

Performance and Cost Trade-off in Tracking Area Reconfiguration: A Pareto-optimization Approach

Sara Modarres Razavi, Di Yuan, Fredrik Gunnarsson and Johan Moe

Linköping University Post Print

N.B.: When citing this work, cite the original article.

Original Publication:

Sara Modarres Razavi, Di Yuan, Fredrik Gunnarsson and Johan Moe, Performance and Cost Trade-off in Tracking Area Reconfiguration: A Pareto-optimization Approach, 2012, Computer Networks, (56), 1, 157-168.

<http://dx.doi.org/10.1016/j.comnet.2011.08.017>

Copyright: Elsevier

<http://www.elsevier.com/>

Postprint available at: Linköping University Electronic Press

<http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-71335>

Performance and Cost Trade-off in Tracking Area Reconfiguration: A Pareto-optimization Approach

Sara Modarres Razavi^{a,*}, Di Yuan^{a,b}, Fredrik Gunnarsson^b, Johan Moe^b

^a*Department of Science and Technology, Linköping University, Sweden*

^b*Ericsson Research, Ericsson AB, Sweden*

Abstract

Tracking Area (TA) design is one of the key tasks in location management of Long Term Evolution (LTE) networks. TA enables to trace and page User Equipments (UEs). As UEs distribution and mobility patterns change over time, TA design may have to undergo revisions. For revising the TA design, the cells to be reconfigured typically have to be temporary torn down. Consequently, this will result in service interruption and “cost”. There is always a trade-off between the performance in terms of the overall signaling overhead of the network and the reconfiguration cost. In this paper, we model this trade-off as a bi-objective optimization problem to which the solutions are characterized by Pareto-optimality. Solving the problem delivers a host of potential trade-offs among which the selection can be based on the preferences of a decision maker. An integer programming model has been developed and applied to the problem. Solving the integer programming model for various cost budget levels leads to an exact scheme for Pareto-optimization. In order to deliver Pareto-optimal solutions for large networks in one single run, a Genetic Algorithm (GA) embedded with Local Search (LS) is applied.

*Corresponding author. E-mail: sarmo@itn.liu.se

Unlike many commonly adopted approaches in multi-objective optimization, our algorithm does not consider any weighted combination of the objectives. Comprehensive numerical results are presented in this study, using large-scale realistic or real-life network scenarios. The experiments demonstrate the effectiveness of the proposed approach.

Keywords: Bi-criteria optimization, reconfiguration, signaling overhead, tracking area.

1. Introduction

Tracing users cost-efficiently is one of the major challenges in mobility management of cellular networks [5]. Tracking Area (TA) is a logical grouping of cells in Long Term Evolution (LTE) networks [1, 2]. TA manages and represents the location of User Equipments (UEs). Similar grouping concept is called Location Area (LA) in circuit-switched domain and Routing Area (RA) in packet-switched domain in GSM, GPRS and UMTS. Note that, although the optimization framework developed in this paper generalizes to other networks, we focus on LTE due to its emphasis on automatic network reconfiguration [1, 2]. The concept of reconfiguration means that the management system has the intelligence of adapting network configurations to changes and trends in UE distribution and demand.

In configuring TAs, a key consideration is to minimize the total amount of signaling overhead. The overall signaling overhead of a network consists of two separate terms: update overhead and paging overhead. In the standard strategy of TA update and UE paging, the Mobility Management Entity (MME) records the TA in which the UE is registered. When UE moves

to a new TA, there will be an update signaling overhead. The location information that the MME has about the idle UE is the registered TA. The paging signaling overhead is incurred by the event that a UE is called. One paging strategy is to command all cells in the TA to broadcast messages to page the UE. (Alternatives have been proposed, where paging is first performed in a subset of the cells in the TA). Having TAs of very small size (e.g., one cell per TA) virtually eliminates paging, but causes excessive registration, whereas TAs of too large size give the opposite effect. Thus the natural objective in TA planning is to reach an optimal balance between update and paging signaling. There are extensive studies on optimization methods to deal with this objective [4, 6, 7, 8, 9, 12, 31, 38].

Consider a TA design that is optimized for a network in the planning phase. As UE distribution and mobility patterns change over time, the optimized TA configuration will no longer perform satisfactorily. To reduce the signaling overhead, TA reconfiguration becomes necessary. In this case, it is not feasible to make a new TA design from scratch, because TA reconfiguration has to use the current design as the starting point.

Reconfiguring TA, such as moving a cell from its home TA to another, usually requires to restart the cell and consequently results in service interruption. Therefore, there is a trade-off between approaching minimum signaling overhead and the cost resulted from reconfiguration. In this study, we propose a bi-objective optimization framework for the TA reconfiguration problem.

Unlike mono-objective optimization problems which have a unique optimal value, in bi-objective problems the solution set is formed by Pareto-

optimal (non-dominated) points. We develop an integer programming model to optimize the overhead by reconfiguration subject to a cost budget constraint. Applying the model to various budget levels leads to a set of Pareto-optimal solutions. For an exact assessment of the Pareto-optimal frontier, the integer model should be run many times. Solving the integer programming model is very time-consuming and sometimes infeasible for large networks. To deal with large-scale networks, we propose a genetic algorithm (GA) embedded with local search (LS) to search for Pareto-optimal solutions in one single run. In the GA approach, we use the concept of dominance in the fitness evaluation, to the contrary of approaches that use a scalarization function or treat the various objectives separately. In our GA algorithm, the amount of dominance is referred to as the Preference Value (PV), which explicitly evaluates the solution in terms of Pareto-optimality.

We demonstrate the performance of the proposed integer model and the GA algorithm via experiments using three large-scale realistic or real-life network scenarios. For the first two scenarios we were able to compare the results from the GA algorithm with the ones computed from the integer model. The last network was only solved by the GA algorithm since it was too large and not feasible to be solved with the integer programming model. The results demonstrate the ability of the approaches to deliver various Pareto-optimal solutions and thus giving the operator the opportunity of selecting a proper trade-off between the two objectives.

In real systems, how often TAs are reconfigured varies by operators' policies. Typically, TAs need to be revised when there are indications of signaling congestion, or new infrastructure is deployed (e.g., a major expansion of

sites). Our optimization framework is intended for adapting TAs to long-term trends, rather than targeting short-term changes with a regularly repeated pattern, e.g., daytime and night-time user distributions. Thus applying the framework is not an on-line process with strict constraint on computing time. Yet it is not desirable to use an exponential-time algorithm, for which the computational effort (both in time and memory requirement) can not be predicted at all. Therefore the computational efficiency is of significance even if not very crucial. Moreover, it is vital that the optimization algorithm is effective in exploring the Pareto frontier. The GA algorithm is designed with these aspects in mind. Another aspect is the impact of the time period of reconfiguration on performance. Clearly, the optimization results, in terms of the Pareto-optimal frontier, vary by the time point at which reconfiguration is considered. Frequent reconfigurations can keep the TA design close to optimal in signaling overhead, but the reconfigurations may due to short-term changes instead of long-term trends, resulting in oscillating TA patterns. In general, it is hard to determine the “right time” of reconfiguration, without knowing the performance of the current design in relation to optimality. The bi-objective optimization approach targets exploring all possible solutions trading signaling overhead against reconfiguration cost. Hence an operator can apply the approach frequently to determine the time for reconfiguration.

