-balancing problem is the minimization of the sum of penalty scores given as

$$\min f_{\text{sho}} = \sum_{c \in C} \sum_{m \in M} pf_{\text{cov}}(1 - cov_{cm}) + pf_{\text{sho}}^{\uparrow} sho_{cm}^{\uparrow}(1 - sho_{cm}^{\downarrow}) + pf_{\text{sho}}^{\downarrow} sho_{cm}^{\downarrow}(1 - sho_{cm}^{\uparrow}), \tag{5.5}$$

where

$$sho_{cm}^{\downarrow} = \begin{cases} 1 & c \in SHO_m^{\downarrow} \\ 0 & otherwise \end{cases}, \tag{5.6}$$

$$sho_{cm}^{\uparrow} = \begin{cases} 1 & c \in SHO_m^{\uparrow} \\ 0 & otherwise \end{cases}, \tag{5.7}$$

and

- pf_{cov} represents the penalty factor for uncovered areas,

- $pf^{\uparrow}_{\text{sho}}$ represents the penalty factor for uplink SHO areas where SHO is not possible in the downlink, and
- $pf^{\downarrow}_{\text{sho}}$ represents the penalty factor for downlink SHO areas where SHO is not possible in the uplink.

5.4 Optimization algorithms

The SHO-balancing problem has been tackled using three fundamentally different optimization algorithms, namely:

- differential evolution (DE, see Section 2.1.6), from the family of evolutionary algorithms;
- differential ant-stigmergy algorithm (DASA, see Section 2.1.7), from the family of swarm-intelligence algorithms; and
- simulated annealing (SA, see Section 2.1.8), from the group of classic meta-heuristic algorithms, targeted at combinatorial optimization problems.

Each of these algorithms shall minimize the objective function value by adopting essentially disparate approaches, hence the diversity of applying algorithms belonging to different families to solve the same optimization problem. Therefore, the result analysis shall establish which of the presented approaches is better suited for solving the SHO-balancing problem.

The following sections describe how the SHO-balancing problem is represented (or mapped) by the internal structure of each of the selected algorithms and their controlling parameters.

5.4.1 DE mapping

The DE algorithm features a parallel direct search method, which utilizes a population of D-dimensional parameter vectors. The SHO-balancing problem is expressed in each component of a vector X of the population, which maps to the CPICH power of one cell under optimization, i.e.

$$X_{aG} = \{x_1, x_2, \ldots x_c, \ldots, x_D\}\,, \tag{5.8}$$

where $x_c \in P_c$ represents a candidate CPICH power setting of cell c, and G indicates the generation of an individual a in the population. Since there are $|C|$ cells in the mobile network, it follows that $D = |C|$.

From the different variants of DE, the most popular one is used here, called *DE/rand/1/bin*. The nomenclature used to name this variant indicates the way the algorithm works:

- *DE* denotes the differential evolution algorithm,
- *rand* indicates that the individuals selected to compute the mutation values are randomly chosen,

Algorithm 5.1 A move in the search space of SA.

$\mathbf{c}' \leftarrow pick\_random\_cell(C)$
repeat
  if $rand() < 0.5$ **then**
    $p_{c'}^{\text{new}} \leftarrow p_{c'} + 0.01$
  else
    $p_{c'}^{\text{new}} \leftarrow p_{c'} - 0.01$
  end if
until $p_{c'}^{\text{new}} \in P_{c'}$
$p_{c'} \leftarrow p_{c'}^{\text{new}}$

- 1 specifies the number of pairs of selected solutions used to calculate the weighted difference vector, and
- *bin* means that a binomial recombination operator is used.

Four control parameters of the search process for DE were considered: the population size (NP), the number of generations for the algorithm to run (G_{max}), the crossover constant (CR), and the mutation scaling factor (F).

5.4.2 DASA mapping

The mapping between the balancing problem and DASA is similar to the one for DE, depicted in Equation (5.8):

$$X_a = \{x_1, x_2, \ldots x_i, \ldots, x_D\} \tag{5.9}$$

In this case, each ant, a, creates its own solution vector, X_a, during the minimization process. At the end of every iteration, and after all the ants have created solutions, they are evaluated to establish if any of them is better than the best solution found so far.

There are six parameters that control the way DASA explores the search space: the number of ants (m), the discrete base (b), the pheromone dispersion factor (q), the global scale-increasing factor (s_+), the global scale-decreasing factor (s_-), and the maximum parameter precision (e).

5.4.3 SA mapping

From the SA perspective, the system under optimization is in a given *state* at each time step during the process. The objective function maps a system state to a value known as the *energy* of the system in that state. A *move* in the search space represents a change in the state of the system. After making a move, the system may exhibit lower or higher energy, depending on the results of the objective function.

Algorithm 5.1 shows the pseudo-code of a move in the search space of possible CPICH power settings, resulting in a new state of the system.

At the first step, a cell, c', is randomly selected from the set of all cells in the network, C. In step 2, a change of +0.01 dB or -0.01 dB is applied with 50% probability to $p_{c'}$. The CPICH power of cell c' is expressed in dBm. The randomly generated

Table 5.1: Technical characteristics of the test network used.

Number of cells	25
Coverage threshold (RSCP)	−115 dBm
SHO window (γ^{sho})	2 dB
User equipment ($p_m^{\uparrow}$)	21 dBm, power class 4
Pixel resolution	25 m^2
Population density	398/km^2

CPICH power setting, $p_{c'}^{\text{new}}$, is checked for validity in step 3, i.e. it must be an element of the set $P_{c'}$. If $p_{c'}^{\text{new}}$ is not a valid CPICH power, step 2 is executed again, generating another random CPICH power. Finally, in step 4, the CPICH power of cell c is replaced by $p_{c'}^{\text{new}}$.

It is important to note that, as long as $|P_{c'}| > 1$, the pseudo-code shown in Algorithm 5.1 shall never be trapped in an endless loop. On the other hand, if $|P_{c'}| < 2$, there are no candidate CPICH powers for cell c' and thus no possibility of optimization by means of CPICH power adjustment. Notice also that the acceptance of a move in the search space is left to SA and its stochastic components.

SA has two parameters to control the search process: the initial temperature ($t_{initial}$) and the total number of iterations or evaluations (it).

5.5 Simulations

The simulations are performed using a standard Monte-Carlo method, assuming the mobile users are uniformly distributed. The SHO conditions of different users depend on the relative received signal quality from different cells and the SHO window, which triggers the addition of a cell to the user's active set [48].

5.5.1 Test network

The test network used for the simulations is a subset of the real UMTS network deployed in Slovenia by Telekom Slovenije, d.d. It represents a network extending over a hilly terrain, combining both rural and middle-dense suburban areas, which contains 25 cells within an area of more than 150 km^2. Table 5.1 shows some characteristics of the test network used, and Figure 5.2 shows the area under radio coverage, A_{covered}, within A_{total}.

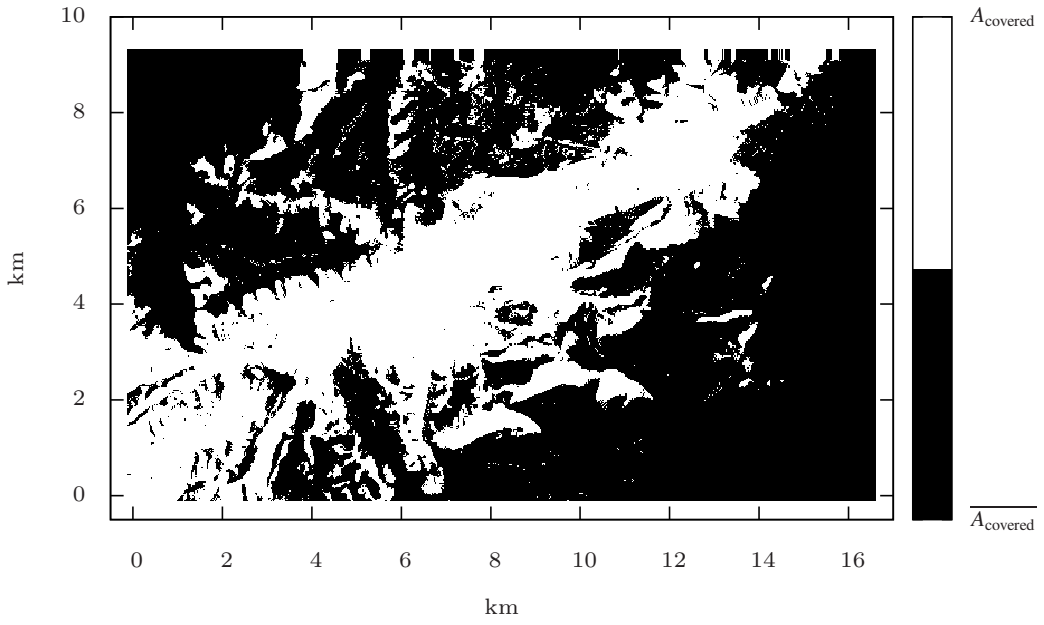


Figure 5.2: Area under radio coverage, A_{covered}, and without radio coverage, $\overline{A_{\text{covered}}}$, within the complete geographical area, A_{total}.

5.5.2 Penalty factors

After extensive experimentation, and working in cooperation with the radio engineers from the Radio Network Department at Telekom Slovenije, d.d., the penalty factors from Equation (5.5) are set to the following values:

- $pf_{\text{cov}} = 15$,
- $pf_{\text{sho}}^{\uparrow} = 13$, and
- $pf_{\text{sho}}^{\downarrow} = 3$.

It is clear that coverage is the most important quality aspect from the network point of view (penalty factor pf_{cov}). Moreover, it imposes the biggest constraint to the optimization process, since the balance between SHO areas should not sacrifice network coverage. Another important characteristic that emerges from these values is the preference for minimizing areas where SHO capability is available in the uplink, but not in the downlink (penalty factor $pf_{\text{sho}}^{\uparrow}$). As it has been described in Section 5.1, consequences of such SHO arrangement produce serious interference rise in neighboring cells (Figure 5.1), which may also result in service inaccessibility. The last factor $pf_{\text{sho}}^{\downarrow}$ imposes a penalty value over areas where SHO capability is available in the downlink, but not in the uplink. Recall that when accessing HSPA services, SHO is available only in the uplink. For this reason, the link throughput may benefit from SHO in the uplink if it is available. The relative lower importance of the last penalty factor

compared with the other ones is directly related to the consequences of such unbalancing of SHO areas may have on the network. In this case only HSPA throughput is affected, while the service accessibility should not be an issue, given there is enough uplink coverage [70].

5.5.3 Algorithm parameters

In this section the algorithm parameter setup, as used during the simulations, is given. In all three cases we have followed the naming conventions as they have been previously introduced in Section 5.4.

5.5.3.1 DE

The parameters controlling the behavior of the DE algorithm have been set as follows:

- $NP = 100$, the population size;
- $G_{max} = 1000$, the maximum number of generations for the algorithm to run;
- $CR = 0.8$, the crossover constant; and
- $F = 0.5$, the mutation scaling factor.

5.5.3.2 DASA

As for DASA, we have set the parameters to the following values:

- $m = 10$, the number of ants;
- $b = 10$, the discrete base;
- $q = 0.2$, the pheromone dispersion factor;
- $s_+ = 0.01$, the global scale-increasing factor;
- $s_- = 0.01$, the global scale-decreasing factor; and
- $e = 1.0^{-2}$, the maximum parameter precision.

5.5.3.3 SA

There are only two parameters controlling SA:

- $t_{initial} = 125$, the initial temperature; and
- $it = 100,000$, the total number of iterations.

SA also allows to define the way the temperature is lowered during the annealing process. In this case, the exponential-lowering schema has been used.

Table 5.2: Solution-quality performance of the three algorithms, after 30 independent runs.

	Best	Worst	Mean	Std. deviation
DE	2,286,292.00	2,286,541.00	2,286,517.09	62.06
DASA	2,286,446.00	2,286,633.00	2,286,592.00	26.19
SA	2,293,350.00	2,295,570.00	2,294,626.50	663.75

5.5.4 Experimental environment

All experiments were carried out on a 4-core Intel i7 2.67 GHz desktop computer with 6 GB of RAM running a 64-bit Linux operating system. The implementation languages used were C and Python, with the latter mostly used as 'glue' to hold the different implementation parts together, as well as for I/O operations. To lower the time needed to run one optimization round, we have implemented the entire objective-function evaluation using OpenCL and executed it on a nVidia GeForce GTX 260. This individual improvement exhibited more than 15x execution time speed-up when compared to the original CPU-only version.

5.6 Results

5.6.1 Algorithm performance

In this section the performance of the selected algorithms is presented. The analysis includes aspects related to solution quality and convergence speed. All experimental results were obtained after 30 independent runs, each of them limited to a maximum of 100,000 evaluations. The gathered results are shown in Table 5.2.

It may be observed that DE reaches the lowest objective function value, closely followed by DASA. Likewise, both algorithms reach very similar results for the worst, mean and standard deviation values. SA, on the other hand, did not achieve comparable values, since its results are behind those of DE and DASA. Notice that even the best SA solution is no better than the worst solution of DASA. Moreover, the standard deviation exhibited by SA is many times bigger to those of DASA and DE, induced by the greater level of variance of its results.

