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

Lucas Benedičič

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

Evaluation Board:

Name and surname of board member with the title, Chairman, institution and address

Name and surname of board member with the title, Member, institution and address

Name and surname of board member with the title, Member, institution and address

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, May 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.
GSM	Global System for Mobile communications.
UMTS	Universal Mobitel Telecommunications System.

S

$A_{covered}$	Area, covered by the mobile network.
A_{total}	Total area under optimization of mobile-network coverage.
$cov(x, y)$	Returns 1 if the coordinate (x, y) is under mobile-network coverage.
f_{cov}	Objective function for the coverage optimization problem.

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.

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 [1].

A model is a simple 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 [2].

Problems that are 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.

Mobile radio networks represent one of the most fast-growing technology markets since the introduction of the Global System for Mobile communications (GSM) [3] as the second generation (2G) mobile networks more than twenty years ago. Its successor, the Universal Mobile Telecommunications System (UMTS) [4] 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 [5–10], 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 Motivation

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 4 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 6, 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 5 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

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. Experimentation and expert observation will always be essential as reference points for understanding the studied phenomena.

2.1 Radio-network optimization

Once a mobile network is launched, an important part of its operation and maintenance is monitoring the quality characteristics and changing parameter values in order to improve its performance.

The evolution from 2G to 3G has introduced not only the technology needed to increase both data and voice capacity, but also a greater complexity in terms of network planning, deployment, and configuration, which have rendered most of the traditionally used methods to be ineffective. In a traditional approach (i.e. manual), during the network planning and maintenance processes, a network planning software tool would execute the analysis, while the human would make the change decisions. Therefore, a radio planning engineer configures 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 the goal is achieved.

Modern 3G radio networks are large and many of their key parameters are interdependent. Since an engineer is not able to cope with the level of complexity present in such systems, 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 optimization.

A common limitation of the implementations of such optimization methods, generally targeting traditional computer architectures of sequential execution, is their inability to meet the requirements needed by real-world mobile networks, since their computational-time complexity make them unfeasible for practical use. When analyzing big real-world networks with thousands of users it is necessary to reduce the execution time of the optimization processes as much as possible, so that they are useful for practical use.

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 [11, 12].

The complexity of these systems has grown even faster than their throughput capacity, thus making it impossible to plan 3G mobile networks with traditional methods. In this sense, an examination of colored coverage maps in conjunction with some

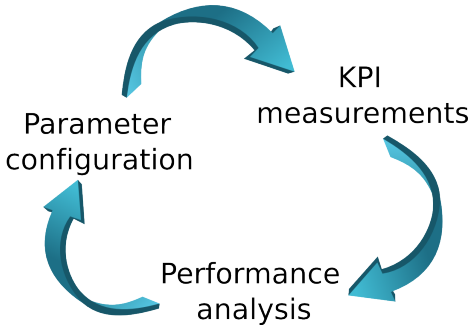

Figure 1: Optimization cycle for 3G mobile networks.

statistical analysis are no longer appropriate tools for network examination. Once a mobile network is launched, an important part of its operation and maintenance is monitoring of quality characteristics and changing parameter values in order to improve performance. In this sense, we may divide these tasks in two phases: analysis and decision [13]. The analysis phase consists of network performance analysis, which mainly focuses on definition and collection of Key Performance Indicators (KPIs). KPIs are quantifiable measurements, agreed to beforehand, that reflect, in this context, network quality factors. The second phase deals with making decisions, based on analytical results collected in the previous phase, about the configuration of particular parameter settings. This process (shown in Figure 1) is repeated until the achieved results are acceptable.

In a traditional method approach (i.e. manual), during network planning and optimization processes, a network planning software tool would execute the analysis, while the human would make the decisions. Consequently, a radio planning engineer configures 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 the goal is achieved.

Modern 3G mobile networks are large and many of their key parameters are interdependent. Since an engineer is not able to cope with the level of complexity present in such systems, 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 *optimization*. The results of the optimization process are expected to provide advice[1] regarding deployment, extension or reconfiguration tasks of a 3G network.

2.2 Optimization of 3G mobile networks

Although wireless networks are increasingly more sophisticated, the need for optimization work is still far from declining. It has been established that most 3G radio network optimization problems are NP-hard, since computational time grows non-polynomially with the increase of the problem size [14]. 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 [13]:

- Network performance improvement: more users covered with the same physical

[1]Some works are already focusing on complete automation of these tasks (i.e. without human intervention).

infrastructure, leaving room for parameter optimization only.

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

- Changing propagation conditions: the allocation of a higher frequency band for UMTS compared to GSM requires deployment of more 3G sites than in 2G networks, mostly in urban areas, thus increasing inter-cell interference.

In this sense, network operators have shown the need to define different optimization targets. These targets are formed by an objective function that maps possible values of configuration parameters into a value. Thus, each element (solution) of the solution space is assigned a comparable value, i.e. a real number. 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 3G network optimization [13]. However, it is possible to optimize for different targets such as coverage, base station location, 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 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 Optimizing base station locations

2.3.1 Problem formulation

Some references [15–17] formulate the base station location problem in terms of the minimum set covering problem (shown in Figure 2). 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: 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.

2.3.2 Proposed solutions

Aydin et al. [18] 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. [15]. A simulated annealing implementation was developed to solve the formulated combinatorial problem. The results presented demonstrate the feasibility of the proposed approach. Tutschku [16] 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 [17] 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. [14] 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 [19] 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 [14] are a subset of the constraints in [19].

Finally, Whitaker et al. [20] 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.

2.4 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 3) is the direction in which the main beam of the horizontal pattern points [21]. The antenna tilt (shown in Figure 4) is defined as the angle of the main beam of the antenna relative to the horizontal plane [21]. 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.

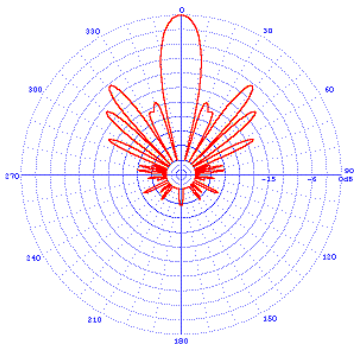

Figure 3: A typical antenna azimuth pattern.

2.4.1 Proposed solutions

Karner [22] 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 improve the results presented in [23] by increasing the number of served users in the target area. In a similar line of work, Jakl [24] 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 [25] 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) [26] throughput.

Zhang et al. [27] 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

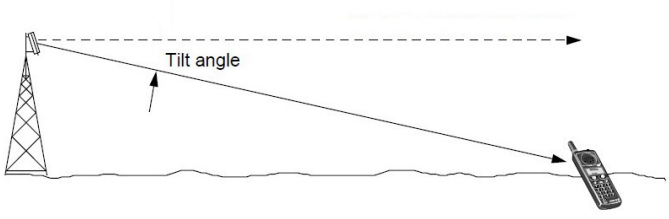

Figure 4: The antenna tilt angle with the horizontal plane.

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.

2.5 Optimizing common pilot channel power

The Common Pilot Channel (CPICH) is used as reference for handover [28], cell selection [29], and cell reselection [30]. 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 [21].

2.5.1 Proposed solutions

Chen and Yuan [31] 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 [32]. The solution approach is based on a linear-integer mathematical model. A very similar problem and proposed solution is presented by Siomina and Yuan [33]. The problem definition slightly differs from [31] as there is no reference to soft handover control. The solution is also implemented as a linear program.

Olmos et al. [34] 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 [35] 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.

2.6 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.

2.6.1 Proposed solutions

Varbrand et al. [36] approach the optimization of service coverage (see section 2.7) 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 [37] 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. [23] 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. [38] 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.

2.7 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:

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

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, which may vary from 10 to 100 meters, creating a grid of a certain resolution. A square is considered covered, if the signal to noise ratio is above a given threshold [39]. It is also common to use a test function $cov(x, y)$, which returns 1 if the square located at (x, y) is covered, and 0 otherwise.

2.7.1 Proposed solutions

Siomina and Yuan [40] 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. [19] 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. [41] 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. [42] 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.

2.8 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 [43]. 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 [43].

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 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.

2.8.1 The reproducibility problem

A quick review of the state-of-the-art in 3G network optimization indicates that software, providing good computational models, is a very expensive tool for science. Moreover, since the vast majority of this software is proprietary, it relies on closed source, formats and protocols, which disclosure is explicitly forbidden by their licenses. This fact creates a big hurdle to one of the key phases of scientific methodology [44]: experimental reproducibility.

2.8.2 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[2] 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 1. The comparison variables arise from the context in which the methods were presented.

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

Table 1: 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 improvement
Linear programming	(formulation dependent)	Coverage network planning.

2.9 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.

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 [45–47]. 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 [11] and GPUs [12]. 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.

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 [40, 48–50]. 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 [49]. 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 [48], 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.

3.0.1 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 [51]. One such system is the DEGIMA cluster [52] 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.

3.0.2 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.

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

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

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 3.4 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. [53]. 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 [53] 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 [49, 54], 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 (1) describes the path loss when there is LOS between the transmitter and the receiver.

$$PL_{\mathrm{LOS}}(d) = 42.64 + 26\log(d) + 20\log(F), \tag{1}$$

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

$$PL_{\mathrm{NLOS}}(d) = L_0 + L_{\mathrm{RTS}} + L_{\mathrm{MSD}}, \tag{2}$$

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 [53], 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. [55]. Consequently, both Equations (1) and (2) have an extra term for signal loss due to clutter (L_{CLUT}), thus redefining the LOS and NLOS path losses as

$$PL_{\mathrm{LOS}}(d) = 42.64 + 26\log(d) + 20\log(F) + L_{\mathrm{CLUT}} \tag{3}$$

and

$$PL_{\mathrm{NLOS}}(d) = L_0 + L_{\mathrm{RTS}} + L_{\mathrm{MSD}} + L_{\mathrm{CLUT}}. \tag{4}$$

3.1.2 GRASS Geographical Information System

As the software environment for PRATO we have chosen GRASS (Geographic Resources Analysis Support System) [56], 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 [57] 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 de-

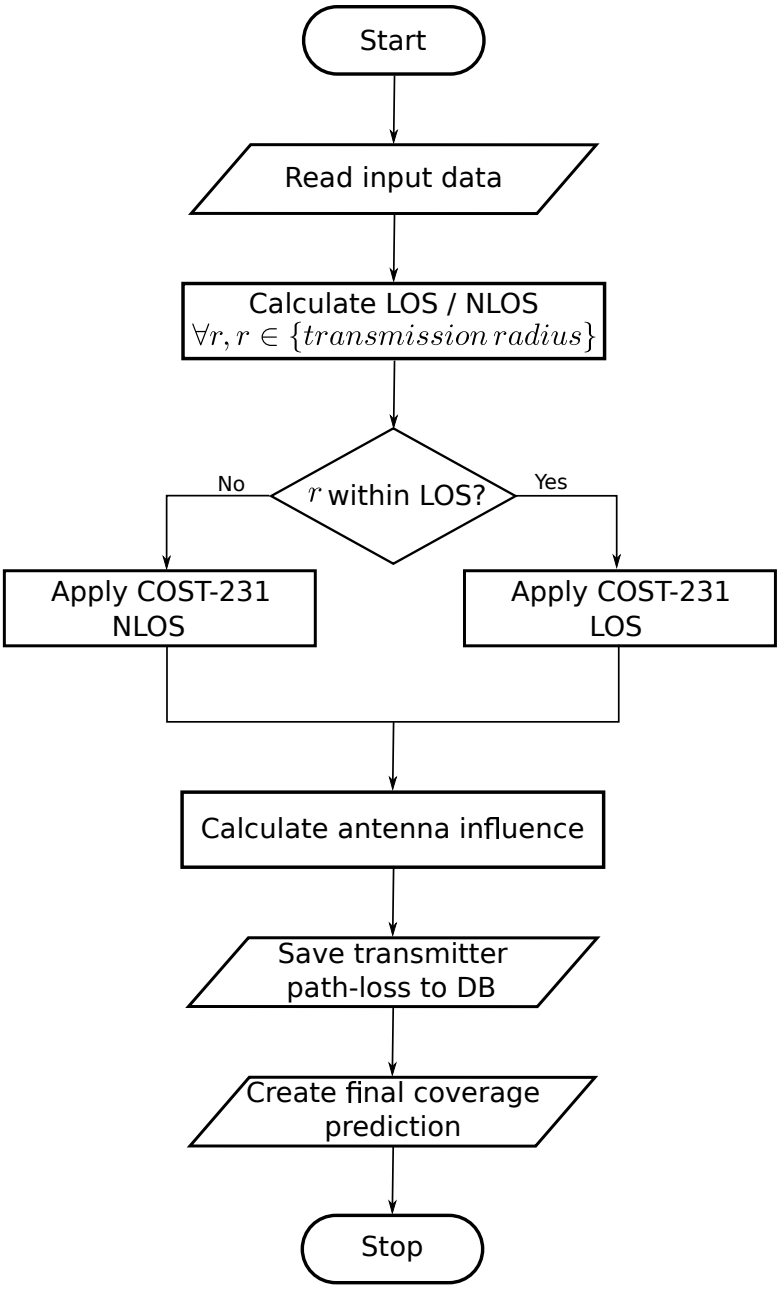

Figure 5: Flow diagram of the serial version.

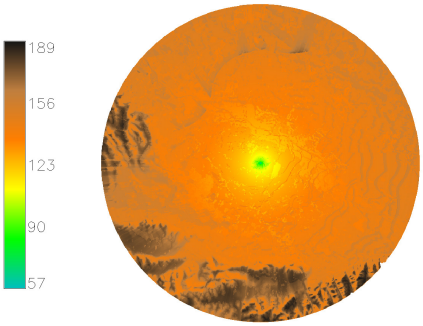

Figure 6: Example of raster map, showing the result of a path-loss calculation from an isotropic source.

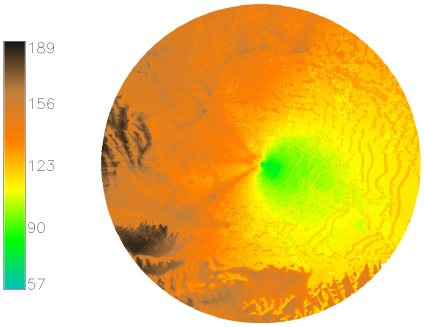

Figure 7: Example of raster map, showing the antenna influence over the isotropic path-loss result.

picted in Figure 5 on page 21, 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. [53], 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 5 on page 21), 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 5 on page 21). For these points, the LOS and NLOS conditions are calculated, with respect to the transmitter (see "Calculate LOS/NLOS" in Figure 5 on page 21). 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 transmitter. Depending on whether the receiver point r is in LOS or NLOS, either Equation (3) or Equation (4) is respectively applied (see "Apply COST-231, LOS" or "Apply COST-231, NLOS" in Figure 5 on page 21).

Figure 6 on page 21 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 5 on page 21). 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 7 on page 21 shows a portion of a raster map, where this calculation step has been applied to the results from Figure 6 on page 21. 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 5 on page 21), 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 5 on page 21). 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 8 on page 24. 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 implement parallelism [58]. 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) [59]. MPI is a message-passing standard which defines syntax and semantics designed to function

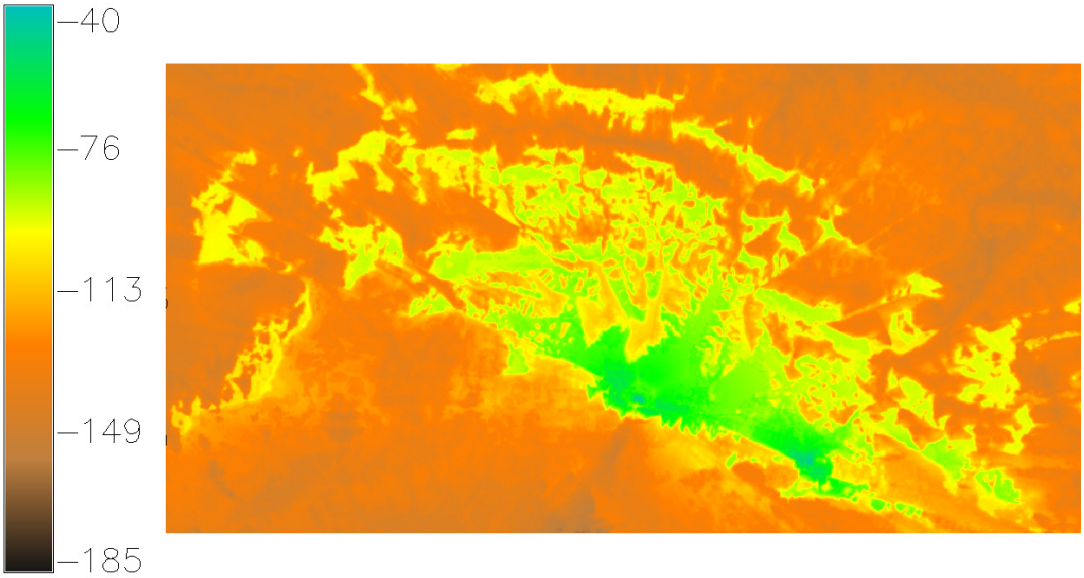

Figure 8: Example of raster map, displaying the final coverage prediction of several transmitters. The color scale is given in dBm, indicating the received signal strength.

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 [53].

