Computationally efficient algorithms for location area planning in future cellular systems

P. Demestichas, N. Georgantas, E. Tzifa, V. Demesticha, M. Striki, M. Kilanioti, M. Theologou*

National Technical University of Athens, Computer Science Division, 9 Heroon Polytechniou Str, Zographou 15773 Athens, Greece

Received 26 July 1999; received in revised form 31 January 2000; accepted 1 February 2000

Abstract

Efficient resource utilisation in future cellular systems is related to the control of the signalling load imposed by the location update and paging operations. Important means for controlling this load is the “proper” planning of location areas. In this paper we define and solve a version of the location area planning problem. Starting from a formal definition and an optimal formulation of the problem, three computationally efficient algorithms, adhering to simulated annealing, taboo search, and genetic algorithm paradigm, are presented.

Keywords: Simulated annealing; Taboo search; Genetic algorithm paradigm

1. Introduction

Legacy and future cellular communications systems, e.g. the future versions of the Global System for Mobile communications (GSM) [1,2], or the Universal Mobile Telecommunications System (UMTS) [3–6], have promised to offer to mobile users a variety of highly sophisticated communication services [7,8]. The success of these systems will depend on the cost-effective provision of Quality of Service (QoS) levels that are comparable to those offered by fixed systems. In this respect, an issue that becomes of primary importance is the definition of planning problems that will enable the cost-effective provision of adequate QoS levels, and the use of corresponding advanced methods and software tools [9–12] for solving these problems. This is the general context of our paper.

Fast service access is an important QoS dimension that requires keeping track of the location of the users, while they move around without being involved in calls, and consequently, routing calls to the users regardless of their location. Mobile systems apply the location area concept and the location update procedure, in order to keep track of the user locations [13,14] (Fig. 1(a)). More specifically, cells are grouped into location areas. Periodically, a roaming mobile terminal that is not involved in a call obtains the identifier of the location area to which the “closest” base station belongs. When a change in the location area identifier is sensed, the location update procedure is invoked. By means of this procedure, the identifier of the new location area is stored in the system database.

Likewise, when the network has to route a call to a mobile terminal, the current cell of the terminal should be determined. Mobile systems apply the paging area concept and the paging procedure, in order to obtain this information [13,14] (Fig. 1(b)). The most general version of the paging procedure may be described as follows. Initially, a paging area is selected. In general, a paging area may be defined as a subset of the location area. A paging message is broadcast in the cells of the selected area. The content of the message indicates to the mobile terminal that it has an incoming call. As the mobile terminal responds from a certain cell its exact position, as well as its availability to accept the call, are determined. In case the mobile terminal does not respond to the message, in other words if the mobile terminal is not in the selected subset of the location area, a new paging area is selected, and the procedure is repeated (this is often called as multiple-step paging). Multiple step paging schemes subsume one step paging schemes applied in legacy mobile systems as a special case. More specifically, in the GSM case the paging message is broadcast to all cells in a location area. This means that the paging area is taken to be equal to the location area and that paging is completed in one step.

The location update and paging procedures impose a significant signalling load on the system also transferred over the radio interface, which is a scarce resource.
Hence, in order to provide acceptable QoS levels in the most cost-efficient manner, methods for controlling this load and especially, the impact on the radio interface are required. Important means in this respect is the “proper” planning of location areas [13–15]. In this perspective, in this paper, we address a general version of the location area planning (LAP) problem. Starting points are the general description and formal statement of the problem. In the sequel, the problem is optimally formulated as a 0–1 linear programming problem [16,17]. However, since the optimal formulation falls into the $\mathcal{NP}$-complete [17,18] category, three computationally efficient algorithms are presented. The three algorithms rely on the simulated annealing [19,20], taboo search [21,22], and genetic algorithm [23–26] paradigms. These are well known techniques for determining near-optimal solutions to hard combinatorial optimisation problems.

The work in this paper is highly complementary to previous work related to the control of the signalling load that is imposed by the location update and paging operations. The more relevant work category is targeted to the design of location areas. This category includes algorithms for partitioning the cell layout into location areas, which are based on simulated annealing [15], genetic algorithms [27,28], or greedy techniques [29,30], and algorithms for determining the optimum size of a location area [31,32]. The contribution of this paper is the provision of a concise definition for a version of the location area planning problem, the provision of the corresponding new optimal formulation, and the design of three computationally efficient solutions to the problem. The simulated annealing and genetic algorithm techniques have already been used for different versions of the LAP problem. The difference in our work lies in the different LAP version considered and in the different approach followed by the genetic algorithm in the quest for the near-optimal solutions. The differentiation regards primary the adopted operators of the genetic algorithm and the imposed constraints to our problem. Moreover, in the context of this paper we experiment with the taboo search technique that has not attracted as much attention as the other two optimisation methods.

The development of algorithms to the LAP problem that

![Diagram](image-url)

Fig. 1. (a) Partition of the cell layout to location areas (LAs). Location area crossings, e.g. from LA1 to LA2, result to location updates. (b) Partition of a location area layout, namely, LA3, to paging areas. In a multiple step paging scheme the message is initially broadcast to one paging area. In case of unsuccessful paging a new paging area is selected.
are based on different techniques is motivated by the high complexity associated with the computation of optimal solutions to (modest size instances of) the problem. Hence, the experimentation with different algorithms may also be seen as a practical way for cross-checking the quality of the solutions. Moreover, the work of this paper may be seen as complementary to other methods that may be applied for reducing the location update and paging signalling load (e.g. Refs. [33–37]) as well as any multiple-step paging strategy [38–42].

