Online Algorithms for the Server and Service Location Problem in 3G Mobile Networks

Olga Goussevskaia, Danielle G. Valente,
Geraldo R. Mateus and Antonio A.F. Loureiro

Department of Computer Science,
Federal University of Minas Gerais, Brazil

olga,dgv,mateus,loureiro@dcc.ufmg.br

Abstract. *How to design an infra-structure that supports multi-service provision and demand-driven resource distribution is a challenging problem that arises in third-generation mobile networks. One of the proposed formulations for this problem has been named as the Server and Service Location Problem. Due to the problem's high computational complexity, heuristic techniques have been proposed to solve it. The time needed to solve large instances of the problem, however, remained unacceptably high, since the attendance of the demand must be made in real time, minimizing the response delays. In this work we emphasize the online nature of the problem. Two online algorithms of much lower computational complexity were implemented, and the experimental results revealed that, for high numbers of mobile users, their performance overcomes the heuristics implemented so far, both in response time and solution quality.*

1. Introduction

The rapid advance of component technology, the pressure to integrate fixed and mobile networks, the developments in the domains of service engineering, network management and intelligent networks, and the desire to have multi-application handheld terminals demand performance improvements beyond the capacity of second-generation technology. Factors such as large number of users with different mobility behaviors in different geographic areas and times of day become critical for the resources to be used efficiently.

There are two distinct entities that we can consider in 3G networks: Radio Base Stations and Servers that providing some specific services. Physically, we consider that these two entities have the same location. This is a situation that arises in content distribution applications, for example. It is a typical scenario where the company Akamai [Akamai,], which analyzes strategic points of the network for content and application distribution, operates. Another possible scenario is when there is a maximum number of active software licences for content or application access. In this case, a licence can become available or unavailable according to changes in demand distribution over time or space. There are costs that are associated with different activities, such as hardware or communication maintenance of the servers. In a content provision scenario, the activation costs of a server can be related to up-to-date information maintenance of a data base system, for example. In such an environment, service provision becomes an important problem. The number of mobile users is variable and they move through attendance regions requesting different kinds of services in different time periods. In this scenario, the network should be able to determine to which server each client should be allocated at different moments according to some metric, such as the distance bridged by the wireless connection, the traffic distribution among servers or type of service being provided. The number of active servers and their locations must be optimized in order to minimize both the costs of server activation and service provision. This problem is known as the Server and Service Location Problem and was discussed for the first time in [Mateus et al., 2000].

This is a combinatorial problem, and, thus, it is very difficult to be guaranteed an optimal solution. The use of heuristics becomes necessary in order to obtain acceptable solutions in polynomial time. In [Mateus et al., 2001], a Lagrangean Relaxation technique [Fisher, 1981, Fisher, 1985, Fisher et al., 1975, Galvão and Santibañez-Gonzales, 1992, Geoffrion, 1978, Reeves, 1993] is used for this purpose and near-optimal solutions were obtained in relatively short time periods. The quality of the Lagrangean heuristic, however, significantly decays when a large number of mobile units is involved. Moreover, the response time of the algorithm gradually increases as the number of variables grows, compromising the system's ability to provide an answer in a reasonable time.

In this case, the decisions have to be made dynamically, in real time and without previous knowledge of future requests, what leads to an online version of the problem. An alternative approach would be to solve the service allocation problem each time a new user enters the system or the demand configuration changes, always maintaining the response time at acceptable levels.

In this work, initially we show the integer programming formulation of the Server and Service Location Problem. Afterwards, an extension for the problem is proposed, aiming to adapt it to a more realistic mobile computing environment, where time restrictions are crucial for its service maintenance. Two greedy online algorithms were implemented to substitute the Lagrangean Relaxation. This new approach has been called the *Online Approach*. The experimental results revealed that, for a large number of mobile users, the performance of the *Online Approach* overcomes the solutions proposed up to this moment, both in response time and solution quality, mainly due to the problem's high computational complexity.

The rest of this paper is organized as follows. In Section 2, we give the formulation of the Server and Service Location Problem. In Section 3, we discuss how online algorithms work and their importance in this scenario. Then, in Section 4, we present the two algorithms proposed to substitute the Lagrangean Relaxation. In Section 5, we present the simulation environment to conduct our experiments and the experimental results. Finally, Section 6 contains our conclusions and future work considerations.

2. Modeling

The Server and Service Location Problem consists of an integer programming model, which aims to minimize equipment maintenance and service provision costs by dynamically defining the optimal number and location of active servers in the system and allocating the demand generated by mobile users to them.

The following notation is used:

- T: set of service classes;
- I: set of servers;
- U: set of active mobile units;
- d_j^t: binary parameter that expresses whether the mobile unit $j \in U$ is demanding a service $t \in T$ or not;
- p^t: number of servers to be selected for every service $t \in T$;
- c_{ij}^t: variable cost of attending the demand d_j^t by server $i \in I$; it is location depended, but is more general than just the distance between the mobile unit and the server. It also dependends on other parameters, such as the number of channels required to provide this service, or the amount of data that has to be transmitted.
- f_i^t: cost of activating server $i \in I$ to provide service $t \in T$; it can depend on the geographic location of the server or on the set of services that it provides. Considering a scenario where the server is running some content provision service, this cost could be associated to some hardware maintenance or up-to-date information maintenance.
- x_{ij}^t: boolean variable, which is set to 1 when the mobile unit $j \in U$, requiring service $t \in T$, is attended by the facility $i \in I$, and set to 0, otherwise;

y_i^t: boolean variable, which is set to 1 when server $i \in I$ is active to provide service $t \in T$, and set to 0, otherwise;

