Analysis of Heuristic Graph Partitioning Methods for the Assignment of Packet Control Units in GERAN

Matías Toril, Iñigo Molina-Fernández ({mtoril,imf}@ic.uma.es)
Communications Engineering Dept., University of Málaga, Málaga, Spain

Volker Wille (volker.wille@nsn.com)
Performance Services, Nokia Siemens Networks, Huntingdon, UK

Chris Walshaw (c.walshaw@gre.ac.uk)
School of Computing and Mathematical Sciences, University of Greenwich, London, UK

February 22, 2010

Abstract. Over the last few years, graph partitioning has been recognized as a suitable technique for optimizing cellular network structure. For example, in a recent paper, the authors proposed a classical graph partitioning algorithm to optimize the assignment of cells to Packet Control Units (PCUs) in GSM-EDGE Radio Access Network (GERAN). Based on this approach, the quality of packet data services in a live environment was increased by reducing the number of cell re-selections between different PCUs. To learn more about the potential of graph partitioning in cellular networks, in this paper, a more sophisticated, yet computationally efficient, partitioning algorithm is proposed for the same problem. The new method combines multi-level refinement and adaptive multi-start techniques with algorithms to ensure the connectivity between cells under the same PCU. Performance assessment is based on an extensive set of graphs constructed with data taken from a live network. During the tests, the new method is compared with classical graph partitioning approaches. Results show that the proposed method outperforms classical approaches in terms of solution quality at the expense of a slight increase in computing time, while providing solutions that are easier to check by the network operator.

Keywords: mobile, network, optimization, graph partitioning, packet control unit

1. Introduction

In recent years, the size and complexity of cellular networks have increased dramatically. In the past, operators have dealt with this problem by increasing their workforce and overdimensioning network resources. However, with the increased level of competition, and the resulting cost pressures, such an approach is not feasible anymore. Hence, in currently deployed radio access technologies, such as GERAN or UTRAN, an efficient network management is crucial to provide high-quality services at a low operational cost. This fact has stimulated intense research activities in the field of automatic network optimization, [1][2][3].

One of the most challenging tasks in cellular network optimization is re-structuring the network based on performance statistics. For this purpose, graph partitioning has been recognized as a powerful tool. Nonetheless, compared to other industries, the application of graph partitioning to cellular network design is still in its infancy. To better understand the potential of these techniques in mobile telecommunications, this paper presents a thorough analysis of the use of graph partitioning algorithms in an example of clustering problem, namely the assignment of cells to Packet Control Units in GERAN.

With the launch of the first packet-data based services, network optimization related to these services has come into focus. The aim of these optimization procedures is to maximize data throughput with existing resources. In GERAN, these services are based on GPRS (General Packet Radio Service) or EDGE (Enhanced Data Rates for Global

Evolution), [4]. Although theoretical peak data rates per Time SLot (TSL) of 20kbps for GPRS and 48kbps for EDGE might be considered acceptable, the actual data rate is often below these theoretical values for several reasons, [4]. In particular, every change of serving cell made by the user (known as cell re-selection) causes interruption of the associated data flow, whose duration depends on mobility management procedures and features in use in the network. In this framework, the assignment of cells to *Packet Control Units* (PCU) within a Base Station Controller (BSC), defined by the operator when designing the network, is key to maximizing user data throughput. Each BSC contains a certain number of PCUs, responsible for the control of packet data traffic, to which the base stations (i.e., cells) of the BSC have to be associated. Since inter-PCU re-selections cause far longer service breaks than intra-PCU re-selections, the PCU plan should minimize the number of PCU re-allocations experienced by users when a change of serving cell takes place. At the same time, the PCU plan should also ensure that the load of all PCUs remains within certain limits.

As already recognized in [5], the Cell-to-PCU Assignment Problem (CPAP) can be formulated as a graph partitioning problem, for which several heuristics have been proposed (for a survey, see [6]). These approaches can be classified as either geometric, [7], or combinatorial. The combinatorial approaches can be categorized still further as local search-based, [8], spectral, [9], or multi-level, [10]. In recent years, the *multi-level Kernighan-Lin* heuristic, [10][11][12], has become the common benchmark against which more refined graph partitioning heuristics are compared. In the context of cellular networks, these heuristics have been applied for the assignment of cells to switches, [13][14][15][16], and location areas, [16][17][18][19], during the planning stage. These methods produce a fixed or arbitrary number of clusters of bounded size that minimize inter-cluster handovers. However, most of these approaches have been conceived for the initial design of the network, and consequently pay little attention to re-planning and maintenance procedures. Thus, performance aspects such as the number of changes in the network configuration and the ease of maintenance of the new solution have traditionally been neglected.

This paper extends and improves the work presented in [5]. A novel heuristic method is proposed for assigning cells to PCUs during network design. As in [5], the method is based on statistics available in the network management system and has been conceived as a network re-planning procedure. The main benefit of re-configuring the network is a reduction of the number of users of packet-data based services that change PCU when changing cell, which should decrease the overall cell re-selection delay in the network. Unlike [5], the new method combines two classical graph partitioning techniques that have been considered separately in the past: the multi-level refinement, [10], and the clustered adaptive multi-start, [20]. Likewise, the new method includes algorithms to ensure the connectivity of cells assigned to the same PCU, which is often neglected in classical graph partitioning algorithms. Thus, the geographical consistency of solutions is improved, which makes visual inspection of solutions easier for operators. Although the need for connectivity was already identified in [5], no explicit actions were taken there to satisfy this condition. Performance assessment is carried out by comparing the proposed method with several classical approaches over an extensive set of graphs constructed from a live cellular network. To the authors' knowledge, no previous work has compared classical graph partitioning algorithms over real graphs from cellular network design. It is well known that the performance of graph algorithms strongly depends on graph attributes. It will be shown here that cellular graphs have important differences compared to those in supercomputing applications, for which most existing partitioning algorithms were

designed, justifying the need for a separate analysis. Hence, the main contributions of this work are: a) a novel graph partitioning algorithm, which, unlike classical ones, takes into account the peculiarities of cellular networks and can be applied to many clustering problems arising when structuring cellular networks, and b) a performance analysis of classical and novel graph partitioning algorithms over graphs from a live cellular network. It should be pointed out that the problem considered here (i.e., the assignment of PCUs) should only be considered as an instance of clustering problem. Thus, it is believed that the results described here can also be extended to similar problems arising in other layers of network hierarchy in both GERAN and UTRAN, since graphs are built from similar statistics.

The paper is organized as follows. Section 2 introduces the formulation of the PCU planning problem. Section 3 describes the proposed graph partitioning algorithm. Section 4 presents the performance analysis over graphs constructed from data of a live network and Section 5 presents the main conclusions of the study.

2. The PCU Planning Problem

This section outlines the CPAP from the network operator's perspective. First, the cell re-selection process in GERAN is presented. The CPAP is then formulated as a graph partitioning problem, highlighting the differences with other classical formulations of the latter problem. Finally, the current method used by operators to solve the CPAP is discussed in order to understand the limitations of this approach.

2.1. Cell Re-selection Process in GERAN

In any mobile communication system, the *HandOver* (HO) process normally ensures that a Mobile Station (MS) is served by the cell that offers the best coverage while connected to the network. Alternatively, when the MS is idle, the *Cell (Re)selection* (CR) process, [21], controls in which cell the MS will initiate future connections. However, when an MS is transferring packet data, it is neither strictly in idle mode, nor has it a permanent connection to the serving cell where the HO mechanism would work. In this situation, the CR is responsible to select to/from which cell user data is sent.

The CR process has a strong influence on the performance of packet-data services in GERAN. In existing GERAN systems, one of the main challenges in providing adequate quality of service is the delay and packet loss incurred during a cell change. Trial results over live networks have shown that interruption associated to cell change ranges from 2 to 12 seconds, [5]. For future GERAN releases, several features have been proposed to reduce this delay, the foremost of which are Packet Broadcast Channel, [4], Network-Assisted Cell Change, [4], and Packet-data Handover, [22]. In the interim, current GERAN networks must be optimized for the existing capabilities.

