Málaga, Diciembre 2008

Informe Ejecutivo

TÍTULO: AFP-1.1: Algoritmos greedy para el problema AFP

RESUMEN: Este entregable está dedicado a la presentación y análisis de varios algoritmos greedy para resolver el problema de la asignación automática de frecuencias (AFP) que estamos considerando en el projecto DIRICOM. Estos algoritmos están basados en, dada una solución inicial, recorrer todos los TRXs de la instancia y asignarle la frecuencia que más reduce el coste. Dependiendo del recorrido elegido, se han propuesto cuatro estrategias distintas.

OBJETIVOS:

1. Presentar diferentes algoritmos greedy para el problema AFP, que se han denominado: default, staticGreedy, dynamicGreedy y componentCost.
2. Mostrar la eficacia de estos algoritmos sobre instancias reales.

CONCLUSIONES:

1. El recorrido de los TRXs afecta a la efectividad de los algorimos greedy presentados.
2. La topología de las instancias hace que los recorridos sean más efectivos.

RELACIÓN CON ENTREGABLES:

PRE: AFP-1.0

CO: —

Málaga, December 2008

Executive Summary

TITLE: AFP-1.1: Greedy algorithms for the AFP problem

ABSTRACT: This deliverable is devoted to presenting and analyzing several greedy algorithms specially tailored to solve the automatic frenquency planning (AFP) problem that is being taken into consideration within the DIRICOM project. These greedy algorithms are based on, given an initial solution, traversing all the TRXs of the instance and assigning them with the frequency that most reduces the planning cost. Depending on the traversing, four strategies have been proposed.

GOALS:

1. Presenting different greedy algorithms for the AFP problem. They are named: default, staticGreedy, dynamicGreedy, and componentCost.
2. Showing the efficacy of these algorithms on real world instances.

CONCLUSIONS:

1. The traversing of the TRXs affects the effectiveness of the greedy algorithms studied.
2. The topology of the instances makes some traversing strategies better than others.

RELATION WITH DELIVERABLES:

PRE: AFP-1.0

CO: —

Greedy algorithms for the AFP problem

DIRICOM

December 2008

1. Introduction

Heuristic algorithms are mandatory when tackling large instances of the AFP problem [1] and, among these kind of techniques, metaheuristics [4] have shown to provide the AFP problem with very accurate solutions [2]. The efficacy of many of these algorithms is mostly based on the usage of local search methods (or greedy algorithms) specially tailored to the version of the AFP problem being addressed, i.e., they are hybrid metaheuristics [6, 5]. This deliverable is devoted to presenting and evaluating four especialized greedy strategies targeted to the AFP problem addressed in the DIRICOM project [3].

The structure of this technical report is as follows. The next section describes the four local search heuristics proposed. Section 3 includes the experimentation performed for evaluating these algorithms over the real-world instances defined within the DIRICOM project.

2. The Greedy Method

The greedy algorithms considered here are local search methods since they are based on iteratively improve one given solution to the problem, i.e., a given frequency plan. With the mathematical formulation used [3], a solution to the AFP problem is obtained by assigning to each transceiver $t_i \in T$, $i = 1, \ldots, n$, one of the frequencies from F_i, the set of valid frequencies for TRX t_i. A solution is therefore encoded as an array of integer values, p, where $p(t_i) \in F_i$ is the frequency assigned to transceiver t_i. That is, the solutions manipulated are tentative frequency plans of the given AFP problem instance.

Algorithm 1 Pseudo-code for the default local search

```
1: input: a solution p
2: improved ← true
3: while improved = true do
4:   improved ← false
5:   // Traverse every TRX t_i
6:   for i ← 1 to n do
7:     Replace frequency p(t_i) with the frequency from F_i that most reduces the objective function value
8:     if the objective function value was reduced then improved = true
9:   end for
10: end while
11: output: a possibly improved solution p
```

The basic local search method used in this work is included in Algorithm 1. It works by traversing all the TRXs of the cellular network in the order they were defined by the operator (lines 6 to 9). Then the frequency that most reduces the AFP cost of the entire plan (Eq. 1 in [3]) is chosen. The size of the instances addressed makes the computational cost of this algorithm unaffordable, so rather than using the Eq. 1 of [3] to compute the AFP cost, an incremental cost function has been used because the increase of the AFP cost caused by the setting $p(t_i) = f$ can be computed as follows:

$$\Delta(p, p(t_i) = f) = \sum_{t \in \hat{T}} \left(C_{\text{sig}}(p, t, t_i) + C_{\text{sig}}(p, t_i, t) \right) \quad . \tag{1}$$

As it can be seen, the stopping condition is met when no further improvement in the current solution is reached. That is, when the search gets stuck in a local minimun. This is called the default local search strategy. The newly proposed ones are based on using a different traversing of the TRXs. That is, instead of adopting the predefined order given by the operator, three approaches are engineered:

- StaticRandom: before starting traversing the TRXs, they are randomly shuffled. The new order is kept during all the subsequent iterations of method.