Some authors have already published their work on implementing parallel versions of GRASS modules for solving different time-consuming tasks [60–62]. 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 asynchronously 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 9 on page 26, 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 9 on page 26, 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 9 on page 26), 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 9 on page 26). 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 9 on page 26), which main task is to assign and distribute the configuration data of different transmitters among idle worker processes.

The flow diagram shown in Figure 10 on page 27 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 10 on page 27). 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 10 on page 27). 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 10 on page 27). 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 10 on page 27). 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 10 on page 27). The master process will keep doing this until all worker processes have finished (see "Any worker still on?" in Figure 10 on page 27), 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 9 on page 26). The final coverage prediction of all transmitters is an aggregation from

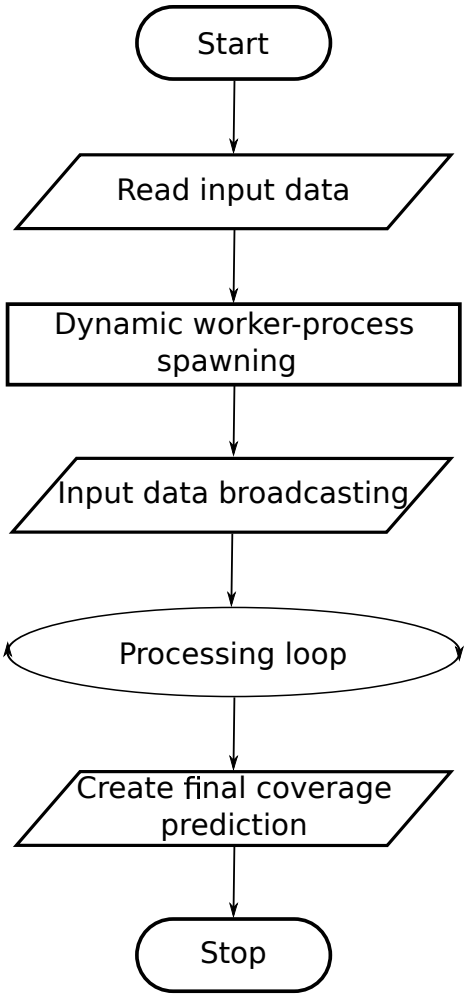

Figure 9: Flow diagram of the master process.

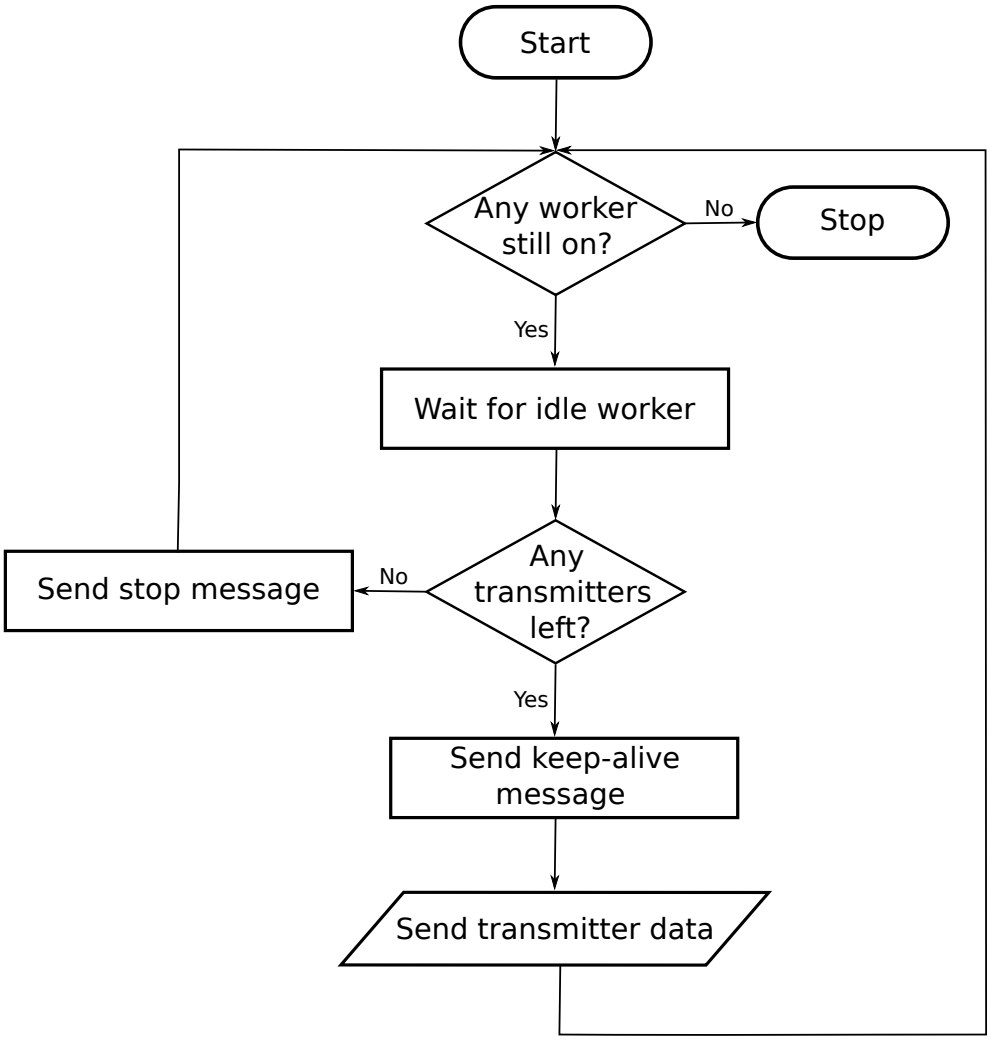

Figure 10: Flow diagram of the "Processing loop" step of the master process.

the individual path-loss results created by each of the worker processes during the "Processing loop" phase in Figure 9 on page 26, 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 11 on page 29, are initialized by data that are received from the master process at initialization time (see "Receive broadcasted data" in Figure 11 on page 29). 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 [63]. 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 11 on page 29). If the master process does not instruct to stop processing (see "Has stop message arrived?" in Figure 11 on page 29), the worker process collects the transmitter configuration sent (see "Receive transmitter data" in Figure 11 on page 29). However, in case a

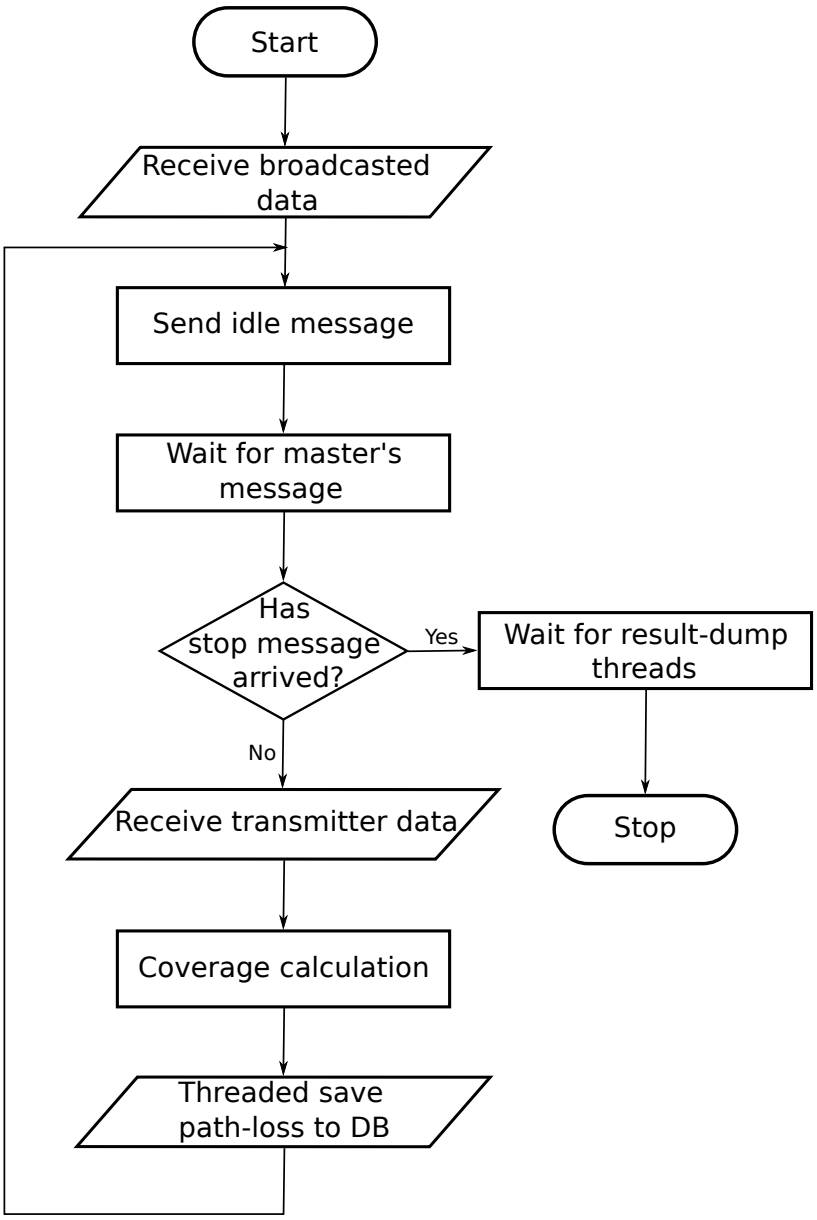

Figure 11: Flow diagram of a worker process.

stop message is received, the worker process will wait for result-dumping threads to finish (see "Wait for result-dump threads" in Figure 11 on page 29) before shutting down. The coverage calculation itself follows a similar design as the serial version (see "Coverage calculation" in Figure 11 on page 29) and it is executed for the received 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 11 on page 29). 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 [64].

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 [65].

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 11 on page 29 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 [66], which may lead to situations such as race conditions [67] 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 [66].

Figure 12 on page 31 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:

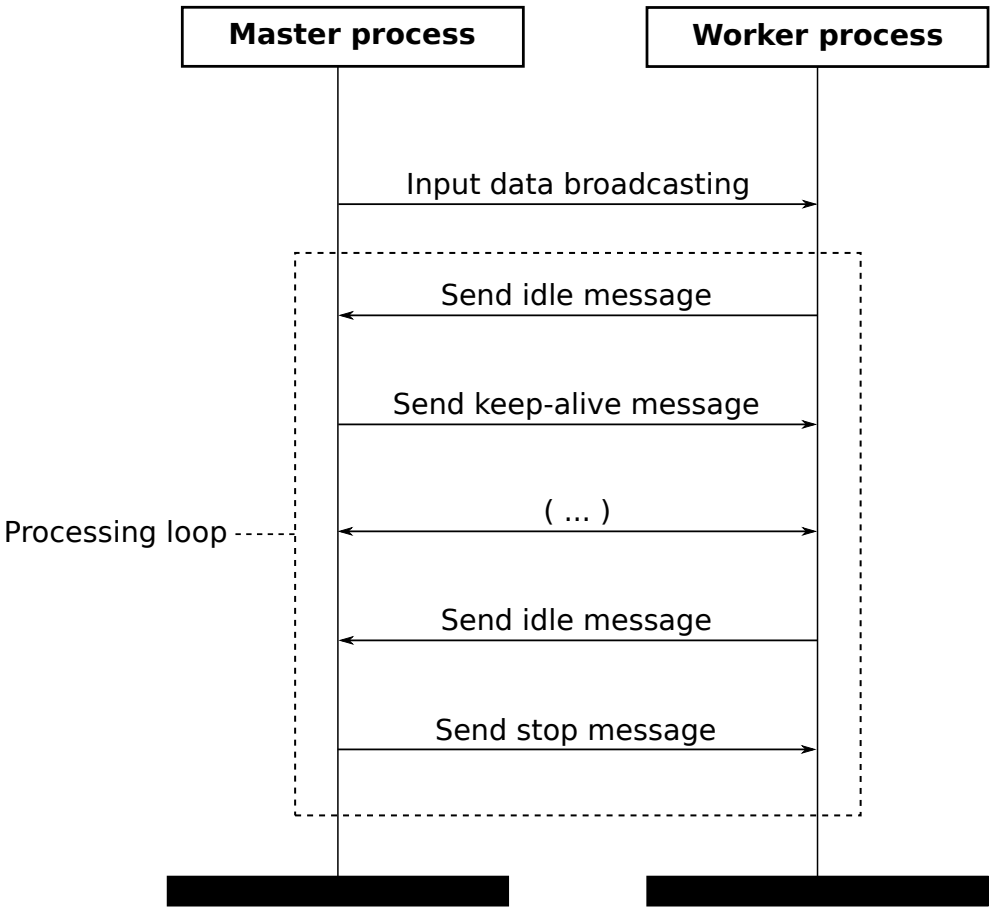

Figure 12: Communication diagram, showing message passing between master and one worker process.

- 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 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 2: 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 2. 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 13 on page 34, 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 [68].

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 14 on page 34. The relative-processing-time plots follow the formula

$$RT = \frac{t_{\mathrm{rd}} + t_{\mathrm{ps}} + t_{\mathrm{db}} + t_{\mathrm{pl}} + t_{\mathrm{cp}}}{t_{\mathrm{total}}}, \tag{5}$$

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 master process, see Figure 9 on page 26.

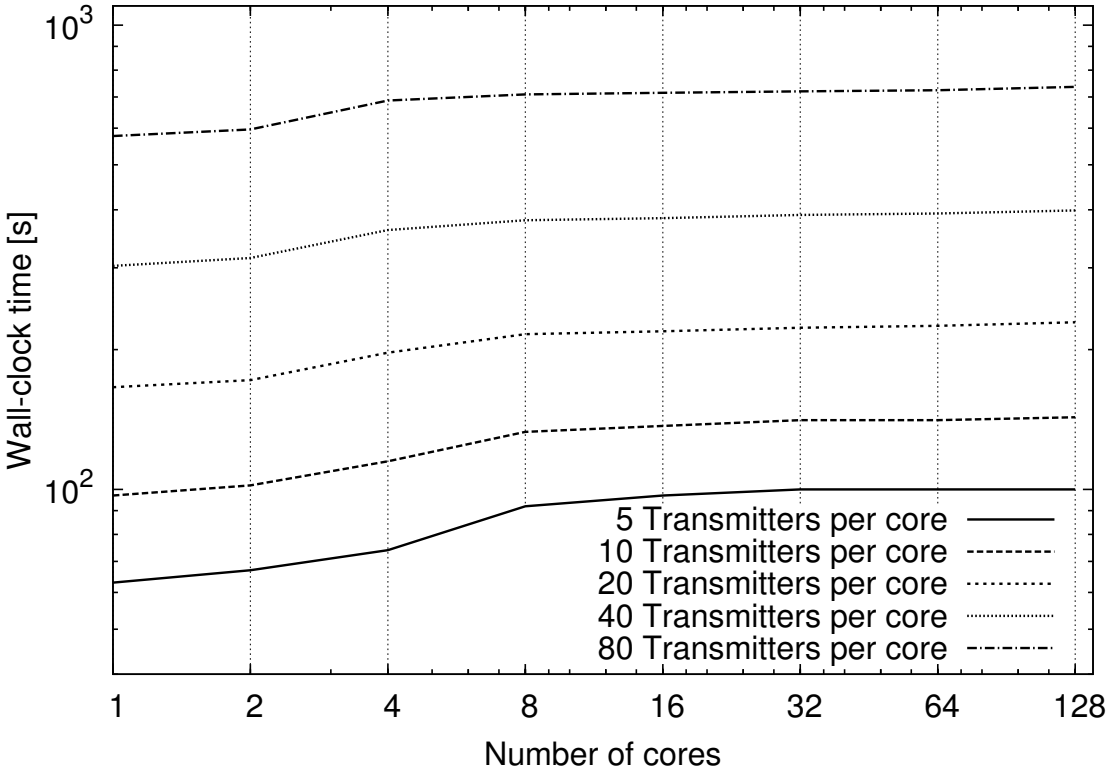

Figure 13: Measured wall-clock time for weak-scalability experiments as shown in Table 2. Experiments performed assigned one MPI worker process per available core. The wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

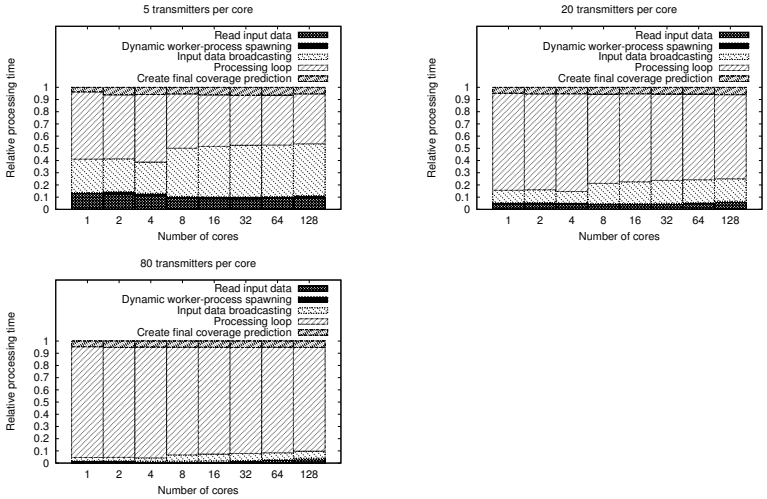

Figure 14: Relative times for the weak-scalability experiments. The relative-processing time axes are expressed in linear scale, whereas the axes representing the number of cores are expressed in base-2 logarithmic scale.

