An Efficient Approach for Distributed Channel Allocation in Cellular Mobile Networks

Yongbing Zhang
Institute of Policy and Planning Sciences, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan
ybzhang@sk.tsukuba.ac.jp

Xiaohua Jia
Department of Computer Science, City University of Hong Kong, Tat Chee Ave., Kowloon, Hong Kong
jia@cs.cityu.edu.hk

Sajal K. Das
Department of Computer Science and Engineering, University of Texas at Arlington, Texas 76203-1366, USA
das@cse.uta.edu

ABSTRACT

We propose a distributed channel allocation algorithm based on a threshold scheme, called D-CAT, for cellular mobile networks. The D-CAT algorithm employs two thresholds: (i) a *heavy threshold* used for determining whether a cell is heavy, i.e., overloaded, and for triggering the channel allocation algorithm; and (ii) a *target threshold* used for indicating the target number of free channels that a heavy cell intends to acquire. We determine the optimal number of free channels as well as the cell(s) from where a heavy cell should import to satisfy its channel demand. Simulation experiments and analyses show that the D-CAT algorithm incurs lower overhead for channel allocation and is more efficient in terms of channel utilization than other distributed channel allocation algorithms. It also outperforms other centralized and distributed algorithms in terms of call blocking probability.

1. INTRODUCTION

Because of the rapid growth in mobile communication users, efficient management and sharing of the scarce radio resources in cellular networks become an important issue. In order to utilize the resources effectively, the geographical coverage area is divided into cells and the radio spectrum is reused in non-interfering cells. Depending on the various technologies such as Frequency Division (FD), Time Division (TD), and Code Division (CD) [13], the radio spectrum is further divided into channels to serve different calls. Many schemes have been proposed to allocate channels to the cells such that the available channels are efficiently used and thus the channel reuse is maximized [1, 3, 7, 9, 11, 12, 14, 17, 18, 20, 21]. The *performance metric* used for measuring the efficiency of a channel allocation scheme is the *call blocking probability*, i.e., the sum of the probabilities of new call blocking as well as forced termination.

DIAL M 2001 7/01 Rome, Italy

Channel allocation strategies can be broadly classified into two categories: *fixed* [20] and *dynamic* [1, 7, 19]. A combination of these two strategies is also possible [18]. A *fixed* allocation (FA) strategy is to allocate a fixed set of channels to each cell permanently. The same set of channels is reused by another cell at sufficient distance away. The advantage of the FA strategy is its simplicity. Its disadvantage, however, is that if the number of calls exceeds the number of channels allocated to a cell, the excess calls have to be blocked. Variations of FA strategies, in which an overloaded cell borrows free channels from its neighboring cells provided that the borrowed channels do not interfere with the existing calls, show significant performance improvement [9, 10, 14, 21]. In contrast, a *dynamic* allocation (DA) strategy is to allocate the channels in the system dynamically. The system maintains and manages a global pool of free channels and assigns the channel with the minimum cost to each arriving call. A cell has no channels to itself but requests for free channels if necessary. A cell can use any channel that does not violate the channel reuse constraint. The DA strategies tend to be more efficient than the FA strategies in conditions of light, non-homogeneous, and time-varying traffic but at the cost of high implementation overhead.

In addition to the issue of how to allocate channels among cells, who plays a key role in a channel allocation decision is also very important. Most of the algorithms in the literature depend on a mobile switching center (MSC) to accomplish channel allocation. These schemes are referred to as *centralized* channel allocation algorithms [1, 8, 9, 17, 21]. The disadvantage of such schemes is that the MSC may be overloaded and the failure of the MSC makes the whole system down. Recently, *distributed* channel allocation strategies have also been proposed [3, 4, 5, 6, 8, 11, 12, 16]. Each base station (BS) at a cell plays a key role in a channel allocation decision and is capable of running the channel allocation algorithm if the cell gets overloaded with channel demand. The main advantage of a distributed algorithm is its high reliability and scalability but its main disadvantage is its high implementation cost, i.e., high overhead cost for message exchanges among the cells, distributed time clock and resource management, etc. The overhead cost may increase exponentially in the worst case as the number of cells in the system increases.

In this paper, we propose a distributed channel allocation algorithm, called D-CAT, based on a *threshold* scheme. The D-CAT algorithm employs two thresholds: (i) a *heavy* threshold used for determining whether a cell is heavy, i.e., overloaded, and for triggering the channel allocation algorithm; and (ii) a *target* threshold used for indicating the target number of free channels that a heavy cell intends to acquire. The heavy threshold is a predefined parameter. However, the target threshold is dynamically determined by the average number of free channels of a cell within the interference distance relative to the heavy cell at a given instant. Each cell maintains and manages a group of channels as its own property and determines itself whether it needs more free channels. When a cell becomes heavy, the event of a new call arrival at the cell triggers the channel allocation algorithm to import free channels. The optimal number of free channels and from where the heavy cell should import are clearly determined using the proposed two-threshold scheme. As a result, the overhead cost for channel allocation is minimized and the available channels among the cells are balanced.