The remainder of the paper is constructed as follows. In Section 2, we review some works that are relevant to our study. In Section 3, the system model is discussed in detail. In Section 4, the integer programming model is presented. In Section 5, we explain our approach for evaluating Pareto-optimal solutions. Section 6 is devoted to the genetic algorithm and local

search. Section 7 gives a method to improve the efficiency of the algorithm. In Section 8, we conduct performance evaluation and present the numerical results. Section 9 concludes the paper.

2. Related Works

One line of research on reducing signaling overhead consists in the development of advanced/selective update and paging strategies. Such alternative strategies for location management have been previously proposed for GSM and UMTS networks. In these strategies, update and paging decisions are made using more individual information of the users. For example, the update decision can be based on movement [4], time [29], distance [41], as well as mobility and call arrival patterns [31, 34]. For paging, various sequential schemes have been proposed in [4, 23, 25, 31, 32, 39]. Although the update and paging schemes proposed in the references are very promising in reducing the signaling overhead, their use requires modifications of system implementation and knowledge of the user mobility pattern (by assuming a mobility model or collecting user information). Hence the standard strategy as described in Section 1 remains widely used and is considered in our study.

Recent developments of TA optimization are presented in [10, 24]. The authors of [10] combine rule base paging with movement-based location update, and derive analytical results of the performance improvement that is achievable by the integrated scheme. In [24], the authors apply stochastic models to characterize the trade-off between paging and update, and provide a unified analytical model for the minimum overall overhead. The results enable sensitivity analysis of the impact of paging and update parameters on

TA size.

For TA design that is more commonly known as LA design in the previous literature, the key objective has been to minimize the total amount of signaling overhead. This minimization problem is known to be *NP*-hard [38] and therefore solutions to large networks are typically obtained by heuristic algorithms, such as insertion and exchange local search [30], simulated annealing [12, 13], and genetic algorithms [18]. Extensions to optimizing multi-layered LAs and jointly designing LA and RA are presented in [22, 40]. The references assume that the design is done from scratch, and hence they do not fit into our system model of TA reconfiguration.

For reconfiguration, minimum service disruption is crucial. In [26], we studied the problem of TA re-optimization given a constant budget level. Solving the problem gives one potential trade-off between signaling overhead and reconfiguration cost. It was shown that the re-optimization problem is an *NP*-hard problem, whether the budget constraint is active or not. Here, we develop and apply an integer programming model to the problem which was previously solved heuristically in [26]. In the current paper, we extend our previous study of addressing reconfiguration by a bi-objective optimization problem [27]. Moreover, we present a GA algorithm targeting Pareto-optimal solutions in one single run.

An intuitive approach for *NP*-hard problems is decomposition. For TA design, this amounts to decomposing the network into smaller partitions, before constructing TAs within each partition. If the partitioning is optimal (i.e., at global optimum, no TA contains cells in different partitions) or close-to-optimal, problem decomposition improves computational efficiency

with little loss in solution quality. Unfortunately, the task of optimal partitioning is far from being straightforward, because determining the number of partitions and the cells of each partition, in its turn, has to balance the two types of signaling overhead. Hence decomposition does not reduce the theoretical complexity, meaning that simple algorithms, such as greedy construction, may lead to very poor partitioning solutions. Indeed, even the simplest case of partitioning a TA into two with maximum difference between the reduction of paging overhead and the increase in update overhead, is a graph max-cut problem that is *NP*-hard [16]. For TA reconfiguration, applying decomposition is more complicated, because there is a given initial configuration, and the reconfiguration cost has to be accounted for.

3. System Model

The signaling overhead of paging and TA update are determined by UE distribution and mobility. From a mobility perspective, a UE is mainly in either of these two states:

- *LTE-Active*: the network knows the cell to which the UE belongs, and the UE can transmit and receive data from the network.
- *LTE-Idle*: the network knows the location of the UE to the granularity of a group of cells (forming a TA).

In this study, by using the network statistics of cell load and handover, which generally represent the active UEs, we estimate the locations of idle UEs and their mobility behavior.

We denote the set of cells by $\mathcal{N} = \{1, \dots, N\}$ and the set of TAs currently in use by $\mathcal{T} = \{1, \dots, T\}$. We use a vector $\mathbf{t} = [t_1, \dots, t_N]$ as a general notation of cell-to-TA assignment, where t_i is the TA of cell i . TA design \mathbf{t} can be alternatively represented by an $N \times N$ symmetric and binary matrix $\mathbf{S}(\mathbf{t})$, in which element $s_{ij}(\mathbf{t})$ represents whether or not two cells are in the same TA, i.e.,

$$s_{ij}(\mathbf{t}) = \begin{cases} 1 & \text{if } t_i = t_j, \\ 0 & \text{otherwise.} \end{cases}$$

For a given time period, let u_i be the total number of UEs in cell i scaled by the time proportion that each UE spent in cell i . For the same time period, h_{ij} is the number of UEs moving from cell i to cell j . The value of h_{ij} can be assessed by handover statistics of active UEs. The amount of overhead of one paging and one update operation are denoted by c^p and c^u , respectively. Moreover, parameter α is the call intensity factor (i.e., probability that a UE has to be paged). The total update and paging signaling overhead is defined by $c_{SO}(\mathbf{t})$, and is calculated by Equation (1):

$$c_{SO}(\mathbf{t}) = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}: j \neq i} (c^u h_{ij} (1 - s_{ij}(\mathbf{t})) + \alpha c^p u_i s_{ij}(\mathbf{t})) \quad (1)$$

Within the outer parentheses of (1), the first term accounts for the update overhead for UEs moving from i to j (if the two cells are not in the same TA), and the second term is the paging overhead incurred in cell j while paging UEs in cell i (if the two cells are in the same TA).

For TA re-optimization, we are given the TA design currently deployed in the network. This solution is denoted by \mathbf{t}^0 . We consider the logical move

of a cell to a new TA as the building element of TA reconfiguration. If the result of reconfiguration is \mathbf{t} , then reconfiguration means to move all cells i from t_i^0 to t_i for which $t_i^0 \neq t_i$. We allow the reduction of the number of TAs, it means that if a TA becomes empty after cell moves, it is simply deleted. To simplify the presentation, we do not consider increasing the total number of TAs, although the solution algorithm can be easily extended to include this option.

For every cell, we define a parameter to represent the cost in service interruption, if the TA of the cell is changed. For convenience (and without loss of generality), we use the UE distribution parameter u_i to measure the impact of service interruption of cell i . Let $\mathbf{d}(\mathbf{t}, \mathbf{t}^0)$ be a binary vector representing cells that have been assigned new TAs, that is, $d_i(\mathbf{t}, \mathbf{t}^0) = 1$ if and only if $t_i^0 \neq t_i$, $i \in \mathcal{N}$. The cost of reconfiguration is denoted by $c_R(\mathbf{t})$ and is computed by Equation (2).

$$c_R(\mathbf{t}) = \sum_{i \in \mathcal{N}} u_i d_i(\mathbf{t}, \mathbf{t}^0) \quad (2)$$

We aim to characterize the trade-off between $c_{SO}(\mathbf{t})$ and $c_R(\mathbf{t})$ of design \mathbf{t} ; to this end, we model the problem with the following bi-objective formulation:

$$\begin{aligned} \min_{\mathbf{t}} \quad & (c_{SO}(\mathbf{t}), c_R(\mathbf{t})) \quad (3) \\ \text{subject to} \quad & s_{ij}(\mathbf{t}) = \begin{cases} 1 & \text{if } t_i = t_j, \\ 0 & \text{otherwise.} \end{cases} \\ & d_i(\mathbf{t}, \mathbf{t}^0) = \begin{cases} 1 & \text{if } t_i^0 \neq t_i, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

Generation of Pareto-optimal or non-dominated solutions is the primal goal in solving a bi-objective problem. A solution is called *Pareto-optimal* if it is not possible to improve a given objective without deteriorating at least another objective [37]. Clearly, it does not make sense to choose a solution that is not Pareto-optimal. A large amount of references for multi-objective optimization are available in the literature [35, 36, 37].

