ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Master Thesis

# Cellular Operators in a Shared Spectrum

Şivan ALTINAKAR

- Mathematics Section, 5th grade -

Supervisors :

Prof. Dominique DE WERRA
Tınaz EKIM-AŞICI
Márk FÉLEGYHÁZI

winter semester 2005-2006

Prof. D. de Werra

Chaire de Recherche Opérationelle Sud-Est
EPFL, FSB - IMA - ROSE
1015 Lausanne

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Hiver 05/06

# Travail pratique de Master

| | |
|---|---|
| **Sujet:** | OPÉRATEURS MOBILES DANS UN SPECTRE PARTAGÉ |
| **Candidat:** | Şivan Altınakar |
| **Responsables:** | Mark Felegyhazi (LCA) and Tınaz Ekim (ROSE) |

## Introduction

De nos jours, les bandes du spectre d'un réseau sans fil sont affectées pour l'usage exclusif d'un opérateur particulier et pour un objectif précis. Souvent, les opérateurs n'exploitent pas d'une façon efficace la bande de fréquence affectée pour eux tandis que d'autres fréquences sont à peine utilisées. Maintenant cette approche monopolistique est souvent critiquée car amène à une utilisation non optimale du spectre. Pour cette raison, on aimerait considérer le cas pour lequel certains opérateurs doivent partager la même bande de fréquence. Cette situation pourrait alors être représentée comme un jeu dans lequel chaque opérateur cherche à maximiser une certaine fonction d'utilité (comme par exemple l'espace de couverture). En termes de théorie des graphes, les sommets pourraient représenter les stations de base et les arêtes une certaine mesure de connectivité qui refléterait l'interférence.

## But du projet

Le but de ce projet est de modéliser le problème sous forme mathématique et en particulier sous forme de graphes et de chercher à résoudre le jeu en utilisant les notions d'équilibre de Nash et d'optimum de Pareto.
Le travail commencera par une recherche de littérature qui déterminera les directions que le candidat pourra développer ainsi que les voies non encore explorées dans le domaine. Par la suite, un travail de modélisation mettra en évidence les différents problèmes de théorie des graphes que le candidat étudiera en détail. A côté des résultats théoriques montrant nos limites et nos possibilités d'apporter une solution au problème formulé, il est important de pouvoir fournir en un temps raisonnable des solutions suffisament bonnes pour être utilisées en pratique. C'est pour cela que l'étudiant proposera des heuristiques pour une résolution approchée.
L'utilisation de la théorie des jeux paraissant prometteuse pour le problème en question, les notions telles que l'équilibre de Nash et les optima de Pareto seront discutées et comparées avec la modélisation par les graphes. Un jeu pourra éventuellement être simulé pour des différentes stratégies des deux joueurs afin de discuter les résultats obtenus.
Les points principaux du projet sont les suivants:

- Modéliser le problème en termes de théorie des graphes.
- Discuter la complexité algorithmique du problème.
- Résoudre le problème à l'aide des algorithmes heuristiques développées.
- Modéliser le problème en termes de théorie des jeux et discuter les différentes notions qui y sont liées.
- Simuler un jeu en fixant une stratégie pour chaque joueur et commenter les résultats obtenus.

## Rapport et présentation orale

Le candidat suivra les indications du professeur et des collaborateurs responsables et les mettra au courant de l'avancement du projet **au moins une fois par semaine**. Une présentation intermédiaire du travail sera fixée ultérieurement.

Chaque phase du projet sera détaillée dans un rapport à remettre en 4 exemplaires le **vendredi 24 février 2006** à midi au plus tard. Le rapport contiendra les points suivants:

1. la présente donnée du sujet;
2. une introduction didactique et motivée du travail;
3. une explication détaillée des résultats mis en évidence ainsi que leur intérêt.
4. les performances obtenues par les méthodes développées avec interprétation des résultats et comparaison avec d'autres méthodes le cas échéant.
5. des suggestions pour une extension et un approfondissement du sujet.
6. une bibliographie (avec des références précises).
7. un CD ou une disquette contenant la version électronique du rapport, les sources LaTeX, ainsi que les codes sources des programmes développés.

## Références

- Magnus H. Halldorson, Joseph H. Halpern, Li (Erran) Li, Vahab S. Mirrokni: "On Spectrum Sharing Games", Proc. ACM Symposium on Principle of Distributed Computing (PODC), pages 107-114, St.John's, Newfoundland, Canada, July 2004.
- Mark Felegyhazi, Jeaan-Pierre Hubaux: "Wireless Operators in a Shared Spectrum", EPFL IC Technical Report (IC/2005/040)

**Abstract**

Due to the increasing number of radio technologies, the available frequency spectrum becomes more and more utilized, hence its clever use becomes a critical issue. Among many proposed solutions, the formulation of the problem as the control of the power of the base stations, also known as the *Power Control* problem, seems a promising idea.

In the present work, we propose to study this problem by first defining a theoretical model. Then, we design a family of non-cooperative games that hopefully stop at a Nash equilibria close to the optimal solution for the network, as well as a few simple tabu search heuristics. We finally developed a java library and a program, in order to experimentally study the behavior of the proposed games and heuristics.

# Acknowledgment

First of all, I would like to thank Professor Dominique de Werra for giving me the opportunity to study under his supervision and whose courses and most enjoyable company have made me realize that my true passion of mathematics and my future path lied in Operations Research.

My special thanks go to Tınaz Ekim-Aşici and Márk Félegyházi, my trustworthy assistants, who have always supported me throughout the course of this work. Their unquenchable enthusiasm encouraged me to put my very best into it. Moreover, Márk and his professor Jean-Pierre Hubaux are the ones who originally proposed this great project to ROSE.

I am also particularly grateful to Daniela, Jocelyne, Bernard and Benjamin, who all belong to ROSE and who took part in this very joyful (but still hard-working) ambiance that I'm sure I will encounter nowhere else. Let us not forget Ivo who, among so many other things, taught me how to put LaTeX in PowerPoint presentations and who was my office colleague for an unfortunately too short period of time.

# Contents

# List of Figures

# 1 Introduction

Due to the increasing number of radio technologies, the available frequency spectrum becomes more and more utilized, and therefore we need to find new and original solutions to satisfy everybody's need, using only a limited range of available frequencies. For instance, today's cellular operators have each a different and non-overlapping frequency band and do their best to use it in an efficient way. However, although widely used, this affectation globally leads to a non-negligible amount of frequency ranges being wasted.

In the present work, we propose to solve this problem by allowing all operators to use the same frequency range. Since this approach leads to interferences between emitting base stations, it is necessary to find a way to limit the interferences by only controlling each existing base station's emission power, while keeping a good coverage of the network. In our paper, we will refer to this problem as the *power control* problem. For this purpose, we first define a model. Then, we propose games where the base stations are non-cooperative players, as well as simple tabu search heuristics that could be used by an organizing entity having the authority to set each base station's power. We finally study the behavior and efficiency of these games and heuristics through a set of simulations.

Here is a brief summary of the topics covered in the present work:

**Section 2:** Previous works in closely related fields are presented.

**Section 3:** Fundamental components of our model, namely the *network* and the *interference*, are defined.

**Section 4:** The first modeling attempt, using edge-deletion, is described.

**Section 5:** The second modeling attempt, using node-deletion, is described.

**Section 6:** The final model, used throughout the rest of this work, is presented and discussed.

**Section 7:** Our game theoretical approach to this problem is explained.

**Section 8:** A global description of the developed program is given, as well as informations on some of its key features.

**Section 9:** The set of the performed simulation is described

**Section 10:** The results of the simulations are discussed

**Section 11:** Proposals for further research are developed.

**Section 12:** A broader and final look at the present work is proposed.

**Appendices:** Extra information on the programs developed during the course of this project is given.

# 2 State of the Art

Recent technological developments have opened up new telecommunication possibilities. This has crowded the available range of frequencies allocated to operators. Saturation of available frequencies lead to new research for a better use of radio frequency. As the field is very innovative and telecommunication technologies are constantly improving, many papers exist on various new ways to tackle the numerous problems encountered.

This present work studies the Power Control problem in a brand new way. But this problem has already been approached from many different angles, for instance in [1] where the point of interest is to determine where in the architecture the Power Control problem is to be situated, to determine the appropriate power level by studying its impact on several performance issues and finally to provide a software architecture for realizing the solution.

On the other hand, [2] proposes an optimization of the pilot power through a mathematical programming approach of this problem, in particular when subject to full coverage. It achieves to show that optimized power levels yield substantial savings in total power consumption when compared to using a uniform pilot power.

In the context of a different problem, [3] also studies the assignment of users to base stations and the power at which they need to emit in order to connect to their respective base stations. A survey of different network design problems and methods is provided in [4].

Furthermore, some researchers are interested in applying problems from graph theory on the field of wireless communications system, such as the Maximum Independent Set problem [5] and the Facility Location problem [6]. This is also the case in [7] where the Power Control problem is tackled by formalizing it as a Minimum Membership Set Cover problem. That article proves that the optimal solution of their problem cannot be approximated in polynomial time closer than with a factor $\ln n$. It also presents an algorithm exploiting linear programming relaxation technique which asymptotically matches this lower bound.

It is also of interest to cite [8], which gives us insights on the complexity of some of the models that we develop. That article shows that for properties that are hereditary on induced subgraphs and several of their restrictions, the node-deletion problem is NP-complete. However, edge-deletion problems seem to be less amenable to such generalizations, although for some common properties (planar, outer-planar, line-graph, transitive digraph) it has been proved to be NP-complete.

Finally, some researchers use game theory to study the behavior of players in the network (usually the service providers), as in [10] or in [9]. In the latter one, the service providers decide the power of the pilot signal of their base stations. That article first identifies possible Nash equilibria in a theoretical setting in which all base stations are located on the vertices of a two-dimensional lattice. It also shows that when this topological assumption is relaxed, finding the Nash equilibria is an NP-complete problem. Finally, it proves that a socially optimal Nash equilibrium exists and that it can be enforced by using punishments.

Another problem, the Channel Assignment problem of access points in a WiFi network, is viewed in [12] as a game. That article provides bounds on the price of anarchy depending on assumptions on the underlying network and the type of bargaining allowed between service providers. The key tool in the analysis is the identification of the Nash equilibria with the solutions to a maximal coloring problem in an appropriate graph.

Finally, [11] surveys the recent literature on game theoretic analysis of ad hoc networks, highlighting its applicability to power control, medium access control, routing and node participation, among other subjects.

# 3 Definition of the Power Control Model's Fundamental Components

In order to construct a good model, we need to identify clearly the building blocks with which we are going to work.

## 3.1 Network and Base Station

We make the following assumptions with respect to the communication system. We first assume that there exists a cellular wireless system (the *network*) with a few *operators*. Each operator controls a set of *base stations*. There also exists a set of *users*, each controlling a wireless device. These users want to use the wireless communication service provided by the base stations of the operators, and are able to attach to any base station of any operator. The radios of the users and the base stations are compatible. Furthermore, the users have the possibility to freely roam across the base stations, independently of the operators.

