# Distributed Wireless Channel Allocation in Networks with Mobile Base Stations [1,2]

**Sanket Nesargi** **Ravi Prakash**
Computer Science Program
University of Texas at Dallas
Richardson, TX 75083-0688.
e-mail: {`sanket,ravip`}`@utdallas.edu`
Phone: (972) 883-2289, 883-2420
Fax: (972) 883-2349

Contact Author: Ravi Prakash

---

[1] This research is supported in part by NSF grant CCR-9704101.

[2] This is an extended version of the paper that will appear in the Proceedings of INFOCOM'99. The additions are: (i) Proofs presented in Section 5.5, and (ii) Additional simulation results in Section 7.

## Abstract

In traditional cellular systems with fixed base stations the channel reuse pattern is static and deterministic. When the cell layout is dynamic, due to the mobility of base stations, the cluster of cells within co-channel interference range changes with time. Consequently, the channel reuse pattern is highly dynamic. Moreover, base stations also need wireless channels to communicate amongst themselves. A communication session between a pair of nodes may have to switch channels due to the movement of other nodes into the neighborhood. Hence, there is a need for new wireless channel allocation algorithms for virtual cellular systems with mobile base stations. In this paper, principles of mutual exclusion pertaining to distributed computing systems are employed to develop such an algorithm. The inter-base station wireless links are referred to as backbone links while the base station-mobile node links are referred to as short-hop links. The proposed algorithm is distributed, dynamic and deadlock free. Disjoint sets of channels are used for backbone and short-hop links. The distributed nature of the channel allocation scheme leads to robustness as the responsibility is no longer centralized at the MTSO. Instead, it is shared among all the mobile base stations. This also makes the algorithm scalable.

# 1 Introduction

Several existing models of mobile computing systems assume a cellular network of mobile nodes and stationary nodes [4]. The geographical area served by the system is divided into regions referred to as cells. Each cell has a fixed *base station* ($BS$). A fixed wireline network connects all the base stations. The mobile nodes, referred to as *mobile hosts* ($MH$s), present in a cell can communicate with other nodes in the system only through the $BS$ of that cell. This communication is through a wireless link between the $MH$ and its $BS$. $MH$s can move from one cell to another during a communication session. There are two approaches to channel selection in such scenarios:

1. A centralized approach [3, 5, 8, 11, 20, 21] in which all requests for channel allocation are forwarded to a central controller that has access to system wide channel usage information. The central controller allocates channels to requests while avoiding interference.
2. A distributed approach [1, 4, 9, 14] in which the base stations and/or the mobile hosts monitor the signal-to-noise ratio of relevant channels and, in some cases, exchange this information. The channel use decisions are made by each node, based on its information, without involving a central controller.

We deviate from the traditional cellular model and assume a *virtual cellular network* where the fixed base stations are replaced by mobile base stations ($MBS$s). The wireline links between the fixed $BS$s are replaced by wireless links between $MBS$s. So, the entire network is wireless. Such networks have civilian and military applications as described later. The inter-$MBS$ links will, henceforth, be referred to as *backbone links* while the $MBS$-$MH$ links will be referred to as *short-hop links*. Figure 1 presents a logical view of the virtual cellular network.

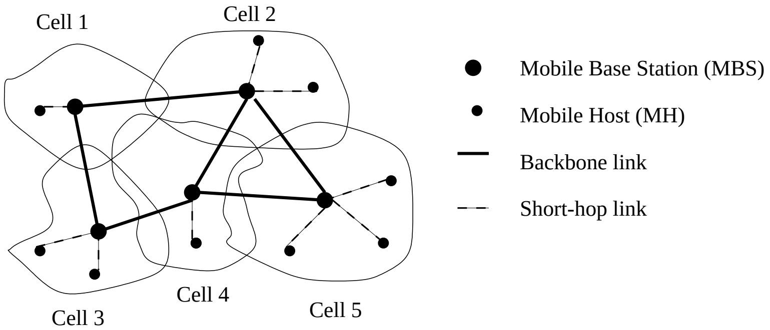

Figure 1: A fully wireless cellular network.

The relative position of cells changes with time. As the relative positions of $MBS$s cannot be determined *a priori*, it does not make sense to pre-allocate fixed sets of channels to each $MBS$ for short-hop links. Dynamic channel allocation is needed. Also, in the interest of scalability and robustness a central controller should be replaced by a distributed channel allocation mechanism. Furthermore, it is our intention to minimize the amount of work that a resource poor $MH$ should have to perform for channel allocation. Instead, the responsibility is to be shared by all the $MBS$s which are assumed to be comparatively resource rich. The reason behind such an assumption will be described later in Section 2.

We view the task of channel allocation as a form of the mutual exclusion problem studied extensively in the operating systems and distributed computing community. A mutual exclusion based algorithm for channel allocation in cellular networks was first presented by one of the authors in [14]. Here we suitably modify and enhance the algorithm for systems with mobile base stations.

The set of wireless channels is partitioned into two disjoint subsets: one subset used exclusively for backbone links and another subset used exclusively for short-hop links. Channel partitioning simplifies the task of channel allocation at the cost of utilization. In the future we will extend our work to a scenario where all channels can be used for backbone as well as short-hop links. Preliminary ideas for such an extension have been presented in [12]. The proposed algorithm is flexible in the sense that it can also be used for traditional cellular systems just by setting the $MBS$ velocity to zero.

In trying to measure the performance of the proposed algorithm through simulation experiments, continuous tracking of all the nodes is computationally expensive. We present a simple approximation strategy in conjunction with discrete event simulation which significantly reduces the cost of simulations. At the same time, it models the mobility of $MBS$s and $MH$s with a degree of accuracy of the experimenter's choice. To ensure the correctness of the algorithm[1] inspite of the approximations, the simulation experiments resort to a conservative channel allocation policy. Thus, simulation experiments using such models yield a conservative estimate of the performance of the algorithm.

The system model is described in Section 2. The problem description is presented in Section 3. The underlying theoretical basis for the algorithm, *i.e.*, interpreting channel allocation as a mutual exclusion problem is described in Section 4. The algorithm and the proof of its correctness are presented in Section 5. The cost saving simulation approximations are described in Section 6. Preliminary simulation results are presented in Section 7 followed by the conclusion in Section 8.

---

[1] By correctness we mean avoidance of co-channel interference.

# 2 System Model

We assume a system consisting of a set of mobile base stations ($MBS$s) and mobile hosts ($MH$s) connected by a completely wireless network. The $MBS$s have more resources than the $MH$s in terms of energy supply, memory, processing power, etc. The $MBS$s would be typically mounted on trucks, tanks, busses, etc. Neighboring $MBS$s communicate using wireless backbone links forming a backbone network. No assumption is made about the mobility pattern of $MBS$s. So, no guarantees are provided about the backbone network being connected all the time. Planning of $MBS$ mobility to avoid network partitioning is a separate problem that has been addressed in [17].

An $MH$ communicates only through a neighboring $MBS$ using a wireless short-hop link. An $MH$-$MBS$ pair can establish a bidirectional short-hop link provided their separation is less than a threshold value $d$. This is equivalent to an $MBS$ having a cell of radius $d$. If a wireless channel is being used to support a short-hop link between an $MH$ and an $MBS$, the same channel cannot be concurrently used to support any kind of communication within a radius $\alpha \times d$ around the involved $MBS$, where $\alpha > 1$. Thus, the short-hop channel reuse distance is $\alpha \times d$.

Two $MBS$s can establish a bidirectional wireless link provided their separation is no more than another threshold value $D$. If two $MBS$s are using a channel for a backbone link, that channel cannot be simultaneously used to support any other communication in a region consisting of the union of two circles, each of radii $\beta \times D$ centered at the respective $MBS$s. Thus, $\beta \times D$ is the backbone channel reuse distance.

We assume that $D > d$. This is consistent with the earlier assumption that $MBS$s have abundant energy supply enabling them to transmit at greater power levels than $MH$s. The variables $D$, $d$, $\alpha$ and $\beta$ are system parameters that depend on the networking hardware, power level of transmissions, fading characteristics, etc. Figure 2 presents a schematic representation of the co-channel interference ranges. Later, in Section 7, we specify the exact values of $\alpha$ and $\beta$ used for the simulation experiments.