In GERAN, the delay associated to a cell change can be minimized by proper planning of the PCUs. As depicted in Figure 1, a PCU is a physical unit within the BSC, which is responsible for the control of the packet data. Its main functions are packet-data radio resource management, connection establishment and management, data transfer and uplink power control. Each BSC contains a certain number of PCUs to which the cells of the BSC have to be associated. The number of PCUs is defined during the network dimensioning stage on a per-BSC basis. For that purpose, software and hardware constraints of the PCU element are taken into account. The most common PCU constraints are the maximum

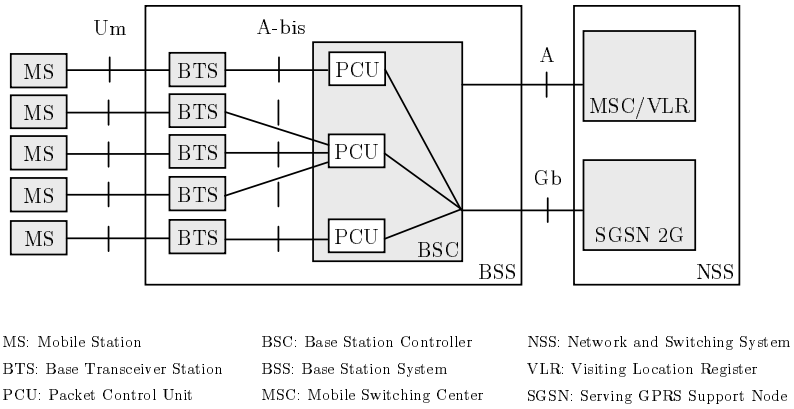


Figure 1. The packet control unit within GERAN A/Gb mode reference architecture.

packet-data traffic handled as well as the maximum number of cells, transceivers and TSLs devoted to data traffic in cells served by the same unit. Once the network is dimensioned, the operator must configure the association between cells and PCUs, which is known as the *PCU plan*.

The main goal of the PCU plan is to connect cells that are close in propagation terms to the same PCU. This objective aims to reduce the number of PCU re-selections suffered by users when a change of serving cell takes place, as CR between cells of the same PCU (intra-PCU CR) causes far shorter service breaks than CR between cells on different PCUs (inter-PCU CR). Therefore, in order to maximize data rates for mobile users of packet-data based services, it is crucial to minimize the number of inter-PCU CRs. In other words, cells between which many CRs occur should be on the same PCU, whilst cells with a weak relationship can be placed on different PCUs.

Ideally, the optimization process should be based on CR statistics of users in Packet-Switched (PS) mode. Unfortunately, such information is not currently available in the network management system, as the CR process is controlled by the MS. In the absence of such PS statistics, it is deemed most suitable to utilize HO statistics related to Circuit-Switched (CS) services. Assuming that the mobility of terminals in PS mode is similar to that of terminals in CS mode, the errors in such analysis should be relatively small. Although such an assumption is debatable, field trial results reported in [5] prove the validity of this approach. Hereafter, the main objective is re-formulated as the minimization of the number of HOs between cells in different PCUs.

As a secondary objective, the optimization process should reject those solutions that lead to large traffic imbalance between PCUs. By avoiding that some PCUs are overloaded, while others are underutilized, the overall capacity of the existing set of PCUs is fully exploited. Nonetheless, putting too much emphasis on perfect balance is not recommended, since this strategy would possibly lead to solutions with worse CR performance. Consequently, the balance criterion should not be included in the objective function to be minimized, but only considered as a constraint. This imbalance constraint is dealt with by limiting the maximum ratio among data traffic loads of the PCUs in a BSC.

Several other performance criteria must be taken into account when these methods are applied during the operational stage as part of daily optimization routine. The time to compute a new solution is a key performance aspect. Every time a new cell or PCU is added to the network, a new solution for the CPAP must be created for the BSC to which it is associated. Since this event is rather frequent, the execution time is one of the

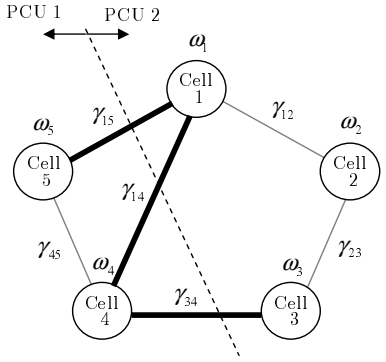


Figure 2. The Cell-to-PCU assignment modeled as a graph partitioning problem.

most relevant criteria to assess the value of an algorithm. In addition, the application of the method in a live environment should minimize the number of changes from the old solution. Since the re-configuration of a new PCU plan might require disabling packet-data transmission on cells involved, the download time must be kept to a minimum. As new cell-to-PCU assignments can only be downloaded sequentially, the smaller the number of cell re-allocations, the higher the cell availability.

2.2. Formulation of the Problem

The CPAP can be formulated as a combinatorial optimization problem known as the *graph partitioning problem*, [6]. In this approach, the network area under optimization is modeled by a simple non-directed weighted graph, as shown in Figure 2. The vertices of the graph represent the cells in the BSC, while the undirected edges between them represent the adjacencies defined by the operator for HO purposes. Since edges are non-directed, it is assumed that adjacencies are bi-directional in nature (i.e., the adjacency between two cells is unique, regardless of the direction of user movement). The weight of each vertex, ω_i, denotes the number of TSLs devoted to packet-data traffic in the cell, while the weight of each edge, γ_{ij}, denotes the number of HOs between cells on its ends. The partitioning of the graph, performed by grouping vertices into a fixed number k of disjoint subsets, referred to as subdomains, reflects the assignment of cells to PCUs. The resulting partition defines a set of edges that join vertices in different subdomains (highlighted in bold), δ, which is referred to as a cut.

Over this graph, the CPAP is modeled as a *bounded, min k-cut problem*, [23], described as follows. Let G be an undirected weighted graph G=(V,E), consisting of a set of vertices V and edges E, vertex weights ω_i and edge weights γ_{ij}. Let B_{aw} and B_{rw} be decimal values defined as absolute and relative weight bounds. B_{aw} represents the capacity limit of the PCU in terms of the maximum number of TSLs, while B_{rw} is the maximum load imbalance ratio between the most heavily loaded and the most lightly loaded PCU in a BSC. The problem consists of partitioning V into k subdomains, S_1, S_2, $\cdots$, S_k, such that

$$\|S_n\| = \sum_{i \in S_n} \omega_i \leq B_{aw} \qquad \forall\, n \in \{1, 2, ..., k\} \tag{1}$$

(i.e., the weight of each subdomain is bounded),

$$\frac{\max(\|S_1\|, ..., \|S_k\|)}{\min(\|S_1\|, ..., \|S_k\|)} \leq B_{rw} \tag{2}$$

(i.e., the weight imbalance ratio is bounded) and

$$\sum_{(i,j)\in\delta(S_1,\ldots,S_k)} \gamma_{ij} \tag{3}$$

(i.e., the sum of the weights of the edges in the cut, referred to as *edge-cut*) is minimized.

For several reasons, operators prefer solutions in which cells in the same PCU are geographically close to each other. Although this property is not strictly necessary for network operation, this sort of solution is easier to check by maintenance personnel. As no vertex coordinates are included in the graph modeling the CPAP, distance must be inferred from connectivity information in the adjacency matrix. Thus, the only way to check that two clusters of vertices in the same subdomain are geographical neighbors is by checking that there exists an edge (or path) that links them. If such a path does not exist, clusters can be arbitrarily far from each other. It can thus be concluded that, in order to keep geographical consistency, it is desirable that each subdomain in the solution is connected, i.e., there exists an internal path between every pair of vertices in the subdomain. This additional constraint is hereafter referred to as the *connectedness property*. It should be pointed out that it is sometimes impossible to fulfill the latter constraint if the original graph consists of two or more isolated clusters of vertices and hence is not itself connected. In a live network, this situation is likely to happen in rural areas, where discontinuous cell coverage is common. Under these circumstances, the best solution is the one that has the minimum number of disconnected subdomains.

2.3. Current Solution Technique

The selection of the PCU to which a cell is assigned is currently a manual process, which is performed during the planning stage. During the operational stage, subsequent increments of data traffic require the extension of the number of PCUs in some BSCs. The required analysis task turns the PCU selection problem into a time-consuming process. Due to effort and expenses, sub-optimal solutions are adopted for the sake of simplicity. Therefore, the optimization of an existing PCU plan is hardly ever considered and, consequently, the number of PCU re-assignments is kept to a minimum. Analysis of real data, given below, shows that, in existing networks, the configuration can be far from optimal, both in terms of the number of PCU re-selections and the imbalance between PCU loads.

3. Novel Method

The following section presents a simple approach to solve the CPAP based on data located in the Network Management System (NMS). Firstly, the need for a heuristic approach is justified based on the high computational load of exact solutions. The proposed algorithm is subsequently presented and some refinements of the algorithm are then suggested.

