ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

DÉPARTEMENT DE MATHÉMATIQUES
Chaire de Recherche Opérationnelle
CH-1015 LAUSANNE

Automatic Management of 802.11 Access Points

Sacha Varone, Frédéric Aviolat, Daniel Rossier

ORWP 04/08
July 2004

Automatic Management of 802.11 Access Points

Sacha Varone* Frédéric Aviolat† Rossier Daniel ‡

Abstract

The automatic configuration of Access Points (APs) is a new subject, since the Wi-Fi technology, which underlies hotspots by a wireless local area network, appears on the world market in 2001. The first market relevance has been in 2002. APs channel assignment at hotspots, and more generally APs configuration and management, has to be done manually, except for very recent APs. In this paper, we intend to partially solve the problem of automatic APs management. The goal to achieve is to have an autonomous system able to perform dynamic channel allocation of WLAN APs, in the context of multiple APs in a restricted area. Moreover, the solution has to be independent from sellers (manufacturers) or owners (generally service providers).

Given a set of APs located nearby each other, the problem to be solved consists in assigning a channel to each AP such that the overall throughput is maximized, or, in other words, such that the overall perturbation is minimized. Moreover, the system has to adapt himself to dynamic variations of the environment, such as the number of associated users, the usage of the APs, and so on.

The solution developed in this paper uses a distributed algorithm to solve the problem. One software agent manages one AP and is able to communicate with its neighbors in order to optimize the global throughput. Tests with different topologies have been done by simulation, as well as some real implementation. These experiments have given good results, even when networks get very dense and have many APs. Comparison with optimal solution on small networks has shown that the performance of the algorithm described in this paper is very close to the optimum.

1 Introduction

The automatic tuning of parameters to the Access Points (APs) is a new subject, since the Wi-Fi technology, which underlies hotspots by a wireless local area network, appears on the market in 2001. The first market relevance has been in 2002 and is now a hype for Swisscom Mobile. Until now, the channel assignment of APs at hotspots, and more generally its setup configuration and management, is done manually ([2, 5]). In this report, we intend to provide a way to tackle the problem of automatic APs management. The goal is to have an autonomous system able to perform dynamic channel allocation of WLAN access points in the context of Hot-Spots, irrespective of who is the service provider or which vendor is used.

This work is based on some previous work done by the author within a team of Swisscom engineers, with a collaboration with engineer school of Fribourg([4, 6, 3, 8]. A patent has been deposit on the process described in this article [7].

2 The problem

Given a set of Access Points located nearby each other, the problem to be solved consists in assigning a channel to each AP such that the overall throughput is maximized, or in other words, such that the overall perturbation rate is minimized.

2.1 Perturbations

The channels that can be used are those associated to the unlicensed 2.4GHz frequency band, as stated by table 1.

There are 14 channels defined by the 802.11b (Wi-Fi) norm. But the countries can restrict the use to only some channels. The availability of channels for some countries is given by table 2.

As the reader can guess, the problem is straightforward optimally solved in Japan!

The overlapping characteristic of this set of channels is the source of the perturbations. Figure 1 shows all the channels and their spectral density (see [1] for details).

*Swisscom Innovations & EPFL - ROSE, Switzerland
†École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
‡Swisscom Innovations, Switzerland

1	2	3	4	5	6	7	8	9	10	11	12	13	14
2.412	2.417	2.422	2.427	2.432	2.437	2.442	2.447	2.452	2.457	2.462	2.467	2.472	2.484

Table 1: Channels and frequencies in the ISM band

Country	Europe (ETSI)	USA (FCC)	France	Japan
Channels	1-13	1-11	10-13	14[1]

Table 2: Available channels by country

Among all channels, only 3 are non overlapping. For example, channels 1, 6 and 11 can be used as non-overlapping ones in the USA.

Experiments have been made to measure the perturbation induced on 2 APs, depending on the difference between their assigned channel numbers (channel distance). These measures, translated into relative error rate are given in table 3. These values are measures in situations where interference is maximal and will be referred to as *theoretical perturbation*.

Channel dist.	0	1	2	3	4	5	6	7	8	9	10	11	12
Perturbation	0.37	1.0	0.56	0.3	0.16	0.11	0.08	0.06	0.04	0.03	0.02	0.01	0.005

Table 3: Theoretical perturbation rate depending on the distance between two channels

Access Points have a certain emission range. Thus, an AP does not interfere in the same way over all the APs in its neighborhood. Without obstacle between two APs, the quantity of generated perturbation depends on their inter-distance. For example, if AP 1 is 5 meters away from AP 2 and 95 meters away from AP 3, the interferences between AP 1 and AP 2 are stronger than the interferences between AP 1 and AP 3.

2.2 The model

2.2.1 Network

Let's consider a set of nearby located APs, such that some ranges are overlapping. This will be modelled by an undirected graph $G = (V, E)$, where the set of vertices V is the set of APs, and the set of edges E represents the Virtual Interference Links (VIL), as described in [6].

As an AP does not interfere in the same way over all the APs in its neighborhood, a weight function w is associated to the edges, so that this function w represents the quantity of interferences that two APs can have. The quantity expressed by w represents a percentage of the perturbation rate measured in the lab, as explained in [4]. Therefore the range of its values goes from 0 to 1. For example, if the perturbation rate generated by AP 1, with channel 1 and AP 2, with channel 4, is referenced as 0.4 then two APs with the same difference of channels (here $4 - 1 = 3$) and a perturbation rate of 0.2 are associated with the value 0.5 for the weight function.

2.2.2 Activity

To estimate the activity of an AP mainly three parameters are used:

1. The usage rate.
   This quantifies the number of data frames sent or received by the AP over a given period of time.
2. The error rate.
   This quantifies the number of incorrect data frames in a given period of time.
3. The association rate.
   This quantifies the number of associations.

The activity of an AP i is then defined as follows:

$$\text{Activity}_i = \frac{a\,\text{UsageRate}_i + b\,\text{ErrorRate}_i + c\,\text{AssociationRate}_i}{a + b + c} \tag{1}$$

The idea behind this choice of parameters is the following: it is more important to get less perturbations on heavily used APs, than on lightly used APs.

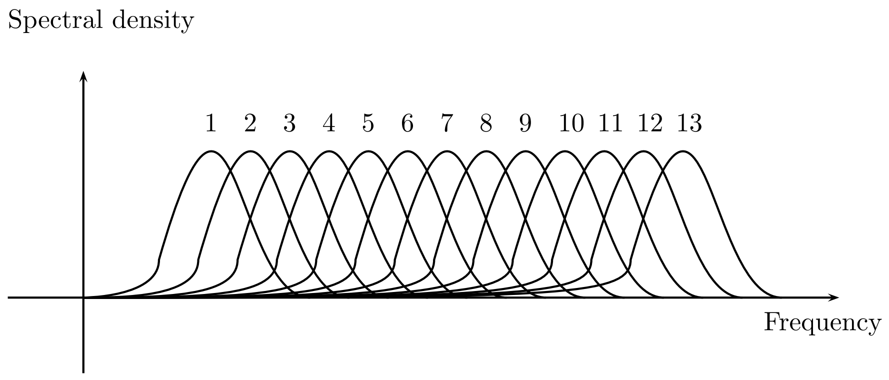


Figure 1: Spectral density of all channels

2.2.3 Variables

The variables of the optimization problem are the channels that have to be assigned to the APs. They will be denoted x_i for each vertex $i \in V$. The admissible values are discrete and depend on the allowed channels in any particular country. For example, in Europe, all x_i must be in the set $\{1, 2, \ldots, 13\}$. The vector of all x_i's is denoted by $\mathbf{x}$.

2.2.4 Objective function

In each optimization problem the goal to achieve is given by a function called the objective function. Solving an optimization problem consists in the search for admissible solutions that optimize the objective function.

We use the following notations:

- w_{ij} is the weight assigned to the edge between APs i and j
- $N(i)$ is the set of neighbors to AP i
- $P(x_i, x_j)$ is the theoretical perturbation rate between channels x_i and x_j, as shown in table 3

Let $\mathbf{x}$ be a channel affectation for AP i and its neighbors:

$$\mathbf{x_j} = \begin{cases} \text{Channel of AP } i \text{ if } j = i \\ \text{Channel of neighbor } j \text{ else} \end{cases}$$

The perturbation of an AP i is expressed as:

$$\text{pert}_i(\mathbf{x}) = \sum_{j \in N(i)} w_{ij}\, P(x_i, x_j) \tag{2}$$

Each agent has to select a channel, taking into account its environment. Therefore, the agent must not only take care to improve himself, but it must also consider its neighbors. We consider two kinds of neighbors:

1. those belonging to the same operator
2. those belonging to competitor operators

and introduce the following notations:

- $N^+(i)$ for the set of neighbors to AP i belonging to the same operator
- $N^-(i)$ for the set of neighbors to AP i belonging to another operator

The objective for AP i is defined as the score function:

$$\begin{aligned} score_i(\mathbf{x}) &= \alpha\, \text{Activity}_i\, \text{pert}_i(\mathbf{x}) \\ &+ \beta \sum_{j \in N^+(i)} \text{Activity}_j\, w_{ij}\, P(x_i, x_j) \\ &+ \gamma \sum_{j \in N^-(i)} \text{Activity}_j\, w_{ij}\, P(x_i, x_j) \end{aligned} \tag{3}$$

where α, β, γ are weights used to balance the importance of the neighbors and the AP itself. Numerical values assigned to α, β and γ defines the strategy. Several strategies have been defined in [8]. For example, γ with a negative value indicates an aggressive competitors' strategy.