The convergence of the best-recorded run of each of the three algorithms is shown in Figure 5.3. It is worth mentioning that every optimization run starts from a different solution, randomly constructed by picking a CPICH power setting, p_c^k, from every P_c, $1 \leq k \leq |P_c|$, $\forall c \in C$. Notice how fast DASA converges to a good solution. After a number of evaluations without improvement, DASA resets itself and continues searching from a new random point within the search space [32]. Similarly, DE converges considerably fast, although not as fast as DASA does. It this case, DE does not reset itself if the current solution cannot be improved. Despite this, and based on the flat profile the graph exhibits towards the end of the optimization run, we are confident that 100,000 evaluations is an adequate stopping criterion for this algorithm. The third algorithm, SA, slowly converges towards the best solution found, even though it is not as good as the solutions found by DE and DASA.

The three convergence profiles shown in Figure 5.3 give a clearer notion about the way these algorithms explore the search space of the SHO-balancing problem.

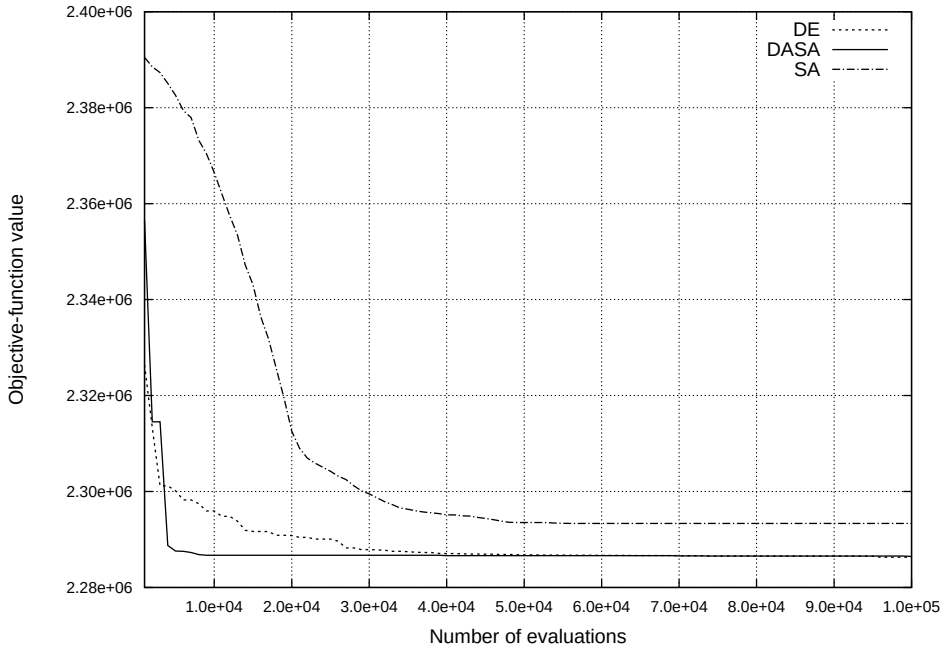


Figure 5.3: Convergence analysis for each algorithm, showing the best results obtained.

The simulation running times have been intentionally omitted, since the implementations used are fundamentally different and therefore not comparable.

5.6.2 Interpretation

Table 5.3 presents the analysis of the obtained results from the network point of view. After 30 independent runs of each of the three algorithms, the best results obtained where evaluated for improvement and decline of each of the measured network aspects. The results are shown in Table 5.3, where '+' indicates improvement and '-' indicates decline of a given criteria. Overall, it may be observed that the measured criteria have been significantly improved. The only exception is the measure labeled as '$SHO^{\downarrow}$, no $SHO^{\uparrow}$', which shows an expected decline, since it is the optimization aspect with the lowest penalty factor value.

Coverage has been improved with an average of 4.29%, whereas the coverage area where there is no SHO capability has been increased 7.74% in average. Areas where SHO is available in both downlink and uplink have also been improved, 3.75% in average. This particular improvement is interesting from the optimization point of view, because it had no explicit penalty factor set. Therefore, it may be understood as a consequence of the criteria completeness the objective function has taken into account.

Table 5.3: Improvement analysis for each of the achieved best solutions.

	Uncovered	Covered, no SHO	SHO	no SHO$^{\downarrow}$, SHO$^{\uparrow}$	SHO$^{\downarrow}$, no SHO$^{\uparrow}$	Total
Before opt.	63.00 %	15.11 %	15.73 %	1.80 %	4.36 %	100.00 %
DE sol.	60.23 %	16.13 %	16.09 %	1.47 %	6.08 %	100.00 %
DASA sol.	60.24 %	16.16 %	16.90 %	1.46 %	5.24 %	100.00 %
SA sol.	60.42 %	16.55 %	15.97 %	1.56 %	5.50 %	100.00 %
DE impr.	+4.40 %	+6.75 %	+2.29 %	+18.33 %	-39.45 %	—
DASA impr.	+4.38 %	+6.95 %	+7.44 %	+18.88 %	-20.18 %	—
SA impr.	+4.09 %	+9.53 %	+1.52 %	+13.33 %	-26.15 %	—
Avg. impr.	+4.29 %	+7.74 %	+3.75 %	+16.85 %	-28.59 %	—

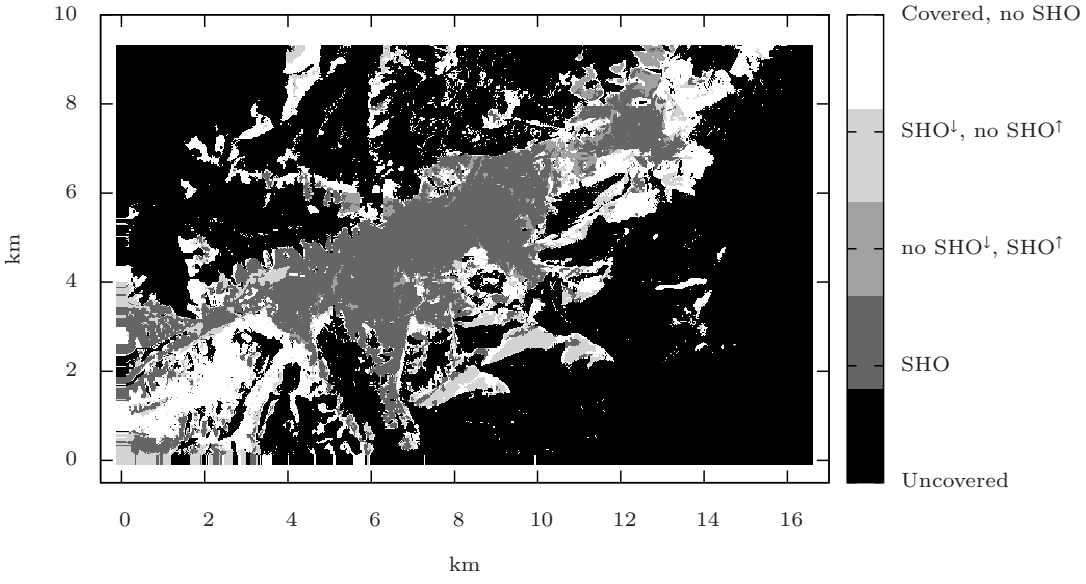


Figure 5.4: Spatial distribution of SHO areas before the optimization.

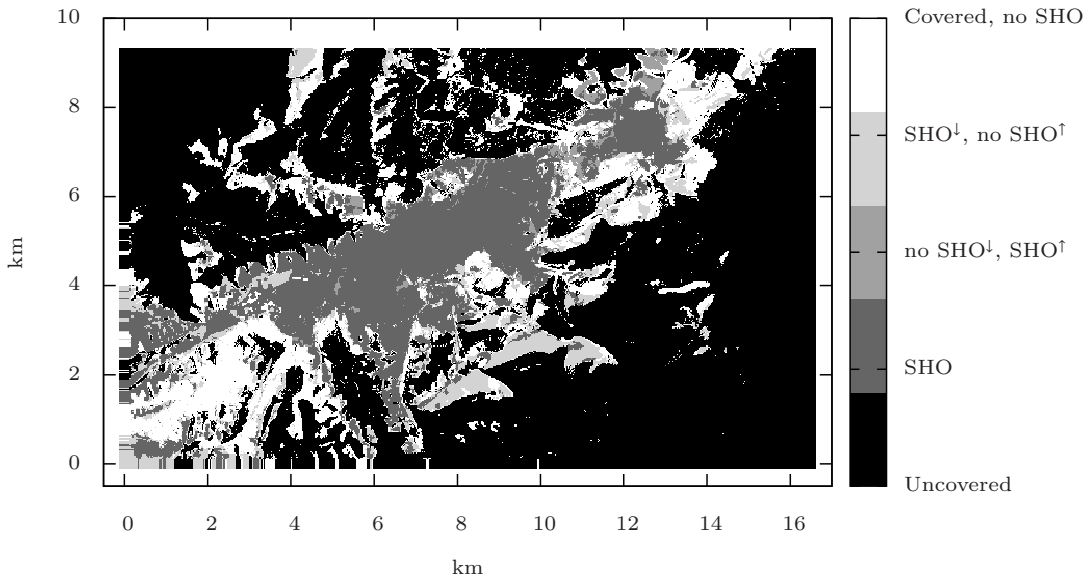


Figure 5.5: Spatial distribution of SHO areas after the optimization.

The second most important optimized aspect in the SHO-balancing problem is the proportion of areas with uplink SHO and no SHO in the downlink (labeled as 'no SHO$^{\downarrow}$, SHO$^{\uparrow}$' in Table 5.3). This particular condition has been improved by almost 17% in average, greatly reducing the possibility of interference in neighboring cells when serving HSPA traffic. The last measured aspect takes into account areas with downlink SHO and no SHO in the uplink (labeled as 'SHO$^{\downarrow}$, no SHO$^{\uparrow}$' in Table 5.3). This condition, although it hasn't improved, does not expose the mobile network to malfunctioning, only to reduced throughput within these specific areas. However, the reduced throughput is relative, since there are many cells capable of serving HSDPA data access, as the downlink SHO condition confirms. For this reason, the serving cell should not only deliver HSDPA, but also take care of the user signaling and power control, received in the uplink. Obviously, this is only feasible in areas where uplink coverage is guaranteed.

It is worth mentioning that the simulation results were obtained for a real mobile network with actual configuration data. Moreover, the hard constraints imposed to the optimization process (CPICH power limited within the ± 2 dB interval) ensure that the resulting configuration may be immediately applied to the mobile network. This fact can be contrasted with the spatial distribution of each of the optimized aspects, before and after applying the optimization results, as it is shown in Figures 5.4 and 5.5. The lack of any prominent visual change in Figures 5.4 and 5.5 is a desired consequence of the fine-tuning procedure the network has been exposed to. Still, the improvements are present precisely over the areas that are most exposed to malfunctioning due to unbalanced SHO, e.g. their borders.

5.7 Summary

This chapter formally introduced a new optimization problem for 3G networks: the SHO-balancing problem. A characterization of the consequences that unbalanced SHO areas have on the quality of HSPA services was also given. Particularly, tackling the SHO-balancing problem was possible due to the improved performance delivered by the evaluation framework PRATO (see Chapter 3).
Using a extension of the state-of-the-art radio-network model presented in ...???, the penalty scores of the objective function were set according to the configuration and layout of a real mobile network, deployed in Slovenia by Telekom Slovenije, d.d.

The balancing problem has been tackled by three optimization algorithms, namely DE, DASA and SA. All three algorithms were able to improve the given network configuration, being DE the most successful one. The presented results confirm that a great proportion of the SHO areas, that were not balanced before the optimization, were correctly balanced, therefore significantly reducing the possibility of HSPA-service failures. Additionally, radio coverage was improved, while all other essential network services were not altered.

One of the key advantages of the presented method is that it targets the optimization of a deployed network, for which the focus is to fine-tune the existing configuration instead of creating complete new solutions. Furthermore, a deployed network has a great number of hard-constraints that should be taken into account at the optimization stage. Yet, the presented approach is simple and versatile enough to be used in practically any working UMTS network. Moreover, the introduced model is applicable for mobile networks in heterogeneous environments, because it imposes no restrictions regarding cell layout or radio-propagation characteristics, which are completely adaptable through PRATO.

6 Framework automatic tuning

The assessment of the radio coverage is essential for network planning and optimization. Consequently, the planning tools used to gather this information play a key role in the decisions made during the planning phase of a radio network. But acquiring the necessary information to support the decision making in this context is a challenging task. Particularly, planning tools have to be adapted for a specific environment and technology in order to improve their accuracy. The complexity of this problem means that radio-coverage prediction is generally a computationally-intensive and time-consuming task, hence the importance of fast and accurate prediction tools.

Within this context, we present an open-source simulation framework for planning and optimization of radio networks. We provide two use cases for the newly deployed LTE network in Slovenia. The first one involves the parameter tuning of an empirical radio-propagation model using a snapshot of field measurements. The other one involves the optimization of clutter losses over different regions of the country, therefore adapting the losses due to land usage to the local conditions of each region.

We report the results of our experimental simulations over three regions of the real LTE network, deployed by Telekom Slovenije, d.d., thus showing the suitability of the parallel framework for tackling real-world planning and optimization problems.

6.1 Introduction

With the advent of long-term evolution (LTE) as the fourth generation (4G) in cellular technology, mobile operators are facing the challenges of deploying a new network. LTE follows the well established universal mobile telecommunication system/high-speed packet access (UMTS/HSPA) combo, targeting higher peak data rates, higher spectral efficiency and lower latency [**?**].

