DonorList: A New Distributed Channel Allocation Scheme for Cellular Networks

Tamer Tulgar and Muhammed Salamah

Eastern Mediterranean University, Department of Computer Engineering,
Famagusta,T.R.N.C
Mersin 10, Turkey
{tamer.tulgar, muhammed.salamah}@emu.edu.tr

Abstract. One of the most important challenges in cellular networks is to utilize the scarce spectrum allocated to the network in the most efficient way. If the channels are statically allocated to the cells, when a large number of mobile hosts move to the cell, that cell may run out of channels resulting in a high call incompletion rate. To overcome this problem, dynamic channel allocation schemes have been proposed. Among these schemes, distributed dynamic channel allocation approaches resulted in good performance results. Nevertheless, distributed allocation schemes must address the problem of efficient co-channel interference avoidance and reducing messaging overhead issues. In this paper, we introduced a new distributed channel allocation scheme namely the DonorList approach, which decreases the amount of messages required per channel allocation while efficiently handling the co-channel interference problem. We also demonstrate the performance results obtained after extensive simulation studies. The results show that the proposed algorithm outperforms the other algorithms recently proposed in the literature.

1 Introduction

In cellular wireless networks a mobile host(MH) can communicate with another MH anytime from anywhere with the help of base stations[1]. The area covered by the cellular network is divided into smaller regions called cells. Each cell is controlled by a base station and a MH communicates with its base station via a wireless link. All base stations in the cellular network can communicate with each other by using a wired network that connects every base station to the mobile switching center(MSC) of the cellular network[2].

A cellular system can use channels either as control channels, which carry control information like call setup data or as communication channels which carry the user data. In this paper, unless specified otherwise, the term *"channel"* will be referring to a *"communication channel"*.

When a call arrives at a cell, the base station should allocate a communication channel to support the incoming call. This process is known as the channel allocation process. If the base station fails to support the call, the call is said to be blocked or dropped. The most basic channel allocation scheme is known as

the fixed channel allocation scheme (FCA), where each cell is preallocated with a fixed number of channels and the number of channels cannot vary depending on the system load [3]. In a FCA system, when a large number of mobile hosts move to the cell, that cell may run out of channels resulting in a high call incompletion rate. Since channels are very scarce resources, a channel allocation algorithm should not only assign a channel to a call but also must care about the channel usage efficiency by trying to increase the channel reuse [4]. For this purpose, dynamic channel allocation (DCA) schemes have been proposed [4],[5].

1.1 Dynamic Channel Allocation Schemes

In DCA schemes, unlike FCA, the number of channels allocated to each cell may vary depending on the needs of the cells. In a DCA scheme, a cell that has used all its nominal channels can borrow free channels from its neighboring cells(donors) to accommodate incoming calls. Additionally, the DCA schemes may be designed to rely on a pre-allocation of channels to the cells, which is also known as the *resource planning* or without any pre-allocation of channels to the cells. The DCA schemes can be classified as centralized dynamic channel allocation (C-DCA) schemes and distributed dynamic channel allocation (D-DCA) schemes.

In C-DCA schemes, only the MSC has access to the channel allocation information of the cells. In this approach, if a cell runs out of channels, the MSC is responsible for allocating new channels to the cell. In C-DCA schemes, the MSC is a single point of failure since it is the only unit which can assign channels to the cells and furthermore C-DCA schemes are not very scalable since the MSC can become a bottleneck under very heavy traffic conditions. To overcome these drawbacks, several D-DCA schemes have been proposed [6],[7],[8],[9],[10],[11],[12].

In a D-DCA scheme, there is no central controller like the MSC but instead every base station shares the responsibility to allocate channels (base stations import/export or borrow/lend channels to/from each other, depending on their *own local channel usage information* of the other cells). In the D-DCA schemes, if a cell needs to borrow/import a channel, it consults its neighbors by sending and receiving messages, and they negotiate together to ensure that no co-channel interference will occur when a channel(s) will be supplied to the cell in need.