3 The solution

This problem belongs to a class of difficult problems, in the sense that the search for an optimal solution takes a much too long time. It is therefore solved by means of heuristics, so that a ”good” solution is expected to be found in a reasonable amount of time.

The developed algorithm, called RME for Resource Management Engine, is based on the concept of autonomous agents: each AP owns an agent that optimizes itself. We describe in the next section the algorithm that is run by each agent, as well as the problematic associated to this approach.

3.1 Agent's algorithm

An agent optimizes itself by trying each possible channel, evaluating it on a score function and choosing the channel that provides the best score.

3.2 Auto-referential equation

In the computation of the score function the activity of the AP and all its neighbors is needed. The activity depends in particular on the error rate, which itself depends on the channel affectation of its neighbors. Since both the activity of the AP and that of its neighbors are calculated, this implies to know the states of all APs in the network! Such a situation would generate a lot of traffic in the network, which is not suitable. Therefore, the error rate is given by an approximation.

The theoretical error rate of AP i is computed as the weighted average of the perturbations induced by its neighbors.

$$\text{theorErrorRate}_i(\mathbf{x}) = \frac{\sum_{j\in N(i)} w_{ij} P(x_i, x_j)}{\sum_{j\in N(i)} w_{ij}} \tag{4}$$

In case of a change in the channel affectation the new error rate e'_i of AP i is approximate as:

$$e'_i = \text{ErrorRate}_i - \text{theorErrorRate}_i(\mathbf{x}) + \text{theorErrorRate}_i(\mathbf{x}') \tag{5}$$

Then, the theoretical activity of an AP i can be computed in the same way as the actual activity, by replacing the actual error rate by its approximation:

$$\text{theorActivity}_i = \frac{a\,\text{UsageRate}_i + b\,e'_i + c\,\text{AssociationRate}_i}{a+b+c} \tag{6}$$

The score is then computed using the theoretical activity rate:

$$\begin{aligned} \text{score}'_i(\mathbf{x}') &= \alpha\,\text{theorActivity}_i\,\text{pert}_i(\mathbf{x}') \\ &+ \beta \sum_{j\in N^+(i)} \text{theorActivity}_j\,w_{ij}\,P(x_i,x_j) \\ &+ \gamma \sum_{j\in N^-(i)} \text{theorActivity}_j\,w_{ij}\,P(x_i,x_j) \end{aligned} \tag{7}$$

3.3 Activation of the agent

The question to be answered once the algorithm is implemented is the following: “When the optimization algorithm must be run ?” An answer is to launch the optimization process each time one of the below situations occurs:

- At the installation of the AP;
  this allows an automatic selection of the appropriate channel at start-up
- A message from a neighbor is received;
  this is to react to a channel change by a neighbor
- A threshold value on the error rate is reached;
  this is to take into account an increase in the error rate, meaning that something not detected previously has happened.

3.4 How to avoid cycling?

One of the problems when dealing with autonomous agents that interact with each other is that some cycling process can occur. To terminate cycling processes, or, better, to avoid them, different approaches have been studied.

The two main cycling processes that can occur, and the chosen solutions are defined in the next sections.

3.4.1 Simultaneous communication

The simultaneous communication is a communication conflict between two neighboring APs, say AP 1 and AP 2. They both run their optimization process at the same time. Therefore, as a result, they could have to change their channel. Once this has been done, the information of channel change is sent to each other simultaneously. In general another run of their optimization process is generated. The same situation as previously appears. This process could go on and never ends up. Therefore, a solution is needed to avoid this simultaneous communication behavior.

A solution is simply to wait a random amount of time before the transmission of information or before the launch of optimization process. The specialist reader can remark that this solution is related to the Aloha MAC layer protocol (contention-based protocol: users transmit a packet when they have data to send. If multiple users transmit at the same time a collision occurs and the packets must be retransmitted according to the random waiting time rule).

3.4.2 Cycling process

A cycling process can occur if the channel allocation is changed several times until a previous state is reached once again. Figure 2 represents this situation for four APs.

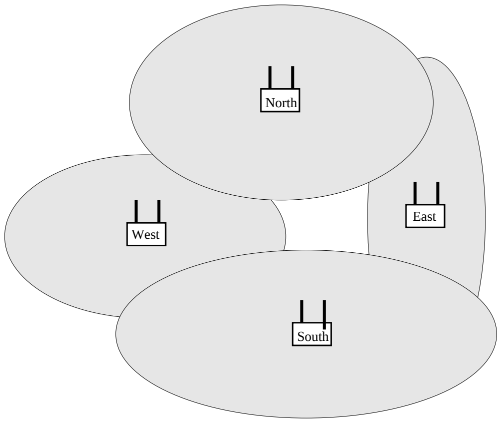


Figure 2: Wireless coverage situation for a set of 4 APS

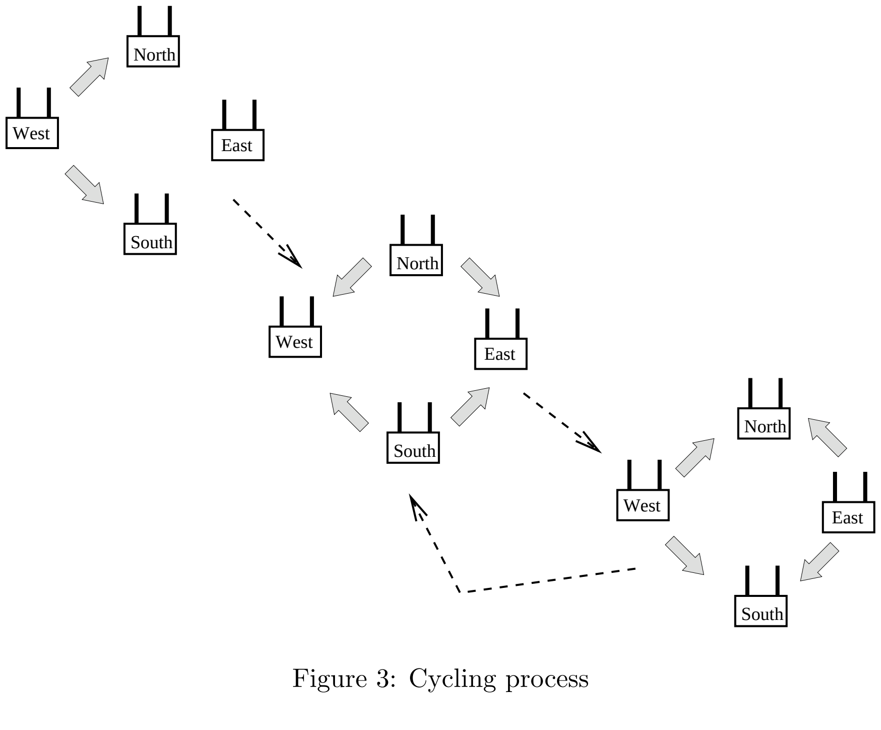


Figure 3: Cycling process

Figure (3) illustrates the cycling process that is explained below. First the AP to the right, say AP west, launches its optimization process, change its channel and informs its neighbors. Then AP north and AP south also launch their optimization process, change their channel and inform their neighbors, AP west and AP east. The situation is now that one represented by the middle drawing. The process is going on and after optimization and channel changes, the two AP east and west inform their neighbors north and south. This situation is represented by the right drawing. Once again, AP north and south launch their optimization process, change their channel and inform their neighbors. The situation is now the same as previously, i.e. the drawing in the middle. Therefore a cycling process has occurred.

Several means have been found to detect such a cycling process. The first one is to maintain an historical list of previous situations (states). Each AP maintains such a list, which contains the k previous channel allocations of the AP and its neighbors. If the best channel found by the agent as well as all its neighbors' channels are the same as one of the k memorized situations, then the channel is not changed.

Another way to avoid cycling is based on the assumption that it is not always very profitable to change a channel. The idea is to compare the profit of a channel change to a threshold value, and to accept a channel change only if the profit exceeds the threshold value.

4 The results

4.1 Simulation

The experiments are performed on a system simulating the agents on a sequential machine. Thus the problem of two simultaneous agents running the algorithm can not appear. Simplifications are also made in the communications between the agents. A list is used to contain the APs which need an optimization process. All APs are put in the list at the initialisation step. Once an AP changes its channel, all its neighbors are added to the list, but in such a way that no AP appears twice in the list. Thus only one optimization process occurs if several neighbors change their channels. This is the same behavior as in the "real" implementation of RME where an agent clears all its messages after its optimization process.

All experiments are run with the numerical values shown in table 4 for parameters a, b and c, which appear in the computation of the activity (equation (1)).

a	b	c
1	3	2

Table 4: Weight values to define the activity of an AP

The strategy parameters, used in equation (3), have the numerical values shown by table 5. This means no competitors and an equally importance between oneself and one's neighbors.

α	β	γ
0.5	0.5	0

Table 5: Weightings used for strategy

4.2 Tests on random networks

The first set of tests uses random networks, with the number of vertices ranging from 2 to 50, plus networks with 100 vertices. The weights w_{ij} on the edges are set either to 1 or using a uniform law $\mathcal{U}(0, 1)$. Networks with different densities are generated, either with a bound on the density, or with a bound on the maximum degree of the vertices. These generates 4 series of experiments, with different network sizes.

For each configuration, 400 graphs are generated and the algorithm is applied on them. The number of associations is randomly set using a uniform law $\mathcal{U}(0, 32)$, and the usage rate is randomly set with the law $\mathcal{U}(0, 1)$. The error rate is computed using the theoretical value of equation (4).