For example in Lausanne's cellular network, the base stations would be the antennae laid by the three usual operators (*Swisscom*, *Orange* and *Sunrise*), and a user would be anyone using a mobile phone within that area.

Every base station has a finite service area around it, within which its *signal* can be received and used. Outside of this area, the signal is so weakened that it just blends in the ambient *noise*. In this project, this area is modeled as a circle centered on a base station and having a fixed radius named *threshold distance* $d_{thresh}$ .

Figure 1 presents an example of a network[1], where each color represents a different operator. Moreover, the edges represent the potential interactions between *neighbor* base stations, i.e. base stations that are closer than the threshold distance and hence can alter each other's transmission quality, depending on their respective emission power.

## 3.2 Interference

We also need to define a way to measure the quality of a transmission. The *Signal to Interference plus Noise Ratio (SINR)* is widely accepted as the standard measure of the quality of the signal originating from base station $A$ and received by user $u$, which is being disturbed by other base stations $B_i$ (see [9]). Note that $u$ is in the range of $A$ and all the $B_i$. In the following

[1] This image is a screenshot of a network randomly generated with our program and displayed in a window.

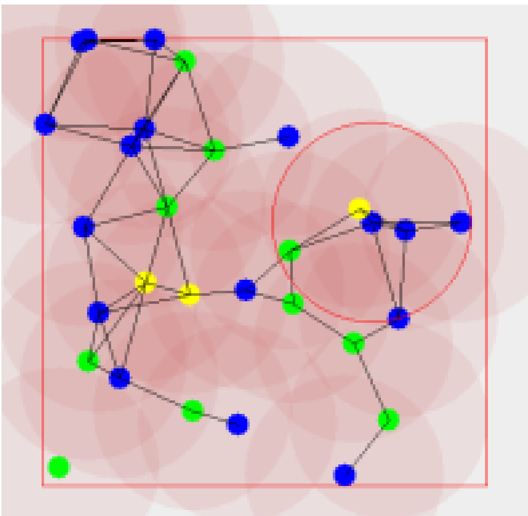

Figure 1: Example of a network with three operators.

formula, $P_A$ is the emission power of a base station $A$, $d_{u,A}$ is the distance between the user and base station $A$. The *Gaussian thermal noise* constant $N_0$ is almost negligible. The *path loss* constant $\alpha$ is between 2 and 5. In fact, both constants depend of the radio signal propagation property of the environment.

$$\mathrm{SINR}_u(A) := \frac{P_A \cdot d_{u,A}^{-\alpha}}{N_0 + \sum_i P_{B_i} \cdot d_{u,B_i}^{-\alpha}}$$

In this present work, unlike in [9, 10], we do not model the users. Hence, it is necessary to define a new way to measure the effect of the signal of one base station on the transmission of another, without using the quality of the signal received by an explicit user.

We propose to introduce the notion of *individual interference of base station $B$ over base station $A$* (where $A$ and $B$ are neighbors), defined as follows:

$$\mathrm{I}(P_B, P_A, d_{B,A}) := \frac{P_B \cdot d_{B,A}^{-\alpha}}{P_A \cdot d_{\mathrm{ref}}^{-\alpha}}$$

This expression can be interpreted as the inverse of the SIR [2] of an *imaginary user* at a very short theoretical *reference distance* ($d_{\mathrm{ref}}$) from the base station $A$ from which it is trying to receive a signal, while being disturbed by the emission of $B$. In a cellular network for instance, the distance from

[2]The SIR is the SINR with $N_0 = 0$.

the *imaginary user* $u$ to $A$ ($= d_{u,A} = d_{\text{ref}}$) would be 1 meter, compared to the distance to $B$ ($= d_{u,B}$) which would be in hundreds of meters. Therefore, here we approximate $d_{u,B} = d_{B,A}$ (see figure 2).

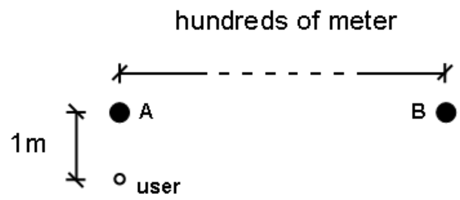


Figure 2: Individual interference of base station $B$ over base station $A$, with *imaginary user* $u$.

The *interference* thus defined has also the following expected behavior: it increases when either $P_B$ increases, $P_A$ decreases or when the distance $d_{B,A}$ increases.

In addition, for a given allocation of power $s$ to each base station, we define the *Interference over* $A$, written $\mathrm{I}_s(A)$, which is the sum of all the individual interferences over $A$ from the neighbor base stations $B_i$, plus a noise factor.

$$\mathrm{I}_s(A) := \sum_i \mathrm{I}(P_{B_i}, P_A, d_{B_i,A}) + \frac{N_0}{P_A \cdot d_{\text{ref}}^{-\alpha}}$$

We also get the following property for an *imaginary user* $u$ at a reference distance $d_{\text{ref}}$ of $A$:

$$\begin{aligned}
\mathrm{I}_s(A) &= \sum_i \mathrm{I}(P_{B_i}, P_A, d_{B_i,A}) + \frac{N_0}{P_A \cdot d_{\text{ref}}^{-\alpha}} \\
&= \frac{\sum_i P_{B_i} \cdot d_{B_i,A}^{-\alpha} + N_0}{P_A \cdot d_{\text{ref}}^{-\alpha}} \\
&\simeq \frac{\sum_i P_{B_i} \cdot d_{u,B_i}^{-\alpha} + N_0}{P_A \cdot d_{u,A}^{-\alpha}} \\
&= \frac{1}{\mathrm{SINR}_u(A)}
\end{aligned}$$

One can now clearly see the relation that is being established with the SINR. The advantage of the *Interference* over the *SINR* is that it can be decomposed into contributions from all interfering base stations.

For example, in a graph representation of this network with directed edges between neighbor base stations, the edge $(A, B)$ could have a weight of $\mathrm{I}(P_B, P_A, d_{B,A})$. Suppose also that an auxiliary node $X$ is added, with an outgoing edge toward each base station $A$ of the network with a weight $N/P_A$ and no incoming edges (see figure 3). You finally obtain a graph where the sum of the incoming edges to a node $A$ is equal to $I_s(A)$. And if you add the weight of all edges in the network, you get the sum of all the interferences in the network. You can relate this representation to the *final model* described later in section 6.

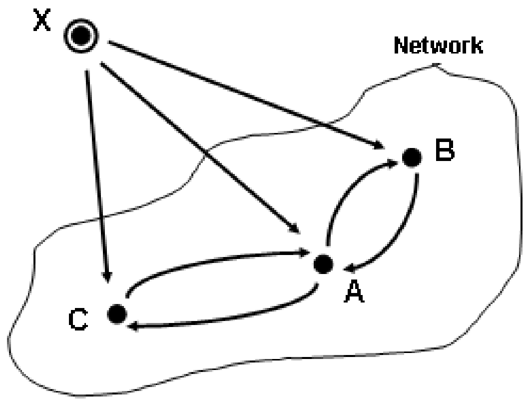


Figure 3: Directed weighted graph, with edges representing the non-null interference between base stations and an auxiliary node $X$ for the noise factor.

## 3.3 Power Control and Setting

In our case, the base stations are already positioned and we are consequently not interested in the *facility location problem* (although we also propose later on to study the effect of the placement in space of the base stations). Our optimization of the global quality of the transmissions in a given network will only be through the *control of the emitting power* of each base station.

In order to simplify our task, we define a finite *set of power levels* $\mathcal{P}$ that can be chosen by each base station. Moreover, to compensate partially the signal decay[3] that is in $1/d^{\alpha}$, we define an exponential sequence of $r$ power

[3] Recall that $\alpha$ is between 2 and 5

levels $\mathcal{P} = \{P_1, \ldots, P_r\}$, ending with a given *maximum power level* $P_{\max}$:

$$
\begin{aligned}
P_1 &= \frac{P_{\max}}{2^{(r-1)}} \\
&\vdots \\
P_i &= \frac{P_{\max}}{2^{(r-i)}} \\
&\vdots \\
P_r &= P_{\max}
\end{aligned}
$$

We finally introduce the term *setting*, which corresponds to allocating an emission power from the set $\mathcal{P}$ to every base station.

# 4 Early Models: First Attempt (based on Edge-Deletion)

In this section and the following one, we present two models that we designed in the early stages of our research. They propose a formulation of our problem expressed as either an edge-deletion or a node-deletion problem. But as we were not fully satisfied with both designed models, we eventually decided to try a different approach (see the *final model* in section 6). For this reason, we did not study any of the theoretical properties of these early models, but preferred to move on to something that looked more promising.

## 4.1 Definition of the Edge- and Node-Deletion Problem

The *node-deletion* (resp. *edge-deletion*) problem can be stated as follows:

**Definition 1 (node-deletion problem, edge-deletion problem)**
*Given a graph $G$ (directed or not), find a set of nodes (resp. edges) of minimum cardinality, whose deletion results in a subgraph satisfying a given property $\pi$.*

A property that is satisfied by a single node, but not by any possible graph, is said to be *non-trivial*. Also, if there exists arbitrarily large graphs satisfying that property, it is called *interesting*.

An other kind of property worth of studying are the ones that are *hereditary on induced subgraphs*, i.e. for a graph satisfying property $\pi$, the deletion of any node does not result in a graph violating $\pi$. A detailed study of the complexity of such properties is presented in [8].

## 4.2 Mutual Disturbance

Our very first model uses a virtual *mutual disturbance function* $m(P_A, P_B, d_{A,B})$.

**Definition 2 (mutual disturbance function, value)**
*We define the* mutual disturbance value *between to base stations as the result of a* function $m$

$$\begin{array}{rrcl} m: & (0, P_{max}] \times (0, P_{max}] \times \mathbb{R}_+^* & \longrightarrow & \mathbb{R}_+^* \\ & (P_A, P_B, d_{A,B}) & \longmapsto & m(P_A, P_B, d_{A,B}) \end{array}$$

*satisfying:*

*1)* $m(P_A, P_B, d) = m(P_B, P_A, d)$

*2)* $m(P_A, P_B, d) \leq m(P_A^{'}, P_B, d) \qquad \textit{if} \quad P_A \leq P_A^{'}$

*3)* $m(P_A, P_B, d) \geq m(P_A, P_B, d^{\,'}) \qquad \textit{if} \quad d \leq d^{\,'}$

*4)* $\textit{if} \quad m(P_A, P_B, d) \leq m(P_A, P_C, d) \qquad \textit{then}$
$\quad m(P_A^{'}, P_B, d) \leq m(P_A^{'}, P_C, d) \qquad \forall P_A^{'} \in (0, P_{max}]$

This means that by lowering one or both emission powers of a pair of base stations, their mutual disturbance value would eventually fall below a given threshold $\delta$.