4. An Integer Programming Model

One approach is to minimize $c_{SO}(\mathbf{t})$ defined in (1) for various reconfiguration cost budgets. In other words, the TA re-optimization problem is solved repeatedly for different limits on $c_R(\mathbf{t})$. If we denote the budget value by B , the budget corresponds to a constraint $c_R(\mathbf{t}) \leq B$ in the 0/1 integer programming model. The model has two sets of binary variables:

- s_{ij} is 1 when i and j are in the same TA and 0 otherwise.
- p_{it} is 1 when cell i belongs to TA t and 0 otherwise.

Here, we formulate our integer programming model:

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}: j \neq i} (c^u h_{ij}(1 - s_{ij}) + \alpha c^p u_i s_{ij}) \quad (4)$$

subject to:

$$\sum_{t \in \mathcal{T}} p_{it} = 1, \forall (i \in \mathcal{N}) \quad (5)$$

$$p_{it} + p_{jt} - 1 \leq s_{ij}, \forall (i, j \in \mathcal{N}, t \in \mathcal{T}) \quad (6)$$

$$s_{ij} + p_{it} - 1 \leq p_{jt}, \forall (i, j \in \mathcal{N}, t \in \mathcal{T}) \quad (7)$$

$$s_{ij} + s_{jk} - s_{ik} \leq 1, \forall (i, j, k \in \mathcal{N}, i \neq j \neq k) \quad (8)$$

$$\sum_{i \in \mathcal{N}} u_i (1 - p_{it_i^0}) \leq B \quad (9)$$

In the presented model, constraint (5) assures that each cell is assigned to one TA. Constraints (6) and (7) define the matrix $S(\mathbf{t})$ and the correlation between s_{ij} and p_{it} . When $p_{it} = p_{jt} = 1$, it means that i and j are in the same TA t , hence $s_{ij} = 1$ as imposed by constraint (6). If $p_{it} = 1$ and $p_{jt} = 0$, then i belongs to the TA t while j does not and therefore $s_{ij} = 0$, as stated in constraint (7). Constraint (8) ensures that if two cells i and k belong to the same TA as cell j , they must also be in the same TA. That is, if $s_{ij} = s_{jk} = 1$, constraint (8) becomes $s_{ik} \geq 1$, forcing $s_{ik} = 1$. Constraint (9) bounds the number of UEs affected by reconfiguration using the budget level. From the definition of variable p_{it} , it is clear that $p_{it_i^0}$ is one when cell i belongs to the currently deployed TA t_i^0 and zero otherwise.

If we solve the model for $B = 0$, we keep the current configuration $\mathbf{t}^0 = [t_1^0, t_2^0 \dots t_i^0 \dots t_N^0]$. The signaling overhead of this configuration is likely not optimum, but on the other hand the reconfiguration cost is zero. This point is among the Pareto-optimal solutions, as we cannot find any solution with better reconfiguration cost. The other Pareto-optimal solutions can be calculated by giving other values to B .

5. A Dominance-based Approach

The solution space of our problem, depending on the scale of the network, can be very large. In view of this and the complexity results in [26], it is

motivated to apply meta-heuristics to deal with this problem for large scale networks and to deliver the Pareto-optimal solutions in a single run. Multi-objective meta-heuristics can be classified into four main categories, based on their solution evaluation strategies:

- Scalar approaches, which transform the problem into a mono-objective problem. A typical example is the *Weighted sum* method, which combines the objective functions by non-negative weights and converts them into one objective function [20]. Another example is the *Goal programming* method that uses a target value for each objective function, and the overall goal is to minimize the deviation from the target values [11].
- Criterion-based approaches which are mainly based on treating the various incommensurable objectives separately, such as the *Parallel* [33] and the *Lexicographic* approach [15]. In the latter, to evaluate a solution against another, the two objective function vectors are compared lexicographically.
- Indicator-based approaches which use performance quality indicators as a search guide [42].
- Dominance-based approaches that use the concept of dominance in the fitness evaluation [17].

To achieve high quality solutions, we consider two aspects: 1) The convergence to the Pareto-optimal frontier, and 2) the diversity in the search procedure. Among the approaches, weighted sum is a frequently used method for

solving multi-objective optimization [20]. Here, we do not use this approach for our problem for three reasons. First, our problem is a combinatorial bi-objective problem, thus the number of Pareto-optimal solutions can be exponential in the problem size [14]. Second, in our problem, there may exist Pareto-optimal solutions which cannot be resulted from any weighted sum of the objective functions. Third, to obtain a diverse set of Pareto-optimal solutions by the weighted sum approach, multiple runs of the algorithm are required. In general, setting the weights is a difficult task as there are many different combination of weights.

We use a dominance based approach to evaluate the solutions. We define a performance metric, which is referred to as the Preference Value (PV), to quantify each solution in terms of Pareto optimality. For TA configuration \mathbf{t} , its PV, denoted by $PV(\mathbf{t})$, is the number of solutions that perform better in both signaling overhead and reconfiguration cost in comparison to \mathbf{t} . In effect, \mathbf{t} represents the amount of domination. Hence, by definition, solution \mathbf{t} with $PV(\mathbf{t})=0$ is Pareto-optimal.

Figure 1 gives an illustrative example of the PV values of eight solutions on a two-dimensional plan, of which the two axes represent signaling overhead and reconfiguration cost, respectively. For solution \mathbf{t} , the corresponding point in the plan is $(C_{SO}(\mathbf{t}), C_R(\mathbf{t}))$. The four points, $(0, 0)$, $(C_{SO}(\mathbf{t}), 0)$, $(0, C_R(\mathbf{t}))$, and $(C_{SO}(\mathbf{t}), C_R(\mathbf{t}))$, together define a rectangular area. Clearly, solution \mathbf{t}' corresponds to a point inside the area, if and only if $C_{SO}(\mathbf{t}') < C_{SO}(\mathbf{t})$ and $C_R(\mathbf{t}') < C_R(\mathbf{t})$. Hence the number of points in the rectangular area is $PV(\mathbf{t})$. In the figure, this is highlighted for the solution with $PV=2$. In general, a solution that performs poorly in either of the objectives tends to

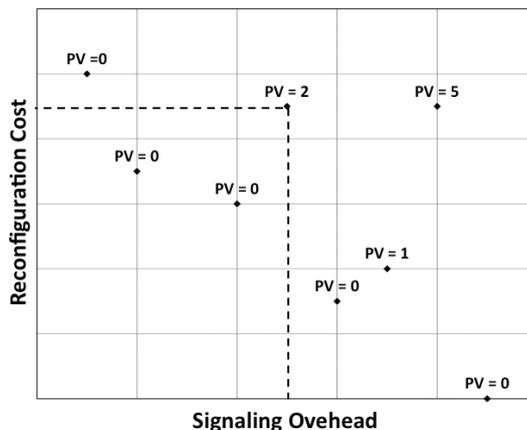


Figure 1: An example defining the PV of each TA reconfiguration solution.

have a high PV, e.g., the solution with PV=5 in the example. Finally, note that there are five solutions for which no other solution has better values in both objectives. These are the Pareto-optimal solutions with PV=0. One of the Pareto-optimal solutions has zero reconfiguration cost; this is the current TA configuration, i.e., solution \mathbf{t}^0 .