The deployment of a new mobile network is always a challenge for mobile operators, who constantly struggle to find the optimal investment in order to provide a competitive network in terms of coverage and quality of service. Indeed, coverage planning remains a key problem that all operators have to deal with. It has proven to be a fundamental issue since the deployment of the first GSM networks, more than 20 year ago.

One of the primary objectives of radio-coverage planning is to efficiently use the allocated frequency band for a geographic area to be satisfactorily reached with the radio stations of the network. To this end, radio-coverage prediction tools are of great importance as they allow network engineers to test different configurations before physically implementing the changes. However, to accurately predict the radio coverage of a mobile network is a very complex task, mainly due to the wide range

of various combinations of hardware and configuration parameters which have to be analyzed in the context of different environments. The complexity of the problem means that radio-coverage prediction is generally a computationally-intensive and time-consuming task, hence the importance of fast and accurate prediction tools.

Although different mathematical models have been proposed for radio-propagation modeling, none of them excels in a network-wide scenario [**?**]. Empirical propagation models usually give good results with a limited computational effort. However, for improved accuracy, the model parameters have to be adapted to better fit a specific network or region within it, mainly because of inaccuracies in input data and environmental changes in the network region, e.g. foliage of trees or snow. Consequently, a combination of different parameters is generally needed in order to reliably calculate radio-propagation predictions for particular environments. Moreover, since the number of deployed cells (transmitters) keeps growing with the adoption of modern standards [**?**], there is a clear need for a radio propagation tool that is able to cope with larger work loads in a feasible amount of time.

To address the afore-mentioned issues, we adapt the parameters of an empirical propagation model to a set of field measurements. The parameter tuning is analytically calculated per cell, in order to increase the accuracy of the calculated predictions. Moreover, we fine tune the signal losses due to land usage (clutter) in a regional basis, using an optimization approach. As a working framework to tackle the presented problems, we use a parallel radio-prediction tool [**?**], thus showing the suitability of the presented framework for real-world planning and optimization of LTE radio networks. Particularly, we show the tool capabilities to handle several parallel radio-prediction runs using a metaheuristic algorithm and distributed objective-function evaluation, while resolving the clutter optimization problem.

This paper is organized as follows. Section 6.2 gives a description of the parallel simulation framework, including some implementation details and performance figures. Section 6.3 introduces principles of radio-propagation prediction, and the mathematical model used. The parameter-tuning problem and the analytical approach for tackling it are presented in Section 6.4, including the simulations performed on the framework and their results. Section 6.5 concentrates on describing the optimization problem involving the regional adaptation of signal losses due to clutter, including the performed simulations the achieved results. Finally, Section 6.6 gives an overview of relevant publications, describing how they relate to our work, before drawing some conclusions.

6.2 Simulation framework

As if was mentioned before, we use the parallel radio-prediction tool PRATO [**?**] as our simulation and optimization framework. PRATO deals with complex calculations over large data sets by applying a work-pool parallel paradigm over a message-passing communication model. As such, PRATO is ready to be deployed over small groups of networked computers as well as over a computer cluster with hundreds of nodes. Since the prediction calculation employs digital elevation models and land-usage data in order to analyze the radio coverage of a geographic area, the framework is implemented as a module of the open source Geographic Resources Analysis Support System (GRASS) [**?**].

6.2.1 Parallel computation on computer clusters

Considering the high computational power needed for evaluating the radio coverage of real mobile networks during optimization, the use of a computer cluster is preferred. A computer cluster is a group of interconnected computers that work together as a single system. To reach high levels of parallel performance and scalability, PRATO performs the parallel decomposition big data sets, distributing the computational load among the computing nodes that belong to the cluster.

Computer clusters typically consist of several commodity PCs connected through a high-speed local network with a distributed file system, like NFS [**?**]. One such system is the DEGIMA cluster [**?**] at the Nagasaki Advanced Computing Center (NACC) of the Nagasaki University in Japan. This system ranked in the TOP 500 list of supercomputers until June 2012[1], and in June 2011 held the third place of the Green 500 list[2] as one of the most energy-efficient supercomputers in the world.

6.2.2 Multi-paradigm parallel programming

The implementation methodology adopted for PRATO follows a multi-paradigm parallel programming approach in order to fully exploit the resources of each of the nodes in a computing cluster. To effectively use a shared memory multi-processor, PRATO uses POSIX threads to implement parallelism [**?**]. By using POSIX threads, multiple threads can exist within the same process while sharing its resources. For instance, an application using POSIX threads can execute multiple threads in parallel by using the cores of a multi-core processor, or use the system resources more effectively, thus avoiding process execution-halt due to I/O latency by using one thread for computing, while a second thread waits for an I/O operation to complete.

To use the computing resources of a distributed memory system, such as a cluster of processors, PRATO uses the Message Passing Interface (MPI) [**?**]. MPI is a message-passing standard, which defines syntax and semantics designed to function on a wide variety of parallel computers. MPI enables multiple processes running on different processors of a computer cluster to communicate with each other. It was designed for high performance on both massively parallel machines and on workstation clusters. Its development is supported by a broadly-based committee of vendors, developers, and users.

6.2.3 Master-worker model

PRATO follows a master-worker paradigm [**?**], where the main process, i.e. the master, produces many sub-problems, which are delivered to be executed by the worker processes. These sub problems are, in this case, the radio-coverage prediction for individual cells of the radio network under optimization.

The master process is the only component that should be run from within the GRASS environment. It is responsible for dynamically starting the worker processes using the available computing nodes, based on the amount of network cells for which the coverage prediction should be calculated. For distributing the work among the worker processes, the master process dispatches the loaded geographic data, before

[1] http://www.top500.org
[2] http://www.green500.org

dispatching them to the multiple worker processes. In this case, the decomposition of the data applies to the digital-elevation and the clutter data only, but it could be applied to any point-based data set, vector or raster. In the next step, the master process starts a message-driven processing loop, which main task is to evaluate the radio-coverage prediction of different transmitters among idle worker processes.

The worker processes, on the other hand, are completely independent from GRASS, i.e. they do not have to run within the GRASS environment nor use any of the GRASS libraries to work. This aspect significantly simplifies the deployment phase to run PRATO on a computer cluster, since no GRASS installation is needed on the computing nodes hosting the worker processes. During the result-saving phase, each of the worker processes sends its results independently from each other, following an asynchronous and decoupled design. Moreover, worker processes do this from an independent thread, which runs concurrently with the radio-prediction evaluation of the next transmitter received from the master process. The overlap between calculation and communication achieved by the use of an auxiliary thread completely hides the latency created by the result-dumping task, and makes better use of the system resources.

6.2.4 Performance

From the performance point of view, PRATO is capable of reaching levels of high efficiency, as the following measurements show.

Figure 6.1 shows the time measurements of a weak-scalability experiment. To measure the weak-scalability properties of PRATO means analyzing the scalability of the parallel implementation in cases where the workload assigned to each MPI process (one process per processor core) remains constant as the number of processor cores, and thus the total size of the problem, is increased.

The time measurements observed from the weak-scalability results show that the wall-clock times are almost constant for bigger problem instances, revealing that the achieved level of scalability gets close-to-linear as the amount of transmitters-per-core increases. Specifically, PRATO scales especially well when challenged with a big number of cells or transmitters (10,240 for the biggest instance) over 128 cores.

Another aspect is the strong-scalability capabilities of PRATO, which represents the performance when increasing the number of computing cores for a given problem size, i.e. the number of cells deployed over the target area does not change, while only the number of cores used is increased. Figure 6.2 shows the speedup factors achieved for a set of strong-scaling experiments. We may see that a close-to-linear scaling is achieved, especially for the biggest problem sizes. Linear scaling occurs when the obtained speedup is equal to the total number of processors used. However, it should be noted that perfect speedup is almost never achieved, due to the existence of serial stages within an algorithm and communication overheads of the parallel implementation.

For more detailed information about PRATO, its design and capabilities, we refer the reader to [**?**].

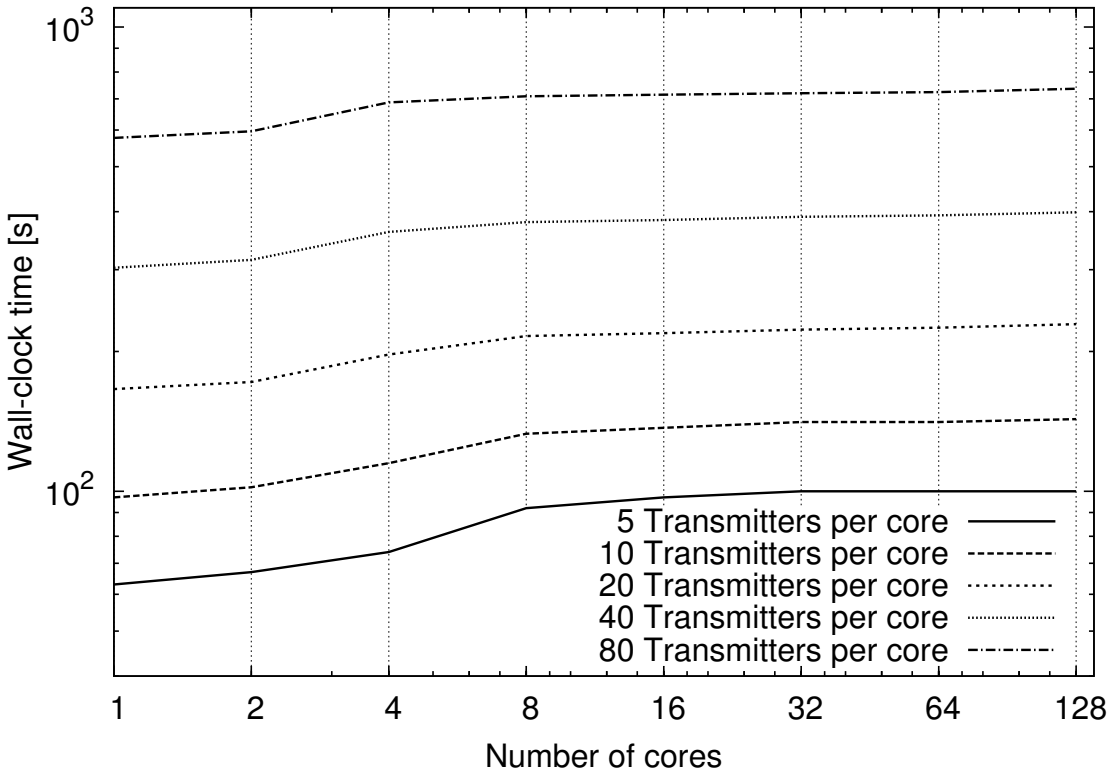


Figure 6.1: Measured wall-clock time for weak-scalability experiments. Experiments performed assigned one MPI worker process per available core. The wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

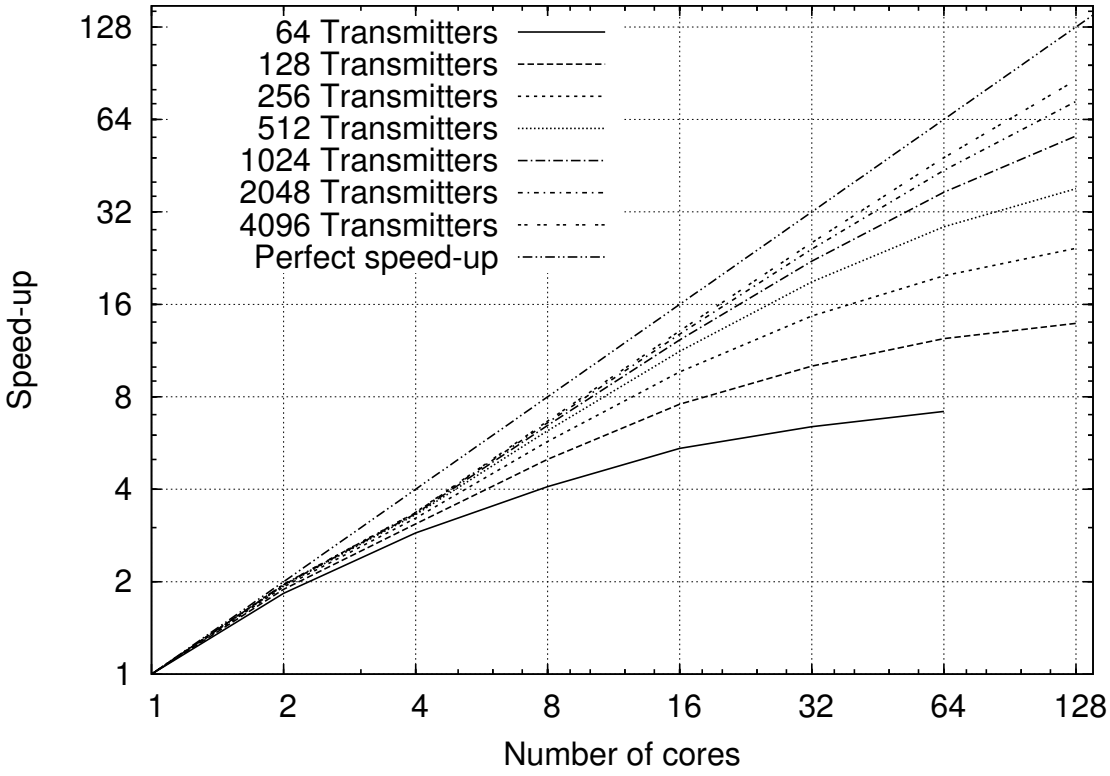


Figure 6.2: Measured speedup for strong-scalability experiments. The speedup axis is expressed in base-2 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

6.3 Radio-propagation prediction

The objective here is to improve the quality performance of a given mathematical model, used for radio-propagation calculations, by fine-tuning its configurable parameters as per-cell basis. In order to do this, we apply a state-of-the-art analytical approach, based on least squares method [**?**], thus combining field measurements to fit the parameters of the mathematical model.