The rest of this paper is organized as follows. Section 2 describes the cellular system model. Section 3 presents the D-CAT algorithm proposed in this paper. Section 4 evaluates the performance of D-CAT in comparison with three other existing algorithms. The final section concludes the paper.

2. SYSTEM MODEL

As shown in Figure 1, the geographical area is typically divided into hexagonal cells in a mobile cellular network. Each cell is served by a *base station* (BS) and the mobile users communicate through wireless links using radio channels. A number of cells (or BSs) are linked to a *mobile switching center* (MSC) through dedicated wire-line links. Each MSC is linked to the fixed telephone network (e.g., PSTN and ISDN) again through a wire-line link and acts as a gateway of the cellular network to the fixed backbone network [2].

In our model, the system has a total of S distinct channels which are initially assigned to the cells in the same way as in the fixed channel assignment scheme. However, no channel belongs to a specific cell permanently. Channels initially assigned to a cell are called the *origin channels* to the cell. Figure 1 shows an example of the initial state of the system where the alphabets *a, b, c, ...* on the cells denote different sets of channels and the set of cells using distinct sets of channels is 7. The cells with the same alphabet are assigned the same set of channels. Any channel can be reassigned to any other cell if necessary provided that the same channel is used farther enough than the *reuse distance*, the minimum distance at which the same channel can be reused without interference. For example, all the cells with the same alphabets in Figure 1 are assigned the same set of channels with the minimum reuse distance. A cell holds the channels assigned to it as its own property and has the right to control them freely; i.e., it can assign, release, or lock any specific channel it owns without negotiation with any other cells. A set of cells in the system forms a *group*, N_i, so that each cell in N_i is located within the minimum reuse distance related to the center of the group, say cell i, as shown in Figure 2. Let the set of cells in N_i that all own channel c be denoted by $P_i(c)$.

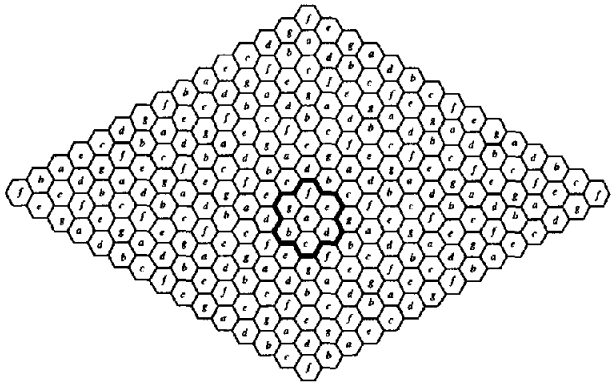

Figure 1: Cellular System

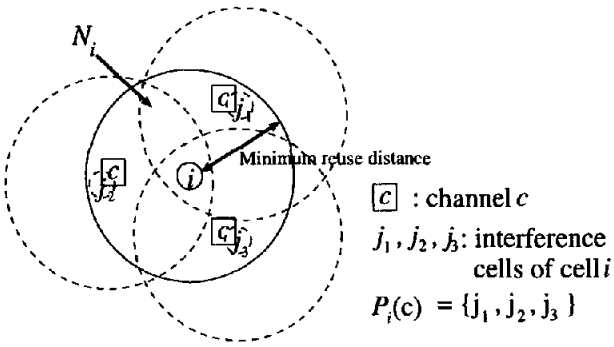

Figure 2: Interference cells

A newly incoming call or a handoff call from an adjacent cell will be assigned a free channel immediately if there are any available. When an active call traverses the boundary of two cells an *inter-cell handoff* occurs and the call releases the serving channel in the original cell and is reassigned a new channel at the adjacent destination cell. In order to improve the channel utilization, a cell may also enforce an active call to release its serving channel and reassign it with a new one in the same cell. This process is called an *intra-cell handoff*.

The proposed D-CAT algorithm employs two thresholds: a *heavy* and a *target* thresholds denoted by T^h and T_i^t, respectively. Cell i is called *overloaded* or *heavy*, if the number of channels, v_i, available at cell i is less than or equal to T^h. A cell intends to import free channels if it becomes heavy but will do its best to respond to channel requests from other cells if it has plenty of free channels. It is assumed that cells in the system are cooperative with each other in the sense that they will react kindly to channel requests from other cells and do their best to satisfy the channel requests.

The heavy and target thresholds are key parameters in our channel allocation algorithm. The D-CAT algorithm is triggered by cell i in the event of a new call arrival when $v_i \leq T^h$. Typical values of T^h are 0 and 1, and determined according to the necessity of the channel allocation policy. If $T^h = 1$, a cell can still assign a channel to a newly arriving call while attempting to import free channels. A value of $T^h > 1$ seems

useless and expensive since there are seldom chances to have two call arrivals simultaneously, and a large T^h causes the algorithm to be run frequently. On the other hand, when $T^h = 0$, a cell attempts to import free channels only if it has no free channels anymore. In this case the new arrival call has to wait for the result of the channel importation and it will be blocked if the channel importing fails. The target threshold, T_i^t, indicates the target number of free channels that cell i intends to acquire. The value of T_i^t is determined by the average number of channels available at a cell in the group N_i as follows.

$$T_i^t = \left\lfloor \frac{\sum_{j \in N_i, j \neq i} v_j}{|N_i|} + 0.5 \right\rfloor, \quad (1)$$

where v_j is the number of channels available at cell j. If $T_i^t = 0$ then let $T_i^t = 1$. Note that each cell in the system has the same T^h value but a distinct T_i^t value.