In this paper, we propose a new D-DCA scheme based on resource planning. The main drawback of the previously proposed D-DCA algorithms is the high messaging overhead per channel allocation. The proposed algorithm employs a donor list, which is a list of import candidate channels and cells, to decrease the messaging complexity and to further improve the call completion probabilities compared to the D-DCA algorithms currently found in the literature. Also, the proposed algorithm is based on an import/export relation rather than a borrow/lend relation, where a cell gains the full control of the imported channels, and can export them to other cells.

The rest of this paper is organized as follows. In section 2, the system infrastructure is presented. In section 3, the proposed DonorList algorithm is explained

in detail and in section 4 the performance evaluation and the simulation results of the algorithm are presented. Finally, in section 5, we present our conclusions.

2 System Model

The cellular system that is used to realize the *DonorList* algorithm contains 144 hexagonal cells, which are organized in a form of 12x12 grid. In the infrastructure of the employed cellular network, the 144 cells are partitioned into 7 reuse groups such that the cells in the same reuse group are apart from each other by at least a minimum distance defined by D_{min} in equation (1), where N is the cluster size which is the number of cells in a reuse group[2]. Each cell in the system, except the ones situated at the borders, has 6 neighbors.

$$D_{min} = \sqrt{3 \times N} \tag{1}$$

In Fig. 1, it can be seen that, the cells belonging to the same reuse group are labeled with a unique *Group ID* using the letters **{A,B,C,D,E,F,}**, and each cell is also labeled with a unique *Cell ID* using the integers ranging from **{1..144}**. The total channel spectrum belonging to the whole cellular system contains S = 280 channels [12]. Each channel is assigned a unique *Channel ID* ranging from 1 to 280. Initially each cell is assigned 40 channels by using the resource planning scheme explained in the next subsection.

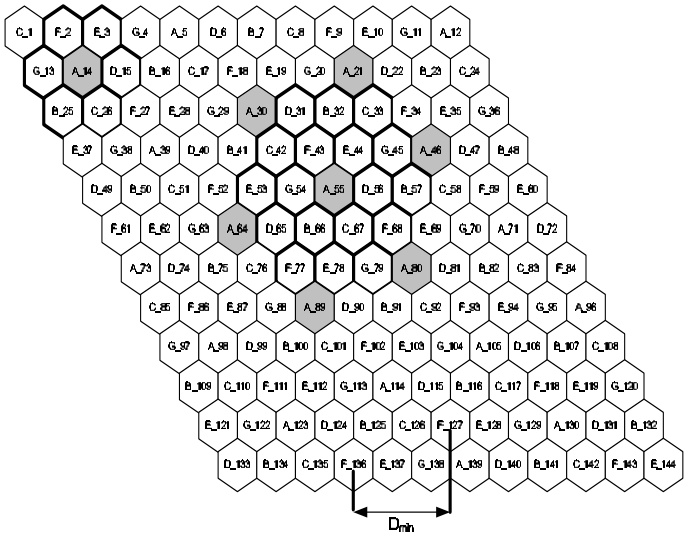

Fig. 1. Cellular System Layout

2.1 Resource planning

Resource planning that will be used by the proposed DonorList algorithm is as follows:

- Partition the whole spectrum of channels(i.e. 280 channels) into 7 disjoint subsets and name them as P1..P7.
- Uniquely assign a channel group to each of the cell groups (A,B,C,D,E,F,G) such that the channels in P1 will only belong to the cells in group-A, channels in P2 will only belong to the cells in group-B and so on.
- Prioritize the channels in each cell in such a way that the smaller *Channel ID* will have a high priority and greater *Channel ID* will have a lower priority.
- The interference neighbors of a cell C_i, denoted as IN_i is defined as set of cells which have a distance smaller than the D_{min} from cell C_i. For example in Fig. 1, the IN_{55} set of the cell C_{55} contains the cells 31, 32, 33, 42, 43, 44, 45, 53, 54, 56, 57, 65, 66, 67, 68, 77, 78, 79.

$$IN_i = \{C_j | distance(C_i, C_j) < D_{min}\} \tag{2}$$

- A cell C_i can import channels only from its interference neighbors, provided that the same channel is not used within the interference distance of C_i.
- A base station assigns high priority channels to the incoming calls (i.e. new and handoff calls) and tries to export the lower priority channels for incoming channel import requests from other cells.