The rest of this document is structured as follows. Section 2 includes the general description of the location area planning problem that is addressed in this paper. Section 3 includes the corresponding formal statement and Section 4 the optimal formulation of the problem. The three computationally efficient algorithms are presented in Sections 5–7. Finally, Section 8 includes results and Section 9 concluding remarks.

2. General problem description

This section provides the general description of the location area planning problem that is considered in this paper. Issues that will be specified in more detail are the input, the objective and the constraints of the problem.

The problem input should provide the following information: (1) elements of the cell structure of the system, and especially, the pairs of cells that are topologically neighbouring in the layout; and (2) the crossing rates between each pair of neighbouring cells. These may be expressed as the number of mobile users that move from a certain cell to a neighbouring one per time unit, and are determined by the system characteristics and the population distribution and mobility models used [43–47]. Finally, the last point that should be specified by the problem input is the rate of incoming calls per cell, i.e. the number of calls per time unit that are directed to users located in the cell. This figure may be obtained by the expected number of users in the cell, and the rate of incoming calls per user.

The objective of the location area planning problem is to find the partition of cells to location areas that minimises a cost function, which is associated with the location update and paging signalling load that is caused by the partitioning. More specifically, the assignment of two neighbouring cells to different location areas means that the crossings between them will result in location updates. From a different perspective, the incoming calls to a cell of a location area result in paging messages that are transmitted to all the cells of the location area.

The constraints of the location area planning problem impose the following issues: (1) the assignment of each cell to a single location area; and (2) the preservation of the location update and paging signalling load in each cell below a given threshold.

Based on the discussion above we may provide the following general statement of the location area planning problem (Fig. 2): “given the cell layout of the system, the crossing rates between neighbouring cells, the rate of incoming calls per cell, the cost of each location update and paging signalling message, and the maximum allowable location update and paging signalling load per cell, find an allocation of cells to location areas that assigns each cell to a location area, preserves the location update and paging load in each cell below a given threshold, and minimises a cost function, which is associated with the number of location update and paging requests (caused by the selected partitioning) and the cost of each type of message”.

In the next sections, the problem described above is formally stated, optimally formulated and solved by means of three computationally efficient algorithms.

3. Formal problem statement

The cell layout of the system is given in the form of a graph \( G(V, E) \). Each node of set \( V \) corresponds to a cell and
each edge of set \( E \) connects neighbouring cells. Let \( H = \{ h(i,j) \mid (i,j) \in E \} \) provide the crossing rates for all the pairs of neighbouring cells, where \( h(i,j) \) denotes the crossing rates from cell \( i \) towards cell \( j \). Likewise, let \( \Lambda = \{ \lambda_m(i) \mid i \in V \} \) provide the incoming calls to the cells of the system. Moreover, let \( c_l \) and \( c_p \) denote the costs of the location update and paging signalling messages, respectively, and \( s_{\text{max}}(i) (i \in V) \) denote the maximum location update and paging signalling load that may be endured in cell \( i \in V \).

Our objective is to find an allocation of cells to location areas, \( L = \{ V_l \mid 1 \leq l \leq l_{\text{max}} \} \), that minimises a cost function that may be generally represented as \( f(L) \). The \( l_{\text{max}} \) quantity denotes the maximum number of location areas that should be designed (\( l_{\text{max}} < |V| \)). The following main factors contribute to the cost function: (1) the crossings between neighbouring cells that are assigned to different location areas, since they result in location updates; and (2) the incoming calls to the cells of a location area, since (in the worst case) the resulting paging messages are broadcast to all the cells of that location area.

The constraints of the problem address the following aspects: (1) each cell should be allocated to a single location area. Hence, \( V_l \cap V_l' = \emptyset \) for all \( (l, l') \in L^2 \); (2) all the cells should be assigned to a location area. Hence, \( \bigcup_{1 \leq l \leq l_{\text{max}}} V_l = V \); and (3) the location update and paging signalling load in each cell, \( s(L, i) (i \in V) \), should not exceed the maximum allowable threshold. Hence, \( s(L, i) \leq s_{\text{max}}(i) (i \in V) \).

Based on the discussion above we may provide the following formal statement of the location area planning problem.

Problem 1 [Location Area Planning Problem (LAP)]:

Given a graph \( G(V, E) \) representing the cell layout, a vector \( H = \{ h(i,j) \mid (i,j) \in E \} \) comprising the crossing rates between pairs of neighbouring cells, a vector \( \Lambda = \{ \lambda_m(i) \mid i \in V \} \) providing the rate of incoming calls in the cells of the system, the costs of the location update and paging signalling message \( c_l \) and \( c_p \), respectively, and the maximum location update and paging signalling load that may be tolerated in each cell \( i \in V \), \( s_{\text{max}}(i) \), find the allocation of cells to location areas \( L = \{ V_l \mid 1 \leq l \leq l_{\text{max}} \} \), that minimises a cost function \( f(L) \), which is associated with the location update and paging signalling load that is imposed by the arrangement, subject to the conditions of assigning each cell to a single location area, i.e. \( V_l \cap V_{l'} = \emptyset \), \( (l, l') \in L^2 \), and \( \bigcup_{1 \leq l \leq l_{\text{max}}} V_l = V \), and of preserving the location update and paging signalling load in each cell \( i \in V \), \( s(L, i) \), below the maximum tolerable threshold \( s_{\text{max}}(i) \).