A heavy cell attempting to obtain free channels is called a *channel importer* or *importer*. On the other hand, a cell that can provide free channels is called a *channel exporter* or *exporter*. A cell can reserve few channels (typically 1 channel) for the next newly arriving calls and assign them to the new arrival calls while processing channel requests from channel importers. When a heavy cell (importer) attempts to import any reserved channel, it must confirm its availability. Before approving a request for a reserved channel from an importer, an exporter can assign the reserved channel to a newly arriving call at the exporter.

The following notations are used in this paper.

S: set of channels used in the system

C_i: set of cells as candidates for channel importing at cell i

V_i: set of channels available at cell i

v_i: number of channels available at cell i, $v_i = |V_i|$

U_i: set of channels used in cell i

u_i: number of channels used in cell i, $u_i = |U_i|$

N_i: set of the neighboring cells within the interference distance of cell i

R_{ij}: set of channels imported by cell i from cell j

R'_{ij}: set of confirmed channels exported from cell j to cell i

T^h: heavy threshold

T_i^t: target threshold at cell i

r_i: number of channels needing to import for cell i, $r_i = T_i^t - v_i$

$P_i(c)$: set of cells that own channel c in N_i

3. THE D-CAT ALGORITHM

The D-CAT algorithm is distributed and dynamic in the sense that each cell runs the channel allocation algorithm independently and depending only on its own state. Each cell in D-CAT maintains a group of channels and treats the channels it holds as its own property. Furthermore, if a cell obtains any free channels from its interfering neighbors, it also keeps them as its property. It is therefore not a *borrower-lender* relation between a channel requester and supplier but instead an *importer-exporter* relation.

When a new call arrives at a heavy cell, the D-CAT algorithm is activated requesting its neighboring interference cells for help, and attempts to import sufficient free channels to satisfy its demand. The messages transmitted between cell i (channel importer) and cell j (possible channel exporter) are classified into four categories as follows.

- *request* message, *request*(i): Message sent by importer i to all the neighboring cells in N_i to request free channels.
- *reply* message, *reply*(j, V_j, U_j): Message from cell $j \in N_i$ responding to importer i. A reply message contains the identifier of cell j, and the sets of channels available and used at cell j. The message also includes the information on the reserved channels in cell j.
- *inform* message, $inform(i, R_{ij})$: Message sent by importer i to the exporters and the other cells in N_i to inform them about its channel acquisition decision and the end of the channel acquisition operation. The requests of the reserved channels are also included in the *inform* message to the exporters.
- *confirm* message, $confirm(j, R'_{ij})$: Message sent by exporter j to importer i to inform it the availability of the requested channels that have been reserved at exporter j. Exporter j can still assign the reserved channels to new arrival calls before sending the confirm message back to importer i.

Each message contains a *timestamp* which equals the time at which the message was sent. A method proposed by Lamport [15] is used in order to synchronize the time of messages transmitted among the cells. The messages can therefore be totally ordered using this method.

The D-CAT algorithm consists of three components – *channel import component*, *channel export component*, and *channel selection component*. The first component is activated by a heavy cell needing free channels and works as the client in the channel acquisition process. The second component, on the other hand, is always active at each cell and ready to perform as the server to receive channel requests from clients. The last component is used for selecting appropriate channels to import and for assigning and reassigning channels in a cell. In the following sub-sections, the three components of D-CAT algorithm and its deadlock freedom are described.

3.1 Channel Import Component

When cell i becomes heavy, the event of a new call arrival makes cell i enter into the "ask-for-help" mode and asks its interference neighbors in N_i for acquiring free channels. The channel acquisition algorithm can be described as the following seven steps.

1. When cell i becomes heavy, i.e., $v_i \leq T^h$, it sends a *request* message, $request(i)$, for free channels to all of its interference neighbors in N_i. When cell i receives a request from another cell with a larger timestamp, it postpones the response. Otherwise, it replies the request immediately.

2. After cell i has received all the *reply* messages from its neighbors, it calculates T_i^t and r_i.

3. Seek for the unused channels in N_i, i.e., $S - \cup_{j \in N_i}(V_j \cup U_j) - (V_i \cup U_i)$. Obtain the set of free channels, R_0, so that $|R_0| = \min\{r_i, |S - \cup_{j \in N_i}(V_j \cup U_j) - (V_i \cup U_i)|\}$. Stop the algorithm if the request is satisfied. Otherwise, go to the next step.

4. Treat the reserved channels in the neighboring cells as the used ones and search for free channels according to the following sub-steps.

   (a) Search for cells $j \in N_i$ such that $v_j > T_i^t$, and denote the set of these cells by C_i.

   (b) Select a channel c that belongs to cell $j \in C_i$ and $|P_i(c)| = 1$. Add channel c to the import channel set, R_{ij}. Delete cell j from C_i if the condition of $v_j > T_i^t$ is violated. Repeat this process until the request is satisfied, there are no more appropriate channels, or C_i becomes empty.