Table 3: 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

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. All times are expressed in seconds. These wall-clock time measurements are plotted in Figure 15 on page 36, 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

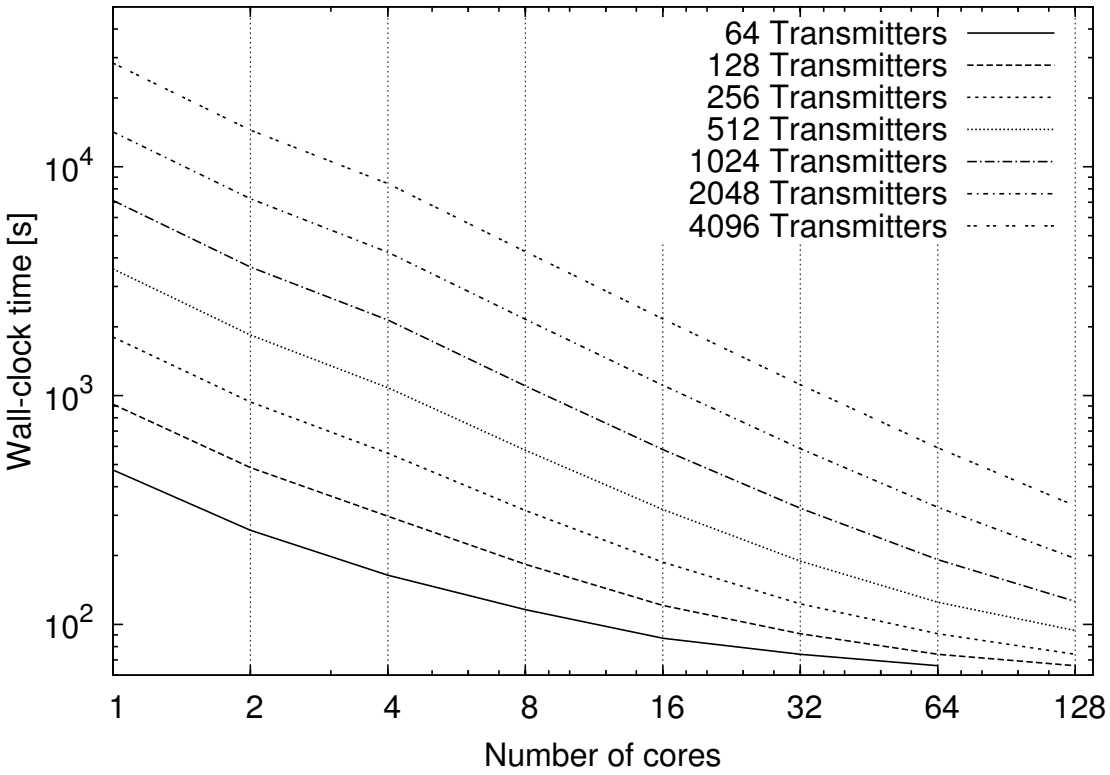

Figure 15: Measured wall-clock time for strong-scalability experiments as shown in Table 3. Experiments performed assigned one MPI worker process per available core. The wall-clock time axis is expressed in base-10 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

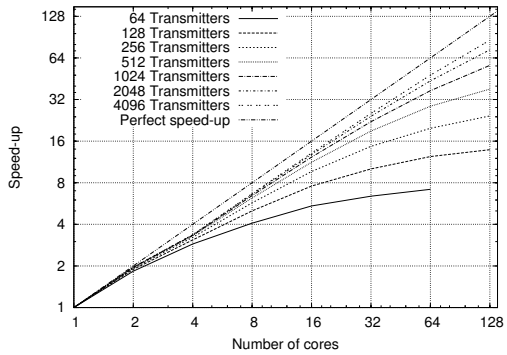

Figure 16: Measured speedup for strong-scalability experiments. The speedup axis is expressed in base-2 logarithmic scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

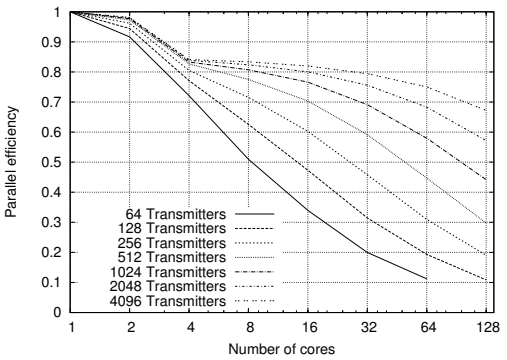

Figure 17: Measured parallel efficiency for strong-scalability experiments. The parallel-efficiency axis is expressed in linear scale, whereas the axis representing the number of cores is expressed in base-2 logarithmic scale.

observed in Figure 15 on page 36 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

$$S(NP) = \frac{execution\,time\,for\,base\,case}{execution\,time\,for\,NP\,cores}, \tag{6}$$

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 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 [69].

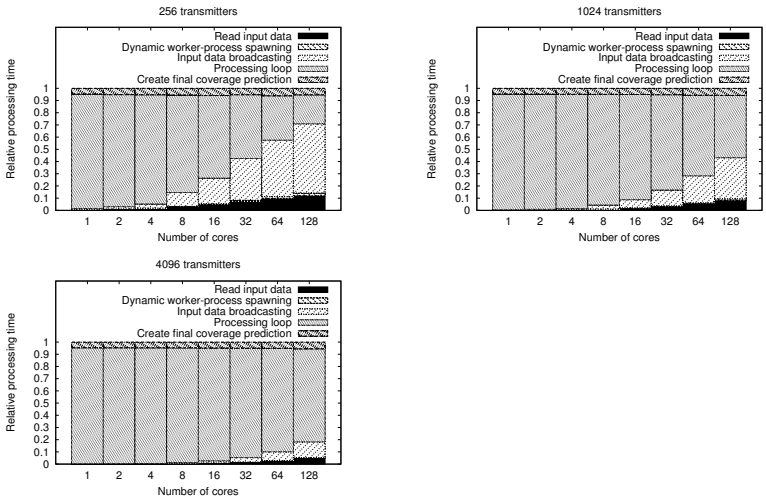

Figure 18: Relative times for the strong-scalability experiments. The relative-processing time axes are expressed in linear scale, whereas the axes representing the number of cores are expressed in base-2 logarithmic scale.

Figure 16 on page 37 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 [68]. 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 15 on page 36 are directly correlated with the speedup factors shown in Figure 16 on page 37.

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:

$$E(NP) = \frac{S(NP)}{NP}, \tag{7}$$

where $S(NP)$ is the speedup as defined in Equation (6), and NP is the number of cores executing worker processes. Figure 17 on page 37 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 17 on page 37, 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 18 on page 38. The relative times shown are calculated using the formula depicted in Equation (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 18 on page 38), 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 [70].

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 12 on page 31), 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 [71]. 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 [72] by using the following metric as an indicator of the load balancing among processes:

$$LB(NP) = \frac{minimum\,execution\,time\,among\,NP\,cores}{processing\,loop\,time\,of\,master\,process}, \tag{8}$$

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 9 on page 26), until the beginning of the last step ("Create final coverage prediction" in Figure 9 on page 26).

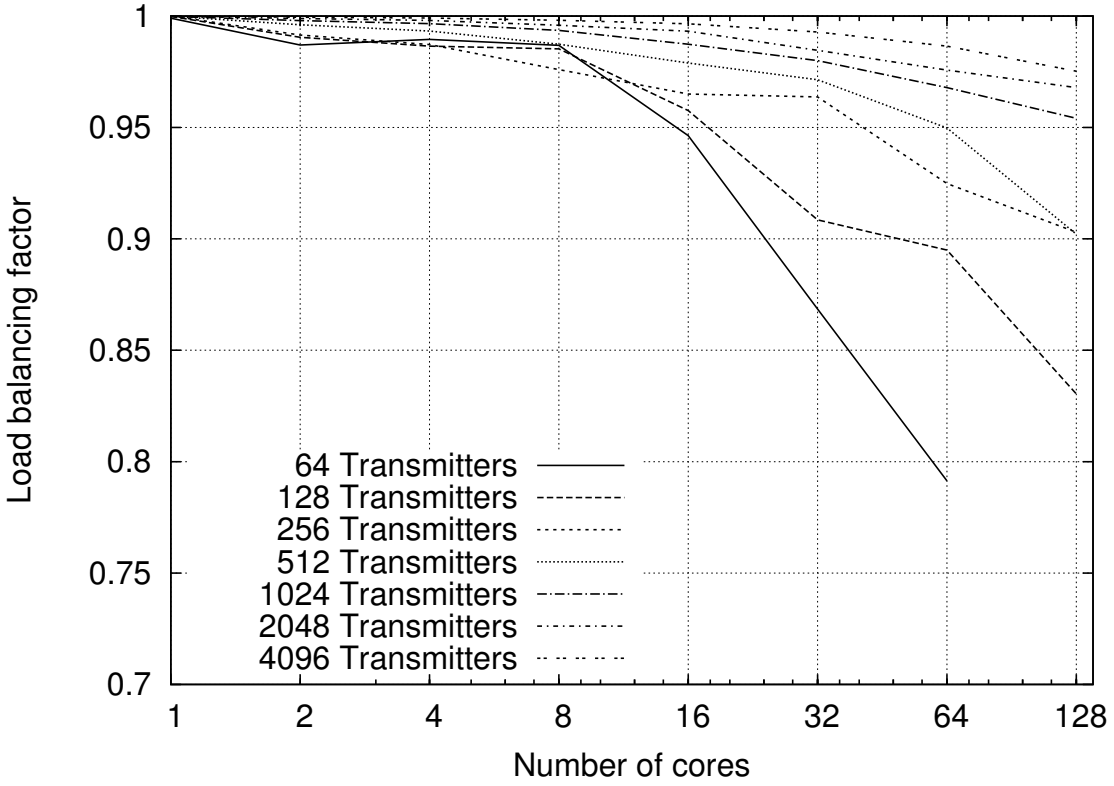

Figure 19: Load balancing among worker processes.

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 19 on page 40, 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. [53]. 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 [73]. 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 [74].

The task-parallelization problem within the GRASS environment has been addressed by several authors in different works. Campos et al. [61] 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. [60] 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. [75] 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 Framework parameter tuning

This chapter 4 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.

4.1 Introduction???

Even after almost 10 years after the launch of the first commercial UMTS 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 hardware, configuration parameters and their evaluation-time complexity.

Although different mathematical models have been proposed for radio propragation modelling, 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 within a bearable error range. Of course, the number of possible combinations of models and parameters grows exponentially if we also take into account various environmental characteristics, such as population density, terrain relief, land use, and a continously growing number of network cells, that are indeed compulsory for keeping the error low.

Some of them are more suitable for free space propagation. Others are better for urban environments, a third groups is even better only at suburban or rural enviroments. Specifically, when talking about radio propagation models, threre are manly three clearly distinguishable groups, namely: statistical models, deterministic models, and combinatorial models.

Statistical models ... ???

Deterministic ???

Combinatorial ???

Despite an interesting pallete of options of commercial tools, available for radio propagation modelling, the common thread among all of the them is the restricted nature of its usage and the lack of adaptation with cumbersume user interfaces.

In this work, we present a self-adaptive radio propagation tool. The tool can fine-tune itself before-hand with field measurements in order to maximize the precision of the radio propagation prediction in the target area.

4.2 Related work???

In [53], Hrovat et al. developed a radio planning tool for the GRASS GIS system. Their work was based on well-known radio propagation models (e.g. Okumura-Hata and COST 231), and a reverse-engineered one, based on a commercial tool by Ericsson. The results reported show that the open source radio planning tool shields comparable results to those of the commercial tools. Moreover, both tools have similar stddev from field measurements.

4.3 Problem description

Our main objective is 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 will combine field measurements in a feedback loop with an optimization algorithm over 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, manually calculated by the radio engineers at Telekom Slovenije, d.d., the algorithm should optimize the model parameters so that an error measurement of the radio-propagation prediction to a given set of field measurements is minimized.

By fine-tuning the model parameters per cell, we achieve region independence since the field measurements are used as reference. Moreover, we aim at lowering the error of the model that is tuned for different types of regions, e.g. urban, suburban and rural. The use of an automated system, backed by a database, effectively facilitates the daily tasks of calculating radio-propagation predictions, since the optimized parameters for a specific cell are directly read from a database. Therefore, it is feasible for the radio engineer to manage many independent sets of optimized parameters for numerous cells. Moreover, the user would decide whether to re-run the parameter optimization, or just read the already optimized parameters directly from the database. Additionally, the level of accuracy for the radio-propagation prediction may be limited, by either setting an optimization-time limit or an minimum error limit, during which the system fine-tunes the parameters of the mathematical model.

???

Indeed, based on experimental results showed in [53], we are already confident that well-known models, like Okumura-Hata and COST 231, provide good results in a feasible amount of time. This means that with no “evolved” mathematical models whatsoever, the results should be satisfactory for the average case, having the extra value of the calculation speed gained on GPUs. Moreover, the good results showed by these well-known models provide a favorable starting point for the evolution of potential better models.

???

4.4 Problem elements

In this section we introduce all the elements taking part of the self-adaptive radio propagation tool.

4.4.1 Ericsson 9999 model

This radio-propagation model was introduced by Ericsson in 2006, as an extension of the well-known Hata model [ref!Comparison_2_11], designed for frequencies up to 2000 MHz. The suitability of this model comes from the fact that it contains a number of adjustable parameters, which adapt the model according to a given scenario. Equation (9) describes the path loss as evaluated by this model.

$$pl(d,\beta) = a_0 + a_1 \log(d) + a_2 \log(H_A) + a_3 \log(d)\log(H_A) - 3.2\left[\log(11.75 \cdot H_R)\right]^2 + 44.49\log(F) - 4.78\left[\log(F)\right]^2, \quad (9)$$

where $\beta = (a_0, a_1, a_2, a_3)$ is the vector containing the tuning parameters of the model, d is the distance (in kilometers) from the transmitter to the topography point, 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 in MHz.

4.4.2 Differential ant-stigmergy algorithm

As the optimization algorithm we have chosen the differential ant-stigmergy algorithm (DASA).

Based on the metaheuristic Ant-Colony Optimization (ACO) [76], the DASA [77] 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 (10):

$$X_a = \{x_1, x_2, \ldots x_i, \ldots, x_D\} \quad (10)$$

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 [77].

4.4.3 Field measurements

The field measurements were taken using a small truck equipped with the spectrum analyzer Rohde & Schwarz [ref!]. The spectrum analyzer was connected to an external omni antenna mounted on the roof of the truck, at roughfly 2 meters above the ground, taking measurements at a rate of ??? per second, with the symbol rate set to ??? Mhz. To accurately establish the measurement location points, a GPS unit was used. The

measurement locations covered all the streets within the target area, with over ??? field-measurement points taken at more than ??? locations.

To minimize the impact of different driving speed, traffic lights and other traffic condictions arising during the measurement round, all field measurements were post-processed so that a single value, the median, is calculated for each of the measured locations. The resulting received signal power was used to estimate the path-loss prediction corresponding to each of the measurements.

4.4.4 Optimization objective

The optimization objective consists of adjusting the parameters of the Ericsson 9999 model, i.e. a_0, a_1, a_2, a_3 in Equation (9), to best fit a given field-measurement set. Our data set consists of N data pairs, (pl_i, m_i), $i = 1, \ldots, N$, where N is the number of field measurements of the current cell, $pl(d_i, \beta)$ represents the path-loss value at measurement point i, as defined in Equation (9), and m_i is the field measurement at the point i. By applying the least squares method [ref!], our goal is to minimize the sum, S^*, of squared residuals, i.e.

$$S^* = \min \sum_{i=1}^{N} r_i^2, \tag{11}$$

where a residual is defined as the difference between a field measurement and a path-loss value predicted by the model, i.e.

$$r_i = m_i - [cell\,power - pl(d_i, \beta)]\,. \tag{12}$$

4.5 Implementation

As the implementation platform for connecting the different system components together, we have chosen the open source Geographic Resources Analysis Support System (GRASS). This Geographic Information System (GIS) software is used for geospatial data management and analysis, spatial modeling, and visualization [ref!grass]. Being an open system, GRASS provided us with a great deal of essential built-in functionality, such as functions for displaying results of geographical maps, importing diffetrent raster and vector formats, database connections for external attributes, geographical coordinates convertion, etc. For additional information about the GRASS, we refer the reader to the numerous guides and tutorials available online.