The problem is formulated as follows:

(M):

$$min \sum_{t \in T} \sum_{i \in I} \sum_{j \in U} c_{ij}^t x_{ij}^t + \sum_{t \in T} \sum_{i \in I} f_i^t y_i^t \quad (1)$$

subject to:

$$\sum_{i \in I} x_{ij}^t = d_j^t \ , \quad \forall j \in U \ , \forall t \in T \ , \quad (2)$$

$$\sum_{i \in I} y_i^t \geq p^t \ , \quad \forall t \in T \ , \quad (3)$$

$$x_{ij}^t \leq y_i^t \ , \quad \forall t \in T \ , \forall i \in I \ , \forall j \in U \ , \quad (4)$$

$$x_{ij}^t \in \{0,1\} \ , \quad \forall t, \forall i \in I \ , \forall j \in U \ , \quad (5)$$

$$y_i^t \in \{0,1\} \ , \quad \forall t \in T \ , \forall i \in I. \quad (6)$$

The objective function (1) minimizes the total cost of activating the selected servers and attending the active mobile units. The set of constraints (2) ensures that each active mobile unit demand for each class of service is attended by a unique server. The set of constraints (3) imposes a minimum number of servers that must be selected for each class of service. The set of constraints (4) guarantees that no mobile unit will be allocated to a non-selected server. Finally, sets (5) and (6) of constraints guarantee the integrality of variables.

The constraint on how far the mobile unit could be away from the server geographically, or in other words, the definition of each server's covering region, is not part of the mathematical model that we present. However, it could easily be incorporated, implying no significant changes on the optimization algorithm, by adding a signal propagation model as a binary matrix A, defining whether a particular demand point d_j^t is covered or not by server y_i. The set of constraints (2) would look like this:

$$\sum_{i \in I} a_{ij} x_{ij}^t = d_j^t \ , \forall j \in U \ , \forall t \in T \quad (7)$$

The capacity in terms of server load is not part of the model either. However, this only becomes a problem in scenarios with extreme traffic overloads.

3. Online Algorithms

Online algorithms are algorithms designed to solve problems where the input data is not known in advance, but it is supplied during the execution of the algorithm [Albers, 2003, Albers, 1996, Ben-David and Borodin, 1994, Goemans, 1994, Irani and Karlin, 1996]. These algorithms are defined in the context of optimization problems, where there is a set $\mathcal{I}$ of inputs and a set $\mathcal{O}$ of outputs, and, for a given input $I \in \mathcal{I}$, we want to obtain an output $O \in \mathcal{O}$. A cost, or a profit, is associated with each pair (I, O). The objective is to maximize the profit or to minimize the total cost, depending whether it is a maximization or a minimization problem.

In an online problem, the input is given as a sequence $\sigma^i = \sigma_1 \sigma_2 \ldots \sigma_i$ of requests or events. The algorithm takes each input and must compute a response after each event, providing an output represented as a sequence $a^i = a_1 a_2 \ldots a_i$. The response can only depend on the previously received input, meaning that there must exist a function f_i such that: $a_i = f_i(\sigma^i)$ [Albers, 1996, Ben-David and Borodin, 1994].

3.1. Performance Evalutaion

It is important to know how to evaluate the performance of an online algorithm. The traditional worst-case analysis is not useful, since it causes any algorithm to achieve its worst-case. A well-known metric to analyze online algorithms is the Competitive Analysis [Ben-David and Borodin, 1994,

Goemans, 1994, Irani and Karlin, 1996]. In this approach, an online algorithm A is compared to an optimal offline algorithm OPT. The OPT algorithm knows the entire input data in advanced and can solve the problem with minimal cost.

Given an input sequence σ, let $C_A(\sigma)$ denote the cost associated with algorithm A, and $C_{OPT}(\sigma)$ the cost associated with algorithm OPT. The algorithm A is said to be c-competitive if there exists a constant $a \geq 0$ such that

$$C_A(\sigma) \leq c \cdot C_{OPT}(\sigma) + a \tag{8}$$

for every input sequence σ. It is assumed that A is a deterministic online algorithm. The factor c is also known as the competitive ratio of A.

3.2. Design Principles

The online algorithm design is an intuitive job, with no defined rules. Some of the principles frequently used are presented in this section.

Greedy Algorithms. Online algorithms generate a response immediately after each request. Hence, in a way, these algorithms are intrinsically greedy. They can be defined by a function f of the current request, the current state and the history. The algorithm evaluates the function for each possible way of serving the current request, and chooses the action, which minimizes the value of the function.

Following this idea, there are three types of greedy algorithms that present their best performance in different situations. The first type is the local greedy algorithm. It simply chooses an action that minimizes the cost of serving the current request. The second type is the retrospective algorithm. It tries to approximate the online solutions to the solutions that an optimal algorithm would generate if the sequence seen up to that moment (including the current request) were the entire sequence. To achieve this, it keeps the history of all states up to the current moment. The third type is the work function algorithm, which has been distinguished for its ability to find optimal and near-optimal solutions. It is a combination of the first two types of greedy algorithms.