4. Optimal formulation

In order to describe the allocation of cells to location areas we introduce the variables \( x_{il} \), \( i \in V \), \( 1 \leq l \leq l_{\text{max}} \) that take the value 1 or 0 depending on whether cell \( i \) is or is not assigned to location area \( l \). Furthermore, we define the variables \( z_{ij} \), where \( (i,j) \in V^2 \), that take the value 1 or 0 depending on whether the cells \( i \) and \( j \) have or have not been assigned to the same location area. The variables \( z_{ij} \) are related to variables \( x_{il}, x_{jl} \), through the relation \( z_{ij} = \sum_{l=1}^{l_{\text{max}}} x_{il} x_{jl} \). This relation may be transformed to a set of linear constraints by applying a technique described in Refs. [48]. The problem of obtaining \( L \) may be reduced to the following optimisation problem.

Problem 1 [Location Area Planning Problem (LAP)]:

Minimise:

\[
f(L) = c_l \sum_{i \in V} \sum_{j \in V} (1 - z_{ij}) h(j,i) + c_p \sum_{i \in V} \sum_{j \in V} z_{ij} \lambda_m(j)
\]

subject to

\[
\sum_{l=1}^{l_{\text{max}}} x_{il} = 1 \quad \forall i \in V
\]

\[
s(L, i) = \sum_{j \in V} (1 - z_{ij}) h(j,i) + \sum_{j \in V} z_{ij} \lambda_m(j) \quad \forall i \in V
\]

\[
s(L, i) \leq s_{\text{max}}(i) \quad \forall i \in V
\]

\[
z_{ij} = \sum_{l=1}^{l_{\text{max}}} x_{il} x_{jl} \quad \forall (i, j) \in V^2
\]

\[
x_{il} \in \{0, 1\} \quad \forall i \in V, 1 \leq l \leq l_{\text{max}}
\]

\[
L = \{ V_l \mid l \in V, 1 \leq l \leq l_{\text{max}} \}
\]

The problem above may be reduced to a 0–1 linear programming problem provided that the non-linear constraints (5) are replaced by a set of linear constraints as shown in Ref. [48]. Relation (1) provides the cost of the location update and paging messages that is imposed by the structure of the location area layout. The set of constraints (2) guarantees that each cell of the system will be assigned to one location area. Constraints (4) guarantee that the location update and paging signalling load in each cell \( i \in V \) will not exceed a given threshold \( s_{\text{max}}(i) \).

The high complexity associated with the computation of the optimal solution of problem 1 necessitates the design of computationally efficient near-optimal algorithms. Such algorithms are presented in the following three sections.

5. Algorithm based on simulated annealing

In this section, we describe the computationally efficient solution that is based on the simulated annealing paradigm.

5.1. Simulated annealing fundamentals

Annealing is the physical process in which a crystal is cooled from the liquid to the solid state. Careful cooling brings the crystal to the solid state. In analogy, a simulated annealing algorithm considers each solution of the optimisation problem as a state, the cost of each solution as the
energy of the state, and the optimal solution as the minimum energy state.

During each phase of the algorithm a new solution is generated by minimally altering the currently best solution. In other words, the new solution is chosen among those that are “neighbouring” to the currently best one. If the cost value that corresponds to the new solution is smaller (i.e. the difference between the cost of the old and the new solution, $\Delta c$, is positive) the new solution becomes the currently best solution. Solutions that increase the cost may be also accepted with probability $e^{\Delta c / CT}$. This mechanism enables the escape from local optima. $CT$ is a control parameter, which may be perceived as the physical analogous of the temperature in the physical process. The algorithm ends when either $CT = 0$ or when a significant number of moves have been performed without improving the cost function.

A formal description of the algorithm is provided in Section 5.3, after presenting in the following section some decisions that have been made in the simulated annealing-based algorithm that has been developed for the LAP.

5.2. Features of the simulated annealing-based algorithm for the LAP

As indicated also in Ref. [49], the development of a simulated annealing based procedure means that the following aspects have to be determined: solution (configuration) space, cost function, neighbourhood structure and cooling schedule (i.e. manner in which the temperature will be decreased).

The following apply with respect to our problem. The configuration space is the set of feasible solutions $\{x_i | i \in V, 1 \leq l \leq l_{max}\}$, i.e. allocations of cells to location areas that satisfy constraints (2)–(7). The cost function is the one introduced by relation (1).

The neighbourhood structure of a solution is produced by reallocating a cell $i$ from its current location area $l_1$ to the location area $l_2$ ($1 \leq l_1, l_2 \leq l_{max}$, $l_1 \neq l_2$) of a neighbour node $j$ ($(i, j) \in E$). The cell $i$ that will be assigned to a different location area and the neighbour- ing cell $j$, to whose location area cell $i$ will be assigned, are randomly chosen.

There are various options regarding the cooling schedule. A recent advanced survey on this aspect may be found in Ref. [50]. The first (simpler and more well-known) scheme identified therein is the geometric cooling scheme. It follows the relations $T_{new} = rCT$ and $CT = T_{new}$. where $CT$ is the temperature in a certain phase of the algorithm (current temperature), and $r$ is a number that usually ranges from 0.95 to 0.99. This scheme is used in the algorithm version that is presented in this paper. The experimentation with more sophisticated schemes [51,52] is left for a future version of this study.

