A Proposal of a Quasi-solution State Evolution Algorithm for Channel Assignment Problems

Nobuo Funabiki[1], Toru Nakanishi[1], Tokumi Yokohira[1], Shigeto Tajima[2], and Teruo Higashino[2]

[1] Department of Communication Network Engineering, Okayama University, 3-1-1 Tsushimanaka, Okayama 700-8530, Japan
[2] Department of Informatics and Mathematical Science, Osaka University, 1-3 Machikaneyama, Toyonaka 560-8531, Japan

Abstract. The channel assignment problem (*CAP*) in a cellular network requires finding a channel assignment to the call requests from cells such that three types of interference constraints are not only satisfied, but also the number of channels (*channel span*) is minimized. This paper presents a three-stage iterative algorithm, called the Quasi-solution state evolution algorithm for CAP (*QCAP*). QCAP evolutes *quasi-solution states* where a subset of call requests is assigned channels and no more request can be satisfied without violating the constraint. The first stage computes the lower bound on the channel span. After the second stage greedily generates an initial quasi-solution state, the third stage evolutes them for a feasible solution by iteratively generating best neighborhoods, with help of the dynamic state jump and the gradual span expansion for global convergence. The performance is evaluated through solving benchmark instances in literature, where QCAP always finds the optimum or near-optimum solution in very short time.

1 Introduction

This paper presents a heuristic algorithm called *QCAP*, a Quasi-solution state evolution algorithm for the Channel Assignment Problem (*CAP*) in a cellular network. Traffic demands for mobile communication have been rapidly increased in voice and data services, due to portability and availability in everyplace with no requirement for hard wires. Besides, the successful introduction of packet communications using mobile networks has accelerated the explosive growth of demands. On the other hand, the electromagnetic spectrum allocated for this system has been limited, because of a variety of significant applications using radio waves. Thus, the efficient use of precious frequency band resources has been an important task in the research/development communities in mobile communication networks. The concept of the cellular network has been widely adopted as the efficiency realization [1]. This cellular network allows the reuse of the same channel in geographically separated regions simultaneously. As a result, CAP has become the critical problem to be solved for its efficient solutions. A solution for CAP is not only requested to avoid the mutual radio interference between closer channels, but also to maximize the channel utilization.

Due to the NP-hardness of CAP [2], a number of polynomial time approximation algorithms have been reported [5]-[16]. In [5], Sivarajan et al. proposed eight different greedy algorithms based on graph coloring algorithms. They referred practical benchmark instances of 21 cells. Their algorithms and these benchmarks have been widely used for performance evaluations in many literature. In [6], Kunz proposed a neural network algorithm using continuous sigmoid neurons for a limited case of CAP, which does not consider the adjacent channel constraint. He provided a practical benchmark instance of 25 cells. In [7], Funabiki et al. proposed another neural network algorithm using binary neurons for the general CAP. In [8], Wang et al. proposed a two-phase adaptive local search algorithm named *CAP3*. They showed its superiority through solving a subset of Sivarajan's benchmarks, Kunz's benchmark, and Kim's benchmarks. In [9], Sung et al. proposed a generalized sequential packing (*GSP*) algorithm with its two variations for CAP, and a lower bound on the number of channels (*channel span*). In [10], Hurley et al. proposed an integrated system called *FASoft*, which incorporates various CAP algorithms. In [11], Rouskas et al. proposed an iterative heuristic algorithm for CAP. They provided benchmark instances of 49 cells. In [12], Beckmann et al. proposed a hybrid algorithm composed of a genetic algorithm and the frequency exhaustive strategy. They evaluated the performance through solving a subset of Sivarajan's benchmarks. In [13], Funabiki et al. proposed a neural network algorithm combined with heuristic methods for CAP. In [14][15], Murakami et al. proposed a genetic algorithm using several schemes to improve the convergence property. This algorithm is only applicable to simple cases of CAP without considering the interference between different channels. They provided benchmark instances of 49, 80, and 101 cells. In [16], Matsui et al. proposed a genetic algorithm to determine the sequence of cells for assigning channels by a greedy method for CAP. Unfortunately, none of existing algorithms can find optimum solutions for small size benchmark instances whose lower bounds are known.