Balancing Costs. In order to choose how to serve the next request, the balancing costs approach says that an online algorithm should roughly try to incur the same cost on each of the possible future inputs. In this scenario, no matter what the future will be, the algorithm tries to make choices that minimize the average of future costs, nevertheless the current cost may not be minimal.

Combining Online Algorithms. There may be several solutions that use online algorithms to solve a given problem. Each of these solutions can be competitive only for a subset of possible inputs. The union of all this subsets would cover all the possible inputs. In this way one can build a generic online competitive algorithm for all possible inputs.

3.3. Examples of Online Problems

Some online problems have been extensively studied, specially after the work of Sleator and Tarjan [Sleator and Tarjan, 1985]. Among them the Ski Rental Problem and the K-Server Problem [Albers, 2003, Goemans, 1994, Irani and Karlin, 1996] are of special interest.

The Ski Rental Problem. Consider a skier that does not know when the ski season is going to end. The skier has to rent the skis, paying \$30 per day, or buy the skis for \$300. The problem is to choose between renting or buying the skis. If the length of the season is known in advance (the offline case), say D, then the obvious solution will be to rent the skis if $D < 300$ or to buy them if $D \geq 300$. In this formulation, the skis are not a commodity, but a necessary item to the practice of the sport. This is a typical online problem, in which a solution is computed on each event and the future is uncertain [Goemans, 1994].

The main idea of the online algorithm is to approximate the performance of the optimal offline algorithm, which knows the future and is able to make decisions to minimize the total cost.

The K-Server Problem. Let K be a set of mobile servers, each of which occupies a single point in a fixed metric space M. A sequence of requests is given and each request must be served before the next request appears. To serve a request at location $x \in M$, the algorithm must move a server to x, unless there is already a server at location x. Whenever the algorithm moves a server from point a to point b, a cost $d_{a,b}$ is associated with this operation, reflecting the distance between two points in the metric space. The objective is to minimize the total distance covered by the mobile servers [Goemans, 1994, Goldbarg and Luna, 2000, Irani and Karlin, 1996].

The Ski Rental and the K-Server problems were presented here in order to show the complexity and importance of online problems. The complexity arises from the necessity of solving problems without any previous knowledge of future input data. It means that, as every new request arrives, the algorithm must compute a new solution, aiming to obtain the least possible cost by the end of the request. The importance of online problems lies in their application to real-time environments, such as the Server and Service Location Problem, studied in this work.

4. The Online Approach

The Server and Service Location Problem, formulated in Section 2, can be viewed as an online problem. The attendance of the demand, which changes unpredictably, must be made in real time, minimizing the response delays. In this section, an alternative way of solving this problem is proposed, emphasizing its online nature.

Given an input set $\sigma^n = \sigma_1, \sigma_2, \ldots, \sigma_n$, where the σ_j corresponds to a new request made by client $j \in U$, and, for some service (d_j^t), each request must be attended immediately. It means that client $j \in U$ must be allocated to a server before the processing of the following request. This allocation must be based solely on the data known up to that moment, aiming to minimize the cost of configuration σ^n, given by:

$$\sum_{t \in T} \sum_{i \in I} \sum_{j \in U} c_{ij}^t x_{ij}^t + \sum_{t \in T} \sum_{i \in I} f_i^t y_i^t \tag{9}$$

and the constraints (2), (3), (4), (5) and (6) must be met.

The Lagrangean Relaxation technique, used to approximately solve the Server and Service Location Problem, provides near-optimal solutions in relatively short time periods. The quality of this heuristic, however, significantly decays when a large number of mobile units is involved. Moreover, the response time of the algorithm gradually increases as the number of variables grows, compromising the quality of service of the system. In this section we propose two greedy algorithms to solve the Server and Service Location Problem. The computational complexity of both of these algorithms is much lower then the Lagrangean Relaxation's, what enabled significantly lower response delays, emphasizing the online nature of the problem.

4.1. Greedy Algorithm 1

Following some design principles presented in Section 3.2, this algorithm is an implementation of a greedy approach [Mettu and Plaxton, 2000]. One important characteristic is that it maintains the same computational cost for all inputs. This allows a constant response delay no matter the configuration of the system and its complexity.

The basic idea of this algorithm is to allocate every new service request to a server that has the nearest geographical location to the point from where the request has been generated, and it has the least activation cost, giving priority to those servers that are already active by subtracting their activation cost from the total evaluated cost. A search, though, is made among the candidate servers, and the one that has the minimal *total cost*, which is equal to the sum of its *service attendance cost* and its *activation cost*, is allocated to the request σ^j originated at the mobile unit $j \in U$. Then,

for each request σ^j from $j \in U$, requiring service $t \in T$, the total cost in server i is given by: $total\_cost_i = c_{ij}^t + f_i^t$

Those servers that are already active, however, are considered to have a null activation cost. This algorithm does not have a memory, since the requests that have already been attended by the system are not taken into account. Its pseudocode is given in Figure 1.

Pseudocode *Greedy Online Server and Service Location (1)*

```
repeat
    read requested service and location of mobile unit j ∈ U
    for all candidate servers i ∈ I do
        if server i is already active then
            activation_cost_i = 0;
        else
            compute activation_cost_i;
        end_if
        total_cost = activation_cost_i + service_provision_cost_ij;
        if total_cost < minimal_cost then
            minimal_cost = total_cost;
            best_server = i;
        end_if
        end_for
        allocate the mobile unit j to the minimal cost server i,
                 or best server;
 end_repeat
end_pseudocode
```

Figure 1: Greedy algorithm 1.