## 4.3 Representation

In this model, we fix a $\delta \in \mathbb{R}_+^*$ and we initially have an undirected graph $G_E(V_E, E_E)$ where the nodes in $V_E$ represent the base stations, and there is an edge in $E_E$ between two nodes if and only if the mutual disturbance value of the corresponding pair of base stations with power $P_{max}$ for both of them is higher than $\delta$.

## 4.4 Objective

We first start with a solution where all the base stations are set to the maximum power $P_{max}$. Our objective is to have a graph with every node's degree being at most $k$, $k$ being a integer that we have chosen. This can be achieved by iteratively choosing lower power settings (not necessarily in the set $\mathcal{P}$, which is not used in this model) for the base stations, resulting in a new graph with mutual disturbance values between pairs of base stations lower than previously, some being now below the threshold $\delta$ and thus not requiring to be represented by an edge between the corresponding base stations.

This means that we try to limit the number of very high disturbance in the network that each base station takes part in.

## 4.5 Edge-Deletion Formulation

One problem we face when we try to express our problem as an edge-deletion problem is that we cannot delete any edge at any time. In fact, for the obtained solution to make sense, we can delete an edge $(A, B)$ if and only if

$$B = \underset{C \in \mathcal{N}(A)}{\text{argmin}}\ m(P_A, P_C, d_{A,C}) \tag{1}$$

or

$$A = \underset{C \in \mathcal{N}(B)}{\text{argmin}}\ m(P_B, P_C, d_{B,C}) \tag{2}$$

where $\mathcal{N}(A)$ represents the set of all the nodes connected to $A$ by an edge.

If (1) is true, then we lower base station $A$'s power until its mutual disturbance value with $B$ is lower than $\delta$, but its mutual disturbance value with the other base stations in $\mathcal{N}(A)$ is still higher than $\delta$ (this is possible because of (1) and the fourth condition in the definition of the *mutual disturbance function*). Conversely, if (2) is true, we should lower $B$'s power in the same way. Finally, if both (1) and (2) are true, we can choose which base station's power we decide to lower.

For this reason, we cannot call our problem a true edge-deletion problem, because we need to generate at every step a subset of edges that we are allowed to delete.

## 4.6 Other Drawback of this Model

The idea of an undirected *mutual disturbance* between two base stations, although conceptually pleasant, is hard to interpret or justify in the case of a cellular wireless system. Consequently, we decided to abandon this model (without doing any analysis of its complexity) and preferred to search for a more realistic and satisfying one.

# 5 Early Models: Second Attempt (based on Node-Deletion)

In this section, we present the second model that we designed, this time using node-deletion (defined in section 4.1) and the notions developed in section 3.

## 5.1 Representation

In this model, we fix a $\beta \in \mathbb{R}_+^*$ and we use the set of power levels $\mathcal{P} = \{P_1, \ldots, P_r\}$ defined in section 3.3. We have a directed graph $G_N(V_N, E_N)$, where for each base station $A$, we have $r$ *power nodes* $A_1$ through $A_r$, each of them representing the corresponding power level $P_1$ through $P_r$. There are two kinds of edges between power nodes of $V_N$ (see also figure 4):

1) Between power nodes associated with different base stations, if the distance $d$ between base station $A$ and $B$ is smaller than the threshold distance, then there exists a directed edge $(A_i, B_j)$ if and only if $\mathrm{I}(P_i, P_j, d) > \beta$.

2) Between power nodes associated with the same base station $A$, for every pair of power nodes $A_i$ and $A_j$, we put $n$ directed edges $(A_i, A_j)$ and $n$ directed edges $(A_j, A_i)$, where $n$ is the number of base stations in our network.

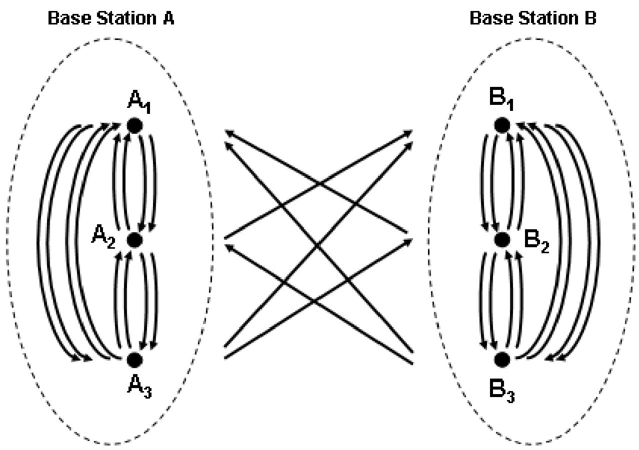


Figure 4: Network with two base stations ($n = 2$) and three power levels ($r = 3$).

## 5.2 Objective and Node-Deletion Formulation

We first choose an integer $k$ between 0 and $n-1$. Our objective is to have an induced subgraph with every power node's number of incoming edges being at most $k$. As $k < n$, such a solution would naturally have at most one power node $A_i$ per base station, because of the second type of edges. We consider the base stations that have no power node left in the solution's subgraph to be inactive, i.e. not used in the obtained network.

This means that we try to limit, for each base station, the number of base stations interfering over it with an interference value above $\beta$.

In this case, we clearly see that this problem can be formulated as finding the minimum number of power nodes to delete in order to have the subgraph satisfying our condition.

## 5.3 Other Bounds Considered

We also considered other bounds for each power nodes, such as:

- The number of *outgoing* edges (to power nodes associated with different base stations) must be at most $k$.
- The number of *incoming* edges must be at most $k_-$ and the number of *outgoing* edges must be at most $k_+$.
- the sum of the number of *incoming* and *outgoing* edges must be at most $k$. In this case, we need to put $2n$ edges in each direction between power nodes associated with the same base station.

## 5.4 Complexity

The node- and edge-deletion problem have been thoroughly studied in [8], depending on particular aspects of the property $\pi$. We present here one important result:

**Theorem 1 (theorem 1, page 254 in [8])**
*The* node-deletion *problem for a non-trivial, interesting property $\pi$ that is hereditary on induced subgraphs is NP-complete.*

As the property "with maximum incoming degree $k$" (or the three other bounds on degrees proposed in section 5.3) is clearly non-trivial, interesting and hereditary on induced subgraphs, this theorem holds for our *node-deletion* problem.

## 5.5 Main Drawback of this Model

Although we managed to express this problem truly as a node-deletion problem, it also has a problem with the way we use to describe how the base stations disturb one another's transmission.

For instance, suppose that a base station $A$ has many neighboring base stations with their interference a little lower than the threshold. In this case, $A$ has no incoming edges although there is a huge interference over it. Conversely, a base station $B$ with a single neighbor which has an interference over $B$ that is slightly higher than the threshold, has an incoming edge from his neighbor and appears to be in a worse position than $A$, although this is not the case in reality. Thus, fixing a threshold $\beta$ is not always useful.

Therefore, in the following and final model, we decided not to use anymore a threshold, but preferred directly expressing the interference of one base station over an other one by a weighted directed edge. Unfortunately, with such a model we could not find anymore a formulation based on either node- or edge-deletion.

# 6 Final Model

After several attempts, we finally decided to forget the node- or edge-deletion formulation, because they seemed to necessitate the use of some kind of threshold in the relation *between pairs of base stations*, which usually gave us an unrealistic model. We therefore present this final model which we think is much more applicable and close to a real-life network situation than the two previous ones.

## 6.1 Preliminary Definitions

In order to express the *final model* in a precise mathematical way, we first need to define a few symbols:

- $\mathcal{P} = \{P_1, \ldots P_r\}$ is the set of available *power levels*,
- $\mathcal{R} = \{1, \ldots r\}$ is the set of exponents used to define power levels
- $\mathcal{V} = \{A, B, \ldots\}$ is the set of all base stations in the network,
- $x_{A,i} \in \{0, 1\}$ , which represents base station $A$ with a power setting of $P_i$ when $x_{A,i} = 1$,
- $w_{x_{A,i}x_{B,j}} = \begin{cases} \mathrm{I}(P_i, P_j, d_{A,B}) & \text{if } A \text{ and } B \text{ are } \textit{neighbors}, \\ 0 & \text{else.} \end{cases}$
- $n_i = N_0/P_i$

Note that a base station is not considered to be its own neighbor.

With these notations, we have a *setting* $s$ for our network when for each base station $A$, exactly one $x_{A,i}$ equals 1 and the rest equals 0. When we have such a setting, the *Interference over A* can be expressed by the following formula:

$$\mathrm{I}_s(A) = \sum_{B\ \in\mathcal{V}} \sum_{i,j\ \in\mathcal{R}} w_{x_{B,i}x_{A,j}}\ x_{B,i}\ x_{A,j} \quad + \sum_{j\ \in\mathcal{R}} n_j\ x_{A,j}$$

## 6.2 Formulation

We now propose two different objective functions for our problem. The first one tries to minimize the sum of the Interference over each base station. It also makes the problem a *quadratic binary program.* On the other hand, the second one optimizes an equally justifiable idea: maximizing the sum of the inverse of the Interference over each base station, which can be related to the signal quality (SINR) of a user nearby that station and attached to it.

The second objective function was formulated based on the preliminary series of tests that were carried out using only the first one. These tests and their results are discussed in section 6.4.

**1) First Version:** ***Interference Minimization (IMin)***
or *minimization of the sum of the Interferences*objective functions)

$$\min z_1 \;=\; \sum_{A\;\in\mathcal{V}} \left( \sum_{B\;\in\mathcal{V}} \; \sum_{i,j\;\in\mathcal{R}} w_{x_{B,i}x_{A,j}}\; x_{B,i}\; x_{A,j} \quad + \sum_{j\;\in\mathcal{R}} n_j\; x_{A,j} \right)$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i\in\mathcal{R}} x_{A,i} && = 1 && , \forall A \in \mathcal{V} \\
& \sum_{B\;\in\mathcal{V}} \; \sum_{i,j\;\in\mathcal{R}} w_{x_{B,i}x_{A,j}}\; x_{B,i}\; x_{A,j} \quad + \sum_{j\;\in\mathcal{R}} n_j\; x_{A,j} && \le \beta && , \forall A \in \mathcal{V} \\
& x_{A,i} && \in \{0,1\} && , \forall A \in \mathcal{V} \text{ and } \forall i \in \{1,\dots,r\}
\end{aligned}$$

**2) Second Version:** ***Signal Maximization (SMax)***
or *maximization of the sum of the SINRs*

$$\max z_2 \;=\; \sum_{A\;\in\mathcal{V}} \frac{1}{\left(\sum_{B\;\in\mathcal{V}} \; \sum_{i,j\;\in\mathcal{R}} w_{x_{B,i}x_{A,j}}\; x_{B,i}\; x_{A,j} \quad + \sum_{j\;\in\mathcal{R}} n_j\; x_{A,j}\right)}$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i\in\mathcal{R}} x_{A,i} && = 1 && , \forall A \in \mathcal{V} \\
& \sum_{B\;\in\mathcal{V}} \; \sum_{i,j\;\in\mathcal{R}} w_{x_{B,i}x_{A,j}}\; x_{B,i}\; x_{A,j} \quad + \sum_{j\;\in\mathcal{R}} n_j\; x_{A,j} && \le \beta && , \forall A \in \mathcal{V} \\
& x_{A,i} && \in \{0,1\} && , \forall A \in \mathcal{V} \text{ and } \forall i \in \{1,\dots,r\}
\end{aligned}$$

Any feasible solution to one of these program should at least be a *setting*, which corresponds to picking an emission power from the set $\mathcal{P}$ for each base station. This is enforced by the essential constraint:

$$\sum_{i\in\mathcal{R}} x_{A,i} = 1 \quad , \; \forall A \in \mathcal{V} \tag{3}$$

There is also an *upper-bound constraint* $\beta$ on the value of the Interference at each base station:

$$\sum_{B \in \mathcal{V}} \sum_{i,j \in \mathcal{R}} w_{x_{B,i} x_{A,j}} \; x_{B,i} \; x_{A,j} \leq \beta \qquad , \forall A \in \mathcal{V} \tag{4}$$

which equivalently means that the quality of the signal quality should be higher than a minimum value. One can optionally set $\beta = \infty$, the problem being then called *unconstrained*.