   (c) Select a channel c that belongs to cell $j \in P_i(c)$ and $P_i(c) \subset C_i$. That is, find a channel c belonging to cells j that satisfies the condition of $v_j > T_i^t$. Add channel c to R_{ij}. Delete cell j from C_i if the condition of $v_j > T_i^t$ is violated. Repeat this process until the request is satisfied, there are no more appropriate channels, or C_i becomes empty.

   (d) Search for a channel c such that c belongs to cell $j \in P_i(c)$ and the following inequality is satisfied.

$$\max\{\min_j(v_j, j \in P_i(c))\} - 1 \geq v_i + \cup_{j \in N_i}|R_{ij}| + |R_0| + 1.$$

   Add channel c to R_{ij}. Repeat this process until the request is satisfied, or there are no more appropriate channels.

5. If cell i still needs more free channels, treat the reserved channels in the neighboring cells as free channels this time and repeat steps 4(a)–4(d).

6. Mark the channels in R_{ij} that are reserved at cell j and send an *inform* message, $inform(i, R_{ij})$, to each channel exporter j to inform which free channels are imported and which reserved channels are requested. Wait for the confirmation of the availability for the reserved channels in R_{ij} but the other free channels are ready for immediate use. Also send an *inform* message, $inform(i, \emptyset)$, to each of the other cells in N_i to inform them about the end of the channel import operation.

7. After receiving the *confirm* message, $confirm(j, R'_{ij})$, from exporter j, make the confirmed channels available and discard the other ones.

3.2 Channel Export Component

Channel requests arrived at cell j are queued in a request queue based on the timestamps of the requests and then processed sequentially. When cell j receives a request from cell i, it processes the request according to the following four steps.

1. If there are no requests under processing, go to step 2. Otherwise, compare the timestamps of the requests. If the newly incoming request from cell i has a smaller timestamp than the request from cell k under processing, then the request from cell k will be aborted and put into the request queue and then go to step 2.

2. Reply cell i with a *reply* message, $reply(j, V_j, U_j)$, and wait until the *inform* message arrives if cell j is not heavy. If cell j is heavy it has no need to wait but can continue its own processing.

3. When cell j receives an *inform* message, $inform(i, R_{ij})$, it locks the requested channels. If any channels in R_{ij} are the reserved ones but still available, cell j will lock these channels and add them to R'_{ij}. It then selects new reserved channels if cell j still has free channels. Cell j can however assign a reserved channel to a new call arrived locally before sending the confirm message to cell i.

4. Send cell i a *confirm* message, $confirm(j, R'_{ij})$, to inform the availability of the reserved channels.

3.3 Channel Selection Component

The irregular assignment of channels in a channel allocation algorithm may cause two problems. Firstly, the channel reuse pattern with such an assignment scheme may not be optimized, since the irregular assignment of channels to a cell may make channels be reused with a distance longer than the minimum reuse distance in order to avoid interference. This degrades the channel utilization of the system. Secondly, the identifiers of channels, i.e., channel bands, owned by a cell may become disjointed. For example, a cell with the origin channels of 1, 2, 3, and 4 may has disjointed channels like 1, 2, 50, and 100 as time passes. A cell therefore has to take a long switching time to use disjointed channels. The *channel selection component* in D-CAT employs a prioritized channel selection scheme to solve these problems.

The channel selection component consists of two sub-components: one used for importing free channels from exporters and the other used for assigning and reassigning channels in a cell. When a heavy cell finds multiple channel candidates in any stage of the channel import process, it prioritizes the channel candidates depending on their identifiers. A channel with an identifier which is the same as

or nearer to any identifiers of the channels initially assigned to the importer has a higher priority. The best candidate will be chosen from these candidates. For example, let cell i be initially assigned with three channels of 5, 6, and 7, and now have only one channel 5. If cell i needs to import free channels and has found four channel candidates, 1, 4, 6, and 9, then it attempts to import these channels with a priority of 6, 4, 9, and 1.

Channel assignment and reassignment in a cell are performed according to the channel origins. Channels except the original channels at a cell are those imported from its interfering neighbors. When a new call arrives at a cell, an original channel is assigned to the call with the highest priority. For two original channels, the one with the smaller identifier has a higher priority. The imported channels are prioritized according to how far their identifiers from those of the original channels. The nearer the identifier of an imported channel to that of any original channel the higher the priority of the imported channel. For two channels with the same priority, the one with the smaller identifier has a higher priority. Furthermore, if an original channel is released while an imported channel is serving an active call, an intra-cell handoff is performed so that the original channel is reassigned to the active call. A free imported channel with a higher priority will also cause an intra-cell handoff if another imported channel with a lower priority is serving an active call.

