Channel Sharing Scheme for Packet-Switched Cellular Networks*

Suresh Kalyanasundaram† Junyi Li‡ Edwin K. P. Chong§
Ness B. Shroff ¶

September 29, 2003

Abstract

In this paper, we study an approach for sharing channels to improve network utilization in packet-switched cellular networks. Our scheme exploits unused resources in neighboring cells without the need for global coordination. We formulate a minimax approach to optimizing the allocation of channels in this sharing scheme. We develop a measurement-based distributed algorithm to achieve this objective and study its convergence. We illustrate, via simulation results, that the distributed channel sharing scheme performs significantly better than the fixed channel scheme over a wide variety of traffic conditions.

*This research was supported in part by the National Science Foundation through grants ECS-0098089, ANI-0099137, ANI-0207892, ANI-9805441, ANI-0099137, and ANI-0207728, and by an Indiana 21st century grant. A conference version of this paper appeared in INFOCOM 99. This work was done when all the authors were at Purdue University.

†Motorola India Electronics Limited, Diamond District, Airport Road, Kodihalli, Bangalore-560008, India. E-mail: Suresh.Kalyanasundaram@motorola.com.

‡Flarion Inc., USA. E-mail:j.li@flarion.comi

§Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA. E-mail: echong@engr.colostate.edu. Phone: 1-970-491-7858. Fax: 1-970-491-2249.

¶School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285, USA. E-mail: shroff@ecn.purdue.edu. Phone: 1-765-494-3471 Fax: 1-765-494-3358.

1 Introduction

The last several years has seen a tremendous growth in wireless networks all around the world. However, compared to its wired counterpart, wireless network capacity is still very scarce. Hence, improving the network utilization is a very important problem in these systems. We will address this problem in the context of packet-switched networks. The reason we consider packet-switched cellular networks is that they have the advantage of statistical multiplexing and can hence deliver a higher throughput. There has been a large-scale effort to develop multiple-access schemes for packet-switched cellular networks. Studies have shown significant improvements in throughput for the packet-switched networks over those obtained for the circuit switched case. (For example, Goodman et al. have shown in [1] that up to 1.64 simultaneous voice conversations can be supported in one channel.)

The framework that we assume throughout this paper is that of a packet-switched cellular mobile system in which each cell contains a base-station that communicates with mobile or stationary users in that cell. (If certain services, such as voice calls, use circuit-switched channels, then it merely reduces the number of channels available for packet-switched data, and the treatment that follows applies without change to those channels that are available for packet-switched data.) This communication is done over a given set of "channels." A channel can be thought of as a generic network resource such that channels used in one cell cannot be used in other cells that are closer than the *minimum reuse distance*. For example, in practical TDMA/FDMA systems, two cells that are closer than the minimum reuse distance are not allowed to use the same carrier frequency (channel in this case). A significant body of research has been conducted on dynamically allocating channels to individual cells under this minimum reuse distance constraint (e.g., see [3, 8], and the references therein) for circuit-switched networks. To simplify the complexities involved in dynamic channel allocation, some authors have proposed "channel borrowing" schemes to improve network utilization over fixed channel allocation [6, 9]. However, these schemes require either channel-locking, which necessitates global coordination, or power control. To avoid these problems, in [10, 11, 12], the authors provide a localized "channel sharing" scheme that achieves significant throughput improvements over fixed channel allocation (FCA), without incurring significant overhead. The basic idea of the channel sharing scheme is that it attempts to alleviate call blocking by sharing channels between neighboring cells, while localizing the channel

coordination.

In this paper, our objective is to extend the channel sharing scheme to packet-switched networks without the need for a central controller. We begin by defining channel sharing in the context of packet-switching. We then develop a distributed sharing scheme that optimizes a minimax objective function, in order to better utilize the network capacity. We next study the convergence of the algorithm. We then illustrate, via numerical examples, that the packet-switched sharing scheme significantly outperforms the fixed channel allocation scheme over a variety of traffic conditions. In Sections 2, 3, and 4, we describe our channel sharing scheme for a 1-D array of cells. This description, while useful in its own right to model highway cells, helps to clarify the concepts and issues involved in our sharing scheme. A similar description can be given for a 2-D array of cells as well. We do not present a detailed description of our scheme for a 2-D array of cells here. We do, however, present numerical results for the channel sharing scheme when applied to a 2-D array of cells in Section 6.

2 The Sharing Scheme for Packet-Switched Cellular Networks

In this section, we provide an efficient but simple channel allocation scheme for packet-switched cellular networks based on the "sharing scheme" proposed in [10, 11, 12]. The sharing scheme attempts to alleviate call blocking by sharing channels between neighboring cells. For illustration, we consider a linear cellular network. Each cell has a base-station that communicates with mobile users in that cell using fixed-size packets. To facilitate the description, we provide some terminology (adopted from [10]). For the sharing scheme, a *meta-cell* is defined to be a pair of neighboring cells. The two adjacent cells that together form a meta-cell are called the *component cells*. Channels are allocated to meta-cells, and these channels are allowed to be shared in any of the component cells of that meta-cell. Figure 1 shows a family of meta-cells in a linear cellular system, each comprising a pair of two adjacent cells. For example, in Figure 1, cells A and B are components of meta-cell (A, B). Note that the meta-cell is a logical notion that we use to help describe our distributed sharing algorithm. *This notion has nothing to do with hierarchical cellular schemes in the*

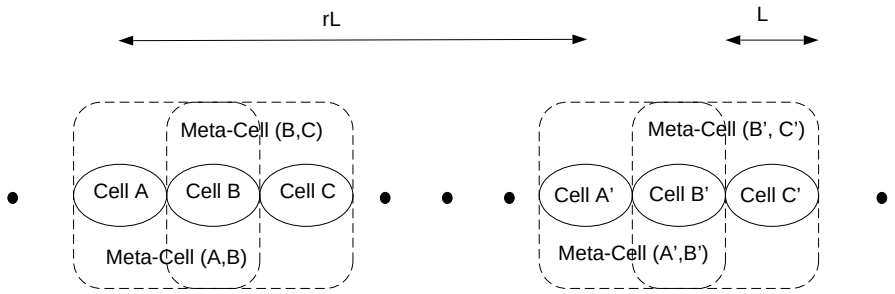


Figure 1: Linear Cellular system

literature.

Our channel allocation scheme for packet-switched cellular networks involves two main steps:

1. Fixed allocation of channels to meta-cells.

2. Intra-meta-cell channel allocation, i.e., allocation of channels to individual cells from those allocated to meta-cells.

2.1 Allocation of Channels to Meta-cells

The fixed allocation of channels to meta-cells was proposed in [10, 11, 12]. The number of channels allocated to a meta-cell is fixed and does not change with time. The channel sharing scheme allows channels to be shared between neighboring cells (namely, cells belonging to the same meta-cell). Consider Figure 1. For this simple linear cellular system, the distance measure $d(X, Y)$ between two cells X and Y is typically given as $d(X, Y) = |c_X - c_Y|$, where c_X and c_Y denote the positions of the centers of cells X and Y, respectively. The minimum reuse distance Δ is defined to be $\Delta = rL$, where L is the width of a single cell and r is an integer. Cells that are assigned the same set of channels are called co-channel cells. In the conventional scheme for fixed channel allocation, each channel is assigned to cells that are exactly a distance Δ apart. We refer to this scheme as the *tightest fixed channel assignment* scheme, in which co-channel cells are exactly r cells apart. For example, in Figure 1, cells A and A' are co-channel cells. Let T denote the total number of distinct channels that are available in this linear cellular system. Thus, the total number of distinct channels available for each cell is T/r (i.e., the reuse factor is r).