One of the main reasons for choosing GRASS as our implementation platform resides in its open nature. This fact helped us not only to achieve operating-system independence, but most importantly to implement our system as native GRASS modules. We achieved this by taking advantage of the very-well documented GRASS Application Programing Interface (API), which is available for languages such as C and Python. Therefore, following the modular composition of GRASS itself, we have implemented a separate GRASS module for each independent component of our system, namely:

- the evaluation component, which consists of the modules *r.eric*9999, *r.sector* and *db.measurement*; and

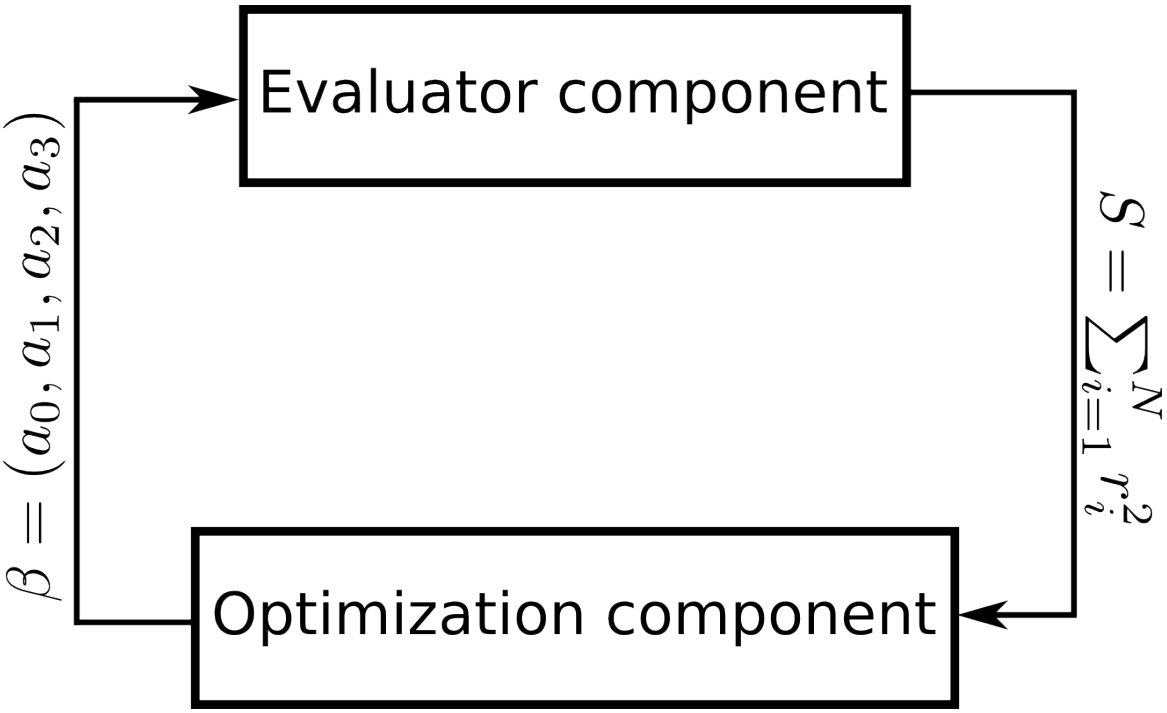

Figure 20: *System architecture and data flow.*

- the optimization component, which consists of a single module called *dasa*.

Figure 20 depicts the different system components and the data flow among them.

4.5.1 Evaluation component

This section describes the different modules that contained in the evaluation component of the system. Their connections and data flow is depicted in Figure 21. This particular component follows a similar internal organization as the radio planning tool developed by Hrovat et al. [53].

4.5.1.1 Path-loss model

The module *r.eric*9999 implements the Ericsson 9999 path-loss model, which was previously introduced in Section 4.4.1.

4.5.1.2 Antenna diagram influence

The module *r.sector* considers the antenna radiation diagram of the current cell and its influence over the path-loss calculation of the isotropic source for a specific region. The module uses the raster map containing the path-loss data for the isotropic source (i.e. the output of module *r.eric*9999), and the radiation diagram of the antenna, including beam direction, electrical and mechanical tilt, and antenna gain, as the input data for calculating the actual path-loss for the currently analyzed cell. The output of this module is saved in a raster map for further processing. Figures 22 shows an example of applying the *r.sector* module to output, generated by the *r.eric*9999 module.

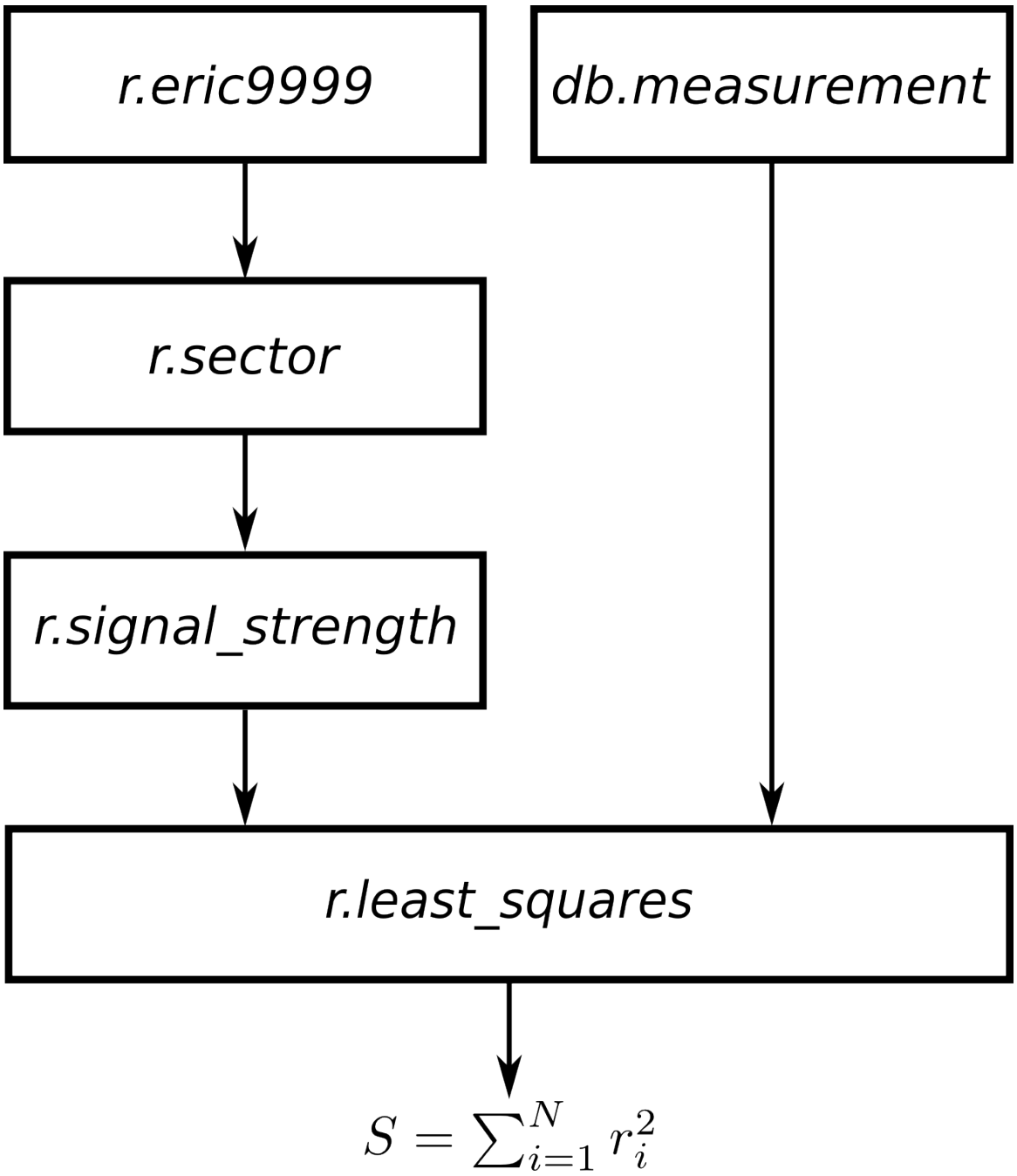

Figure 21: *Evaluation component structure and data flow.*

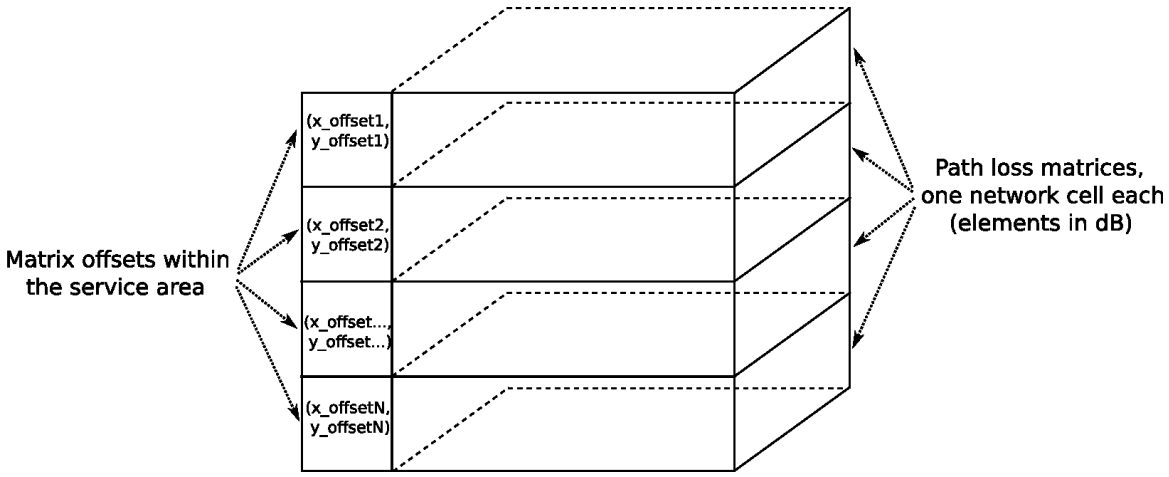

Figure 22: *Example run of the r.sector module.*

4.5.1.3 Received signal strength

The output of the module *r.sector* is used as input for calculating the coverage prediction of the cell being analyzed. Each point within the raster contains the received signal streght or RSCP [ref!], resulting from the combination of the path-loss and the cell transmit power. The output of module in a raster map for further processing.

4.5.1.4 Least squares

Once the coverage prediction has been calculated and saved, this module continues the evaluation sequence by comparing each measurement of the current cell within the area, retrieved from the database with the module *db.measurement*, with the predicted received signal strength jus calculated. Consequently, the objective function defined in Equation (11) is applied for each field measurement i to calculate the cost of the current solution $\beta = (a_0, a_1, a_2, a_3)$.

4.5.2 Optimization component

Because of the metaheuristic nature of the DASA algorithm, it does not need any problem-specific knowledge to be a good optimization tool. On the other hand, there are some parameters that influence the way DASA explores the search space. After some experimental simulations, we have set the configuration 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.

Table 4: *Network statistics.*

	Cells $[m]$	Area $[km^2]$
Net_1	77	100
Net_2	23	306.25
Net_3	129	405

Table 5: *Network parameters.*

Parameter	Net_1	Net_2	Net_3
p_c^T	15.00 W	19.95 W	15.00 W
τ_0	$1.55{\cdot}10^{-14}$ W	$1.55{\cdot}10^{-14}$ W	$1.55{\cdot}10^{-14}$ W
γ_c	0.01	0.02	0.015

4.6 Simulations???

4.6.1 Test networks

All the test networks, Net_1, Net_2 and Net_3 are subsets of a real UMTS network deployed by Mobitel Telecommunication Services, Inc. in Slovenia. The path-loss predictions are calculated using the COST231 model [78], using a digital evaluation model of 100 m^2 resolution as input data and a receiver height of 1.5 m above ground. The requirements for SIR coverage were provided by experts of the Radio Network Department at Mobitel Telecommunication Services, Inc.

Net_1 is deployed over a densely populated urban area. For this reason, the SIR coverage threshold is a lower, since network capacity is the dominating factor, whereas coverage is flexible because of a higher cell density, i.e. more base stations per surface unit. Net_2 represents a network deployed over a dominant rural area, meaning that network capacity may be reduced at the cost of better coverage, since each cell must cover a greater area. The last network, Net_3, represents a suburban area with a highly-dense populated, but relatively small, downtown center, where a compromise between network capacity and coverage has to be achieved.

Based on the data available, we have produced network configurations based on the attenuation-based approach. These configurations represent what could be an initial network setup by common planning standards [79]. Moreover, such configurations are also very straightforward to calculate by a network planner. Table 4 shows some statistics of the test networks used. The parameter values used during experimentation are shown in Table 5.

4.6.2 Algorithm parameter settings

After short experimentation, we determined the parameter settings for the optimization algorithm. There was no fine tuning of parameters for each problem instance. Nevertheless, we gained valuable information regarding the agent's behavior that we used to set the following parameter values:

- *increase rate* was set to 0.2 dB;
- *decrease rate* was set to -0.1 dB;

Table 6: *Optimization results.*

	Attenuation-based		Parallel agents	
	Total power [W]	Average cell power [W]	Total power [W]	Average cell power [W]
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

- *number of agents* was set to 16; and
- 10,000 *changes per agent* were allowed.

4.6.3 Experimental environment

All experiments were done on a 4-core, hyper-threading, Intel i7 2.67 GHz desktop computer with 6 GB of RAM running a 64-bit Linux operating system. The GPU hardware was ATI HD5570, with 1 GB DDR3 RAM. The implementation language used was C, with OpenCL and OpenMPI extensions.

4.6.4 Optimization results

The results achieved by our optimization approach improved the objective significantly, as it is shown in Table 6. Results show that we reduced pilot power usage in all networks and kept the service area under full coverage. Moreover, we may see the solution for Net_1 improved the attenuation-based setting by more than 300%. For Net_2, the improvement observed is around 232%, with an improvement of more than 170% for Net_3. These means that network capacity 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 [79].

After collecting data from ten independent runs, we generated convergence graphs, shown in Figures ???. The graphs contain feasible solutions only, i.e. solutions that meet the full-coverage constraint. Unfeasible solutions were marked with a value of inferior quality than the worst solution found by the algorithm in all ten runs. In case of Net_1, the value was set to 428, for Net_2 the value was set to 129 and for Net_3 the value was set to 1,435.

The analysis of convergence graphs of Net_1 and Net_2 shows that the algorithm quickly converges at the beginning, followed by a steady improvement of intermediate solutions. In Net_1 we notice additional improvement of the solutions found even at towards the end. This fact suggests that longer runs would potentially find even better solutions in this case. For the instance Net_3, we observe a slower initial convergence, with steady improvement of intermediate solutions and no significant solution enhancement towards the end. This fact, together with the aforementioned results, suggest that this problem instance presents a more difficult optimization case than Net_1 and Net_2. Further investigation would be needed to determine the source of this behavior. Nevertheless, the improvement observed is, in average, around 100%.

4.7 Conclusion???

In this paper, we have addressed the problem of providing full coverage to a service area of a UMTS network by using a minimum amount of pilot power. We have put emphasis on the confluence of a real-world problem, with live data from a deployed mobile network, with state-of-the-art parallel GPU hardware and implementations that, to the best of our knowledge, has never been dealt-with before.

We have presented a parallel-agent approach, which is aimed at giving good solutions to big problem instances in an acceptable amount of time. The experimental results show that our approach is able to find competitive solutions, when compared to other common radio-planning methods [79]. The presented results also demonstrate that our algorithm is able to find high quality solutions even for large networks, that contain many cells over a large service area. This fact indicates that our approach could be successfully applied to bigger problem instances.

GPU architectures not only allow implementation of parallel heuristics in a natural way, they also substantially improve the performance of the optimization process. We reported and validated the great performance gain by experimentation on problem instances of different sizes.

After successfully implementing the objective-function evaluation on GPU, we realized that the efficiency of this approach was limited by the CPU-to-GPU data transfers. Nevertheless, even with such implementation, we have already obtained substantial speed-up.

To deal with the CPU-to-GPU data transfer issue, we implemented a fully-enabled GPU optimization system that achieved impressive speed-up. Still, we had to consider different data representation schemes for the problem elements, so to avoid memory limitations on the GPU device. 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. [80, 81]) 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.

5 Experimental evaluation: the service coverage problem

In the context of coverage planning and control, the power of the CPICH signal determines the coverage area of the cell. It also impacts the network capacity, and thus the quality of service. Pilot power is the parameter that allows us to control the strength of the CPICH signal. A higher power for pilot signals means better coverage. On the other hand, more pilot power translates in decreased network capacity.

We consider 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.

5.1 Introduction

The coverage problem in third-generation (3G) mobile networks has received a great deal of attention in the last years. Its complexity demands the confluence of different skills in areas such as propagation of radio signals, telecommunications and information systems, among others.

The problem becomes even more complex as conflicting measures to compare different proposed solutions are taken into account. These aspects include network capacity, quality of service, service coverage, and, recently, issues related with human exposure to electromagnetic fields generated by base station antennas [82]. Public opinion has been extremely sensitive regarding this issue, and thus many countries have already imposed safety standards to limit the electromagnetic field levels.

It is clear that even after almost 10 years after the launch of the first commercial UMTS 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 common pilot channel (CPICH). The CPICH transmit power is common to many different planning and optimization problems in UMTS networks [83].

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 [79]. 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.

From the network 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 [79]. Moreover, as applications demand for mobile internet access and data services increases [84], so does the pressure on existing network infrastructure, making parameter optimization the only viable short-term solution [83].

There are different approaches in the literature that are able to solve the coverage problem [83,85]. Some of them even claim to achieve near-optimal solutions [40]. As a matter of fact, such formulations have proven useful only for small network instances and often fail when challenged with real-world networks.

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 [82, 86], but also in other fields [50, 87]. Nevertheless, we have not observed any similar optimization method for solving the service coverage problem in mobile networks [88]. 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 [89]. We tackle the problem of computational-time complexity when dealing with big problem instances by implementing a parallel version of our agent-based algorithm entirely on GPU. This minimizes overhead when deploying a larger number of agents working in parallel over the service area, limited only by the amount of memory available.

We tested our algorithm on subsets of a real UMTS network deployed in Slovenia. All network-related data were provided by the mobile operator Mobitel Telecommunication Services, Inc. The results show that the solutions found, and more importantly their quality, are greatly improved when compared to other common planning techniques.