3 The Proposed DonorList Algorithm

In the cellular system described above, the proposed *DonorList* algorithm is executed separately by each cell. Each cell employs a channel usage threshold (C_t) which is used to warn a cell about its remaining number of *available* channels. Let us define the channel usage ratio of a cell C_i (CU_i) as the ratio of the number of busy channels of C_i to the number of the available channels of C_i, which is given in equation (3). When CU_i raises above C_t, C_i queries the cells in its IN_i, and collects information about which channels can be imported and forms a list called *the donor list*.

$$CU_i = \frac{number\_of\_busy\_channels\_of\_C_i}{number\_of\_total\_channels\_of\_C_i} \tag{3}$$

When the cell C_i runs out of available channels, it consults its donor list and asks for channel(s) starting from the cell(s) placed at the top of the list. If those cells can still export the channel to the cell C_i, they send their corresponding confirmations. If all these cells agree to export the channel to C_i, the exporter cells deallocate the exported channel to make sure that no co-channel interference will occur. If a suitable channel cannot be found at the first row of the donor list, the cell C_i moves to the next row in the list and repeats the process. If the cell C_i queries all the cells in the donor list and cannot find a channel to import, it drops the call.

By the addition of the *DonorList* idea, the cells which need to import a channel, send *request* messages only during the donor list formation, and then they only need to send messages to the cells listed in the donor list. In this way, the algorithm tries to reduce the total number of messages required per successful allocation.

The proposed algorithm is composed of five modules which are: The incoming call module, receive acquire message module, receive confirm message module, build donor list module and the intrahandoff module.

3.1 The Incoming Call Module

Fig. 2 below shows the flowchart for processing an incoming call. When a call arrives at C_i, if the cell contains at least one available channel, it allocates the channel to the call immediately. After a channel is allocated to a call, the cell checks if its CU_i ratio is higher than the threshold C_t. If CU_i is higher than the C_t, the cell sends request messages to all the cells in its IN_i and updates its donor list.

If no channels are available, then the cell checks if there is at least one entry in its donor list. If the donor list is not empty, the cell sends an *acquire* message with the format acquire(msgid,tocell,callid,fromcell,requestedchannelid,timestamp) to each cell which currently own the requested channel listed in the donor list entry and removes the entry from the donor list. Also, the cell inserts the call information to a list called the waiting calls list. However, if the donor list is empty, the cell blocks or drops the call.

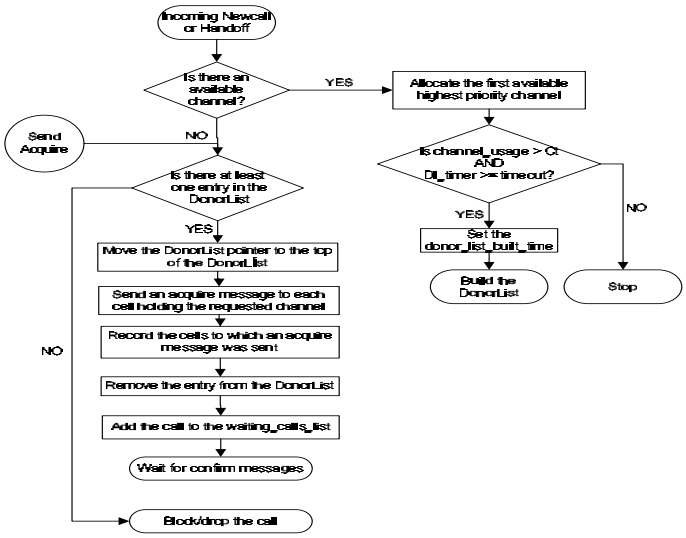

Fig. 2. Incoming Call Module

If the received signal signal strength(RSS) of a call drops below a predefined value RSS_{edge} , the RSS values received from the neighbor cells are calculated and the call is transferred to the control of the base station which provides the maximum RSS. This process is known as the handoff process. When a handoff occurs the handoff call is transferred to the new basestation as an incoming call and the new base station tries to allocate a channel to this incoming call.