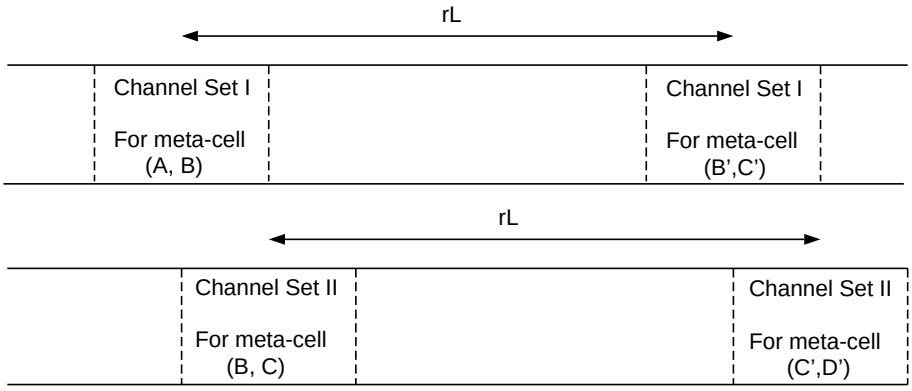


Figure 2: The same set of channels can be used in meta-cells (A, B) and (B', C') and in meta-cells (B, C) and (C', D')

In the sharing scheme, we attempt to alleviate call blocking by sharing channels between neighboring cells, while localizing the channel coordination using the meta-cell idea. As stated above, a meta-cell can be designated by the pair of its component cells. For example, in Figure 1, cells A and B are components of meta-cell (A, B).

To assign channels to meta-cells, we next define the distance measure $d((X, Y), (X', Y'))$ between two meta-cells (X, Y) and (X', Y'). Recall that in the sharing scheme, when a channel is assigned to a meta-cell, it can be used by a mobile user in any cell belonging to that meta-cell. Thus, we have to ensure that the distance measure between any component cells of two meta-cells assigned the same set of channels complies with the minimum reuse distance requirement. Consequently, we define $d((X, Y), (X', Y'))$ as the minimum of the distance measures between the component cells of meta-cells $(X, Y), (X', Y')$, i.e.,

$$d((X, Y), (X', Y')) = \min\{d(X, X'), d(X, Y'), d(Y, X'), d(Y, Y')\}. \quad (1)$$

For example, in Figure 1, the distance $d((A, B), (A', B'))$ between meta-cells (A, B) and (A', B') is $(r - 1)L$, which is equal to the distance between cells B and A'.

We call meta-cells that are assigned the same set of channels *co-channel meta-cells*. To allocate a maximum number of channels to each meta-cell, co-channel meta-cells must be deployed as close as possible. Therefore, we assign the same set of channels to meta-cells that are exactly the minimum reuse distance apart, i.e., rL in this case. For example, in Figure 2, meta-cells (A, B) and (B', C') are assigned the same set of channels (i.e., they are co-channel meta-cells). Consider a particular channel in this set. Now, when the channel is used simultaneously in meta-cells (A, B) and (B', C'), the shortest possible reuse distance is

between cells B and B', which is exactly the minimum reuse distance rL. Thus, the same channel can be independently used in cell A or B and cell B' or C'.

It is easy to see that in this channel assignment scheme, each meta-cell is assigned $T/(r+1)$ distinct channels. In other words, the reuse factor of our channel assignment scheme is $r' = r+1$. However, in the tightest fixed channel assignment scheme, the number of channels assigned to each meta-cell is T/r, so the cost we pay for allowing channels to be "shared" is $T/r - T/r' = T/r(r+1)$. Nevertheless, by increasing the reuse factor in this fashion we facilitate a simple way for channels to be shared because of which we may be able to use as many as $2T/r'$ channels in the same cell. The increase in complexity is relatively small because we do not require a central controller to allocate channels to individual calls.[1] Moreover, unlike other channel borrowing schemes, in the sharing scheme we do not need to use channel locking [13] or power control techniques [9] to share channels. Note that, in the channel sharing scheme, two cells can share channels without the need to prevent usage of those channels in other cells that are within the minimum reuse distance. The channel sharing scheme achieves this by packing channels less tightly than in the FCA scheme.

2.2 Intra-meta-cell Channel Allocation

We now discuss the allocation of channels to component cells from among those allocated to meta-cells. For the purposes of illustration, we focus on the uplink case (transmission from mobile to base-station). The algorithm and analysis that follows can be applied without any change to the downlink case as well. The allocation of channels to cells cannot take place on a per-packet basis because the relatively short packet transmission times will require adjacent base-stations to be in constant contact to determine which channels are unused. This could result in an enormous overhead. Therefore, we propose the following scheme for sharing. At the end of every U time units, we allocate channels to individual cells from those allocated to their respective meta-cells. Once the allocation is done, for the next U time units, mobiles can only use these channels for the transmission of packets to the base-station. So, this allocation has to be done in a way that will efficiently handle traffic disparities among cells.

The ability of the sharing scheme to handle the varying nature of traffic depends on the

[1] In fact, the channel sharing scheme can be thought of as a variant of the FCA scheme, in that it requires very little complexity for sharing, unlike traditional DCA schemes.

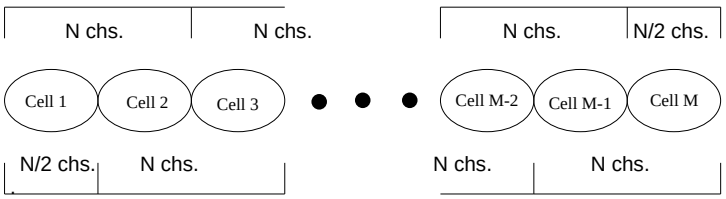


Figure 3: Number of channels allocated to different meta-cells

choice of U. There is clearly a trade-off in choosing U. The shorter the update duration U, the more efficient the sharing and the more responsive the scheme to traffic variations. On the other hand, the shorter the update duration, the greater the overhead and complexity of the coordination protocol between base-stations.

We assume that there are a total of M cells. We also assume that each meta-cell is allocated N channels using the procedure described in the previous section. We assume that the two edge cells are both assigned $N/2$ channels (as shown in Figure 3). Note that, for convenience, we have assumed that N is even.