3.1. Preliminary Issues

In the context of live cellular networks, the size of the PCU planning problem prevents computationally expensive methods from being applied. The number of cells, adjacencies and BSCs (i.e., vertices, edges and problem instances, respectively) rapidly increases when a whole operator network is considered. For instance, a typical GERAN network comprises 200 BSCs, each one containing 150 cells and 1800 defined adjacencies on average.

Therefore, the speed of the method is critical, since the higher the time complexity of computation, the less the tool will be used, as it increases the workload on the NMS. Thus, operators might be forced to limit either the geographical area or the number of times this method can be applied in the network. Ideally, such a method to select a PCU should be run every time a new cell or PCU is added in the network or a significant change of user mobility trends is detected.

Most formulations of the graph partitioning problem are known to be NP-complete. The bounded, min-cut partitioning problem is NP-complete for arbitrary k, [24]. For fixed k, it is solvable in polynomial time, [25], but existing algorithms have a high-order polynomial complexity in the number of vertices, which is still exponential in k. The bounded, connected, min k-cut problem also proves to be NP-complete, since it can be reduced by transformation to a complete graph by adding zero-weight edges to the integer bin-packing problem, which is known to be NP-complete, [24]. Therefore, in general, it is not possible to compute optimal partitionings for graphs of interesting size in a reasonable amount of time. It can thus be concluded that the large size of the network to be optimized and the high time complexity of exact solutions justify the development of heuristic algorithms.

3.2. Outline of Proposed Method

In this work, complex techniques have been discarded for ease of implementation in live networks. In principle, the simplest approach is to refine the solution currently implemented in the network. Since this work deals with the problem of PCU re-assignment, an initial solution is normally available. However, experience with real networks has shown that this strategy based on the refinement of a previous solution is not the most effective one. In contrast, a repartitioning of a graph can also be computed by simply partitioning the new graph from scratch. This approach normally provides a significant performance improvement and will thus be followed here.

The core of the proposed method consists of a set of multi-level graph partitioning routines. While traditional graph partitioning methods work directly on the original graph, *Multi-Level* (ML) techniques first coarsen the graph by collapsing vertices and edges in order to reduce the size of the graph. Initial solutions are efficiently computed on smaller versions of the graph and later uncoarsened to obtain the partition of the original graph. After each uncoarsening step, local search-based refinement techniques are applied on small portions of the graph that are close to the partition boundary. This technique dramatically improves on single-level local search techniques in terms of solution quality and compares favorably in terms of runtime. From the previous explanation, it is clear that the ML strategy consists of three stages: *coarsening*, *initial partitioning* and *uncoarsening*. The following paragraphs describe the algorithms used in this work on each stage.

3.2.1. *Coarsening*

During the first stage, a sequence of smaller graphs is constructed from the original graph. In most coarsening schemes, sets of vertices (often pairs) are collapsed to form single vertices on the next level coarser graph. Several heuristics can be used to define a set of edges in which no two edges are incident on the same vertex (referred to as a *matching*). The effectiveness of a matching scheme depends on how successful it is in removing a significant amount of edge-weight from the successive coarser graphs. In this work, a matching is computed by selecting the heaviest edges on the graph during each coarsening step, [26]. It is worth noting that this method differs from the one suggested in [11], where vertices are visited in random order and their heaviest edge is selected. By hiding

the heaviest edges in a greedy fashion, a larger decrease of edge-cut is normally achieved when partitioning the coarser version of the graph. For this purpose, on each version of the graph, the edges are sorted by weight and selected in decreasing order, until no vertex is left unmatched or all edges have been selected. This matching method is referred to as *Sorted Heavy Edge Matching* (SHEM). The time complexity of the latter algorithm is dominated by that of the *Quicksort* algorithm, [27], used to rank the edges (i.e., $O(|E|log|E|)$ in the average case).

3.2.2. *Initial Partitioning*

The second stage of the ML algorithm computes an initial partition of the coarsest graph. In the simplest approach, [26], the coarsening process follows until the number of vertices in the coarsest graph is the same as the number of subdomains, k. Thus, the partition is built assigning each vertex v_i in the coarsest graph to subdomain S_i. However, this strategy sometimes suffers from slow reduction of graph size after some coarsening steps. Alternatively, the coarsening phase might end when the number of vertices falls below a certain threshold (e.g., c times the number of subdomains). In the coarsest graph, the *Greedy Graph Growing Partitioning* (GGGP) algorithm, [28], is adopted to build the initial partition. In its original version, a bisection of a graph is built by growing a subdomain incrementally around an arbitrary seed vertex. At each step, unassigned vertices are ordered in terms of the weight of edges to vertices already in the growing region. In this work, the k-way GGGP algorithm described in [23] has been used. In this algorithm, k initial seed vertices (instead of just 2) are chosen and neighboring vertices are alternately added to the subdomain with the smallest weight. Since the initial partition is performed over a simplified version of the graph, the time complexity of this algorithm is normally not a critical issue in ML approaches.

Two different strategies can be used to choose the seed vertices in the GGGP algorithm. The first one aims to maximize the average distance among seed vertices, where the distance between a pair of vertices is defined as the number of edges in the shortest path between them in the graph, [29]. More robust partitions are thus obtained at the expense of an increased runtime due to the calculation of the distance between every pair of vertices. The *Floyd-Warshall* (FW) approach, [27], is commonly used for this task, giving rise to the FW-GGGP algorithm. Unfortunately, the complexity of the FW algorithm is $O(|V|^3)$, which makes this approach infeasible for graphs of reasonable size. In the second strategy, the seed vertices are chosen arbitrarily. To improve the robustness of the method, a limited number of partitioning trials can be made with randomly selected seed vertices and the best solution in terms of edge-cut is selected, [28]. However, the effectiveness of this naive multi-start approach, hereafter referred as *Random-GGGP* (R-GGGP), is limited by the fact that random local optima in large problems tend to all have average quality and little variance. This means that the chance to improve a solution with this approach quickly diminishes from one iteration to the next.

The drawbacks of the aforementioned methods can be overcome by adaptive strategies that make the most of the multiple attempts. These methods exploit the globally convex structure of the optimization surface exhibited by large graph partitioning problems, which makes that the best local minima show strong similarities, [30]. The *Clustered Adaptive Multi-Start* (CAMS) method, [20], is here adopted to build the initial partition of the coarsest graph. Being a ML method itself, CAMS iteratively simplifies the graph not only based on the graph structure, as SHEM does, but also on the similarities of previous solutions. The algorithm starts with the generation of a limited set of solutions by the

naive multi-start strategy (i.e., R-GGGP). On each subsequent iteration, the algorithm identifies those groups of vertices that are assigned to the same subdomain in all solutions. A simplified version of the graph is thus built by collapsing all these vertices. A new set of solutions is then created from the refinement of the old set with the new simplified graph structure. As the iteration progresses, the graph gets simpler, since collapsed vertices remain unaltered by the refinement process. This iterative process ends when the number of vertices in the coarsest graph is equal to number of subdomains or all refined solutions coincide within an epoch. The main parameter of the algorithm is the number of solutions in each generation, s. This parameter controls the trade-off between diversification and intensification within the algorithm. Experiments showed that a value of $s = 5$ is a good compromise between solution quality and runtime.

It should be pointed out that, although there are other population-based partitioning methods (e.g., genetic algorithm, [31], ant colony, [32]), these do not aim to progressively simplify the graph (as CAMS does) and therefore tend to have a slower convergence to their best solution. In contrast, CAMS finds its best solution in a few iterations, leading to shorter runtimes for the same solution quality. Thus, CAMS is more suitable when good quality results are required quickly.

3.2.3. *Uncoarsening/Refinement*

During the last phase, the partition of the coarser graph is projected back to the original graph. As graphs get finer, the more degrees of freedom can be used to improve the partitioning. The refinement process developed in this work is based on the modification of the *Kernighan-Lin* (KL) heuristic, [8], proposed by *Fiduccia and Mattheyses* (FM), [33]. These hill-climbing algorithms consist of a number of passes through the vertices. Given a partition of a graph, the KL algorithm swaps those subsets of vertices in different subdomains that yield the greatest possible edge-cut reduction. The main strength of the algorithm is its ability to escape from local minima, because it explores movements that temporarily increase the edge-cut. The FM variant differs from the KL algorithm in that it moves only a single vertex at a time instead of swapping pairs of vertices. The time complexity of each step is thus reduced from $O(|V|^2 log|V|)$ to $O(|E|)$ at the expense of a higher imbalance, which is acceptable in the application considered here. On each pass, the edge-cut reduction of every possible move of a vertex is computed and vertices are ranked by benefit. The top vertex that still maintains the balance and connectedness constraints is selected to move.