The idea is to automatically adapt the parameters of the mathematical model for each of the cells targeted by the optimization. That is, starting from an a-priori best-known set of parameters, empirically calculated by the radio engineers at Telekom Slovenije, d.d., this approach adapts the model parameters so that the deviation of the radio-propagation prediction to a given set of field measurements is minimized.

To calculate the radio-coverage predictions we use a mathematical model based on the well-known empirical Okumura-Hata formula [**?**]. Other more accurate methods exist, like the ones based on ray tracing [**?**, **?**]. However, these methods are more sensible to deviations in input data, like digital elevation models and buildings, and are still inefficient in terms of the computational effort required to achieve satisfying results.

Empirical methods for radio-propagation predictions, on the other hand, give acceptable results within a feasible amount of time. For this reason, they have become the industry standard for non-deterministic propagation-loss calculations [**?**,**?**,**?**,**?**,84, 109].

6.3.1 Radio-prediction model

The Okumura-Hata model has been largely studied and shown to be suitable for predicting radio propagation of LTE networks [**?**]. Moreover, this mathematical model is especially appropriate for tuning, as it contains a number of adjustable parameters, which adapt the model according to a given scenario and its local conditions. In its primary form, the model distinguishes the distance of receiver from the transmitter, the frequency used and the effective antenna height, i.e. the antenna height above the receiver's level, in order to calculate the path loss in line-of-sight (LOS) conditions. However, the for distinguishing non-line-of-sight (NLOS) conditions, the terrain profile and Earth shape are added to the original formula. Moreover, empirical corrections due to different land usage are also included to improve the quality of the calculated predictions. Equation (6.1) describes the path loss when there is LOS between the transmitter and the receiver.

$$\begin{aligned} pl_{\mathrm{LOS}}(d_{(x,y)},\beta) = a_0 + a_1\log(d_{(x,y)}) + a_2\log(H_A) + \\ a_3\log(d_{(x,y)})\log(H_A) - 3.2\left[\log(11.75\cdot H_R)\right]^2 + \\ 44.49\log(F) - 4.78\left[\log(F)\right]^2, \end{aligned} \quad (6.1)$$

where $\beta = (a_0, a_1, a_2, a_3)$ is the vector containing the tuning parameters of the model, $d_{(x,y)}$ is the distance (in kilometers) from the transmitter to the topography point with coordinates (x, y), H_A is the effective antenna height (in meters) of the transmitter, H_R is the antenna height (in meters) of the receiver, and F is the frequency, expressed

Table 6.1: Clutter-category label numbers and their land-usage meanings for the radio-prediction model.

Clutter category	Description
0	Urban area without buildings, mostly roads
1	Suburban area
2	Urban area
3	Dense urban area
4	Agricultural area
5	Forestall area
6	Swamp area
7	Dry open land area with special vegetation
8	Dry open land area without special vegetation
9	Water area
10	Industrial area
11	Park area

in MHz. On the other hand, in NLOS conditions, the path loss is calculated as in Equation (6.2).

$$pl_{\text{NLOS}}(d_{(x,y)}) = \sqrt{\left[\alpha K(d_{(x,y)})\right]^2 + E(d_{(x,y)})^2}, \quad (6.2)$$

where α is the knife-edge diffraction control parameter, $K(d_{(x,y)})$ is the knife-edge diffraction loss (in dB) and $E(d_{(x,y)})$ is the correction due to the Earth sphere (in dB). The latter two values are calculated on the topography point with coordinates (x, y).

In this work, the terrain profile is used for LOS determination. In order to adequately predict propagation effects due to foliage, buildings and other fabricated structures, additional loss factors based on the land usage (clutter data), are included. This technique is adopted by several propagation models for radio networks [**?**, **?**, **?**]. Consequently, we introduce an extra term for signal loss due to clutter, thus defining the model-predicted path loss as

$$pl(d_{(x,y)}, \beta) = pl_{\text{LOS}}(d_{(x,y)}, \beta) + pl_{\text{NLOS}}(d_{(x,y)}) + pl_{\text{CLUT}}(d_{(x,y)}), \quad (6.3)$$

where $pl_{\text{CLUT}}(d_{(x,y)})$ is clutter loss at the topography point with coordinates (x, y), expressed in dB.

In our case, we recognize twelve different clutter categories representing loss values due to different land usage. These categories, including their label numbers and descriptions, are depicted in Table 6.1.

In any case, the effectiveness of decision-making during radio-network planning is tightly coupled with the precision achieved by the propagation model used. In order to obtain a radio-propagation model that most accurately reflects the propagation characteristics of the area covered by each radio cell in the network, the parameters of the mathematical model are tuned using field-measurement campaigns. Parameter

tuning using this method depends on existing field-measurement data, which are taken before hand over the area covered by the target network cells.

6.4 Parameter tuning of the radio-prediction model

The objective here is thus improve the quality performance of a given mathematical model, used for radio-propagation calculations, by fine-tuning its configurable parameters as per-cell basis. In order to do this, we apply a state-of-the-art analytical approach, based on least squares method [**?**], thus combining field measurements to fit the parameters of the mathematical model.

The idea is to automatically adapt the parameters of the mathematical model for each of the cells targeted by the optimization. That is, starting from an a-priori best-known set of parameters, empirically calculated by the radio engineers at Telekom Slovenije, d.d., this approach adapts the model parameters so that the deviation of the radio-propagation prediction to a given set of field measurements is minimized.

In the context of our work, this translates to fine tuning the parameters of the LOS part of the path-loss prediction model, i.e. $pl_{\mathrm{LOS}}(d_{(x,y)}, \beta)$. The adjustable parameters are the elements of vector $\beta = (a_0, a_1, a_2, a_3)$, namely

a_0 the reference loss or offset;

a_1 the loss slope due to distance of the receiver from the transmitter;

a_2 the loss slope due to height of the transmitter antenna;

a_3 the loss slope due to the combined effect of the distance and height of the antenna.

The parameter tuning is performed per cell to improve local fitting of the radio predictions. The steps to be completed are: collect and process field-measurement data, and solve the particular system of linear equations. The resulting solution is the vector β of the target cell, with its values locally adapted.

In the following, we denote the default parameters of the radio-propagation model with these values

a_0 38.0,

a_1 32.0,

a_2 -12.0, and

a_3 0.1.

6.4.1 Field measurements

In mobile networks, a moving mobile constantly performs cell selection/reselection and handover in order to keep the best possible connection to the network. Within this context, the best connection is selected by measuring the signal strength or quality of the neighboring cells. In LTE networks, a mobile measures two parameters on reference signal, namely the Reference Signal Received Power (RSRP) and the Reference Signal Received Quality (RSRQ).

For a certain frequency bandwidth, RSRP measures the average received power over the resource elements that carry cell-specific reference signals. RSRP is applicable in idle (e.g. waiting for a call) and connected (e.g. during a call) modes. During the procedure of cell selection/reselection in idle mode, RSRP is used. On the other hand, RSRQ is only applicable when the mobile is in connected mode.

The radio-coverage calculation involves predicting the network coverage over a certain region, and thus to the users' mobiles within it. Hence, in the first place, we are interested on accurately predicting the best connection the mobile would select in idle mode and the RSRP field measurements it uses.

In our case, the field measurements, representing the RSRP at a given location, are collected using a small truck equipped with the spectrum analyzer Rohde & Schwarz. The spectrum analyzer is connected to an external omni antenna mounted on the roof of the truck, at roughly 2 meters above the ground, taking measurements at a rate of 50 Hz???, with the symbol rate set to ??? Mhz. To accurately establish the measurement location points, a GPS unit was used. The measurement locations covered most of the streets within the target area, with over 300,000 individual field-measurement points taken at more of more than 150 network cells.

To minimize the error impact in measured RSRP and estimated location, all field measurements are processed so that a single value, the median[3], is calculated for each of the measured locations. The resulting RSRP is then used to estimate the path-loss prediction at the location corresponding to each of the measurements, which resolution matches those of the digital elevation model and clutter data.

6.4.2 Linear least squares

It is important to note the linear relationship between the predicted loss $pl(d_{(x,y)}, \beta)$ and the components of vector β, for this is a necessary condition for the least-squares method to find a solution. Following this method, the parameter fitting of the propagation model is based on the minimization of an observed error, i.e. the sum of the squared difference between the predicted and measured RSRP,

$$E(c) = \sum_{i=1}^{m_c} (p_c - pl_c(d_{(x,y)}, \beta_c) - fm_i)^2, \quad (6.4)$$

where $E(c)$ is the observed error for cell c, p_c is the transmit power of cell c, and fm_i is the i-th field measurement out of a set of measurements of cell c with cardinality m_c.

6.4.3 Simulations

The simulations carried out for this part of our work comprise building the matrices of observed-error values, $E(c)$, for each cell c in the target network. The linear systems of equations are then individually solved by applying the linear least squares method [**?**], which involves the evaluation of one radio-coverage prediction per cell. Each system holds a unique solution for each cell c, denoted by the vector β_c.

[3]The median is the numerical value separating the higher half of a sample set from the lower half.

Table 6.2: Percentage of clutter-category proportions for each of the test networks used. The category legend is given in Table 6.1.

	Cat. 0	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Cat. 6	Cat. 7	Cat. 8	Cat. 9	Cat. 10	Cat
Net_1	0.53	4.53	1.68	0.45	71.89	17.94	0.07	0.00	0.03	2.21	0.67	0.0
Net_2	0.91	5.53	9.48	3.84	29.73	48.57	0.14	0.03	0.03	0.76	0.86	0.1
Net_3	0.15	3.99	1.14	0.11	26.50	67.13	0.26	0.00	0.00	0.36	0.36	0.0

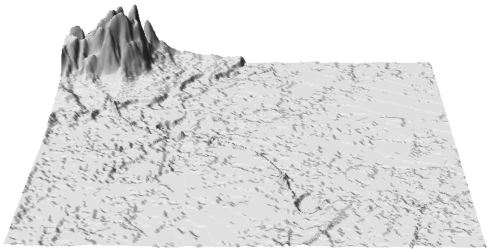


Figure 6.3: Terrain profile for network Net_1, dominated by an agricultural area.

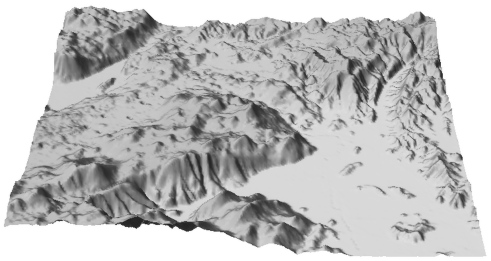


Figure 6.4: Terrain profile for network Net_3, dominated by hills.

6.4.3.1 Test networks

All the test networks, Net_1, Net_2, and Net_3 are subsets of a real LTE network deployed in Slovenia by Telekom Slovenije, d.d. The path-loss predictions are calculated using a digital elevation model and clutter map of 25 m^2 resolution and a receiver height of 1.5 m above ground level. A radius of 16 km is taken around each network cell, thus limiting the path-loss prediction to a distance where the it is still feasible for the mobile to connect to a cell, with a RSRP greater or equal to -124 dBm [**?**]. At the same time, the selected calculation radius provides enough overlap among neighboring cells to also calculate the network coverage over the whole region. Table 6.3 provides more information about the test networks used.

Net_1 represents a network deployed over a dominant agricultural area with almost flat terrain, some forests and waters streams. Net_2, on the other hand, is deployed over a densely populated urban area, containing high buildings, parks and avenues. The last network, Net_3, represents a network deployed over hilly terrain, including some smaller villages and vast forests. It is important to note that the number of deployed cells is directly related to the population density within the region of each test network, as the number of cells from Table 6.3 show. For a clearer representation of the test networks used, we present the proportion of each land-usage (clutter)

Table 6.3: Several properties of the test networks used for the experimental simulations.

	Number of cells	Area (km^2)	Field-measurement proportion (%)
Net_1	12	103.74	5.41
Net_2	130	1298.02	12.02
Net_3	6	386.38	2.30

category in Table 6.2.

The terrain profiles are most relevant for Net_1 and Net_3, since none of them comprehends an urban environment. Note that the terrain shown in Figure 6.3 is mostly flat, since the agricultural area is predominant there. In contrast, the terrain for Net_3 is dominated by hills, which are mostly covered by dense forests, with some small villages in the valleys (see Figure 6.4).

6.4.3.2 Experimental environment

The simulations were carried out on 40 computing nodes of the DEGIMA cluster [?] at the Nagasaki Advanced Computing Center (NACC) of the Nagasaki University in Japan. This system ranked in the TOP 500 list of supercomputers until June 2012[4], and in June 2011 held the third place of the Green 500 list[5] as one of the most energy-efficient supercomputers in the world.

The computing nodes are connected by a LAN, over a Gigabit Ethernet interconnect. The reason for using a high-end computer cluster as DEGIMA is to exploit the parallel nature of PRATO. However, this does not mean that the presented simulation cannot be carried out serially in a commodity desktop computer. PRATO does support serial calculation of radio-propagation predictions for several cells.