4.2. Greedy Algorithm 2

The second greedy algorithm uses a strategy with memory. All the attended requests originated by the mobile units up to a certain moment are stored by the system. Using this information it becomes possible to give a priority to those servers that have a greater number of mobile units allocated to them. This priority is made by decreasing (multiplying) the activation costs of these servers by a factor *fix_descr*, defined as:

$$fix\_decr_i = \sum_{t \in T} \sum_{j \in U} x_{ij}^t / \sum_{t \in T} \sum_{j \in U} \sum_{k \in I} x_{kj}^t, \forall i \in I$$

The service attendance costs are decreased according to the distance between the origins of the requests and the locations of the candidate servers. The servers are divided into three groups. The first group is composed of the nearest servers and receives the lowest *service attendance cost*. The second group is the neighbours of the first group and they receive slightly higher costs. The third group are the remaining servers that do not have their costs reduced at all. This strategy intends to give priority to the nearest servers. If server $i \in I$ is located inside the same *area zone* (defined in Section 5.1) where the request has been generated, then the *service provision cost* is decreased by a factor *var_decr*, given by: $var\_decr_i = 20\%$.

Pseudocode *Greedy Online Server and Services Location (2)*

```
repeat
    read requested service and location of mobile unit j ∈ U
    for all candidate servers i ∈ I do
        compute fix_decr_i
        compute var_decr_i
        compute activation_cost_i;
        compute service_provision_cost_ij;
        total_cost = (activation_cost_i × fix_decr_i) +
                     (service_provision_cost_ij × var_decr_i);
        if total_cost < minimal_cost then
            minimal_cost = total_cost;
            best_server = i;
        end_if
    end_for
    allocate the mobile unit j to the minimal cost server i;
    memorize this allocation
 end_repeat
end_pseudocode
```

Figure 2: Greedy algorithm 2 (with memory).

If server $i \in I$ is located inside a neighbor (adjacent) area zone where the request has been generated, then: $var\_decr_i = 10\%$. The pseudo-code for this algorithm is given in Figure 2.

5. Experiments

In this section, we present the simulation environment used to conduct the experiments and the experimental results.

5.1. Simulation Environment

In this work, we use the mobility simulator described in [Mateus et al., 2003] to represent a hypothetical city, a typical contemporary metropolitan area. The city was divided into geographical zones (e.g., a city center and some suburbs), and populated with different groups of mobile units, whose mobility behavior, and demand for different kinds of services was generated to simulate a typical 24-hour day input data for the model. Given the generated demand at a certain period of time, a server allocation is made in order to attend this demand.

The geographic area was simulated as a twenty-kilometer-radius radial city, composed of four area types: city center, urban, suburban, and rural. Figure 3 is a representation of such a city. The model consists of 32 area zones (network areas), eight per city area type, which are connected via high-capacity routes, representing the most frequently selected streets for movement support. Four peripheral and four radial routes are defined. Three types of movement attraction points (locations that attract population movements and at which people spend considerable time periods) are distributed over the city area. Examples are workplaces, residences and commercial areas. Figure 4 presents the assumed distribution of movement attraction points over the whole city area.

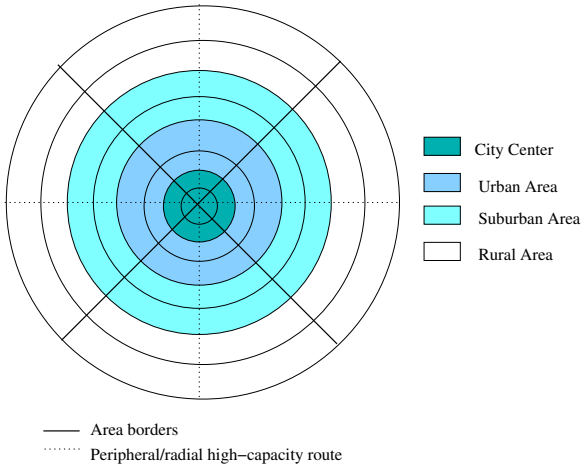

Figure 3: City area model consists of area zones connected via high-capacity routes.

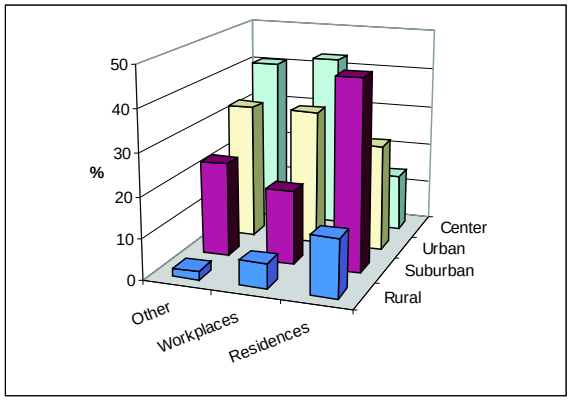

Figure 4: Distribution of movement attraction points over the city area.