From the computational perspective, the most challenging task is the fulfillment of the connectedness constraint during the refinement process. To satisfy this constraint, only vertices on the boundaries of subdomains should be moved to another subdomain. For that purpose, articulation vertices in connected subdomains must first be identified. A vertex a is said to be an *articulation vertex* of the subdomain S that it belongs to, if there exist vertices v, w in S, such that a, v, w are distinct, and every path between v and w contains a. To determine if a vertex is an articulation vertex, a version of the graph of a subdomain is constructed by removing that vertex. The resulting graph is checked for connectedness by traversing the graph as a tree in a Depth-First Search (DFS) manner, [27]. In this algorithm, some vertex is randomly defined as the root and exploration progresses by expanding the adjacent vertices and backtracking when a vertex with no new adjacent vertices is found. If any vertex in the subdomain is not reached in the search, the original (i.e., removed) vertex is an articulation vertex. Once articulation vertices have been identified, a candidate list is generated with the remaining vertices. Thus, a

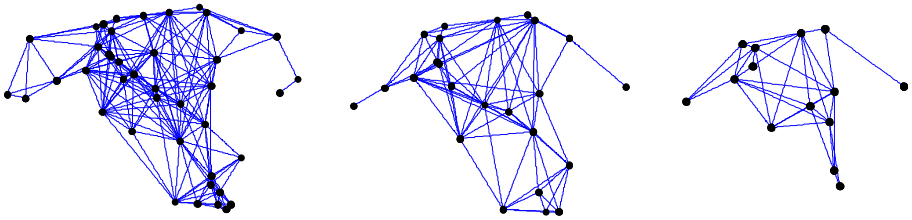


Figure 3. An example of graph coarsening.

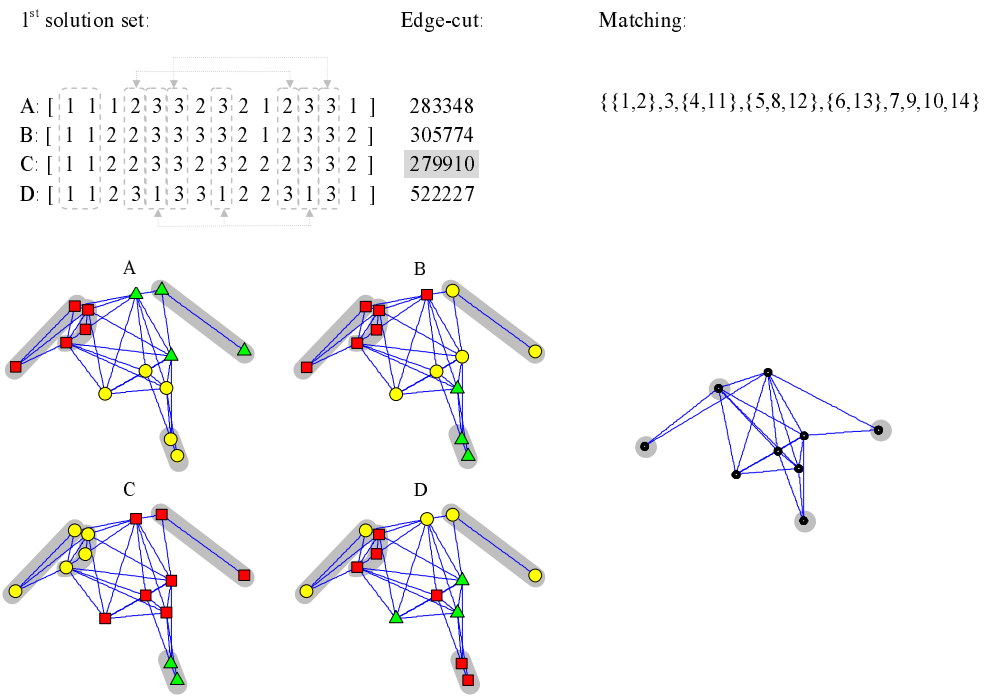
vertex can be moved only if: (a) it does not destroy the connectedness of the subdomain it leaves, (b) it does not lead to violation of weight constraints, and (c) it does not empty the subdomain it leaves. The worst-case time complexity of the connected refinement algorithm is $O(|V|^2(|V| + |E|))$, as a DFS must be performed for every vertex in the source subdomain after each vertex exchange. Hence, the overall time complexity of the proposed algorithm is driven by the latter algorithm. Although this time complexity seems to be high, in practice runtime did not follow that trend for two reasons. Firstly, the worst-case value is often a very pessimistic estimation of the running time of an algorithm, and this is especially true for the connected FM refinement algorithm with the type of graphs used here. Secondly, the graphs handled in this application are of medium size, so the hidden constants and terms in the O-notation might have an influence on the runtime.

3.3. Example of Partitioning by ML-CAMS

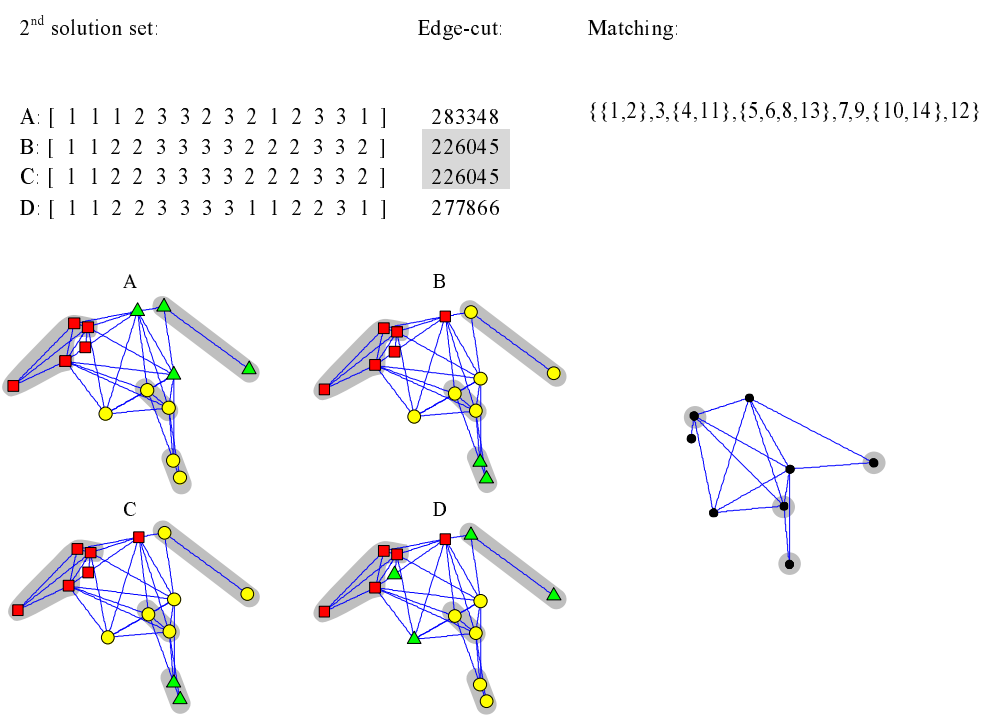
An example of partitioning in a real problem instance is presented. In the example, 42 base stations in a BSC must be distributed into 3 PCUs, provided that the PCU load imbalance ratio is less than 3.

Figure 3 presents the coarsening process, consisting of 2 coarsening steps. The original graph is depicted on the left, while the coarsened graph (where the initial partitioning is performed) is on the right. In the original graph, each vertex is represented on the site where the base station is physically located, as if it were on a map. Graph weights have been removed for clarity. From the original graph, it is clear that CPAP graphs are non-planar and can be highly heterogeneous, especially in terms of local structure, unlike the graphs derived from meshes in the supercomputing area, which have frequently been used to develop and test partitioning algorithms. Another distinctive feature, not shown in the figure, is the heterogeneity of the vertex weights, the largest of which can be up to eight times that of the smallest. This property makes the balancing of subdomain weights more complicated. In the figure, it is observed that, after coarsening, the number of edges and vertices in the graph is significantly reduced. Not shown is the fact that edges with large weights are hidden so that it is easier to find a good partition.

Figure 4 shows the initial partitioning performed by CAMS. Figure 4 (a) shows the result of the first iteration (step 0). The algorithm starts by computing s initial solutions by repeated use of R-GGGP. For space reasons, $s = 4$ in the example. In the figure, the four solutions (denoted by A-D) are described numerically (top) and graphically (bottom). The numerical representation shows the assignment of vertices to subdomains. Each column corresponds to one of the fourteen vertices in the graph and each of the four rows represents one partition. In the graphical representation, each of the three different symbols stands for a different PCU in the BSC. From the edge-cut values, it can be deduced that C is the best partition, since it has the lowest edge-cut (highlighted by a shaded box). In this set of solutions, the algorithm identifies clusters of vertices that are grouped under the



(a) Step 0



(b) Step 1

Figure 4. An example of initial partitioning by the Clustered Adaptive Multi-Start algorithm.