Let us consider the time interval $[(n-1)U,\ nU]$. Let $\lambda_i(n)$ be the packet arrival rate in the i-th cell for the interval $[(n-1)U,\ nU]$. Let $x_i(n)$ be the number of channels allocated from meta-cell $(i, i+1)$ to cell i for the interval $[(n-1)U,\ nU]$. We note here that cell i, besides getting $x_i(n)$ channels from meta-cell $(i, i+1)$, also gets $N - x_{i-1}(n)$ channels from meta-cell $(i-1, i)$. The load ρ_i at cell i is given by:

$$\rho_1(n) = \frac{\lambda_1(n)}{N/2 + x_1(n)}$$

$$\rho_i(n) = \frac{\lambda_i(n)}{N - x_{i-1}(n) + x_i(n)} \text{ for } i = 2, 3, \ldots, (M-1) \tag{2}$$

$$\rho_M(n) = \frac{\lambda_M(n)}{N - x_{M-1}(n) + N/2}.$$

Allocation of channels to component cells is done based on the solution to the following optimization problem.

$$\min_{[x_1(n),\ldots,\ x_{M-1}(n)]} \max_{1 \le i \le M} \rho_i(n) \tag{3}$$
$$\text{subject to} \quad 0 \le x_i(n) \le N$$

and $x_i(n) \in \mathbb{Z}^+$ for $i \in \{1, 2, \ldots, (M-1)\}$ and for all n, where $\mathbb{Z}^+$ is the set of all non-negative integers.

The function $\rho(\cdot)$, defined in Eq. (2), is a measure of the traffic load in each cell. Given fixed arrival rates at each cell, our objective is to find values of the allocations $x_1(n), \ldots, x_{M-1}(n)$ to minimize the worst case value of ρ over all the cells. *This strategy has the effect of allocating more channels to those cells that have higher arrival rates.* It is well known that for very high values of ρ there is a dramatic non-linear increase in the average packet waiting time and the packet dropping probability. Hence allocating more channels to cells that have higher arrival rates (which has the effect of making the ρ in each cell as close to each other as possible) increases the overall utilization.

We can take several different approaches to solving the problem in Eq. (3). First, we can solve it as an integer programming problem and allocate channels based on the solution obtained. Alternatively, we can relax the requirement that $x_i(n)$ be an integer, then compute the solution and interpret the result probabilistically (as discussed in Section 5). We choose to take the latter approach because of the computational complexity involved in solving the integer programming problem. Moreover, our approach provides insights that lead to the ideas described in the distributed sharing scheme. Therefore, from now on, we will drop the integer constraints on the $x_i(n)$ in the problem in Eq. (3).

It can be shown that if the solution $\boldsymbol{x}^*(n) = [x_1^*(n), x_2^*(n), \ldots, x_{M-1}^*(n)]$ to the system of equations

$$\rho_1(n) = \rho_2(n) = \cdots = \rho_M(n) \tag{4}$$

is such that it satisfies the constraints

$$0 < x_i^*(n) < N, \ i = 1, 2, \ldots, (M-1), \tag{5}$$

then $\boldsymbol{x}^*(n)$ is the optimal solution to the minimax problem in Eq. (3) without the integer constraints. We provide a proof of this fact in the Appendix (Theorem 3). This solution to the minimax optimization problem is intuitive. We see that any further attempt to reduce $\max\{\rho_1, \rho_2, \ldots, \rho_M\}$ from the optimal solution of $\rho_1 = \rho_2 = \cdots = \rho_M$ would have to increase the number of channels allocated to each cell. Since we have already made use of all the channels, such a reduction is impossible.

For some of the results in this paper we assume that the solution to Eq. (4) satisfies the constraints in Eq. (5). This is a reasonable assumption because a few cells will need to be permanently overloaded to obtain a set of λ_i that violates this assumption. Such

permanently overloaded cells can be handled by allocating more fixed channels to the meta-cell. For transient overloaded cells, our assumption is reasonable and will almost always be satisfied.

The above solution to the minimax problem tells us that the optimal solution is obtained by making the load (ρ) in all the cells equal. This observation leads naturally to an algorithm similar to load balancing [14]. In the following, we develop a distributed scheme to allocate channels to component cells from those allocated to meta-cells based on the above solution to the minimax problem in Eq. (3).

3 Distributed Sharing Scheme

In the distributed sharing scheme, the components of each meta-cell interact among themselves and decide how to allocate the channels belonging to the meta-cell between them. This allocation is updated periodically every U time units, as described before. The distributed sharing scheme is an iterative scheme and the way the iteration proceeds is described in Section 3.2. Henceforth, we will consider only the time interval $[(n-1)U,\ nU]$ and, for simplicity, we will not explicitly display the dependence of λ, x, and ρ on n.

We describe the distributed sharing scheme in the following sections.

3.1 Allocation Update Operation

Let us consider the case when cells i and $i+1$ are interacting to decide the number of channels to be allocated to each of them from among those belonging to meta-cell $(i, i+1)$. The update

$$x_i^{new} = \frac{\lambda_i(x_{i+1}^{old} + N) - \lambda_{i+1}(N - x_{i-1}^{old})}{\lambda_i + \lambda_{i+1}} \tag{6}$$

would make $\rho_i = \rho_{i+1}$, and hence is a plausible candidate for updating the x_i. However, the updates done according to Eq. (6) does not guarantee feasibility, i.e., that x_i satisfies $0 \le x_i \le N$. To ensure feasibility, we suggest the following modification to Eq. (6):

$$x_i^{new} = \begin{cases} 0 & \text{if } \frac{\lambda_i(x_{i+1}^{old}+N)}{\lambda_{i+1}(N-x_{i-1}^{old})} \le 1 \\ N & \text{if } \frac{\lambda_i x_{i+1}^{old}}{\lambda_{i+1}(2N-x_{i-1}^{old})} \ge 1 \\ \frac{\lambda_i(x_{i+1}^{old}+N)-\lambda_{i+1}(N-x_{i-1}^{old})}{\lambda_i+\lambda_{i+1}} & \text{otherwise.} \end{cases} \tag{7}$$

Essentially, the above modification takes the projection of the solution obtained from Eq. (6) on to the interval $[0, N]$.

3.2 Interaction Mechanism

In this section, we describe how the sharing operation iterations proceed by providing three different interaction mechanisms.

3.2.1 Mechanism 1: Serial Synchronous Mechanism

At the end of each U time units, do the following:

1. Set $k := 0$, initial condition $\boldsymbol{x}^{(0)} = [x_1^{(0)}, x_2^{(0)}, \ldots, x_{M-1}^{(0)}]$.

2. If termination condition has not been reached (see Section 3.3), do the following:

   (a) Update $x_1^{(k)}, x_2^{(k)}, \ldots, x_{M-1}^{(k)}$ to $x_1^{(k+1)}, x_2^{(k+1)}, \ldots, x_{M-1}^{(k+1)}$ successively using Eq. (7).

   (b) Set $k := k + 1$, and go to step 2.

A good choice for the initial condition $\boldsymbol{x}^{(0)}$ is the final allocation for the previous U time units. In words, the algorithm works as follows: We first let cells 1 and 2 interact to decide x_1, then let cells 2 and 3 decide x_2, and so on. Once x_{M-1} has been decided, we revert back to cells 1 and 2 and the variable x_1. We continue the above process until the termination condition is satisfied. The final allocation is based on the x_i at the time the termination condition is satisfied.