By definition, calculating the PV of a solution requires the knowledge of the entire solution space. This calculation is clearly not practical. The algorithm presented in the next section accumulates the points in signaling overhead and reconfiguration cost while searching the solution space, and use the points to obtain an approximation of PV. As more and more points get accumulated during the search, the approximation is continuously improved.

6. A Genetic Algorithm

We develop a Genetic Algorithm (GA) [17] embedded with Local Search (LS) [3]. There are two reasons for choosing a GA approach for the problem:

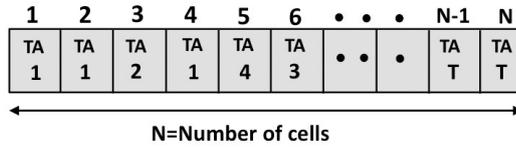


Figure 2: Solution vector representation.

1. The encoding of solutions is simple by means of integer-valued vectors.
2. We are looking for Pareto-optimal solutions, which by themselves form a population of solutions, in one single run. Thus a population-based meta-heuristic approach is a reasonable algorithm candidate.

For solution encoding, fixed length vectors with a size equal to N are used. The elements in each vector represent the TA numbers which the cells belong to. Figure 2 illustrates the solution vector representation.

Figure 3 summarizes the principle design of the algorithm. The bulk of the algorithm consists in the following steps. First, an initial population of the solutions, in form of vectors representing the corresponding TA configurations, is generated, and their PV values are calculated. In the figure, POPSIZE denotes the population size. Next, a subset of the population is selected for the two key operators of GA: crossover and mutation. For each of the operators, up to POPSIZE new solutions are generated. These solutions expand the initial population. Each time a new solution is generated, those solutions that are dominated by the new one will have their PV increased by one. In the next step, a set of elite solutions, consisting in those having PV up to a threshold, denoted by PV-MAX, are selected; the remaining, inferior solutions are discarded. The set of elite solutions goes through an intensification step, in which a local search algorithm is applied

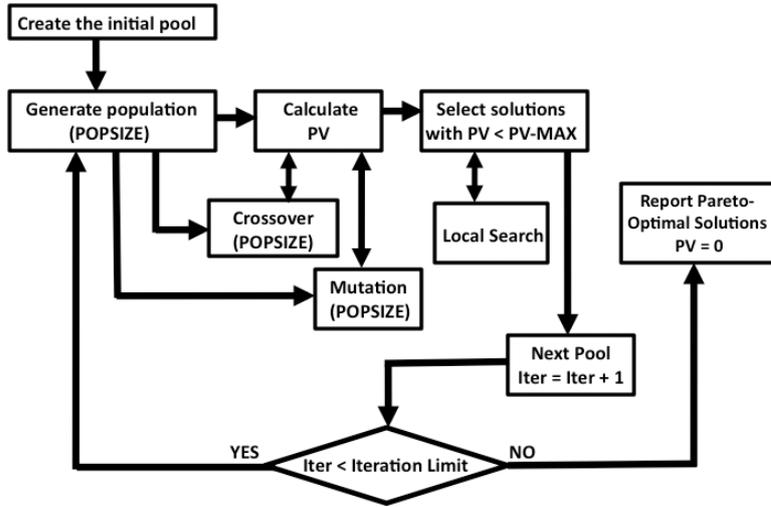


Figure 3: Principle design in finding Pareto-optimal configurations.

to make trial modifications in order to seek improvements in the two objectives. The improved set of solutions becomes the initial pool for the next iteration. The algorithm stops after a fixed number of iterations (denoted by Iteration Limit in the figure). The algorithm keeps track on solutions with $PV=0$, which form the Pareto-optimal set.

6.1. Population Initialization

Generating the first population of a GA plays an important role in approaching good solutions rapidly. The population must be rich enough to enable high-quality solutions. In order to set the first population, we generate an initial pool as follows.

The current TA configuration \mathbf{t}^0 , which is among the Pareto-optimal solutions, is used as the starting point. To create diversity in the initial pool, we apply the local search algorithm discussed in [26]. Starting from \mathbf{t}^0 , this local search algorithm iteratively updates the TA design. In every iteration,

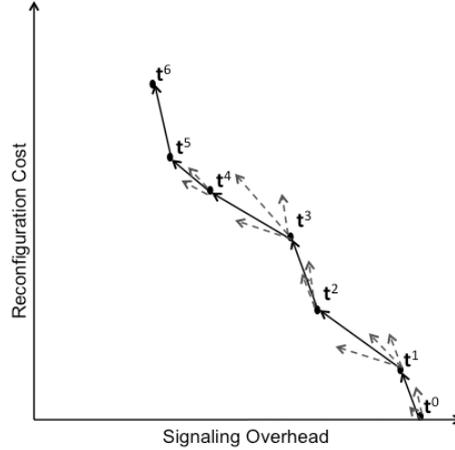


Figure 4: The local search algorithm for creating the initial pool.

the algorithm considers cells that may be moved, and among these selects the cell move that results in the largest improvement in signaling overhead. This is repeated, without accounting for the reconfiguration cost, until no further improvement can be obtained. In [26], the goal was to find the optimum reconfiguration regardless of cost, while here we keep all configurations encountered on the way to the lowest found signaling overhead. The initial pool consists of all the configuration points visited by the local search algorithm. Figure 4 illustrates the local search procedure. The dashed arrows represent the possible moves from t^n to t^{n+1} , where n is the iteration counter. The solid arrows show the moves with the largest improvement in the signaling overhead. From Figure 4, we can see that, while the local search starts from t^0 and searches for configuration points with lower signaling overhead, the reconfiguration cost of those points are increased. The reason is that more cells change configuration compared to the initial design t^0 .

All points in the initial pool will be inside the first population. For gener-

ating the rest of the population, the algorithm randomly picks a configuration from the initial pool and perturbs the TA configuration of 20% of the cells. This is repeated until the population size reaches POPSIZE. To avoid poor configurations, during the perturbation, a cell can change TA, only if it is geographically located on the boundary of its TA. This is the case if the cell has at least one neighboring cell with positive handover and the neighboring cell is currently assigned to a different TA. In addition, the new TA of the cell is picked among the TAs of the neighboring cells.

6.2. Crossover

The role of the crossover operator is to inherit some characteristics of two parents to generate offspring [37]. In the crossover operator, two parents are chosen randomly with the preference of having lower PV values. The elements are swapped between some randomly chosen points to make two new offspring. Figure 5 explains the 2-point crossover method applied in this study. It is apparent from Figure 5 that the cells in each offspring follow one of the parents' TA assignments, and therefore the output offspring from the crossover operator are valid solutions.

In our GA algorithm, we repeat the crossover operation until the number of offspring is equal to POPSIZE. In order to avoid identical offspring, we make sure that the chosen parents are different, and the two crossover points are chosen with the condition that the two parents differ in at least one position between the two points.

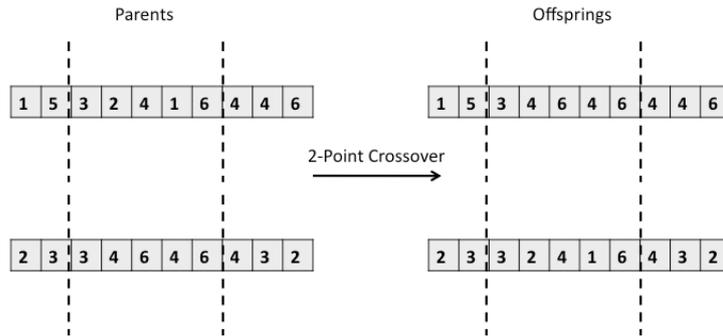


Figure 5: The 2-point crossover method.