The mobility of $MH$s is modeled by the following three steps through which the $MH$s loop:

1. When an $MH$ enters an $MBS$'s cell, the $MH$ stays in that cell for a period of time determined by a probability distribution.

2. At the end of this period the $MH$ enters an arbitrarily selected neighboring cell. Note that due to the mobility of $MBS$s the set of neighboring cells changes with time.

3. If there is no neighboring cell to migrate to, the $MH$ stays in the same cell as before. For modeling purposes this is treated just like an entry into a new cell. This ensures that the

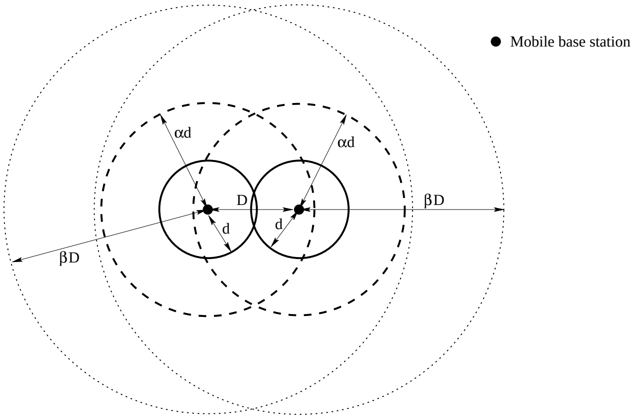

Figure 2: Co-channel interference ranges for backbone and short-hop links.

number of $MH$s in the system remains unchanged.

Two realistic scenarios where such a mobility model would be applicable are the battlefield scenario and the public transport scenario.

In a battlefield, soldiers (equivalent to $MH$s) may move along with a tank (equivalent to an $MBS$) for a while before moving towards and connecting with a neighboring tank. This switch from one tank's cell to another would continue for the system lifetime. In a civilian scenario, passengers may be traveling on a bus with several passengers carrying their $MH$s. A bus may have an antenna to communicate with each $MH$ making the inside of the bus into a cell. All the busses, acting as $MBS$s, may communicate with their neighbors and route messages across their wireless backbone network. From time to time, passengers get off a bus at bus-stops and board another bus, thus moving out of one cell and into another neighboring cell.

Two kinds of wireless channels are available: communication channels and control channels. Communication channels are used to support backbone and short-hop links. Control channels are used to send messages generated by the channel allocation algorithm. We assume that the maximum range of a communication channel is equal to $D$, the range of a backbone link. However, control channel transmissions can be sent at a higher power level, if required, so that an $MBS$ can communicate with all the other $MBS$s within the backbone co-channel interference range. Henceforth, when we use the term *channel* or *wireless channel* we mean communication channels. The wireless channels are assumed to be orthogonal to each other. So, only co-channel interference is considered.

# 3 Problem Description and Contributions

In cellular networks that have fixed cell layout, the *cluster* of cells that are within co-channel interference range do not change with time. Several fixed, dynamic, or hybrid channel allocation algorithms have been proposed for systems with fixed cell layout [1, 3, 4, 5, 6, 8, 9, 11, 14, 15, 20, 21]. When the base stations are mobile the cluster of cells within co-channel interference range changes with time. As a result, the channel reuse pattern is highly dynamic. Existing channel allocation algorithms are not designed to handle such dynamism. Hence, there is a need for new wireless channel allocation algorithms for mobile computing systems with mobile base stations. The base stations also need wireless channels to communicate amongst themselves.

This raises some interesting issues:

1. Should the available frequency spectrum be divided into two disjoint sets of channels: a set of backbone channels and another set of $MBS$-$MH$ short-hop communication channels?
2. If the answer to the first question is in the affirmative, how many channels should be designated as backbone channels? As the base stations move, their adjacency graph changes. Hence, the required number of backbone channels changes with time. Note that the separation of backbone channels from short-hop channels will simplify the channel allocation problem at the cost of efficiency of channel utilization. If the adjacency matrix of base stations is sparse, several backbone channels may be unutilized, and they cannot be used for $MBS$-$MH$ communication either.

Concurrent presence of backbone and short-hop links with different signal strengths and range has some similarities with hierarchical cellular systems [19] having smaller microcells overlaid with larger umbrella cells. However, there are several important differences as well: (i) In hierarchical systems the relative configuration of microcells and umbrella cells remains unchanged. (ii) In hierarchical systems at least one node connected by a wireless link is fixed. However, in the proposed system both base stations connected by a backbone link may be moving. Hence, solutions for hierarchical cellular systems cannot be directly applied in the proposed scenario.

The problem at hand is to:

1. Develop a dynamic channel allocation algorithm for backbone as well as short-hop links.
2. Make channel allocation decisions in a distributed fashion so as to tide over the absence of any fixed central controller in a virtual cellular network, as well as to provide scalability and robustness.

3. Keep the involvement of the resource poor $MH$s to a minimum.

For simplicity we assume that the set of wireless channels is partitioned into two disjoint subsets: one subset used exclusively for backbone links and another used exclusively for short-hop links.

## Contributions of this paper

A distributed, dynamic channel allocation algorithm is presented. The algorithm does not need a central network switch. The $MBS$s make all the decisions based on the information available locally. An $MBS$ needs to exchange information with only its neighboring $MBS$s within the co-channel interference range. Unlike the *fixed channel allocation (FCA)* algorithms, the proposed algorithm can adapt to changing load distribution in the network. It is more robust than existing *dynamic channel allocation (DCA)* algorithms as it does not depend on a central network switch whose failure can bring down the entire network. The algorithm also exploits the temporal locality of load distribution to make quick decisions about channel allocation.

The salient features of the proposed algorithm are:

1. *Bounded latency*: No mobile node that wishes to acquire a wireless channel for a communication session is made to wait indefinitely before it is either allocated a channel or is informed of a failure to do so. Bounded latency is desirable to guarantee a certain quality of service to the users.

2. *Deadlock freedom*: There is no possibility of finding a set of mobile base stations involved in a circular wait while trying to satisfy channel allocation requests. So, the algorithm always makes progress. Resources are not wasted in detecting or resolving deadlocks.

3. *Symmetry*: All the $MBS$s follow the same procedure for channel allocation. Hopefully, there will be no need to drastically redesign hardware, or develop new software if more $MBS$s or $MH$s are to be added.

4. *Low system overhead and network traffic*: As the proposed algorithm adapts to the locality of load distribution, each new channel allocation request is handled with an exchange of zero or a small number of messages between the mobile base stations.

5. *Concurrency*: Requests for channel allocation originating independently and concurrently in different parts of the network can be processed simultaneously.

# 4 Theoretical Framework and Basic Idea

Mutual exclusion is an extensively studied problem in the fields of operating systems and distributed computing [2, 7, 10, 16, 18]. Multiple processes may be concurrently trying to access a shared resource. For various reasons it is necessary that these processes get access to the shared resource in a mutually exclusive fashion. A process is said to be in its critical section during the time it is using the shared resource.

## 4.1 Channel Allocation vs. Mutual Exclusion

In the context of a pair of communicating nodes ($MH$-$MBS$ or $MBS$-$MBS$), the use of a particular channel to support a communication session is equivalent to a critical section execution by the communication session where the channel is the shared resource. Several neighboring cells may be concurrently trying to choose channels to support sessions in their region. This can lead to conflicts because the number of communication channels is limited. The resolution of such conflicts is similar to the mutual exclusion problem [2, 18]. The system has two distinct classes of shared resources: short-hop channels and backbone channels. Mutual exclusion for one class of resources is entirely independent of mutual exclusion for the other.

However, the channel allocation problem is more general than the mutual exclusion problem. First, an $MBS$ may be supporting multiple short-hop and/or backbone communication sessions, each session using a different communication channel. This is equivalent to an $MBS$ being in multiple, distinct critical sections concurrently. Second, existing mutual exclusion algorithms for distributed systems [2, 7, 10, 16, 18] assume that a node specifies the identity of the resource it wants to access in a critical section. Depending on the availability of that resource, appropriate decisions can be made. However, in distributed channel allocation, a cell asks for *any* channel as long as there is no co-channel interference. Due to the non-specificity of the request and because neighboring mobile base stations make channel allocation decisions independently based on locally available information, the decision process becomes more difficult. Third, depending on whether a channel is being used for short-hop or backbone link, the co-channel interference range is different. This is quite different from the approach taken by distributed mutual exclusion models that make no distinction between who is using a resource or for what application: as long as a resource is being used by one process in the system, it *cannot be used anywhere* in the system.