3.2.2 Mechanism 2: Parallel Synchronous Mechanism

We now describe an alternative synchronous interaction mechanism. This interaction mechanism does sharing operations in parallel and hence will converge faster. The algorithm is described as follows:

At the end of each U time units, do the following:

1. Set $k := 0$, initial condition $\boldsymbol{x}^{(0)} = [x_1^{(0)}, x_2^{(0)}, \ldots, x_{M-1}^{(0)}]$.

2. If termination condition has not been reached (see Section 3.3), do the following:

(a) Update $x_1^{(k)}, x_3^{(k)}, \ldots$ to $x_1^{(k+1)}, x_3^{(k+1)}, \ldots$ simultaneously using Eq. (7). Then update $x_2^{(k)}, x_4^{(k)}, \ldots$ to $x_2^{(k+1)}, x_4^{(k+1)}, \ldots$ simultaneously using Eq. (7).

(b) Set $k := k + 1$, and go to step 2.

This interaction mechanism works as follows: Meta-cells (1,2), (3,4), (5,6), etc., together carry out an update operation. Then meta-cells (2,3), (4,5), (6,7), etc., together carry out an update operation based on the updated values from the previous update operation. Then we revert back to meta-cells (1,2), (3,4), (5,6), etc., with the updated values, and so on.

3.2.3 Mechanism 3: Asynchronous Mechanism

In the asynchronous mechanism, we do not require meta-cells to coordinate their update operations with other meta-cells. Instead, each meta-cell can perform update operations at arbitrary times and in an asynchronous fashion (see Section 5 for a discussion of possible rules for update initiation). Whenever an update is initiated in a meta-cell, say $(i, i+1)$, we update x_i according to Eq. (7). Note that we do not require a global clock to carry out this operation.

In Section 4, where we provide convergence results for the distributed algorithm, we have assumed the serial synchronous mechanism because of its notational simplicity. It is easy to see from the proofs that the same ideas carry over to the parallel synchronous interaction mechanism and the asynchronous interaction mechanism with straightforward modifications. Because the sharing operations are carried out in parallel, the parallel synchronous mechanism will converge faster to the optimal solution. The asynchronous mechanism simplifies implementation by eliminating the coordination and perhaps a global clock required with the other two synchronous mechanisms. From an implementation point of view, the asynchronous mechanism is the simplest.

3.3 Termination Condition

The termination condition determines when to stop the interaction between cells. The distributed sharing scheme converges to the optimal solution and has the *descent property*, i.e., the maximum cell load (ρ_i) at the end of each iteration is at most as large as the maximum cell load at the end of the previous iteration. (We prove these results in the next section.) Therefore, the iterations can be terminated based on any of the following criteria:

1. Terminate after a fixed number of steps.

2. Terminate after the difference between successive iterates falls below a certain threshold, i.e., terminate when

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_\infty < \epsilon_a. \tag{8}$$

3. Terminate after the relative difference between successive iterates falls below a certain threshold, i.e., terminate when

$$\frac{\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}\|_\infty}{\|\boldsymbol{x}^{(k)}\|_\infty} < \epsilon_r. \tag{9}$$

4 Analysis of the Distributed Algorithm

In this section, we show that the distributed algorithm has the descent property (discussed in Theorem 1). We also show that if the optimal solution to the minimax problem results in $\rho_1(\boldsymbol{x}^*) = \rho_2(\boldsymbol{x}^*) = \cdots = \rho_M(\boldsymbol{x}^*)$, then the distributed algorithm converges to the optimal solution. As has been mentioned earlier, to simplify notation we assume the serial synchronous interaction mechanism for our proofs. The proofs apply (with straightforward modifications) to both the parallel synchronous interaction mechanism and the asynchronous interaction mechanism.

We first consider the case of two cells, say $(i, i+1)$, with cell i having α channels from meta-cell $(i-1, i)$ and cell $i+1$ having β channels from meta-cell $(i+1, i+2)$. The update equation (7) can now be rewritten as follows:

$$x_i = \begin{cases} 0 & \text{if } \lambda_i(\beta + N) \leq \lambda_{i+1}\alpha \\ N & \text{if } \lambda_i\beta \geq \lambda_{i+1}(N + \alpha) \\ \frac{\lambda_i(\beta+N) - \lambda_{i+1}\alpha}{\lambda_i + \lambda_{i+1}} & \text{otherwise} \end{cases} \tag{10}$$

We prove that, for a fixed α and β, x_i given by Eq. (10) results in the minimax solution for the two cells under consideration, namely cells i and $(i+1)$.

Lemma 1 *The value of x_i^* given by Eq. (10) is the optimal solution to the following problem:*

$$\begin{aligned} \min_{x_i} \max \quad & \{\rho_i, \rho_{i+1}\} \\ \text{subject to} \quad & 0 \leq x_i \leq N, \end{aligned} \tag{11}$$

where

$$\rho_i = \lambda_i/(\alpha + x_i)$$

$$\rho_{i+1} = \lambda_{i+1}/(N - x_i + \beta).$$

Proof: We will consider 3 cases.

Case 1: When $0 < (\lambda_i(\beta + N) - \lambda_{i+1}\alpha)/(\lambda_i + \lambda_{i+1}) < N$ (i.e., $0 < x_i^* < N$):

This is just a special 2-cell case of the more general M-cell case proved in the Appendix (in proof of Theorem 3).

Case 2: When $\lambda_i(\beta + N) \leq \lambda_{i+1}\alpha$ (i.e., $x_i^* = 0$):

The proof is by contradiction. Assume that $x_i^* = 0$ is not the optimal solution for this case. This implies that there exists x_i' such that $0 < x_i' \leq N$ and

$$\max\left\{\frac{\lambda_i}{\alpha + x_i'}, \frac{\lambda_{i+1}}{N - x_i' + \beta}\right\} < \max\left\{\frac{\lambda_i}{\alpha}, \frac{\lambda_{i+1}}{N + \beta}\right\}. \tag{12}$$

But we know that $\lambda_i/\alpha \leq \lambda_{i+1}/(\beta + N)$. Therefore, we have that

$$\frac{\lambda_{i+1}}{N - x_i' + \beta} < \frac{\lambda_{i+1}}{N + \beta}. \tag{13}$$

But the above equation implies that $x_i' < 0$, which is a contradiction. Therefore, we have $x_i^* = 0$ as the optimal solution for this case.

Case 3: When $\lambda_i\beta \geq \lambda_{i+1}(N + \alpha)$ (i.e., $x_i^* = N$):

For the sake of contradiction assume that $x_i^* = N$ is not the optimal solution for this case. This implies that there exists x_i' such that $0 \leq x_i' < N$ and

$$\max\left\{\frac{\lambda_i}{\alpha + x_i'}, \frac{\lambda_{i+1}}{N - x_i' + \beta}\right\} < \max\left\{\frac{\lambda_i}{\alpha + N}, \frac{\lambda_{i+1}}{\beta}\right\}. \tag{14}$$

But we know that $\lambda_i/(\alpha + N) \geq \lambda_{i+1}/\beta$. Therefore, $\lambda_i/(\alpha + x_i') < \lambda_i/(\alpha + N)$, which implies that $x_i' > N$, and we have obtained a contradiction. Hence, we have $x_i^* = N$ as the optimal solution for this case.

These three cases show that x_i^*, as given by Eq. (10), is the optimal solution to the problem defined in Eq. (11). □

We now prove that the distributed algorithm has the descent property by making use of Lemma 1.