## 6.3 Possible Graph Representation

We can represent this problem in a graph $G_F(V_F, E_F)$, where $V_F$ is the set of *power nodes* corresponding to each base stations (as in the node-deletion model in section 5.1). If base station $A$ and $B$ are separated by a distance $d$ smaller than the threshold distance, than there exists a weighted directed edge $(A_i, B_j)$ in $E_F$ between the pairs of power nodes of these two base stations, with the value $\mathrm{I}(P_i, P_j, d)$. We also include the auxiliary $X$ node of the previous model, for taking into account the noise factor.

In this context, a *setting* corresponds to an induced subgraph where every base station is represented by exactly one power node, and the $X$ node is present. Consequently, the sum of all the weights of the edges present in this subgraph is equal to the value of that setting with the first objective function.

Also note that we do not allow base stations to be turned off (for reasons explained in the following section), a property expressing this would therefore not be *hereditary*, and thus is beyond the scope of [8].

## 6.4 Preliminary set of Results and their Implications

It is very interesting to analyze the preliminary set of tests that led us to these two non-equivalent formulations of the problem, with two different objective functions.

When we first designed this model, we only considered the *Interference minimization* objective function, and we also incorporated an additional constraint for maintaining a good *coverage* everywhere. We also allowed some base stations to be turned off (and hence have no interference over them), as long as the area around them was covered in a satisfactory way by other neighboring base stations.

### 6.4.1 Uniform Setting of Power

After running a series of test with a *brute force* algorithm finding an optimal solution on a few randomly generated networks with 5 base stations and 3-4 levels of power, we observed that:

- in the *unconstrained* case, the best solution is obtained by setting some of the base stations to $P_{\max}$, and the others are turned off.

- when we progressively lowered the upper-bound, the network tended to react by chunks, meaning that base stations that were close to each other would act similarly, i.e. change together to the same level.

This is partially due to the following property: consider a network composed of only two base stations that are set at the same power setting $P_i$ and where we neglect $N_0$. We get an objective function value of

$$I_s(A) + I_s(B) = \frac{P_B \cdot d_{B,A}^{-\alpha}}{P_A \cdot d_{\text{ref}}^{-\alpha}} + \frac{P_A \cdot d_{A,B}^{-\alpha}}{P_B \cdot d_{\text{ref}}^{-\alpha}} = \frac{P_i \cdot d_{B,A}^{-\alpha}}{P_i \cdot d_{\text{ref}}^{-\alpha}} + \frac{P_i \cdot d_{A,B}^{-\alpha}}{P_i \cdot d_{\text{ref}}^{-\alpha}} = 2 \cdot \frac{d_{A,B}^{-\alpha}}{d_{\text{ref}}^{-\alpha}}$$

When we move away from this *uniformly* distributed power setting solution, for example by setting the power of $B$ one power level lower (i.e. by dividing its power by two, see section 3.3), we get

$$I_{s'}(A) + I_{s'}(B) = \frac{P_i/2 \cdot d_{B,A}^{-\alpha}}{P_i \cdot d_{\text{ref}}^{-\alpha}} + \frac{P_i \cdot d_{A,B}^{-\alpha}}{P_i/2 \cdot d_{\text{ref}}^{-\alpha}} = (2 + \frac{1}{2}) \cdot \frac{d_{A,B}^{-\alpha}}{d_{\text{ref}}^{-\alpha}}$$

which is higher than our previous solution, and thus less good. This lead us to the intuition that the uniform maximum power level is the optimal solution in the unconstrained case for Interference Minimization.

We first tried to prove mathematically that intuition, but all our attempts have remained unsuccessful so far. Then, we tried to use finer power level sequences, as well as other sequences, without managing to find better solutions that would differ from the uniform one, thus giving some more credits to our first observations. Therefore, we propose this as *an open question for further research on this topic.*

This also motivated us to consider an alternative objective function which would have another less trivial optimal solution in the unconstrained case. We chose the Signal Maximization objective function, because it makes equally sense in the context of our problem and it uses the same notions as the previous objective. However, the first objective remains much easier to understand and interpret in the context of a graph.

### 6.4.2 (Almost) Equivalent Solutions

One can easily show that given a setting, an almost equivalent but better solution is obtained by setting each base stations' power one level higher (only when this is possible for all base stations, of course). The two solutions differ only by a very tiny value, due to the noise component, and are even exactly equal when $N_0 = 0$.

Another related effect worth of noticing is that when only one base station's power is lowered one level, the *Interference* over this base station doubles.

### 6.4.3 Coverage Constraint and inactive Base stations

We also tried to include an extra constraint on the *coverage* of the area. However, we could not come up with a satisfactory expression of the quality of the coverage around a base station, and we finally gave up that idea after a few series of tests.

In solving Interference Minimization problems, having a base station turned off is very interesting, since the interference over it is defined as null. In fact, the extreme case of all base stations being turned off constitutes the absolute best solution when there is no coverage constraint. Therefore a coverage constraint is necessary if turning off some base stations is allowed.

# 7 Game Theory

We present general notions of game theory and we describe the game we use with our model.

## 7.1 General Game Theory Definitions

We define a *strategic-form game* as follows:

**Definition 3 (strategic-form game)**
*A very general definition of a* strategic-form game *is any* $\Gamma$ *of the form*

$$\Gamma = \left(\ N,\ (C_i)_{i\in N},\ (u_i)_{i\in N}\ \right)$$

*where* $N$ *is a non-empty set of* players *in the game and* $C_i$ *is the non-empty set of* strategies *available to the player* $i$*. A* strategy profile $c = (c_j)_{j\in N}$ *is a combination of strategies that the players in* $N$ *might choose in* $C = \times_{j\in N} C_j$ *. The* utility function $u_i$ *of player* $i$ *is a function* $u_i : C \longrightarrow \mathbb{R}$ *representing the utility payoff for the player* $i$ *for this profile.*

*To play the game, each player simultaneously chooses a strategy (usually the strategy that maximizes its expected utility payoff), resulting in a strategy profile* $c$*. The outcome of the game for each player is the value of its utility function for the strategy profile* $c$*.*

We also define an additional notion very common in game theory.

**Definition 4 (Nash equilibrium, stable strategy profile)**
*A strategy profile* $c$ *is a* Nash equilibrium *(or is called* stable*) if and only if no player can improve his utility by unilaterally deviating from* $c$ *and choosing an other strategy for himself while the other players keep their initial strategy.*

Finally, we also define:

**Definition 5 (price of anarchy)**
*The* price of anarchy *is the ratio between the quality (from a global point of view) of the strategy profile obtained with the worst Nash equilibrium of the game, and the best quality that would be obtained with a strategy profile chosen by a central authority.*

## 7.2 Game Played in Our Model

In the game we design for our model, the players are the base stations, and the set of available strategies for each player is the set of emission power $\mathcal{P}$ at which this base station can emit[4]. Therefore, a strategy profile is equivalent to a power setting, and we will by extension say that a setting is *stable* if its corresponding strategy profile is stable. The utility functions we use are described in the next sections.

However, we do not play exactly as in the definition of a strategic-form game, but rather sequentially pick more or less randomly a player (i.e. base station) and try to maximize its utility value by changing only its strategy in the global strategy profile. We repeat this process infinitely unless a Nash equilibria is reached, in which case we stop the game. This can also be considered as a *Nash-program*, used to find Nash equilibria in a strategic-form game.

## 7.3 Utility Functions Used

We present here five different utility functions, three of which we will test in our simulations.

First of all, we need to define the positive increasing function $f$ for a given *upperbound constraint* $\beta < \infty$:

$$f(x) = \begin{cases} log(x \cdot \beta) & \text{if } x \geq \frac{1}{\beta} \\ 0 & \text{else} \end{cases}$$

When $\beta = \infty$, we define this function as $f(x) = log(1 + x)$. The parameter $x$ here is meant to represent an estimation of the quality of the signal with $1/\mathrm{I}_s(A)$, and if this value is below the threshold of $1/\beta$, the value of the function $f$ should be zero (see figure 5).

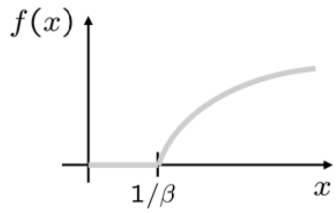


Figure 5: Graph of the function $f$ with $\beta < \infty$.

We also define $g(y) = f(1/y)$ and $\mathcal{F}(A)$ as the set of neighbor base stations of $A$ that also belong to its operator.

Given a strategy profile $c$ (and its equivalent setting $s$), the utility functions of a base station $A$ should depend (at least partially) on the Interference $\mathrm{I}_s(A)$ over itself. The first two functions we define are:

[4]An alternative game we could have designed would have the operators as the players.

- `SINR` : $u_{1,A}(c) = 1/\mathrm{I}_s(A))$
  We directly use the estimation of the SINR of a user at 1m of the base station.

- `base` : $u_{2,A}(c) = g(\mathrm{I}_s(A))$
  We use the function $f$ to obtain a utility of zero if the Interference $\mathrm{I}(A)$ is higher than $\beta$.

The following three functions try to also take into account for a base station the effect of its strategy on the base stations that belong to the same operator. Thus the next utility functions are not purely egoistical anymore. The parameter $\gamma$ is used to put the emphasis on either the base station or its neighbors, its default value being $\gamma = 1$.