Moreover, existing distributed mutual exclusion algorithms do not impose any upper bound on the elapsed time between a node's request for the resource and the granting of that resource. These algorithms are not suitable for the channel allocation problem which requires that decisions be made in real-time. So, a conservative approach that makes the channel allocation decisions quickly needs to be adopted. Such an approach may drop calls that a more general but time consuming approach would have supported. This is a trade-off that has to be accepted.

The mobility of base stations adds yet another degree of complexity to the problem. A short-hop channel may be concurrently assigned to two different short-hop sessions that are, initially, not in each other's co-channel interference range. However, due to the movement of the $MBS$s and the $MH$s, during the lifetime of these communication sessions, the nodes involved in the two sessions may start approaching the co-channel interference region of each other. In order to avoid any conflict, at least one session will have to switch to another channel. A similar situation may arise for two backbone links that were initially far apart and were using the same channel. In the context of mutual exclusion, this is analogous to the pre-emption of the critical section execution of a node with a subsequent attempt to acquire another shared resource and enter a different critical section.

The channel allocation algorithm proposed in the next section is based on the ideas in the Ricart-Agrawala algorithm for mutual exclusion [16].

## 4.2 Basic Idea

An $MBS$ makes all channel allocation decisions on behalf of the $MH$s in its cell. In some situations described in the next section, based on local information, an $MBS$ can assign a channel to a short-hop link between itself and an $MH$ without causing any interference, and without having to consult the neighboring $MBS$s. Otherwise, requests time-stamped with Lamport's clock [7] are sent by the $MBS$ to neighboring $MBS$s to determine the channel to be assigned for the short-hop link. As already mentioned in Section 2, in the context of short-hop channel allocation, the list of neighboring $MBS$s consists of all $MBS$s within a distance of $\alpha \times d$ from the $MBS$ in question. The information received in the replies from the neighboring $MBS$s is used to determine the channel to be allocated to the short-hop link. The distributed nature of the algorithm, and the finite but non-deterministic propagation delays of messages between $MBS$s can lead to co-channel interference if a naive channel selection strategy is employed: multiple cells in each other's interference range may concurrently and independently decide to use the same channel for short-hop links in their cells. Such a possibility is prevented by associating a priority among concurrent requests based on their timestamps. As in the Ricart-Agrawala algorithm for mutual exclusion [16], an $MBS$ with a higher priority request defers replying to a lower priority channel allocation request.

Similarly, if channels have to be transferred from a lightly loaded cell[2] to a heavily loaded

---

[2] Cell load is the ratio of the number of short-hop channels the cell can use to set up links without any interference

neighbor, conflict is avoided in the following manner: having selected a communication channel for transfer, based on a round of message exchange with its neighbors, the mobile base station sends the channel identity to the neighboring mobile base stations. Only if all the neighboring mobile base stations approve of the selection is the channel transferred, otherwise not.

The set of channels allocated to a cell varies with time. Unlike existing DCA algorithms [20, 21], a newly acquired channel (acquired through transfer, for example) is not relinquished by a cell on completion of the communication session it was supporting in the cell. Instead, the channel remains allocated to the same cell until it has to be transferred to a neighboring cell. This enables the algorithm to adapt to temporal and spatial changes in load distribution. It also helps reduce the traffic due to channel allocation requests in the fixed wire network.

# 5 Channel Allocation Algorithm

The data structures and the strategy to allocate channels for short-hop links build upon those described in [14] for traditional cellular systems where the base stations are fixed. However, additional information needs to be maintained to allocate channels for backbone links without causing co-channel interference. Hence, the data structures described in [14] have to be augmented. For the sake of completeness and readability the entire algorithm is presented instead of only presenting the differences from the algorithm presented in [14].

## 5.1 Data Structures

All the communication channels in the system are collectively represented by a set $Spectrum$. $Spectrum$ is divided into two disjoint sets: $Spectrum_s$ and $Spectrum_b$ for short-hop and backbone links, respectively. We assume that all the channels can be totally ordered. The channel with the lowest frequency band is considered to be the first channel and the channel with the highest frequency band is the $n^{th}$ channel, where $n$ is the total number of channels in $Spectrum_b$. Similar ordering applies for channels in $Spectrum_s$.

The set of short-hop channels allocated to cell $C_i$ is represented by $Allocate_{s,i}$. Initially, $Allocate_{s,i}$ is an empty set for every cell $C_i$. The set of channels being used in a cell constitute its *busy* set. Unlike [14], the busy channels of cell $C_i$ are distinguished into $Busy_{s,i}$ and $Busy_{b,i}$ denoting the channels being used for short-hop and backbone links, respectively. For cell $C_i$, $Busy_{s,i}$ is always a subset of $Allocate_{s,i}$. When a new short-hop communication request originates in $C_i$, one of the non-busy channels in $Allocate_{s,i}$ is assigned to support the communication session. If there

---

to the number of $MH$s that are requesting such channels. Lower the demand to availability ratio, lower the load.

is no such channel, then after a round of message exchange with the neighbors, a non-busy channel that is in $Spectrum_s$, but not in the short-hop *Allocate* set of the cell or any of its neighbors is added to $Allocate_{s,i}$ as well as $Busy_{s,i}$. This channel is used to support the short-hop session. If such an attempt fails, $C_i$ tries to transfer a non-busy channel from the *Allocate* set of its neighbors to $Allocate_{s,i}$. If such a transfer is not possible, the communication request is dropped. Otherwise, the communication is successfully completed.

Also, cell $C_i$ maintains a transfer set, $Transfer_{s,i}$, consisting of the channels earmarked as candidates for possible transfer from $C_i$ to one of its neighbors to support short-hop links involving their base stations. *Transfer* sets are initially empty for all the cells. All these sets are maintained by the corresponding mobile base stations.

Several communication requests may originate in a cell concurrently. These new requests may be ordered according to a policy decided *a priori*, like arrival time or *MH_id*. Only after the mobile base station has made a channel allocation decision about one locally originating request, does it process the next locally originating communication request in the sequence.

## 5.2 Short-hop Channel Allocation

(A) When a communication session is to be set-up in cell $C_i$, the following actions are taken by its mobile base station ($MBS_i$):

```
1. $T_i \leftarrow T_i + 1$, where $T_i$ is the Lamport's clock at $MBS_i$;

2. $RT_i \leftarrow T_i$ /* $RT_i$ has the time-stamp of this channel request. */

3. If $(Available_{s,i} \leftarrow Allocate_{s,i} - Busy_{s,i} - Transfer_{s,i}) \neq \Phi$, then

       A highest order channel $k$ from $Available_{s,i}$ is selected to set-up the short-hop link;
       $Busy_{s,i} \leftarrow Busy_{s,i} \cup \{k\}$;
       Go to step 10;

   else /* $Available_{s,i} = \Phi$ */

       Send time-stamped REQUEST messages to each neighboring $MBS$ within short-hop co-channel interference range.[3]


4. When $MBS_i$ has received REPLY messages from each of its neighbors to which REQUESTs were sent, containing their short-hop Allocate, Busy and Transfer sets, it takes the union of $Allocate_{s,i}$ and the short-hop Allocate sets received in the REPLY messages, and stores the
```

---

[3] The set of neighbors changes with time due to $MBS$ mobility. We assume that an $MBS$ knows the identity of neighboring $MBS$s by listening to their beacons or by employing proximity determination protocols of the type described in [13].

result in $Interfere_{s,i}$. Due to the mobility of $MBS$s it is possible that an $MBS$ to which a REQUEST was sent has moved out of the co-channel interference range and can, or will, no longer send a REPLY. If $MBS_i$ does not receive a REPLY from $MBS_j$ within a timeout period $MBS_i$ assumes that: (i) $MBS_j$ has either crashed or moved out and does not send any subsequent messages to $MBS_j$, (ii) a reply has been received from $MBS_j$ indicating that its Allocate, Busy and Transfer sets are empty. Also, if $MBS_k$ that was hitherto out of short-hop co-channel interference range moves into the range during the execution of the channel allocation process, $MBS_i$ sends a REQUEST to $MBS_k$ and waits for its REPLY.