Theorem 1 *The distributed algorithm has the following descent property: If* $\{\boldsymbol{x}^{(k)}\}$ *is the sequence of iterates obtained from the distributed algorithm, and if* $\{\rho_i^{(k)}\}$ *are the function values obtained using Eq. (2), then we have:*

$$\max_{1\leq i\leq M} \rho_i^{(k)} \leq \max_{1\leq i\leq M} \rho_i^{(k-1)}. \tag{15}$$

Proof: Consider $\max_{1\leq i\leq M} \rho_i$ before and after updating $x_i^{(k-1)}$ to $x_i^{(k)}$. Because of Lemma 1, we have

$$\max\left\{\rho_i(x_{i-1}^{(k)}, x_i^{(k)}), \rho_{i+1}(x_i^{(k)}, x_{i+1}^{(k-1)})\right\} \leq \max\left\{\rho_i(x_{i-1}^{(k)}, x_i^{(k-1)}), \rho_{i+1}(x_i^{(k-1)}, x_{i+1}^{(k-1)})\right\}. \tag{16}$$

The above observation along with the fact that updating $x_i^{(k-1)}$ to $x_i^{(k)}$ will affect only ρ_i and ρ_{i+1} (since we update one variable at a time) establishes descent. □

If the optimal solution $\boldsymbol{x}^*$ satisfies $0 < x_i^* < N$, then $\boldsymbol{x}^*$ is the only point that will remain unchanged under the distributed algorithm iterations. We prove this result in the following proposition.

Proposition 1 *If the point* $\boldsymbol{x}^* = [x_1^*, x_2^*, \ldots, x_{M-1}^*]$ *is such that* $\rho_1^* = \cdots = \rho_M^*$ *and*

$$0 < x_i^* < N,\ i = 1, 2, \ldots, (M-1), \tag{17}$$

then $\boldsymbol{x}^*$ *is the only fixed point of the distributed algorithm iteration.*

Proof: It is clear that the given point $\boldsymbol{x}^*$ is a fixed point, since any more iterations will leave $\boldsymbol{x}^*$ unchanged. (Fixed points are points that do not change under the update operation of the algorithm.) We next consider all other points that are fixed points of the iteration, and show that these cannot occur under the assumption in Eq. (17). Although there are many possible fixed points of the algorithm, there are three important cases, which we consider below case-by-case. We use the ∞ superscript to denote those points that are fixed points of the iteration.

Case 1: Consider the point $\boldsymbol{x}^{(\infty)}$ satisfying $\rho_1^{(\infty)} = \rho_2^{(\infty)} = \cdots = \rho_i^{(\infty)}$ and $\rho_{i+1}^{(\infty)} = \rho_{i+2}^{(\infty)} = \cdots = \rho_M^{(\infty)}$, $\rho_i^{(\infty)} > \rho_{i+1}^{(\infty)}$ and $x_i^{(\infty)} = N$. It is easy to see that any iteration of the algorithm will not alter $\boldsymbol{x}^{(\infty)}$ (it is therefore a fixed point). We now show that this situation cannot happen under the assumption given in Eq. (17).

Because ρ_i^* is the optimal minimax solution, we have

$$\rho_j^* \leq \rho_j^{(\infty)}, \text{ for } j = 1, 2, \ldots, i. \tag{18}$$

This implies that

$$\begin{aligned}
\frac{\lambda_1}{N/2 + x_1^*} &\leq \frac{\lambda_1}{N/2 + x_1^{(\infty)}} \\
\frac{\lambda_2}{N - x_1^* + x_2^*} &\leq \frac{\lambda_2}{N - x_1^{(\infty)} + x_2^{(\infty)}} \\
&\vdots \\
\frac{\lambda_i}{N - x_{i-1}^* + x_i^*} &\leq \frac{\lambda_i}{N - x_{i-1}^{(\infty)} + N}.
\end{aligned} \tag{19}$$

From the above equations we have

$$\begin{aligned}
x_1^* &\geq x_1^{(\infty)} \\
x_2^* - x_1^* &\geq x_2^{(\infty)} - x_1^{(\infty)} \\
&\vdots \\
x_{i-1}^* - x_{i-2}^* &\geq x_{i-1}^{(\infty)} - x_{i-2}^{(\infty)} \\
x_i^* - x_{i-1}^* &\geq N - x_{i-1}^{(\infty)}.
\end{aligned} \tag{20}$$

Adding the above equations, we have that $x_i^* \geq N$, which contradicts Eq. (17).

Case 2: Consider a point $\boldsymbol{x}^{(\infty)}$ satisfying $\rho_1^{(\infty)} = \rho_2^{(\infty)} = \cdots = \rho_i^{(\infty)}$ and $\rho_{i+1}^{(\infty)} = \rho_{i+2}^{(\infty)} = \cdots = \rho_M^{(\infty)}$, $\rho_i^{(\infty)} < \rho_{i+1}^{(\infty)}$ and $x_i^{(\infty)} = 0$. Then, clearly, this point is also a fixed point of the iteration. We now show that this situation also cannot happen under the assumption given in Eq. (17). Since ρ_i^* is the optimal minimax solution, we have

$$\rho_j^* \leq \rho_j^{(\infty)}, \text{ for } j = i+1, i+2, \ldots, M. \tag{21}$$

This implies that

$$\begin{aligned}
\frac{\lambda_{i+1}}{N - x_i^* + x_{i+1}^*} &\leq \frac{\lambda_{i+1}}{N + x_{i+1}^{(\infty)}} \\
\frac{\lambda_{i+2}}{N - x_{i+1}^* + x_{i+2}^*} &\leq \frac{\lambda_{i+2}}{N - x_{i+1}^{(\infty)} + x_{i+2}^{(\infty)}} \\
&\vdots \\
\frac{\lambda_M}{3N/2 - x_{M-1}^*} &\leq \frac{\lambda_M}{3N/2 - x_{M-1}^{(\infty)}}.
\end{aligned} \tag{22}$$

From the above equations we have

$$\begin{aligned} x^*_{i+1} - x^*_i &\geq x^{(\infty)}_{i+1} \\ x^*_{i+2} - x^*_{i+1} &\geq x^{(\infty)}_{i+2} - x^{(\infty)}_{i+1} \\ &\vdots \\ x^*_{M-1} - x^*_{M-2} &\geq x^{(\infty)}_{M-1} - x^{(\infty)}_{M-2} \\ -x^*_{M-1} &\geq -x^{(\infty)}_{M-1}. \end{aligned} \tag{23}$$

Adding the above equations, we have that $x^*_i \leq 0$, which contradicts Eq. (17).