- `baseWithFriends` :
  $$u_{3,A}(c) = g(\mathrm{I}_s(A)) + \gamma \cdot \sum_{B \in \mathcal{F}(A)} g(\mathrm{I}_s(B))$$
  We add to the `base` of A the sum of the `base` for its friends.

- `baseWithFriendsScaled` :
  $$u_{4,A}(c) = g(\mathrm{I}_s(A)) + \gamma \cdot \frac{\sum_{B \in \mathcal{F}(A)} g(\mathrm{I}_s(B))}{|\mathcal{F}(A)|}$$
  We add to the `base` of A the average of the `base` for its friends.

- `basePlusOperatorNeighbourhood` :
  $$u_{5,A}(c) = g(\mathrm{I}_s(A)) + \gamma \cdot \frac{g(\mathrm{I}_s(A)) + \sum_{B \in \mathcal{F}(A)} g(\mathrm{I}_s(B))}{1 + |\mathcal{F}(A)|}$$
  We add to the `base` of A the average of the `base` for its friends and also its own.

## 7.4 Very Important Observation

It is crucial to realize that the final settings obtained *do absolutely not depend* on whether we are trying to solve the Interference Minimization or the Signal Maximization! It is only the utility function, in conjunction with the way the next considered base station is chosen at each step during the game, that hopefully guides the players to a Nash equilibrium.

Our main goal is to design *good* utility functions that hopefully reach Nash equilibria close to the optimal setting for the network, either for Interference Minimization or Signal Maximization.

# 8 Program Implementation

In order to run a big number of simulations on our model, we developed a library and some interfaces in Java. In this section, we will first give some general information on the program and its components. Then, we will explain how the *games* (or more precisely the *Nash-programs*) and some other *heuristics* we implemented work.

## 8.1 General Information

### 8.1.1 Software and Hardware

The program we developed was written in java 1.5 (or equivalently java 5.0, following the new terminology). The graphical possibilities offered by java enabled us to easily visualize in a window the networks we were working on. The simulations were performed on a Dell computer with a 600 MHz Intel Pentium III processor and with 128 MB of RAM. Finally, the data was processed using Matlab.

### 8.1.2 How to Use the Source Code on the CD

As there are many parameters that need to be set for a run, the option we chose was to set all the parameters in the interfaces' source code and to re-compile it each time. Therefore, in the case of `MultipleRunLauncher`, a mechanism was included that copied the custom-edited source code of the interface each time it was run so that one could reuse it.

The easiest way to use the provided source code would be to:

1. copy the whole content of the `all` folder on the CD to a new folder on your hard-drive.

2. replace the default `MultipleRunLauncher.java` file in that folder with the `MultipleRunLauncher.java` of the run you wish to reproduce

3. compile all the files *WITH an up-to-date compiler for java 1.5*

4. call the program with the command:
   `java -classpath . MultipleRunLauncher true`

The reader is encouraged to edit the source code of the interfaces if he wants to perform some runs of his own.[5]

[5]Please, feel free to e-mail me if you have any questions or concerns, at: sivan.altinakar@epfl.ch

The model representation and algorithm classes can also be used as libraries for other programs. A simple graphical interface is currently under development. It will be used to present, during the final defense, small examples of what can be done with the tools that were created in the course of this project.

## 8.2 Overview of the Classes

The source code is divided in 4 categories: model representation, algorithms, interfaces and useful tools. The idea was to write some sort of a library of objects and methods (i.e. model representation group), on which the different algorithms, namely games and heuristics (second group), would be built. Finally, we wrote some specific programs (or interfaces) to use those algorithms in the desired way. For instance, `SharedSpectrumSolver` is a simple testing program, and on the other side, `MultipleRunLauncher` is a very elaborate program capable of performing thousands of consecutive runs with a set of configurations to test on a particular network, and which also outputs the results in several different formats, some meant to be understood by humans, some others immediately ready to be processed in a program such as Matlab.

In the appendices, we provide a complete list of the different classes with their specific features.

## 8.3 Games and Algorithms Implementation

In figure 6, we explain how a *game* is implemented, as well as a parallel *simple heuristics* that consists of keeping in memory the best strategy profile encountered so far (with respect to one of our objective function)[6].

In the rest of the text, we will call the act of one base station changing its strategy a *move*.

In our procedure, there are three stopping criteria. The first one is the usual maximum number of iteration that should be performed. The second one is that the current strategy profile is a Nash equilibrium (i.e. after the previous move, all the base stations had a chance to change their strategy, but did not do it). The third one is a maximum number of iteration to perform which do not result in a move after the last move.

The choice of the base station at each iteration can be done in different ways. The ones we used are discussed in section 8.5.

[6] This is done in the abstract superclass `Algorithm`.

**Program**

**Initialization:** the parameters needed are

- the network on which to work
- the objective function, either *Interference Minimization* or *Signal maximization*
- the upperbound constraint $\beta$ (if defined)
- the initial strategy profile (=power setting)
- the parameters of the game and the heuristic (utility function, choice of next base station,...)

**Result:**

- the final strategy profile reached (result of the *game*)
- the strategy profile encountered that had the best objective function value (result of the *algorithm*)

**Procedure:**

While a *stopping criteria* is not met, perform the steps

1. choose a base station
   (using the selected procedure for choosing the next base station)
2. choose a strategy for this base station
   (using the selected utility function)
3. update the best strategy profile encountered
   (if necessary)

Figure 6: Overview of the program used for performing one simulation.

Finally, we also implemented a method[7] that finds the optimal solution value for networks with at most 10 base stations and 7 power levels. This allows us to compare the quality of the solutions we find with our *games* and *algorithms* to the optimal one.

## 8.4 Additional Fine-Tuning Capabilities

We can further control and change the behavior of a game or heuristic by limiting the number of strategies that a base station can move to. We do this by defining a range which specifies how many power levels below and above

[7] In the class `BruteForce`.

its current level a base stations can use to try to maximize its utility. The point of this would be to control the number of iterations needed for a base station to change from a given strategy to an other one, because by changing too fast (or immediately), it would maybe not allow the other base stations to progressively adapt to its move, thus not allowing the group to globally move to a good strategy profile.

Another option we implemented is the possibility to keep a *tabu list* in which we *put base stations* after we have chosen them at an iteration. Base stations are maintained in that list for a predetermined number of iteration after they have been chosen, and they cannot be chosen for a future iteration during that time. With this element, our heuristics becomes a very simple *tabu search*.

## 8.5 Different Ways to Choose the Next Base Station

We defined four different *procedures*, or ways to choose a base station at each iteration. The first two are oriented toward simulating games, while the other two are more oriented toward very simple tabu searches[8].

- `RandomSearch`
  At each iteration, chooses randomly a non-tabu base station. Stops when no move has been observed for more than $n$ iterations, where $n$ is five times the total number of base stations.

- `SequenceSearch`
  Orders randomly the base station in a sequence, and then cycles through that sequence until no move has been performed for a whole cycle. This results in a final solution that is a *Nash equilibrium* or *stabilized*.
  (Note: a tabu list is meaningless in this case.)

- `GlobalTabuSearch`
  At each iteration, chooses the non-tabu base station with the least utility. Stops when no move has been observed for more than $n$ iterations, where $n$ is size of the tabu list plus 3. This means that the algorithm has started cycling in the tabu list.
  (Note: this algorithm uses no random numbers, and thus when repeatedly applied on a network with the same initial solution, it will always produce the same outcome.)

[8]These procedures are implemented in the class `MultipleRunAlgorithmCore` as subclasses of the `TabuSearchCore` class with some of its parameters set by default, in order to get the expected behavior in each case.

- `DistributedTabuSearch`
  At each iteration, first chooses an operator randomly (respecting the proportion of base stations owned by each operator), then chooses the non-tabu base station with minimum utility that belongs to it. Stops when no move has been observed for more than $n$ iterations, where $n$ is 5 times the size of the tabu list times the number of operators.

# 9 Simulations

The objective of these simulations is to study some of the numerous parameters individually. Our results can be used in future researches for limiting the number of parameters to study.

## 9.1 General Hypothesis

Before doing any simulations, one first needs to specify some of the parameters of our network.

### 9.1.1 Environment Parameters

We choose to simulate a cellular network and all the distances will be expressed in kilometers. It is important to note that the value of $N$ and $\beta$ depend on the chosen unit. Here is our set of parameters:

- $N = 0.0001$
- $\alpha = 4$
- $d_{\text{thresh}} = 10$ km

### 9.1.2 Power Parameters

For the set of available powers, we chose arbitrarily a maximum value of 100 and decided to consider 5 power levels. We get the following set:

$$\mathcal{P} = \{6.25, 12.5, 25, 50, 100\}$$

The choice of 5 levels is realistic enough in practice[9].

### 9.1.3 Upper-Bound Constraint $\beta$

Finally, we decided to work with an *unconstrained* problem, i.e. $\beta = \infty$ (see 6.2 for the definition), although our program would allow us to set it a finite value. This parameter is really crucial, and it would most certainly require many tests to determine a good value. As this task would require more time then we had at hand, we chose not to take it into account. However, this would be an important topic to work on for anyone who wishes to do some further research on this model.

[9] It is also interesting to note that on a network with 10 base stations, the `BruteForce` algorithm would take for instance 40 minutes for 7 levels, instead of 3 minutes with 5.

## 9.2 Studied Networks

We studied two different networks of 10 base stations, with 2 operators having each 5 base stations. Some additional data for a few other networks generated with the `PredefinedNetworkLayout` class is also available on the CD.

### 9.2.1 Random Network

A randomly generated connected network (see figure 7).

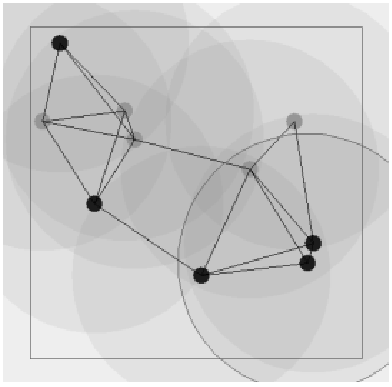

Figure 7: Random Network in a square area of 25.65 km of side.

### 9.2.2 Pyramidal Network

A network with a pyramidal structure, and where the base stations have been randomly attributed to each operator (see figure 8).

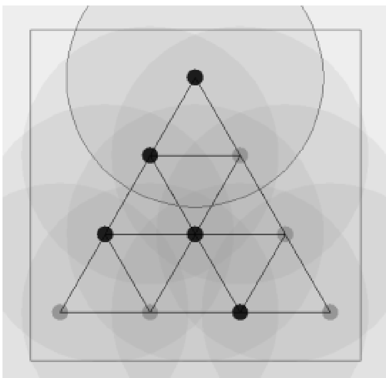

Figure 8: Pyramidal Network with small triangles of 7 km of side.