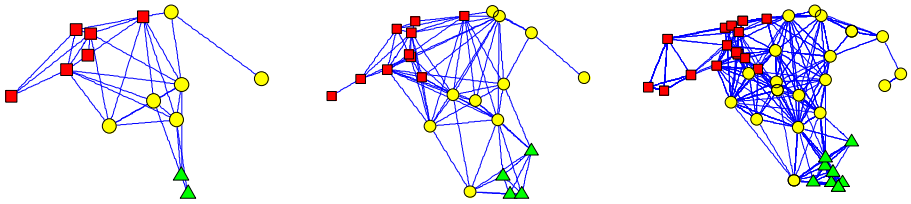


Figure 5. An example of graph uncoarsening.

same subdomain in all solutions. In particular, four clusters have been identified in this case (inside dashed lines). Note that this process is not concerned with the assignment of a single vertex to the same subdomain in all solutions (e.g., 9$^{\text{th}}$ vertex), but with entire clusters of vertices being assigned the same in all solutions (e.g., {1,2}, {4,11}, {5,8,12} and {6,13}). By matching these clusters of vertices, a simplified version of the graph (right) is obtained, as in any classical coarsening algorithm. Step 0 ends with the refinement of the solutions over the coarsened graph, resulting in a new set of solutions, presented in Figure 4 (b). It is observed that, although some solutions do not change after the refinement process (e.g., 1$^{\text{st}}$ solution), most of them are improved as their edge-cut is decreased (e.g., 2$^{\text{nd}}$, 3$^{\text{rd}}$ and 4$^{\text{th}}$ solution). Likewise, new vertices are now assigned consistently in all solutions, which is used to simplify the graph further. Refining this 2$^{\text{nd}}$ set of solutions over the new coarsened graph results in no change (not shown here) and the algorithm stops.

Figure 5 presents the uncoarsening process, consisting of 2 uncoarsening and refinement steps. The initial partitioning by CAMS is on the left, while the final solution is on the right.

3.4. Refinements to Outlined Method

In the proposed method, a repartitioning of a graph is computed by simply partitioning it from scratch. Since no concern is given to the initial solution, this approach tends to produce lots of changes in network configuration. By simply changing subdomain labels in the new solution in accordance with the old partitioning, the number of changes can be minimized. For every subdomain in the new solution, the remapping is performed by searching the subdomain in the initial solution with which it shares the largest number of vertices (or associated weight). A one-to-one correspondence is then defined between the two solutions.

At the same time, the connectedness constraint poses several problems in the presence of isolated cells (i.e., cells where no outgoing or incoming HO occurs). Although not normally considered in the literature, this situation is rather common in live networks. Although these vertices do not have an influence on edge-cut, they contribute to the weight of the subdomains and must be identified beforehand. Once identified, the subdomain to which these vertices belong will be considered connected if the remaining vertices are connected. Otherwise, these subdomains would tend to break into isolated clusters of vertices during the refinement process, due to the absence of articulation vertices.

4. Performance Analysis

The above-described method was applied to a collection of graphs constructed with data collected from a live GERAN. The aim of the analysis was to test the capability of the algorithm to solve the PCU assignment problem in a realistic environment. During

the analysis, the performance of different graph partitioning methods was estimated and compared. For clarity, the preliminary conditions are described first and subsequently the performance results are discussed.

4.1. Analysis Set-up

A programme was created to test the method with real data. The inputs of the programme were the HO statistics for a 2-week period and the configuration of GPRS, both residing on the NMS of a live GERAN. The network area under study comprised 8952 cells distributed over 61 BSCs. On these cells, an average of 12.7 adjacencies were actively used for HO purposes. As the CPAP is solved on a per-BSC basis, the set of problem instances considered in the analysis consisted of 61 CPAP graphs. It is worth noting that robust performance estimations are expected, since the collection of graphs covers a large geographical area with very different propagation and mobility environments.

As an optimization constraint, the set of cells assigned to a PCU could comprise a maximum of 256 GPRS TSLs, due to physical hardware limitations (i.e., B_{aw}=256). Likewise, based on operator demand, a maximum load imbalance ratio of 2 was permitted, which meant that the weight of the smallest subdomain had to be at least half the weight of the maximum one (i.e., B_{rw}=2). Finally, the connectedness constraint was enforced whenever possible.

4.2. Assessment Methodology

During the analysis, several graph partitioning approaches were tested. The main concern was to evaluate the proposed method that combines multi-level, adaptive multi-start and iterative refinement techniques. Specifically, the proposed multi-level method uses SHEM for coarsening, CAMS for initial partitioning and FM for uncoarsening/refinement. This method will hereafter be referred to as ML-CAMS.

To quantify the improvement of ML-CAMS on network performance, the initial operator solution (denoted as OI) was evaluated. In addition, several classical approaches were tested for comparison purposes. The first method was the FM refinement of the operator solution (denoted as OR for Operator Refined). The next ones were three variants of the GGGP algorithm. The first variant was the FW-GGGP method, which uses the FW algorithm to select seed vertices. The implemented version of this method was deterministic (i.e., the output solution was the same in any run of the algorithm). The second variant was the R-GGGP method based on the repeated use of GGGP with random seed selection and greedy refinement. The diversity provided by multiple attempts normally leads to very high-quality solutions, which can be used to estimate the performance of the optimal solution. The third variant was the adaptive multi-start heuristic, CAMS, where R-GGGP is used to build a set of initial solutions, which are checked for similarities to simplify the original graph iteratively. Another method was the traditional ML method (denoted as ML), where the initial partitioning is found by coarsening the graph until the number of vertices is the same as the number of subdomains. Finally, the analysis considered the proposed ML-CAMS method, which coarsens the graph partially and uses CAMS only for the initial partitioning.

All methods shared the same FM refinement algorithm, except R-GGGP, which used the greedy variant. Unless stated otherwise, the number of passes in the FM algorithm was set to 4. Likewise, all ML methods used the SHEM algorithm in the matching process, except inside CAMS, where the matching is built based on the similarities of previous

solutions. Finally, it is worth noting that OR, FW-GGGP and ML are deterministic. Consequently, a single run of these algorithms was performed. In contrast, CAMS, ML-CAMS and R-GGGP are randomized (and hence produce a different solution for each different random seed). To ensure the statistical confidence of results, performance figures for CAMS and ML-CAMS were computed from the average of 100 independent runs, while figures for R-GGGP corresponded to 1000 independent runs of the algorithm (i.e., the edge-cut of the best attempt and the runtime of the whole series).

The previous algorithms were all implemented from scratch. Although several codes are available in the public domain for ML coarsening and refinement operations, some limitations prevent them from being applied in the case considered here. First, it is sometimes not possible to adjust the maximum weight imbalance between subdomains, since perfect balance is almost always targeted. Likewise, despite some actions are normally taken to avoid non-connected solutions, experiments have shown that, in some cases, disconnected subdomains still exist in the final solution. Hence, although these problems can be circumvented with later post-processing, alternate routines have been developed for this work. In addition, avoiding the mixture of modules in different programming languages reduces the influence of implementation issues on algorithmic performance.

4.3. Assessment Criteria

Several performance indicators were evaluated during the assessment of the different methods. From the operator perspective, the main figure of merit is the inter-PCU HO ratio. Formally, this quantity represents the edge-cut normalized by the overall sum of edge weights and it is hereafter referred to as the *edge-cut ratio*. The PCU load imbalance ratio and the number of PCUs with unconnected cells were considered as secondary criteria. Finally, the execution time was also evaluated. For that purpose, the different routines were run on a Windows-based computer with a clock frequency of 2.4GHz and 1GByte of RAM.