Case 3: Consider a point $\boldsymbol{x}^{(\infty)}$ satisfying $\rho^{(\infty)}_1 = \cdots = \rho^{(\infty)}_i$, $\rho^{(\infty)}_{i+1} = \cdots = \rho^{(\infty)}_j$, $\rho^{(\infty)}_{j+1} = \cdots = \rho^{(\infty)}_M$, with $\rho^{(\infty)}_i < \rho^{(\infty)}_{i+1}$, $\rho^{(\infty)}_j > \rho^{(\infty)}_{j+1}$, $x^{(\infty)}_i = 0$, and $x^{(\infty)}_j = N$. Then, clearly, this point is also a fixed point of the iteration. We now show that this situation also cannot happen under the assumption given in Eq. (17). Since ρ^*_i is the optimal minimax solution, we have

$$\rho^*_k \leq \rho^{(\infty)}_k, \text{ for } k = i+1, \ldots, j. \tag{24}$$

This implies that

$$\begin{aligned} \frac{\lambda_{i+1}}{N - x^*_i + x^*_{i+1}} &\leq \frac{\lambda_{i+1}}{N + x^{(\infty)}_{i+1}} \\ \frac{\lambda_{i+2}}{N - x^*_{i+1} + x^*_{i+2}} &\leq \frac{\lambda_{i+2}}{N - x^{(\infty)}_{i+1} + x^{(\infty)}_{i+2}} \\ &\vdots \\ \frac{\lambda_j}{N - x^*_{j-1} + x^*_j} &\leq \frac{\lambda_j}{2N - x^{(\infty)}_{j-1}}. \end{aligned} \tag{25}$$

The above implies that

$$\begin{aligned} x^*_{i+1} - x^*_i &\geq x^{(\infty)}_{i+1} \\ x^*_{i+2} - x^*_{i+1} &\geq x^{(\infty)}_{i+2} - x^{(\infty)}_{i+1} \\ &\vdots \\ x^*_j - x^*_{j-1} &\geq N - x^{(\infty)}_{j-1}. \end{aligned} \tag{26}$$

Adding the above equations, we have that $x^*_j - x^*_i \geq N$, which contradicts Eq. (17).

Any other fixed point is similar to one of the cases above, and hence will lead to contradictions similar to one of the three cases above. This is because a fixed point has to be such that for each cell i one of the following statements must be true of both its neighbors:

1. The neighbor has the same cell load ρ.

2. If a neighbor (say cell $i+1$) has a higher cell load, then cell $i+1$ has all the channels from the meta-cell $(i, i+1)$.

3. If a neighbor (say cell $i+1$) has a smaller cell load, then cell i has all the channels from meta-cell $(i, i+1)$.

Therefore, the given $\boldsymbol{x}^*$ is the only fixed point of the iteration. □

We next prove a lemma that we will use to show the convergence of the distributed algorithm.

Lemma 2 *If the optimal solution* $\boldsymbol{x}^* = [x_1^*, x_2^*, \ldots, x_{M-1}^*]$ *is such that*

$$\rho_1^* = \rho_2^* = \cdots = \rho_M^* \tag{27}$$

then the distributed algorithm has the following property:

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|_\infty \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty. \tag{28}$$

Proof: Note that because $\boldsymbol{x}^*$ is a fixed point of the iteration and it satisfies the condition given in Eq. (27),

$$\begin{aligned} x_1^* &= \frac{\lambda_1(x_2^* + N) - \lambda_2 N/2}{\lambda_1 + \lambda_2} \\ x_2^* &= \frac{\lambda_2(x_3^* + N) - \lambda_3(N - x_1^*)}{\lambda_2 + \lambda_3} \\ &\vdots \\ x_{M-1}^* &= \frac{\lambda_{M-1}(N/2 + N) - \lambda_M(N - x_{M-2}^*)}{\lambda_{M-1} + \lambda_M}. \end{aligned} \tag{29}$$

Also, if we let

$$\begin{aligned} x_1^{(k+1)} &= \frac{\lambda_1(x_2^{(k)} + N) - \lambda_2 N/2}{\lambda_1 + \lambda_2} \\ x_2^{(k+1)} &= \frac{\lambda_2(x_3^{(k)} + N) - \lambda_3(N - x_1^{(k+1)})}{\lambda_2 + \lambda_3} \\ &\vdots \\ x_{M-1}^{(k+1)} &= \frac{\lambda_{M-1}(N/2 + N) - \lambda_M(N - x_{M-2}^{(k+1)})}{\lambda_{M-1} + \lambda_M}, \end{aligned} \tag{30}$$

then we have that

$$\begin{aligned} |x_1^{(k+1)} - x_1^*| &= \frac{\lambda_1 |x_2^{(k)} - x_2^*|}{\lambda_1 + \lambda_2} \\ |x_2^{(k+1)} - x_2^*| &\leq \frac{\lambda_2 |x_3^{(k)} - x_3^*|}{\lambda_2 + \lambda_3} + \frac{\lambda_3 |x_1^{(k+1)} - x_1^*|}{\lambda_2 + \lambda_3} \\ &\vdots \\ |x_{M-1}^{(k+1)} - x_{M-1}^*| &= \frac{\lambda_M |x_{M-2}^{(k+1)} - x_{M-2}^*|}{\lambda_{M-1} + \lambda_M}. \end{aligned} \tag{31}$$

Now, if we allow projections as in Eq. (7), then

$$\begin{aligned} |x_1^{(k+1)} - x_1^*| &\leq \frac{\lambda_1 |x_2^{(k)} - x_2^*|}{\lambda_1 + \lambda_2} \\ |x_2^{(k+1)} - x_2^*| &\leq \frac{\lambda_2 |x_3^{(k)} - x_3^*|}{\lambda_2 + \lambda_3} + \frac{\lambda_3 |x_1^{(k+1)} - x_1^*|}{\lambda_2 + \lambda_3} \\ &\vdots \\ |x_{M-1}^{(k+1)} - x_{M-1}^*| &\leq \frac{\lambda_M |x_{M-2}^{(k+1)} - x_{M-2}^*|}{\lambda_{M-1} + \lambda_M}, \end{aligned} \tag{32}$$

because $0 \leq x_i^* \leq N$, for $i = 1, 2, \ldots, (M-1)$. Further, since $|x_i^{(k)} - x_i^*| \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty$, $i = 1, 2, \ldots, (M-1)$, it follows that

$$|x_i^{(k+1)} - x_i^*| \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty, \quad i = 1, 2, \ldots, (M-1).$$

Therefore,

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|_\infty \leq \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty.$$

□

We next prove that the distributed algorithm iterates converge to the optimal solution.

Theorem 2 *Let $\{\boldsymbol{x}^{(k)}\}$ be the sequence of iterates obtained from the distributed algorithm as described above, and let the optimal solution $\boldsymbol{x}^*$ satisfy the assumption in Eq. (27). Then $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*$.*

Proof: Write $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty = \epsilon_0$. Define

$$\begin{aligned} t_1 &= \frac{\lambda_1}{\lambda_1 + \lambda_2} \\ t_i &= \frac{\lambda_i + \lambda_{i+1} t_{i-1}}{\lambda_i + \lambda_{i+1}}, \qquad i = 2, \ldots, M-2 \\ t_{M-1} &= \frac{\lambda_M t_{M-2}}{\lambda_{M-1} + \lambda_M}. \end{aligned}$$