Each computing node of DEGIMA features one of two possible configurations, namely:

- Intel Core i5-2500T quad-core processor CPU, clocked at 2.30 GHz, with 16 GB of RAM; and
- Intel Core i7-2600K quad-core processor CPU, clocked at 3.40 GHz, also with 16 GB of RAM.

During the simulation runs, the nodes equipped with the Intel i5 CPU host the worker processes, whereas the master process runs on a computing node featuring an Intel i7 CPU, and performing all its I/O operations on the local file system.

All the nodes are equipped with a Linux 64-bit operating system (Fedora distribution). As the message passing implementation we use OpenMPI, version 1.6.1, which has been manually compiled with the distribution-supplied gcc compiler, version 4.4.4.

6.4.3.3 Results

The results of applying the linear least squares method to fit the parameters of the radio-prediction model to a set of field measurements presented in this section. We have prepared bar charts showing the cumulative distribution of the absolute error between the prediction points and the field measurements. Each bar represents an open interval, expressed in dB, denoting the proportion of points that deviate from the prediction for the expressed number of dB. For example, in Figure 6.5, we may see that the proportion of predicted points differing from the field measurements in 35 dB or more is around 16%, whereas the proportion differing in less than 5 dB is 10%. These numbers represent the test network Net_1 before applying the model-parameter fitting. For comparison, in Figure 6.6, the absolute-error distribution for

[4]http://www.top500.org
[5]http://www.green500.org

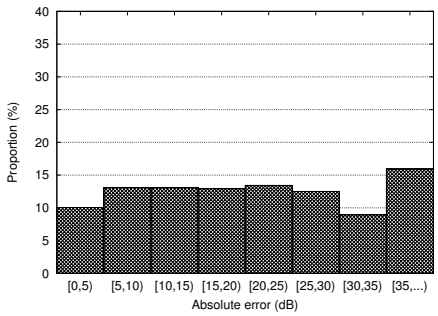


Figure 6.5: Error distribution of the radio prediction for network Net_1 with default parameter values.

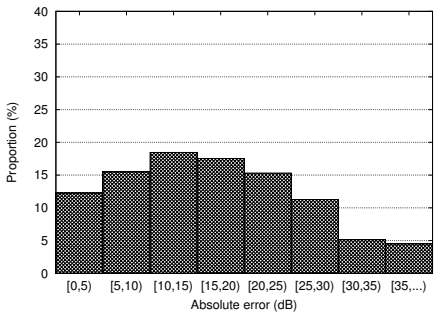


Figure 6.6: Error distribution of the radio prediction for network Net_1 with fitted parameter values.

the same test network is given, but with the model parameters fitted to the available field measurements. Notice how the proportions describing the biggest deviation have dropped to under 5% (35 dB and more) and less than 6% (30 dB to 35 dB). Moreover, it is clear how all proportions improved, raising the bars towards the left-hand side.

The error distributions of the radio-propagation prediction for test network Net_2 using default parameters and fitted ones are given in Figures 6.7 and 6.8, respectively. In this case, the improvement is even more significant than for the first test network, clearly showing that the tuned propagation model represents the local radio propagation conditions within this environment more accurately.

For the third test network, Net_3, the error distributions are depicted in Figure 6.9 using the default parameters, and Figure 6.10 for the tinned ones. Similarly as for the first test network, Net_1, we may see a how the proportion of highest error deviation has been lowered over those with lower deviation values.

The overall results confirm that fitting the parameters of the radio-propagation model to the field measurements within a certain region significantly improve the quality of the calculated propagation prediction.

Despite the presented results are encouraging, there are some specific reasons behind the considerable better results for Net_2, when compared to those of Net_1 and Net_2. Clearly, the number of available field measurements directly affects the quality of the calculated results, making the least squares approximation rougher and thus less precise (see Table 6.3 in Section 6.4.3.1). However, it is out of the scope of this work, in which we focus is on the applicability of PRATO as a tool for planning and optimization of LTE networks, to assess the quality of the achieved results. Nevertheless, for the sake of completeness, we would like to point out that some researchers have already started working on different ways on how to improve this aspect [?, ?].

6.5 Clutter optimization

In order to better adapt the radio-prediction calculation to a given regional environment, the signal losses due to clutter have to be optimized. As it was mentioned before, there are several reasons for the signal loss values to be inaccurate. Among them, we can mention seasonal changes, like tree foliage or snow, construction or demolition of buildings and parks, different kinds of forests or agricultural areas. These

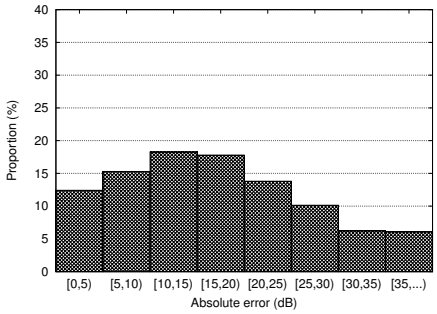


Figure 6.7: Error distribution of the radio prediction for network Net_2 with default parameter values.

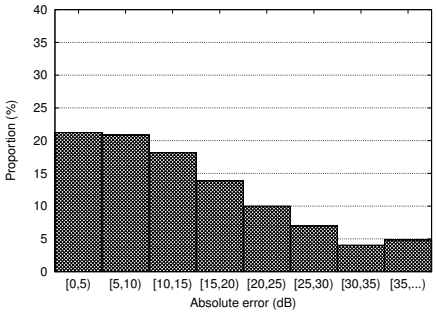


Figure 6.8: Error distribution of the radio prediction for network Net_2 with fitted parameter values.

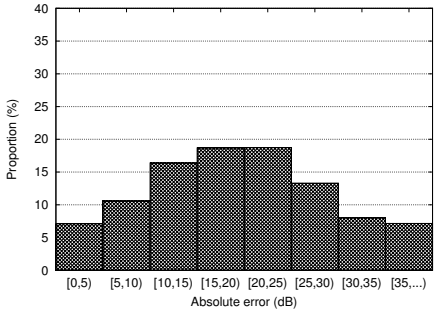


Figure 6.9: Error distribution of the radio prediction for network Net_3 with default parameter values.

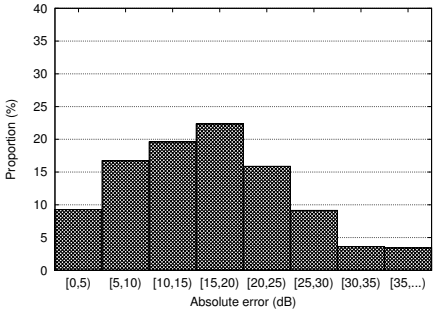


Figure 6.10: Error distribution of the radio prediction for network Net_3 with fitted parameter values.

changes are only noticeable through regular field-measurement campaigns and land-usage data updates. The challenge is thus to combine both in order to improve the prediction results even further.

In the following, we use a metaheuristic algorithm in order to optimize the clutter losses, i.e. the signal loss due to terrain influence, of several regions within the mobile network under optimization. This is done over a group of network cells within different regions of the target network, e.g. agricultural, urban or hilly. In terms of radio-network planning, a regional classification of signal loss due to clutter further improve the accuracy of the radio-coverage prediction, thus enhancing the model by distinguishing, for example, different types of agricultural, park and vegetation areas.

Compared to the parameter tuning of the mathematical model presented in Section 6.3.1, we are not using an analytical approach for tackling this problem, since we wish to potentially include corrections for possible errors in the NLOS part Equation 6.3. For this reason, we turn our attention to metaheuristic algorithms [1] in general and swarm intelligence in particular [**?**]. From this last family of metaheuristic algorithms, we choose the differential ant-stigmergy algorithm (DASA) [**?**].

Several authors have shown the benefits of using metaheuristic algorithms for tackling radio-related problems [**?**,**?**,**?**,98]. In our case, an analytical approach may also be possible, but our objective is to show the capabilities of PRATO when combining an optimization algorithm with parallel and distributed objective-function evaluation. For the following simulations, the DASA runs as the master process, whereas the workers evaluate the current solution in parallel for all the involved cells (see Figure 6.11).

6.5.1 Optimization objective

The optimization objective consists of adjusting the loss values of the different clutter categories, i.e. $pl_{\mathrm{CLUT}}(d_{(x,y)})$ of Equation (6.3), to best fit a set of field measurements in a given region containing multiple cells. We have three data sets at our disposal, one for each of the three regions: the first for Net_1, the second for Net_2, and the third for Net_3. Each region is independently optimized, so that the radio-propagation predictions of its network cells, which already have their model parameters fitted, minimize the mean-squared error against the field measurements, namely

$$F^*(R) \quad = \quad \sum_{i=1}^{m_c} \frac{(p_c - pl_c(d_{(x,y)}, \beta_c) - fm_i)^2}{m_R} \quad \forall c \quad \in \quad R, \quad (6.5)$$

where $F^*(R)$ is the optimization objective to be minimized, R is one of the three regions, p_c is the transmit power of cell c, fm_i is the i-th field measurement out of a set of measurements of cell c with cardinality m_c, m_R is the number of field measurements of all the cells within the region R, and $pl_c(d_{(x,y)}, \beta)$ is the path loss of cell c at coordinate (x, y), where β contains per-cell fitted parameter values of the prediction model.

For a reference of the different clutter categories our prediction model recognizes, see Table 6.1.

6.5.2 Differential ant-stigmergy algorithm

As it has been mentioned before, the optimization algorithm we have chosen for the clutter-optimization problem is the (DASA). Based on the metaheuristic Ant-Colony Optimization (ACO) [**?**], the DASA [**?**] provides a framework to successfully cope with high-dimensional numerical optimization problems. It creates a fine-grained discrete form of the search space, representing it as a graph. This graph is then used as the walking paths for the ants, which iteratively improve the temporary best solution.

The mapping between the clutter-optimization problem and DASA is as follows:

$$X_a = \{x_1, x_2, \ldots x_i, \ldots, x_{12}\} \tag{6.6}$$

where X_a is the solution vector of ant a during the minimization process, and x_i represents the i-th clutter category within a given region. At the end of every iteration, and after all the ants have created solutions, they are evaluated to establish if any of them is better than the best solution found so far.

There are six parameters that control the way DASA explores the search space: the number of ants, the discrete base, the pheromone dispersion factor, the global scale-increasing factor, the global scale-decreasing factor, and the maximum parameter precision.

For a more in-depth explanation about these parameters and the DASA algorithm itself, we refer the reader to [**?**].

6.5.3 Simulations

In this case, the simulations for each test network comprise multiple iterations of several steps. An iteration begins by generating a solution vector for each of the ants in the DASA population. The following step involves the parallel evaluation of each solution, i.e. one radio-coverage prediction per cell. The final step is calculating the objective-function value, as defined in Equation (6.5), before sending it to the DASA for it to generate the next set of solutions. Figure 6.11 depicts the way PRATO performs the objective-function evaluation in parallel over the worker processes, while the optimization algorithm runs on the master one.

Compared to the experiments presented in Section 6.4.3, a much higher number of evaluations is needed for this kind of optimization. In this context, it is of key importance to perform the radio-coverage prediction for multiple cells in a parallel manner. Otherwise, such an approach would not be feasible, since the time required to reach a reasonable solution would be excessive.

The loss value (in dB) each clutter category may take has been limited to the interval [0,40]. This information has been provided by the radio experts of the Radio Network department at Telekom Slovenije, d.d.

As stopping criteria of the optimization runs for Net_1 and Net_3, we have fixed the maximum number of iterations to 200, whereas for Net_2 this values has been set to 500 iterations, since this networks comprises the largest number of network cells. Overall, the framework completed 48,000 objective-function evaluations in the former case, and 120,000 in the latter one.

Regarding the parameters that control algorithm behavior, we have set them to the following values

- $m = 240$, the number of ants;

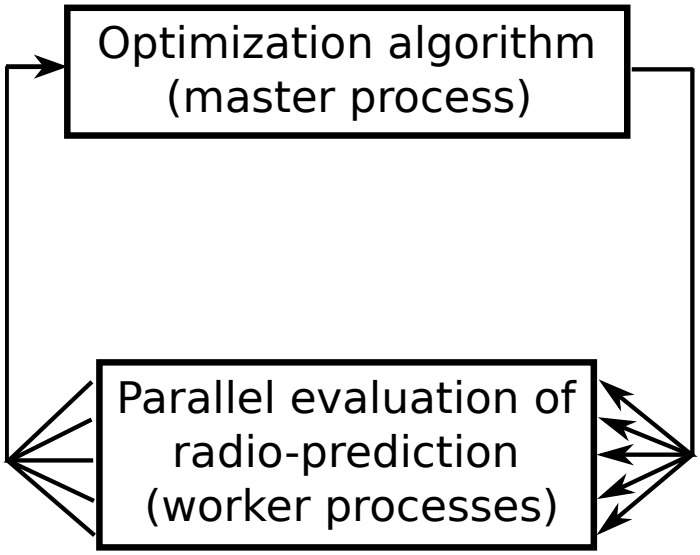


Figure 6.11: *PRATO architecture and data flow during the clutter-optimization phase.*

- $b = 10$, the discrete base;
- $q = 0.2$, the pheromone dispersion factor;
- $s_+ = 0.01$, the global scale-increasing factor;
- $s_- = 0.01$, the global scale-decreasing factor; and
- $e = 1.0^{-2}$, the maximum parameter precision.