5. If $(Free_{s,i} \leftarrow Spectrum_s - Interfere_{s,i}) \neq \Phi$, then a channel of the highest order is selected from $Free_{s,i}$ and added to $Allocate_{s,i}$. This channel is used to support the communication session. So, it is added to $Busy_{s,i}$ as well. Then go to step 10.

6. If $Free_{s,i} = \Phi$, it does not mean that no channel is available for allocation. Perhaps, the communication session can be supported by transferring a channel. $MBS_i$ takes the union of $Busy_{s,i}$, $Transfer_{s,i}$, and the short-hop $Busy$ and $Transfer$ sets received in the REPLY messages in step 4, and stores the result in $Interfere_{s,i}$.

7. If $(Free_{s,i} \leftarrow Spectrum_s - Interfere_{s,i}) = \Phi$, then the short-hop channel request is dropped. Otherwise, the channel of the lowest order in $Free_{s,i}$ is chosen for the transfer.

8. Let the channel selected for transfer be $k$.
   $Busy_{s,i} \leftarrow Busy_{s,i} \cup \{k\}$;
   $Allocate_{s,i} \leftarrow Allocate_{s,i} \cup \{k\}$;
   $MBS_i$ sends TRANSFER($k$) messages to all the neighbors in the co-channel interference range.

9. If all the neighboring cells to which the TRANSFER message was sent reply AGREED:

   Channel $k$ is used to support the communication session.
   $MBS_i$ sends RELEASE($k$) messages to all the neighboring cells.
   Go to Step 10.

   Otherwise: /* Some cells have sent REFUSE message. */

   $Allocate_{s,i} \leftarrow Allocate_{s,i} - \{k\}$;
   $Busy_{s,i} \leftarrow Busy_{s,i} - \{k\}$;
   $MBS_i$ sends KEEP($k$) messages to the $MBS$s of all the neighboring cells.
   $MBS_i$ selects the next channel from $Free_{s,i}$, with order greater than that of $k$, and steps 8 and 9 are repeated. [4] To avoid excessive channel transfer overheads, under heavy load situations, the number of transfer attempts can be limited to the minimum of a THRESHOLD value (parameter of the algorithm) and the cardinality of $Free_{s,i}$. If all attempts to transfer a channel fail, the communication request is dropped.

---

[4] The KEEP messages can be piggybacked on TRANSFER messages, if they are going to the same cell.

If $MBS_i$ does not receive an AGREED or REFUSE message from $MBS_j$ (to which the TRANSFER message was sent) within a timeout period, $MBS_i$ assumes that: (i) $MBS_j$ has crashed or moved out of co-channel interference range and is therefore of no consequence, (ii) an AGREED message is received from $MBS_j$.

10. Once a cell has decided to drop a request or to use a channel to support the corresponding communication session it sends all the deferred REPLYs to its neighbors.

11. When a short-hop communication session terminates in cell $C_i$, the corresponding channel is deleted from the set $Busy_{s,i}$.

(B) When $MBS_j$ receives a REQUEST message from $MBS_i$ with timestamp $T_i$:

$T_j \leftarrow T_j + 1;$
$T_j \leftarrow \max(T_j, T_i + 1);$

$MBS_j$ sends a REPLY message to $MBS_i$ if $MBS_j$ itself is not requesting a channel for a short-hop link, or if $MBS_j$ is requesting a channel and $MBS_i$'s request's time-stamp is smaller than $MBS_j$'s request's time-stamp (i.e., $T_i < RT_j$ or $T_i = RT_j$ and $i < j$). Otherwise, the REPLY is deferred.[5] As $MBS_i$ only uses the union of the $Busy_{s,j}$ and $Transfer_{s,j}$ sets received in the REPLYs, in Step $(A)$.6, and never uses the two sets separately, the communication overheads can be reduced by taking their union at $MBS_j$ and sending the result, rather than both the sets, in the REPLY message. Therefore, the REPLY message contains $Allocate_{s,j}$, and the union of $Busy_{s,j}$, and $Transfer_{s,j}$.

(C) When $MBS_j$ receives TRANSFER($k$) message from $MBS_i$: if $k \notin Allocate_{s,j}$ then send AGREED($k$) message to $MBS_i$. Otherwise, if $k \in Allocate_{s,j}$:

If $(k \in Busy_{s,j})$ OR $(k \in Transfer_{s,j})$ then send REFUSE($k$) message to $MBS_i$. Otherwise $Transfer_{s,j} \leftarrow Transfer_j \cup \{k\}$; Send AGREED($k$) message to $MBS_i$.

(D) When $MBS_j$ receives a RELEASE($k$) message, the following actions take place.

$Allocate_{s,j} \leftarrow Allocate_{s,j} - \{k\};$
$Transfer_{s,j} \leftarrow Transfer_{s,j} - \{k\};$

(E) When $MBS_j$ receives KEEP($k$) message, the following actions take place.

$Transfer_{s,j} \leftarrow Transfer_{s,j} - \{k\};$

If, after sending an AGREED message to $MBS_i$, $MBS_j$ does not receive either a KEEP or a RELEASE message from $MBS_i$ within a timeout period, $MBS_j$ assumes that it has: (i) moved out of range of $MBS_i$, (ii) received a KEEP message from $MBS_i$ and acts accordingly.

---

[5] The timeout duration in Step A.4 is large enough so that REPLY deferral does not cause timeouts.

## 5.3 Backbone Channel Allocation

Backbone channel allocation is similar to short-hop channel allocation, except for the following differences:

1. If a wireless channel is needed to establish a backbone link between two mobile base stations, $MBS_i$ and $MBS_j$, all the mobile base stations that are within a distance of $\beta \times D$ of at least one of $MBS_i$ and $MBS_j$ are polled to gather information about backbone channels in use in this region. This is consistent with the assumption made in Section 2 about the co-channel interference range of a backbone channel being $\beta \times D$.

2. There is no notion of allocate and transfer sets for backbone links. Let $MBS_i$ and $MBS_j$ be the mobile base stations between which a backbone link is to be established. REQUESTs are sent to all the $MBS$s within the backbone co-channel interference range of this $MBS$ pair. The timestamp of the backbone channel requests is a tuple of the form $(max(T_i,\ T_j), MBS_i, MBS_j)$. As in the short-hop channel allocation case, requests are prioritized by their timestamps: lexicographically lower timestamp indicates higher priority.

3. A mobile base station receiving the REQUEST immediately sends or defers the REPLY depending on whether (i) it is also involved in establishing a backbone link with a neighboring $MBS$, and (ii) the timestamp of its backbone channel request.

4. The REPLY sent by $MBS_k$ contains $Busy_{b,k}$: the set of backbone channels $MBS_k$ is currently using for its backbone links.

5. On receiving all the replies, the $Interfere_{b,i,j}$ set consists of the union of $Busy_{b,i}$, $Busy_{b,j}$ and the busy backbone sets received in the replies.

6. $Free_{b,i,j} \leftarrow Spectrum_b - Interfere_{b,i,j}$.
   If $Free_{b,i,j} = \Phi$, no backbone channel is available. Otherwise, the highest order channel is selected from $Free_{b,i,j}$ to establish a backbone link and that channel is added to $Busy_{b,i}$ and $Busy_{b,j}$.

7. When a backbone link between $MBS_i$ and $MBS_j$ is broken, the channel that was being used to support the link is deleted from $Busy_{b,i}$ and $Busy_{b,j}$.

## 5.4 Channel Reconfiguration During Connection Lifetime

There are two situations in which a backbone or a short-hop link between two nodes may have to switch from one channel to another:

1. If an $MH$ moves out of one cell into another while involved in a communication session, handoff has to take place. The short-hop link between the $MH$ and the old $MBS$ has to be terminated and a new short-hop link has to be established between the $MH$ and the new $MBS$. As the problem has already been extensively studied, and existing solutions are applicable in the proposed system model, we skip the details of handoffs.
2. As stated earlier in Section 4, there is a possibility that a short-hop link between an $MBS$-$MH$ pair or a backbone link between a pairs of $MBS$s may have to switch to a different channel during the lifetime of the corresponding session. This is not handoff as the pair of communicating nodes does not change. This switch from one channel to another is necessitated primarily by the mobility of $MBS$s as described below.