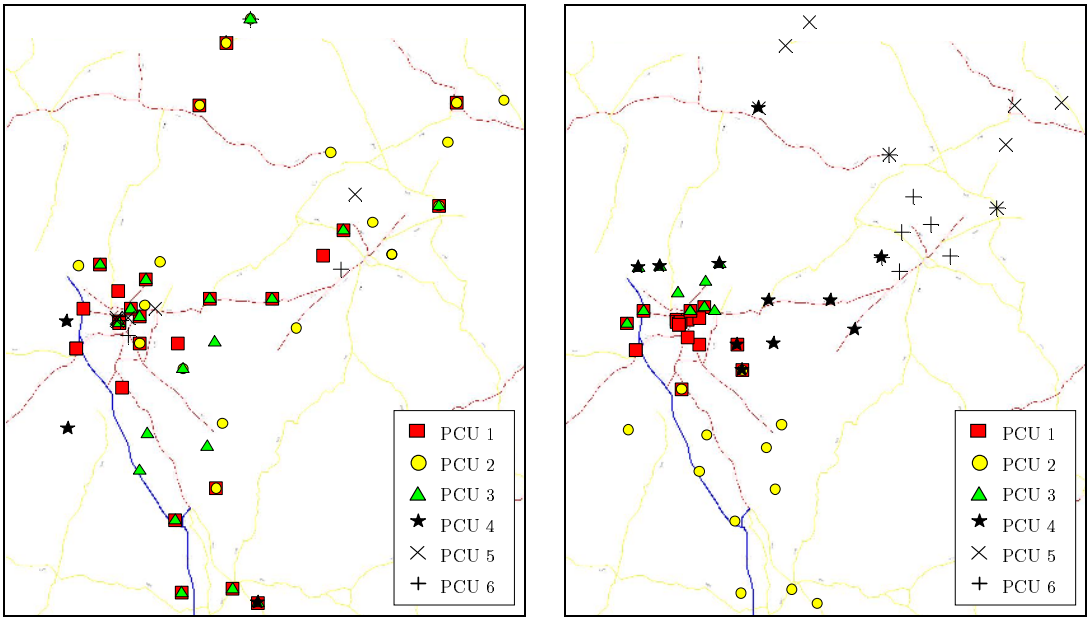
To assess the value of a given method, the trade-off between runtime and solution quality was evaluated following the methodology described in [34]. Most algorithms contain a parameter that allows the user to specify how long the search should continue before giving up (e.g., passes in the FM algorithm, runs in random seed selection). This parameter is commonly denoted as the intensity of the algorithm. For each intensity value, λ, and instance of the problem, p, the edge-cut, Q, and runtime, T, were averaged across the different runs, r, to give

$$\overline{Q}_{\lambda,p} = \frac{1}{n_r}\sum_{r=1}^{n_r} Q_{\lambda,p,r}\ , \qquad \overline{T}_{\lambda,p} = \frac{1}{n_r}\sum_{r=1}^{n_r} T_{\lambda,p,r}\ , \tag{4}$$

where n_r stands for the number of independent runs. To estimate the overall performance, relevant indicators were aggregated across the different instances, p, of the problem as

$$\overline{Q}_{\lambda} = \sum_{p=1}^{n_p} \overline{Q}_{\lambda,p}\ , \qquad \overline{T}_{\lambda} = \sum_{p=1}^{n_p} \overline{T}_{\lambda,p}\ , \tag{5}$$

where n_p stands for the number of instances (i.e., BSCs). For the sake of clarity, normalization against the performance of some reference solution is used to calculate the final performance figures. Thus, the normalized edge-cut, Q_λ, and normalized runtime, T_λ, are defined as



(a) Old PCU plan (b) New PCU plan

Figure 6. Map of the packet control unit plans in a BSC.

$$Q_\lambda = \overline{Q}_\lambda / Q^*_\lambda \,, \qquad T_\lambda = \overline{T}_\lambda / T^*_\lambda \,, \tag{6}$$

where Q^*_λ stands for the edge-cut of the best solution a priori (i.e., R-GGGP) and T^*_λ stands for the runtime of the fastest method (i.e., ML with no refinement). By using several intensity values, the trade-off between Q^*_λ and T^*_λ can be investigated to give an indication of the algorithmic performance of the different approaches.

4.4. Analysis Results

4.4.1. *Overall Radio Network Performance Comparison*

From the application of the methods, new PCU plans were built for every BSC in the network area. To visually inspect the impact of one of these plans, Figure 6 provides a map view of the old and new PCU plans for a single BSC. Concretely, the new plan was built by ML-CAMS. On the map, a symbol displays the location of a base station. Each of the six different symbols stands for a different PCU in the BSC. Figure 6 (a) shows how, in the old plan, base stations on the same PCU (denoted by the same symbol) do not always form a contiguous area. It is remarkable that base stations on the same site are sometimes assigned to two or even three different PCUs. This issue is especially serious in the vicinity of main roads, where most CRs will take place. By contrast, the new plan in Figure 6 (b) shows that cells belonging to the same PCU are grouped in geographical clusters. Therefore, for a randomly selected drive route in the area of the BSC, the number of intra-PCU CRs in the new PCU plan should be higher than the respective value in the old PCU plan.

This effect is corroborated in Table I, where the existing assignment is compared with the solutions obtained by the different methods. For comparison purposes, the edge-cut for the case where every cell has its own PCU is reflected as the sum of edge weights

Table I. Overall performance of graph partitioning methods based on handover statistics.

Heuristic method	OI	OR	FW-GGGP	ML	CAMS	ML-CAMS	R-GGGP
Sum of edge weights [$\cdot 10^6$]				243.7			
Total edge-cut [$\cdot 10^6$]	82.3	37.3	29.6	19.3	18.0	17.7	17.7
Edge-cut ratio [%]	33.8	15.3	12.1	7.9	7.4	7.3	7.3
Normalized edge-cut	4.65	2.11	1.68	1.10	1.02	1.00	1
Avg. max. no. of TSLs in subd.	91.4	93.2	90.5	90.9	92.1	94.1	91.5
Avg. weight imbalance ratio	3.55	2.66	1.88	2.00	1.90	1.91	1.82
Total no. of disconnected subd.	192	90	37	12	13.0	13.4	29
Total no. of changes	-	2870	4924	4536	4564	4550	4631
CPU time [s]	-	231	261	200	1703	455	74110

in the area. From the table, it is clear that all methods achieve a significant edge-cut reduction, when compared to the initial operator solution (i.e., OI). In particular, results show that the edge-cut ratio can be reduced from 33.8% ($=\frac{82.3}{243.7}$) to 7.3% ($=\frac{17.7}{243.7}$) (i.e., fivefold reduction). This result highlights the bad quality of the manual solution and justifies the need for the optimization process. Although all optimization methods share the same refinement algorithm, results show that it is beneficial to start from scratch with an initial partition where the load is well distributed among PCUs, which is done in all methods except OR. The large difference in edge-cut between OR and ML-CAMS or R-GGGP solutions proves this statement. Among the methods that partition the graph from scratch, ML-CAMS and R-GGGP show the lowest edge-cut ratio. It is worth noting that, although ML-CAMS shows similar performance on average to the best solution found by R-GGGP, some individual ML-CAMS attempts outperformed R-GGGP. Likewise, ML-CAMS solution has 10% less edge-cut ratio than the traditional ML method. From these results, it can be concluded that the edge-cut performance of ML-CAMS is similar to the best method a priori (i.e., R-GGGP).

At the same time, the load of the PCUs in the area is more evenly balanced in any of the new plans. This re-distribution of the number of GPRS TSLs per PCU translates into a reduction of the average weight imbalance ratio. With ML-CAMS, the imbalance ratio is nearly halved (i.e., $\frac{3.55}{1.91} = 1.86$). This load equalization provides spare capacity on each PCU, thus allowing the addition of new base stations to each of the PCUs without the need to build a new plan. It should be pointed out that most methods achieve the edge-cut reduction at the expense of a slight imbalance among subdomains (i.e., weight imbalance ratio close to 2). Nonetheless, enough margin is still available, since the average maximum number of TSLs under a PCU in the ML-CAMS solution is still only 37% ($=\frac{100*94.1}{256}$) of the 256 TSLs limit.

Similarly, the number of disconnected subdomains is greatly reduced by the method proposed when compared with other approaches (i.e., thirteen-fold reduction from the initial solution). This outcome stems from the sequence of connectedness checks within the refinement algorithm. Nonetheless, in some isolated cases, the algorithm was still unable to correct the lack of connection, due to the fact that the original graph was disconnected. It is also worth remarking that, although the refinement algorithm is shared among methods, the number of disconnected subdomains is very much dependent on the quality of the initial solution.