6.5.4 Results

The results calculated by the optimization process are shown in Table 6.4. The solutions are given for each of the test networks, along with typical default loss values due to clutter. Hyphens represent clutter categories for which there are no field measurements available. Consequently, it is not possible to calculate an objective-function value for them.

The optimized loss for the first clutter category, 0, representing urban area without buildings, is larger than the default value in all three networks. This may be attributed to the fact that ...??? As for the category 1, representing suburban area, the value for Net_1 is lower than the default one, mainly because this network is deployed over a predominant agricultural area, i.e. suburban areas are less dense here. On the other hand, the value for Net_2 is larger, indicating a building density above the average, whereas for Net_3, the value could not be calculated due to lack of measurements. A similar behavior may be observed for category 2, representing urban area. However, for category 3, representing dense urban area, the optimized losses of all three networks are lower than the default value, indicating dense urban areas here are not as dense as the average case. Representing agricultural area, category 4 gets a value very close to the default one for Net_1 and Net_3, whereas for Net_2 the value is lower, indicating the most of this kind of land is not being exploited near the city. As for category 5, representing forests, the results are well corresponded with the kind of forest dominating each of the test networks, being those of Net_1 and Net_3 more

Table 6.4: Clutter-category losses after the optimization. The default losses for each clutter category are given along the solutions for each of the test networks. All values are expressed in dB.

Clutter category	Default	Net_1	Net_2	Net_3
0	5.0	13.71	11.30	17.90
1	15.0	12.39	16.67	-
2	13.0	16.04	17.04	15.69
3	28.0	19.59	18.01	23.00
4	12.0	11.48	9.71	10.80
5	20.0	16.26	11.62	16.26
6	15.0	-	-	-
7	8.0	-	13.49	-
8	5.0	-	13.50	-
9	1.0	17.50	5.60	-
10	20.0	8.26	16.75	16.63
11	8.0	-	18.93	-

dense due to leave foliage, whereas as for Net_2 the forests are mostly coniferous and more sparse. Keeping the default loss values for categories 6, 7 and 8, we move on to category 9, representing water, for which the results of all three networks indicate creeks and rivers in these areas are mostly surrounded by forests (Net_1) or buildings (Net_2), since none of the regions lays by the sea. As for the industrial areas, denoted by clutter category 10, show lower loss values than the typical default, indicating very sparse industrial buildings (Net_1) and a higher density of mostly commercial buildings for Net_2 and Net_3).

These results clearly show the benefit of the optimization of losses due to land usage or clutter, by finely adapting these values to local conditions within the environment where these networks are deployed.

6.5.4.1 Statistical analysis

Because of the stochastic nature of the optimization algorithm, we have collected the results from 30 independent runs, in order to have enough data for the results to be statistically relevant. In other words, the robustness of the solutions presented in the previous section is analyzed here.

To this end, Table 6.5 shows the solutions reached by the DASA for each of three test networks. The calculated signal losses are thus depicted with the minimum, maximum and average values for every clutter category, along with their standard deviation. Similarly as before, hyphens represent clutter categories for which there are no field measurements available, and thus they cannot be optimized.

6.6 Related work

There are a few examples of radio-network simulators available for LTE [?,?,?], mostly developed for academic environments, they are not targeted at real-world environments. A Matlab-based LTE simulator has been proposed in [?]. It implements a

Table 6.5: Statistical analysis of the optimization solutions for each test network. All values are expressed in dB.

	Net_1				Net_2				Net_3			
Category	Min	Max	Avg	St.dev.	Min	Max	Avg	St.dev.	Min	Max	Avg	St.d
0	13.36	13.97	13.71	0.15	11.22	11.40	11.30	0.04	17.72	17.85	17.90	0.
1	-	-	-	-	14.25	19.20	16.67	1.87	-	-	-	
2	15.84	16.21	16.04	0.08	16.99	17.11	17.04	0.03	15.64	15.72	15.69	0.
3	19.15	20.07	19.59	0.25	17.95	18.12	18.01	0.04	22.68	23.20	23.00	0.
4	11.35	11.62	11.48	0.05	9.63	9.77	9.71	0.03	10.73	10.84	10.80	0.
5	16.00	16.54	16.26	0.14	11.45	11.80	11.62	0.08	16.19	16.30	16.26	0.
6	-	-	-	-	-	-	-	-	-	-	-	
7	-	-	-	-	12.71	14.52	13.49	0.39	-	-	-	
8	-	-	-	-	12.25	15.79	13.50	0.83	-	-	-	
9	16.07	18.80	17.50	0.83	4.67	6.26	5.60	0.35	-	-	-	
10	7.79	8.54	8.26	0.19	16.68	16.87	16.75	0.05	16.50	16.68	16.63	0.
11	-	-	-	-	18.62	19.20	17.04	0.13	-	-	-	

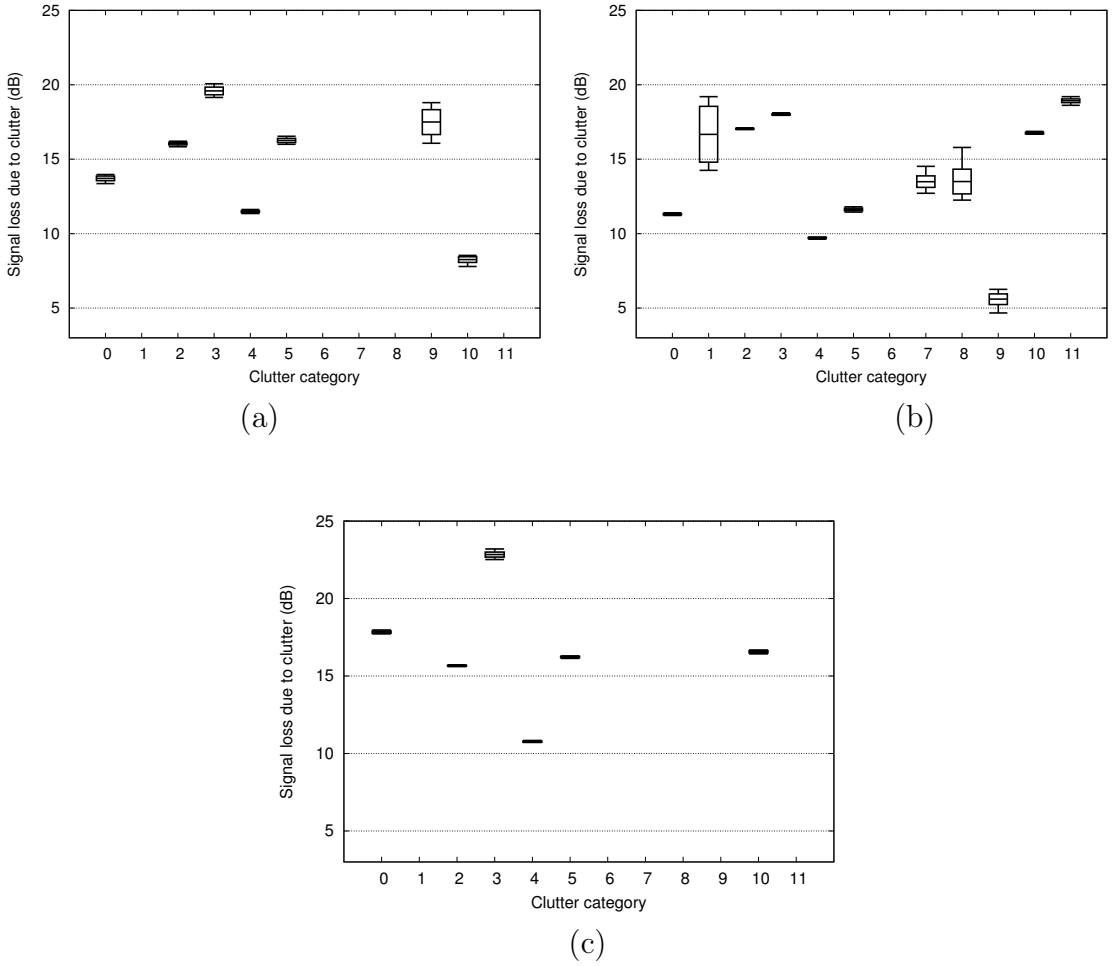


Figure 6.12: Box plots showing the statistical analysis values of Table 6.5, showing the solutions of the clutter-optimization process for each test network: (a) Net_1; (b) Net_2; (c) Net_3.

standard LTE downlink physical layer, including Adaptive Modulation and Coding (AMC), multiple users, MIMO transmission and scheduler. Despite being open source and freely available, the fact of being implemented in Matlab make it very restrictive in terms of tackling bigger problem instances of real networks. A promising tool in this sense is presented in [**?**], where the authors implement a full-stack LTE system in C++. Although the tool has no capability for graphically displaying the simulations, it could be implemented, since the source code is available. In our opinion, the main drawback of this tool is the lack of documentation, which makes it very time-consuming to continue extending this work without the direct help of one of the original authors. In [84], the authors present a tool for radio-coverage calculations of wireless networks. Implemented as a module of the open-source GRASS system, it made an ideal basis for expanding it with parallel-computation capabilities.

As an extension to the well-known NS-2 network simulator, Filiposka and Trajanov [**?**] introduced a module for radio-propagation predictions, which takes the terrain profile into account. In this case, the authors focus on the relief and leave out signal loss due to land-usage, which is a key factor for acquiring more realistic radio-propagation predictions.

Following an optimization-oriented approach [**?**], the authors study the effects of the location accuracy while performing a semi-automatic optimization of the parameters of a radio-propagation model. When compared to our work, where we take a fully automatic optimization approach, the advantage is clear, since no human intervention is needed during the optimization process. Besides, they conclude that locations with a median accuracy of around 60 mts, may be used for parameter tuning. In this sense, the GPS-informed location of the field measurements we had available, has been tested to be within this limit.

6.7 Sumary

We have presented an open-source simulation framework for planning and optimization of radio networks (PRATO). Based on extensive experimental simulations, we have shown the suitability of PRATO by tackling two planning and optimization cases, tested over the newly deployed LTE network in Slovenia. The first one involved the parameter tuning of an empirical radio-propagation model using a snapshot of field measurements. The other one consisted in optimizing the clutter losses over different regions of the country, therefore adapting the losses due to land usage to the local conditions of each region.

The encouraging results indicate that PRATO is applicable in planning and optimization of real-world radio networks in general, and LTE in particular. Moreover, even computational intensive tasks such as stochastic optimization, involving thousands of radio-prediction evaluations, are feasible thanks to the parallelization capabilities provided by the framework.

Acknowledgments

The authors would like to especially thank the staff at the Nagasaki Advanced Computing Center (NACC) of the Nagasaki University for their support while making the

computer cluster DEGIMA available for the experimental simulations used in this work. We are also grateful with the radio engineers at Telekom Slovenije, d.d., for supplying the network data sets and sharing their professional expertise throughout the creation of this work.

This project was co-financed by the European Union, through the European Social Fund.

7 Performance assessment within real network-planning scenarios

In this chapter, real-world, network-planning activities on different radio networks are presented. The objective is to compare PRATO with a radio-prediction enterprise tool in terms of quality and performance. To this end, the engineers of the Radio Network department at Telekom Slovenije, d.d., provided us with their radio-prediction industrial software and guidelines to perform some typical radio-planning activities.

7.1 Measurements and simulation comparison

We have also developed two modules for the evaluation of the accuracy of the designed radio coverage prediction tool with respect to the actually measured results. The first module, db.CompareResults, compares the calculated values with the values obtained during a field measurement campaign. The module reads each row of the text file with field measurement data and saves the x and y coordinates, the measured signal level and the name of the cell. The module first maps the coordinates to the nearest coordinates in the database (as they do not necessarily coincide), finds the corresponding row in the database and then extracts the received signal level for the required cell. The data for the measured and calculated signal level, their difference in decibels along with the location, the used path loss prediction model and the cell name are finally written to the output text file (Table 3).

The second module, r.EvaluateSimulations, compares the field measurements and the strongest simulation signal levels, neglecting the information about the serving cell. The module output is a textual file including the same fields as the output of the db.CompareResults module. The module enables comparison of field measurements with simulation results calculated with the designed radio coverage prediction tool or the commercial tool TEMS. Because of different input raster files (a GRASS raster file includes values for maximal received power while a TEMS raster file contains path loss values) additional selection is done with the "GRASS MaxPower raster". In the case of the TEMS input raster file, an average transmitted power value must be entered for the receive power calculation.

7.2 Coverage-prediction performance analysis

The performance and accuracy of the developed modules for radio signal coverage prediction was investigated by comparing simulation results and field measurements. The reference values were obtained by comparing field measurements with simulation

results acquired from the professional radio signal coverage prediction program TEMS. In both simulation tools we used the modification of the Okumura-Hata propagation model [23].

The performance of the new software package was investigated for different types of networks (GSM, UMTS) and terrains (hilly and almost flat rural, urban, and sub-urban). The evaluation of the developed software for different frequency bands is presented first, followed by an analysis for a different terrain type.

The accuracy of the GRASS prediction software can be verified from the charts on Fig. 6, 7 and 8. On the left side, the charts comparing the measurements and calculations with the GRASS radio coverage prediction software are depicted, while graphs showing the comparison between the measurements and calculations with the TEMS software package are on the right.