5.3. Formal algorithm description

The simulated annealing-based algorithm may be described as follows:

Algorithm 1: [Simulated annealing-based algorithm for the LAP]

Step 0. Initialisation. Get an initial solution, $IS$, and an initial temperature value $T$. The currently best solution ($CBS$) is $IS$, i.e. $CBS = IS$, and the current temperature value ($CT$) is $T$, i.e. $CT = T$.

Step 1. If $CT = 0$, or if the stop criterion is satisfied, the procedure ends and a transition to Step 6 is performed.

Step 2. A new solution ($NS$) that is neighbouring to $CBS$ is found.

Step 3. The difference of the costs of the two solutions, $CBS$ and $NS$ is found, i.e. the quantity $\Delta c = C(CBS) - C(NS)$ is computed.

Step 4. If $\Delta c \geq 0$ then the new solution becomes the currently best solution, i.e. $CBS = NS$. Otherwise, if $\Delta c < 0$, then if $e^{\Delta c / CT} > \text{rand}(0, 1)$, the new solution becomes the currently best solution, i.e. $CBS = NS$.

Step 5. The cooling schedule is applied, in order to calculate the new current temperature value $CT$, and a transition to Step 1 is performed.

Step 6. End.

Various alternatives may be applied for realising the stop criterion in Step 1. The algorithm stops when no improvement has been made after a given number of temperature decreases (in other words consecutive moves or alterations of the currently best solution), or when $CT = 0$.

6. Algorithm based on taboo search

In this section, we describe the computationally efficient solution that is based on the taboo search paradigm.

6.1. Taboo search fundamentals

The basic idea of the taboo search method is to explore the search space of all feasible solutions through a sequence of moves. At each iteration of the algorithm, a solution is selected as the best one. A set of moves is applied on this solution, and hence, a number of neighbouring solutions are obtained. A subset of these solutions (moves) is classified as forbidden, i.e. they will be placed in the taboo set. Moreover, a subset of the taboo moves may be overridden, according to the aspiration criteria. The new best solutions will be selected among those that are neighbouring and either not taboo, or taboo and aspirant. The specific solution selected is the one that is best in the improvement of the objective function. Solutions are placed in the taboo set to escape from local optima and prevent cycling. The aspiration criteria are applied to maintain some promising solutions.

Lets assume that $S$ denotes the set of solutions to the
optimisation problem. In our case, \( S \) will contain the set of possible allocations of cells to location areas. The algorithm is associated with a set of \( k \) moves, which may be denoted as \( M_s = \{m_1, \ldots, m_{|M_s|}\} \). The application of these moves to a solution \( s \in S \) leads to solutions \( M_s(s) = \{m_1(s), \ldots, m_{|M_s|}(s)\} \). The subset of feasible solutions \( N_s(s) \subseteq M_s(s) \) is the neighbourhood of \( s \). The algorithm starts with an initial solution \( s_0 \in S \) and determines a sequence of solutions, \( s_0, s_1, \ldots, s_k \in S \). Let us assume that at the \( k \)th iteration solution \( s_k \) is considered to be the best one. A new better solution \( s_{k+1} \) will be selected from the set \( N_s(s_k) \). The selection process starts by determining the taboo set \( T_s(i) \subseteq N_s(s_k) \) and the (so-called) aspirant set \( A_s(i) \subseteq T_s(i) \). The new best solution \( s_{k+1} \) is the one from the set \( (N_{s_k}(s_k) - T_s(i)) \cup A_s(i) \) that improves more the objective function.

Moreover, a taboo search procedure may use the following operators: (1) intensification, which favours the search in certain areas of the solution space, where some promising characteristics are found; and (2) diversification, which tends to spread the exploration to different areas of the solution space. Diversification is a method for escaping from local optima.

A formal description of the algorithm is provided in Section 6.3, after presenting in the following section some decisions that have been made in the taboo search-based algorithm that has been developed for the LAP.

6.2. Features of the taboo search based algorithm for the LAP

The development of a taboo search based procedure means that the following aspects have to be determined: solution (configuration) space, cost function, set of moves, and taboo and aspiration conditions.

The solution (configuration) space is the set of feasible allocations \( \{x_{ij}\} \in V, 1 \leq i \leq l_{\text{max}} \) of cells to location areas that satisfy constraints (2)–(7). The cost function is the one introduced by relation (1).

A “move” with respect to our problem is to take a cell \( i \) that lies on the boundaries of a location area \( l_1 \) and to assign it to an adjacent location area \( l_2 \) \( (1 \leq l_1, l_2 \leq l_{\text{max}}, l_1 \neq l_2) \). In other words, a move is to assign cell \( i \) that lies on the border of its current location area \( l_1 \), to the location area \( l_2 \) of a neighbouring cell \( j \) such that \( (i,j) \in E \). Obviously, the overall set of moves contains all the possible reallocations of cells that lie on the borders of their location areas to adjacent location areas.

When a cell \( i \) is moved from one location area \( l_1 \) to another \( l_2 \), the move is introduced into the taboo list. This is done to prevent cycling. The move stays in the taboo set for a given number of algorithm iterations.

The taboo status of a move is overruled if the improvement in the objective function is more than a given level. This move is then placed in the aspiration set.

6.3. Formal algorithm description

The simulated annealing-based algorithm may be described as follows:

Algorithm 2: [Taboo search-based algorithm for the LAP]

Step 0. The counter of the algorithm iterations, \( k \), is initialised, i.e. \( k = 1 \). An initial solution \( s_k \) is obtained. The set of taboo and aspirant moves is initialised, i.e. \( T_s(k) = \emptyset \), and \( A_s(k) = \emptyset \).