3.4 Deadlock Freedom of D-CAT algorithm

The deadlock freedom of D-CAT algorithm can be proved similarly to that in [6]. Since the cells can send messages autonomously and concurrently, the synchronization problem of time for channel requests is the same as that in a distributed system. It is assumed that the communication link is reliable and each message contains a timestamp showing when it was sent. Furthermore, it is assumed that the messages sent by a cell arrive at a cell in the order in which they are sent. According to Lamport [15], the channel *request* messages originating from different cells can be totally ordered by their timestamps. As described in Section 3.1, a cell in "ask-for-help" mode sends *reply* messages only to the channel importers with lower timestamps but postpones the others. Since the time ordering of channel requests are known by all the cells, there is no loop for delaying *reply* messages among the cells. Therefore, the cell whose request has the lowest timestamp can always receive all of *reply* messages from its neighbors. After determining the imported channels, it sends an *inform* message to all of its neighbors. A channel exporter receiving an *inform* message can decide immediately whether it approves the requested channels or rejects some of them, and send a *confirm* message back to the channel importer. The channel importer determines how many channels it can import successfully and then processes the postponed replies. Since the operation at each step in the process of acquiring free channels is time-limited, no channel request will wait forever.

4. PERFORMANCE EVALUATION

The performance of D-CAT is evaluated with respect to two metrics: the *implementation cost* and the *call blocking probability*. The number of messages transmitted between the BSs and the delay of message transmission are taken into account in the implementation cost. Two distributed algorithms are chosen to compare with D-CAT. These are D-LBSB proposed by Das *et al.* [8] and an enhanced *search* algorithm due to Cao and Singhal [3], referred to as D-ES in this paper. We also compare D-CAT with one centralized algorithm proposed by Zhang and Das [21], referred to as CAT in this paper.

4.1 Implementation Cost Comparison

Let the total number of cells in the system is denoted by N. The message delay between the BSs and between a BS and the MSC is fixed to be δ. The postponed response delay experienced at a channel exporter, which is denoted by T_d in [3, 6], in D-ES and D-CAT is denoted by δ_d. Cao and Singhal [3, 6] showed that δ_d has a negligible effect on the message delay. In D-ES, n_p denotes the number of interference primary neighbors of a cell, m denotes the conflict rate, and n_u denotes the update message. Since the execution time of a channel allocation algorithm is much smaller than the message delay, it is not taken into account in the comparison. In order not to bring any bias to the comparison results for the D-LBSB and CAT algorithms, the case of locking only three co-channels for each lender cell is taken into account in these two algorithms. It is also assumed that a heavy cell needs X channels and each channel exporter can offer only one channel. Since in D-ES a heavy cell borrows only one channel during each channel acquisition operation, it needs to run the algorithm X times to obtain X channels.

In D-CAT, the messages transmitted between the BSs and the corresponding message delay can be listed as follows.

1. Channel importer i sends a *request* message to each cell j in N_i. Therefore, the total number of request messages is $|N_i|$ and the delay of message transmission is δ.

2. After receiving the request from importer i, cell j sends a *reply* message back to importer i. Thus, the total number of reply messages is also $|N_i|$ and the delay of message transmission is δ.

3. Importer i sends an *inform* message to each cell in N_i to inform its channel importation decision and notice the end of the channel importation operation. The message to each channel exporter also includes a confirmation request for the reserved channels. Therefore, the total number of messages is $|N_i|$ and the delay of message transmission is δ.

4. Exporter j sends a *confirm* message to importer i to inform it the availability of the reserved channels. Assuming that there are x channels needing to confirm and each channel belongs to a distinct cell, the number of confirm messages is $x (x \leq X)$ and the delay of message transmission is δ.

The total number of messages transmitted between the cells is therefore $3|N_i| + x$ for the D-CAT algorithm. The delay for message transmission for channel acquisition is $2\delta + \delta_d$ if at least one imported channel has no need to be confirmed, i.e., $\theta = 0$, and $4\delta + \delta_d$ only if all of the imported channels

Table 1: Implementation cost comparison of the algorithms.

Scheme	Number of messages	Message delay
CAT	$N + 4X + 1$	2δ
D-LBSB	$2(N-1) + 2(\lvert N_i \rvert + 3)X$	$4\delta(1 + 3X)$
D-ES	$(3\lvert N_i \rvert + 3n_p m + n_u)X$	$(2(1+m)\delta + \delta_d)X$
D-CAT	$3\lvert N_i \rvert + x$	$2(1+\theta)\delta + \delta_d$

need to be confirmed, i.e., $\theta = 1$. Table 1 shows the total number of messages transmitted between the cells and the delay of message transmission for importing X channels for CAT, D-LBSB, D-ES, and D-CAT.

It is observed that D-CAT yields the least message complexity compared with all of other algorithms. The fact that D-CAT shows the best among the three distributed algorithms can be explained as follows. Since the channels in D-CAT are held by each cell, there is no need to perform the channel returning operation as in D-ES and D-LBSB. Furthermore, a channel importer in D-CAT confirms the reserved channels only once since it has the information of all the reserved channels at each exporter. D-ES, on the other hand, has to check every reserved channel it attempts to import separately and in the worst case it needs to confirm multiple times for importing a single channel. Furthermore, since D-CAT can import multiple channels within one channel allocation operation and the reserved channels are treated with low priorities, the message delay in most cases in D-CAT will be reduced to $2\delta + \delta_d$.