We begin our discussion by introducing previous work and a description of the coverage problem, where we formally introduce some of its key elements. We then discuss our parallel-agent approach in detail, as well as the strategies used for result comparison. Having introduced the parallel-agent approach, we move on to describe the GPU implementations of its two key elements: the objective-function evaluator and the agents. Simulations and experimentation, performed on three sub-networks of a real mobile network deployed in Slovenia, follow. We conclude with an overview of the achieved results, from the optimization as well as the implementation points of view, and discuss future research directions.

5.2 Previous work

In [40], Siomina and Yuan considered the problem of minimizing the total amount of pilot power subject to a full coverage constraint. They tackled the problem with an iterative linear programming approach, reporting very good results for some small-sized test networks. The authors also noted that bigger problem instances could not be solved because of hardware constraints on the target platform.

In a different work [88], we tackled the full-coverage problem of the service area under optimization. Reported experimentation on the same problem instances as in [40] showed improved quality at the cost of longer running time. Moreover, our

approach was not limited by any hardware restrictions, since it successfully tackled bigger problem instances, finding high quality solutions even for large networks. Such solutions were found in a reduced amount of time by increasing the number of agents deployed during optimization without compromising solution quality.

The algorithm in [88] is the basis for the optimization algorithm presented in this paper, with some essential improvements, namely:

- the original implementation was completely CPU-based and used a black-box coverage evaluator;
- the algorithm presented here contains improvements in the agent's behavior during optimization, including the introduction of the so-called "special" agents and some fine-tuning of their step sets.

5.3 Problem description

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. We consider the pilot power minimization problem subject to a full coverage constraint of the service area.

Because the mathematical model of the problem is not of primary interest here, we will just outline it, so that all problem elements are formally defined and represented. For additional information regarding mathematical models of comparable problems, see [83].

5.3.1 Problem elements

We start by considering a UMTS network of m cells and use C to denote the set of cells, i.e. $C = \{1, ..., m\}$. A pixel grid of a given resolution represents the service area for which the signal propagation predictions are known. Let n denote the total number of pixels in the service area and let S denote the pixel set, i.e. $S = \{1, ..., n\}$. We also denote att_{cs}, $0 \leq att_{cs} \leq 1$, as the attenuation factor between a cell c and a pixel s, which is calculated by performing signal propagation predictions for every pair of $c \in C$ and $s \in S$.

For every $c \in C$, we define p_c^T as the total transmission power available in cell c. This power is shared among all channels in the cell (i.e. CPICH, other common channels, and dedicated traffic channels). We define p_c as the amount of power allocated to the pilot signal of cell c, where p_c may adopt any value from a finite set of possible pilot power levels, $P_c = \{p_c^1, p_c^2, ..., p_c^T\}$. Consequently, the received pilot power of cell c in pixel s is $att_{cs}p_c$.

Considering the full coverage constraint, each pixel in the service area should have at least one cell covering it. We assume that a pixel s is under coverage of a cell c if its signal-to-interference ratio, SIR, at pixel s is not lower than a given threshold, γ_c, i.e.

$$SIR(c, s) = \frac{p_c att_{cs}}{\sum_{i \in C} p_i^T att_{is} + \tau_0} \geq \gamma_c \tag{13}$$

where τ_0 is the thermal noise. In (13), we are assuming that all cells in the network operate at full power, which is the worst case scenario. This ensures that even under heavy user traffic, full coverage of the service area is maintained, because of the cell-breathing principle [79]. The same assumption has also been used in [40, 89].

The optimization problem corresponds to finding the pilot power levels p_c, for all cells $c \in C$, such that coverage of at least b pixels is guaranteed, while the total amount of pilot power used is minimized. Since we are considering full coverage, we denote $b = n$.

5.3.2 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 [90]. 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.

5.3.3 Optimization objective and constraints

The optimization objective is defined as follows

$$P^* = \min \sum_{c \in C} p_c; \tag{14}$$

subject to

$$\frac{\sum_{s \in S} cov(s)}{b} = 1, \tag{15}$$

where

$$cov(s) = 1 \;\; if\, and\, only\, if \;\; \exists c | SIR(c, s) \geq \gamma_c \tag{16}$$

$$cov(s) = 0 \;\; otherwise$$

The definition of (16) provides us a simple way of asserting the coverage of a given pixel, s. It follows that if the pilot signal of at least one cell c satisfies the imposed SIR threshold, γ_c, the pixel is covered and hence $cov(s) = 1$.

5.4 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 attenuation-based pilot power, presented in [85], in which a pixel of the service area is always covered by the cell with the highest attenuation-factor value. The second strategy is our parallel-agent approach, based on ideas inspired by two-dimensional cellular automata [91] and metaheuristics [92]. A detailed description is given in section 5.4.2.

Similar criteria for result comparison have also been used in [40].

5.4.1 Attenuation-based pilot power

The first heuristic for setting the pilot power of all cells in the network is known as attenuation-based, since it relies on the attenuation factor, att_{cs}. A pixel of the service area, s, is always covered by the cell exhibiting the maximum att_{cs}. Whenever the maximum available power, p_c^T, 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 pixel s. Hence, under this assumption, we identify the cell c covering pixel s as

$$p_{c(s)} = \min_{c \in C} p_{cs} \tag{17}$$

Picking the cells conforming to (17) and setting the pilot powers accordingly, we achieve full coverage of the service area with a solution exhibiting a total pilot power of

$$P^{Att} = \sum_{c \in C} \max p_{c(s)} \tag{18}$$

The procedure to find a cell $c(s)$ for every pixel in the service area consists in sorting, in descending order, all pixels by the maximum attenuation factor value, att_{cs}, among all cells, i.e. $att_{c(s),s}$. The solution is thus established by the first b pixels of the sorted sequence, taking the maximum pilot power setting for a cell into account, i.e. $p_{c(s)}$.

5.4.2 Parallel-agent approach

In the parallel-agent approach, a set of autonomous worker agents explore the geographic area, targeted to be under mobile network coverage, in order to optimize the pilot power consumption of the network. Each agent randomly moves over the service area as it dictates different changes to the pilot power of the cells. An objective-function evaluator performs radio propagation predictions based on each agent's proposed change.

The search process during optimization is strictly random. However, several physical properties that are exclusive to the problem being solved are being exploited during 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 of the independent nature in agent's behavior, a parallel implementation is fairly straightforward to achieve. 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. Figure 23 gives an overview of the optimization system architecture. Within this GPU-only architecture, agents work in a parallel and autonomous manner, while the evaluator reacts to agents' changes.

5.4.2.1 The agents

The agents apply the pilot power changes based exclusively on local information. Each of them encapsulates a set of steps that is consistently applied as it randomly moves

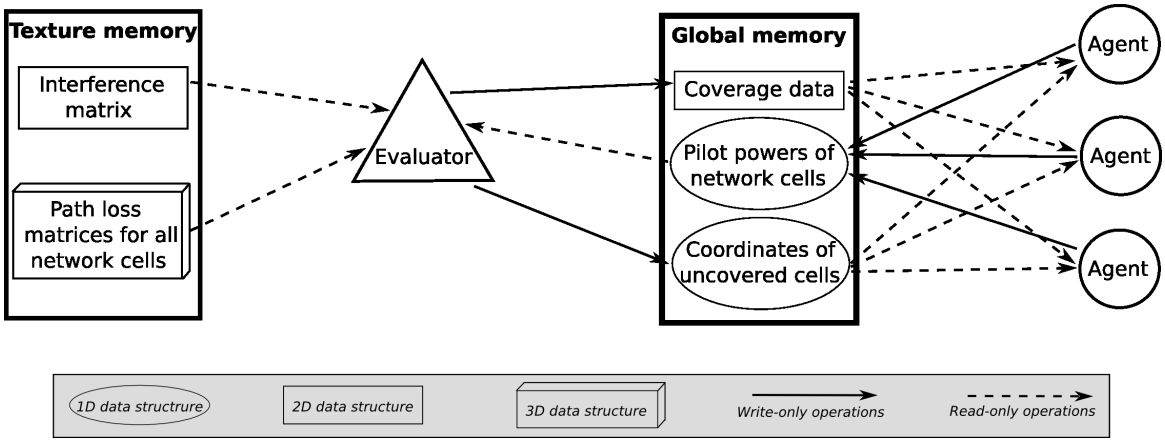

Figure 23: *Architecture of the optimization system on GPU.*

Table 7: *Pseudo-code of the agent's behavior.*

Step	
	repeat
1	**if** *Special agent* ( ) **and** $\mathsf{U} > 0$ **then**
2	$s = random\ element\ from\ U$
	else
3	$s = random\ element\ from\ S$
	end if
4	*move to* s
5	**if** $\mathsf{B(s)} = 0$ **then**
6	*apply* SS_0 *//increase power*
7	**else if** $\mathsf{B(s)} \geq 1$ **then**
8	*apply* SS_1 *//decrease power*
	end if
	while not (*stopping criteria*)

through the service area of the network. Whenever an agent arrives at a pixel s, it identifies the set of cells covering the current pixel, namely

$$B(s) = \{c \in C \mid SIR(c, s) \geq \gamma_c\} \tag{19}$$

The step set applied from this point on directly depends on the cardinality of $B(s)$, while the agent's movement over the service area is determined by the cardinality of set U, $U \subset S$, which is defined as

$$U = \{s \in S \mid \forall c \in C : SIR(c, s) < \gamma_c\} \tag{20}$$

The agent's behavior is dictated by the pseudo-code shown in Table 7. Steps 1 to 4 are responsible for guiding the agent's movement. The coordinates are selected randomly from two sets. The first, S, is the set of all pixels in the service area. The other one, U, is the set of pixels that are currently not being covered by the network. Only "special" agents may select pixel coordinates of the set U. It follows, that an agent's movement depends on its "specialty" and the number of pixels not covered by the current solution. During steps 5 to 8, the agent applies step sets SS_0 and SS_1 based on the number of cells in $B(s)$.

Table 8: *Pseudo-code of step set* SS_0.

Step	
	repeat
1	$c' = next\,cell\,with\,maximum\,att(s)$
2	$p_{c'} = Adjust\,pilot(c', increase\,rate)$
3	**while** $(p_{c'} \notin P_{c'})$

Table 9: *Pseudo-code of step set* SS_1.

Step	
	repeat
1	$c' = next\,random\,cell(B(s))$
2	$p_{c'} = Adjust\,pilot(c', decrease\,rate)$
3	**while** $(p_{c'} \notin P_{c'})$

If the agent's current location, at pixel s, is not covered by any cell (i.e. $\mathsf{B(s)} = 0$), the step set SS_0 (shown in Table 8) is applied. It starts by selecting the cell with the maximum attenuation factor at pixel s (step 1). If many cells have the same value, one is randomly picked out of them. Once c' is uniquely identified, the agent changes its pilot power by *increase rate* dB (step 2). By using a higher *increase rate*, the network shall potentially cover s, as well as some neighboring pixels. Areas without coverage usually contain many uncovered pixels grouped together, forming irregular uncovered islands.

The step set SS_1 in Table 9 is applied whenever the agent's current location, at pixel s, is covered by one or more cells (i.e. $\mathsf{B(s)} \geq 1$). The first step randomly selects a cell from $B(s)$. The agent shall decrease the pilot power of c' in step 2. This practice keeps the coverage constraint valid over s, although it might potentially break it on other pixels. Ideally, every pixel would be covered by exactly one network cell, although this is just a representation of a perfect solution that is almost entirely unreachable, because of the irregularity in network topology and terrain.

In both step sets, SS_0 and SS_1, the agent makes sure that the new pilot power setting, i.e. after applying the change, is an element of $P_{c'}$. If this is not the case, cell c' is discarded and another cell is selected at the first step of SS_0 and SS_1, adjusting its pilot power accordingly.

The values *increase rate* and *decrease rate* are configurable parameters that should be set before starting the optimization process. They indicate the dB adjustment proposed to the pilot power of cell c' and are based on the physical properties of the problem being solved. Namely, lowering the pilot power of a cell decreases the interference at pixel s. Moreover, the target SIR value at pixel s is reduced under lower interference; thus coverage of this pixel may be achieved with lower pilot power. On the other hand, by increasing the pilot power of a cell with the maximum att_s, we improve coverage by evenly distributing the power among different network cells, since the selected cell, c', is, on average, the nearest to the present location.

5.4.2.2 The evaluator

The evaluator represents a central component of the optimization system, since it reacts to agents' changes by recalculating the value of the objective function, i.e.

coverage of the service area and the total pilot power used by all the cells in the network being optimized.

After a short initialization, during which the path-loss matrices for all the cells and the interference matrix for the whole area are calculated, the evaluator computes the service area coverage, based on the pilot powers supplied as the initial solution from which the search process begins. Initial solutions are randomly generated sets, containing valid pilot power settings that fulfill the coverage constraint. The evaluator then waits for agents' changes to arrive and calculate subsequent objective function values accordingly.

It is responsibility of the evaluator to maintain a special part of memory, intended for keeping track of the uncovered pixels in the service area, constantly updated. Whenever the current solution is not valid because of uncovered pixels, i.e. (15) does not hold, some "special" agents randomly select an uncovered pixel coordinate from this portion of memory so that a valid solution may be reached again. It should be noted that these "special" agents shall only apply step set SS_0 for as long as the solution is not valid. The portion of "special" agents that may work in correcting the current solution is an optimization parameter.

As it has been mentioned before, this constraint-repairing strategy enhances certain properties of the search process performed, namely:

- increased exploration of the search space, as different regions are also being inspected, and
- to enable the algorithm to escape from local optima, leading the search to other areas containing potentially good solutions.

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

5.5 Implementation

We have chosen the Open Computing Language (OpenCL) [93] as the implementation platform of our optimization system on GPU.

OpenCL is an open parallel computing API designed to enable GPUs and other co-processors 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 [94]. 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 speed-up on different hardware without the need of changing the implementation.

One unfortunate consequence of the vendor variety is that NVIDIA's CUDA [95] and OpenCL documentation present disparate naming conventions for some key components. For the sake of consistency, in Table 10, 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 10: *Terminology translation between OpenCL and CUDA [96].*

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(LMF)	__syncthreads( )
get_local_id(012)	threadIdx.xyz
get_group_id(012)	blockIdx.xyz
get_global_id(012)	*(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.

The evaluator was completely implemented on the GPU, because its performance has a great impact on the speed of the optimization system as a whole. 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. Therefore, it is a challenging task to accommodate all the needed elements on GPU memory, which is notably smaller than the RAM memory usually available on modern desktop computers.

5.5.1 Objective function evaluation on GPU

As it was mentioned before, the crucial importance of the evaluator as an element of the optimization system made it the first component to be implemented on the GPU and, thus, have a greater gain on the performance speed-up of the system.

The GPU implementation of the objective function evaluator is based on work done by Hrovat et al. [53]. 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.

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 78000 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 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

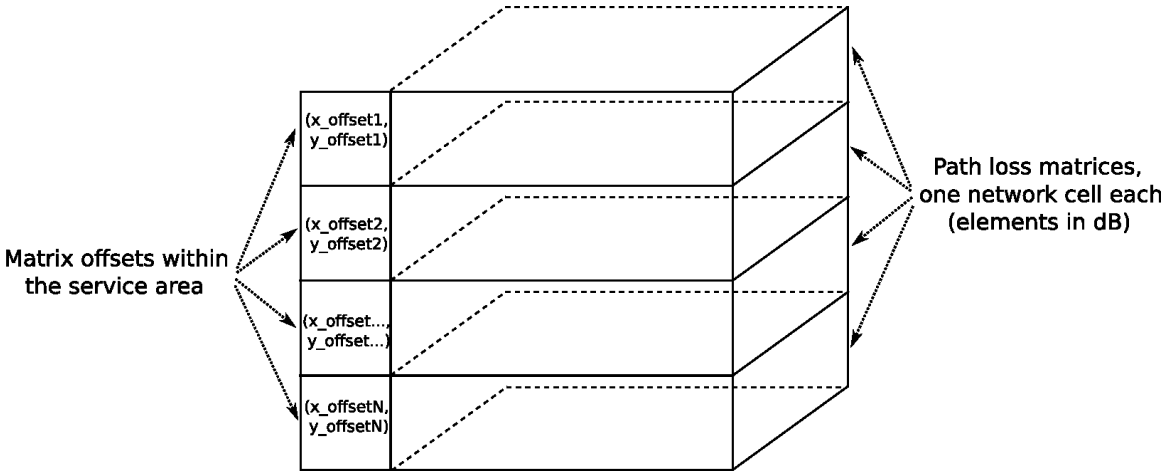

Figure 24: *Memory organization of the path-loss matrices for network cells.*

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 [79]. 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 [79], 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 24. 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.

5.5.2 Parallel agents on GPU

The autonomous and programmable nature of the agents make them ideal for parallel and GPU-based implementations. By lowering the complexity of the step sets applied at each time, we were able to tackle a large optimization problem with outstanding results in very short time.