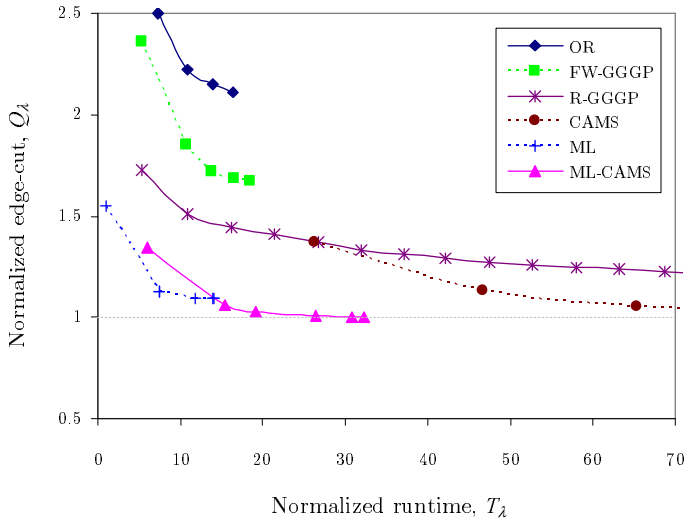


Figure 7. Plots of convergence behavior of the different methods.

As an outcome of the PCU re-planning procedure, a large number of changes is suggested by all strategies that build the initial partition from scratch (e.g., 4550 out of 8952 cells must be reallocated in the ML-CAMS solution). Although this figure might seem excessive, when compared to the number of cells in the trial area, it should be pointed out that a significant part of these must be performed to counteract the uneven PCU load in the initial solution. This result also points out the bad quality of the initial solution.

4.4.2. *Edge-cut vs Runtime Tradeoff*

The analysis subsequently focuses on the trade-off between the normalized edge-cut, Q_λ, and the normalized runtime, T_λ, in the different methods. For that purpose, the intensity of deterministic methods (i.e., OR, FW-GGGP and ML) was varied though the number of passes in the FM algorithm, while the intensity of random methods (i.e., CAMS and GGGP) was controlled by the number of iterations. In ML-CAMS, both parameters were simultaneously adjusted. Figure 7 shows that OR is outperformed by the rest of methods. As in other application areas, ML proves the best method whenever reduced runtime is targeted. On the other end, R-GGGP and CAMS should only be used for benchmarking purposes, due to their large execution time. While R-GGGP shows slow convergence, CAMS can significantly speed up the process of finding a very high-quality solution (note that, albeit not shown, R-GGGP converges to a normalized edge-cut value of 1, as it is the reference solution in terms of edge-cut). Finally, it is observed that ML-CAMS outperforms CAMS in terms of solution quality and runtime, which demonstrates the benefits from combining both ML approaches to enhance the efficiency of the method. Further analysis shows that, for all methods, most of the improvement in solution quality is achieved in the first pass of the refinement algorithm. Hence, greedy refinement performs competitively with FM refinement over this type of graphs.

4.4.3. *Influence of Optimization Constraints*

From the operator side, it is essential to know the sensitivity of the solution quality to the variation of constraints. As the number of trials in the optimization process is limited for practical reasons, the constraints cannot be freely modified and must therefore be set

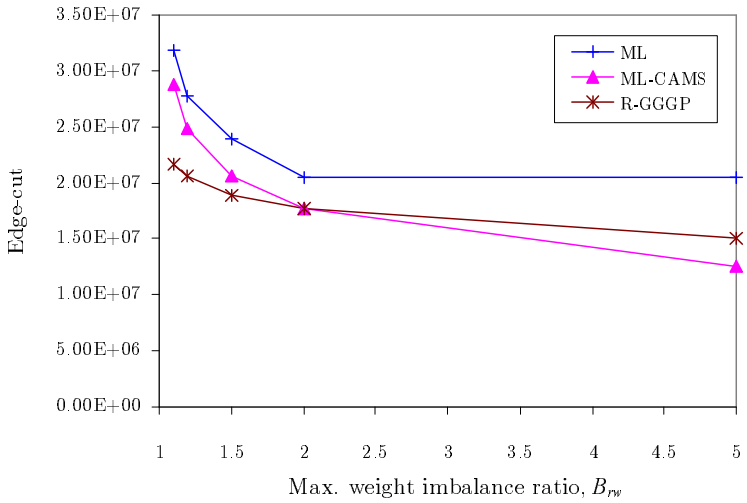
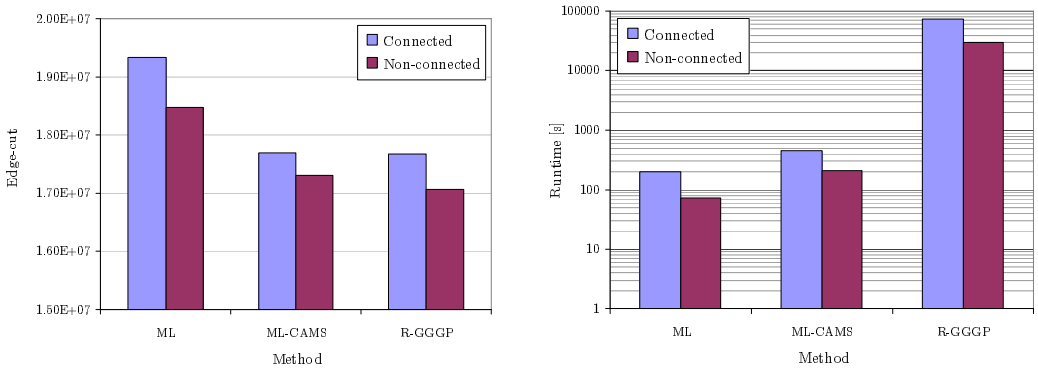


Figure 8. Sensitivity of the edge-cut to the weight imbalance constraint.

before the optimization process is launched. For clarity, the analysis is hereafter restricted to the ML, ML-CAMS and R-GGGP methods.

First, the analysis focuses on the imbalance constraint. Figure 8 presents the edge-cut dependence on the maximum weight imbalance ratio, B_{rw}. As expected, the total edge-cut decreases when the maximum load imbalance between PCUs is increased. From the figure, it is also deduced that relaxing the imbalance constraint beyond the value of 2 (which has been considered so far) hardly gives any edge-cut improvement for ML. By contrast, the same constraint relief is used by ML-CAMS to decrease the total edge-cut by 29%. Also remarkable is the fact that the solution achieved after 1000 R-GGGP attempts is worse than the ML-CAMS solution, although its runtime is two orders of magnitude larger. The origin of this effect can be found in the enlargement of the feasible solution space after relaxing the imbalance constraint. As a result, the naive multi-start approach needs more attempts to find the best solution when compared to ML-CAMS, which speeds up the convergence to the best solution. On the other hand, all methods encounter difficulties in finding an optimum solution for imbalance ratios lower than 1.5. For instance, the edge-cut of the ML-CAMS solution increases by 62% when the imbalance ratio changes from 2 to 1.1. Nonetheless, it can be observed that ML-CAMS experiences gentle degradation with the reinforcement of the weight imbalance constraint. Under these conditions, the naive multi-start approach in R-GGGP proves to give better quality solution at the expense of a higher running time. This result was expected, since the increased diversity of the search given by random trials becomes more valuable as the optimization problem gets more complicated.

Finally, the analysis focuses on the connectedness constraint. Figure 9 presents the performance of the different methods with and without the connectedness constraint in the refinement algorithm. Figure 9 (a) shows the influence of the constraint on the total edge-cut. It is observed that the elimination of this constraint leads to a limited edge-cut reduction in all methods. For instance, ML-CAMS only achieves a 2% edge-cut reduction when the connectedness constraint is eliminated. Obviously, the lack of a connectedness constraint entails an increase of the number of disconnected subdomains, which proved to be almost three-fold for all methods. Figure 9 (b) shows how the total runtime changes when the connectedness constraint is removed. From the figure, it can be deduced that



(a) Edge-cut (b) Runtime

Figure 9. Sensitivity analysis to the connectedness constraint.

the computational load can be significantly reduced by suppressing the connectedness constraint. This reduction is more than twice for any of the methods (note the logarithmic scale). Thus, it can be inferred that the computational load required by the connectedness checks in the connected refinement algorithm is more than half of the load of the overall algorithm. Nonetheless, the execution time of ML-CAMS for the whole network area is 8 minutes, which is low enough for network re-planning purposes. From these results, it can be concluded that the improvement of spatial consistency in the solutions achieved by enforcing the connectedness constraints more than compensates for the increase in edge-cut and runtime.