3.2 Receive Acquire Message Module

When a cell receives an *acquire* message, it uses the algorithm given in Fig. 3 to process the message. So, when cell C_i receives an *acquire* message, first it checks if it has any waiting calls. If it has, the received *acquire* message is inserted into a queue, named as the acquire queue. If the waiting calls list of C_i is empty and the acquire queue is empty, then the *acquire* message is replied with a *confirm* "ok" message confirming that the requested channel is available or with a *confirm* "not ok" message informing the requesting cell that the requested channel is busy.

If there are queued *acquire* messages, the new *acquire* message is inserted into the acquire queue and all the messages in the queue are replied in the ascending order of their timestamps with corresponding *confirm* "ok" or *confirm* "not ok" messages. In any case, if the cell sends a *confirm* "ok" message, it immediately marks the requested channel as reserved.

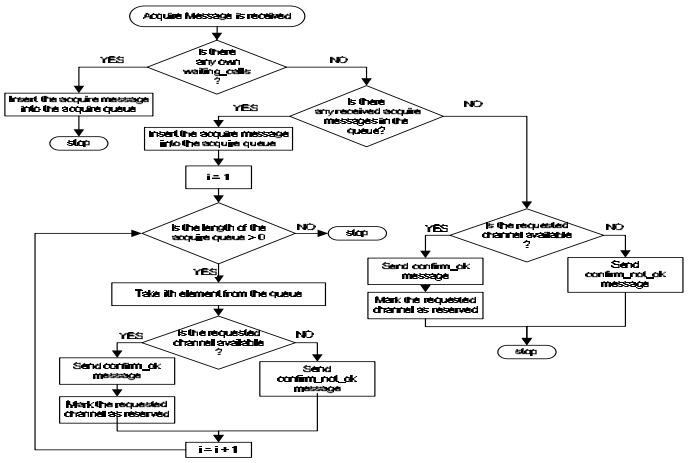

Fig. 3. Receive Acquire Module

3.3 Receive Confirm Message Module

The details of the processing of a received *confirm* message are shown in Fig. 4. If cell C_i receives *confirm* "ok" messages from all the cells which own the requested

channel, it imports the requested channel and sends a *release* message to each owner cell so that the cells which currently own the channel can remove the requested channel from their channel sets. Also, C_i removes the call from its waiting calls list.

If any of the owner cells send a *confirm* "not ok" message, the cell sends *keep* messages to the cell(s) which sent *confirm* "ok" messages, so that the channels they marked can be used again as available channels by their owners. Also, the cell deletes the call from the waiting calls list. Then, the cell runs its send acquire procedure which is shown in figure 2, so that the next entry in the donor list can be processed and new *acquire* messages can be send for another channel import attempt.