QCAP evolutes quasi-solution states through three stages to provide the high quality solution in short computation time. A *quasi-solution state* represents a channel assignment to a subset of call requests where no more call request in any cell can be assigned a channel without violating the constraint. When the full set of call requests is assigned channels, it becomes a solution. The *first stage* computes the lower bound on the channel span. The *second stage* greedily generates an initial quasi-solution state. The *third stage* iteratively evolutes quasi-solution states to a feasible channel assignment, by iteratively generating best neighbor states, while schemes of the *dynamic state jump* and the *gradual span expansion* are used together for global convergence. The performance is evaluated through solving benchmark instances, where the comparisons with existing results confirm the superiority of QCAP.

2 Problem Formulation of CAP

CAP in this paper follows the common problem formulation defined by Gamst et al. [3] as in literature. The servicing region in a cellular network is managed as a set of disjoint hexagonal cells. Each cell occupies a unit area for providing communication services to users that are located in the cell area. When a user requests a call for communication services, a channel must be assigned to the user through which voice or data packets are communicating between the user's mobile terminal and a base station. This channel assignment must satisfy the constraints to avoid the mutual radio interference between closer channels. In CAP, the following three types of constraints have been considered:

1) *Co-Channel Constraint (CCC)*: the same or its adjacent channels cannot be reused in the cells that are located within a specified distance from each other in the network. This set of channel-reuse prohibited cells is called a *cluster*. In a cluster, any pair of channels assigned to call requests from the cells must have a specified channel distance.
2) *Adjacent Channel Constraint (ACC)*: adjacent channels cannot be assigned to adjacent cells in the network simultaneously. In other words, any pair of channels assigned to adjacent cells must have a specified distance. The distance for ACC is usually larger than that for CCC.
3) *Co-Site Constraint (CSC)*: any pair of channels in the same cell must have a specified distance. The distance for CSC is usually larger than that for ACC.

The channel distance is described by the difference on the channel indices in the channel domain. In this paper, the cell cluster size for CCC is denoted by "Nc", the channel distance to satisfy CCC is by "cij", the distance for ACC is by "acc", and the distance for CSC is by "cii" as in existing papers. The goal of CAP is to find a channel assignment to every call request with the minimum number of channels or *channel span* subject to the above three constraints. The three constraints in an N-cell network are altogether described by an $N \times N$ symmetric *compatibility matrix* C. A non-diagonal element c_{ij} $(i \neq j)$ in C represents the minimum distance to be separated between a channel in cell i and a channel in cell j. A diagonal element c_{ii} in C represents the minimum distance between any pair of channels in cell i.

A set of call requests in the N-cell network is given by an N-element *demand vector* D. The i-th element d_i in D represents the number of channels to satisfy the call requests in cell i. Let a binary variable x_{ik} represent whether channel k be assigned to cell i $(x_{ik} = 1)$ or not $(x_{ik} = 0)$ for $i = 1, ..., N$ and $k = 1, ..., M$. Note that M represents the channel span required for the instance. Then, CAP is defined as follows:

$$\begin{aligned}
&\text{minimize } M \text{ such that} \\
&x_{ik} = 0 \text{ or } 1, \text{ for } i \in \{1, ..., N\} \text{ and } k \in \{1, ..., M\} \\
&\sum_{k=1}^{M} x_{ik} = d_i, \text{ for } i \in \{1, ..., N\}
\end{aligned}$$