4.2 Simulation Experiments

We conducted simulation studies to evaluate D-CAT and compare it with D-LBSB, D-ES, and CAT in terms of call blocking probability with uniform and non-uniform traffic. The results shown in Figures 3–5 were obtained with 90% confidence interval and within 5% of the sample mean. The simulated cellular system contains $N = 15 \times 15$ hexagonal cells as shown in Figure 1. The letters, $a, b, c, ...$, on the cells in Figure 1 denote distinct sets of channels and the cells with the same letter are assigned with the same set of channels. Each cell is initially assigned $S_c = 40$ channels. Incoming call arrival at each cell is assumed to follow a Poisson process with a mean λ. The holding time of a call is assumed to be distributed based on an exponential distribution with a mean $1/\mu$ of 180 secs (3 mins). The parameters used in CAT are as follows: $T^h = 0$, $\Delta = 2$, $p_s = 20\%$ and $c_{min} = 0.05S_c$. The degree of coldness at a cell, h, in D-LBSB is 0.1. The parameters used in the simulation for CAT and D-LBSB are chosen from the best combination to yield low call blocking probability. The number of reserved channels at each cell in both D-ES and D-CAT is 1. The target threshold, T_i^t, in D-CAT is determined by using Equation 1. The heavy threshold, T^h, was examined in the simulation study for the cases of $T^h = 0, 1, 2$ but the results shown in Figures 3–5 used the values of $T^h = 0, 1$, which produce the best performance. Two kinds of call demands, *uniform* and *non-uniform*, were simulated for the algorithms under consideration. In the previous case, call arrival at each cell is identical, whereas in the latter case, a cell can get congested

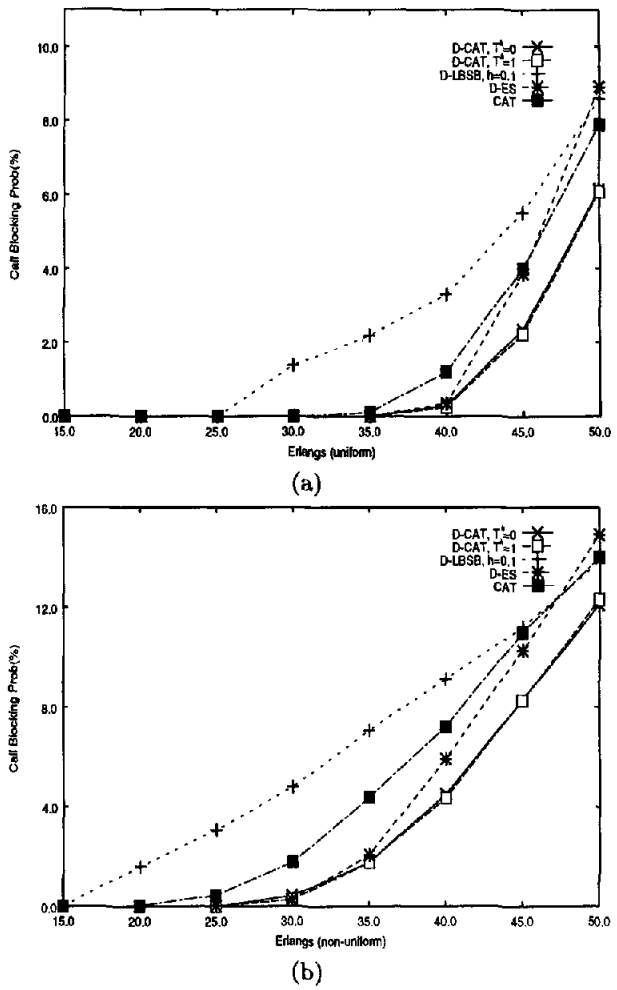

Figure 3: Comparison of call blocking probability.

from time to time. That is, a cell gets congested from λ to 3λ with a probability of 0.001 and a congested cell will return to the normal state with a probability of 0.01.

Figures 3(a) and 3(b) show the *call blocking probability* of all the algorithms under consideration. It is observed that D-CAT outperforms others in terms of the call blocking probability. The reasons for this can be explained as follows. In D-LBSB and CAT, the co-channel locking scheme is over conservative where some channel locking is not necessary in order to avoid co-channel interference, and the channel lender selection scheme is over pessimistic where a heavy cell is not allowed to borrow any channels from moderate cells. Even though D-CAT and D-ES behave similarly when the call demand is below 40 Erlangs (i.e., the cell capacity), D-ES degrades faster than D-CAT. This is because that D-CAT employs a channel selection scheme that always imports the best channels from the channel candidates and assigns the best channel to the incoming calls. Channels in D-CAT are therefore utilized more efficiently, especially at a high channel demand. It is also observed that the value of the heavy threshold, T^h, has no significant impact on the call blocking probability.