The population was divided into mobile unit groups according to the mobility characteristics of the individuals and to the kind of demand they generate. The demand was generated for various types of services, such as Voice, Video and Web requests. Examples of user groups are: 24-hour delivery boy, common worker, housekeeper and taxi driver. A call distribution and a set of requested services were associated with each group. Some examples are:

- *24h Delivery Boy*: Call Distribution: Poisson with mean equal to 10 min; Service: Voice with duration exponentially distributed with mean equal to 80 s.
- *Common Worker*: Call Distribution: Poisson with mean equal to 14 min; Services: Voice with duration exponentially distributed with mean equal to 80 s, Web with duration exponentially distributed with mean equal to 180 s, Video duration normally distributed with mean equal to 1 h.

A movement table was associated with each of these groups in order to determine their mobility behavior. This was done by dividing the day into time periods, and associating a probability

with a group being at a certain location at a given moment. Tables 1 and 2 show two examples of movement tables.

Common Worker	
00:00 - 07:00	Home
07:00 - 12:00	Work
12:00 - 13:30	Work 75% Busi 20% Shop 5%
13:30 - 18:00	Work
18:00 - 22:00	Home 70% Shop 15% Busi 15%
22:00 - 24:00	Home 80% Random 20%

Table 1: Movement table of a Comon Worker.

Taxi Driver	
00:00 - 07:00	Home
07:00 - 08:00	Busi
08:00 - 09:00	Random
09:00 - 10:00	Res
10:00 - 11:00	Busi
11:00 - 12:00	Res
...	...

Table 2: Movement table of a Taxi Driver.

Table 1 shows that, between 12:00 and 13:30, a common worker has a 75% probability of being at a work location, a 20% probability of being at an area with high concentration of business attraction points, and only a 5% probability of being at a shopping center. On the other hand, in Table 2, a taxi driver can be at any location between 8:00 and 9:00.

5.2. Methodology

In order to evaluate the performance of the proposed algorithms, three different allocation approaches were simulated, and their performances compared.

Common Allocation Approach. This approach, implemented in [Mateus et al., 2003], is a simulation of the way demand attendance is performed in today's traditional cellular networks. The geographic area is divided into cells and usually one server (Radio Base Station) is responsible for the service provision to all mobile units physically inside the cell region. Each area zone of the simulated city area was treated as a separate cell.

Online by Time-Step or Lagrangean Allocation Approach. This approach, also implemented in [Mateus et al., 2003], has as its main characteristic a periodic reconfiguration of the system's resources (a sequence of activations and deactivations of servers) based on the current demand distribution, and the current location of mobile units.

Let a *time-step* be a fixed amount of time. All information regarding the system's current state (i.e., which services are being requested, and from which locations) is collected during a *time-step* and then is used to built a Server and Service Location Problem for each *time-step*. The Lagrangean Relaxation technique is then applied to find the closest solution to the optimum. The solution is then used to reconfigure the system's state, what means the demand is attended by the lowest cost at that instant.

We suppose that all state transitions occur at the beginning of a *time-step*. If we consider a *time-step* as the inter-arrival time of two service requests then the *Online by Time-Step Allocation Approach* reduces to the *Greedy Allocation Approach*, which is described in the following section.

Online or Greedy Allocation Approach. In this work, this approach is being proposed as a simplified way of solving the Server and Service Location Problem. In spite of its simplicity, it is more efficient in terms of response delays. The problem is solved each time a new service request is received by the system, always maintaining the response time at acceptable levels. The two greedy algorithms, described in Section 4, were used to allocate each new service request to the most suitable (at least locally considered) server.

5.3. Experimental Results

This section evaluates the three allocation approaches described above using simulation.

System Costs. The evaluation of system costs was performed through the values of the objective function of the integer programming problem. It means that all costs involved in server activation (f_i^t), and all costs involved in demand attendance (c_{ij}^t), were taken into account.

Figures 5 to 10 show a comparison among the costs of *Common*, *Online by Time-Steps* or *Lagrangean* and *Online* or *Greedy* allocation approaches. Results for different numbers of mobile units, and different cost proportions are analyzed. In Figures 5, 6, and 7 fixed costs are considered to be, in average, 10 times higher than variable costs, what makes the system more sensible to server activation costs. In Figures 8, 9, and 10, on the other hand, variable costs (c_{ij}^t) are considered to be, in average, 10 times higher than fixed costs (f_i^t). This proportion makes the system configuration more sensible to the distances between mobile units and servers.

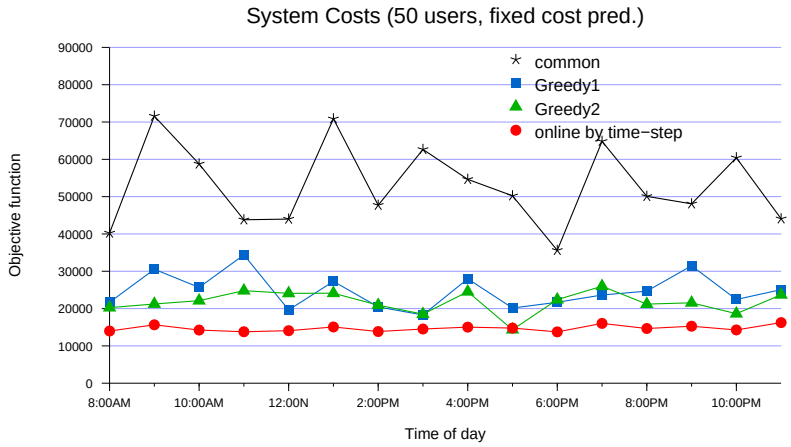

Figure 5: System costs (50 users, fixed cost predominance).