## 9.3 Experiment

In our experiment, we test for each of the four predefined procedures (`SequenceSearch`, `RandomSearch`, `GlobalTabuSearch` and `Distributed TabuSearch`) all the possible configurations created from the following parameter sets. Each individual configuration is repeated 20 times, in order to take the average of the results. For each parameter, the tested possibilities are (see section 7 and 8 for detailed information):

- **Objective Function:** Interference Minimization and Signal Maximization
- **Utility Function:** `base`, `baseWithFriendsScaled` and `basePlusOperatorNeighbourhood` utility functions, where the latter two were tested with the following values of $\gamma \in \{0.2, 1.0, 5.0\}$. This amounts in 7 utility functions.
- **Initial Setting:** every base station is set to the minimum power level (PMin), to the maximum power level (PMax) or individually to a random power lever (PRan)
- **Range:** *free range* (any power level can be chosen) and *1-step range* (the allowed levels are the current one, one level higher and one level lower, when possible)
- **Tabu List Size:** without a tabu list and with a tabu list size of 1, 3, 5 and 7 (except for `SequenceSearch`)

This results in roughly 20 thousand runs. The best solution for both objective functions is also found using *Brute Force*.

# 10 Results

The data gathered from the simulations is in the following folders on the CD:

- `serie_001` : Random network
- `serie_002` : Pyramidal network

## 10.1 Duration and general number of performed iterations

For both networks, the first essential result is that almost all runs stopped either because they had reached a stability point or, in the cases of procedures choosing base stations with minimum utility such as `GlobalTabuSearch` and `DistributedTabuSearch`, also because they had started cycling among the base stations with minimum utility. The few remaining ones stopped because of the *maximum number of iteration without move* criteria. Moreover, the run-time for a single run is usually less than half a second (see figure 9).

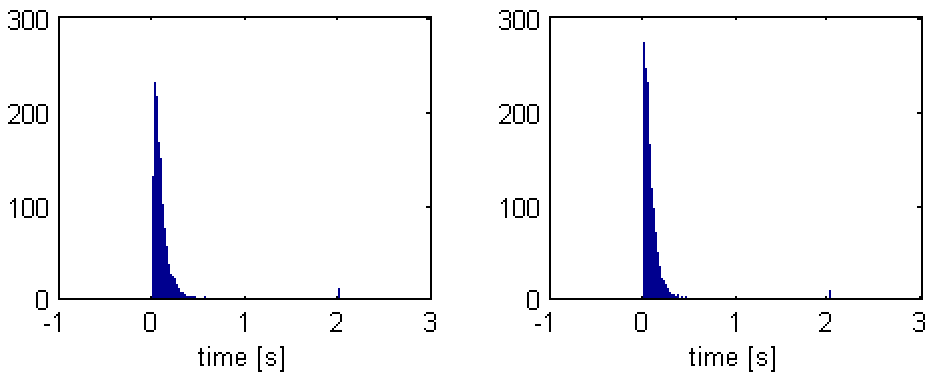


Figure 9: Histograms of run-time for the *Random* (left) and *Pyramidal* (right) network.

We can also observe that, although some parameters highly influence the number of iterations needed, `SequenceSearch` (SEQ) and `GlobalTabuSearch` (GTS) perform an average of 15-25 iterations, while `RandomSearch` (RAN) and `DistributedTabuSearch` (DTS) perform an average of 25 to 35 iterations. This will be discussed again at the end, but this time by fixing some parameters, in order to be really able to compare the performances of these procedure. Also, for the sake of simplicity, in the rest of this section we will refer to these different ways to choose the next base station by these acronyms (their detailed definition can be found at section 8.5).

One particular parameter worth of interest that affects these measures is, as expected, the range of available strategies (or power levels) in which a base station can pick its next move. Limiting this range to the actual power level, the one right below and the one right above (*1-step* range as opposed to a *free* range), usually doubles the number of iterations needed by GTS and DTS.

## 10.2 Nash equilibria at the end of the games

In figures 10 and 11, we illustrate the percentage of runs that ended at a Nash equilibria for different utility functions, procedures and ranges. We studied seven utility functions, one being `base` (BA), and the other six being `baseWithFriendsScaled` (BWFS) and `basePlusOperatorNeighbourhood` (BPON), each with three values of $\gamma$ (see sections 7.3 and 9.3). The results for $\gamma \in \{0.2,\ 1,\ 5\}$ are represented in an increasing order by individual bars for the *free range (1)* and the *1-step range (2)*. SEQ is not represented here, because we defined it to only stop when it reaches a Nash equilibrium, which always happened.

For RAN, we observe that it usually reaches a Nash equilibrium 99% of the time, the remaining 1% being due to the procedure not randomly picking the remaining base stations that can improve their utility early enough, and hence performing a number of iteration without a move larger than the allowed limit.

For DTS and GTS, let us first only consider the case of a *free range*. In the *Pyramidal* network, both of them reach a Nash equilibrium 80-90% of the time. As said earlier, in the remaing cases they stop because they have started cycling among the base stations with minimum utility. However, in the *Random* network, this score drops to 60-75% for DTS and to 30-60% for GTS. Then, we observe that a *1-step* range diminishes the efficiency of these procedure, particulary in the case of the Pyramidal network.

This confirms our intuition that a structured layout of the base stations (such as in the Pyramidal network) is a special case which gives us systematically better results with these two procedures, but changing a parameter such as the range immediately disturbs it and leads to a big drop in efficiency, GTS being more dependent of that factor than DTS.

We can also notice that BWFS with the highest value of $\gamma$ systematically gives much worse results than the other utility functions.

Finally, please note that at this point, these results are an average on the different initial solutions and tabu list lengths. We will progressively study

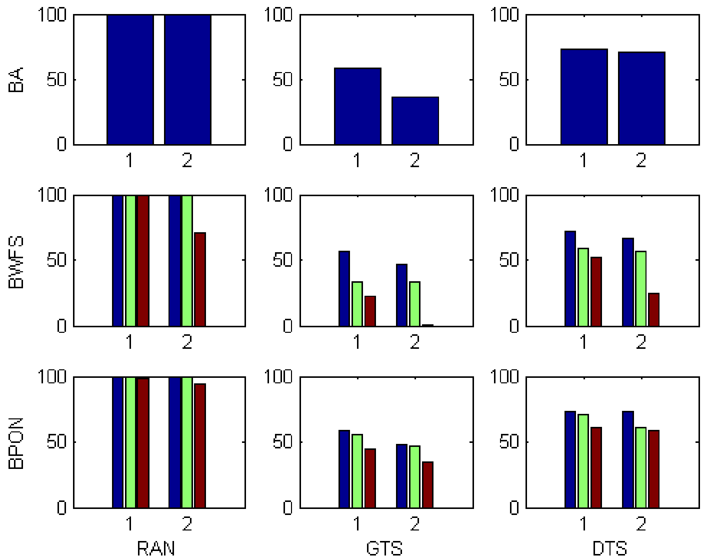


Figure 10: Percentage of runs that ended at a Nash equilibria for different utility functions, procedures and ranges, in the Random network.

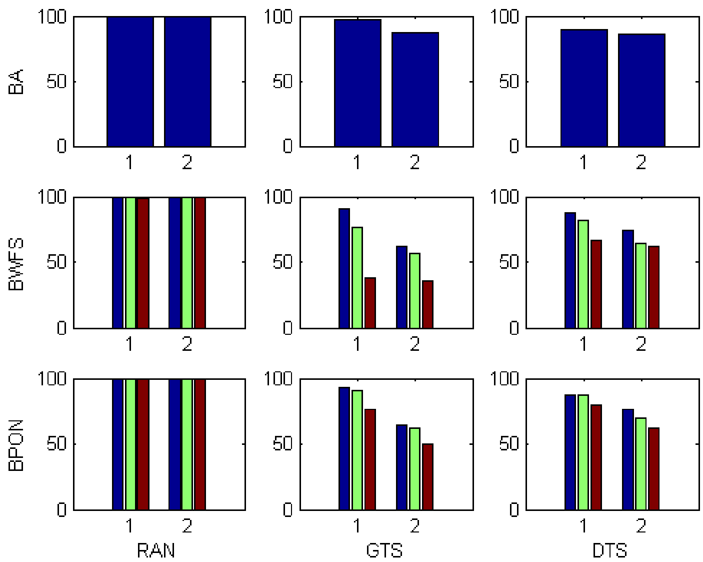


Figure 11: Percentage of runs that ended at a Nash equilibria for different utility functions, procedures and ranges, in the Pyramidal network.

the effect of each of these parameters individually in the following sections.

## 10.3 Reached Nash Equilibira

In the *Random* network, for five out of seven utility functions, we only found a unique Nash equilibrium where every base station sets its power to the highest level allowed, named the *PMax* setting or strategy profile. The other remaining two are BWFS and PBPON with the highest value of $\gamma$, and have respectively one and two Nash equilibria composed of respectively 4 and 9 base stations at maximum power level, and the remaining others at the minimum power level. Mixed solutions such as these, with either maximum or minimum power levels, are named *PMaxMin* settings or strategy profiles.

In the *Pyramidal* network, they all have a unique Nash equilibrium, six at PMax and one at PMin (i.e. everybody at the minimum power level). The latter one is again BWFS with the highest $\gamma$.

As expected, the more you take into account, not only the Interference over yourself, but also the Interference over neighboring base stations which belong to the same operator, the less likely you are to choose the maximum power level as the best choice for you. In our case, this behavior is achieved by choosing a high $\gamma$. We can also notice that in that sense, BWFS is much more radical than BPON. This is most probably due to the fact that in the part influenced by $\gamma$ in BWFS, you do not consider yourself, as opposed to that same part in BPON where you also include yourself. Note that in the former, your utility drops significantly when you do not have any "friends" nearby, independently of your strategy choice.

Finally, it is very interesting to observe that by limiting the range, we manage to create new additional Nash equilibria close to the usual ones, while of course the usual ones still remain valid. Note that in our case, this only worked for the Random network.

## 10.4 Initial Strategy Profile

As PMax is a Nash equilibrium for most of the utility functions, starting at this strategy profile generally results in an immediate stop of the games.

In the other cases, all the games with the different utilities reach their Nash equilibrium 95-100% of the time with the RAN procedure, independently of their starting point. For GTS and DTS, we observe that it is strangely easier to reach their Nash equilibrium (usually PMax) starting from PMin than from PRan.

However, when we limit the range to *1-step*, the efficiency we observe drops tremendously, especially when starting from PMin, probably because it is not anymore possible to go all the way up with just a few iterations from a uniform setting to another uniform one. This results in the total inability for the games with certain utilities to reach a Nash equilibrium using GTS.

## 10.5 Tabu List Length

For the results discussed in this section, we only used the *free-range* and PRan as the initial strategy profile. In figures 12 and 13, we illustrate the percentage of runs that ended at a Nash equilibria, depending on the absence or length of the tabu list. We also do not need to consider SEQ, which does not use a tabu list. We tested the following lengths: 0 (no list), 1, 3, 5 and 7, in our two networks which had 10 base stations each. In the figures, these five lengths are grouped in an increasing order, and in the case of BWFS and BPON, for $\gamma = 0.2$ (group 1), $\gamma = 1$ (group 2) and $\gamma = 5$ (group 3).