4.2.1 Prevention of cycling

Table 6 shows the number of graphs on which a cycling process has been detected, using an history. Over 980,000 networks, cycling appeared only 4273 times, which is about 0.4%. Thanks to the historical list of previous states, infinite cycling is avoided. This shows that cycling appears very rarely, but in case of appearance, the algorithm still behaves correctly.

Cycling cases mostly appears only once during the simulation. For example, on the simulation series on graphs with all weights set to 1 and max degree bounded, cycling appeared once on 1937 graphs, twice on 84 graphs, three times on 4, four times on 1 graph, and never more than four times.

Experiment	Density	Max degree
Weights	bounded	bounded
random	202 / 215600	1656 / 274400
all = 1	236 / 215600	2179 / 274400

Table 6: Number of graphs with cycling avoided / Total number of graphs

4.2.2 Average global score

The average global score for a set of experiments $\mathcal{S}$ is computed as follows:

$$\overline{\text{score}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \text{score}(s)$$

Figure 4 shows the average global score obtained on the graphs of the 4 series of simulations performed, in function of the number of APs.

Although the score value is used as the objective function, its actual value is not very significant, since this value is not normalized and depends in particular on the number of APs. A more interesting performance indicator is the average error rate.

4.2.3 Average global error rate

Let $G = (V, E)$ be a network.

The average error rate of a solution $\mathbf{x}$ is defined as:

$$\overline{\text{errRate}}(\mathbf{x}) = \frac{1}{|V|} \sum_{i \in V} \text{theorErrorRate}_i(\mathbf{x})$$

where $\text{theorErrorRate}_i(\mathbf{x})$ is defined as in equation (4).

The maximum error rate of a solution $\mathbf{x}$ is defined as:

$$\text{maxErrRate}(\mathbf{x}) = \max_{i \in V} \text{theorErrorRate}_i(\mathbf{x})$$

where $\text{theorErrorRate}_i(\mathbf{x})$ is defined as in equation (4).

The average global error rate on a set $\mathcal{S}$ of experiments is computed as:

$$\overline{\text{errRate}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \overline{\text{errRate}}(s)$$

and the average maximum error rate as:

$$\overline{\text{maxErrRate}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \text{maxErrRate}(s)$$

Figures 5 show the average global error rate in function of the number of APs. As the error rate is a number in $[0, 1]$, this result shows a good performance of our algorithm. Moreover, the maximum errror rate is less than 25% above the average global error rate with a bounding density, but can be up to 75% above the average global error rate with a bounding degree.

4.2.4 Average number of optimization runs

Another interesting performance measure is the number of optimization run, which is the number of times the optimization algorithm is run by any of the agents. Clearly, this number increases with the number of APs. Figure 7 shows the number of optimizations relative to the number of APs.

This result shows that a stable state is achieved using only a few optimisation runs.

4.2.5 Average number of channels' changes

The number of channel's change can be understood as a usability factor. Indeed, this value should be kept small, since a change in the AP's channel may lead to a disconnection with its associated devices. Figure 8 shows this average number of channels' changes.

These results indicates that the number of channels' changes per AP is very small: less than 1.5 changes per AP, even for networks as large as 100 APs. This also means that the algorithm is very efficient and converges quickly to a stable solution.

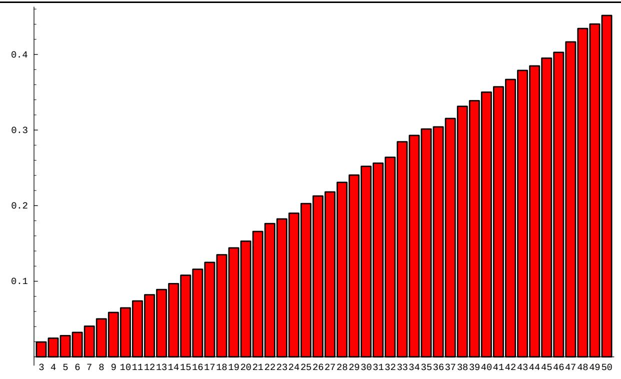
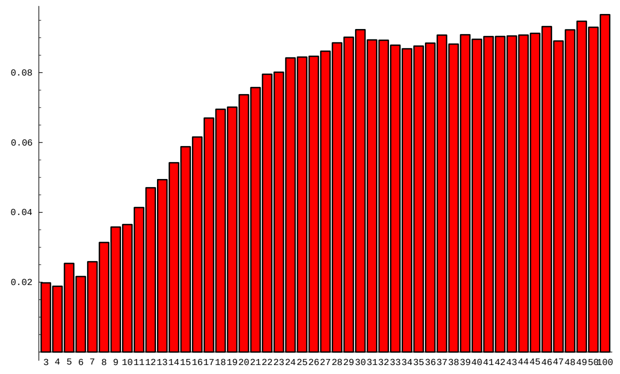
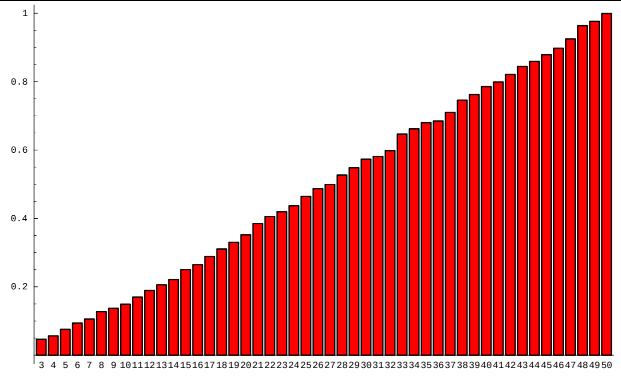
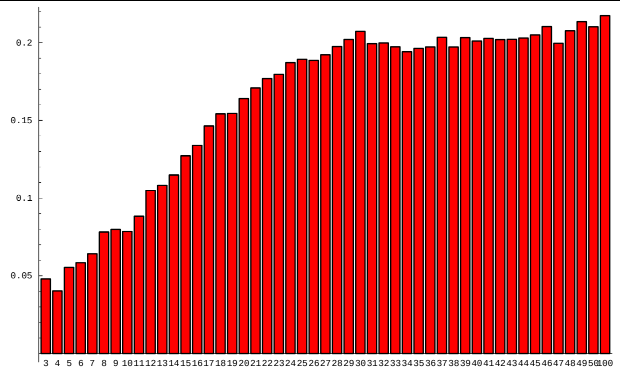
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 4: Average score by APs' number

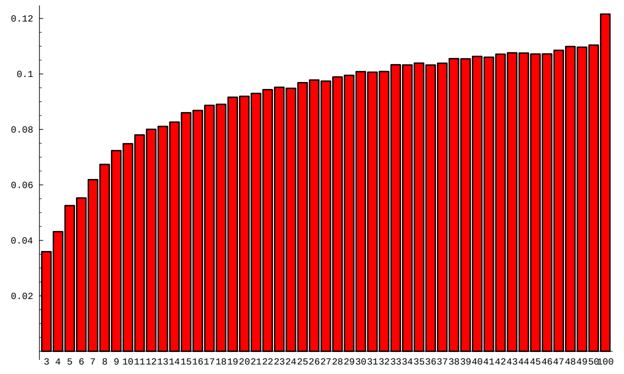
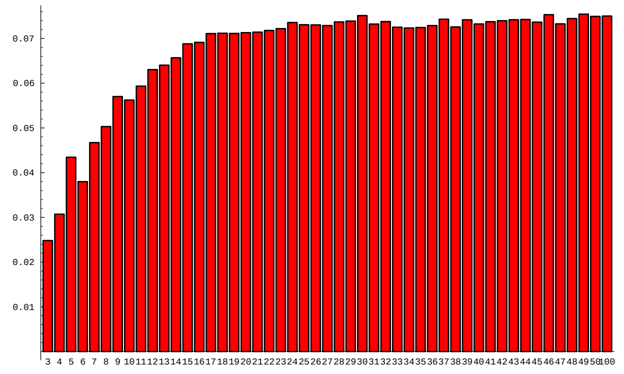
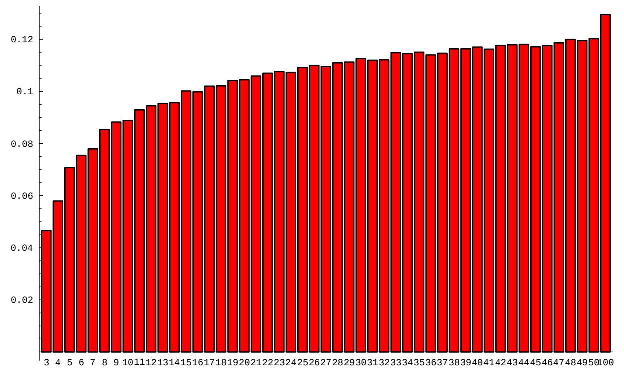
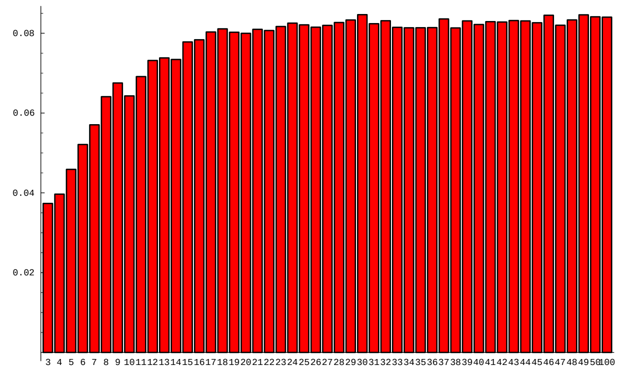
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 5: Average mean error rate by APs' number