6.3. Mutation

The mutation operator randomly modifies the elements of TA configuration vectors to promote diversity. In our algorithm, a configuration is randomly chosen from the population with the preference of having low PV to enter the mutation operator. In the selected configuration, 5% of the elements are mutated. In our GA algorithm, we repeat the mutation operation in POPSIZE times. Similar to the perturbation procedure described in Section 6.1, the mutation of a cell may take place only if the cell is on the boundary of its TA, and the TA of that cell can only be changed to a neighboring TA.

6.4. PV Local Search Algorithm

Usually the solutions obtained from GA can be improved by some simple modifications. In this study, during each iteration of GA, we use PV Local Search (LS) to further improve the performance. For each solution given to LS, the algorithm considers moving cells to other neighbor TAs one by one. Among these moves, the first move which results in a lower PV value is

chosen, as long as the point defined by signaling overhead and reconfiguration cost has not been visited until then. If the LS gets stuck in the situation where no move results in unvisited point with lower PV, the algorithm moves to an unvisited point with equal PV. The algorithm stops if all possible moves lead to visited points or higher PV values. All points visited by LS are stored and considered as visited in later runs of LS.

The goal for using LS in this stage of GA is to first find new solutions with lower PV to improve GA performance, and second to look for new Pareto-optimal solutions. The next pool in the GA algorithm consists of solutions with $PV < PV\text{-MAX}$ after LS.

It is possible to tune the number of points entering the LS by giving a value to PV-MAX. For example by giving $PV\text{-MAX} = \text{POPSIZE}$, all the points will be considered as input to LS. PV-MAX is set to be lower than POPSIZE to save computing effort in case of large-scale networks.

7. Efficiency Improvement

There are two computational bottlenecks in the suggested algorithm. First, the PV of a solution is a relative value that is set in relation to other solutions. Therefore, in order to calculate and update the PV of each solution, we have to compare its signaling overhead and cost to all other solutions. Second, points that are visited should be stored in order to avoid being generated repeatedly. Ideally, one would like to record all the configuration points found by the algorithm. This is however computationally unaffordable, since the number of accumulated solutions grows rapidly from one iteration to another. In this section, we propose a method to resolve these bottlenecks by

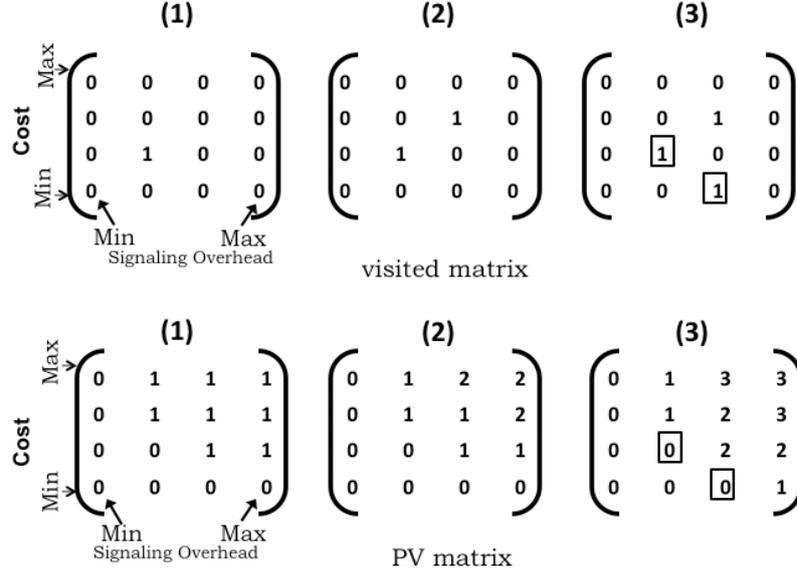


Figure 6: Updating Visited-matrix and PV-matrix, while adding a new solution in each stage. The elements in the boxes represent Pareto-optimal solutions.

quantizing the two objective values.

The quantization of the bi-objective space approximates the signaling overhead and reconfiguration cost values by a fixed and large number of intervals. The approximation results in a grid of pixels. Each pixel corresponds to a two-dimensional box with given ranges in overhead and cost. A pixel is used to represent all TA configurations with signaling overhead and cost values falling into the ranges of the pixel. For practical implementation, we map the grid to matrix of which the dimensions match those of the grid. We define two matrices; each of them will help solving one of the mentioned bottlenecks.

7.1. Visited matrix

To represent the configurations found in the progress of the algorithm, we define the Visited matrix. It is a binary matrix to represent whether a grid element has been so far visited or not. If an element of this matrix is one, it means that we have already visited a solution having signaling overhead and cost within that pixel, otherwise the value is zero. The upper part of Figure 6 shows a small example of how the Visited matrix gets updated while new solutions are found by the algorithm. Moving from the first matrix to the third (left to right), in each step one solution is added to the Visited matrix by changing one element from zero to one.

7.2. PV matrix

In order to find out the PV of a solution in constant time, we define a PV matrix. This matrix has the same dimension as Visited matrix, and it is used to store the PV of each solution. Each time the algorithm finds a new solution, it is used to update the PV matrix by increasing all the dominated elements to the right and up of the corresponding pixel of the new solution by one. With this method of updating the PV matrix, the value of each element in the PV matrix represents the number of solutions that dominate the solution of the corresponding element. The lower part of Figure 6 illustrates a small example of how to update the PV matrix while adding a new solution. Note that the Pareto-optimal solutions correspond to the elements which are ones in the Visited matrix and zeros in the PV matrix.

8. Performance Evaluation

We present results of performance evaluation for realistic or real-life data of three large-scale networks. In real-life networks, splitting a site into different TAs is not a common practice. Therefore, although all the discussions before considered cell-level TA assignment, the evaluation of the three networks is done at the site level. It is assumed that 5% of the UEs are paged in every site ($\alpha = 0.05$). The overhead of a single update c^u is set ten times as much as c^p [21].

For each of the first two networks, a reference scenario of UE distribution and mobility is defined. The scenario contains load and handover statistics of the network. The initial TA configuration, \mathbf{t}^0 , is optimal for the reference scenario. We generate another UE scenario, called scenario I, by modifying the load and the handover statistics. It is considered that the reference scenario has evolved to scenario I over time. Our aim is to find the Pareto-optimal solutions of reconfiguration for scenario I. The third network is a real-life case, and \mathbf{t}^0 is the configuration used in the past few years. We apply our algorithm to find the Pareto-optimal solutions for reconfiguring \mathbf{t}^0 for the up-to-date UE distribution and mobility data.

The integer programming model defined has been implemented in the Gurobi optimizer [19]. The solver has been run on a server with 2.4 GHz CPU and 7 GB RAM. For the first network, the integer programming model delivers all the exact Pareto-optimal solutions. For the second network, some but not all of the exact Pareto-optimal solutions can be calculated. Due to network size and memory limitation, the integer programming model cannot be applied to the third network. The GA algorithm is implemented in MAT-

LAB. The computations are run on a processor of type Intel Core 2 Duo with the clock speed of 2.1 GHz.