For the agents' kernel, only one thread block is launched. It contains as many threads as there are agents deployed during the optimization, organized in a 1D grid. Each thread randomly generates a coordinate within the service area, using the system time in milliseconds as a random seed. Because OpenCL lacks functions for random-number generation, we implemented a simplified version of Marsaglia's generator [97]. Afterwards, each thread analyzes the received signals at the current coordinate by applying step sets SS_0 or SS_1, as it has been explained in section 5.4.2. Each thread

saves in shared memory the outcome of its analysis, containing the *id* of the network cell (at position $2\times threadIdx$), and the pilot power setting (at position $2\times threadIdx+1$). The new pilot power is calculated as the dB difference from the previous one, based on the values of *increase rate* and *decrease rate*. Since both numbers, i.e. the cell *id* and the pilot power, are of type *unsigned short*, there is enough room in a 16 Kb shared memory block to allocate up to 4,096 independent agents. Therefore, this number is not bounded by shared-memory size, since most GPUs have a limit in the number of threads per block of 256, 512, etc. The last step involves saving the new pilot powers back to their containing vector in global memory. This is done by only one of the threads within the block, to avoid memory-access conflicts. At this moment, the sign of updated pilot powers is changed to indicate that coverage re-calculation is needed. Clearly, the vector containing pilot powers in global memory is of type *int*, as it must allow signed values. Nevertheless, a single pilot power setting never exceeds 65,535 in milliwatt units. In case there is more than a new setting for a specific network cell, the median is calculated and applied as the new pilot power for that cell.

Even though coalesced access is not achieved by the agents' kernel, its sole implementation provided enhanced performance, since the compulsory data transfers between the CPU and the GPU at each time step during optimization are minimized. It also produces truly parallel behavior of the agents, as they apply pilot power changes at the same time.

5.6 Simulations

5.6.1 Test networks

All the test networks, Net_1, Net_2 and Net_3 are subsets of a real UMTS network deployed by Mobitel Telecommunication Services, Inc. in Slovenia. The path-loss predictions are calculated using the COST231 model [78], using a digital evaluation model of 100 m^2 resolution as input data and a receiver height of 1.5 m above ground. The requirements for SIR coverage were provided by experts of the Radio Network Department at Mobitel Telecommunication Services, Inc.

Net_1 is deployed over a densely populated urban area. For this reason, the SIR coverage threshold is a lower, since network capacity is the dominating factor, whereas coverage is flexible because of a higher cell density, i.e. more base stations per surface unit. Net_2 represents a network deployed over a dominant rural area, meaning that network capacity may be reduced at the cost of better coverage, since each cell must cover a greater area. The last network, Net_3, represents a suburban area with a highly-dense populated, but relatively small, downtown center, where a compromise between network capacity and coverage has to be achieved.

Based on the data available, we have produced network configurations based on the attenuation-based approach. These configurations represent what could be an initial network setup by common planning standards [79]. Moreover, such configurations are also very straightforward to calculate by a network planner. Table 11 shows some statistics of the test networks used. The parameter values used during experimentation are shown in Table 12.

Table 11: *Network statistics.*

	Cells $[m]$	Area $[km^2]$
Net_1	77	100
Net_2	23	306.25
Net_3	129	405

Table 12: *Network parameters.*

Parameter	Net_1	Net_2	Net_3
p_c^T	15.00 W	19.95 W	15.00 W
τ_0	$1.55{\cdot}10^{-14}$ W	$1.55{\cdot}10^{-14}$ W	$1.55{\cdot}10^{-14}$ W
γ_c	0.01	0.02	0.015

5.6.2 Algorithm parameter settings

After short experimentation, we determined the parameter settings for the optimization algorithm. There was no fine tuning of parameters for each problem instance. Nevertheless, we gained valuable information regarding the agent's behavior that we used to set the following parameter values:

- *increase rate* was set to 0.2 dB;
- *decrease rate* was set to -0.1 dB;
- *number of agents* was set to 16; and
- 10,000 *changes per agent* were allowed.

5.6.3 Experimental environment

All experiments were done on a 4-core, hyper-threading, Intel i7 2.67 GHz desktop computer with 6 GB of RAM running a 64-bit Linux operating system. The GPU hardware was ATI HD5570, with 1 GB DDR3 RAM. The implementation language used was C, with OpenCL and OpenMPI extensions.

5.6.4 Optimization results

The results achieved by our optimization approach improved the objective significantly, as it is shown in Table 13. Results show that we reduced pilot power usage in all networks and kept the service area under full coverage. Moreover, we may see the solution for Net_1 improved the attenuation-based setting by more than 300%. For Net_2, the improvement observed is around 232%, with an improvement of more than 170% for Net_3. These means that network capacity 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 [79].

After collecting data from ten independent runs, we generated convergence graphs, shown in Figures 25, 26 and 27. The graphs contain feasible solutions only, i.e. solutions that meet the full-coverage constraint. Unfeasible solutions were marked

Table 13: *Optimization results.*

	Attenuation-based		Parallel agents	
	Total power [W]	Average cell power [W]	Total power [W]	Average cell power [W]
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

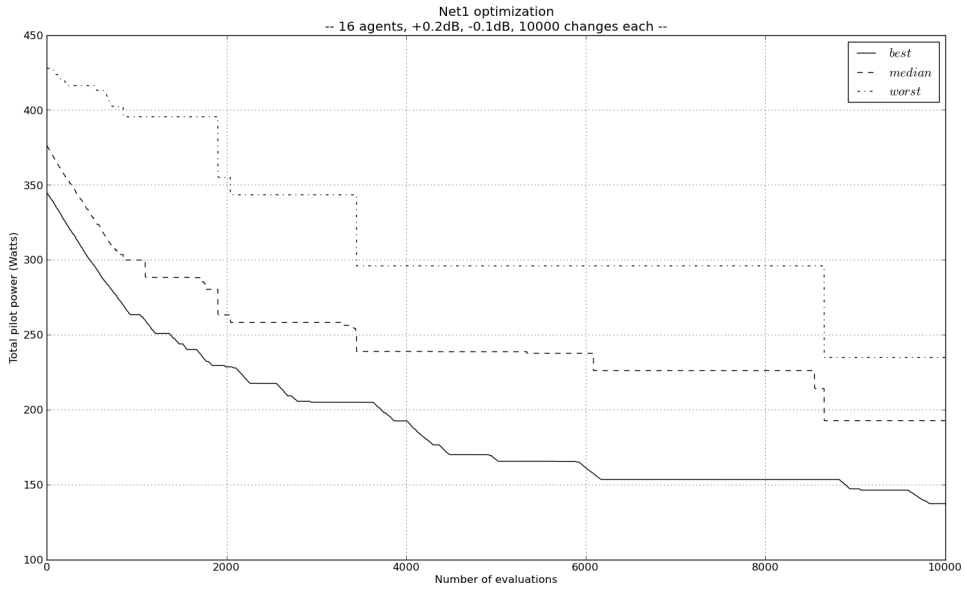

Figure 25: *Convergence results for network Net_1.*

with a value of inferior quality than the worst solution found by the algorithm in all ten runs. In case of Net_1, the value was set to 428, for Net_2 the value was set to 129 and for Net_3 the value was set to 1,435.

The analysis of convergence graphs of Net_1 and Net_2 shows that the algorithm quickly converges at the beginning, followed by a steady improvement of intermediate solutions. In Net_1 we notice additional improvement of the solutions found even at towards the end. This fact suggests that longer runs would potentially find even better solutions in this case. For the instance Net_3, we observe a slower initial convergence, with steady improvement of intermediate solutions and no significant solution enhancement towards the end. This fact, together with the aforementioned results, suggest that this problem instance presents a more difficult optimization case than Net_1 and Net_2. Further investigation would be needed to determine the source of this behavior. Nevertheless, the improvement observed is, in average, around 100%.

5.6.5 Implementation results

After measuring the quality of the solutions given by our parallel-agent approach, we present the experimental results regarding efficiency of different implementations. For this purpose only execution times and speed-up factors are presented. The running time was measured for ten independent runs on each platform. The resulting average

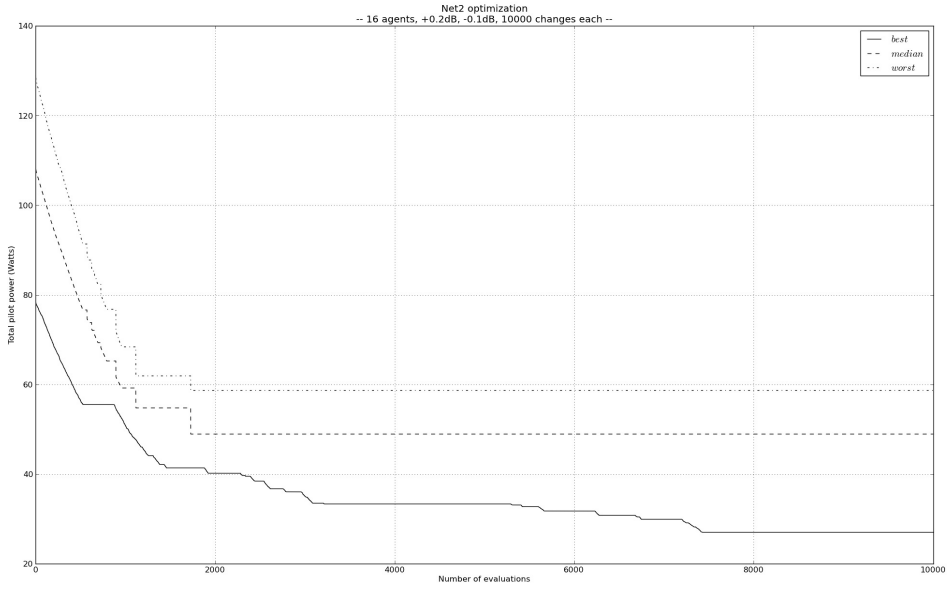

Figure 26: *Convergence results for network* Net_2.

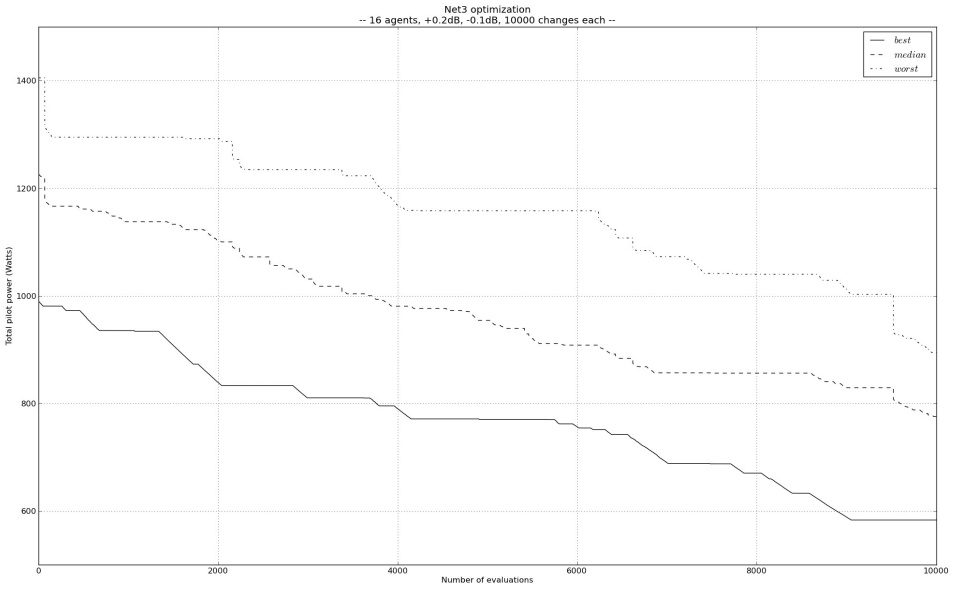

Figure 27: *Convergence results for network* Net_3.

Table 14: *Implementation-efficiency measures.*

	CPU evaluator + MPI agents	GPU evaluator + MPI agents		GPU evaluator + GP
	Avg. time [sec]	Avg. time [sec]	Speed-up	Avg. time [sec]
Net_1	105,455	346	305x	67
Net_2	33,700	195	173x	46
Net_3	191,900	506	379x	117

times are given. The number of changes per agent was limited to 100, while all other algorithm parameters were kept at the same values as previously stated in section 5.6.2.

The results, shown in Table 14, are reported for different sizes of problem instances, i.e. Net_1, Net_2 and Net_3. These networks provide different combinations of network cells and service area size. Results are presented for a CPU-only implementation, including objective-function evaluation on CPU and MPI-based agents, GPU objective-function evaluation and MPI-based agents, and GPU objective-function evaluation including agents on the same GPU. The implementation combining CPU-based evaluator and MPI-based agents is the basis for the speed-up calculation on the other platforms.

It should be noted that the MPI implementation of the agents, used for the first and second measured setups, is not fully parallel, since internal synchronizations at network level are performed, so agents' changes arrive in a serial fashion to the evaluator, before being analyzed.

Function evaluation on the GPU communicating with agents over MPI provides the second measured setup. The evaluator implementation takes advantage of shared memory for thread collaboration within a block and texture memory for constant elements, as is has been explained in section 5.5.1. 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%.

5.7 Summary

In this paper, we have addressed the problem of providing full coverage to a service area of a UMTS network by using a minimum amount of pilot power. We have put emphasis on the confluence of a real-world problem, with live data from a deployed mobile network, with state-of-the-art parallel GPU hardware and implementations that, to the best of our knowledge, has never been dealt-with before.

We have presented a parallel-agent approach, which is aimed at giving good solutions to big problem instances in an acceptable amount of time. The experimental results show that our approach is able to find competitive solutions, when compared to

other common radio-planning methods [79]. The presented results also demonstrate that our algorithm is able to find high quality solutions even for large networks, that contain many cells over a large service area. This fact indicates that our approach could be successfully applied to bigger problem instances.

GPU architectures not only allow implementation of parallel heuristics in a natural way, they also substantially improve the performance of the optimization process. We reported and validated the great performance gain by experimentation on problem instances of different sizes.

After successfully implementing the objective-function evaluation on GPU, we realized that the efficiency of this approach was limited by the CPU-to-GPU data transfers. Nevertheless, even with such implementation, we have already obtained substantial speed-up.

To deal with the CPU-to-GPU data transfer issue, we implemented a fully-enabled GPU optimization system that achieved impressive speed-up. Still, we had to consider different data representation schemes for the problem elements, so to avoid memory limitations on the GPU device. 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. [80, 81]) 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.

6 Experimental evaluation: the SHO alignment problem

This chapter introduces a static network simulator 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.

6.1 Introduction and motivation

In mobile networks, handover is one of the main features that allows user's mobility [21]. 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 Universal Mobile Telecommunications System (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 [21].

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 [21]. 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 [98], 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

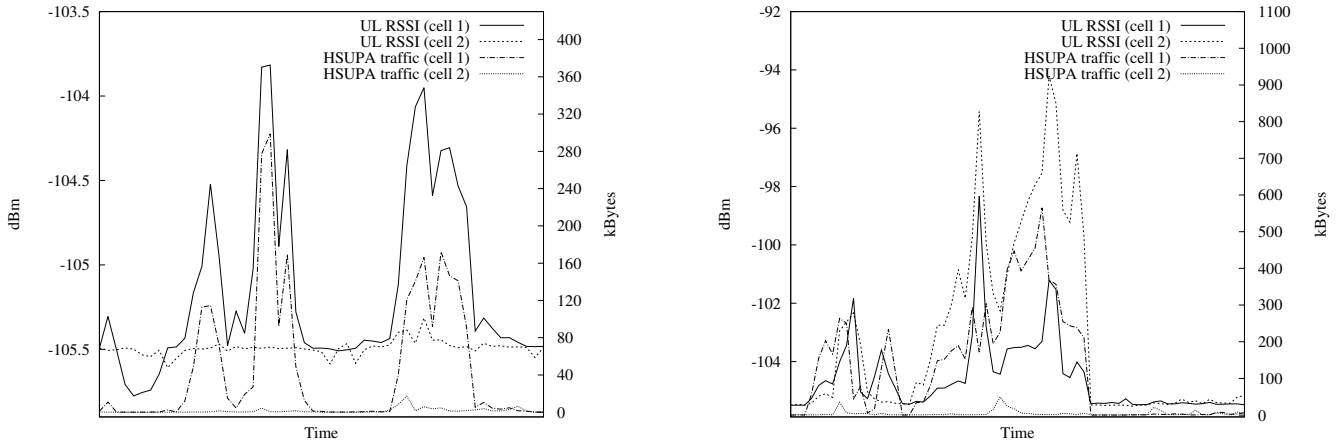

Figure 28: HSUPA traffic and uplink interference with: (a) balanced downlink and uplink SHO conditions; (b) unbalanced downlink and uplink SHO conditions.

High Speed Downlink Packet Access (HSDPA) does not support SHO, while the High Speed Uplink Packet Access (HSUPA) does. This particular distinction has some key implications in the balanced distribution of SHO areas, and thus in the quality of HSPA services [99].

Despite several built-in mechanisms that allow the 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 28, which shows interference behavior during a HSPA connection in normal SHO conditions (a), and 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 [99] 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 appearance of problematic situations as shown in Figure 28.

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, we have decided to define an optimization problem which objective is to find a CPICH power level configuration for all the cells in the working 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), but only the CPICH powers of the cells are to be changed.

In this paper we utilize a static network simulator, based on a state-of-the-art mathematical model [13], 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 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.

The remainder of this paper is organized as follows: in Section 6.2 we give an overview of other works related to CPICH-power and SHO optimization in UMTS. The static network model is presented in Section 6.3, where all the elements of the mathematical model and the objective function are defined. In Section 6.4 we introduce and shortly describe the optimization algorithms used to tackle the balancing problem. The simulations, including their environment and parameter setup, are described in Section 6.5, followed by their results in Section 6.6. We conclude with Section 6.7 by giving a conclusion and guides for future work.