With the RAN procedure, all the games with the different utilities reach their Nash equilibrium 97-100% of the time, regardless of the presence or the length of the tabu list.

In the case of GTS and DTS, the tabu list is used to prevent them from cycling among a few base stations with minimum utility, as at each step, it is the non-tabu base station with minimum utility that gets chosen, either in the whole network for GTS or among the base stations of a randomly picked operator for DTS.

Hence, we observe that with a length set lower than respectively 5 and 3, they are totally useless for the Random Network, meaning they never reached a Nash equilibrium in our simulations. Furthermore, it seems that the longer the list, the better the chances of reaching a Nash equilibrium. This is confirmed by the fact that in our case with a network of 10 base stations, a longer list of length 9 would exactly be the SEQ procedure, which globally achieves the best results in the minimum number of iterations, thus being our best procedure so far for choosing the next base station at each iteration. Also note that the initial order in which the base stations are added to the list determines the order of the sequence.

We finally remark that, regardless of the tabu list length, DTS reaches a Nash equilibrium more often than GTS, the difference being more significant in the Random network than in the Pyramidal one. This is probably due to the random element in DTS. Also, both DTS and GTS generally experience difficulties in reaching a Nash equilibrium with BWFS utilities that have less trivial Nash equilibria, due to a high value of $\gamma$.

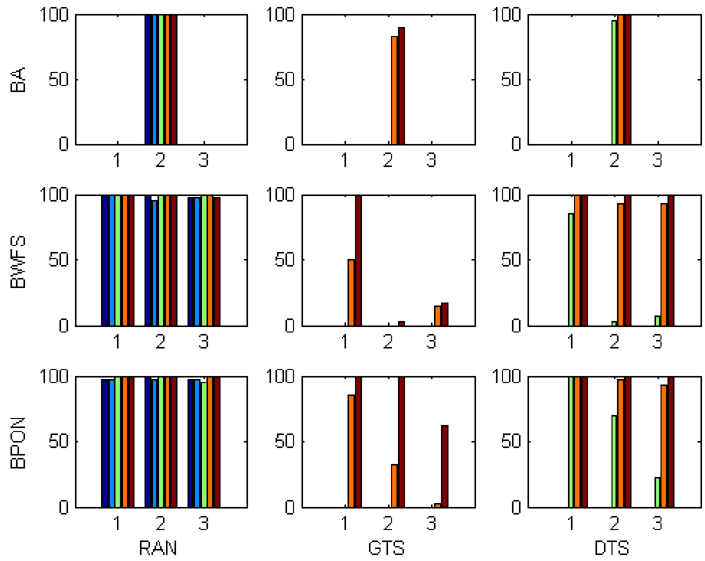


Figure 12: Percentage of runs that ended at a Nash equilibria depending on the tabu list length, in the Random network.

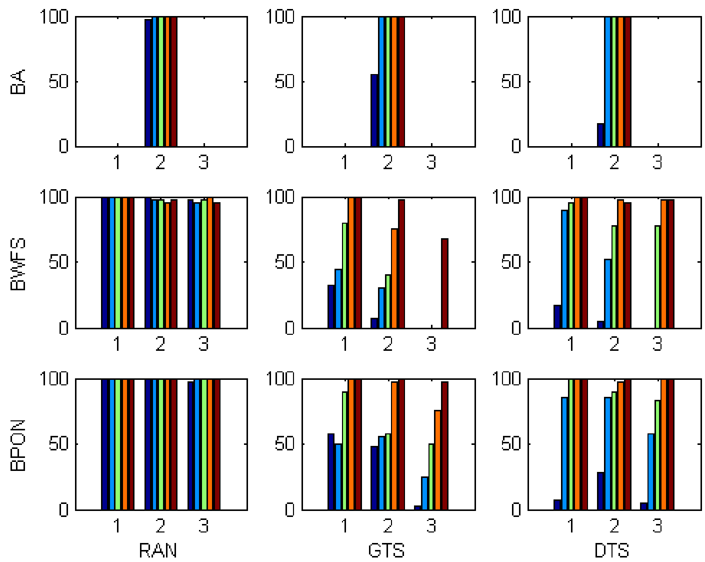


Figure 13: Percentage of runs that ended at a Nash equilibria depending on the tabu list length, in the Pyramidal network.

## 10.6 Objective Function Value

Our objective when we designed these games, was to define some rules by which each base station could play individually. These rules would also need to eventually guide the whole network to a strategy profile, or equivalently a power setting, with an objective function value (with either IMin or SMax) that would be close to the optimal solution.

It is interesting to note that we obtain completely opposite results with our two objective functions.

### 10.6.1 IMin: Interference Minimization

For this objective function, the `BruteForce` algorithm tells us that the *optimal solution* is in fact PMax, the *unique Nash equilibrium* for almost all utility functions in both considered networks (see section 10.3). Therefore, most of the games we designed are totally successful in guiding the base stations to the optimal solution and our Price of Anarchy is equal to one. This also implies that our tabu search heuristics reach the optimal solution and stop. This is a great achievement!

However, let us not forget that this PMax solution has only been observed for $\beta = \infty$. In our preliminary series of test (see section 6.4), we remarked that when we lower the value of $\beta$, we get more balanced and less trivial solutions. The next step would most certainly be to study if the utility functions are able to adapt to these new more complex optimal solutions, for finite values of $\beta$. Two principal points of interest to study for the games would be to check if we still get a *unique* Nash equilibrium and if this Nash equilibrium is still close to the optimal solution for the network (or at least one among several Nash equilibria, in the case where the first point is false).

### 10.6.2 SMax: Signal Maximization

For this objective function in both considered networks, the optimal solution is a PMaxMin setting, i.e. a setting where some base stations emit at the maximum power level, while the others emit a the minimum power level. As this solution is not at all a Nash equilibrium for any of our utility functions, the results we get are really bad. The tabu search heuristics only encounters the optimal solution (or comes close to it) on very rare and purely accidental occasions, while the game is conducting the network to the PMax solution. To compare with the previous objective where we almost always stopped at the optimum, with the SMax objective we reached the optimum less than 20 times out of some ten thousand simulations.

Obviously, our utility functions are not adapted to this objective function and we need to design some new ones that are better suited for it. Moreover, it is not interesting or relevant to consider the Price of Anarchy of the games based on these utility functions, because these games are not even *trying* to go close to the optimal solution. It is not as if in their *attempt* to guide the networks toward their optimal solution, they were trapped in Nash equilibria with lower objective function values.

## 10.7 Summary

We can summarize all our observations by studying the results we obtain with a specific set of parameters and with with the BA, BWFS and BPON utility functions, the latter two with only one value of $\gamma$.

- **low value of $\gamma$**
  We choose for $\gamma$ the lowest value of the three considered, because we observed that higher values tend to give us Nash equilibria different than the unique PMax otherwise obtained. In that sense, BWFS is also much more radical than BPON.
- **free range**
  We do not restrict the range, as this tends to create additional unwanted Nash equilibria.
- **PRan**
  We decide to start from a random power setting, in order to observe how the games manages to reach a Nash equilibrium.
- **tabu list of length 5 for GTS and DTS**
  We set a value not too low, because otherwise these two procedures tend to get trapped into cycling among a few solutions with a very low utility value. For RAN, we choose not to use a tabu list, as this factor does not seem to influence the procedure.

In the table below, we can see the average number of iterations needed before the games stop, depending on the type of procedure for choosing the next base station and the network. We performed 120 runs (40 for each utility function) for each cell to obtain the average. In each case (except for GTS in the Random network, marked with the * sign) the games *always* stopped at the PMax strategy profile (or equivalently power setting), because it is a Nash equilibrium for the three utility functions considered.

| | | Random network | Pyramidal network |
|---|---|---|---|
| `RandomSearch` | (RAN) | 32 | 31 |
| `SequenceSearch` | (SEQ) | 20 | 20 |
| `GlobalTabuSearch` | (GTS) | 23* | 18 |
| `DistributedTabu` | (DTS) | 50 | 44 |

This example illustrates the following points of interest:

- **IMin and SMax**
  For the IMin objective function, the unique Nash equilibrium of the games at PMax is exactly the optimal solution for this network, therefore whenever the game and the corresponding heuristic stop, they do it at the optimum.
  For SMax, on the contrary, the optimal solution is a PMaxMin which is composed of some base stations at the maximum power level and the others at the minimum, and which is a Nash equilibrium for none of the considered utility functions. Consequently our utility functions are definitely not well suited for this objective, because the heuristics can reach or come close to the optimal very rarely and purely accidentally, while the game is wandering in the solution space toward the PMax Nash equilibrium.
- **Random and Pyramidal networks**
  The games with GTS and DTS (which choose at each iteration a non-tabu base station with minimum utility, respectively in the whole network and among the base stations of a randomly picked operator) necessitate significantly less iterations in structured networks. However, this factor seems not to affect RAN and SEQ (which choose at each iteration the next base station respectively randomly and following a predefined sequence).
- **GTS and DTS**
  DTS works better than GTS, probably because its random component helps it a little bit more to avoid cycling. However, when it does not get trapped (as in the more structured Pyramidal network), GTS works much faster than all the others.
- **SEQ**
  This procedure can be considered the best one, because it consistently stops at a Nash equilibrium and it achieves it usually faster than the other ones.

# 11 Suggestions for Further Research

## 11.1 Effects of the upper-bound constraint $\beta$

Through the study of a theoretical case of our problem, where $\beta = \infty$, we observed that for several studied networks, the optimal solutions of the two objective functions we have defined seemed to follow a pattern. For this reason, we propose the following two open questions:

**Open Question 1:** In the *Interference Minimization* case, does the maximum setting for all base stations always give us the optimal solution?

**Open Question 2:** In the *Signal Maximization* case, is the optimal setting always only composed of maximum and minimum power levels?

Answering these questions would be a natural next step to take. But it would also be of great interest to study the situations where $\beta < \infty$, for which we have observed that the optimal solutions for the networks get more complex and balanced. In that case, are the *utility functions* we defined still efficient and do the Nash equilibria at which they stabilized allow us to come close to the system optimal for the Interference Minimization objective?

When our utility functions are not effective anymore, one would also need to design new ones that better manage their relationship with their neighbors and enforce a $\beta < \infty$, in order to stabilize at more balanced solutions. This is also applicable to the Signal Maximization, for which our games do not work at all. A possible way to do this would be to introduce a pricing on the power levels used. And maybe the use of a limited range, allowing for example any power level below the current one and limiting the available power levels above (but not too much), could also help us.

## 11.2 Different Modeling Approaches

In future works, it would surely be interesting to also consider new games. For instance, we observed that the *sequential* choice of the base stations gave us the best results. This could motivate us to try a game where at each iterations, every base stations chooses *simultaneously* the strategy that maximizes its utility. Moreover, we could also try to propose a game where the players are the operators, instead of the base stations.