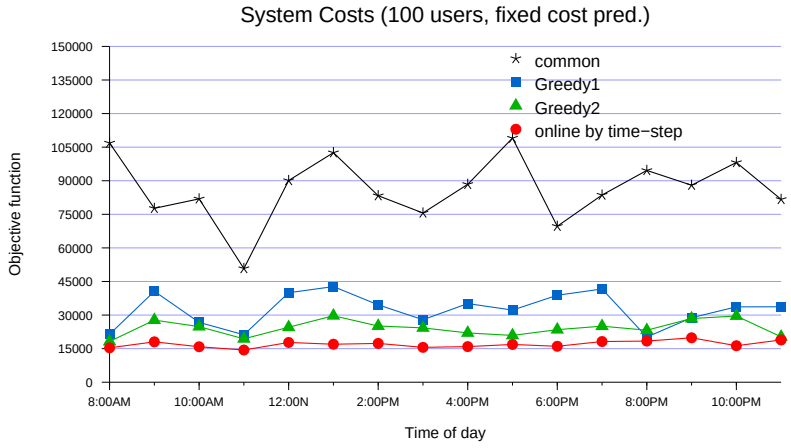

Figure 6: System costs (100 users, fixed cost predominance).

In all six figures, the predominance of *Common Approach* costs over *Online by Time-Step Approach* and *Greedy Approach* costs can be seen. This result could be predicted, since the *Common Approach* does not take any cost evaluation into account during its demand attendance process. Comparing Figures 5, 6, and 7, it can be seen that, as the number of mobile units increases, the relation between the costs achieved by *Common Approach* and the other two approaches remains and becomes even more explicit. This is due to the fixed cost predominance and the fact that *Common Approach* does not take them into account when performing the server allocation. While the optimization algorithms try to allocate the maximum number of users at each server, in order to minimize

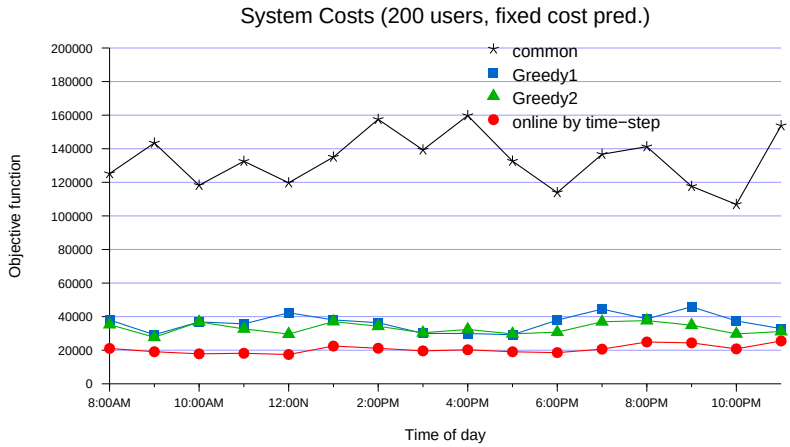

Figure 7: System costs (200 users, fixed cost predominance).

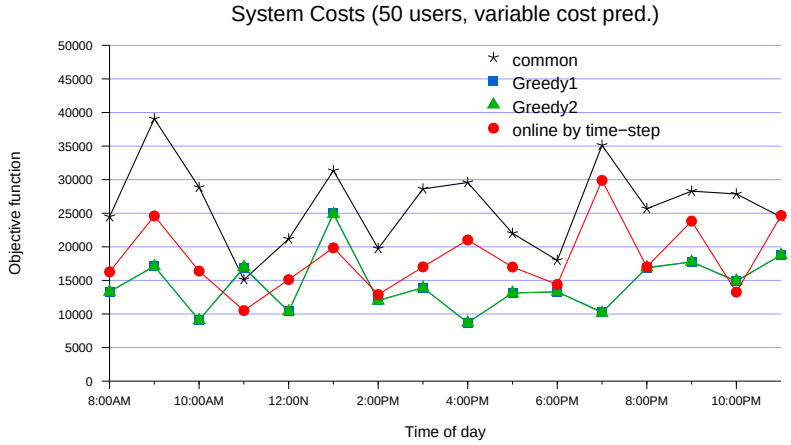

Figure 8: System costs (50 users, variable cost predominance).

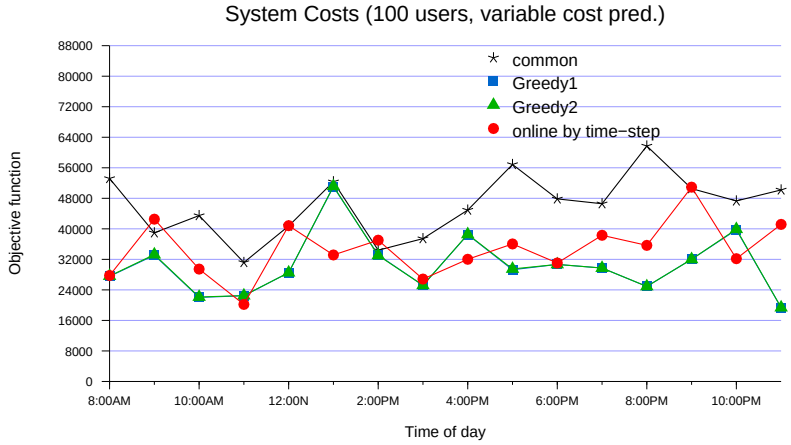

Figure 9: System costs (100 users, variable cost predominance).

the high activation costs, the *Common Approach* just activates them all.