6.2 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 [100] and Internet access by means of HSPA [101].

Some authors tackle optimization problems at the planning stage of the network [102, 103], considering, among other variables, base station locations and hardware. The fact is that 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 tools available when it comes to improvement of the quality of service and troubleshooting the network in the short term.

Optimizing SHO by means of CPICH is an established way of enhancing network capacity when high speed services like HSDPA an HSUPA coexist with legacy technologies [104]. The CPICH transmit power is typically between 5% to 10% of the total downlink transmit power of the base station [105], 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 [21, 106, 107]). The most effective ones are those based on optimization methods [102, 105, 108–110]. Such a wide spectrum of proposed 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.

To the best of our knowledge, there is no reference in the literature to a simulation-based approach to find active downlink and uplink SHO areas. Additionally, as far as we know, the SHO balancing problem as described in this paper has not yet been tackled by any formal optimization method.

6.3 Mobile network model

Following the representation of a static network model from [13], 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 [13].

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

6.3.1 Basic elements

We start by considering a UMTS network with a set of antenna installations (cells), N. A pixel grid of a given resolution represents the service area, A, within which there is a set of mobiles, M. We denote $L^{\downarrow}_{im}$ as the downlink attenuation factor between cell $i \in N$ and mobile $m \in M$. Similarly, we define $L^{\uparrow}_{mi}$ as the uplink attenuation factor between mobile m and cell i. The attenuation factor values are calculated by performing signal propagation predictions for every pair (i, m), $i \in N$, $m \in M$, using the commercial radio planning tool TEMS^{TM} CellPlanner [111], which is used in the Radio Network Department at Telekom Slovenije, d.d.. These predictions already include losses and gains from cabling, hardware, and user equipment.

By introducing a change step of 0.01 dB and bounding the CPICH power of a cell i to ± 2 dB, relative to the CPICH power setting the cell had before optimization, we define a finite set of candidate CPICH power settings for cell i as $P_i = \{p_i^1, p_i^2, ..., p_i^K\}$. By limiting the possible CPICH power settings a cell i may have, we are delineating two important aspects of the problem. First, since we are optimizing a live network, we do not want 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.

6.3.2 Coverage

A mobile m within the area A is under network coverage if at least one cell i covers it. We define the downlink coverage by means of the received signal code power (RSCP)

[21]. Following the current network settings, and including a margin for interference derived from HSDPA [99], the RSCP threshold is set to -115 dBm ($3.16227766 \cdot 10^{-12}$ mW). So, for any pair (i, m), $i \in N$, $m \in M$, the coverage of mobile m by cell i is defined as

$$cov_{im} = \begin{cases} 1 & if\ RSCP_{im} \geq -115\,dBm \\ 0 & otherwise \end{cases}. \tag{21}$$

If m is covered by more than one cell, we refer to the cell with the highest RSCP as the best server, an we denote it as i^*.

6.3.3 SHO areas

To obtain a realistic outline of the areas where a mobile may potentially maintain connections to more than one cell, we use a static version of the active set [13]. Therefore, we introduce a SHO window, γ^{SHO}, and a maximum active set size, n^{MAX}. Both parameters are taken from the current configuration of the network. The cells to which a mobile $m \in M$ may maintain concurrent connections are part of the set
where $i \in N$, $L^{\downarrow}_{i^*m}$ is the downlink attenuation factor of the cell with the strongest signal (best server), and p_{i^*} is its CPICH power. Since the number of elements in $SHO^{\downarrow}_m$ is at most n^{MAX}, the weakest links are removed if there are more present. This method is well suited for configurations with no hysteresis, since dynamic effects are ignored in static models [13].

Additionally, in the uplink, we define the set of cells to which a mobile can potentially be in SHO as
where $i \in N$, $L^{\uparrow}_{mi}$ is the uplink attenuation factor from mobile m to cell i, and $P^{\uparrow}_m$ is the uplink transmit power of mobile m.

Because of the static nature of the model, we are neglecting mobility and interference by narrowing the SHO window to 2 dB [13].

6.3.4 Optimization objective

Using the elements defined in Section 6.3, 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 6.1.

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 circumstances and it also defines the mutual importance of the different situations taken into account at the optimization phase.

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

where

and

- pf_{COV} represents the penalty factor for uncovered areas,
- $pf^{\uparrow}_{\mathrm{SHO}}$ 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.

6.4 Optimization algorithms

We tackled the problem of balancing SHO areas using three 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 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. 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.

6.4.1 Differential evolution

Differential evolution (DE) [112] is a simple and powerful evolutionary algorithm proposed for global optimization. A wide range of optimization problems have been solved by applying DE [113]. 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:

$$X_{aG} = \{x_1, x_2, \ldots x_i, \ldots, x_D\}\,, \tag{22}$$

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 $D = \mathsf{N}$.

In each generation, DE produces new parameter vectors by adding the weighted difference between two population vectors to a third one [112]. 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 [114].

6.4.2 Differential ant-stigmergy algorithm

Based on the metaheuristic Ant-Colony Optimization (ACO) [76], the differential ant-stigmergy algorithm (DASA) [77] 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 (22):

$$X_a = \{x_1, x_2, \ldots x_i, \ldots, x_D\} \tag{23}$$

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 [77].

6.4.3 Simulated annealing

As the third optimization algorithm to tackle the balancing problem we have chosen simulated annealing (SA) [115], 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 [116].

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

Table 15: 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	$p_{i'} = p_{i'}^{\mathrm{NEW}}$

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, $p_{i'}^{\mathrm{NEW}}$, is checked for validity in step 3, i.e. it must be an element of the set $P_{i'}$. If $p_{i'}^{\mathrm{NEW}}$ 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 $p_{i'}^{\mathrm{NEW}}$.

It is important to note that, as long as P_{i'} > 1, the algorithm shown in Table 15 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.

6.5 Simulations

The simulations are performed using a standard Monte-Carlo method, assuming the mobile users are uniformly distributed. The path-loss data were calculated in advance, using the commercial radio planning tool TEMS^{TM} CellPlanner [111]. 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 [21].

6.5.1 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 16 shows some properties of the test network used.

6.5.2 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 (**??**) are set to the following values:

- $pf_{\mathrm{COV}} = 15$,

Table 16: Test network properties.

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	$398/km^2$

- $pf_{\text{SHO}}^{\uparrow} = 13$, 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 6.1, one of the consequences of such SHO arrangement produces serious interference rise in neighboring cells (Figure 28), 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. Remember 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 [99].

6.5.3 Algorithm parameters

In this section we enumerate the parameters and their values used during the optimization process. In all three cases we have followed the naming conventions as they appear in the original publications [77, 112, 115].

6.5.3.1 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.

6.5.3.2 DASA

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

Table 17: Algorithm performance after 30 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

- $m = 10$, 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.

6.5.3.3 SA

There are only two parameters controlling SA, the initial temperature and the total number of iterations or evaluations:

- $t_{initial} = 125$,
- $it = 100,000$.

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

6.5.4 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.

6.6 Results

6.6.1 Algorithm performance

In this section we examine the performance of the three selected algorithms in terms of 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 17.

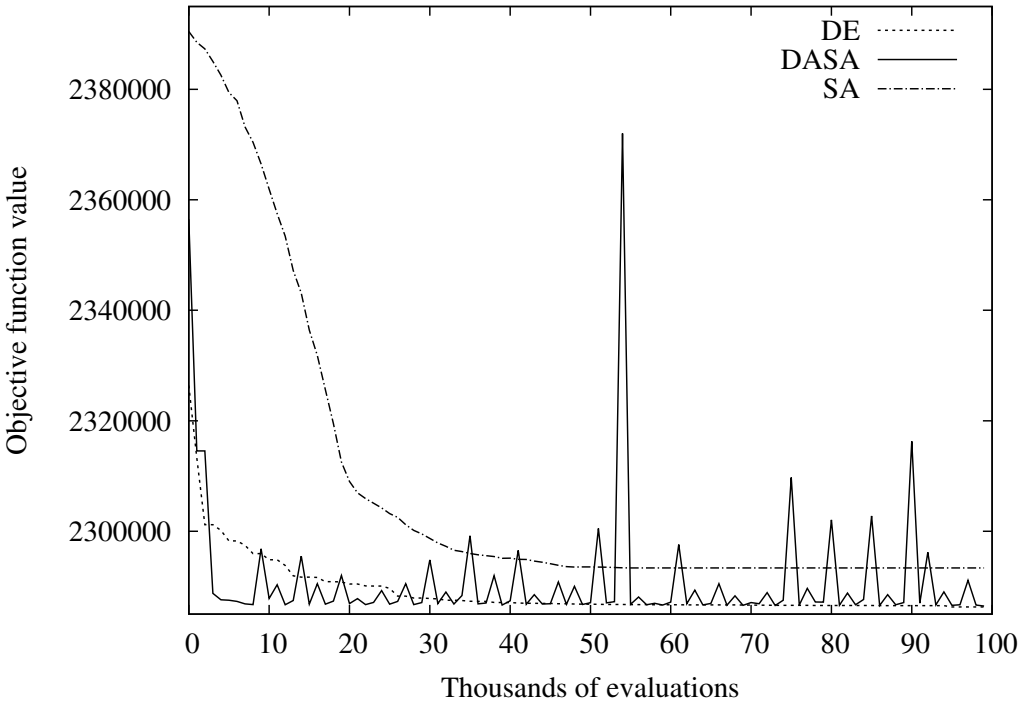

Figure 29: Algorithm convergence for the best obtained results.

As we may observe, 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 similar 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, inducing 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 29. It is worth mentioning that every optimization run starts from a different solution, randomly constructed by picking a CPICH power setting, p_i^k, from every $P_i = \{p_i^1, p_i^2, ..., p_i^K\}$, $1 \leq k \leq \mathsf{P\_\{i\}}$, $\forall i \in N$. 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 [77], hence the jagged profile on the graph. 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 last 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 29 give a clearer notion about the way these algorithms explore the search space of the balancing problem.

Running times of the algorithms are intentionally omitted, since the implementations used are fundamentally different and therefore not comparable.

Table 18: Optimization results.

	Uncovered area	Covered area, no SHO	Normal SHO area	no SHO$^{\downarrow}$, SHO$^{\uparrow}$
Before optimization	63.00 %	15.11 %	15.73 %	1.80 %
DE solution	60.23 %	16.13 %	16.09 %	1.47 %
DASA solution	60.24 %	16.16 %	16.90 %	1.46 %
SA solution	60.42 %	16.55 %	15.97 %	1.56 %
DE improvement	+4.40 %	+6.75 %	+2.29 %	+18.33 %
DASA improvement	+4.38 %	+6.95 %	+7.44 %	+18.88 %
SA improvement	+4.09 %	+9.53 %	+1.52 %	+13.33 %
Avg. improvement	+4.29 %	+7.74 %	+3.75 %	+16.85 %

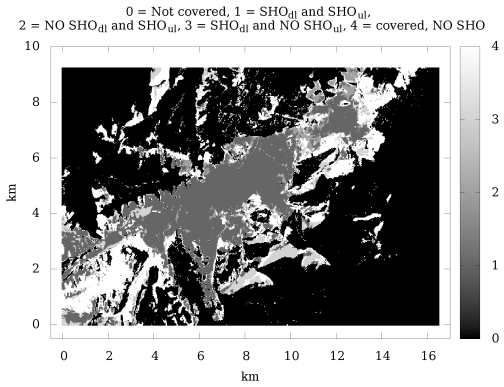

Figure 30: Spatial distribution of SHO areas, before optimization.

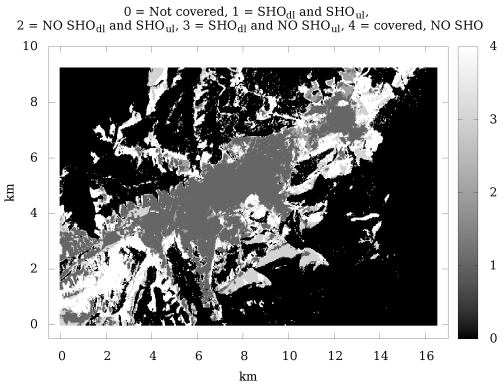

Figure 31: Spatial distribution of SHO areas, after optimization.

6.6.2 Interpretation

Table 18 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 18, where '+' indicates improvement and '-' indicates decline of the given criteria. We may observe 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 change, 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. The SHO areas, where this facility is available in both the downlink and uplink, has 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 we understand this enhancement as a consequence of the completeness of criteria, taken into account in the objective function.

The second most important optimized aspect in the 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 18). 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 18). 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 should 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 these results were obtained for a working mobile network with live 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 30 and 31.

The lack of any prominent visual change in Figures 30 and 31 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.

6.7 Summary

We have presented the problem of balancing SHO areas in UMTS networks and characterized some of the consequences unbalanced SHO areas have on the quality of HSPA services. By using a static network simulator, based on a state-of-the-art mathematical model, we have located downlink and uplink SHO areas. Both the mathematical model and the penalty scores of the objective function have been 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. To the best of our knowledge, there is no reference in the literature to a simulation-based approach to find active SHO areas in the downlink and uplink. Additionally, as far as we know, the SHO balancing problem, as described in this paper, has not yet been tackled by any formal optimization method.

All three algorithms were able to improve the given network configuration, being DE the most successful one. Based on this fact, we may say that DE was the best suited algorithm for solving our problem. The presented results confirm that a great proportion of the non-aligned SHO areas, that were present before the optimization, were corrected, therefore significantly reducing the possibility of HSPA-service failures within these areas. Additionally, network coverage has been 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 put on fine-tuning an 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 our approach is simple and versatile enough for it to be used in practically any working UMTS network. Moreover, our model is applicable for mobile networks in heterogeneous environments, because it imposes no restrictions regarding cell layout or radio propagation characteristics.

To further improve the presented results, dynamic effects, such as fast power control, should be included in the simulations, since it is a valuable element of 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.

7 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 3 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).

8 References

[1] Law, A. *Simulation modeling and analysis*. Boston, MA [etc.]: McGraw-Hill, (2007). 1

[2] Maria, A. Introduction to modeling and simulation. In *Proceedings of the 29th conference on Winter simulation*, 7–13. IEEE Computer Society, (1997). 1

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

[4] 3GPP. General UMTS architecture, v4.0.0. http://www.3gpp.org, (accessed May 2009). 1

[5] Amaldi, E., Capone, A., and Malucelli, F. Radio planning and coverage optimization of 3G cellular networks. *Wireless Networks* **14**(4), 435–447 August (2007). 1

[6] Siomina, I. and Yuan, D. Minimum pilot power for service coverage in WCDMA networks. *Wireless Networks* **14**(3), 393–402 June (2007). 1

[7] Chen, L. and 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

[8] Chen, L. and 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*, 1488–1492. ACM, (2009). 1

[9] Gordejuela-Sánchez, F., López-Pérez, D., and 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*, 1404–1409. ACM, (2009). 1

[10] Siomina, I. and 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

[11] Gorder, P. Multicore processors for science and engineering. *Computing in science & engineering* **9**(2), 3–7 (2007). 2.1, 3

[12] Wen-mei, W. *GPU Computing Gems Emerald Edition: Applications of GPU Computing Series*. Morgan Kaufmann, (2011). 2.1, 3

[13] Nawrocki, M., Aghvami, H., and Dohler, M. *Understanding UMTS radio network modelling, planning and automated optimisation: theory and practice*. John Wiley & Sons, (2006). 2.1, 2.2, 6.1, 6.3, 6.3.3

[14] Amaldi, E., Capone, A., Malucelli, F., and e Informazione, D. Planning UMTS base station location: Optimization models with power control and algorithms. *IEEE Transactions on Wireless Communications* **2**(5), 939–952 (2003). 2.2, 2.3.2

[15] Hao, Q., Soong, B., Gunawan, E., Ong, J., Soh, C., and Li, Z. A low-cost cellular mobile communication system: a hierarchicaloptimization network resource planning approach. *IEEE Journal on Selected Areas in Communications* **15**(7), 1315–1326 (1997). 2.3.1, 2.3.2

[16] 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*, volume 3. 2.3.1, 2.3.2

[17] Mathar, R. and Niessen, T. Optimum positioning of base stations for cellular radio networks. *Wireless Networks* **6**(6), 421–428 (2000). 2.3.1, 2.3.2

[18] Aydin, M., Yang, J., and Zhang, J. *Applications of Evolutinary Computing*, volume 4448 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, (2007). 2.3.2

[19] Amaldi, E., Capone, A., and Malucelli, F. Radio planning and coverage optimization of 3G cellular networks. *Wireless Networks* **14**(4), 435–447 August (2007). 2.3.2, 2.7.1

[20] Raisanen, L. and Whitaker, R. Comparison and evaluation of multiple-objective genetic algorithms for the antenna placement problem. *Mobile Networks and Applications* **10**, 79–88 February (2005). 2.3.2

[21] Holma, H. and 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, 2.5, 6.1, 6.2, 6.3.2, 6.5

[22] Karner, W. *Optimum default base station parameter settings for UMTS networks*. Number September. Citeseer, (2003). 2.4.1

[23] Gerdenitsch, A., Jakl, S., Toeltsch, M., and Neubauer, T. Intelligent algorithms for system capacity optimization of umts fdd networks. *IEE Conference Publications* **2003**(CP494), 222–226 (2003). 2.4.1, 2.6.1