5. Conclusions

Despite their potential and widespread use in other industries, graph partitioning techniques have been very rarely used in the design and optimization of live cellular networks. To learn more about the benefit of graph partitioning for network operators, this technique has been applied to a particular problem, representative of those faced when structuring a cellular network, namely the assignment of cells to packet controllers in GERAN. One of the main challenges in providing packet-data services over existing GERAN networks is the delay and packet loss incurred during a cell change. Previous field trials have shown that intra-PCU cell re-selections experience far shorter transmission delays than inter-PCU cell re-selections. To enhance end-user experience of packet-data services, it is therefore required to maximize the ratio of intra-PCU cell re-selections to inter-PCU cell re-selections, while also ensuring that the load of the different PCUs in the same BSC is evenly balanced. In this work, a method to optimize the cell-to-PCU assignment has been proposed. The problem of assigning cells to PCUs has been formulated as a graph partitioning problem. The proposed method extends the traditional multi-level refinement heuristic used in [5] with adaptive multi-start techniques and connectedness checks. The method has been tested over an extensive set of graphs constructed from a live GERAN. Based on CS-HO statistics, it has been shown that ML-CAMS can reduce the intra-PCU cell re-selection ratio by 79%. At the same time, it is also possible to balance the load of the PCUs in the BSC, since the weight imbalance ratio was reduced from 3.5 to 1.9. The geographical consistency of the solution is also improved, as the number of disconnected subdomains is reduced by 93%. From these results, it can be concluded that

the proposed ML-CAMS method outperforms classical multi-level approaches in terms of solution quality at the expense of a slight increase in computing time. This superiority is more evident when the imbalance constraint is relaxed, which is acceptable to most operators.

It is worth noting that, although the present work is restricted to the assignment of PCUs in GERAN, similar principles can be applied to solve the clustering problem in other network layers and radio access technologies. For instance, the proposed method can be used to assign sites to BSCs in GERAN, [35]. In this case, the weight of each vertex represents the busy-hour traffic per site and each subdomain can have a different weight limit, since BSCs in the network may have a different capacity. In addition, the algorithm must check that both the site-to-BSC distance and the number of re-allocated base stations is kept below a certain limit. Similar constraints have to be considered in UTRAN for assigning base stations (called Node-B's) to Radio Network Controllers. The only difference is the fact that edge weights are obtained from soft-handover statistics. Likewise, the proposed method can be used to assign cells to location, routing and tracking areas in GERAN, UTRAN and E-UTRAN. In this case, edge weights are the number of idle users moving between cells (which can be estimated from handover statistics), vertex weights are the number of paging requests originated per cell and the subdomain weight limit is the maximum capacity of the paging channel. In E-UTRAN, the algorithm should be modified to allow a vertex to be assigned to more than one subdomain, since a base station (i.e., eNode-B) can belong to several tracking areas at the same time.

References

1. J. Lempiainen, M. Manninen, Radio Interface System Planning for GSM/GPRS/UMTS, Kluwer Academic Publishers, 2001.
2. J. Laiho, A. Wacker, T. Novosad, Radio Network Planning and Optimisation for UMTS, John Wiley & Sons, 2002.
3. A. R. Mishra, Fundamental of Cellular Network Planning and Optimisation, John Wiley & Sons, 2004.
4. T. Halonen, J. Melero, J. Romero, GSM, GPRS and EDGE Performance: Evolution Toward 3G/UMTS, John Wiley & Sons, 2002.
5. M. Toril, V. Wille, R. Barco, Optimization of the assignment of cells to packet control units in GERAN, IEEE Communications Letters 10 (3) (2006) 219 – 221.
6. K. Schloegel, G. Karypis, V. Kumar, Graph partitioning for high performance scientific simulations, in: J. Dongarra, I. Foster, G. Fox, K. Kennedy, A. White (Eds.), CRPC Parallel Computing Handbook, Morgan Kaufmann, 2000.
7. G. L. Miller, S.-H. Teng, W. Thurston, S. A. Vavasis, Automatic mesh partitioning, in: A. George, J. R. Gilbert, J. Liu (Eds.), Sparse Matrix Computations: Graph Theory Issues and Algorithms. (An IMA Workshop Volume), Springer-Verlag, 1993.
8. B. W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell System Technical Journal 49 (1970) 291–307.
9. A. Pothen, H. D. Simon, K.-P. Liou, Partitioning sparse matrices with eigenvectors of graphs, SIAM Journal on Matrix Analysis and Applications 11 (3) (1990) 430–452.
10. B. Hendrickson, R. Leland, A multilevel algorithm for partitioning graphs, in: Proc. 1995 ACM/IEEE Conference on Supercomputing, ACM Press, 1995.
11. G. Karypis, V. Kumar, Multilevel k-way partitioning scheme for irregular graphs, Journal of Parallel and Distributed Computing 48 (1) (1998) 96–129.
12. C. Walshaw, M. Cross, Mesh partitioning: a multilevel balancing and refinement algorithm, SIAM Journal of Scientific Computing 22 (1) (2000) 63–80.
13. A. Merchant, B. Sengupta, Assignment of cells to switches in PCS networks, IEEE/ACM Transactions on Networking 3 (5) (1995) 521–526.

14. D. Saha, A. Mukherjee, P. S. Bhattacharjee, A simple heuristic for assignment of cells to switches in a PCS network, Wireless Personal Communications 12 (2000) 209–224.
15. S. Pierre, F. Houeto, A tabu-search approach for assigning cells to switches in cellular mobile networks, Computer Communications 25 (5) (2002) 465–478.
16. I. Demirkol, C. Ersoy, M. U. Caglayan, H. Delic, Location area planning and cell-to-switch assignment in cellular networks, IEEE Transactions on Wireless Communications 3 (3) (2004) 880–890.
17. J. Plehn, The design of location areas in a GSM-network, in: Proc. 45th IEEE Vehicular Technology Conference, 1995, pp. 871–875.
18. P. Gondim, Genetic algorithms and location area partitioning problem in cellular networks, in: Proc. 46th IEEE Vehicular Technology Conference, 1996, pp. 1835–1838.
19. P. S. Bhattacharjee, D. Saha, A. Mukherjee, An approach for location area planning in a personal communication services network (PCSN), IEEE Transactions on Wireless Communications 3 (4) (2004) 1176–1187.
20. L. Hagen, A. Kahng, Combining problem reduction and adaptive multi-start: A new technique for superior iterative partitioning, IEEE Transactions On Computer-Aided Design of Integrated Circuits and Systems 16 (7) (1997) 709–717.
21. 3GPP TS 05.08, Digital cellular telecommunications system (Phase 2); Radio subsystem link control (Nov 2000).
22. 3GPP TS 43.129, Packet Switched Handover for GERAN A/Gb Mode; Stage 2 (May 2004).
23. R. Krishnan, R. Ramanathan, M. Steentrup, Optimization algorithms for large self-structuring networks, in: Proc. INFOCOM '99, Vol. 1, 1999, pp. 71–78.
24. M. Garey, D. Johnson, Computers and Intractability: A Guide to NP-Completeness, W.H. Freeman and Company, California, 1979.
25. O. Goldschmidt, D. S. Hochbaum, A polynomial algorithm for the k-cut problem for fixed k, Mathematics of Operations Research 19 (1) (1994) 24–37.
26. A. Gupta, Fast and effective algorithms for graph partitioning and sparse matrix ordering, IBM Journal of Research and Development 41 (1/2) (1997) 171–184.
27. T. H. Cormen, C. Stein, R. L. Rivest, C. E. Leiserson, Introduction to Algorithms, McGraw-Hill Higher Education, 2001.
28. G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM Journal on Scientific Computing 20 (1) (1998) 359–392.
29. R. Ramanathan, M. Steenstrup, Hierarchically-organized, multi-hop mobile wireless networks for quality-of-service support, Mobile Networks and Applications 3 (2) (1998) 101–119.
30. K. D. Boese, A. Khang, S. Muddu, A new adaptive multi-start technique for combinatorial global optimizations, Operation Research Letters 16 (1994) 101–113.
31. T. N. Bui, B. R. Moon, Genetic algorithm and graph partitioning, IEEE Transactions on Computers 45 (7) (1996) 841–855.
32. P. Korosec, J. Silc, B. Robic, Solving the mesh-partitioning problem with an ant-colony algorithm, Journal of Parallel Computing 30 (5-6) (2004) 785–801.
33. C. Fiduccia, R. Mattheyses, A linear time heuristic for improving network partitions, in: Proc. 19th ACM/IEEE Design Automation Conference, 1982, pp. 175–181.
34. C. Walshaw, Multilevel refinement for combinatorial optimisation problems, Annals of Operations Research 131 (2004) 325–372.
35. M. Toril, V. Wille, Optimization of the assignment of base stations to base station controllers in GERAN, IEEE Communications Letters 12 (6) (2008) 477–479.

Acknowledgements

This work has been supported by the Spanish Ministry of Science and Innovation (grant TEC2009-13413).