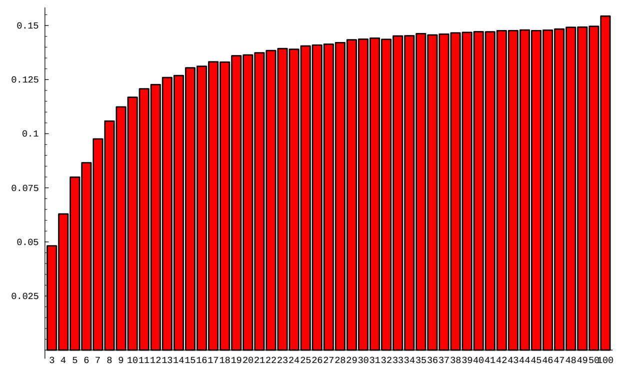
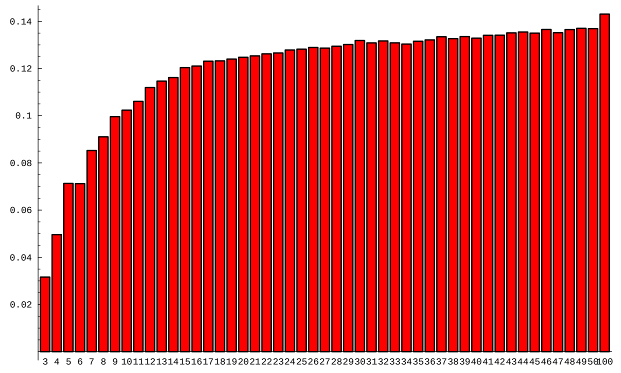
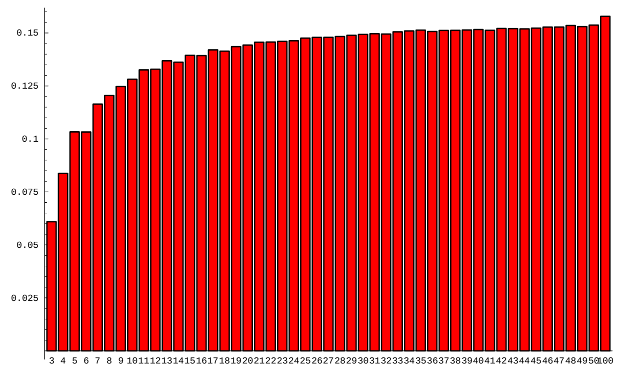
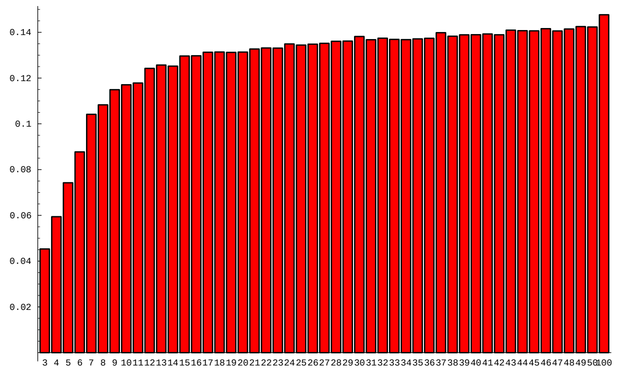
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 6: Average maximum error rate by APs' number

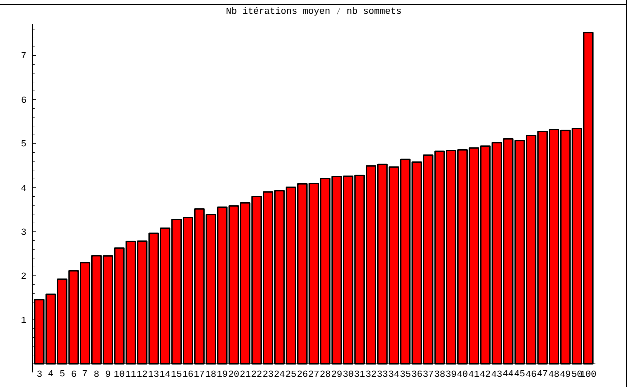
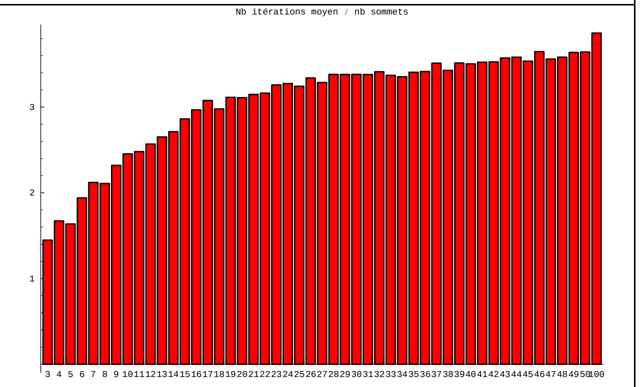
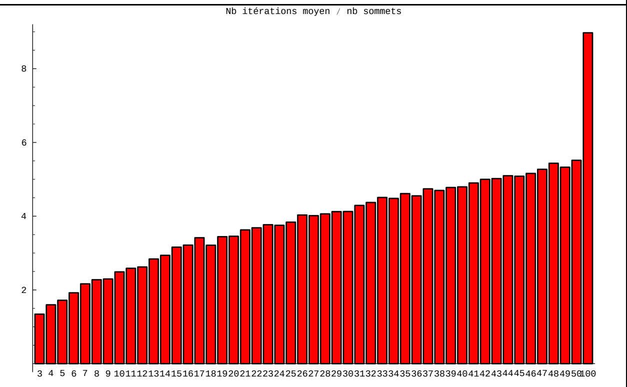
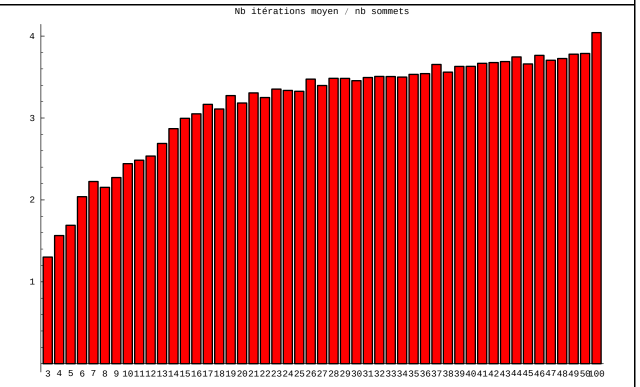
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 7: Average number of optimization runs by APs' number

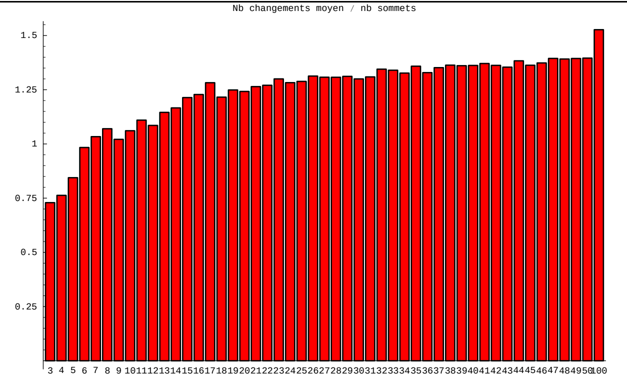
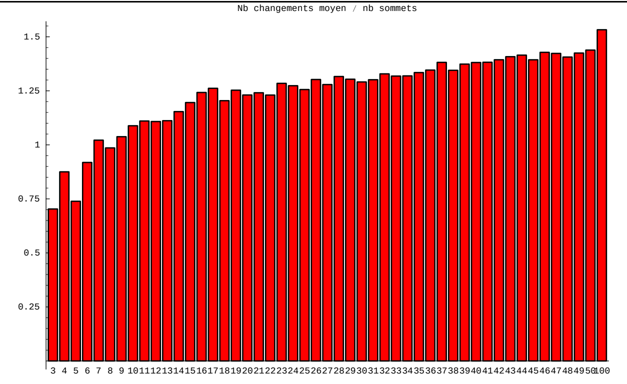
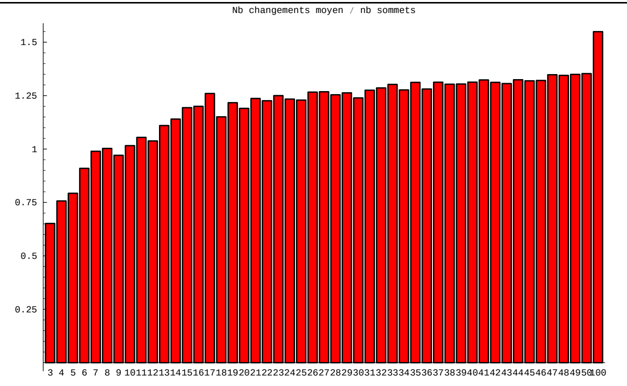
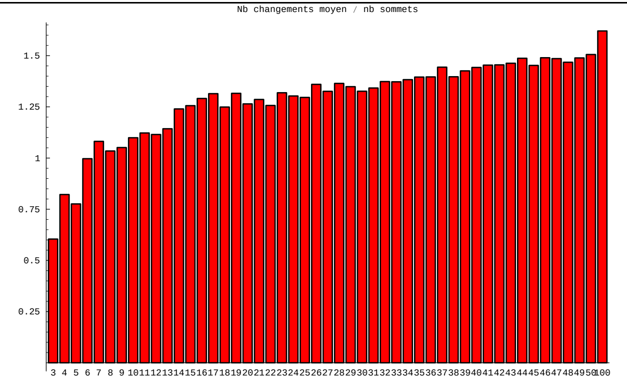
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 8: Average number of channels' changes by APs' number