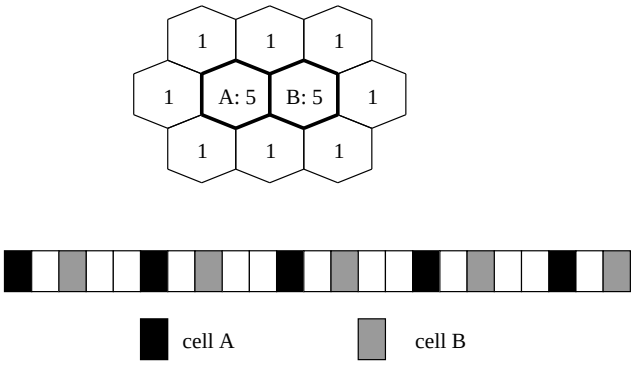

Fig. 1. A CAP example for the improved lower bound formula.

$$|k-l| \geq c_{ij}, \text{ for } k,l \in \{1,...,M\}, i,j \in \{1,...,N\} \text{ and } x_{ik} = x_{jl} = 1. \quad (1)$$

3 Proposal of QCAP

3.1 Lower Bound Computation

The first stage of QCAP computes the lower bound LB on the channel span for the initial value of M using the following three formulas, based on [4][9] with some improvement. The first lower bound formula gives the channel span to afford every call request in one cell while satisfying CSC by cii:

$$LB_1 \geq 1 + cii\,(d_i - 1)\,. \quad (2)$$

When two cells in the same cluster have the same number of call requests in an instance such as Rouskas's benchmarks, additional channels are required to satisfy these call requests from both cells simultaneously:

$$LB_1 \geq 1 + cii\,(d_i - 1) + cij \quad (3)$$

where cells i and j are located in the same cluster to be mutually interfered by CCC or ACC, and have the same number of call requests. Figure 1 illustrates an example channel assignment with $cii = 5$ and $cij = 2$ for this special case. Each number inside a hexagonal cell represents the number of call requests. Each of five call requests from cell A is assigned a channel with the interval of five channels, and each of four requests from cell B is assigned a channel between two neighboring channels assigned to cell A. Consequently, two additional channels are necessary to assign the last call request from cell B to satisfy $cij = 2$.

The second lower bound formula gives the channel span to afford every request call from one cell, namely the *center cell*, and from its surrounding adjacent

cells while satisfying ACC:

$$LB_2 \geq 2acc + (d_i - 2)(2acc - 1) + cij \left(\sum_{j \in P_i} d_j \right) \tag{4}$$

where P_i consists of the cells that are adjacent to cell i. The first two terms represent the number of channels that the channel assignment to the center cell forbids from the assignment to the surrounding cells to satisfy ACC. The last term represents the number of channels required to satisfy the call requests in the surrounding cells to satisfy CCC.

The third lower bound formula gives the channel span to afford every call request from the same cluster of cells while satisfying CCC:

$$LB_3 \geq 1 + cij \left(\sum_{j \in Q_i} d_j - 1 \right) \tag{5}$$

where the cell set Q_i consists of the cells that are located in the same cluster as cell i.

3.2 Greedy Initial State Generation

The second stage adopts the *requirement exhaustive strategy* in [5], to produce an initial quasi-solution state with the lower bound span, where as many call requests as possible are assigned channels greedily. The *unsatisfied cell index list* L_{cell} is initialized by sorting cells in descending order of cell degrees [5]:

$$degree_i = \left(\sum_{j=1}^{N} d_j c_{ij} \right) - c_{ii}. \tag{6}$$

An *unsatisfied cell* represents a cell where some of its call requests are not assigned channels. A variable indicating the number of assigned channels, $assign_i$, is introduced for cell i, where $assign_i < d_i$ indicates cell i is an unsatisfied cell. In this list generation, the tiebreak is resolved randomly when two or more cells have the same degree. In addition, when LB_1 gives the lower bound LB for an instance in the first stage, each call request from the corresponding cell is assigned a channel with the interval of cii, and this assignment is fixed throughout search process. The fixed cell index is denoted by F in this paper for convenience.