From Eq. (32) we have that

$$\begin{aligned}
|x_1^{(k+1)} - x_1^*| &\leq \frac{\lambda_1 |x_2^{(k)} - x_2^*|}{\lambda_1 + \lambda_2} \leq \frac{\lambda_1 \epsilon_0}{\lambda_1 + \lambda_2} = t_1 \epsilon_0 \\
|x_2^{(k+1)} - x_2^*| &\leq \frac{\lambda_2 \epsilon_0}{\lambda_2 + \lambda_3} + \frac{\lambda_3}{\lambda_2 + \lambda_3} t_1 \epsilon_0 = t_2 \epsilon_0 \\
|x_3^{(k+1)} - x_3^*| &\leq \frac{\lambda_3 \epsilon_0}{\lambda_3 + \lambda_4} + \frac{\lambda_4}{\lambda_3 + \lambda_4} t_2 \epsilon_0 = t_3 \epsilon_0 \\
&\vdots \\
|x_{M-2}^{(k+1)} - x_{M-2}^*| &\leq \frac{\lambda_{M-2} \epsilon_0}{\lambda_{M-2} + \lambda_{M-1}} + \frac{\lambda_{M-1}}{\lambda_{M-2} + \lambda_{M-1}} t_{M-3} \epsilon_0 = t_{M-2} \epsilon_0 \\
|x_{M-1}^{(k+1)} - x_{M-1}^*| &\leq \frac{\lambda_M}{\lambda_{M-1} + \lambda_M} t_{M-2} \epsilon_0 = t_{M-1} \epsilon_0.
\end{aligned} \tag{33}$$

After the $(k+1)$-st iteration we have

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|_\infty \leq \max\left\{t_1 \epsilon_0, t_2 \epsilon_0, \ldots, t_{M-1} \epsilon_0\right\}.$$

Because $t_{M-2}\epsilon_0$ is the maximum among all the terms, it follows that

$$\|\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*\|_\infty \leq t_{M-2} \epsilon_0. \tag{34}$$

Since $\lambda_i > 0$, for $i = 1, 2, \ldots, M$, it follows that $t_{M-2} < 1$. Additionally, t_{M-2} is independent of k. Therefore,

$$\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_\infty \leq (t_{M-2})^k \|\boldsymbol{x}^{(0)} - \boldsymbol{x}^*\|_\infty \tag{35}$$

and since $t_{M-2} < 1$, we have

$$\boldsymbol{x}^{(k)} \to \boldsymbol{x}^*. \tag{36}$$

□

The serial distributed algorithm, as described above, has an appealing geometric interpretation. We note that when the optimal minimax solution satisfies Eq. (27), it is just the solution to the following system of equations:

$$\begin{aligned}
-(\lambda_1 + \lambda_2) x_1 + \lambda_1 x_2 &= \lambda_2 N/2 - \lambda_1 N \\
\lambda_3 x_1 - (\lambda_2 + \lambda_3) x_2 + \lambda_2 x_3 &= \lambda_3 N - \lambda_2 N \\
&\vdots \\
\lambda_M x_{M-2} - (\lambda_{M-1} + \lambda_M) x_{M-1} &= \lambda_M N - 3N\lambda_{M-1}/2.
\end{aligned} \tag{37}$$

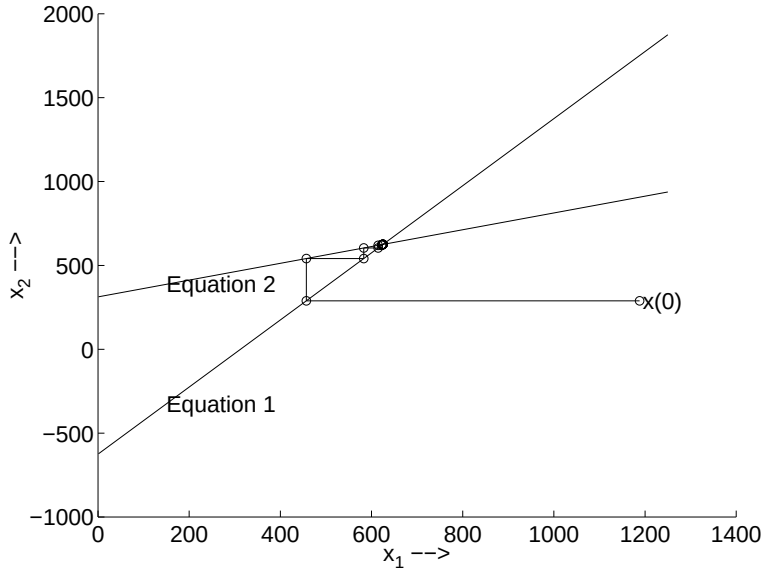


Figure 4: Convergence of the serial distributed algorithm

Given an initial $\boldsymbol{x}^{(0)}$, fixing $x_2^{(0)}$, we find an $x_1^{(1)}$ that satisfies the first of the above equations. Then, fixing $x_1^{(1)}$ and $x_3^{(0)}$, we find an $x_2^{(1)}$ that satisfies the second equation above, and so on. In general, fixing $x_{i-1}^{(k+1)}$ and $x_{i+1}^{(k)}$, we find $x_i^{(k+1)}$ by solving the i-th equation above. In Figure 4, we illustrate how this scheme converges to the optimal solution for the 3-cell case. Note that, while Eq. (37) can be solved for the x_i in terms of the λ_i and stored in each base-station, the implementation would have to be centralized because each base-station would need knowledge of all the λ_i to determine its x_i.

5 Discussion

In this section, we discuss some issues related to our scheme.

1. We note that usually we will not know the exact value of the packet arrival rate $\lambda_i(n)$ beforehand. Therefore, we may need to estimate it. One possibility is simply to use the measured packet arrival rate over the last L time units. There is a trade-off in choosing L. If we choose L large, then our sharing algorithm will not react well to a change in the packet arrival rate that has just begun. On the other hand, if we choose L small, then the sharing algorithm would react to even small transient changes in the packet arrival rate. In the simulation, presented in the next section, we have chosen $L = U$.

2. We have to terminate our algorithm after a finite number of iterations. The longer we run the iterations the more balanced the load is over all the cells. But each iteration increases the overhead because we require adjacent base-stations to inform each other of the latest value of their respective $x_i^{(k)}$. In our simulation results described in the next section, we only allowed one iteration. Even so, the performance, as shown in Section 6, is quite good.

3. In our algorithm, we have assumed that the allocation is updated periodically. However, alternative update rules are possible. For example, an update can be initiated by an overloaded cell or by an underloaded cell. Specifically, when an overloaded cell realizes that its load (ρ) is higher than a certain maximum threshold, it can initiate an update. Similarly, when the ρ value in an underloaded cell drops below a certain lower threshold, it can initiate an update. Alternatively, a cell can initiate an update when its ρ value rises or drops by more than a certain percentage since the last operation.

4. We have shown the convergence of the distributed algorithm to the optimal solution only when the optimal solution satisfies Eq. (27). But, as we have noted before, our distributed algorithm has the descent property irrespective of whether or not the optimal solution satisfies Eq. (27). From a practical viewpoint, we can terminate the algorithm after just a few iterations and still expect the performance gain to be substantial. For example, as mentioned earlier, in the simulation results presented in the next section, we terminate the iterations after exactly one complete cycle and still obtain fairly significant gains over the FCA scheme.

5. The solution we have given assumes that the x_i are all real, while, in practice, they have to be integers. We could arbitrarily round off the x_i or take the floor or ceiling to satisfy the integer requirements. Alternatively, we suggest the following probabilistic method of handling the integer constraints. The number of channels we allocate to cell i from meta-cell $(i, i+1)$ can be decided as follows: Allocate $\lfloor x_i \rfloor$ channels to cell i with a probability of $\lceil x_i \rceil - x_i$ and allocate $\lceil x_i \rceil$ channels with a probability of $x_i - \lfloor x_i \rfloor$. The above method ensures that the expected value of the number of channels allocated to cell i is x_i.