Step 1. If the stop criterion is satisfied, the procedure ends and a transition to Step 9 is performed.

Step 2. The set of moves \( M_s \) is applied to solution \( s_k \), and hence, a new set of solutions \( M_s(s_k) \) is produced. Each move refers to the reallocation to an adjacent location area of a cell that, according to solution \( s_k \), lies on the borders of its current location area.

Step 3. The feasibility of the solutions of the \( M_s(s_k) \) set is examined. The subset of these solutions that are feasible, i.e. satisfy the relations (2)–(7), forms the set of solutions that are neighbourable to \( s_k \), \( N_{s_k}(s_k) \subseteq M_s(s_k) \).

Step 4. The set of solutions, \( C(s_k) \), that are candidate for obtaining the best solution status, and therefore, for replacing solution \( s_k \) in the next algorithm iteration, is formed through the relation \( C(s_k) = N_{s_k}(s_k) \cup T_s(k) \). Moreover, solution \( s_k \) is appended in the \( C(s_k) \) set.

Step 5. The \( A_s(k) \) set is formed. More specifically, the cost function value that is scored by the solutions in the \( T_s(k) \) set is computed. Those solutions that improve the objective function (1) by more than a given level are removed from the \( T_s(k) \) set and placed in the \( A_s(k) \) set.

Step 6. The set of solutions \( C(s_k) \) is enhanced through the relation \( C(s_{k+1}) = C(s_k) \cup A_s(k) \).

Step 7. The solution of the \( C(s_{k+1}) \) set that is the best in improving the objective function (1) becomes the best solution that will be used in the next algorithm iteration, \( s_{k+1} \).

Step 8. The next algorithm iteration is prepared. Therefore, the set of taboo moves that will be used in the next algorithm iteration is updated through the relation \( T_s(k + 1) = T_s(k) \cup M_s(s_k) \). Moreover, solutions that have stayed in the taboo set for more than a given number of algorithm iterations are removed from the \( T_s(k + 1) \) set.

Finally, \( A_s(k + 1) = \emptyset \) and \( C(s_{k+1}) = \emptyset \). The algorithm iteration counter is increased, i.e. \( k = k + 1 \), and a transition to Step 1 is performed.

Step 9. End.

Various alternatives may be applied for realising the stop criterion in Step 1. The algorithm stops when the maximum number of algorithm iterations is conducted, or no improvement has been made after a given number of algorithm iterations.

The avoidance of local optima is accomplished through
the application of the diversification operator in its simpler version, which is to perform several random restarts.

7. Genetic algorithm

In this section, we describe the computationally efficient solution that is based on the genetic algorithm paradigm.

7.1. Genetic algorithm fundamentals

In general, genetic algorithms maintain a set of problem solutions, which may be seen as the equivalent of a population of individuals. A string, also called a chromosome (Fig. 3(a)), is used for representing a solution. During each algorithm iteration, or generation in more strict genetic algorithm terms, the solutions are rated with respect to their quality, or fitness. Consequently, some solutions will be selected and used for the generation of a new population. This generation relies on the so-called genetic algorithm operators. In general, genetic algorithms use the selection (reproduction), crossover, mutation and replacement operators.

The selection operator aims at selecting the solutions that will reproduce. The usual choice is to select the solutions that exhibit the best performance with respect to the fitness function (in analogy with real life, where individuals with higher fitness have a bigger probability to reproduce). A technique for guaranteeing the convergence of a genetic algorithm is to retain in the new population the best solutions of the previous population.

The crossover operator is applied with probability \( p_c \) after the selection (reproduction) operator. The basic idea of this phase is to select two “parent” solutions from the current set of solutions and to combine them to create two children as illustrated in Fig. 3(b). The mutation operator is applied after the crossover operator with probability \( p_m \). The mutation operator produces a new solution by modifying one or more gene values of an existing solution as illustrated in Fig. 3(c).

In general, the crossover and mutation operators generate solutions from the already available population of solutions. The replacement operator is applied to the population of the already available and the generated solutions in order to create the new population of available solutions.

A formal description of the algorithm is provided in Section 7.3, after presenting in the following section some decisions that have been made in the genetic algorithm that has been developed for the LAP.

7.2. Features of the genetic algorithm for the LAP

The construction of a genetic algorithm requires that the following points be addressed: (1) the aspects that are represented by the genetic chromosome should be chosen; (2) the set of genetic operators should be chosen; (3) the fitness function should be defined; and (4) the probabilities for controlling the genetic operators should be determined.

The chromosome in our case consists of a sequence of \( uVl_{\text{max}} \) genes (Fig. 4(a)). The first \( l_{\text{max}} \) genes indicate the location area in which the first cell belongs. Hence, only one of the \( l_{\text{max}} \) genes is set equal to 1, while the rest \( l_{\text{max}}-1 \) are set to 0. In the same manner, the genes with index from \( l_{\text{max}}+1 \) to \( 2l_{\text{max}} \) designate the location area in which the second cell is allocated, etc.

The fitness function of a solution is taken as the inverse of the objective function (1). This is done to straightforwardly express that solutions that yield lower objective function values are seen as more “fit” from the algorithm point of view.

In the sequel, we describe in more detail the decisions regarding the specific genetic algorithm operators.