Let there be a short-hop link between $MBS_i$ and $MH_i$ using a short-hop channel $i$. Concurrently, let there be another short-hop link in the network between $MBS_j$ and $MH_j$ also using channel $i$. Let the initial separation between $MBS_i$ and $MBS_j$ be greater than $\alpha \times d$, the short-hop co-channel interference range. At a later time, while the two short-hop sessions are still in progress, let one or both of $MBS_i$ and $MBS_j$ start moving towards the other. As stated earlier in the system model, the $MH$s tend to move with the $MBS$ to which they are connected. When the separation between $MBS_i$ and $MBS_j$ becomes smaller than $\alpha \times d$ the two short-hop links $MBS_i$-$MH_i$ and $MBS_j$-$MH_j$ using the same channel start interfering with each other. At least one of these links has to switch to another short-hop channel to avoid any further interference. Without loss of generality, let us assume that the $MBS_i$-$MH_i$ link has to switch to another short-hop channel. The procedure followed is equivalent to the termination of the old short-hop session between the node pair immediately followed by a new short-hop channel allocation between them.

Similarly, let us consider two backbone links: between mobile base station pairs $MBS_i$-$MBS_j$ and $MBS_k$-$MBS_l$. Both $MBS_i$ and $MBS_j$ are more than $\beta \times D$ distance away from $MBS_k$ and $MBS_l$. So, the two mobile base station pairs are not within backbone co-channel interference range of each other. Both backbone links are using the same backbone channel without interfering with each other. Subsequently, if these nodes move so as to be within the backbone co-channel interference range of each other, at least one link will have to switch to another channel.

The responsibility for channel reconfiguration lies with the mobile base stations. When the hitherto far apart $MBS_i$ and $MBS_j$ detect that they have moved within distance $\beta \times D$ of each other, they exchange information about their backbone channel usage and make the appropriate channel reconfiguration decisions. When $MBS_i$ and $MBS_j$ detect that they have moved to within $\alpha \times d$ of each other, they exchange information about their short-hop channel usage and determine if any short-hop links need to be switched to other channels. We assume that the underlying MAC sub-layer and network layer protocols for node beacons along with timestamps and location stamps enable $MBS$s to determine their distance from each other. These protocols are described in [13].

### 5.5 Proof of correctness

**Lemma 1** *The channel allocation algorithm avoids co-channel interference.*

**Proof sketch:** The algorithm for short-hop channel allocation is similar to that described in [14]. So, refer to the corresponding proof for non-interference during short-hop channel allocation.

As backbone and short-hop links use disjoint sets of channels, there is no interference between these two types of links. Also, once a channel has been tentatively marked in cell $C_i$ for possible transfer to $C_j$, all other transfer attempts for that channel are refused until a final transfer decision has been made with regard to it. So, a channel cannot be simultaneously transferred to multiple cells that may be in each other's interference range.

A backbone channel is selected to support a link if and only if that channel is not being used to support any other backbone link in the backbone co-channel interference range.

During the lifetime of a session, channel reconfiguration is performed to avoid interference due to node mobility.

∎

**Lemma 2** *Each new request for a communication session originating in a cell $CB_i$ causes a finite number of messages to be exchanged between the mobile base stations of the cell and its neighbors.*

**Proof:** The following situations can arise:

1. If the channel request for a short-hop link can be satisfied locally (Allocate set − Busy sets − Transfer sets $\neq \Phi$), no messages are exchanged between the mobile base stations.
2. If the request is for a short-hop link and $Free_i \neq \Phi$ in step (A).5, at most $2N$ messages are generated to allocate a channel to the communication session, where $N$ is the number of neighboring cells in the short-hop co-channel interference range: $N$ REQUESTs from $MBS_i$ to its neighbors, and a REPLY from each neighbor to $MBS_i$.
3. If the request is for a short-hop link and $Free_i \neq \Phi$ in step (A).7, $5N \leq$ *messages needed to make a channel allocation decision* $\leq 2N+3N\times minimum(\mid Free_i \mid, SHORT\_HOP\_THRESHOLD)$. Besides the $2N$ messages already mentioned, at most $N$ TRANSFER(k) messages from $MBS_i$ to its neighbors, an AGREED or REFUSE message from each neighbor to $MBS_i$,

and finally a RELEASE(l) or KEEP(l) to each neighbor are needed per channel transfer attempt. The number of attempts is upper bound by the minimum of $\mid Free_i \mid$ and a SHORT_HOP_THRESHOLD. Therefore, the message complexity is $O(N)$.

4. Similarly, if the request is for a backbone link, the number of messages needed to make a channel allocation decision is upper bound by $2N'$, where $N'$ is the number of mobile base stations in co-channel interference range of the backbone link to be established.

$\blacksquare$

**Lemma 3** *The channel allocation algorithm is deadlock free.*

**Proof:** New channel requests originating concurrently in different cells get totally ordered by their timestamps. A mobile base station with REPLYs pending to its own REQUESTs, sends REPLYs to all REQUESTs with a lower timestamp and defers other REPLYs. As the same ordering of channel requests is seen by all the nodes, there is no circular deferring of REPLYs among the mobile base stations.

During the interval between sending a TRANSFER(l) message to other $MBS$s, and receiving either a REFUSE or an AGREED message from them, an $MBS$ does not suspend replying to TRANSFER(l) messages it may itself receive from other $MBS$s. Instead, it responds to such transfer attempts with a REFUSE message during this interval. This conservative policy may lead to some requests, that could have otherwise been supported, being dropped. However, it avoids any circular wait during the channel transfer attempts, thus preventing deadlocks. $\blacksquare$

## 6 Parameter Modifications for Simulation Speedup

In Section 5.4 it was mentioned that the mobile base stations need to keep track of the $MBS$s in their proximity for the purpose of channel reconfiguration and to avoid co-channel interference. This would require continuous tracking of the separation between pairs of mobile base stations making the simulation very expensive. Moreover, there may also be need for frequent channel reconfigurations which is an expensive operation. This has the potential to slow down the simulation.

A means to increase the speed of the simulation experiments would be to perform channel reconfigurations at fixed time intervals. However, a naive implementation of such a policy may lead to co-channel interference, thus violating the correctness of the simulation experiments. Therefore, we propose to employ a conservative simulation strategy with the following properties:

1. The short-hop link establishment/break process between an $MH$-$MBS$ pair can be initiated as soon as the $MH$ makes the request for such a link (call arrival) or sends a disconnect message.
2. Inter-$MBS$ distances are measured at fixed intervals of time, and it is only at these times that backbone link establishment and tear-down as well as channel reconfigurations for short-hop and backbone links are performed.

The cost of simulation speedup is reduced channel utilization, as will become obvious shortly.

Let the *inter-reconfiguration interval* be $t$, and let the upper bound on the speed of $MBS$s be $s$. Then, the distance an $MBS$ could have moved during the interval is bound by $s \times t$.

When allocating a channel for a short-hop link between an $MH$ and $MBS_i$, $MBS_i$ considers all $MBS$s within distance $\alpha d + 2st$ to be within its short-hop co-channel interference range. Let the short-hop channel selected to establish the link be channel $i$. This ensures that at the time of this short-hop link establishment the minimum separation between $MBS_i$ and another mobile base station $MBS_j$ using channel $i$ at the same time is $\alpha d + 2st$. The maximum time until the next reconfiguration is $t$. If until this next reconfiguration time both $MBS_i$ and $MBS_j$ move directly towards each other at the maximum speed $s$ they cannot get any closer than $\alpha \times d$ until reconfiguration is performed, at which time one of the short-hop links will be made to switch to another channel.

Similarly, a link is established between two $MBS$s provided their separation is no more than $D - 2st$, instead of $D$ as stated earlier. Also, the backbone co-channel interference range is considered to be equal to $\beta D + 2st$, *i.e.*, every base station within this distance of at least one of the two $MBS$s is polled. As a result, even if the two $MBS$s participating in the backbone link keep moving directly away from each other, by the next channel reconfiguration occurrence, their separation from each other will be no more than $D$, *i.e.*, they will still be in backbone range. Also, all other $MBS$s using the same backbone channel will be at least $\beta \times D$ away from these two $MBS$s until the next reconfiguration.