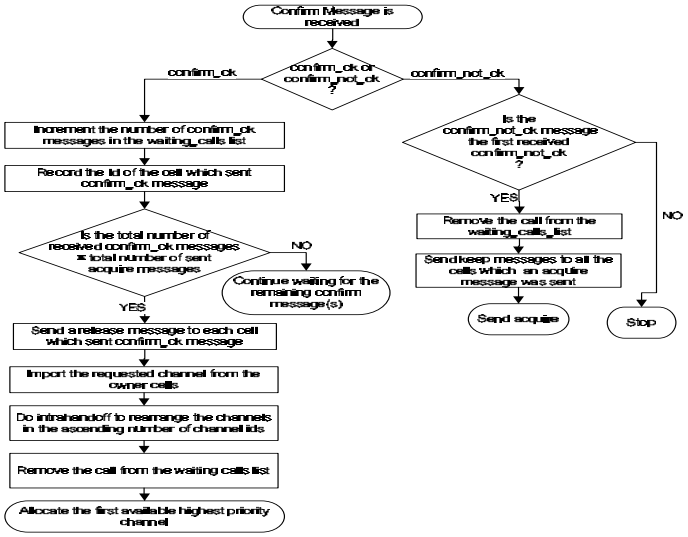

Fig. 4. Receive Confirm Module

3.4 Intrahandoff Module

The intrahandoff module is triggered whenever a channel is deallocated at a cell(i.e. after an outgoing handoff, a terminated call or a successful channel import). This module moves the ongoing calls allocated at the low priority channels to the available high priority channels. In this way, the low priority channels are tried to be left available for possible import requests. This strategy raises the chance of finding at least one donorlist entry and so the successful import ratio of the algorithm.

3.5 Build Donor List Module

Fig. 5 illustrates the algorithm which builds a donor list. When a build donor list event is triggered, as explained in the incoming call module, a cell C_i sends request(from cell, to cell) messages asking for the channel information of all the IN_i cells. On receiving request message, every cell send the set of its available(AC) and busy channels(BC) immediately. After all the reply(from cell, to cell, AC, BC) messages arrive at C_i, all the available channels are combined into a single AC set and all the busy channels are combined into a single BC set. Then, the candidate channels set are calculated by the set difference of AC and BC sets. The second set difference of the candidate channels set and the channels owned by C_i gives the real *candidates* set.

The calculated candidate channels set is then divided into subsets according to the common cells which own each channel. Each donor list entry is formed by selecting the channel with the maximum *Channel ID* and the cells which own the selected channel for each subset (i.e. an entry is formed for each subset). These entries is then inserted into the donor list in the order of descending number of channels in each subset. After insertion, the entries with the same number of channels are resorted in the order of ascending number of cells.

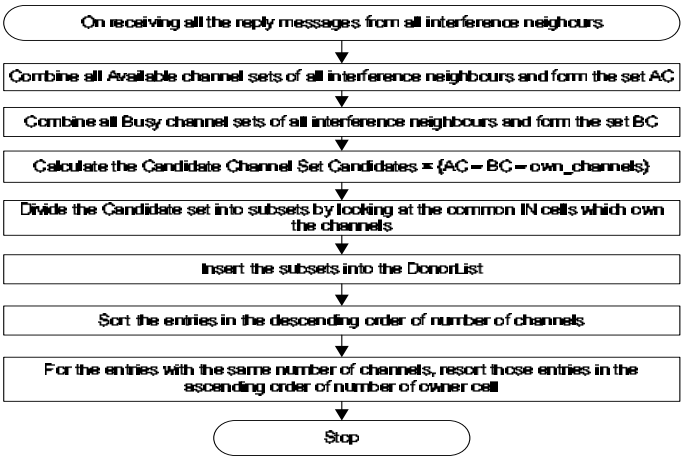

Fig. 5. Build Donor List Module

3.6 Deadlock Freedom of the Proposed DonorList Algorithm

In the proposed DonorList algorithm each message is timestamped using Lamport timestamps [13]. Also, it is assumed that the wired network connecting the basestations and the MSC is reliable and no messages will be lost and also the messages will be received at the cells in the order that they were sent. Based on

these assumptions, request messages coming from different cells can be totally ordered by their timestamps [12].

Since the timestamps of the messages are known to the cells the message with the smallest timestamp(highest priority) will always receive the replies it is waiting for. Also since there is a timer determining how long a cell will wait for replies, there is no infinite waiting.

In a D-DCA algorithm, the channels act as the critical shared resource in the sense that, two or more cells, which are apart from each other closer than the Dmin, should not access the same channel concurrently. Since the DonorList Algorithm is ensuring no co-channel interference, the shared resource is not accessed concurrently. Therefore, with the features explained above, the DonorList algorithm is deadlock free.