Selection. It is based on a random process that favours the selection (for reproduction) of solutions that have higher fitness function values.

Crossover. The technique works as illustrated in Fig. 4(b).
In this figure the allocation of cells to location areas (shaded areas in Fig. 4(b)) for two different parents is shown when the network is a $2 \times 2$ square grid. During the crossover operator, first, we randomly choose the crossover point $x \cdot |L|$, $x \in (1, |L|)$, and hence, the cell layout in these solutions is separated in two sectors, namely, sets $W_1$ and $W_2$. After the application of the crossover operator, the cells in set $W_2$ of parent 1 (2) will be allocated to the location areas of the cells in $W_1$ of parent 2 (1). On the produced children, the following aspects should be considered. First, the location area must constitute a connected sub-graph of the overall graph that represents the cell layout. In other words, there must be a path, consisting of cells of the location area that connects each cell of the location area with the rest of the cells of the same area. Second, in case the produced children contain unconnected location areas, two
alternatives could be followed. The first and simplest alternative is to reject this child (solution). The second alternative is the following. Let $W$ be the subset of cells of a location area that is not connected to the rest of the cells. Each cell of $W$ will be reallocated, from the current location area, to the location area of one of its neighbours in the graph $G(V, E)$.

The latter approach is adopted in the example presented.
in Fig. 4(b). More specifically, in this figure, the selected crossover point entails that set $W_1$ comprises cell-1 and cell-2 while set $W_2$ comprises cell-3 and cell-4. In the 1st parent, LA_I consists of cell-1, cell-2 and cell-4 while LA_III comprises cell-3. Likewise, in the 2nd parent, LA_I comprises cell-1, LA_II comprises cell-2 and LA_III comprises cell-3 and cell-4. As we have mentioned above, after the crossover operator is applied, set $W_1$ of child 1 (2) will be the set $W_1$ of parent 1 (2) while set $W_2$ will be the set $W_2$ of parent 2 (1). More specifically, in the 1st child, LA_I consists of cell-1 and cell-2 while LA_III consists of cells 3 and 4. In the 2nd child, LA_I consists of cell-1 and cell-4, LA_II comprises cell-2 and LA_III comprises cell-3. From the obtained results, we may observer that LA_I of child 2 consists of two non-connected sub-graphs. Therefore, a post-processing is needed in order to work out this deficiency. In this respect, cell-4 is reallocated to the location area of one of its neighbours, namely LA_II. This post-processing, results in a new child that consists of connected location areas.

**Replacement.** The aim of the operator is to replace solutions that score small fitness function values.

**Mutation.** The technique works as illustrated in (Fig. 4(c)). This operator is applied for changing the value of a

![Figure 6](image-url)
number of genes \(g_m\) of a given chromosome while assuring that the new chromosome will still be a feasible solution.

In summary, the following important parameters of the algorithm have to be configured: (1) the size of the current population of solutions; (2) those that are associated with the crossover operator, i.e. the size of the generated population of solutions and the \(p_c\) value; and (3) those that are associated with the mutation operator, i.e. the \(g_m\) and \(p_m\) values.

7.3. Formal algorithm description

Based on the discussion above a general description of the genetic algorithm is the following:

Algorithm 3: [Genetic algorithm for the LAP]

Step 0. Initialisation. The initial population of solutions (IPS) is created by means of a random procedure or a heuristic algorithm. The current population of solutions (CPS) is the IPS, i.e. \(CPS = IPS\).

Step 1. If the stop criterion is satisfied the algorithm stops and a transition to Step 6 is performed.

Step 2. The reproduction operator is applied to the solutions of CPS, so as to select the set of the mating solutions, MS that will be allowed to reproduce. The solutions in the MS set are selected based on a random process that favours solutions that exhibit the best fitness function values.

Step 3. The crossover and mutation criteria are applied to the set of the best solutions MS, so as to obtain the generated population of solutions GPS.

Step 4. The generated solutions are temporarily appended to the current population of solutions. This yields the temporary population of solutions \(TPS = CPS \cup GPS\). Subsequently, the replacement operator is applied to the TPS set. This yields the set of solutions, RS, which should be removed from the TPS set. It holds that \(|RS| = |GPS|\).

The new CPS set is \(CPS = TPS \setminus RS\).

Step 5. A transition to Step 1 is performed.

Step 6. End.

8. Results

In this section, we will assess the performance of the proposed location area planning techniques. Our focus will be on the investigation of the performance of the location area planning algorithms in diverse environments. In this respect, two sets of experiments will be used. The experiments are differentiated from the cell size, the mobility rate of the users as well as the user density used. In the first set of experiments, we focus on a microcellular system wherein a large number of users exist at various mobility levels. Next, in the second set of experiments, we extend our study in a macrocellular system wherein a large number of users exist at various mobility levels.