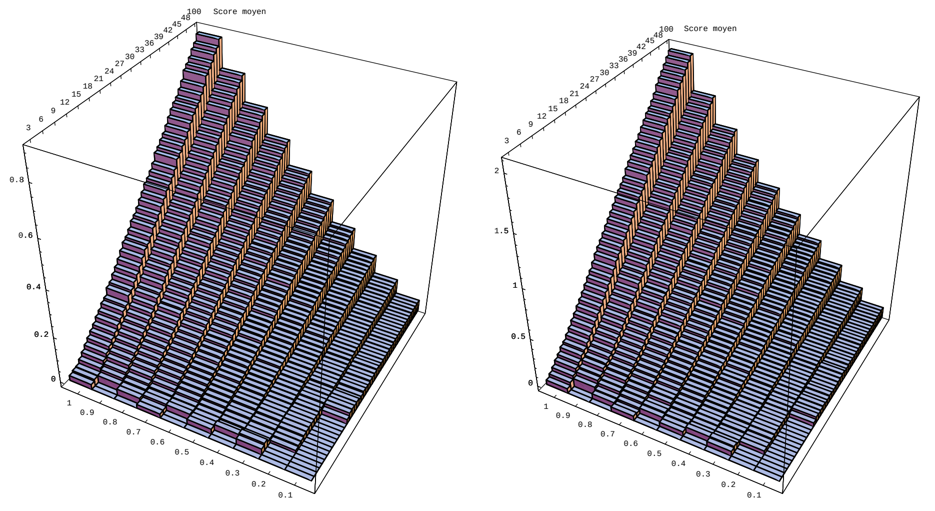


Figure 9: Average score by density and number of APs

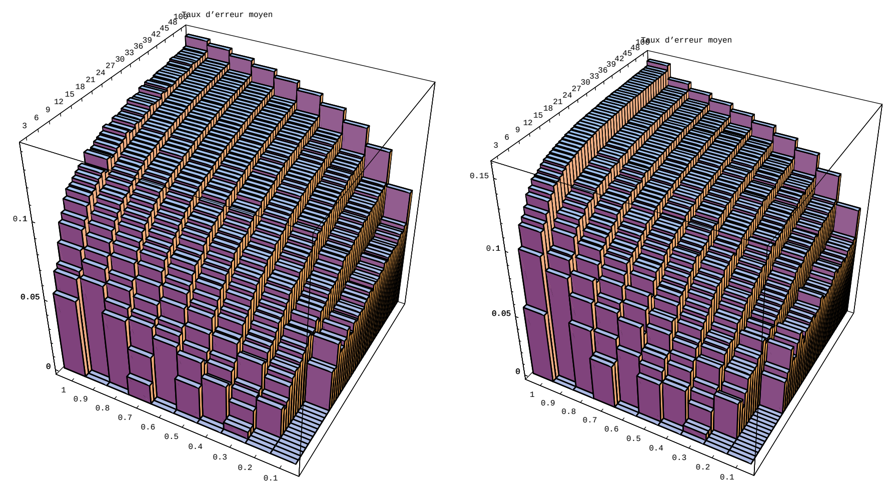


Figure 10: Average error rate by density and number of APs

4.3 Detailed results on random networks

We show below more detailed results on the generated random networks. The influence of the graph's density on the algorithm is put into perspective.

4.3.1 Results by density and number of APs

In the graphs below, the upper axis shows the number of APs in the graph (from 3 to 50 and 100), the lower axis shows the density of the graph (from 0.1 to 1), and the left axis shows some performance measure.

Score and error rate

The score seems to grow linearly with the density of the graph as well as with the number of APs. The average error rates (both average and maximum) are very low for very simple networks, then grow up to a reasonably small value (around 0.15).

Number of optimization runs and number of channel changes

The number of optimization runs per AP and the number of channels' changes per AP don't grow much as the network's density increases. Surprisingly, they are even much smaller when the density is 1. This case could be explained by the fact that, when a network is completely connected, the best solution always contains adjacent vertices that are set to the same channel; this comes from the fact that using the same channel induces less perturbation as using two channels at distance 1 or 2 from each other.

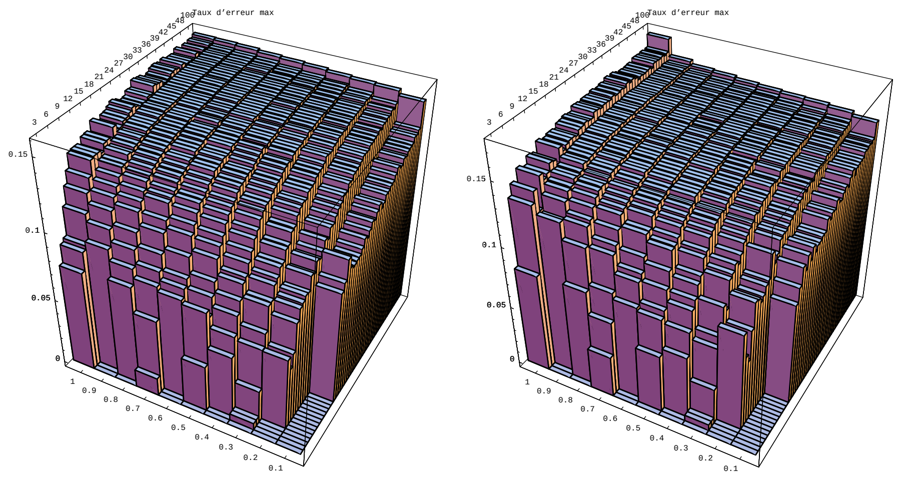


Figure 11: Average maximum error rate by density and number of APs

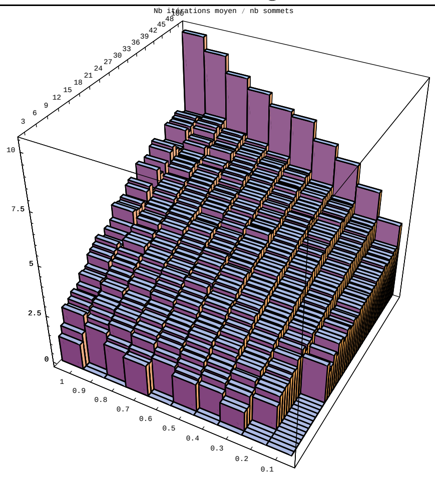
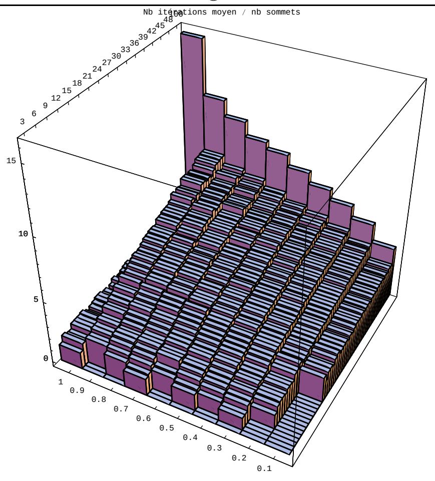
Random weights	All weights = 1
	

Figure 12: Average number of optimization runs by APs' number and density

4.3.2 Results by maximum degree and number of APs

In the graphs below, the upper axis shows the number of APs in the graph (from 3 to 50 and 100), the lower axis shows the maximum degree of the graph (between 2 and 15), and the left axis shows

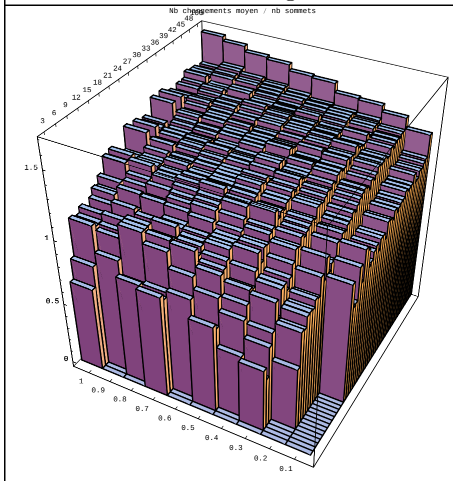
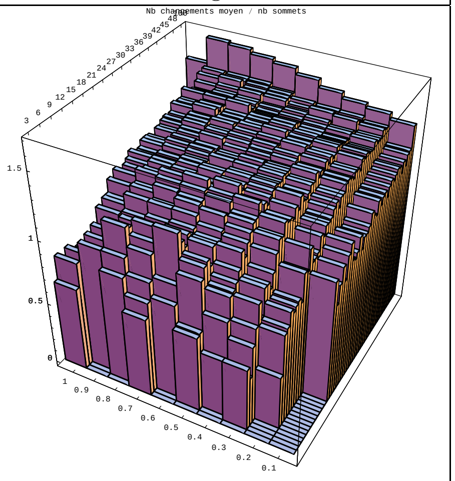
Random weights	All weights = 1
	