4 Performance Evaluation

The performance of the DonorList algorithm is evaluated by extensive simulation studies with different C_t values and under various loads(see Table 1). The simulation program is written in Matlab v.6.5 R13[14] and implements the complete DonorList algorithm.

To evaluate the performance of the algorithm under realistic conditions, non-uniform traffic was applied. The non-uniform traffic was realized with two cell states; the normal state and the hot state[12]. The λ values for the given Erlang Loads are calculated by using the state diagram shown in Fig. 6 and equation (4). Mean cell-state change times are given in Table 1. Also, since the messages are transmitted through the wired network between the base stations, it is assumed that the message loss is negligible[2].

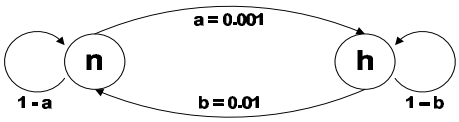

Fig. 6. Cell State Change State Diagram

$$Erlang = b/(a+b) * \lambda * T + a/(a+b) * 3\lambda * T \tag{4}$$

When in the normal state, a cell receives new calls with the exponentially distributed arrival rates λ and this arrival rate triples to 3λ when the cell enters the hot state [9],[12]. To eliminate the border effect, results were collected from the inner 121 cells to make sure that the cells that will provide the statistics have exactly 6 neighbors.

Table 1. Simulation Parameters

Parameter	Value
Arrival rate in normal state	λ
Arrival rate in hot state	3λ
Mean call duration*(T)*	180 secs.
Probability of cell state change from normal to hot	0.001
Probability of cell state change from hot to normal	0.01
Ct	87%,95%
Erlang Loads	20,25,30,35,40,45,50
Time required to transmt and process a meassage	2 msecs[8],[9]

Table 2. Message Complexities

Algorithm	No of msgs. per allocation	Overall No. of. msgs.
D-CAT	3N+x	3N+x
DonorList	3dk	2N+3dk

4.1 Message Complexity of the Proposed Algorithm

Let N be the number of cells in IN_i of any cell C_i. When the cell C_i needs to form its donor list, it sends N number of request and receives N number of reply messages.

On trying to import channels from the cells in its donor list, it sends k *acquire* messages to the cells holding the channel and receives k *confirm* messages, where k is the number of cells holding the channel. If the *confirm* messages are all with "ok" the cell C_i sends k *release* messages to the cells which hold the channel, otherwise it sends k *keep* messages.

If the cell C_i cannot allocate a channel in the first hit, it repeats the above process d-1 times, where d is the number of accesses to the donor list, until it finds a channel to export.

So as a total of $2N+3dk$ messages are exchanged per channel export process. Table 2 shows the comparison between the message complexities of the D-CAT[12] and the proposed algorithm.

4.2 Results

In this section the simulation results, which are illustrated in Figure 7, will be discussed. For most of the results, the 95% confidence level for the measured data is less than 5% of the sample mean.

The performance results will be studied in terms of call incompletion probability, the channel utilization, mean number of messages per channel allocation and mean channel allocation delay under different traffic loads and various channel threshold values. Also, the call incompletion probability results will be compared with another threshold based distributed channel allocation algorithm,