In the first set of experiments, the network used is the one depicted in Fig. 5(a). According to this cell plan, each base station covers half a block in all four directions as proposed in Ref. [53]. The specific microcellular network choice does not simplify the computational effort. Other topologies of the same size and connectivity degree could have been chosen instead. The density of users is expressed in terms of the number of users per square kilometre (users/km\(^2\)) and is taken equal to 10,000, i.e. \(\rho = 10,000\) users/km\(^2\). The incoming calls to a user form a Poisson process with mean \(\lambda_u\), expressed in terms of incoming calls per hour per user. A fluid model [43,44] is used for modelling the mobility behaviour of users. Fluid models suggest that the traffic flow out of an area is similar to the flow of a fluid. Such models have some limitations, which are out of the scope of this paper, but are adequate in capturing the macroscopic mobility behaviour of users, which is enough for our purposes. Adopting one of the simpler forms of the model, we obtain that the average number of boundary crossings out of cell-\(i\) per time unit is \(N_{out}(i) = \rho_v L_v\), i.e. proportional to the population density within the cell, denoted as \(\rho\), the cell perimeter, denoted as \(L_v\), and the average velocity of the users \(v\). Regarding the cost of the location update and paging signalling messages, we assume that \(c_\ell = c_p = 1\), which means that the relative cost of the location update message is equal to the respective cost of the paging message. Nevertheless, this is an assumption that does not significantly affect the comparisons we want to make. The algorithms have been coded in the C++ programming language and the tests were run on a SUN workstation.

The first set of results, depicted in Fig. 6, aims at demonstrating the efficiency of the heuristic algorithms for a given rate of incoming messages when the mobility levels are gradually increased. In this respect, Fig. 6 depicts the rate of location update, paging signalling load and total signalling cost versus the average levels of user mobility when the rate of incoming messages is taken to be equal to 1, i.e. \(\lambda_{in} = 1\). As previously mentioned, the network considered is the one depicted in Fig. 5(a), and the user density assumed is \(\rho = 10,000\) users/km\(^2\).

The results obtained from the application of the application of the genetic, simulated annealing and taboo search algorithm are depicted in Fig. 6(a)–(c), respectively. From the derived results, we may observe that all algorithms achieve very promising results. More specifically, for \(v = 3\) km/h, the achieved overall signalling cost is approximately 17,000 messages/h (around 5 messages/s), 16,500 messages/h (around 4 messages/s) and 15,300 messages/h (less than 4 messages/s) when the genetic, simulated annealing and taboo search algorithm is applied, respectively. Comparing to the worst case scenario, according to which each cell is allocated to a different location area for the specific mobility level, the obtained total signalling cost is reduced by 44, 46 and 49% for the three heuristics, respectively, for \(v = 3\). Furthermore, at higher mobility levels, i.e. \(v = 10\), the algorithms exhibit an even better performance since the reduction raises at 62, 66 and 63% for the
respective heuristic algorithms. Such results are achieved since as mobility levels increase, users move faster between cells, which results in higher location update cost. In order to compensate for that, the planning algorithms deploy fewer location areas, and hence reduce the location update rates.

Regarding the performance of the three algorithms, we may observe that the total signalling costs achieved are comparable, although those of the taboo search algorithm are slightly better with respect to the other two algorithms. Specifically, the taboo search algorithm exhibits an overall improved behaviour, which is up to 10% for $v = 7$. The comparable performance is strong evidence on the quality of the three algorithms. In general, a comparison with the optimal algorithm would be convincing on the quality of the solutions of the heuristic algorithms. However, an optimal LAP algorithm exhibits a very high complexity. This means that the provision of solutions to problem instances, as those of this paper may not be achieved. Hence, the contribution of this paper, i.e. the development of three different computationally efficient algorithms and the realisation of comparative studies, is a solid basis (or the only way) for being convinced on the quality of the solutions.

Fig. 7. Location update and paging signalling load for micro cells when $\lambda_m = 2$ versus the average velocity of the users in case: (a) genetic algorithm is applied; (b) the Simulated Annealing algorithm is applied; and (c) the Taboo Search algorithm is applied.
Comparing the obtained location update and paging costs, we may observe that the respective figures are not always balanced. This is because we have used a general form for the objective function in which the relative cost of the location update message is equal to the respective cost of the paging message. It is noted that by choosing the algorithm or by slightly reconfiguring the cost function we could place emphasis on the reduction of one of the two cost factors.

Fig. 7 presents the location update, paging cost and total signalling cost versus the average levels of user mobility when the rate of incoming messages is taken to be equal to 2, i.e. $\lambda_m = 2$ and the user density is maintained at $\rho = 10,000$ users/km$^2$. Similarly, to the previous set of results, the three algorithms exhibit similar performance. More specifically, for $v = 3$ km/h, the achieved overall signalling cost is approximately 6 messages/s, less than 6 and 5 messages/s when the genetic, simulated annealing and taboo search algorithm is applied, respectively. Comparing to the worst case scenario, according to which each cell is allocated to a different location area, the obtained total signalling cost is reduced by 29, 32 and 34% for the three heuristics, respectively, for $v = 3$. Furthermore, at higher mobility levels, i.e. $v = 10$, the algorithms exhibit an even better performance since the reduction raises at 52, 58 and...
57% for the respective heuristic algorithms. Comparing now to the previous set of results (for $\lambda_{in} = 1$), the reduction, although still significant, is ranging in lower levels due to the increased rate of incoming messages.

Similarly to the previous test, the three heuristics show a comparable performance, though the taboo search algorithm yields, again, slightly better results over the other two algorithms. Specifically, the taboo search algorithm exhibits an improved behaviour, which is up to 2% for $v = 7$. Again, this improvement may be eliminated through a better tuning of the algorithms’ input parameters. All the other observation that were made in the previous set of results, i.e. on the balancing of the two cost factors, are still valid in this test.