## 7 Simulation Experiments and Results

We conducted simulation experiments to evaluate the performance of the proposed algorithm. We assumed a system composed of 100 mobile base stations and 1000 mobile hosts. All the nodes in the system were always located within a square of side 15 kilometer. Initially, the mobile base stations were uniformly distributed within the square with the immediate neighbors along the $x$- and $y$- axes at distance of 1.5 kilometer from each other, as shown in Figure 3

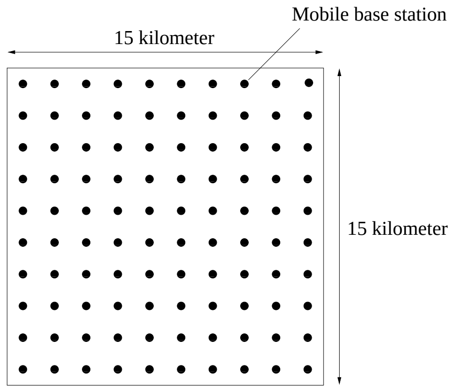

Figure 3: Initial position of 100 mobile base stations in simulation model.

The mobile base stations moved in a random fashion in the square at a constant speed, $s$, of 10 kilometer per hour. Every $MBS$ randomly chose any point in the square as its next destination and moved towards it until it got there. Then, it chose the next point and moved towards it, and so on. This was repeated until the end of the simulation. Initially, each $MH$ was associated with an $MBS$ in whose cell it was resident and an equal number of $MH$s were associated with each $MBS$. The $MH$s moved with the $MBS$ with which they were associated. The time an $MH$ spent in an $MBS$'s cell before moving into a neighboring $MBS$'s cell was exponentially distributed with a mean of 30 minutes. If the old $MBS$ did not have any neighbor, the $MH$ continued to stay in the same cell for a period of time obtained from the distribution mentioned above.

The length of a short-hop communication session (also referred to as a *call*) was exponentially distributed with a mean of 3 minutes. For a given $MH$ the time between two successive short-hop channel requests (the reciprocal of the *call arrival rate*) was also exponentially distributed and the mean value was varied to simulate various levels of channel demand.

We assumed that the number of backbone channels was large enough so that no backbone link establishment was prevented for lack of a channel. We made this assumption because the purpose of our experiments was to study the impact of mobility and channel demand on short-hop channel allocation. However, in our simulation experiments we did measure the number of backbone channels being used to get an idea of how many such channels are really needed. We also varied the total number of short-hop channels in the system from 20 to 100. .

We assumed that the range of a cell ($d$) is 1 kilometer. In the initial configuration the entire 15 $km \times 15$ $km$ square is served by at least one $MBS$ with some overlap between neighboring cells. Two $MBS$s can have a backbone link between them provided they are not more than $2 - 2st$ kilometer apart, *i.e.*, $D$ is equal to 2 kilometer. The values of $\alpha$ and $\beta$ were set to 4 and 3, respectively. So, if an $MBS$ is using a channel for a short-hop link, the same channel can be used by another $MBS$ to support a short-hop link provided the other $MBS$ is *at least* $4d = 4$ *kilometer* away. Also, two $MH$s concurrently using the same channel for their respective short-hop links can never be less than 2 kilometer from each other, thus avoiding co-channel interference. The same channel can be used to concurrently support two backbone links provided both $MBS$s in one pair are at least $3D = 6$ *kilometer* away from both $MBS$s in the other pair.

In each run of the simulation, no data collection was performed until the first $15,000$ calls were completed. This was done to filter out the impact of any initial transient effects. Then, data collection was performed until the next $50,000$ calls were either completed or dropped due to lack of channels.

We conducted experiments described below. For each experiment, the following values were measured: (i) percentage of calls that were dropped due to non-availability of short-hop channels, (ii) average number of channel allocation messages sent per call, (iii) number of short-hop links reconfigured during each reconfiguration stage, (iv) average number of backbone links in existence during the simulation period, and (v) average number of backbone channels in use during the simulation period.

## 7.1 Experiment I: Short-hop Link Characteristics

In this experiment the short-hop call arrival rate and the number of channels in the short-hop spectrum, $Spectrum_s$, were varied. Their impact on the number of dropped calls was studied. Calls that could not be connected initially, or were discontinued at the time of handoff or reconfigurations due to non-availability of channels are all counted as dropped calls. The simulation results are shown in Figure 4. For these experiments the inter-reconfiguration interval is set to 2 minutes.

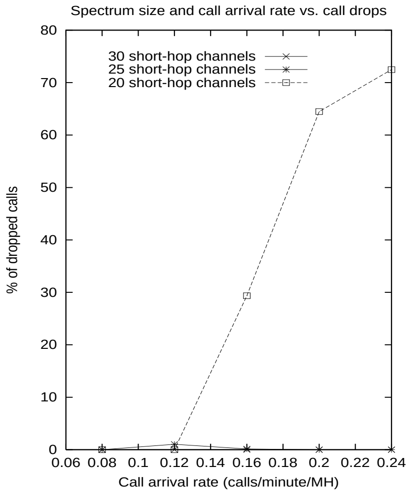

Figure 4: Impact of spectrum size and call arrival rate on the number of call drops.

When the call arrival rate is low, no call is dropped as it is possible to find an available channel from $Spectrum_s$. However, as the call arrival rate increases, for a given size of $Spectrum_s$, the availability of free channels decreases. Beyond a point, calls start getting dropped. With further increase in the call arrival rate, the percentage of dropped calls increases steadily. This holds for all the $Spectrum_s$ sizes that were simulated. Obviously, for a given call arrival rate, the greater the size of $Spectrum_s$ the higher the availability of free short-hop channels and lower the percentage of dropped calls.

The impact of call arrival rate and the size of $Spectrum_s$ on the number of messages required to establish and maintain short-hop links is shown in Figure 5. When $Spectrum_s$ is large and/or arrival rate is low, the number of short-hop messages needed to allocate a channel is small, and remains almost unchanged. Also, note that the number of short-hop messages in such situations is considerably smaller than the expected number of $MBS$s within short-hop co-channel interference range. This is because when a short-hop channel request arrives at an $MBS$, with high probability the $MBS$ finds that it has a channel in its *Allocate* set that is not *busy*. So, the channel request can be satisfied locally without having to send any REQUESTs to the neighboring $MBS$s. Even if there is no free channel in the *Allocate* set, our experiments indicate that the channel request can be satisfied in one round of REQUEST and REPLY exchanges, with negligible instances when channel transfers are required.

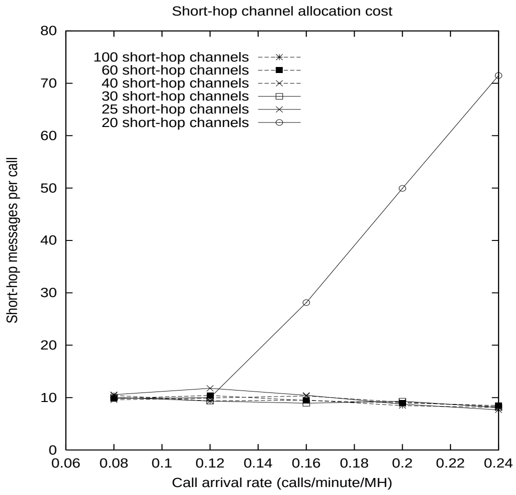

Figure 5: Impact of spectrum size and call arrival rate on the cost of maintaining short-hop links.

However, for smaller sizes of $Spectrum_s$, and beyond a certain call arrival rate, the likelihood of finding a free channel diminishes. So, a larger fraction of call arrivals result in the exchange of REQUESTs and REPLYs, and even in channel transfers. So, the number of short-hop messages per call increases with the rate of call arrival. This is evident in the figure when only twenty short-hop channels are available.

Most of the time a system will operate in situations of low to moderate load. In such operating conditions the number of messages exchanged for channel allocation will be small. So, the proposed algorithm will incur low overheads. Note that we have made no assumptions about the mobility pattern of $MBS$s. So, there will be periods when several $MBS$s can come close to each other requiring a larger number of messages to be exchanged per call. During periods when the $MBS$s are far apart, there will be fewer $MBS$s in a given $MBS$'s short-hop co-channel interference range leading to fewer messages. The numbers reported here are long term averages which even out the temporal variations.