The size of the Visited matrix and PV matrix determines the resolution of the grid in quantizing the two objective functions. Setting the size involves the trade-off between computational efficiency and solution quality. A low resolution enables faster computation, at the price of poor accuracy, because the algorithm does not distinguish between solutions having objective values within the same grid pixel. Using a high resolution gives the opposite effects. In the experiments, we emphasize on accuracy in order to obtain a reliable picture of the algorithm's performance. By (2), the reconfiguration cost is defined in cell load. Hence the amount of difference between any two TA configurations in their reconfiguration costs will be at least the minimum cell load in the network, $\min_{i \in \mathcal{N}} u_i$. In the experiments, we use $\min_{i \in \mathcal{N}} u_i$ as the quantization step for the reconfiguration cost, meaning that the quantization is in fact exact. For signaling overhead, there is no obvious lower bound on the minimum difference between solutions. For this objective, we set the quantization step to reach a high accuracy in measuring the improvement over the signaling overhead of the initial solution, i.e., improvement over $c_{SO}(\mathbf{t}^0)$; TA configurations are considered equivalent in the objective, only if the values differ no more than 0.15% of $c_{SO}(\mathbf{t}^0)$.

8.1. Network 1

The first set of data is from a cellular network of the downtown area of Lisbon, that is provided by the EU MOMENTUM project [28]. The network consists of 60 sites and 164 cells. The optimum configuration for the reference scenario, \mathbf{t}^0 , is computed by the model in [38]. There are 7 TAs in

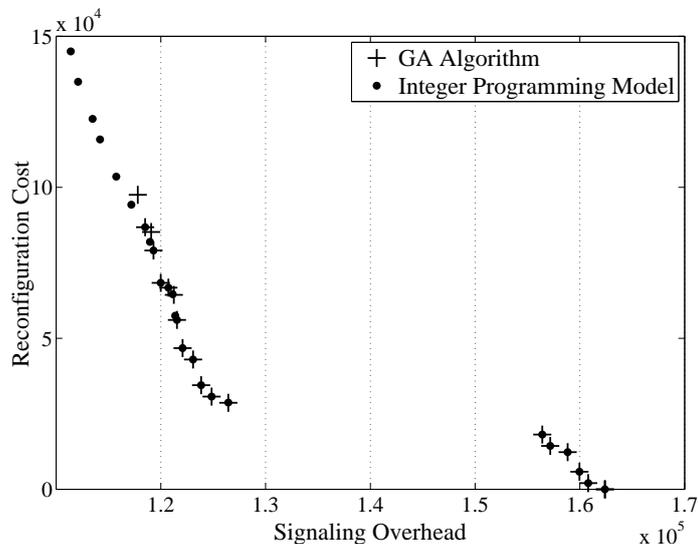


Figure 7: Pareto-optimal solutions of Network 1.

t^0 . Figure 7 shows the Pareto-optimal solutions found for the Lisbon network by the two approaches. The black dots represent the exact Pareto-optimal solutions found by the integer programming model. There are 25 Pareto-optimal solutions in Figure 7; the model has however been run more than this number to find the points. Solving the model for a given cost budget B takes an average time of 20 minutes. Thus finding all the 25 points takes at least 8 hours and 20 minutes. The plus signs in Figure 7 illustrate the heuristic Pareto-optimal solutions obtained by the GA algorithm with the following parameters: POPSIZE = 100, Iteration Limit = 10, and PV-MAX = 20. The size of the PV matrix and Visited matrix is 650-by-700. It took about 1 minute for GA to give the points. The observations from Figure 7 are as follows.

- By successively allowing higher reconfiguration cost, there is a jump

Table 1: Minimum-overhead solutions found by the two approaches.

	$c_{SO}(\mathbf{t})$	$c_R(\mathbf{t})$	c_{SO} Improvement over \mathbf{t}^0
Integer Prog. Model	1.1140×10^5	1.4499×10^5	31.40%
GA Algorithm	1.1764×10^5	9.5504×10^4	27.84%

in the improvement of overhead. This shows the importance of approaching as many Pareto-optimal solutions as possible to facilitate the decision making process of TA revision.

- The performance of the GA algorithm is close to optimality. It did not approach the point with the minimum overhead and highest reconfiguration cost. However, the relative performance difference is small. Table 1 compares the minimum-overhead solutions. The overhead improvement of the integer programming model is 5.60% over the GA algorithm. Note that for achieving this extra improvement, the reconfiguration cost will increase by 51.81%.

8.2. Network 2

The second data set represents a realistic deployment scenario for a network in one of the capital cities of Europe. The network consists of 75 sites and 225 cells. The optimum design for the reference scenario, \mathbf{t}^0 , has 22 TAs. Figure 8 shows the Pareto-optimal solutions of this network by the two approaches. The integer programming model enables some but not all of the exact Pareto-optimal solutions. It takes at least 1 hour to find each solution point. When B approaches 2000, the time for finding a solution jumps rapidly to 8 hours. Therefore, searching for exact Pareto-optimal solutions

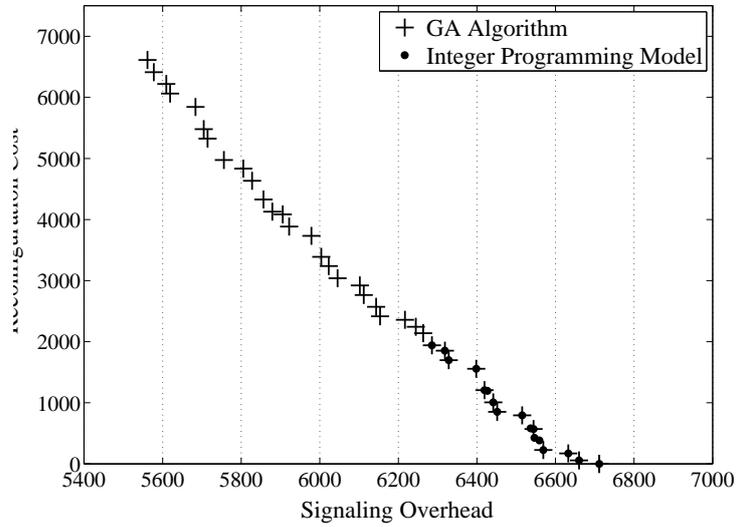


Figure 8: Pareto-optimal solutions of Network 2.

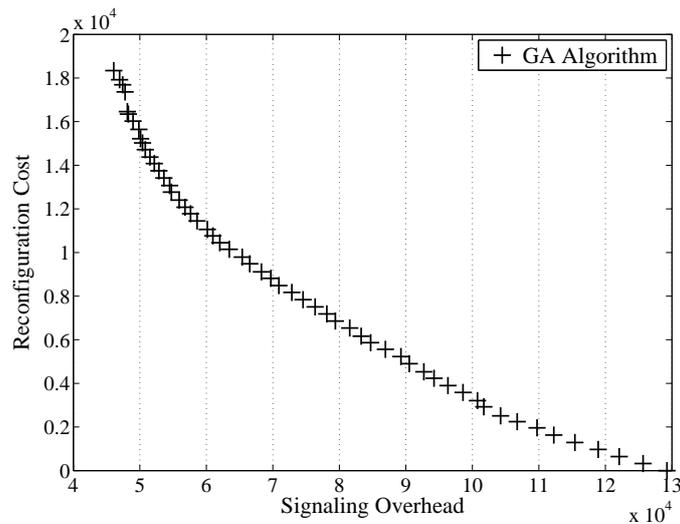


Figure 9: Pareto-optimal solutions of Network 3.

for $B \geq 2000$ is not computationally feasible. To get the heuristic Pareto-optimal solutions from the GA algorithm in Figure 8, we set the following