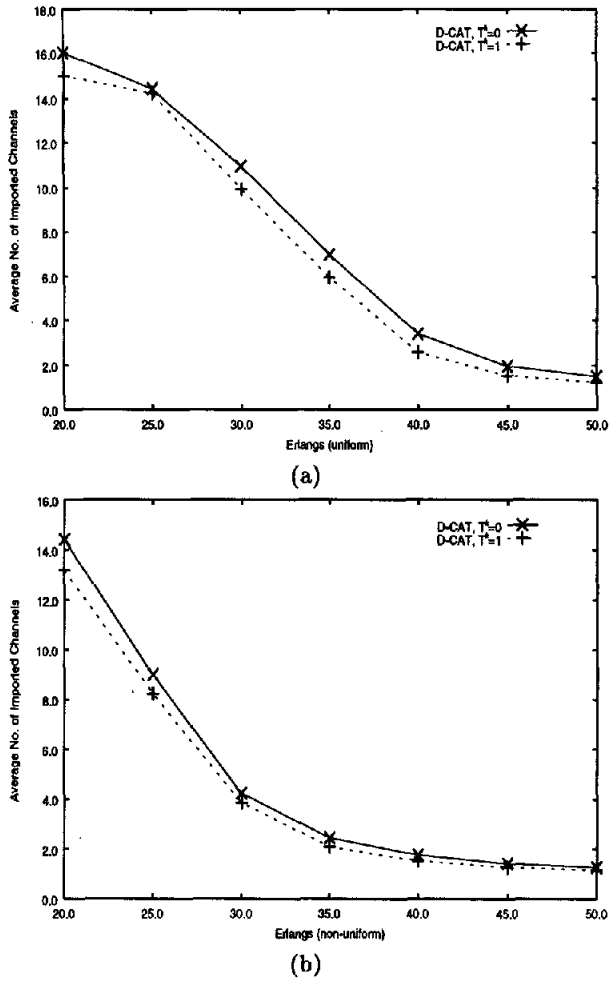

Figure 4: Average number of channels imported during one channel acquisition process.

Figures 4(a) and 4(b) show the average number of channels successfully imported during one channel acquisition operation in D-CAT. The more the number of channels imported in one channel acquisition operation, the lower the overhead cost needed for transmitting the messages between the cells and for running the channel allocation algorithm. It is observed that D-CAT can find out multiple free channels for a heavy cell each time under most practical operation conditions, e.g., on an average more than 10 channels in a call demand of 30 Erlangs and more than 3 channels even in a call demand near to 40 Erlangs as shown in Figure 4(a). This fact confirms the efficiency of the threshold scheme for channel allocation. A heavy cell in D-ES, on the other hand, attempts to borrow only one channel during each channel allocation operation.

Figures 5(a) and 5(b) show the *channel import request ratio*, defined by the ratio of the number of channel import requests to the total number of call arrivals, in D-CAT and D-ES. It is observed from Figures 5(a) and 5(b) that T^h has little effect on the channel import request ratio in D-CAT when the call demand is low, but the effect becomes greater afterwards. This result indicates that if the calling traffic is not very high, i.e., lower than the cell capacity, $T^h = 1$ is preferable. On the other hand, when the calling traffic becomes near to or greater than the cell capacity, letting $T^h = 0$ reduces the channel import request ratio, resulting in a much lower overhead for running the algorithm. It is observed that in D-CAT when $T^h = 0$ the channel import request ratio is lower than that in D-ES for both uniform and non-uniform calling traffic. For example, the channel import request ratio in D-CAT is lower than that in D-ES by over 30% when the call demand is near to or greater than 40 Erlangs. This means that a cell in D-ES frequently needs free channels and has to run the channel acquisition algorithm more often than D-CAT.

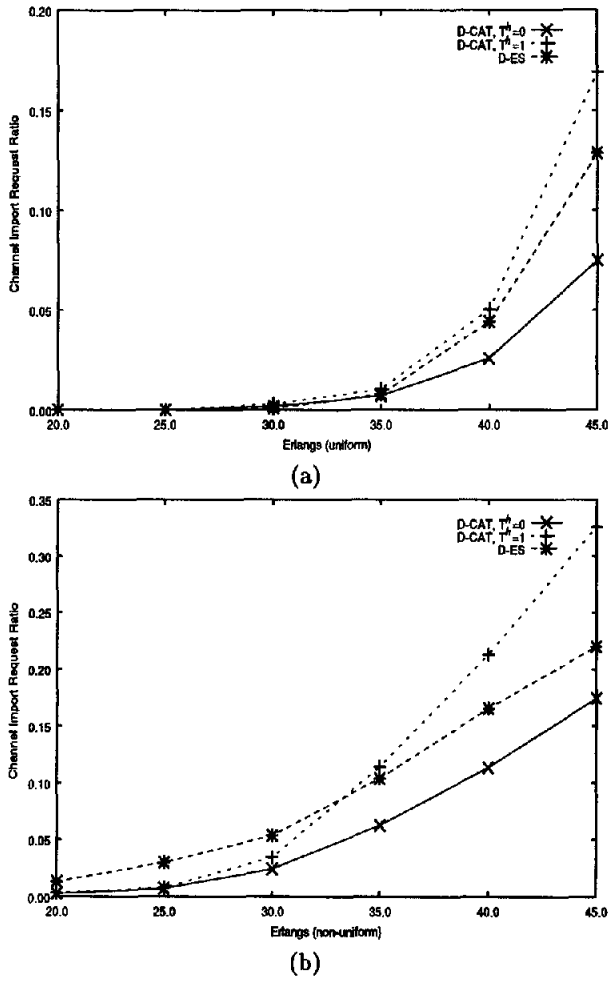

Figure 5: Channel import request ratio.