3.3 Quasi-Solution State Evolution

Next State Generation The third stage evolutes quasi-solution states by repeatedly generating best neighbor states in terms of the following cost function to evaluate the violation by the assignment of channel k to cell i:

$$cost_{ik} = \sum_{\substack{j=1 \\ c_{ij} \geq 1}}^{N} \sum_{l=k-c_{ij}+1}^{k+c_{ij}-1} w_j x_{jl}. \tag{7}$$

A next state generation is initiated by randomly selecting one unsatisfied cell from L_{cell} to avoid biased movements. Then, one channel is selected for its assignment, such that this channel is not only assigned to this cell currently, but also minimizes $cost_{ik}$ where the tiebreak is always resolved by selecting the least index channel. The cell weight w_j is introduced to encourage the channel assignment to cells that are hard to be assigned otherwise as in [17]. Here, two auxiliary conditions are imposed in the channel selection. One is the prohibition of selecting a channel in a *tabu list* to avoid cyclic state transitions. The tabu list describes the channels that have been selected within the predefined number of iteration steps T_{tabu} since its last selection to the corresponding cell. Another is the prohibition of selecting channels conflicting with the fixed cell assignment.

However, if QCAP only repeats transitions to best neighbor states, it may cause stagnation of state changes. To provide a hill-climbing capability of escaping from local minimum, one channel is sometimes randomly selected, which is called the *random selection*. The random selection is applied when the state has not been improved during the constant number of steps. Besides, the weight w_i associated with each unsatisfied cell in L_{cell} is incremented by 1 at the same time, so as to encourage these cells to be assigned channels more progressively than others.

To retrieve a quasi-solution state after this new channel assignment, every conflicting assignment with it is sought a new feasible channel assignment. If an assignable channel is found, it is assigned to the cell. Otherwise, the channel assignment is cleared, and the cell is inserted into L_{cell} if it is not there. After every conflicting one is handled, each cell in L_{cell} is checked whether a new channel can be assigned or not. If assignable, this channel is assigned there.

Dynamic State Jump Even several trials of the random selection may not provide enough state fluctuations to escape from local minimum. In such situations, QCAP induces the *dynamic state jump* for big changes from previous states while maintaining the achieved solution quality. Firstly, the best state in terms of the number of satisfied call requests is retrieved as the initial state for different evolutions. The best state that has been visited is memorized in QCAP. Then, channel assignments in this state are repeatedly shuffled until the half of assigned call requests in each cell may receive different channels from current ones in each dynamic state jump. In each shuffle movement, one cell is first randomly selected from cells that have movable assignments. Then, one call request with an assigned channel in this cell is randomly selected. The call request is assigned a randomly selected new channel if there is a channel that satisfies the two conditions: 1) the channel is not currently assigned to any call request in this cell, and 2) the new assignment is compatible with other channel assignments. When such a channel does not exist, the scheme is terminated. After the dynamic state jump, each cell in L_{cell} is sought a channel assignment, to retrieve a quasi-solution state, and the tabu list is cleared.

Gradual Span Expansion In some CAP instances, the lower bound on the channel span in the first stage is too small to afford every call request. In such cases, QCAP gradually expands the channel span until it reaches a feasible solution. Actually the span expansion is carried out when the state in QCAP has not been improved after several trials of the dynamic state jump. In each span expansion, the best state is first retrieved as a current state as in the dynamic state jump. Then, the channel span M is incremented by ΔM given by the following formula:

$$\Delta M = \left\lfloor \alpha \left(\sum_{i=1}^{N} d_i - \sum_{i=1}^{N} assign_i \right) / Nc \right\rfloor ,$$
$$\text{if } \Delta M < 1 \text{ then } \Delta M = 1 \tag{8}$$

where Nc is the cluster size for CCC, α is a constant parameter, and the *floor function* $\lfloor x \rfloor$ returns the maximum integer smaller than or equal to x. This equation is derived from a conjecture that a new channel can afford another call request from every cell cluster. After M is expanded, cells in L_{cell} are sequentially assigned these expanded channels by the requirement exhaustive strategy. Then, the dynamic state jump is applied for a better restarting state. At the same time, every cell weight w_i is initialized by 1 for $i = 1, ..., N$, to reset the search direction.