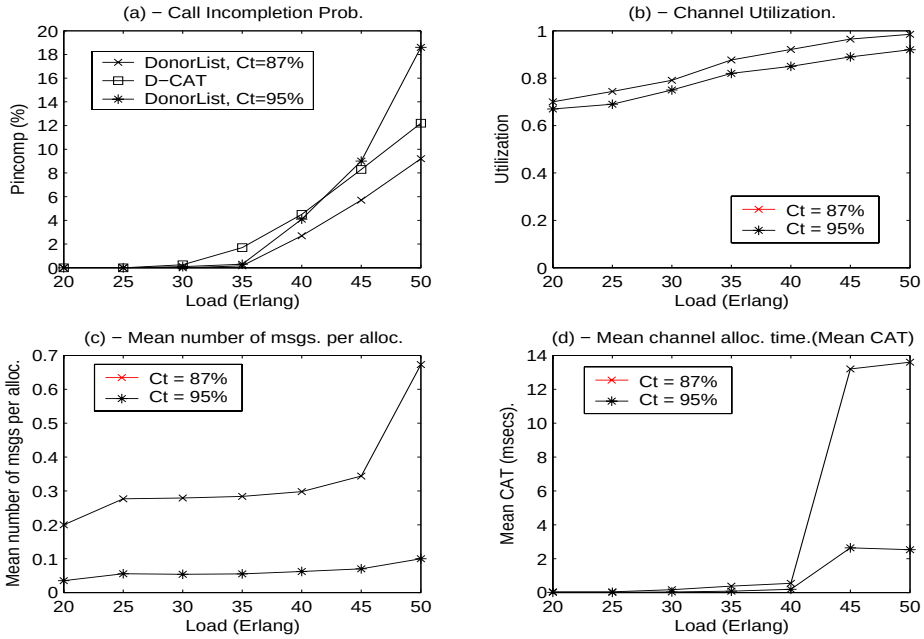

Fig. 7. Performance results

named as D-CAT[12], which proved to have a better call completion performance than [15],[16],[17].

The Figure 7a shows the call incompletion probability(P_{incomp}) results and their comparison with D-CAT. As illustrated in the Figure, as load increases P_{incomp} increases as expected. Under low load (Erlang 20 and 25), both the proposed DonorList algorithm and the D-CAT algorithm produces zero P_{incomp} values. For loads greater than Erlang 35 (i.e. at heavy load, which is the condition under which an algorithm demonstrates its performance strengths and weaknesses) and for $C_t = 87$, the proposed DonorList algorithm outperforms D-CAT by maintaining a P_{incomp} that is 30% lower on the avarage, than that of D-CAT algorithm.

The channel utilization performance is illustrated in Figure 7b. As seen in the figure, the utilization increases as load increases too. For $C_t = 87\%$, the channel utilization is always maintained above 70% and it reaches saturation ($\simeq 100\%$) at Erlang 50.

The mean number of messages per channel allocation results are shown in Figure 7c. As seen in the figure, the mean number of messages per allocation are always less than 1. Also, as given in table 2, the proposed algorithm never produces messages higher than 3N, where the D-CAT has 3N+x messages. This can be easily be proved by the fact that, in the proposed algorithm the value k can be maximum 3. Also, the simulation results show that the algorithm finds

a channel to import in at most 2 donor list accesses (i.e. the value d can be maximum 2). Therefore, in the worst case, the maximum number of messages which will be produced by the proposed algorithm will be 2N+3dk where N=18, d=2 and k=3, which is equal to 54 messages. On the other hand, the D-CAT will have a 3N+x messages, which will be equal to 54+x. So, even under the worst case scenario, the number of messages produced by the proposed DonorList algorithm is lower than the number of messages produced by the D-CAT.

Figure 7d represents the mean time spent for each channel allocation. As seen in the figure, under all load values, the time spent for each channel allocation is lower than tolerable maximum delay, which is 100msecs[2]. The sudden increase seen when the system is heavily loaded (Erlang 45 and 50) can be explained as follows: Under heavy loads, all of the 144 cells will receive incoming calls very frequently. This causes an importer cell to successfully import a channel after the second access to the donor list or sometimes to block/drop the call. Therefore, under high loads, the worst case scenario explained in the previous paragraph occurs and since the number of massages reach the maximum, the time needed to send and process the messages increases as well.

Finally, for all the performance metrics discussed above, the proposed algorithm highly depends on the correct choice of the threshold value, C_t. If high C_t values are selected, the cells will not update their donor lists until a very high percentage of their channels become busy. This will result in low number of entries in their donor lists and high P_{incomp} values. Also, since the number of exporter cells will be low, at high C_t values, the number of messages per channel allocation and the mean channel allocation time will be lower.