5. CONCLUSIONS

In this paper, a distributed dynamic channel allocation algorithm called D-CAT based on a two-threshold scheme has been proposed for mobile cellular networks. It has been shown that D-CAT outperforms other centralized and distributed algorithms in terms of the call blocking probability. The D-CAT algorithm also yields a lower message complexity and a shorter message transmission delay than the existing distributed algorithms. It has been observed that a

heavy cell in D-CAT can import multiple channels, in comparison with only one channel in D-ES, during each channel acquisition operation; e.g., a heavy cell can import more than 3 channels on an average in D-CAT when the call demand is near to the cell capacity. Furthermore, the channel import request ratio in D-CAT is lower than that in D-ES by over 30% when the call demand is near to or greater than the cell capacity.

6. REFERENCES

[1] A. Baiocchi, F.D. Priscoli, F. Grilli, and F. Sestini. The geometric dynamic channel allocation as a practical strategy in mobile networks with bursty user mobility. *IEEE Trans. Vehi. Tech.*, 44(1):14–23, February 1995.

[2] U. Black. *Mobile and Wireless Networks.* Prentice-Hall PTR, 1996.

[3] G. Cao and M. Singhal. Efficient distributed channel allocation for mobile cellular networks. In *Proc. IEEE 7th Int. Conf. Comput. and Commun. Networks*, pages 50–57, October 1998.

[4] G. Cao and M. Singhal. An adaptive distributed channel allocation strategy for mobile cellular networks. In *Proc. 18th IEEE Int. Conf. Performance, Computing, and Commun.*, pages 36–42, February 1999.

[5] G. Cao and M. Singhal. Distributed fault-tolerant channel allocation for mobile cellular networks. In *Proc. INFOCOM'99*, pages 584–591, March 1999.

[6] G. Cao and M. Singhal. An adaptive distributed channel allocation strategy for mobile cellular networks. *J. Parallel and Dist. Comput.*, 60(4):451–473, April 2000.

[7] D.C. Cox and D.O. Reudink. Increasing channel occupancy in large scale mobile radio systems: Dynamic channel reassignment. *IEEE Trans. Vehi. Tech.*, VT-22(4):218–222, November 1973.

[8] S.K. Das, S.K. Sen, R. Jayaram, and P. Agrawal. An efficient distributed channel management algorithm for cellular mobile networks. In *Proc. IEEE Int. Conf. Universal Personal Commun.*, pages 646–650, October 1997.

[9] S.K. Das, S.K.Sen, and R. Jayaram. A dynamic load balancing strategy for channel assignment using selective borrowing in cellular mobile environment. *Wireless Networks*, 3(5):333–348, October 1997.

[10] S.K. Das, S.K.Sen, and R. Jayaram. A noval load balancing scheme for the tele-traffic hot spot problem in cellular networks. *Wireless Networks*, 4(4):325–340, June 1998.

[11] X. Dong and T.H. Lai. Distributed dynamic carrier allocation in mobile cellular networks: Search vs. Update. In *Proc. IEEE 17th Int. Conf. Dist. Comput. Syst.*, pages 108–115, 1997.

[12] N. Garg, M. Papatriantafilou, and P. Tsigas. Distributed list coloring: How to dynamically allocate frequencies to mobile base stations. In *Proc. 8th Annual IEEE Symp. Parallel and Dist. Process.*, pages 18–25, October 1996.

[13] Ed. J.D. Gibson. *Mobile Communications Handbook.* CRC Press, 1999.

[14] H. Jiang and S.S. Rappaport. CBWL: A new channel assignment and sharing method for cellular communication systems. *IEEE Trans. Vehi. Tech.*, 43(2):313–322, May 1994.

[15] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. of the ACM*, 21(7):558–565, July 1978.

[16] R. Prakash, N. Shivaratri, and M. Singhal. Distributed dynamic channel allocation for mobile computing. In *Proc. 14th ACM Symp. Principles of Distributed Computing*, pages 47–56, 1995.

[17] G.L. Stuber. *Principles of Mobile Communication.* Kluwer Academic Publisher, Inc., Boston, 1996.

[18] J. Tajima and K. Imamura. A strategy for flexible channel assignment in mobile communication systems. *IEEE Trans. Vehi. Tech.*, 37(2):92–103, May 1988.

[19] S. Tekinay and B. Jabbari. Handover and channel assignment in mobile cellular networks. *IEEE Commun. Mag.*, pages 42–46, November 1991.

[20] M. Zhang and T.S.P. Yum. Comparisons of channel-assignment strategies in cellular mobile telephone systems. *IEEE Trans. Vehi. Tech.*, 38(4):211–215, November 1989.

[21] Y. Zhang and S.K. Das. An efficient load-balancing algorithm based on a two-threshold cell selection scheme in mobile cellular networks. *Comput. Commun.*, 23(5-6):452–461, March 2000.