The same fact, however, cannot be observed comparing Figures 8, 9 and 10. In those figures

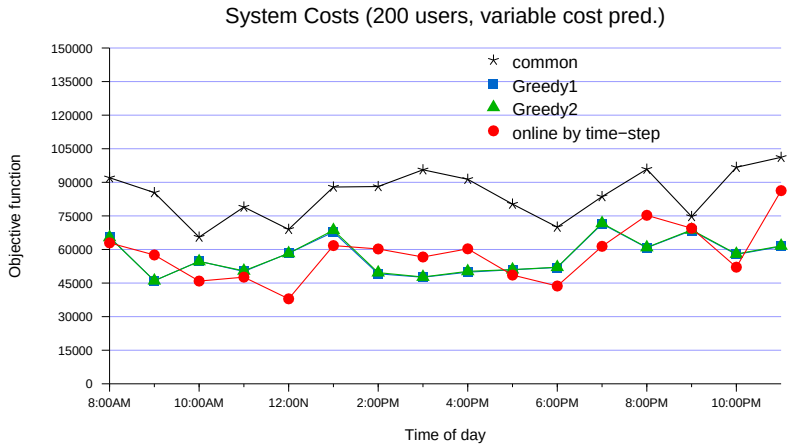

Figure 10: System costs (200 users, variable cost predominance).

the differences between the three approaches are no longer so explicit. This is due to the variable cost predominance, since it significantly increases the problem's complexity (because the number of local minimums increases). The Lagrangean Relaxation, in this case, is not able to achieve the optimal solution anymore, remaining with an approximated solution after a certain amount of iterations. This fact is illustrated in Figure 11. It can be observed that, when variable costs predominate and the number of mobile units is increased, the percentage of optimal solutions obtained by Lagrangean Relaxation decays. If for 50 mobile units the heuristic achieves the optimum in almost 100% of the cases, for 1000 mobile units it remains with an approximate solution in more than 80% of the cases. The performance of the *Greedy Approach* also decays due to its inability to analyze the combinatory nature of the problem, which becomes much heavier in this scenario. It is worth noting, however, that as the number of mobile units increases, the performance of the *Greedy Approach* approximates to the performance of Lagrangean Relaxation. And, as it will be shown in the next section, the delay caused by the *Greedy Approach* is significantly lower than the one caused by Lagrangean Relaxation.

Execution Times. Figures 12 and 13 demonstrate the time delays of the three allocation approaches. It can be seen that the time spent by *Common* and *Online or Greedy* approaches are almost constant, whereas Lagrangean Relaxation's time (*Online by Time-Step Approach*), in spite of being polynomial, significantly increases as the number of mobile units is increased, in particular when variable costs predominate.

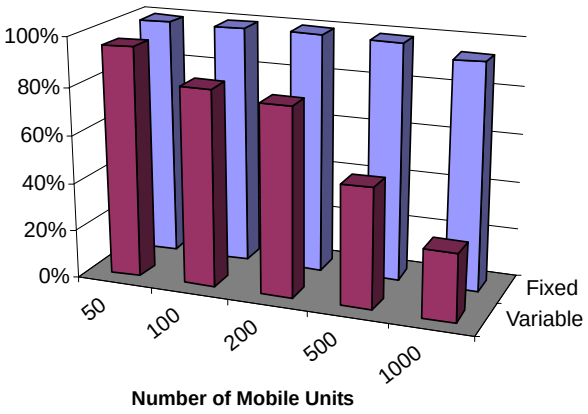

Figure 11: % of optimal solution for Lagrangean Relaxation.

Since the Server and Service Location Problem is intrinsically an online problem, and response delays are crucial for maintaining the quality of service of the system, it can be deduced that, when the system must support more than 500 mobile users, the *Online or Greedy Approach* is more appropriate, since the quality of its solutions is close to that of Lagrangean Relaxation and its execution time remains constant as the number of users increases.

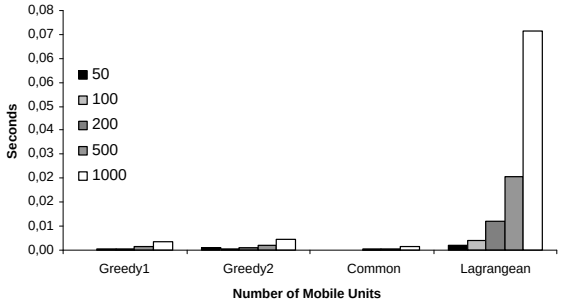

Figure 12: Average exec time for different # of users (fixed cost predominance).

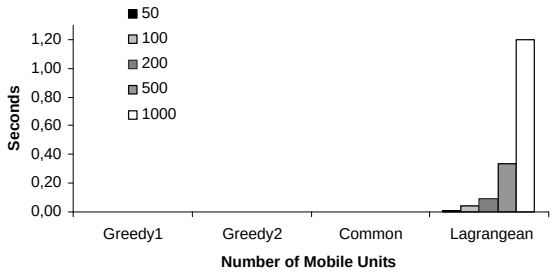

Figure 13: Average exec time for different # of users (var cost predominance).