On the other hand, at low C_t values, the entries in the donor list may become out of date (i.e. the reported available candidate channels may become busy).

The results show that the recommended C_t value for a stable and high-performance *DonorList* algorithm is 87%.

5 Conclusion

This paper presented a threshold based distributed channel allocation algorithm for cellular/wireless networks. The main goal of the study is to provide low call incompletion probabilities and high utilization and throughput values while keeping the number of messages for channel import processes as low as possible. The obtained results from extensive simulation studies prove that the algorithm succeeded in achieving the mentioned performance goals. Also the results show that the proposed algorithm overperforms the previously proposed algorithms in terms of the performance goals stated above. As the future work, adapting the algorithm for different service types (i.e. voice, video and data) and providing QoS to these services are being worked on. Also, the performance of the proposed algorithm under various user mobility conditions is a part of the current phase of this study.

References

1. Prakash, R., Shivaratri, N., Singhal, M.: Distributed dynamic fault-tolerant channel allocation for cellular networks. IEEE Transactions on Vehicular Technology **48** (1999) 1874–1888
2. Rappaport, T.S.: Wireless Communications-Principles and Practice. second edn. Prentice Hall, Upper Saddle River, NJ 07458 (2002)
3. Katzela, I., Naghshineh, M.: Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey. IEEE Personal Communications **3** (1996) 10–31
4. Perros, H.G., Elsayed, K.M.: Call admission control schemes: a review. IEEE Communications Magazine **34** (1996) 82–91
5. Zhang, M., Yum, T.S.P.: Comparisons of channel-assignment strategies in cellular mobile telephone systems. IEEE Transactions on Vehicular Technology **38** (1989) 211–215
6. Prakash, R., Shivaratri, N.G., Singhal, M.: Distributed dynamic channel allocation for mobile computing. In: PODC '95: Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing, ACM Press (1995) 47–56
7. Naghshineh, M., Schwartz, M.: Distributed call admission control in mobile/wireless networks. IEEE Journal on Selected Areas in Communications **14** (1996) 711–717
8. Dong, X., Lai, T.H.: Distributed dynamic carrier allocations in mobile cellular networks: search vs. update. In: ICDCS '97: Proceedings of the 17th International Conference on Distributed Computing Systems (ICDCS '97), IEEE Computer Society (1997) 108
9. Cao, G.: Integrating distributed channel allocation and adaptive handoff management for qos-sensitive cellular networks. Wirel. Netw. **9** (2003) 131–142
10. Gupta, S.K.S., Srimani, P.K.: Updatesearch: A new dynamic channel allocation scheme for mobile networks that can adjust to system loads. The Journal of Supercomputing **17** (2000) 47–65
11. Haung, Y.R., Ho, J.M.: Distributed call admission control for a heterogeneous pcs network. IEEE Trans. Comput. **51** (2002) 1400–1409
12. Zhang, Y., Das, S.K., Jia, X.: D-cat: an efficient algorithm for distributed channel allocation in cellular mobile networks. Mob. Netw. Appl. **9** (2004) 279–288
13. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of ACM **21** (1978) 558–565
14. Mathworks: Matlab v6.5 R.13. http://www.mathworks.com (Last Visited: April 2005)
15. Cao, G., Singhal, M.: Efficient distributed channel allocation for mobile cellular networks. In: In the Proceedings of the IEEE 7th International Conference on Computers and Communication Networks, IEEE (1999) 50–57
16. Das, S., Sen, S., Jayaram, R.: D-lbsb: A distributed load balancing algorithm for channel assignment in cellular mobile networks. Journal of Interconnetion Networks **1** (2000) 195–220
17. Das, S., Y.Zhang: An efficient load-balancing algorithm based on a two treshold cell selection scheme in mobile cellular networks. Computer Communications **23** (2000) 452–461