## 7.2 Experiment II: Impact of Reconfiguration Interval

First, we measured the number of short-hop links that have to be assigned a different channel at the time of reconfiguration, with the reconfiguration interval equal to 2 minutes. Then, we also measured the impact of varying the reconfiguration interval on (i) the cost of maintaining short-hop links, and (ii) the number of backbone links and the cost of setting up such links. The simulation results are shown in Figures 6 and 8.

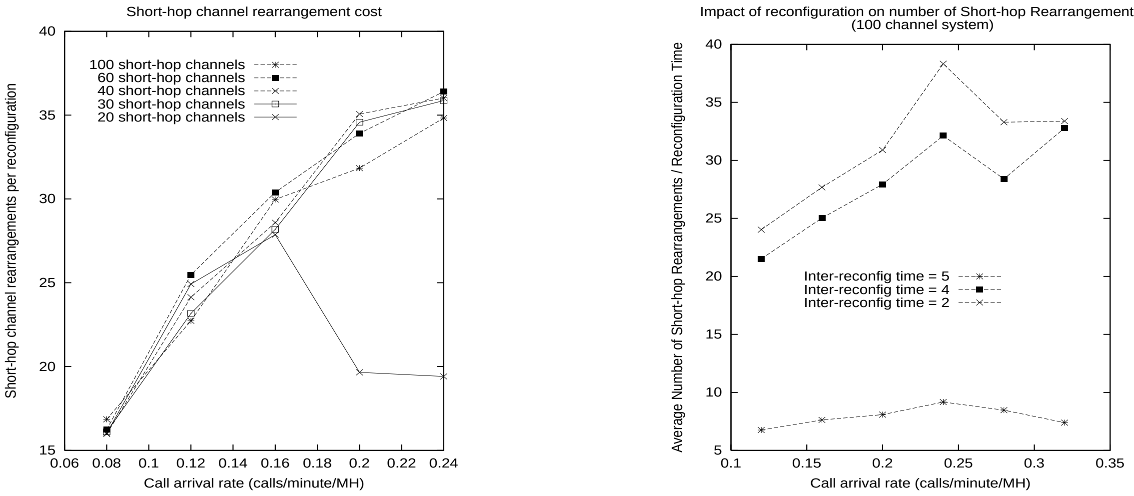

Figure 6: Impact of channel reconfigurations on short-hop links.

**Impact on number of channel rearrangements:** During channel reconfiguration (which happens once every two minutes), the number of calls in progress is small if the call arrival rate is low. Hence, the number of short-hop links requiring channel rearrangements at the time of reconfiguration is also small.

As the call arrival rate increases more calls are in progress at the time of reconfiguration.

There is a higher probability of two short-hop links, using the same channel, to come within each other's interference range due to the movement of $MBS$s, and the movement of $MH$s along with them. Hence, the number of channel rearrangements required during the reconfiguration time also increases and channels are available for such rearrangement. This increase in rearrangements is observed only up to a certain call arrival rate. We will refer to this threshold call arrival rate as the *knee* of the curve. Beyond the *knee*, the number of rearrangements decreases with increasing call arrival rate. We believe that this is due to two reasons:

1. At higher arrival rates several calls get dropped right in the beginning due to the non-availability of channels, thus leaving fewer short-hop links that can potentially require channel rearrangements.
2. At the time of reconfiguration, interfering short-hop links may be unable to find free channels to switch to. Thus the number of successful reconfigurations decreases. This is consistent with the increase in the number of dropped calls at higher call arrival rates, as shown in Figure 4.

Greater the size of $Spectrum_s$, higher the call arrival rate at which the *knee* is reached. For example, in the first graph of Figure 6, the knee is reached for a twenty short-hop channel system when the call arrival rate is close to 0.16. The curves for systems with larger number of short-hop channels start flattening around 0.22 to 0.24, indicating that they are reaching their respective knees.

The second graph in Figure 6 shows the impact of different reconfiguration intervals on the number of short-hop rearrangements. Given that the call duration is exponentially distributed, with mean duration of 3 minutes, if the inter-reconfiguration duration is five minutes, a large fraction of calls start and finish between two successive channel reconfigurations. Only a small fraction of calls, during the entire simulation period, experience reconfiguration. On the other hand, if the inter-reconfiguration interval is 2 minutes, most calls experience at least one reconfiguration phase, and some may experience multiple such phases during their lifetime. Hence, the high number of short-hop channel rearrangements when the inter-reconfiguration interval is 2 minutes.

**Impact on short-hop messages:** Figure 7 indicates that the smaller the inter-reconfiguration interval, the higher the number of short-hop messages per call. This is consistent with the fact that when the reconfiguration interval is small more calls may have to switch to a different frequency, and this may require exchange of messages between neighboring $MBS$s. What is more interesting is that the number of messages shows a decline as the call arrival rate increases. One would have expected the communication overhead to increase as the load increases. However, the load is not high enough for a hundred channel system to start dropping calls. So, what could be the explanation?

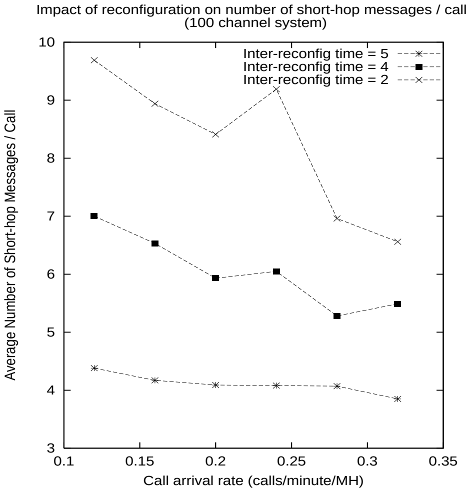

Figure 7: Impact of channel reconfigurations on number of short-hop messages.

The high communication overheads at low loads, in Figure 7, can be explained by revisiting the impact of *Allocate* sets on the algorithm's performance. An $MBS$ has to communicate with its neighboring $MBS$s to acquire a channel and add that channel to its *Allocate* set. Until the channel is deleted from the *Allocate* set it can be used to support successive calls in the same $MBS$'s cell without additional short-hop messages. Thus, the short-hop communication overhead, incurred in acquiring a channel once, gets amortized over multiple short-hop calls. However, at the time of reconfiguration, when two $MBS$s find that they have the same channel in their *Allocate* sets, at least one of them has to delete the channel from the set and incur short-hop communication overhead to acquire another channel. This happens more often when the inter-reconfiguration interval is small. Furthermore, when the call arrival rate is low, the probability of a short-hop channel in the *Allocate* set being used multiple times before being deleted from the set is further reduced. These two factors contribute towards a short inter-reconfiguration interval resulting in higher communication overhead.

Even though a short inter-reconfiguration interval is expected to improve channel reuse (which is a desirable trait, especially in situations of high load), it proves to be counter-productive in situations of low load. Moreover, a shorter inter-reconfiguration interval also increases the overall simulation effort. Hence, it is important to choose the inter-reconfiguration interval wisely.

Also, one may recall that in Section 5 we suggested that $MBS$s should add channels into their respective *Allocate* sets from the lower end of the *Spectrum*. This has been shown to improve channel reuse in situations of high load. However, in situations of low load, when two hitherto far apart $MBS$s come close enough to each other they are likely to have intersecting *Allocate* sets, necessitating some calls to switch to other channels. Had the initial assignment of channels to *Allocate* sets been done in a random fashion, rather than from the lower end of the spectrum, perhaps, the likelihood of such intersections, and the resultant channel switches by calls, could have been avoided.

*Thus, minor modifications that tend to yield better performance in high load situations result in degraded performance in low load situations, and vice versa.* A channel allocation scheme that can selectively implement various *optimizations* with varying system load will yield the best performance.