Fig. 6 and 7 are presenting the simulation results from both tools and the field measurements in suburban environment for 900MHz and 2040MHz. Received power charts clearly shows that the simulation results match the measured values rather well. Slightly better agreement can be perceived in the 900MHz frequency band (Fig. 7). The deviation among the measurements and simulations for both software applications is depicted in the second raw of diagrams in Fig. 6 and 7. It is evident that the difference between the diagrams on the left- and on the right-hand side for both frequencies is minor. Thus, it can be concluded that the results from the developed radio coverage tool are comparable with the results from the TEMS application and are independent from the used frequency band.

Additional analyses were done on different terrain types. The analyses for the flat rural environment are depicted in Fig. 8. The curves on the charts showing the difference between the measurements and simulations for both software applications have similar course. This confirms applicability of the developed software also for arbitrary terrain types.

The developed radio coverage software gives similar results as the professional TEMS software irrespective of the operational frequency or chosen terrain type. The computed values are comparable also for different distances between the base station and the receiver. Some negligible differences between the results originate from the fact that the implemented path loss model in the TEMS software used in the simulation is not entirely available and thus cannot be realized in the GRASS software in a completely identical way.

Additionally, execution performance of the developed modules was evaluated in terms of the required processing times on our system (processor Intel Core2 Quad CPU 2,66GHz, disk WD2500KS, OS Linux RHEL5). The simulated configuration included eight transmission antennas on four locations (base stations), therefore requiring four model and eight sector computations. Two different geographic regions were used: a small one, "Ljutomer", encompassing all transmission locations (15 x 13km, resolution 25m) and a large one "Slovenia" (whole Slovenia, 285 x 185km, resolution 100m). The effective transmission radius was limited to 10km. The results for single core execution are given in Table 4. The hataDEM model was not simulated for the whole Slovenia region since its clutter map was not available to us. The r.MaxPower module was not run with DBF database on the whole Slovenia region since the internal GRASS DBF processing is very memory inefficient, keeping the whole database in the main memory and hence running out of memory for large regions.

7.3 Summary

Precise and efficient planning of the wireless telecommunication systems requires efficient and exact radio signal coverage calculations. The high price and limited functionalities of the existing professional network planning tools compels to look for alternative solutions. The needs can be fulfilled using an open-source system which gives possibilities to improve the existing models based on measurements, or to develop entirely new path loss prediction models. This paper presented a radio signal coverage prediction software tool developed for the open-source GRASS system. After a short introduction of the GRASS GIS system, a detailed description of the coverage prediction software was given. The tool enables a high level of flexibility and adaptability. It is composed of several GRASS modules for path loss calculation, a sectorization module, a module for radio signal coverage calculation, and additional modules for preparing input data and analyzing simulation results. Modules can be used individually or through the r.radcov module, written in Python, which interconnects individual modules into a complete radio signal propagation software package. At the end, the developed software was evaluated by comparing with the field measurements and simulation results obtained from a professional software application. The radio signal coverage prediction software implementation was quite straightforward, as API is well developed and documented. The set of built-in C functions is adequate. The possibility to study parts of the already implemented code is also very helpful. Extensive performance analyses showed satisfactory results. Compared to a professional network planning tool, the computation speed is slightly lower while the result accuracy is completely comparable irrespective of the terrain type or operational frequency. For better agreement between simulations and measurements, additional model tuning will be performed. In our future work, we also plan to expand the functionalities of the developed software package and build additional path loss modules for the urban and hilly rural environments that will also include the elements of ray tracing techniques and additional environment data [29]. The achievement made so far represents a strong base for future work and is interesting both from the point of view of researchers as well as network developers. The whole source code of the radio signal coverage prediction tool together with detailed instructions will be publicly available at http://commsys.ijs.si/en/software/grassradiocoverage tool. The tool can be freely used, modified and upgraded with new path loss modules.

8 Conclusion and further work

Comparison of our experimental results with other algorithms dealing with the same and similar problems would be useful. However, this task is not straightforward, since the results of several works (e.g. [110, 111]) depend on black-box evaluations, making experimental association very difficult, if possible at all.

All in all, we consider that the present work provides a robust foundation for future work on grid-based metaheuristics with expensive objective-function evaluation.

In future work, we will consider further analysis of our parallel-agent approach, including experimentation with different parameters, in order to gain better understanding of the dynamics leading the metaheuristic during the search process. Multi-GPU environments present an interesting possibility, where evaluator(s) and worker agents are run on separate GPU devices.

To further improve the presented results, dynamic effects, such as fast power control, should be included in the simulations, particularly for recognizing dynamic functionality like SHO in a WCDMA mobile system. Another extension of the current work is to incorporate antenna tilt as an additional objective of the optimization process. This should certainly include experimentation with models and algorithms that support multiobjective optimization.

9 Acknowledgments

The authors would like to especially thank the radio engineers at Telekom Slovenije, d.d., for cooperating and sharing their professional expertise throughout the creation of this work. This project was co-financed by the European Union, through the European Social Fund.

The contents presented in Chapter **??** are the results of joint research work, conducted with the Nagasaki Advanced Computer Center (NACC) of the Nagasaki University. In this context, T. Hamada, head of NACC, acknowledges support from the Japan Society for the Promotion of Science (JSPS) through its Funding Program for World-leading Innovative R&D on Science and Technology (First Program).

References

[1] Talbi E.G.. *Metaheuristics: from design to implementation.* Wiley 2009. 1, 1.1, 2.1.2, 2.1.3, 2.1.4, 4.4, 6.5, 9

[2] Law AM. *Simulation modeling and analysis.* Boston, MA [etc.]: McGraw-Hill 2007. 1

[3] Maria A.. Introduction to modeling and simulation in *Proceedings of the 29th conference on Winter simulation*:7–13IEEE Computer Society 1997. 1

[4] 3GPP . Functionality in early GSM releases http://www.3gpp.org accessed May 2009. 1

[5] 3GPP . General UMTS architecture, v4.0.0 http://www.3gpp.org accessed May 2009. 1

[6] Amaldi Edoardo, Capone Antonio, Malucelli Federico. Radio planning and coverage optimization of 3G cellular networks *Wireless Networks.* 2007;14:435–447. 1

[7] Siomina Iana, Yuan Di. Minimum pilot power for service coverage in WCDMA networks *Wireless Networks.* 2007;14:393–402. 1

[8] Chen L., Yuan D.. Automated planning of CPICH power for enhancing HSDPA performance at cell edges with preserved control of R99 soft-handover *Proceedings of IEEE ICC'08.* 2008. 1

[9] Chen L., Yuan D.. Fast algorithm for large-scale UMTS coverage planning with soft-handover consideration in *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*:1488–1492ACM 2009. 1

[10] Gordejuela-Sánchez F., López-Pérez D., Zhang J.. A two-step method for the optimization of antenna azimuth/tilt and frequency planning in OFDMA multihop networks in *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*:1404–1409ACM 2009. 1

[11] Siomina I., Yuan D.. Enhancing HSDPA performance via automated and large-scale optimization of radio base station antenna configuration in *Proc. 67th IEEE Vehicular Technology Conference (VTC2008-Spring)*:2061–2065 2008. 1

[12] Luke Sean. *Essentials of Metaheuristics* . Lulu 2009. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/. 2.1.1, 2.1.1, 2.1.3

[13] Dantzig George. Maximization of a linear function of variables subject to linear inequalities *New York.* 1951. 2.1.2

[14] Karmarkar Narendra. A new polynomial-time algorithm for linear programming in *Proceedings of the sixteenth annual ACM symposium on Theory of computing*:302–311ACM 1984. 2.1.2

[15] Bazaraa Mokhtar S, Sherali Hanif D, Shetty Chitharanjan Marakada. *Nonlinear programming: theory and algorithms.* Wiley-interscience 2006. 2.1.2

[16] Glover Fred. Improved linear integer programming formulations of nonlinear integer problems *Management Science.* 1975;22:455–460. 2.1.2

[17] Fu Michael C. Optimization for simulation: Theory vs. practice *INFORMS Journal on Computing.* 2002;14:192–215. 2.1.2

[18] Glover Fred. Future paths for integer programming and links to artificial intelligence *Computers & Operations Research.* 1986;13:533–549. 2.1.3

[19] Kochenberger Gary A, others . *Handbook of metaheuristics.* Springer 2003. 2.1.3

[20] Blum Christian, Roli Andrea. Metaheuristics in combinatorial optimization: Overview and conceptual comparison *ACM Computing Surveys (CSUR).* 2003;35:268–308. 2.1.3

[21] Talbi El-Ghazali. *Metaheuristics: from design to implementation*;74. Wiley 2009. 2.1.3, 2.1.4, 2.1.4

[22] Creutz Michael. Microcanonical monte carlo simulation *Physical Review Letters.* 1983;50:1411–1414. 2.1.3

[23] Kargupta H., Goldberg D. E.. Search, blackbox optimization, and sample complexity in *4th Workshop on Foundations of Genetic Algorithms (FOGA)*:291–324Morgan Kaufmann 1996. 2.1.4

[24] Barthelemy J-FM, Haftka Raphael T. Approximation concepts for optimum structural design - a review *Structural optimization.* 1993;5:129–144. 2.1.4

[25] Tantar A-A, Melab Nouredine, Talbi E-G, Parent B, Horvath D. A parallel hybrid genetic algorithm for protein structure prediction on the computational grid *Future Generation Computer Systems.* 2007;23:398–409. 2.1.4

[26] Benedičič Lucas, Štular Mitja, Korošec Peter. Pilot power optimization in UMTS: a multi-agent approach in *Proceedings of the 13th International Multiconference Information Society - IS 2010*;AJožef Stefan Institute 2010. 2.1.4, 4, 4.1

[27] Gauch Jr H.G.. *Scientific method in practice.* Cambridge University Press 2002. 2.1.4

[28] Storn R., Price K.. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces *Journal of global optimization.* 1997;11:341–359. 2.1.6, 2.1.6

[29] Das S., Suganthan P.N.. Differential evolution: A survey of the state-of-the-art *Evolutionary Computation, IEEE Transactions on.* 2010:1–28. 2.1.6

[30] Price K.V., Storn R.M., Lampinen J.A.. *Differential evolution: a practical approach to global optimization.* Springer-Verlag New York Inc. 2005. 2.1.6

[31] Dorigo M., Birattari M., Stutzle T.. Ant colony optimization *Computational Intelligence Magazine, IEEE.* 2006;1:28–39. 2.1.7

[32] Korošec Peter, Šilc Jurij, Filipič Bogdan. The differential ant-stigmergy algorithm *Information Sciences.* 2012;192:82–97. doi: 10.1016/j.ins.2010.05.002. 2.1.7, 2.1.7, 5.6.1

[33] Kirkpatrick S., Gelatt C.D., Vecchi M.P.. Optimization by simulated annealing *Science.* 1983;220:671. 2.1.8

[34] Suman B., Kumar P.. A survey of simulated annealing as a tool for single and multiobjective optimization *Journal of the operational research society.* 2005;57:1143–1160. 2.1.8

[35] Nawrocki M., Aghvami H., Dohler M.. *Understanding UMTS radio network modelling, planning and automated optimisation: theory and practice.* John Wiley & Sons 2006. 2.2, 2.3

[36] Amaldi Edoardo, Capone Antonio, Malucelli Federico. Planning UMTS base station location: Optimization models with power control and algorithms *Wireless Communications, IEEE Transactions on.* 2003;2:939–952. 2.3

[37] Amaldi Edoardo, Capone Antonio, Malucelli Federico. Radio planning and coverage optimization of 3G cellular networks *Wireless Networks.* 2008;14:435–447. 2.3

[38] Gordejuela-Sánchez Fernando, Zhang Jie. LTE access network planning and optimization: a service-oriented and technology-specific perspective in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*:1–5IEEE 2009. 2.3

[39] Han Tao, Ansari Nirwan. Optimizing Cell Size for Energy Saving in Cellular Networks with Hybrid Energy Supplies in *Global Telecommunications Conference (GLOBECOM 2012), 2012 IEEE* 2012. 2.3

[40] Lee Suk-Bok, Pefkianakis Ioannis, Meyerson Adam, Xu Shugong, Lu Songwu. Proportional fair frequency-domain packet scheduling for 3GPP LTE uplink in *INFOCOM 2009, IEEE*:2611–2615IEEE 2009. 2.3

[41] Razavi Sara Modarres, Yuan Di. Performance improvement of LTE tracking area design: a re-optimization approach in *Proceedings of the 6th ACM international symposium on Mobility management and wireless access*:77–84ACM 2008. 2.3

[42] Siomina Iana, Yuan Di. Minimum pilot power for service coverage in WCDMA networks *Wireless Networks.* 2007;14:393–402. 2.3, 4.8

[43] Hao Q., Soong B.H., Gunawan E., Ong J.T., Soh C.B., Li Z.. A low-cost cellular mobile communication system: a hierarchicaloptimization network resource planning approach *IEEE Journal on Selected Areas in Communications.* 1997;15:1315–1326. 2.3.1.1, 2.3.1.2

[44] Tutschku K.. Demand-based radio network planning of cellular mobilecommunication systems in *IEEE INFOCOM'98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*;3. 2.3.1.1, 2.3.1.2

[45] Mathar R., Niessen T.. Optimum positioning of base stations for cellular radio networks *Wireless Networks.* 2000;6:421–428. 2.3.1.1, 2.3.1.2