Figure 13: Average number of channels' changes by APs'number and density

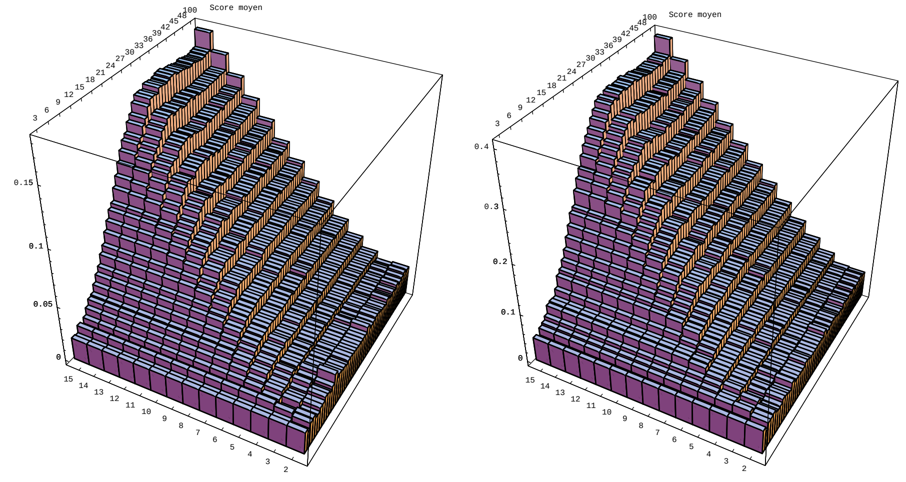


Figure 14: Average score by maximum degree and number of APs

some performance measure.

Score and eror rate

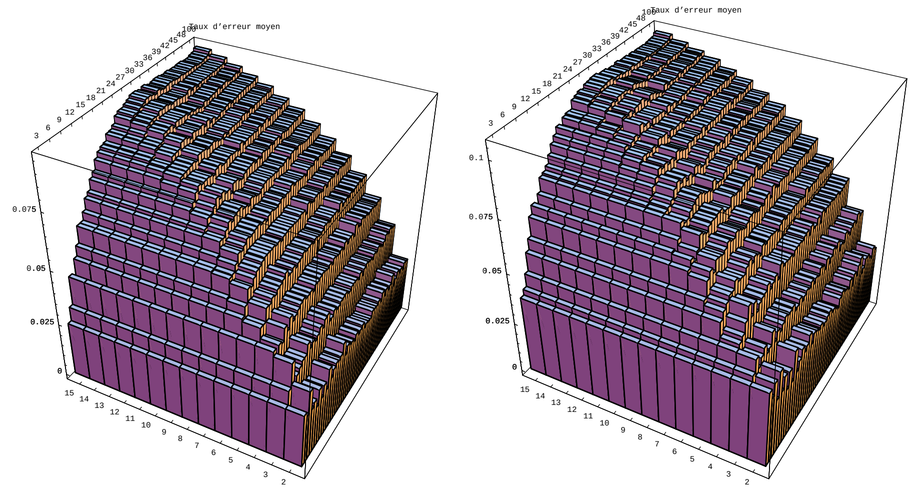


Figure 15: Average error rate by maximum degree and number of APs

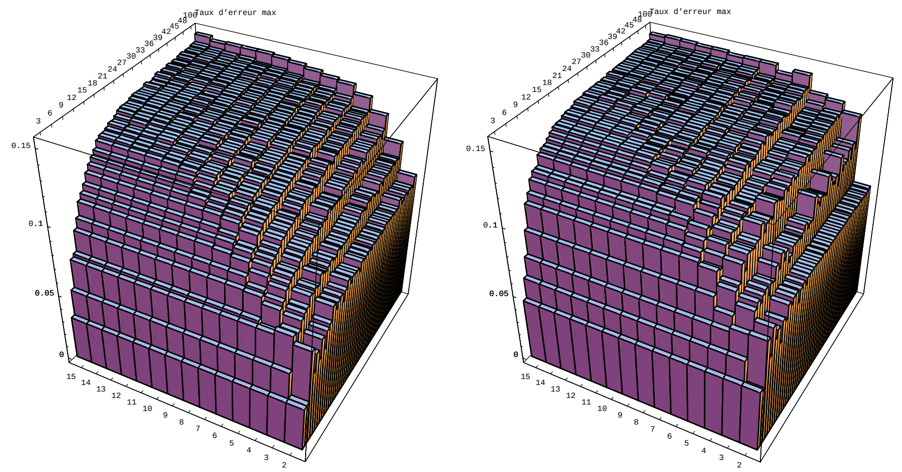


Figure 16: Average maximum error rate by maximum degree and number of APs

Results are similar to the ones where the density is fixed. If the maximum degree is bounded then score and error rates (average and maximum) do no more increase with the number of APs. This is because, above a certain number of vertices, bounding the maximum degree implies a density decrease. This shows that the complexity of the problem depends essentially on the density, rather than on the maximum number of neighbors.

Number of optimization runs and number of channels' changes

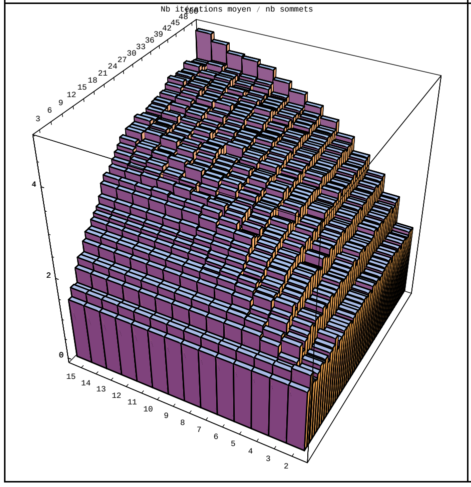
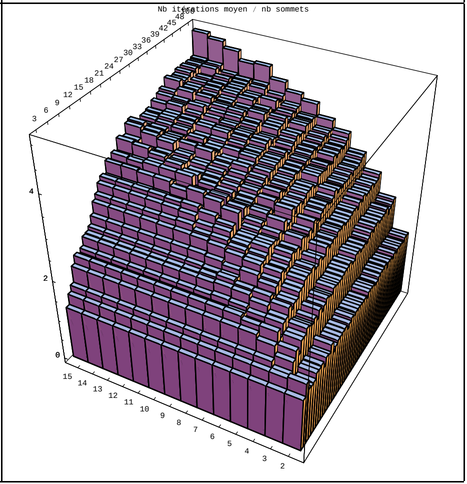
Random weights	All weights = 1
	

Figure 17: Average number of optimization runs by maximum degree and APs' number

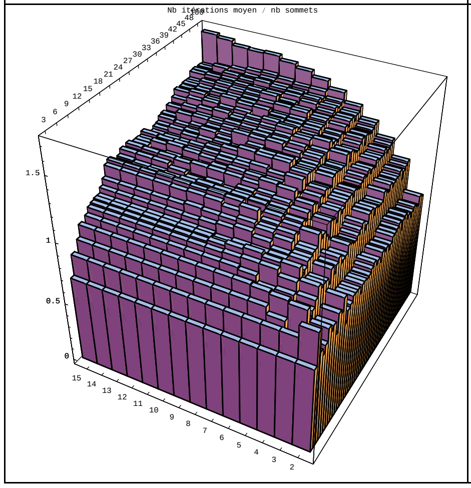
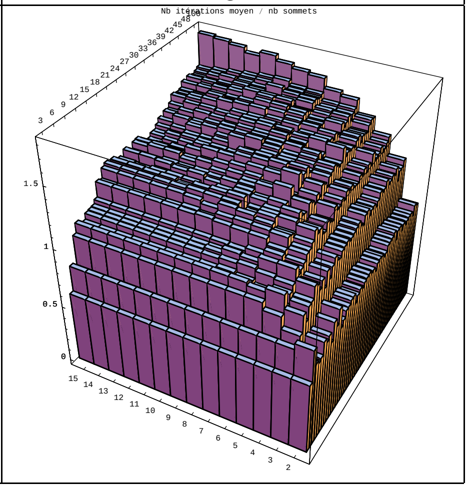
Random weights	All weights = 1
	

Figure 18: Average number of channels' changes by maximum degree and APs' number

The results are similar to those with varying density. Same observations as above can be made concerning bounding the maximum degree.

4.4 Comparison with an optimal solution

For all generated graphs with 5 vertices an optimal solution is also computed using complete enumeration. For larger graphs, this method is too much time-consuming and therefore could not be done in reasonable time.

4.4.1 Average comparisons

Figure 19 shows the average global score. RME algorithm is compared to optimal scores.

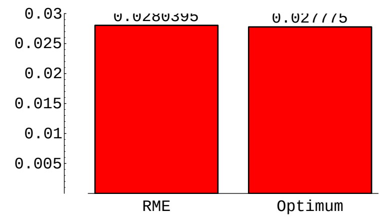
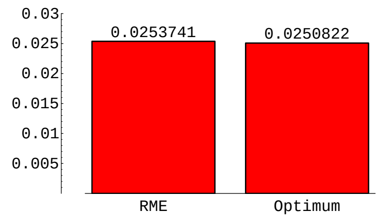
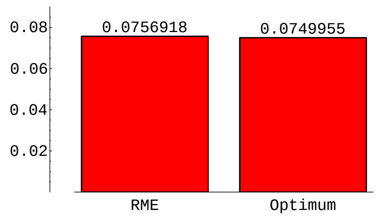
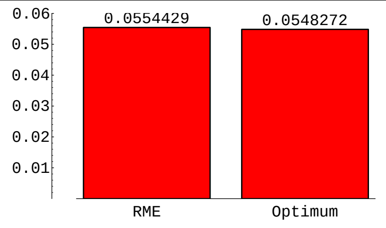
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 19: Average score on graphs with 5 APs, compared with optimum

It is amazingly striking that in average RME is almost as good as an optimal solution. RME is around 1% above the optimum value.