- DynamicRandom: at every iteration, the TRXs are randomly shuffled. That is, the traversing changes at every step of the local search.
- componentCost: The TRXs are ranked with respect to their component cost, CC. Given a frequency plan p and a TRX t, $CC(p,t)$ is defined as

$$CC(p,t) = \sum_{u \in T, u \neq t} C_{\text{sig}}(p,t,u) \tag{2}$$

that is, $CC(p,t)$ is the value with which TRX t contributes to the total cost of the frequency plan p. This ranking allows the TRXs incurring in the strongest interference to be assigned in the beginning so as to quickly fix low quality assignments and to lead to further improvements. The method slightly different (Algorithm 2) because the component cost of every TRX has to be updated after every new frequency assignment (line 9) so as to reassign again first those TRXs that most contribute to the AFP cost.

Algorithm 2 Pseudo-code for the componentCost local search

1: **input:** a solution p
2: *improved* $\leftarrow$ **true**
3: **while** *improved* $=$ **true do**
4:     *improved* $\leftarrow$ **false**
5:     Rank every TRX t_i with $CC(p,t_i)$
6:     **for** $i \leftarrow 1$ to n **do**
7:         Replace frequency $p(t_i)$ with the frequency from F_i that most reduces the objective function value
8:         **if** the objective function value was reduced **then** *improved* $=$ **true**
9:         Update $CC(p,t_i)$
10:     **end for**
11: **end while**
12: **output:** a possibly improved solution p

	Seattle			Denver		
	$\bar{x}$	σ_n	min	$\bar{x}$	σ_n	min
default	4432.46	458.10	3407.39	108120.01	2089.19	103728
staticRandom	4457.96	427.05	3376.77	107781.72	2334.35	103520
dynamicRandom	4493.52	483.66	2875.81	107512.32	2308.78	101943
componentCost	4534.65	434.28	3625.16	107312.49	1804.80	102799

Tabla 1: Planning costs reached by different greedy algorithms

3. Experiments

This section presents the experiments performed to check the efficacy and efficienty of the four greedy algorithms on two real world instances, namely Seattle and Denver, of the AFP problem (see [3]). The largest instance, Los Angeles, has been discarded since one single run of any local search takes more than 10 hours at least, what makes this study unaffordable.

All the algorithms have been executed 30 times, each one starting from a different solution (i.e., frequency plan) which is generated randomly. Table 1 includes the average, $\bar{x}$, the standard deviation, σ_n, and the minimum value, min, over these 30 independent runs of the four greedy algorithms for the Seattle and Denver instances. As it can be observed, the resulting costs are very similar for the two instances. In the smaller one (Seattle), the default strategy reaches the lower cost on average (4432.46). This means that the order in which the operator defined the instance is specially well suited for this greedy algorithm since the frequency assignment is done by properly reducing the interference in the cellular network. The most complex strategy, componentCost, is the worst performing one on this Seattle instance. It may happen that the component cost of several unrelated TRXs is the same (the most highest one) so therefore the assignment is taking place on different parts of the network that and, when their new frequency plans have to be put together, no low cost frequency assignment is available. The picture is completely reversed when one considers the results on the Denver instance. That is, now the default strategy is the worst performing one and the componentCost is the best on average (the AFP costs are 108120.01 and 107312.49, respectively). The explanation has to do with the instance topology [3], which includes somehow several clusters. Therefore, joining several parts of the networks does not imply violating many constraints. Finally, it is worth mentioning the average behavior of the random traversing strategies. They two are always between the default and the componentCost algorithms. The interesting point here is that the dynamicRandom has always reached the minimum planning cost (see the *min* column in Table 1) of the two instances. It may be a subject of further research to better analyze the traversing that reached this costs and check whether it holds for different initial solutions.

Referencias

[1] K. I. Aardal, S. P. M. van Hoesen, A. M. C. A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for frequency assignment problems. *Annals of Operations Research*, 153(1):79 – 129, 2007.

[2] E. Amaldi, A. Capone, F. Malucelli, and C. Mannino. *Handbook of Optimization in Telecommunications*, chapter Optimization Problems and Models for Planning Cellular Networks, pages 917 – 939. Springer, 2006.

[3] DIRICOM. Real-world instances for the Automatic Frequency Planning problem. Technical Report AFP-1.0-2008, The DIRICOM project, 2008.

[4] F. W. Glover and G. A. Kochenberger. *Handbook of Metaheuristics*. Kluwer, 2003.

[5] G. Raidl. A unified view on hybrid metaheuristics. In *Hybrid Metaheuristics (HM 2006)*, LNCS 4030, pages 1 – 12, 2006.

[6] E-G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(2):807 – 819, 2002.