[46] Aydin Mehmet, Yang Jun, Zhang Jie. *Applications of Evolutinary Computing*;4448 of *Lecture Notes in Computer Science.* Berlin, Heidelberg: Springer Berlin Heidelberg 2007. 2.3.1.2

[47] Raisanen L., Whitaker R.. Comparison and evaluation of multiple-objective genetic algorithms for the antenna placement problem *Mobile Networks and Applications.* 2005;10:79-88. 2.3.1.2

[48] Holma H., Toskala A.. *WCDMA for UMTS: Radio access for third generation mobile communications, Third Edition.* John Wiley & Sons, Inc. New York, NY, USA 2005. 2.3.2, 2.3.3, 2.4.2, 5.2, 5.3.2, 5.5

[49] Karner W.. *Optimum default base station parameter settings for UMTS networks.* No. SeptemberCiteseer 2003. 2.3.2.1

[50] Gerdenitsch A., Jakl S., Toeltsch M., Neubauer T.. Intelligent algorithms for system capacity optimization of UMTS FDD networks *IEE Conference Publications.* 2003;2003:222-226. 2.3.2.1, 2.3.4.1

[51] Jakl S.. *Evolutionary Algorithms for UMTS Network Optimization.* PhD thesis, Technische Universitat Wien 2004. 2.3.2.1

[52] Siomina I., Yuan D.. Enhancing HSDPA performance via automated and large-scale optimization of radio base station antenna configuration in *Proc. 67th IEEE Vehicular Technology Conference (VTC2008-Spring)*:2061–2065 2008. 2.3.2.1

[53] Wikipedia . High-Speed Downlink Packet Access — Wikipedia, The Free Encyclopedia 2010. `http://en.wikipedia.org/wiki/Hsdpa`, [Online; accessed 19-April-2010]. 2.3.2.1

[54] Gordejuela-Sánchez Fernando, López-Pérez David, Zhang Jie. A two-step method for the optimization of antenna azimuth/tilt and frequency planning in OFDMA multihop networks in *IWCMC '09: Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing*(New York, NY, USA):1404–1409ACM 2009. 2.3.2.1

[55] UMTSWorld.com . Handover — UMTSWorld.com 2010. http://www.umtsworld.com/technology/handover.htm, [Online; accessed 12-April-2010]. 2.3.3

[56] Lte Info . UMTS cell selection procedure — 3g Lte Info 2010. http://www.3glteinfo.com/umts-cell-selection-procedure, [Online; accessed 12-April-2010]. 2.3.3

[57] Flore D., Brunner C., Grilli F., Vanghi V.. Cell Reselection Parameter Optimization in UMTS in *Wireless Communication Systems, 2005. 2nd International Symposium on*:50–53 2005. 2.3.3

[58] Wikipedia . Soft handover — Wikipedia, The Free Encyclopedia 2010. http://en.wikipedia.org/wiki/Soft_handover, [Online; accessed 19-April-2010]. 2.3.3.1

[59] Siomina Iana, Yuan Di. Automated planning of CPICH power allocation for enhancing HSDPA performance at cell edges 2007:112–118. 2.3.3.1

[60] Garcia-Lozano M., Ruiz S., Olmos J.. CPICH power optimisation by means of simulated annealing in an UTRA-FDD environment *Electronics Letters.* 2003;39:1676. 2.3.3.1

[61] Siomina I., Varbrand P., Yuan Di. Automated optimization of service coverage and base station antenna configuration in UMTS networks *IEEE Wireless Communications.* 2006;13:16–25. 2.3.4.1

[62] Siomina Iana. P-CPICH Power and Antenna Tilt Optimization in UMTS Networks 2005:268–273. 2.3.4.1

[63] Gerdenitsch A., S. Jakl, Toeltsch M.. The use of genetic algorithms for capacity optimization in UMTS FDD networks *Proc. 3rd International Conference on Networking (ICN'04).* 2004. 2.3.4.1

[64] 3GPP . Requirements for support of radio resource management (FDD), v4.17.0 http://www.3gpp.org [Online; accessed May-2009]. 2.3.5

[65] Siomina Iana, Yuan Di. Minimum pilot power for service coverage in WCDMA networks *Wireless Networks.* 2007;14:393–402. 2.3.5.1, 3, 4.2, 4.3.2, 4.4, 4.6.1, 4.7, 4.8

[66] Valkealahti K., Hoglund A., Parkkinen J., Hamalainen A.. WCDMA common pilot power control for load and coverage balancing in *Proceedings of PIMRC*;2Citeseer. 2.3.5.1

[67] Valkealahti K., Hoglund A., Parkkinen J., Flanagan A.. WCDMA Common Pilot Power Control with Cost Function Minimization *statistics.* 2002;1000:1. 2.3.5.1

[68] IST-MOMENTUM . Models and Simulations for Network planning and Control of UMTS http://momentum.zib.de, [Online; accessed 15-April-2010]. 2.3.7, 4.6.1

[69] Third Generation Partnership Project . *Technical specification group services and systems aspects: Network architecture v3.6.0 (Release 1999).* http://www.3gpp.org/. 2.4.2

[70] Holma H., Toskala A.. *HSDPA/HSUPA For UMTS.* John Wiley & Sons 2006. 2.4.2, 5.1, 5.3.2, 5.5.2

[71] Laiho J., Wacker A., Novosad T.. *Radio Network Planning and Optimisation for UMTS.* John Wiley & Sons, Inc. New York, NY, USA 2002. 2.4.3, 5.2

[72] Holma H., Toskala A.. *WCDMA for UMTS: Radio access for third generation mobile communications, Third Edition.* John Wiley & Sons, Inc. New York, NY, USA 2005. 2.4.3, 4.1, 4.3.2, 4.5.1, 4.6.1, 4.7, 4.8

[73] Nawrocki M., Aghvami H., Dohler M.. *Understanding UMTS radio network modelling, planning and automated optimisation: theory and practice.* John Wiley & Sons 2006. 2.4.3, 4.1, 4.2, 4.3, 5.3

[74] Stone J.E., Gohara D., Shi G.. OpenCL: A parallel programming standard for heterogeneous computing systems *Computing in Science and Engineering.* 2010;12:66–73. 2.5.1

[75] Munshi A., others . The OpenCL specification version 1.0 *Khronos OpenCL Working Group.* 2009. 2.5.1

[76] Nvidia C.. Compute Unified Device Architecture Programming Guide *NVIDIA: Santa Clara, CA.* 2007;83:129. 2.5.1

[77] Kloeckner Andreas. GPU from Python with PyOpenCL and PyCUDA `http://www.bu.edu/pasi/materials/`, [Online; accessed 5-April-2010]. 2.3, 9

[78] Maple C., Guo L., Zhang J.. Parallel genetic algorithms for third generation mobile network planning in *Parallel Computing in Electrical Engineering, 2004. PARELEC 2004. International Conference on*:229–236IEEE 2004. 3

[79] Crainic T.G., Di Chiara B., Nonato M., Tarricone L.. Tackling electrosmog in completely configured 3G networks by parallel cooperative meta-heuristics *Wireless Communications, IEEE.* 2006;13:34–41. 3

[80] Soldani D., Alford G., Parodi F., Kylvaja M.. An autonomic framework for self-optimizing next generation mobile networks in *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*:1–6IEEE 2007. 3

[81] Gorder P.F.. Multicore processors for science and engineering *Computing in science & engineering.* 2007;9:3–7. 3

[82] Wen-mei W.H.. *GPU Computing Gems Emerald Edition: Applications of GPU Computing Series.* Morgan Kaufmann 2011. 3

[83] Valcarce A., De La Roche G., Jüttner Á., López-Pérez D., Zhang J.. Applying FDTD to the coverage prediction of WiMAX femtocells *EURASIP Journal on Wireless Communications and Networking.* 2009;2009:1–13. 3, 4.1

[84] Hrovat A., Ozimek I., Vilhar A., Celcer T., Saje I., Javornik T.. An open-source radio coverage prediction tool in *Proceedings of the 14th WSEAS international conference on Communications*:135–140World Scientific and Engineering Academy and Society (WSEAS) 2010. 3.1, 3.1.1, 3.2.1, 3.2.3, 3.4, 4.5.1, 6.3, 6.6

[85] Benedičič Lucas, Štular Mitja, Korošec Peter. A GPU-based parallel-agent optimization approach for the service-coverage problem in UMTS networks *Computing and Informatics.* 2013:In press. 4

[86] Esposito A., Tarricone L., Luceri S., Zappatore M.. Genetic Optimization for Optimum 3G Network Planning: an Agent-Based Parallel Implementation *Novel Algorithms and Techniques in Telecommunications and Networking.* 2010:189–194. 4.1

[87] Cunningham B.M., Alexander P.J., Candeub A.. Network growth: Theory and evidence from the mobile telephone industry *Information Economics and Policy.* 2010;22:91–102. 4.1

[88] Cheung G., Tan W., Yoshimura T.. Real-time video transport optimization using streaming agent over 3G wireless networks *IEEE transactions on multimedia.* 2005;7:777. 4.1

[89] Vasile M., Locatelli M.. A hybrid multiagent approach for global trajectory optimization *Journal of Global Optimization.* 2009;44:461–479. 4.1

[90] Chen L., Yuan D.. Automated planning of CPICH power for enhancing HSDPA performance at cell edges with preserved control of R99 soft handover *Proceedings of IEEE ICC'08.* 2008. 4.1, 4.3.2

[91] Siomina I., Yuan D.. Pilot power optimization in WCDMA networks 2004;4. 4.2, 4.4

[92] Lieska Kai, Laitinen Erkki, Lahteenmaki Jaakko. Radio coverage optimization with genetic algorithms in *Personal, Indoor and Mobile Radio Communications, 1998. The Ninth IEEE International Symposium on*;1:318–322IEEE 1998. 4.2

[93] Thampi Ajay, Kaleshi Dritan, Randall Peter, Featherstone Walter, Armour Simon. A sparse sampling algorithm for self-optimisation of coverage in LTE networks in *Wireless Communication Systems (ISWCS), 2012 International Symposium on*:909–913IEEE 2012. 4.2

[94] Nawrocki M., Aghvami H., Dohler M.. *Understanding UMTS radio network modelling, planning and automated optimisation: theory and practice*. John Wiley & Sons 2006. 4.3, 5.1, 5.3, 5.3.3, 5.3.3, 5.3.3

[95] Värbrand P., Yuan D.. A mathematical programming approach for pilot power optimization in WCDMA networks 2003. 4.3.3

[96] Sarkar P.. A brief history of cellular automata *ACM Computing Surveys (CSUR).* 2000;32:107. 4.4

[97] Marsaglia G.. Seeds for random number generators *Communications of the ACM.* 2003;46:90–93. 4.5.2

[98] Benedičič Lucas, Štular Mitja, Korošec Peter. Balancing downlink and uplink soft-handover areas in UMTS networks in *Evolutionary Computation (CEC), 2012 IEEE Congress on*:1–8IEEE 2012. 5, 6.5

[99] Chen L., Sandrasegaran K., Basukala R., Madani F.M., Lin C.C.. Impact of soft handover and pilot pollution on video telephony in a commercial network in *Communications (APCC), 2010 16th Asia-Pacific Conference on*:481–486IEEE 2010. 5.2

[100] Chen L., Yuan D.. Coverage planning for optimizing HSDPA performance and controlling R99 soft handover *Telecommunication Systems.* 2011:1–12. 5.2

[101] Eisenblätter A., Füngenshuch A., Geerdes H. F., Junglas D., Koch T., Martin A.. Optimization methods for UMTS radio network planning in *Intl. Conference on Operations Research*:31–38Springer 2003. 5.2

[102] Ghosh S.C., Whitaker R.M., Allen S.M., Hurley S.. Optimising CDMA Cell Planning with Soft Handover *Wireless Personal Communications.* 2011:1–27. 5.2

[103] Chen L., Yuan D.. CPICH Power Planning for Optimizing HSDPA and R99 SHO Performance: Mathematical Modelling and Solution Approach in *Proc. IFIP 1st Wireless Days Conference (WD)*:1–5 2008. 5.2

[104] Siomina I.. Pilot power management in WCDMA networks: coverage control with respect to traffic distribution in *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*:276–282ACM New York, NY, USA 2004. 5.2

[105] Ying S., Gunnarsson F., Hiltunen K., Res E., Beijing C.. CPICH power settings in irregular WCDMA macro cellular networks *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003.* 2003;2. 5.2

[106] Garcia-Lozano M., Ruiz S., Olmos J.. CPICH power optimisation by means of simulated annealing in an UTRA-FDD environment *Electronics Letters.* 2003;39:1676–7. 5.2

[107] Lempiäinen J., Manninen M.. *UMTS Radio Network Planning, Optimization and QoS Management for Practical Engineering Tasks.* Kluwer Academic Publishers Norwell, MA, USA 2004. 5.2

[108] Siomina I., Yuan D.. Minimum pilot power for service coverage in WCDMA networks *Wireless Networks.* 2008;14:393–402. 5.2

[109] Cichon D.J., Kurner T.. Propagation prediction models *COST 231 Final Rep.* 1995. 6.3

[110] Gerdenitsch A.. *System capacity optimization of UMTS FDD networks.* Ph.D. dissertation, Technische Universität Wien 2004. 8

[111] Türke U., Koonert M.. Advanced site configuration techniques for automatic UMTS radio network design *Proc IEEE VTC 2005 Spring.* 2005. 8

List of Figures

List of Tables