Figures 20 and 21 respectively show the average global error rate and the average worst (maximum) error rate, obtained on the graphs with 5 APs. Results are compared with the average of optimal scores.

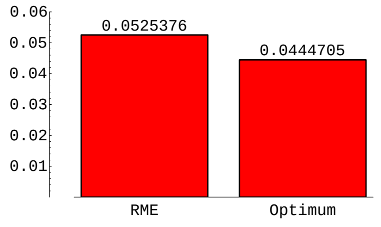
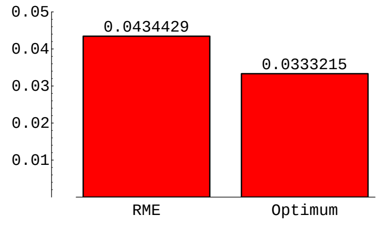
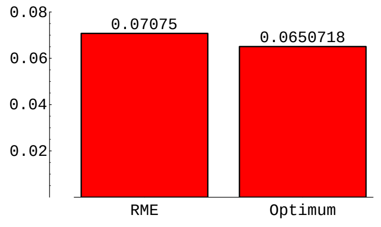
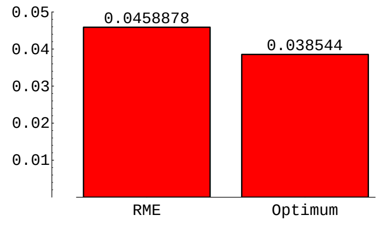
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 20: Average mean error rate on graphs with 5 APs compared with optimum

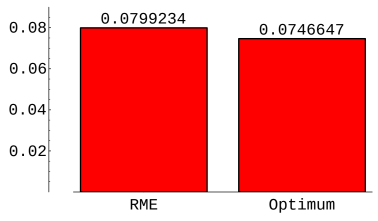
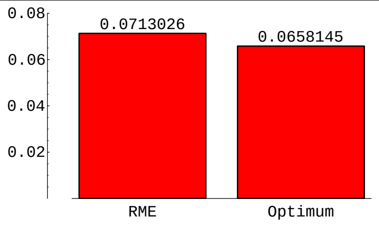
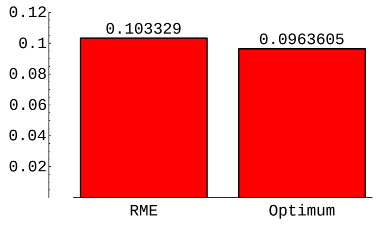
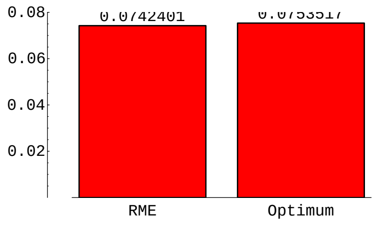
Exper. / Weights	Density bounded	Max degree bounded
random		
all = 1		

Figure 21: Average maximum error rate on graphs with 5 APs compared with optimum

4.4.2 Detailed comparisons by density

In all following graphics, RME values are shown on the left in red, and optimal values are shown on the right in blue.

Figures 22, 23 and 24 respectively show the average score, the average error rate and the average worst (maximum) error rate, depending on the density, obtained on the graphs with 5 APs compared with the average of optimal scores.

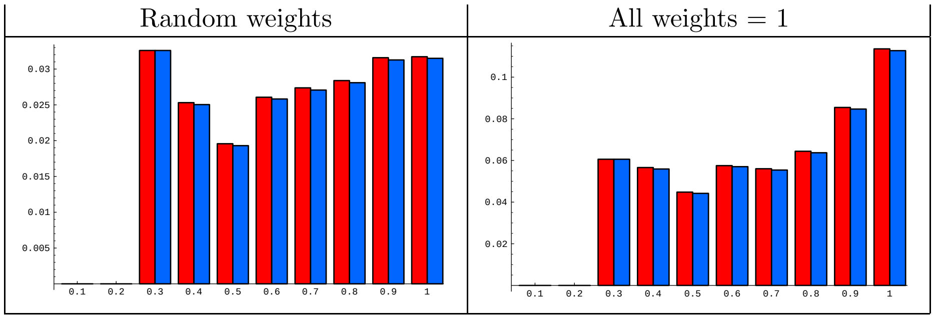


Figure 22: Average score on graphs with 5 APs, by density compared with optimum

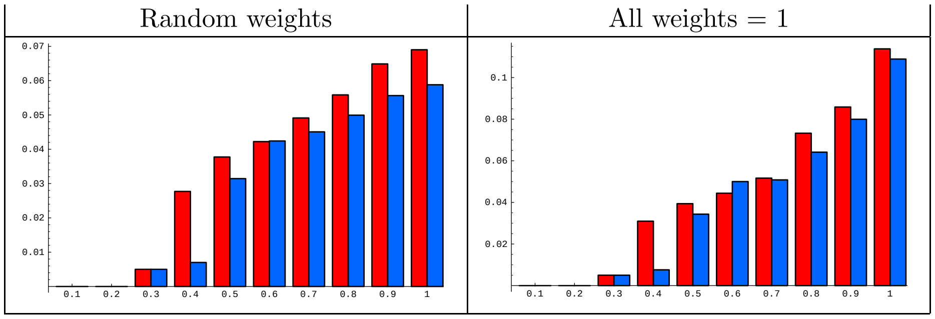


Figure 23: Average error rate on graphs with 5 APs, by density compared with optimum

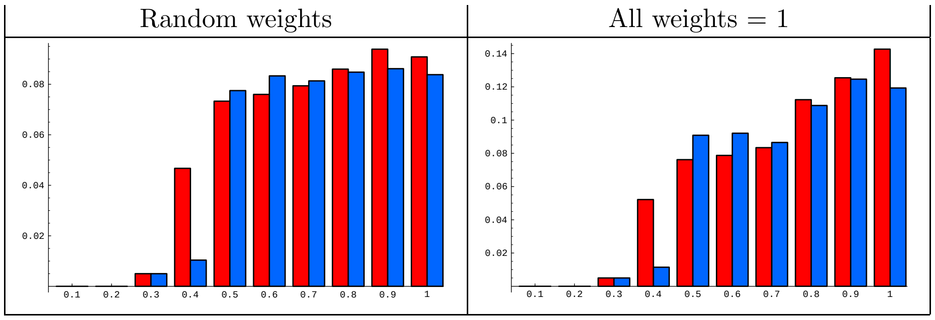


Figure 24: Average maximum error rate on graphs with 5 APs by density compared with optimum

Results are similar to the average ones, with RME being slightly above optimum. It even happens, for some densities, that the maximum error rate is worse in the optimum solution than in the RME one. This is possible, since the objective function does not optimize this parameter in particular.

4.4.3 Detailed comparisons by maximum degree

Figures 25, 26 and 27 respectively show the average score, the average error rate and the average worst (maximum) error rate, depending on the maximum degree of the graph. Results are compared with the average of optimal scores.

Results are similar to the average ones, with RME being slightly above optimum. Again, it sometimes happens that the maximum error rate is worse in the optimum solution than in the RME one. This is possible, since the objective function does not optimize this parameter in particular.

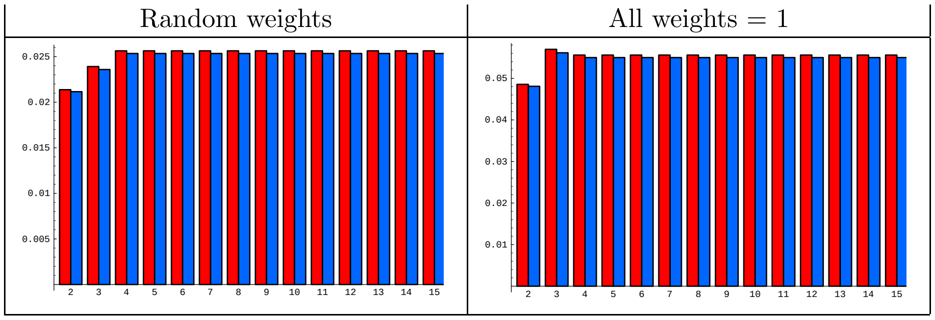


Figure 25: Average score on graphs with 5 APs, by max degree compared with optimum

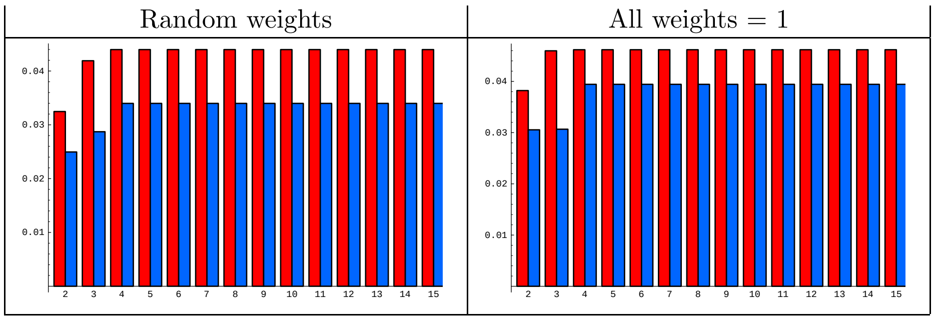


Figure 26: Average error rate on graphs with 5 APs by max degree compared with optimum