4 Simulation Results

Benchmark instances in Tables 1- 4 are solved to evaluate the performance of QCAP. A total of 10 runs are repeated with different random numbers in each instance. The tables show the instance number, the constraint parameters (Nc, acc, cii), the lower bound on the channel span (LB) in the first stage of QCAP, the average channel spans in solutions (M), and the average computation time (seconds) on Pentium-III 800 MHz by QCAP in each instance. Besides, the existing results on channel spans in literature are also summarized there. Note that only the best result among several versions of their algorithms in [5], [9], [10], and [11] is described to simplify comparisons and save the space. In Table 4, "$> x$" indicates that they cannot find a feasible assignment using x channels.

These tables suggest that QCAP finds the optimum solution with the lower bound on the channel span by any run for each CAP instance in less than one second, except for three instances 23, 25, and 27. In two instances 23 and 25, QCAP always finds a solution that requires one more channel than the lower bound. The more accurate lower bound formula might clarify that the obtained solutions by QCAP are their real lower bounds. On the other hand, the existing algorithms require several versions of procedures and/or many repeated runs with different random numbers to reach the optimum solution. For example, the genetic algorithm in [12] can reach the lower bound solution for the hard instance 10 by Sivarajan in only 1 run among 50 runs with different random numbers. Besides, it takes several thousand of iteration steps for convergence.

The simplicity and the search efficiency reveal that the proposed QCAP is a very practical and powerful tool to solve the important task of channel assignments in the cellular network.

Table 1. Simulation Results for CAP instances by Sivarajan.

Instance	Constraint				QCAP		Existing results					
No.	Nc	acc	cii	LB	M	time(s)	[5]	[8]	[9]	[10]	[11]	[12]
1	12	2	5	427	427	0.238	460	-	440	427	440	-
2	7	2	5	427	427	0.094	447	433	436	427	436	427
3	12	2	7	533	533	0.002	536	-	533	-	533	-
4	7	2	7	533	533	0.036	533	533	533	-	533	533
5	12	1	5	381	381	0.003	381	-	381	-	381	-
6	7	1	5	381	381	0.000	381	381	381	-	381	381
7	12	1	7	533	533	0.003	533	-	533	-	533	-
8	7	1	7	533	533	0.002	533	533	533	-	533	533
9	12	2	5	258	258	0.087	283	-	273	258	287	-
10	7	2	5	253	253	0.329	270	263	268	253	269	253
11	12	2	7	309	309	0.026	310	309	309	-	309	-
12	7	2	7	309	309	0.019	310	309	309	-	309	309
13	12	2	12	529	529	0.007	529	529	529	-	529	-

Table 2. Simulation Results for CAP instances by Kunz.

Instance		QCAP		Existing results	
No.	LB	M	time(s)	[8]	[10]
14	73	73	0.003	73	73

5 Conclusion

This paper has presented QCAP, a quasi-solution state algorithm for the channel assignment problem in the cellular network. The performance is evaluated through solving benchmark instances, where the comparisons to the existing CAP algorithms confirm the extensive search capability and the efficiency of QCAP. The study on the tighter lower bound of the channel span is essential to further improve the performance.

Table 3. Simulation Results for CAP instances by Rouskas.

Instance	Constraint				QCAP		Existing results	
No.	Nc	acc	cii	LB	M	time(s)	[5]	[11]
15	12	2	5	468	468	0.237	486	470
16	7	2	5	468	468	0.158	481	470
17	12	2	7	484	484	0.119	520	484
18	7	2	7	484	484	0.301	523	484
19	12	1	5	413	413	0.175	422	414
20	7	1	5	346	346	0.052	349	346
21	12	1	7	484	484	0.005	484	484
22	7	1	7	484	484	0.004	484	484
23	12	2	5	273	274	3.557	307	298
24	7	2	5	253	253	0.245	275	266
25	12	2	7	273	274	5.098	330	301
26	7	2	7	262	262	0.628	297	275
27	12	2	12	447	447	1.015	447	447

Table 4. Simulation Results for CAP instances by Murakami.