We also developed other models formulated as node- and edge-deletion problems. These models were not perfectly adapted to the problem we proposed to solve, mainly because they were based on thresholds of disturbance or interference between only pairs of base station. However, we have strong confidence in the fact that they can be useful for other problems or can still be used in our problem. For example, one of their possible use could be to provide us an approximation of the optimal power setting, and this way give us a good starting point for our games and heuristics.

## 11.3 Experimental Research

The libraries and programs that were created during this project, although perfectly running, could benefit from a global optimization of the source code. This could be partially achieved by improving (and maybe limiting) the output offered to the user and by writing specific procedures for the games and the algorithms, instead of a unique fully parameterizable method.

The simulation we performed investigated only a very limited range of possible networks. Instead of having two operators with the same number of base stations, one could consider either unbalanced repartitions of the users among more than two operators, or a unique operator trying to find the system optimal. Other factors to test would be the repartition in space of the base stations, as well as their density and number.

Moreover, the experimental approach would be of great help to study empirically the *upper-bound* and its effect on the network, as well as the impact of other parameters on the behaviors of our games and heuristics.

# 12 Conclusion

The problem we studied in this project is the optimization of the quality of the transmissions in a wireless communication system, through the control of the emission power of some base stations owned by operators, and without explicitly considering the users of that system and the quality of their transmissions.

We approached this problem by different angles. We first designed several models for it, and then concentrated on one for the rest of the project. Then, we proposed a sequential game that could be played in a non-cooperative way by the base stations. Our objective was that by playing this game, the base stations would finally stabilize in a situation which would not be too far from the optimal solution for the network. In order to study the behavior of this game, we developed a very comprehensive program in Java that could simulate it, run some simple heuristics, as well as find the optimal solutions for networks of small sizes.

The models that were formulated as either edge- or node-deletion problems proved to be difficult to apply in real-life situations. For this reason, we eventually settled for a more standard one that had the advantage of being much more realistic and applicable. Then, we studied the properties of this latter model and the associated game, under some special conditions, through a set of simulations on two small networks: a random and a structured one. We obtained that almost all our utility functions had a unique Nash equilibrium, where everybody is emitting at the maximum allowed power. This point was most of the time reached by our games, and as it is also the optimal value for the objective function *Interference Minimization*, we managed for this objective to reach the system optimal every time the games stabilized. In the case of the other objective, we observed that the optimal solution was composed of base stations emitting at either maximum or minimum power, and hence the games and the algorithms were only purely accidentally passing by good solutions, but never stopped there on their way to the maximum power setting for everybody.

Some further research on this *Power Control* problem can be done either theoretically or experimentally. For instance, one could study mathematically some of the properties of the model, in particular the *open questions* proposed and the effect of $\beta$, discussed at the end of this report. Another very important aspect, as we have seen, is to propose new utility functions that better manage to guide the base stations toward the various optimal solution, possibly by introducing a pricing on the power level used. Finally, one could also fully take advantage of the already developed programs to perform a very thorough set of simulations with many different hypothesis, and study their effect on the solutions we obtain.

# References


**General Power Control and Physical Model**

These articles detail the previous works on the topic of *power control*. The different models used inspired us for our own model.

[1] Kawadia and Kumar, *Principles and Protocols for Power Control in Wireless Ad Hoc Networks*, IEEE Journal on Selected Areas in Communications, 2005.

[2] Värbrand and Yuan, *A Mathematical Programming Approach for Pilot Power Optimization in WCDMA Networks*, Proceedings of ATNAC '03.

[3] Yates, *A Framework for Uplink Power Control in Cellular Radio Systems*, IEEE Journal on Selected Areas in Communications, vol.13 no.7 (1995).

[4] Whitaker and Hurley, *Evolution of Planning for Wireless Communication Systems*, Proceedings of HICSS'03.

**Graph Theory**

These articles describe graph theory results and algorithms applied to radio transmissions (or that we tried to apply, in the case of the last one) that we considered in the early stage of development of our model.

[5] Chamaret, Josselin, Kuonen, Pizarroso, Salas-Manzanedo, Ubeda and D. Wagner *Radio Network Optimization with Maximum Independent Set Search*, Proceedings of IEEE Vehicular Technology Conference '97, 770-774.

[6] Abrams, Könemann, Meyerson, Munagala and Plotkin, *Facility Location with Interference*, manuscript, 2001.

[7] Kuhn, von Rickenbach, Wattenhofer, Welzl and Zollinger, *Interference in Cellular Networks: The Minimum Membership Set Cover Problem*, Proceedings of COCOON 2005.

[8] Mihalis Yannakakais, *Node- and Edge-Deletion NP-Complete Problems*, Proceeding of the $10^{th}$ Annual ACM Symposium on theory of computing, 253-264, 1978.

### Game Theory

These articles use game theory to study the behavior of problems related to wireless communication systems.


[9] Márk Félegyházi and Jean-Pierre Hubaux, *Wireless Operators in a Shared Spectrum*, technical report, EPFL, 2005.

[10] Lee, Mazumdar and Shroff, *Downlink Power Allocation for Multi-Class CDMA Wireless Networks*, IEEE/ACM Transactions on Networking [online], 2005.

[11] Srivastava *et al.*, *Using Game Theory to Analyze Wireless Ad Hoc Networks*, PhD Qualifier Presentation.

[12] Halldórsson, Halpern, Li and Mirrokni, *On Spectrum Sharing Games*, Proceedings of the PODC 2004, 107-114.

# Index

# A Model Representation

The classes in this section are at the core of the program. They allow to store models as well as compute things such as the distance between two base stations or the Interference over a base station. They are meant to be used (with the classes of appendix B) as a library for other programs, such as the interfaces of appendix C.

## A.1 Class `ModelParameters`

Stores the general parameters of the network, such as: environment constants $\alpha$ and $N_0$, threshold distance, network dimensions, number of power levels $r$ and maximum power level $P_{\max}$. It also has functions for computing the distance and the interference of a base station oven an other one.

## A.2 Class `Network`, `Operator`, `BaseStation`, `IncomingEdge`

Store the fundamental components of the network, such as: the network, operators, base stations and the relationships between them. These classes also define functions for constructing a random network and computing the Interference over a base station.

## A.3 Class `PredefinedNetworkLayout`

Defines specific network layout constructors as subclasses of the `Network` class. The available predefined layouts are (see simulation series on the CD for pictures):

- Pyramidal Network
- Regular and Random Mono-Line Network
- Regular and Random Dual-Line Network
- Random Dual-Layer Network
- Triangular Network

They all construct networks with 10 base stations, except the last one which can also construct networks with 20, 30 and 40 base stations.

# B Algorithms

## B.1 Class `Solution`

Defines a solution object, with information such as the setting and the objective function's value corresponding to it.

## B.2 Class `Algorithm`

The abstract superclass of all algorithms. It defines the general implementation of a game or a heuristic (see also section 8.3). The following abstract methods need to be defined in subclasses, in order to obtain a complete and working algorithm:

- `initializeSettingOfCurrentSol(...)`
- `performNextMove()`
- `stoppingCriteriaIsMet()`

The parameters needed are:

- the network on which to work
- the objective function, either *Interference Minimization* or *Signal maximization*
- the *upperbound constraint*, if defined (see section 6.2)
- whether the violation of this constraint makes the solution unfeasible or just less attractive but still feasible
- the initial power setting (=strategy profile)

Moreover, this class encompasses methods for:

- evaluating and comparing solutions
- testing settings and their quality in several different ways.

## B.3 Class `BruteForce`

Performs a brutal search of the best solution in the complete space of solutions. This class is used mainly for checking the efficiency of our heuristics and compare their results to the networks best solution.

## B.4 Class `TabuSearchCore`

This procedure is meant to be *very* flexible and fully parameterizable. It can behave as a *game* as well as a *tabu search* heuristic. Its interesting features are:

- choosing the seed of the random number generator (for reproducibility)
- determining how to choose the next base station
- choosing the utility function
- choosing the range of the strategies around the current one, among which to maximize the utility
- choosing whether to keep a tabu list or not, and in the former case its length

Some other parameters were also implemented, although they were finally not used for our research. Those possibilities are:

- to update the utility of the neighboring base stations at only fixed timesteps, instead of immediately after changing the setting of a base station.
- to compute the utility of a base station, knowing only the interference value at the present base station and the settings of the other base stations from the same operator (but not knowing the power setting of the base stations controlled by other operators)
- to empty the tabu list when a *move* is performed (i.e. when a base station changes its strategy)
- to specify a maximum number of iteration without a move allowed for each operator, instead of only specifying a global maximum number of iterations for the algorithm.

To be able to use this class more easily, four subclasses of it (with some parameters already set) were written in `MultipleRunAlgorithmCore`.

## B.5 `UtilityFunction`

This class stores the following *utility functions* for the `TabuSearchCore` class (see section 7.3 for a detailed description of each function):

- `SINR`
- `base`
- `baseWithFriends`
- `baseWithFriendsScaled`
- `basePlusOperatorNeighbourhood`

## B.6 Class `MultipleRunAlgorithmCore`

Defines 4 procedures as subclasses of `TabuSearchCore` with some `TabuSearchCore` parameters set by default in order to get the expected behaviour (see section 8.5 for more details):

- `RandomSearch`
- `SequenceSearch`
- `GlobalTabuSearch`
- `DistributedTabuSearch`

An extension to the `BruteForce` class, with extra capabilities for outputting results, is also defined in this file.

# C Interfaces

## C.1 Class SharedSpectrumSolver, DrawNetwork

A basic program to test and perform a single run with an algorithm. DrawNetwork is used in SharedSpectrumSolver and MultiRunLauncher to give a visual representation of a network.

## C.2 Class MultiRunLauncher

Allows the user to define, for some custom algorithms, a set of configurations to be repeatedly run on a network. The particular procedures to be tested are defined in MultipleRunAlgorithmCore. The variable parameters are:

- Objective Function: either *Interference Minimization* or *Signal Maximization*
- Utility Function: any of the utility functions defined in UtilityFunction
- Initial Setting: a random setting or any uniform setting
- Range: a *free range* or any symmetrical range
- Tabu List Size: no tabu list or any tabu list size

This interface also manages to create a specific directory for each series of simulation, with the following files:

- copies of some of the java source code specific to this series of simulation
- a log_file.txt with the condensed results of all the performed runs, as well as its Matlab friendly translation summary.txt
- for each run, the details of its usual console output (det_XXXXXXXXX.txt)
- for each run, the evolution of the current solutions value at each iteration (mat_XXXXXXXXX.txt)

This program was used to produce the data sets of our simulations.

## C.3 Class SSS (not included on CD)

This graphical interface is currently under development and will be used for presenting the capabilities of the developed library during the final defense.