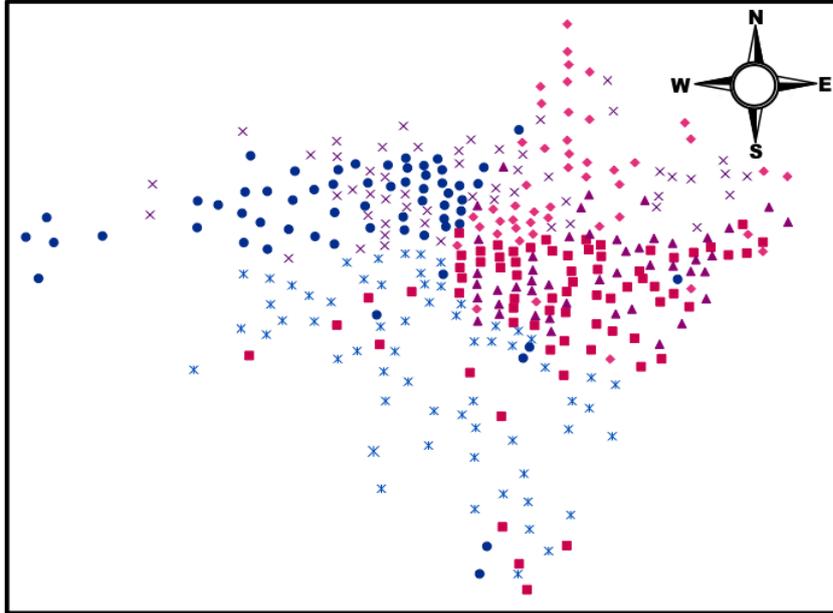


Figure 10: The initial TA design \mathbf{t}^0 of Network 3. The sites with the same symbol are in the same TA.

parameters: $\text{POPSIZE} = 100$, $\text{Iteration Limit} = 10$, and $\text{PV-MAX} = 20$. The size of the PV matrix and Visited matrix is 700-by-700. It took about 10 minutes for GA to give all these points. Below are the observations made from the figure.

- The shape of the Pareto frontier differs from that of the first network. The curve in Figure 8 is close to linear, meaning that for obtaining improvement in overhead, the reconfiguration cost scales up proportionally.
- The exact Pareto-optimal solutions computed by the integer programming model and those points found by the GA algorithm have a close-

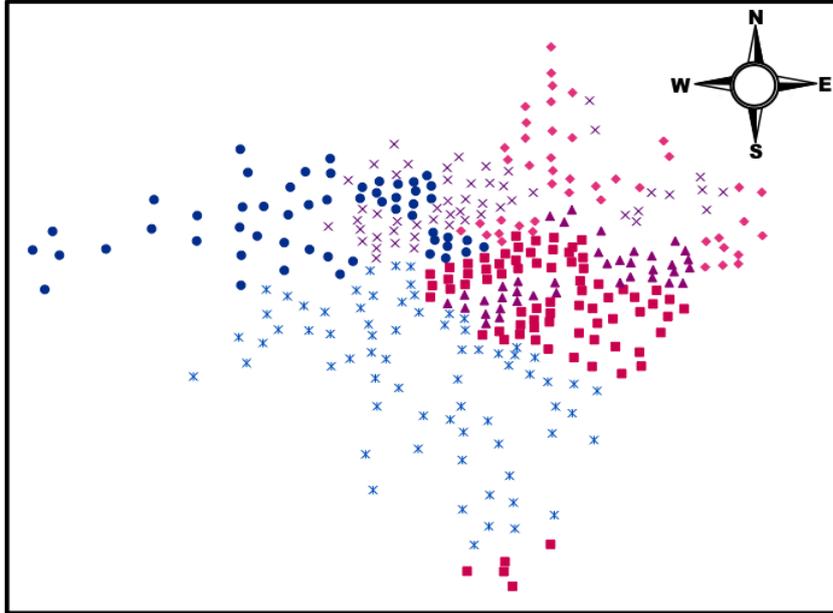


Figure 11: The Pareto-optimal design with the lowest signaling overhead of Network 3.

to-perfect match. Thus The GA algorithm performs very well and time-efficiently.

8.3. Network 3

The experiments for the third network use real-life data. The network is in use in a capital city of Asia. The network consists of 339 sites and 978 cells. The number of TAs in the current configuration is 6. The network size exceeds the solver's memory limitation and therefore it is not possible to use the integer programming model. The size of the PV matrix and Visited matrix is 1200-by-1400. It took 2 hours and 20 minutes for the GA algorithm to find the Pareto-optimal solutions in Figure 9 with the following parameters: POPSIZE = 300, Iteration Limit = 3, and PV-MAX = 20.

After the third iteration, no new Pareto-optimal solution was found. Figure 9 shows the Pareto-optimal solutions of this network obtained by the GA algorithm. The observations from this figure are as follows.

- The smooth Pareto-optimal frontier indicates that the decision maker has a large set of available trade-offs between the signaling overhead and the reconfiguration cost.
- The current TA configuration of the network is far from optimum in terms of signaling overhead. The Pareto-optimal solutions show that it is possible to decrease the overhead by 64%. Figure 10 shows the initial TA design \mathbf{t}^0 for Network 3. Figure 11 illustrates the Pareto-optimal solution having the lowest signaling overhead. Each specific symbol in the two figures represents the sites inside one TA. In Figure 11, 111 sites have changed TAs in comparison to Figure 10, i.e, about 32% of the sites in the network have been reconfigured. There are some parts in both figures giving the impression that the TAs are disjoint. The reason is the existence of highways which made the direct handover possible between those parts of the city.
- As was mentioned in Section 3, our solution algorithm does not consider increasing the number of TAs. In case of Network 3, the number of TAs is small in \mathbf{t}^0 . Additional improvements are likely, if an increase in the number of TAs is allowed. However, even without any increase, the algorithm finds reconfiguration solutions that significantly improve the performance of the network.

9. Conclusions

A bi-objective optimization framework has been formulated to approach Pareto-optimal solutions for the trade-off between the signaling overhead and the TA reconfiguration cost. We have formulated an integer programming model and proposed a dominance-based GA algorithm to solve the problem. The integer programming model provides the exact Pareto-optimal solutions and the GA algorithm is simple in implementation and efficient in performance for large-scale networks. The experiments for several networks demonstrate that the characteristic of the Pareto-optimal frontier differs by network, and that the proposed GA algorithm provides close-to-optimal solutions for large-scale networks.

The mathematical optimization framework and the solution algorithm can potentially be implemented in the MME to enhance the functionality of mobility management in LTE networks. Using statistics of cell load and handover, the optimization procedure is run as an automated process to continuously estimate the performance of the current TA configuration, and discover long-term improvements in signaling overhead along with the corresponding trade-off in reconfiguration cost. On top of the optimization process, a module analyzing the optimization results and generating candidate reconfiguration plans is needed. The module implements the operator's preference and policies in TA reconfiguration, based on, for example, thresholds and criteria on the frequency and amount of signaling congestion, as well as the performance of the reconfiguration options. Design and implementation of the module form an interesting line of further research.

Acknowledgments

The work has been partially supported by CENIIT, Linköping University, and the Swedish Research Council. We would like to thank the reviewers and the journal editor for their valuable comments that have enabled us to improve the paper's content and clarity.