Competitive Analysis. To make the competitive analysis of the developed online algorithms, their results were compared to the Lagrangean Relaxation solution. The competitive ratio of the first greedy algorithm, c_1, was calculated as the highest value among all ratios between the total costs achieved by this algorithm and the total costs achieved by Lagrangean Relaxation in the *Online by Time-Step Approach* at each *time-step*: $c_1 = 2.61$.

The competitive analysis of the second greedy algorithm, c_2, was obtained in a similar manner: $c_2 = 2.24$.

It means that the costs achieved by the greedy algorithms are, in the *worst case*, c_1 and c_2 times higher than those achieved by the Lagrangean Relaxation. This result shows that these algorithms have quite similar efficiency when compared to a more sophisticated technique such as the Lagrangean Relaxation and present an attractive alternative for solving the Server and Service Location Problem in a real time environment.

6. Conclusions and Future Work

The number and the location of active servers in a mobile computing environment that guarantee the attendance of mobile unit requests efficiently and minimize the costs involved has been defined as the Server and Service Location Problem. The solutions proposed up to this moment need a previous knowledge of all or at least part of the inputs. Moreover, their computational complexity is unacceptably high when the number of variables grows. These conditions may not be acceptable or even possible in a real system, where the future is uncertain and users expect fast responses to their requests.

In this work, we presented an extension for the Server and Service Location Problem, aiming to adapt it to a more realistic mobile computing environment, where time restrictions are crucial for quality of service maintenance. Two greedy algorithms were proposed, implemented, and evaluated for the extended problem. The computational complexity of both of these algorithms is much lower then that of the heuristics proposed up to this moment. The experimental results revealed that, for a high number of mobile users, their performance overcomes the heuristics implemented so far, both in response time and solution quality.

Online problems are frequently defined as metric task systems [Irani and Karlin, 1996]. It is a generic model used to describe a vast class of problems as a sequence of tasks that must be executed. A possible extension for this work is to define the Server and Service Location Problem as a metric task system and solving it using the associated algorithms.

An important modification of the model is to add capacity restrictions on the servers, both in terms of radio coverage, power control and the maximum number of attended users. This improve-

ment would turn the problem even closer to a real 3G mobile computing environment.

An alternative application of the proposed architecture is content distribution. Given the knowledge about demand distribution for different kinds of services, the requested data can be moved to the closest servers to those locations. Additional issues would have to be considered in such an environment, such as: memory availability for the moved contents, content replacement policies, and the minimum number of requests needed for a particular service to be moved/replicated in a different server.

References

Akamai. http://www.akamai.com.

Albers, S. (1996). Competitive online algorithms. volume LS-96-2 of *Lecture Series*. BRICS.

Albers, S. (2003). Online algorithms: A survey. *Mathematical Programmig*, 97:3–26.

Ben-David, S. and Borodin, A. (1994). A new measure for the study of online algorithms. *Algorithmica*, 11(1):73–91.

Fisher, M. (1981). The lagrangean relaxarion for solving integer programming problems. *Management Science*, 27:1–18.

Fisher, M. (1985). An application oriented guide to lagrangean relaxation. *Interfaces*, 15(2):10–21.

Fisher, M., Northup, W., and Shapiro, J. (1975). *Using Duality to Solve Discrete Optimization Problems: Theory and Computational Experience*. North-Holland Publishing Company.

Galvão, R. and Santibañez-Gonzales, E. (1992). A lagrangean heuristic for the p_k-median dynamic location problem. *European Journal of Operational Research*, 58:250–262.

Geoffrion, A. (1978). Lagrangean relaxarion applied to capacitated facility location problems. *AIIE Transactions*, 10:10–20.

Goemans, M. X. (1994). Advanced algorithms. Technical Report MIT/LCS/RSS-27, Massachusetts Institute of Technology.

Goldbarg, M. C. and Luna, H. P. L. (2000). *Otimização Combinatória e Programação Linear: Modelos e Algoritmos*. Campus, Rio de Janeiro.

Irani, S. and Karlin, A. R. (1996). *Approximation Algorithms for NP-Hard Problems*, chapter On Online Computation, pages 521–564. PWS Publishing Company, New York.

Mateus, G. R., Goussevskaia, O., and Loureiro, A. A. (2001). The server and service location problem in the third generation mobile communication systems: Modelling and algorithms. In *Workshop on Mobile Internet Based Services and Information Logistics, Informatik 2001*, pages 242–247, Vienna, Austria.

Mateus, G. R., Goussevskaia, O., and Loureiro, A. A. (2003). Simulating demand-driven server and service location in third generation mobile systems. In *International Conference on Parallel and Distributed Computing*, volume 2790 of *Lecture Notes in Computer Science*, pages 1118–1128, Klagenfurt, Austria. Springer-Verlang.

Mateus, G. R., Loureiro, A. A., Rodrigues, R. C., and Goussevskaia, O. (2000). Server location in mobile computing. In *WCNC'2000 - 2000 IEEE Wireless Communications and Networking Conference*, pages 839–844, Chicago, USA.

Mettu, R. R. and Plaxton, C. G. (2000). The online median problem. In *41st IEEE Symposium on Foundations of Computer Science*, pages 339–348, Redondo Beach, California.

Reeves, C. (1993). *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell, London, UK.

Sleator, D. D. and Tarjan, R. E. (1985). Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208.