**Impact on backbone links:** The cost of setting up backbone links and their number is independent of the number of short-hop links and the short-hop call arrival rate. In our simulation the cost depends on the time between successive reconfigurations. This is because we add an extra $2st$ distance to the backbone co-channel interference range ($\beta D$) to ensure that if two backbone links do not interfere at the time of reconfiguration they also do not interfere until the next reconfiguration. Hence, greater the interval between successive reconfigurations, greater the *virtual* co-channel interference range. As backbone allocation messages have to be exchanged with all $MBS$s within the *virtual* backbone co-channel interference range, the number of such messages increases with an increase in the inter-reconfiguration interval ($t$) as shown in Figure 8.

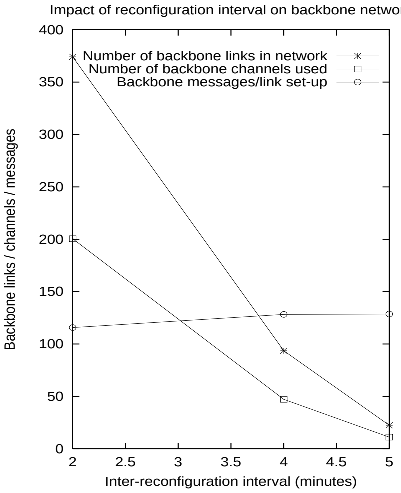

Figure 8: Impact of channel reconfigurations on backbone links.

Also, a backbone link can be established between two $MBS$s only if the distance between them is less than $D - 2st$. Therefore, as the value of $t$ increases, the number of $MBS$s pairs that are eligible for backbone links reduces. This leads to a reduction in the number of backbone links and the number of backbone channels used as shown in Figure 8.

When the reconfiguration interval is equal to 2 minutes, backbone links can be established between two $MBS$s provided they are within $D - 2st$, i.e., 1.33 kilometer of each other. Also, each participating $MBS$'s co-channel interference region is a circle of radius 6.67 kilometers and area at most 140 square kilometers. Actually, for $MBS$s that lie closer to the edges of the simulated area, the co-channel interference area is smaller. Our approximate calculations yielded the average co-channel interference region of a backbone link to be about 120 square kilometers resulting in a backbone channel reuse factor of 2. This is consistent with the simulation results where the number of backbone links is approximately double the number of backbone channels used. As the geographical area covered by the system increases, the reuse factor will also increase.

## 8 Conclusion

A mobile computing system with no fixed nodes was presented. The problem of concurrently allocating channels for backbone as well as short-hop links was formalized. Wireless channel allocation was interpreted as a variation of the mutually exclusive resource allocation problem.

A distributed dynamic channel allocation algorithm for systems with mobile base stations was presented. The algorithm built on the ideas first presented in the Ricart-Agrawala mutual exclusion solution. Disjoint sets of wireless channels were used for backbone and short-hop links. This policy was adopted to simplify the channel allocation process. The algorithm was shown to avoid co-channel interference and deadlock. Also, it imposed only modest communication overheads. The percentage of dropped calls is negligible for low and moderate load conditions. Due to node mobility, communication sessions using a particular channel without any interference may start experiencing interference at a later time. So, channel reassignment is very important to avoid co-channel interference. Our simulation experiments showed that the overheads imposed by channel rearrangement are small.

The proposed algorithm can be used, without any modification, in cellular systems with *fixed base stations*. Afterall a cellular network is an instance of the network described in this paper: (i) the speed of the base stations is zero, (ii) backbone links do not need wireless channels. The algorithm is scalable as the control is distributed among $MBS$s.

We propose to extend our work to an algorithm in which every channel can be used to support backbone and short-hop communication sessions. Such a generalization will improve the utilization of channels, but also increase the complexity of the channel allocation process. We also intend to simulate the performance of the proposed algorithm for larger geographical areas, a greater number of nodes, and a larger number of calls. Furthermore, we also intend to measure the variations in the algorithm's performance with time as $MBS$s converge towards or diverge from small regions in the system area.

### Acknowledgments

The authors thank S. Venkatesan, Kartik Chandran and all the other members of the Distributed Systems Laboratory for invaluable advice and stimulating discussions.

## References

[1] Y. Akaiwa and H. Andoh. Channel Segregation – A Self-Organized Dynamic Channel Allocation Method: Application to TDMA/FDMA Microcellular System. *IEEE Journal on Selected Areas in Communications*, 11(6):949–954, August 1993.

[2] K. M. Chandy and J. Misra. The Drinking Philosophers Problem. *ACM Transactions on Programming Languages and Systems*, 6(4):632–646, October 1984.

[3] J. C.-I. Chuang. Performance Issues and Algorithms for Dynamic Channel Assignment. *IEEE Journal on Selected Areas in Communications*, 11(6):955–963, August 1993.

[4] L. J. Cimini, Jr., G. J. Foschini, and C.-L. I. Distributed Dynamic Channel Allocation Algorithms for Microcellular Systems. In J.M. Holtzman and D.J. Goodman, editors, *Wireless Communications — Future Directions*, pages 219–241. Kluwer Academic Publishers, 1993.

[5] S. A. El-Dolil, W. C. Wong, and R. Steele. Teletraffic Performance of Highway Microcells with Overlay Macrocell. *IEEE Journal on Selected Areas in Communications*, 7:71–78, January 1989.

[6] T. J. Kahwa and N. D. Georganas. A Hybrid Channel Assignment Scheme in Large Scale, Cellular Structured Mobile Communication Systems. *IEEE Transactions on Communication*, 26(4), April 1978.

[7] L. Lamport. Time, Clocks and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(7):558–565, July 1978.

[8] W. C. Y. Lee. New Cellular Schemes for Spectral Efficiency. *IEEE Transactions on Vehicular Technology*, VT-36, November 1987.

[9] K. Madani and H. A. Aghvami. Performance of Distributed Control Channel Allocation (DCCA) Under Non-Uniform Traffic Condition in Microcellular Radio Communications. In *Proceedings of the IEEE International Conference on Communications*, pages 206–210, 1994.

[10] M. Maekawa. A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems. *ACM Transactions on Computer Systems*, pages 145–159, May 1985.

[11] R. Mathar and J. Mattfeldt. Channel Assignment in Cellular Radio Networks. *IEEE Transactions on Vehicular Technology*, 42(4):647–656, November 1993.

[12] R. Prakash. Distributed Wireless Channel Allocation in Cellular Systems with Mobile Base Stations. In *Workshop on Nomadic Computing*, Geneva, April 1997.

[13] R. Prakash and R. Baldoni. Architecture for Group Communication in Mobile Systems. In *Proceedings on the IEEE Symposium on Reliable Distributed Systems (to appear)*, October 1998.

[14] R. Prakash, N. Shivaratri, and M. Singhal. Distributed Dynamic Channel Allocation for Mobile Computing. In *Proceedings of the* $14^{th}$ *ACM Symposium on Principles of Distributed Computing*, pages 47–56, Ottawa, Canada, August 1995.

[15] S. S. Rappaport. The Multiple-Call Hand-Off Problem in High-Capacity Cellular Communications Systems. *IEEE Transactions on Vehicular Technology*, 40(3):546–557, August 1991.

[16] G. Ricart and A. K. Agrawala. An Optimal Algorithm for Mutual Exclusion in Computer Networks. *Communications of the ACM*, 24(1):9–17, January 1981.

[17] C. Shields, Jr., V. Jain, S. Ntafos, R. Prakash, and S. Venkatesan. Fault-Tolerant Mobility Planning for Rapidly Deployable Wireless Networks. In *Proceedings of the IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems*, LNCS. Springer-Verlag, April 1998.

[18] M. Singhal. A Dynamic Information-Structure Mutual Exclusion Algorithm for Distributed Systems. *IEEE Transactions on Parallel and Distributed Systems*, 3(1):121–125, January 1992.

[19] C. W. Sung and W. S. Wong. User Speed Estimation and Dynamic Channel Allocation in Hierarchical Cellular System. In *Proceedings of the* $44^{th}$ *Vehicular Technology Conference*, pages 91–95. IEEE, 1994.

[20] M. Zhang and T.-S. P. Yum. Comparisons of Channel-Assignment Strategies in Cellular Mobile Telephone Systems. *IEEE Transactions on Vehicular Technology*, 38(4):211–215, November 1989.

[21] M. Zhang and T.-S. P. Yum. The Nonuniform Compact Pattern Allocation Algorithm for Cellular Mobile Systems. *IEEE Transactions on Vehicular Technology*, 40(2):387–391, May 1991.