References

- [1] 3GPP. TR 25.913 – v7.3.0, Requirements for EUTRA and EUTRAN, 3G Release 7, 2006.
http://www.3gpp.org/ftp/specs/archive/25_series/25.913/.
- [2] 3GPP. TR 36-series, 3G Release 8, 2008.
http://www.3gpp.org/ftp/specs/archive/36_series/.
- [3] E.H.L. Aarts and J.K. Lenstra, *Local Search in Combinatorial Optimization*, Wiley, 1997.
- [4] I.F. Akyildiz, J.S.M. Ho and Y.B. Lin, Movement-based location update and selective paging for PCS networks, *ACM/IEEE Transactions on Networking*, vol. 4, pp. 629-638, 1996.
- [5] I.F. Akyildiz, J. McNair, J.S.M. Ho, H. Uzunalioglu, and W. Wang, Mobility management in next generation wireless systems, *Proc. of IEEE*, vol. 87, pp. 1347-1384, 1999.
- [6] S.Z. Ali, Design of location areas for cellular mobile radio networks, in *Proc. of IEEE Vehicular Technology Conference (VTC '02)*, pp. 1106-1110, 2002.

- [7] Y. Bejerano, M.A. Smith, J. Naor, and N. Immorlica, Efficient location area planning for personal communication systems, *ACM/IEEE Transaction on Networking*, vol. 14, pp. 438-450, 2006.
- [8] P.S. Bhattacharjee, D. Saha, A. Mukherjee, and M. Maitra, Location area planning for personal communication services networks, in *Proc. of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '99)*, pp. 95-98, 1999.
- [9] E. Cayirci and I.F. Akyildiz, Optimal location area design to minimize registration signaling traffic in wireless systems, *IEEE Transactions on Mobile Computing*, vol. 2, pp. 76-85, 2003.
- [10] J.M. Chang, C.Y. Chang, T.H. Lin, and H.C. Chao, Using a novel efficient location management approach in cellular networks, in *Proc. of the 1st IEEE International Conference on Ubi-Media Computing*, pp. 149-154, 2008.
- [11] A. Charnes and W.W. Cooper, Goal programming and multiple objective optimization, *European Journal of Operational Research*, vol. 1, pp. 39-45, 1977.
- [12] I. Demirkol, C. Ersoy, M.U. Caglayan, and H. Delic, Location area planning in cellular networks using simulated annealing, in *Proc. of IEEE Conference on Computer Communications (INFOCOM '01)*, pp. 13-20, 2001.

- [13] I. Demirkol, C. Ersoy, M.U. Caglayan and H. Delic, Location area planning and cell-to-switch assignment in cellular networks, *IEEE Transactions on Wireless Communications*, vol. 3, pp. 880-890, 2004.
- [14] M. Ehrgott and X. Gandibleux (editors), *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, Kluwer Academic Publishers, 2002.
- [15] P.C. Fishburn, Lexicographic orders, utilities and decision rules: a survey, *Management Science*, vol. 20, pp. 1442-1471, 1974.
- [16] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman & Co., 1979.
- [17] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Welsey, 1989.
- [18] P.R.L. Gondim, Genetic algorithms and location area partitioning problem in cellular networks, in *Proc. of IEEE Vehicular Technology Conference (VTC '96)*, pp. 1835-1838, 1996.
- [19] Gurobi 3.0.1, Reference Manual, 2010.
- [20] H. Isermann, Proper efficiency and the linear vector maximum problem, *Operations Research*, vol. 22, pp. 189-191, 1974.
- [21] K. Kyamakya and K. Jobmann, Location management in cellular networks: classification of the most important paradigms, realistic simulation framework, and relative performance analysis, *IEEE Transactions on Vehicular Technology*, vol. 54, pp. 687-708, 2005.

- [22] C.Y. Lee, S.J. Kim and T. Park, A design of multi-layered location registration areas in microcellular systems, *Telecommunication Systems*, vol. 14, pp. 107-120, 2000.
- [23] G.L. Lyberopoulos, J.G. Markoulidakis, D.V. Polymeros, D.F. Tsirkas, and E.D. Sykas, Intelligent paging strategies in future mobile telecommunication networks, *IEEE Transactions on Vehicular Technology*, vol. 44, pp. 534-554, 1995.
- [24] E. Martin, L. Liu, M. Weber, P. Pesti, and M. Woodward, Unified analytical models for location management costs and optimum design of location areas, in *Proc. of the 5th International ICST Conference on Collaborative Computing, Networking, Applications and Worksharing (COLLABORATECOM '09)*, 2009.
- [25] S. Mishra and O.K. Tonguz, Most recent interaction area and speed-based intelligent paging in PCS, in *Proc. of IEEE Vehicular Technology Conference (VTC '97)*, pp. 505-509, 1997.
- [26] S. Modarres Razavi and D. Yuan, Performance improvement of LTE tracking area design: a re-optimization approach, in *Proc. of the 6th ACM International Workshop on Mobility Management and Wireless Access (MobiWac '08)*, pp. 77-84, 2008.
- [27] S. Modarres Razavi, D. Yuan, F. Gunnarsson, and J. Moe, Optimizing the tradeoff between signaling and reconfiguration: a novel bi-criteria solution approach for revising tracking area design, in *Proc. of IEEE Vehicular Technology Conference (VTC '09)*, 2009.

- [28] IST-2000-28088 MOMENTUM project. <http://momentum.zib.de>.
- [29] Z. Naor and H. Levy, Minimizing the wireless cost of tracking mobile users: an adaptive threshold scheme, in *Proc. of IEEE Conference on Computer Communications (INFOCOM '98)*, pp. 720-729, 1998.
- [30] J. Plehn, The design of location areas in a GSM network, in *Proc. of IEEE Vehicular Technology Conference (VTC '95)*, pp. 871-875, 1995.
- [31] G.P. Pollini and C.L. I, A profile-based location strategy and its performance, *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 1415-1424, 1997.
- [32] C. Rose and R. Yates, Minimizing the average cost of paging under delay constraints, *Wireless Networks*, vol. 1, pp. 211-219, 1995.
- [33] J.D. Schaffer, Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, *International Conference on Genetic Algorithms*, pp. 93-100, 1985.
- [34] S.K. Sen, A. Bhattacharya and S.K. Das, A selective location update strategy for PCS users, *Wireless Networks*, vol. 5, pp. 313-326, 1999.
- [35] R.E. Steuer, *Multiple Criteria Optimization: Theory, Computation, and Application*, John Wiley & Sons, 1986.
- [36] R.E. Steuer, L.R. Gardiner, and J. Gray, A bibliographic survey of the activities and international nature of multiple criteria decision making, *Journal of Multi-Criteria Decision Analysis*, vol. 5, pp. 195-217, 1996.

- [37] E.G. Talbi, *Metaheuristics: From Design to Implementation*, John Wiley & Sons, 2009.
- [38] D.W. Tcha, T.J. Choi, and Y.S. Myung, Location-area partition in a cellular radio network, *Journal of the Operational Research Society*, vol. 48, pp. 1076-1081, 1997.
- [39] W. Wang, I.F. Akyildiz, G.L. Stüber and B. Chung, Effective paging schemes with delay bound as QoS constraints in wireless systems, *Wireless Networks*, vol. 7, pp. 455-466, 2001.
- [40] T. Winter, Mobility management and network design for UMTS, in *Proc. of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '04)*, 2004.
- [41] V.W.S. Wong and V.C.M. Leung, An adaptive distance-based location update algorithm for PCS networks, in *Proc. of IEEE International Conference on Communications (ICC '01)*, pp. 2001-2005, 2001.
- [42] E. Zitzler and S. Kunzli, Indicator-based selection in multi-objective search, in *Proc. of the 8th International Conference on Parallel Solving from Nature (PPSN '04)*, pp. 832-842, 2004.