Instance	Constraint					QCAP		Existing results		
No.	N	Nc	acc	cii	LB	M	time(s)	[14]	[15]	[16]
28	49	7	1	1	22	22	0.006	24	24	22
29	49	7	1	1	21	21	0.005	-	24	-
30	49	7	1	1	26	26	0.008	-	>24	-
31	80	7	1	1	22	22	0.031	>24	-	24
32	101	7	1	1	22	22	0.038	>24	-	24

References

1. A. H. MacDonald, "Advanced mobile phone service: the cellular concept," Bell Syst. Tech. J., vol. 58, pp. 15–41, Jan. 1979.
2. W. K. Hale, "Frequency assignment: theory and applications," Proc. IEEE, vol. 68, no. 12, pp. 1497–1514, Dec. 1980.
3. A. Gamst and W. Rave, "On frequency assignment in mobile automatic telephone systems," in Proc. GLOBECOM, 1982, pp. 309–315.
4. A Gamst, "Some lower bounds for a class of frequency assignment problems," IEEE Trans. Veh. Technol., vol. VT-35, no. 1, pp. 8–14, Feb. 1986.
5. K. N. Sivarajan, R. J. McEliece, and J. W. Letchum, "Channel assignment in cellular radio," in Proc. 39th IEEE Veh. Technol. Conf., 1989, pp. 846–850.
6. D. Kunz, "Channel assignment for cellular radio using neural networks," IEEE Trans. Veh. Technol., vol. 40, no. 1, pp. 188–193, Feb. 1991.
7. N. Funabiki and Y. Takefuji, "A neural network parallel algorithm for channel assignment problems in cellular radio networks," IEEE Trans. Veh. Technol., vol. 41, no. 4, pp. 430–437, Nov. 1992.
8. W. Wang and C. K. Rushforth, "An adaptive local-search algorithm for the channel assignment problem (CAP)," IEEE Trans. Veh. Technol., vol. 45, no. 3, pp. 459–466, Aug. 1996.
9. C. W. Sung and W. S. Wong, "Sequential packing algorithm for channel assignment under cochannel and adjacent-channel interference constraint," IEEE Trans. Veh. Technol., vol. 46, no. 3, pp. 676–686, Aug. 1997.
10. S. Hurley, D. H. Smith, and S. U. Thiel, "FASoft: a system for discrete channel frequency assignment," Radio Science, vol. 32, no. 5, pp. 1921–1939, Sep.–Oct. 1997.
11. A. N. Rouskas, M. G. Kazantzakis, and M. E. Anagnostou, "Minimization of frequency assignment span in cellular networks," IEEE Trans. Veh. Technol., vol. 48, no. 3, pp. 873–882, May 1999.
12. D. Beckmann and U. Killat, "A new strategy for the application of genetic algorithms to the channel-assignment problem," IEEE Trans. Veh. Technol., vol. 48, no. 4, pp. 1261–1269, July 1999.
13. N. Funabiki, N. Okutani, and S. Nishikawa, "A three-stage heuristic combined neural-network algorithm for channel assignment in cellular mobile systems," IEEE Trans. Veh. Technol., vol. 49, no. 2, pp. 397–403, March 2000.
14. H. Murakami, Y. Ogawa, and T. Ohgane, "Channel allocation in mobile communications using the genetic algorithm," IEICE Tech. Report, RCS98-48, pp. 87–94, 1998.
15. H. Murakami, Y. Ogawa, and T. Ohgane, "Fixed channel allocation method in mobile radio communications using the genetic algorithm," IEICE Trans., vol. J83-B, no. 6, pp. 769–779, June 2000.
16. S. Matsui and K. Tokoro, "A fast genetic algorithm using allocation order for fixed channel assignment in mobile communications," IEICE Trans., vol. J83-B, no. 5, pp. 645–653, May 2000.
17. B. Selman and H. Kautz, "Domain-independent extensions to GSAT: solving large structured satisfiability problems," in Proc. 13th Int'l Joint Conf. Artificial Intelligence, pp. 290–295, 1993.