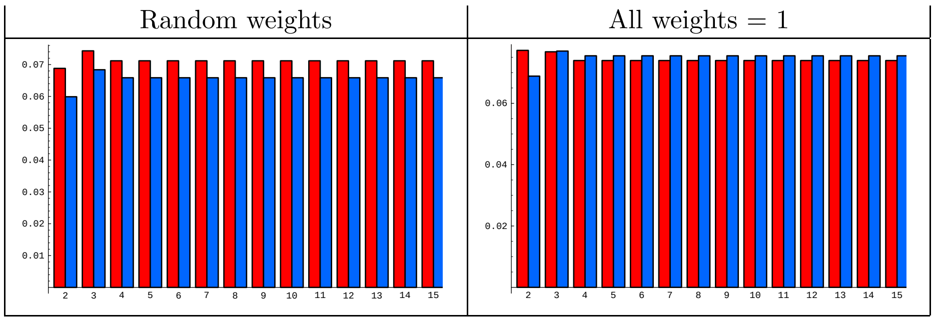


Figure 27: Average maximum error rate on graphs with 5 APs, by max degree compared with optimum

4.5 Tests on the EIF network

This set of tests are based on real-world network of access points, as installed at EIF[2]. The topology of the network is shown in Figure 28. The goal of these tests is to evaluate the robustness of the

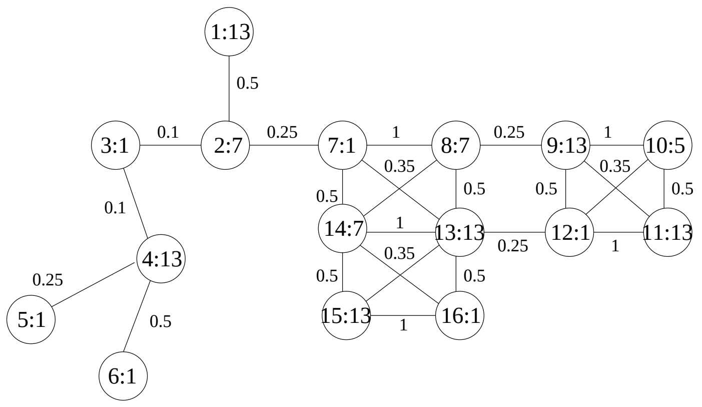


Figure 28: Network of APs installed at EIF, with reference solution

algorithm in a real-world situation. We chose this network because of its reasonable size (16 APs). It has already been configured, in particular the weights of the virtual links between the APs have been determined.

We chose the following reference set of parameters:

- Association rate: 10 users on all APs;
- Usage rate: 1.0 (full load) on all APs;
- Error rate: computed theoretically, according to equation (4).

An initial solution is set on channel 1: all the APs use the same channel. With these settings, the RME algorithm runs 36 times and outputs the solution shown in table 7 and in the circles of Figure 28; the performance values are

- Mean error rate: 0.062460
- Min error rate: 0.005000
- Max error rate: 0.128333

AP	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Channel	13	7	1	13	1	1	1	7	13	5	13	1	13	7	13	1

Table 7: Solution found with reference parameters on EIF network

4.5.1 Uniform variation of one parameter

In this series of experiments, we keep all parameters to their reference settings, except one that varies uniformly on all APs.

- Association rate varies with values in the following set: $\{0, 1, 2, 4, 6, 8, 10, 12, 16, 20\}$.
- Usage rate varies with values in the following set: $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$.
- Instead of using the theoretical error rate its value varies from 0.0 to 1.0 by 0.1 steps.

For each experiment the algorithm is run from an initial solution, where all the APs use channel 1. For all of these tests, we obtained after 36 optimization runs the same solutions as the reference one. Of course, the values of the performance indicators change because they depend on the parameters. This shows that the algorithm is not sensible to uniform variations of the parameters. That means that, if any parameter is read incorrectly in the same way on all access points, the behavior of the algorithm will not be affected.

[2]Ecole d'Ingénieurs de Fribourg, Switzerland

4.5.2 Variation of the error rate on one AP

The error rate on one AP is set to an erroneous value. It varies from 0.0 to 1.0 by 0.1 steps. We made this experiment on each of the following APs: #11, #12, #13, #14. Those APs have the highest number of links in the network and are therefore good candidates for our tests.

The same solution as the reference one is obtained after 36 optimization runs, for each of these tests. The values of the performance indicators change since they depend on the error rate parameter. This shows that the algorithm is robust to an erroneous value of the error rate of one AP, even if it is very far from the correct value.

4.5.3 Simulation of moving users

In this last experiment moving users in the network are simulated. We start from the reference configuration, with 20 users on AP #1, and 2 users on each of the other ones. The RME algorithm gives the reference solution after 36 optimization runs.

Then 18 users are moved from AP #1 to AP #2, thus having now 2 users on AP #1 and 20 users on AP #2. And then on, until 20 users are on AP #16.

Results are shown in table 8.

20 on	#iter	Mean score	Min score	Max score	Mean error	Min error	Max error	Solution
1	36	0.033925	0.000280	0.102450	0.065399	0.005000	0.128333	*
2	16	0.028751	0.000275	0.076241	0.062460	0.005000	0.128333	*
3	16	0.027439	0.000275	0.076241	0.062460	0.005000	0.128333	*
4	16	0.027394	0.000380	0.076241	0.062460	0.005000	0.128333	*
5	16	0.027363	0.000380	0.076241	0.062460	0.005000	0.128333	*
6	16	0.027376	0.000275	0.076241	0.062460	0.005000	0.128333	*
7	16	0.028844	0.000275	0.079616	0.062460	0.005000	0.128333	*
8	16	0.030192	0.000275	0.087742	0.062460	0.005000	0.128333	*
9	21	0.028704	0.000275	0.076241	0.062460	0.005000	0.128333	**
10	16	0.028955	0.000275	0.076241	0.062460	0.005000	0.128333	**
11	16	0.028549	0.000275	0.076241	0.062460	0.005000	0.128333	**
12	18	0.028512	0.000275	0.076241	0.062460	0.005000	0.128333	*
13	16	0.030039	0.000275	0.082991	0.062460	0.005000	0.128333	*
14	16	0.030698	0.000275	0.103030	0.062460	0.005000	0.128333	*
15	16	0.029190	0.000275	0.079616	0.062460	0.005000	0.128333	*
16	16	0.027724	0.000275	0.078604	0.062460	0.005000	0.128333	*

Table 8: Results of moving users on EIF network

The solution indicated by (*) is the reference solution of table 7, and the solution indicated by (**) is the solution shown in table 9.

AP	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Channel	13	7	1	13	1	1	1	7	13	**1**	**9**	1	13	7	13	1

Table 9: Other solution found with moving users on EIF network

Results again show that the algorithm is very stable and very fast: if a same solution is found then there is no channels change (there are 16 optimization runs altogether, one by each of the APs). If another solution is found, few iterations (i.e. few channel changes) are needed.

We also observe that the RME algorithm takes well into account the association rate, adapting the channels to situations where the throughput of the most used AP has to be maximized.

5 Conclusion and perspectives

The objective of this project was to develop an algorithm to assign channels to Access Points so that the perturbations between neighboring APs is minimized. We designed an objective function (score) taking into account the perturbation as well as the relative usage rate of the APs and the number of its associated stations. We choose to use an agent-based algorithm, able to adapt each AP channels'allocation to any given situation.

The experiments we made show that the RME algorithm gives very good results, even when networks get very dense and have many vertices. In real-world networks, the maximum number of neighbors of an AP (the maximum degree of a vertex in the network) is usually bounded; in this case, the score and the error rates are also bounded. Comparisons with the optimal solution on small networks have shown that the performance of the RME algorithm is very close to the optimum.

The algorithm runs fast and converges quickly, without changing the AP's channels too many times. It is robust to homogenous erroneous measures and to erroneous measures on a single AP. It takes into account the relative importance of the APs, as the "moving users" experiment has shown.

From this positive experience, there is much hope that such a system can improve significantly the quality of service in hotspots.

References

[1] M. Burton. Channel overlap calculations for 802.11b networks. white paper, 2002. Cirond.

[2] G. Mateus et al. Optimal network design for wireless. *Annals of Operations Research*, 106:331–345, 2001.

[3] Fiorenzo Gamba, Jean-Frédéric Wagen, and Daniel Rossier. Towards adaptive wlan frequency management using intelligent agents. In *ADHOC-NOW'03*, Montreal, Canada, October 2003. 2nd International Conference on Ad-hoc Networks and Wireless.

[4] Vincenzo Inguscio and Eric Marchon. Autonomous wireless LAN management. Master's thesis, EIF, 2002.

[5] D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *Proceeding of MOBICOM'02*, pages 107–118, September 2002.

[6] Ferran Moreno Blanca and Daniel Rossier. Wlan resource management - network management of wlan networks. Technical report, Swisscom Innovations, March 2003.

[7] Daniel Rossier, Sacha Varone, Jean-Frédéric Wagen, Fiorenzo Gamba, Vincenzo Inguscio, and Eric Marchon. System für die dynamische zuweisung von trägerfrequenzen zu zugriffspunkten eines lokalen funknetzes. Patent 03405356.1, May 2003.

[8] Sacha Varone. WLAN resource management – Network management of WLAN networks – Deliverable 2. Technical report, Swisscom Innovations, 2003.