Fig. 8 presents the location update, paging cost and total signalling cost versus the average levels of user mobility when the rate of incoming messages is taken to be equal to 3, i.e. $\lambda_{in} = 3$ and the user density is maintained at $\rho = 10,000$ users/km$^2$. More specifically, for $v = 3$ km/h, the achieved overall signalling cost is approximately 7 messages/s, less than 7 messages/s and around 6.5 messages/s when the genetic, simulated annealing and taboo search algorithm is applied, respectively. Comparing to the worst case scenario, the obtained total signalling cost
is reduced by 20, 22 and 25% for the three heuristics, respectively, for $v = 3$. Moreover, for $v = 10$, the algorithms achieve a reduction of 45, 48 and 52%, respectively. We may observe, again that the three heuristics show comparable performance. The taboo search algorithm continues to exhibit an improved behaviour, which is up to 7% for $v = 10$.

In the second set of experiments, the network used is depicted in Fig. 5(b). We have considered a macrocellular system wherein cells are assumed hexagonal with a radius $R = 2$ km. The density of users is taken equal to 1000, i.e. $\rho = 1000$ users/km$^2$. Similarly to the previous set of experiments, the incoming calls to a user form a Poisson process with mean $\lambda_{in}$, while the fluid model is used for modelling the mobility behaviour of users.

Following the same line of study as in the first set of experiments, Fig. 9 aims at demonstrating the efficiency of the heuristic algorithms in a macrocellular system for a given rate of incoming messages and gradually increased mobility levels. In this respect, the location update, paging signalling load and total signalling cost is depicted versus the average levels of user mobility when the rate of
incoming messages is taken to be equal to 1, i.e. $\lambda_\text{ini} = 1$. More specifically, Fig. 9(a)–(c), depict the results obtained from the application of the genetic, simulated annealing and taboo search algorithm, respectively. From the derived results, we may observe that the algorithms achieve very good results even in this macrocellular system. Particularly, for $v = 10 \text{ km/h}$, the obtained total signalling cost is approximately 8.5, 8.3 and 8 messages/user/h originated from 12,560 users/cell when the genetic, simulated annealing and taboo search algorithms is applied, respectively. Comparing to the worst case scenario, the obtained total signalling cost is reduced by 22, 24 and 26% through the application of the genetic, simulated annealing and taboo search algorithm, respectively, for $v = 10$. As mobility levels increase, the algorithms exhibit, again, an even better performance. Specifically, at high mobility levels the total signalling cost is reduced by 49, 54, and 55% for the respective heuristic algorithms. Similarly, to the first set of experiments, such improvement is owned to the deployment of less location areas, which yields, primary, reduced location update rates.

The results derived from the application of taboo search...
algorithm are again better by 3–8%, depending on the mobility level. Another issue that is also valid in this set of results is that the cost factors may become more balanced by slightly reconfiguring the cost function.

Fig. 10 presents the location update, paging cost and total signalling cost versus the average levels of user mobility when the rate of incoming messages is taken to be equal to 2, i.e. $\lambda_{in} = 2$ and the user density is maintained at $\rho = 1000 \text{ users/km}^2$. The achieved overall signalling cost is a little more than 11 messages/user/h, around 11 messages/user/h and less than 11 messages/user/h originated from 12,560 users/cell for $v = 10 \text{ km/h}$, when the genetic, simulated annealing and taboo search algorithm is applied, respectively. Comparing to the worst case scenario, the respective reduction is approximately 5, 8 and 9% for the three heuristics, respectively, for $v = 10$. At high mobility levels the reduction raises at 37, 38 and 43% for the respective algorithms. It is noted here that, as the problem scale increases the reduction decreases due to the inherent complexity of the problem. Similarly to the previous tests, the taboo search algorithm shows again an improved performance, which ranges from 0.5 to 8% as the mobility levels increase.

Fig. 11 illustrates the location update, paging cost and total signalling cost versus the average levels of user mobility for $\lambda_{in} = 3$ and $\rho = 1000 \text{ users/km}^2$. In this respect, for $v = 10 \text{ km/h}$, the obtained total signalling cost is approximately 13, 12 and less than 12 messages/user/h originated from 12,560 users/cell when the genetic, simulated annealing and taboo search algorithms is applied, respectively. The entailed improvement, comparing to the worst case scenario, is 2, 2.1 and 2.3% for the three heuristics, respectively, for $v = 10$. Similarly to the previous tests, this reduction increases further for higher mobility levels. The taboo search algorithm shows an overall improved performance that may reach up to 5% at high mobility levels.

9. Conclusions

This paper addressed an important planning problem for future cellular systems. More specifically, the importance of the location area-planning problem was explained. In the sequel, the problem was generally described, formally stated, and optimally formulated. The computational complexity associated with the computation of the optimal solution motivated the presentation of three computationally efficient algorithms that adhere to the simulated annealing, taboo search, and genetic algorithm paradigms. These algorithms were applied to test cases and corresponding results were presented.

The algorithms presented herewith are suitable for application in the design, or the re-engineering, phases of the system. An interesting issue for future study is to direct our work towards schemes that may dynamically handle the changing user behaviour patterns that occur in the network. This extension of our work will lead to algorithms that are suitable for use in the management domain.

References


RACE II project R2066, Mobile Networks (MoNet), CEC deliverable No. R2066/SESA/GA2/DS/P/054/b/1, Signalling traffic requirements for UMTS, December 1993.


RACE II project R2066, Mobile Networks (MoNet), CEC deliverable No. R2066/SESA/GA2/DS/P/054/b/1, Signalling traffic requirements for UMTS, December 1993.