[24] Jakl, S. *Evolutionary Algorithms for UMTS Network Optimization*. PhD thesis, Technische Universitat Wien, (2004). 2.4.1

[25] Siomina, I. and 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). 2.4.1

[26] Wikipedia. High-speed downlink packet access — Wikipedia, the free encyclopedia, (2010). `http://en.wikipedia.org/wiki/Hsdpa`, [Online; accessed 19-April-2010]. 2.4.1

[27] Gordejuela-Sánchez, F., López-Pérez, D., and Zhang, J. 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*, 1404–1409 (ACM, New York, NY, USA, 2009). 2.4.1

[28] UMTSWorld.com. Handover — UMTSWorld.com, (2010). `http://www.umtsworld.com/technology/handover.htm`, [Online; accessed 12-April-2010]. 2.5

[29] 3g Lte Info. UMTS cell selection procedure — 3g Lte Info, (2010). `http://www.3glteinfo.com/umts-cell-selection-procedure`, [Online; accessed 12-April-2010]. 2.5

[30] Flore, D., Brunner, C., Grilli, F., and Vanghi, V. Cell Reselection Parameter Optimization in UMTS. In *Wireless Communication Systems, 2005. 2nd International Symposium on*, 50–53, (2005). 2.5

[31] Chen, L. and 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). 2.5.1

[32] Wikipedia. Soft handover — Wikipedia, the free encyclopedia, (2010). `http://en.wikipedia.org/wiki/Soft_handover`, [Online; accessed 19-April-2010]. 2.5.1

[33] Siomina, I. and Yuan, D. Automated planning of cpich power allocation for enhancing hsdpa performance at cell edges. , 112–118 (2007). 2.5.1

[34] Garcia-Lozano, M., Ruiz, S., and Olmos, J. CPICH power optimisation by means of simulated annealing in an UTRA-FDD environment. *Electronics Letters* **39**, 1676 (2003). 2.5.1

[35] Chen, L. and Yuan, D. Fast algorithm for large-scale umts coverage planning with soft handover consideration. , 1488–1492 (2009). 2.5.1

[36] Siomina, I., Varbrand, P., and Yuan, D. Automated optimization of service coverage and base station antenna configuration in UMTS networks. *IEEE Wireless Communications* **13**(6), 16–25 (2006). 2.6.1

[37] Siomina, I. P-cpich power and antenna tilt optimization in umts networks. , 268–273 (2005). 2.6.1

[38] Gerdenitsch, A., S., J., and Toeltsch, M. The use of genetic algorithms for capacity optimization in UMTS FDD networks. *Proc. 3rd International Conference on Networking (ICN'04)* March (2004). 2.6.1

[39] 3GPP. Requirements for support of radio resource management (FDD), v4.17.0. . `http://www.3gpp.org`, [Online; accessed May-2009]. 2.7

[40] Siomina, I. and Yuan, D. Minimum pilot power for service coverage in WCDMA networks. *Wireless Networks* **14**(3), 393–402 June (2007). 2.7.1, 3, 5.1, 5.2, 5.3.1, 5.4

[41] Valkealahti, K., Hoglund, A., Parkkinen, J., and Hamalainen, A. WCDMA common pilot power control for load and coverage balancing. In *Proceedings of PIMRC*, volume 2. Citeseer. 2.7.1

[42] Valkealahti, K., Hoglund, A., Parkkinen, J., and Flanagan, A. WCDMA Common Pilot Power Control with Cost Function Minimization. *statistics* **1000**(1), 1 (2002). 2.7.1

[43] IST-MOMENTUM. Models and Simulations for Network planning and Control of UMTS. `http://momentum.zib.de`, [Online; accessed 15-April-2010]. 2.8

[44] Gauch Jr, H. *Scientific method in practice*. Cambridge University Press, (2002). 2.8.1

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

[46] Crainic, T., Di Chiara, B., Nonato, M., and Tarricone, L. Tackling electrosmog in completely configured 3G networks by parallel cooperative meta-heuristics. *Wireless Communications, IEEE* **13**(6), 34–41 (2006). 3

[47] Soldani, D., Alford, G., Parodi, F., and 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*, 1–6. IEEE, (2007). 3

[48] Saleh, A., Redana, S., Hämäläinen, J., and Raaf, B. On the coverage extension and capacity enhancement of inband relay deployments in LTE-Advanced networks. *Journal of Electrical and Computer Engineering* **2010**, 4 (2010). 3

[49] Shabbir, N., Sadiq, M., Kashif, H., and Ullah, R. Comparison of Radio Propagation Models for Long Term Evolution (LTE) Network. *arXiv preprint arXiv:1110.1519* (2011). 3, 3.1.1

[50] Valcarce, A., De La Roche, G., Jüttner, Á., López-Pérez, D., and Zhang, J. Applying FDTD to the coverage prediction of WiMAX femtocells. *EURASIP Journal on Wireless Communications and Networking* **2009**, 1–13 (2009). 3, 5.1

[51] Shepler, S., Eisler, M., Robinson, D., Callaghan, B., Thurlow, R., Noveck, D., and Beame, C. Network file system (NFS) version 4 protocol. *Network* (2003). 3.0.1

[52] Hamada, T. and Nitadori, K. 190 TFlops astrophysical N-body simulation on a cluster of GPUs. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–9. IEEE Computer Society, (2010). 3.0.1

[53] Hrovat, A., Ozimek, I., Vilhar, A., Celcer, T., Saje, I., and Javornik, T. Radio coverage calculations of terrestrial wireless networks using an open-source GRASS system. *WSEAS Transactions on Communications* **9**(10), 646–657 (2010). 3.1, 3.1.1, 3.2.1, 3.2.3, 3.4, 4.2, 4.3, 4.5.1, 5.5.1

[54] Sarkar, T., Ji, Z., Kim, K., Medouri, A., and Salazar-Palma, M. A survey of various propagation models for mobile communication. *Antennas and Propagation Magazine, IEEE* **45**(3), 51–82 (2003). 3.1.1

[55] Neskovic, A. and Neskovic, N. Microcell electric field strength prediction model based upon artificial neural networks. *AEU-International Journal of Electronics and Communications* **64**(8), 733–738 (2010). 3.1.1

[56] Neteler, M. and Mitasova, H. *Open source GIS: a GRASS GIS approach*, volume 689. Kluwer Academic Pub, (2002). 3.1.2

[57] Stallman, R. et al. GNU general public license. *Free Software Foundation, Inc., Tech. Rep* (1991). 3.1.2

[58] Butenhof, D. *Programming with POSIX threads*. Addison-Wesley Professional, (1997). 3.2.2

[59] Gropp, W., Lusk, E., and Skjellum, A. *Using MPI: portable parallel programming with the message passing interface*, volume 1. MIT press, (1999). 3.2.2

[60] Akhter, S., Chemin, Y., and Aida, K. Porting a GRASS raster module to distributed computing Examples for MPI and Ninf-G. *OSGeo Journal* **2**(1) (2007). 3.2.3, 3.4

[61] Campos, I., Coterillo, I., Marco, J., Monteoliva, A., and Oldani, C. Modelling of a Watershed: A Distributed Parallel Application in a Grid Framework. *Computing and Informatics* **27**(2), 285–296 (2012). 3.2.3, 3.4

[62] Sorokine, A. Implementation of a parallel high-performance visualization technique in GRASS GIS. *Computers & geosciences* **33**(5), 685–695 (2007). 3.2.3

[63] Blazek, R. and Nardelli, L. The GRASS server. In *Proceedings of the Free/Libre and Open Source Software for Geoinformatics: GIS-GRASS Users Conference*, (2004). 3.2.3.2

[64] Liao, W., Coloma, K., Choudhary, A., Ward, L., Russell, E., and Pundit, N. Scalable design and implementations for MPI parallel overlapping I/O. *IEEE Transactions on Parallel and Distributed Systems* **17**(11), 1264–1276 (2006). 3.2.3.2

[65] Bosque, J. and Pastor, L. A parallel computational model for heterogeneous clusters. *IEEE Transactions on Parallel and Distributed Systems* **17**(12), 1390 (2006). 3.2.3.3

[66] Siegel, S. and Avrunin, G. Verification of halting properties for MPI programs using nonblocking operations. *Recent Advances in Parallel Virtual Machine and Message Passing Interface* , 326–334 (2007). 3.2.3.3

[67] Clemencon, C., Fritscher, J., Meehan, M., and Rühl, R. An implementation of race detection and deterministic replay with MPI. *EURO-PAR'95 Parallel Processing* , 155–166 (1995). 3.2.3.3

[68] Ofcom. Table of base station totals. [Online, accessed October 2012]. Available: http://stakeholders.ofcom.org.uk/sitefinder/table-of-totals/. 3.3.2.1, 3.3.3.1

[69] Cruz Villaroel, F. *Particle flow simulation using a parallel FMM on distributed memory systems and GPU architectures.* PhD thesis, Faculty of Science, University of Bristol, November (2010). 3.3.3.1

[70] Clarke, D., Lastovetsky, A., and Rychkov, V. Dynamic Load Balancing of Parallel Computational Iterative Routines on Highly Heterogeneous HPC Platforms. *Parallel Processing Letters* **21**(02), 195–217 (2011). 3.3.4

[71] Bhandarkar, M., Kale, L., de Sturler, E., and Hoeflinger, J. Adaptive load balancing for MPI programs. *Computational Science-ICCS 2001* , 108–117 (2001). 3.3.4

[72] Watts, J. and Taylor, S. A practical approach to dynamic load balancing. *IEEE Transactions on Parallel and Distributed Systems* **9**(3), 235–248 (1998). 3.3.4

[73] of Pretoria, U. Q-rap: Radio planning tool. [Online, accessed October 2012]. Available: http://www.qrap.org.za/home/. 3.4

[74] Open Source Geospatial Foundation. Quantum GIS. [Online, accessed October 2012]. Available: http://www.qgis.org/. 3.4

[75] Huang, F., Liu, D., Tan, X., Wang, J., Chen, Y., and He, B. Explorations of the implementation of a parallel IDW interpolation algorithm in a Linux cluster-based parallel GIS. *Computers & Geosciences* **37**(4), 426–434 (2011). 3.4

[76] Dorigo, M., Birattari, M., and Stutzle, T. Ant colony optimization. *Computational Intelligence Magazine, IEEE* **1**(4), 28–39 (2006). 4.4.2, 6.4.2

[77] Korošec, P., Šilc, J., and Filipič, B. The differential ant-stigmergy algorithm. *Information Sciences* **192**(1), 82–97 (2012). doi: 10.1016/j.ins.2010.05.002. 4.4.2, 4.4.2, 6.4.2, 6.4.2, 6.5.3, 6.6.1

[78] Cichon, D. and Kurner, T. Propagation prediction models. *COST 231 Final Rep* (1995). 4.6.1, 5.6.1

[79] Holma, H. and Toskala, A. *WCDMA for UMTS: Radio access for third generation mobile communications, Third Edition.* John Wiley & Sons, Inc. New York, NY, USA, (2005). 4.6.1, 4.6.4, 4.7, 5.1, 5.3.1, 5.5.1, 5.6.1, 5.6.4, 5.7

[80] Gerdenitsch, A. *System capacity optimization of UMTS FDD networks.* PhD thesis, Technische Universität Wien, June (2004). 4.7, 5.7

[81] Türke, U. and Koonert, M. Advanced site configuration techniques for automatic UMTS radio network design. *Proc IEEE VTC 2005 Spring* (2005). 4.7, 5.7

[82] Esposito, A., Tarricone, L., Luceri, S., and Zappatore, M. Genetic Optimization for Optimum 3G Network Planning: an Agent-Based Parallel Implementation. *Novel Algorithms and Techniques in Telecommunications and Networking* , 189–194 (2010). 5.1

[83] Nawrocki, M., Aghvami, H., and Dohler, M. *Understanding UMTS radio network modelling, planning and automated optimisation: theory and practice*. John Wiley & Sons, (2006). 5.1, 5.3

[84] Cunningham, B., Alexander, P., and Candeub, A. Network growth: Theory and evidence from the mobile telephone industry. *Information Economics and Policy* **22**(1), 91–102 (2010). 5.1

[85] Siomina, I. and Yuan, D. Pilot power optimization in WCDMA networks. **4** March (2004). 5.1, 5.4

[86] Cheung, G., Tan, W., and Yoshimura, T. Real-time video transport optimization using streaming agent over 3G wireless networks. *IEEE transactions on multimedia* **7**(4), 777 (2005). 5.1

[87] Vasile, M. and Locatelli, M. A hybrid multiagent approach for global trajectory optimization. *Journal of Global Optimization* **44**(4), 461–479 (2009). 5.1

[88] Benedičič, L., Štular, M., and Korošec, P. Pilot power optimization in UMTS: a multi-agent approach. In *Proceedings of the 13th International Multiconference Information Society - IS 2010*, volume A. Jožef Stefan Institute, October (2010). 5.1, 5.2

[89] Chen, L. and 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). 5.1, 5.3.1

[90] Värbrand, P. and Yuan, D. A mathematical programming approach for pilot power optimization in WCDMA networks. (2003). 5.3.2

[91] Sarkar, P. A brief history of cellular automata. *ACM Computing Surveys (CSUR)* **32**(1), 107 (2000). 5.4

[92] Talbi, E. *Metaheuristics: from design to implementation*. Wiley, (2009). 5.4

[93] Stone, J., Gohara, D., and Shi, G. OpenCL: A parallel programming standard for heterogeneous computing systems. *Computing in Science and Engineering* **12**, 66–73 (2010). 5.5

[94] Munshi, A. et al. The OpenCL specification version 1.0. *Khronos OpenCL Working Group* (2009). 5.5

[95] Nvidia, C. Compute Unified Device Architecture Programming Guide. *NVIDIA: Santa Clara, CA* **83**, 129 (2007). 5.5

[96] Kloeckner, A. GPU from Python with PyOpenCL and PyCUDA. `http://www.bu.edu/pasi/materials/`, [Online; accessed 5-April-2010]. 10, 7

[97] Marsaglia, G. Seeds for random number generators. *Communications of the ACM* **46**(5), 90–93 (2003). 5.5.2

[98] Third Generation Partnership Project. *Technical specification group services and systems aspects: Network architecture v3.6.0 (Release 1999).* http://www.3gpp.org/. 6.1

[99] Holma, H. and Toskala, A. *HSDPA/HSUPA For UMTS*. John Wiley & Sons, (2006). 6.1, 6.1, 6.3.2, 6.5.2

[100] Chen, L., Sandrasegaran, K., Basukala, R., Madani, F., and Lin, C. Impact of soft handover and pilot pollution on video telephony in a commercial network. In *Communications (APCC), 2010 16th Asia-Pacific Conference on*, 481–486. IEEE, (2010). 6.2

[101] Chen, L. and Yuan, D. Coverage planning for optimizing HSDPA performance and controlling R99 soft handover. *Telecommunication Systems* , 1–12 (2011). 6.2

[102] Eisenblätter, A., Füngenshuch, A., Geerdes, H. F., Junglas, D., Koch, T., and Martin, A. Optimization methods for UMTS radio network planning. In *Intl. Conference on Operations Research*, 31–38. Springer, September (2003). 6.2

[103] Ghosh, S., Whitaker, R., Allen, S., and Hurley, S. Optimising CDMA Cell Planning with Soft Handover. *Wireless Personal Communications* , 1–27 (2011). 6.2

[104] Chen, L. and 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). 6.2

[105] Laiho, J., Wacker, A., and Novosad, T. *Radio Network Planning and Optimisation for UMTS*. John Wiley & Sons, Inc. New York, NY, USA, (2002). 6.2

[106] 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*, 276–282. ACM New York, NY, USA, (2004). 6.2

[107] Ying, S., Gunnarsson, F., Hiltunen, K., Res, E., and Beijing, C. CPICH power settings in irregular WCDMA macro cellular networks. *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003* **2** (2003). 6.2

[108] Garcia-Lozano, M., Ruiz, S., and Olmos, J. CPICH power optimisation by means of simulated annealing in an UTRA-FDD environment. *Electronics Letters* **39**(23), 1676–7 (2003). 6.2

[109] Lempiäinen, J. and Manninen, M. *UMTS Radio Network Planning, Optimization and QoS Management for Practical Engineering Tasks*. Kluwer Academic Publishers Norwell, MA, USA, (2004). 6.2

[110] Siomina, I. and Yuan, D. Minimum pilot power for service coverage in WCDMA networks. *Wireless Networks* **14**(3), 393–402 (2008). 6.2

[111] TEMS CellPlanner. http://www.ericsson.com/tems/. 6.3.1, 6.5

[112] Storn, R. and Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization* **11**(4), 341–359 (1997). 6.4.1, 6.4.1, 6.5.3

[113] Das, S. and Suganthan, P. Differential evolution: A survey of the state-of-the-art. *Evolutionary Computation, IEEE Transactions on* (99), 1–28 (2010). 6.4.1

[114] Price, K., Storn, R., and Lampinen, J. *Differential evolution: a practical approach to global optimization*. Springer-Verlag New York Inc., (2005). 6.4.1

[115] Kirkpatrick, S., Gelatt, C., and Vecchi, M. Optimization by simulated annealing. *Science* **220**(4598), 671 (1983). 6.4.3, 6.5.3

[116] Suman, B. and Kumar, P. A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the operational research society* **57**(10), 1143–1160 (2005). 6.4.3

List of Figures

List of Tables

Index of Algorithms