6 Numerical Results

In this section we present simulation results. We will consider two types of systems: linear or one-dimensional cellular system; and a two-dimensional hexagonal cellular system. We also consider the impact of the granularity of the minimum number of channels that can be shared on the performance of our scheme..

6.1 1-D Case

We consider a linear cellular system of 30 cells with the end cells assumed connected to avoid edge effects. We assume packet arrivals in each cell to be Poisson and packet sizes to be fixed at 576 bits/packet. Note that we make the Poisson packet arrival assumption only for the aggregate traffic from all the users in one cell. The objective is to observe the performance of the sharing scheme in the presence of traffic non-uniformities. Towards this end, we fix the packet arrival rate in all cells, except cell 15, at 500 packets/sec. We vary the packet arrival rate in cell 15 from 500 packets/sec to 2900 packets/sec and make various observations. While we consider the case of the overloaded cell (namely, cell 15) remaining the same in the simulation, it should be clear from the dynamic nature of the intra-meta-cell channel allocation that the channel sharing scheme can perform well in scenarios where the overloaded cell varies with time. If the same cell remains permanently overloaded, then such static schemes as deploying micro or pico cells will work. However, such schemes will not perform well under dynamically changing overloaded cells.

In the fixed channel allocation scheme, each cell is allocated a carrier that can handle 1250 packets/sec. We assume a frame duration of 0.016 seconds, and 20 slots per frame duration. Thus, when a given slot in a particular frame is allocated to a cell under the channel sharing scheme, the same slot in all subsequent frames are also allocated to the same cell until a reallocation of that slot occurs. To make a fair comparison, we assume that for all reuse factors, T/r is fixed, where T is the total number of distinct channels available. That is, we assume that the number of channels allocated to a cell in the FCA scheme is fixed regardless of the reuse factor. In the simulations, we choose $U = 0.48$ seconds. We terminate the distributed sharing algorithm (serial synchronous interaction mechanism) after exactly one iteration, and, as the results show, we obtain fairly significant gains over the FCA scheme. Since we need just one iteration to get satisfactory results, the overhead required

	U = .16 s	U = .48 s	U = 1.6 s
reuse = 2	1.48%	.498%	.15%
reuse = 3	1.32%	.442%	.133%
reuse = 4	1.23%	.415%	.125%

Table 1: Fraction of overhead traffic for updates

to implement the sharing scheme is minimal (as can be seen from Table 1). The simulations use the estimated packet arrival rates and not the exact packet arrival rates. We leave the packet arrival rate in the congested cell as a parameter. We normalize the packet arrival rates using the packet arrival rate in the other cells, i.e., 500 packets/sec.

In Table 1, we compare the percentage overhead due to information exchange when we stop the algorithm after one iteration for various update durations and reuse factors. Whenever the cells belonging to a meta-cell interact, we assume that two packets will be required. For example, when x_i is being updated, one packet each would be required for cell i first to obtain information about the latest value of x_{i+1} and then to inform cell $i+1$ about the updated value of x_i. We note here that these overhead packets could also be transmitted over the wired network interconnecting the base-stations (if such a wired network exists, which is typically the case).

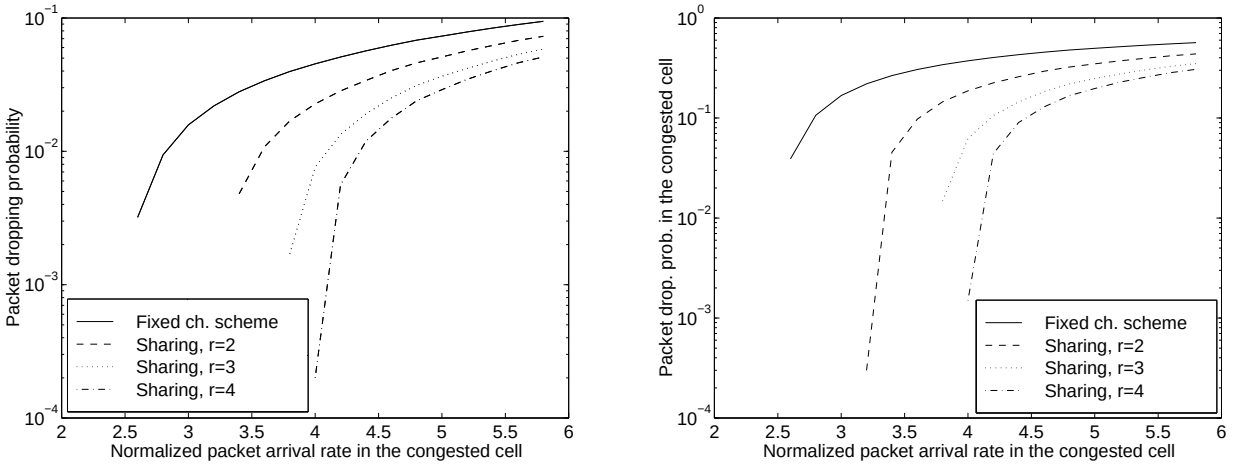
In Figures 5(a) and 5(b), we compare the packet dropping probabilities of the sharing scheme and the fixed channel allocation scheme for various reuse factors. In Figure 5(a), we plot the overall packet dropping probabilities (averaged across all cells), and in Figure 5(b), the comparison is between the packet dropping probabilities in the most congested cell (namely, cell 15). We assume that all packets have strict delay requirements and that a packet that waits for more than 200 packet-transmission periods gets dropped. We employ this scenario so that the improvement obtained is only due to the channel sharing scheme, and we can avoid complicated scheduling disciplines. Packets are transmitted on a first-come-first-served basis. In the simulations, we assume that the base-station has knowledge of the states of individual queues and schedules uplink traffic using a control channel. Therefore, there is no loss of capacity due to contention for the channel among different users. We see that the sharing scheme performs consistently better than the fixed channel scheme. For example, for an overall packet dropping probability of 0.01, while we can support a

normalized packet arrival rate of only 2.8 in the fixed channel scheme, we can support a normalized packet arrival rate of up to 4.8 with the sharing scheme for a reuse factor of 4. Similarly, for a typical value of packet dropping probability in the congested cell of 0.1, the improvement in the packet arrival rate that can be supported is 29% if the reuse factor is 2, and as much as 61% if the reuse factor is 4. As expected, we also find that the higher the reuse factor, the better the performance. This improvement is due to the fact that the difference between the number of channels allocated to cells in the FCA scheme over the channel sharing scheme reduces on the order of $1/r^2$ $(T/r - T/(r+1))$. We know that a meta-cell receives $(T/r)r/(r+1)$ channels. Therefore, for fixed T/r, the number of channels allocated to a meta-cell increases with r.

In Figures 6(a) and 6(b), we compare the waiting time of a packet before it gets transmitted to the base-station. In Figure 6(a), the overall average waiting time (averaged over all cells) for a packet is compared. In Figure 6(b), we plot the average waiting time for a packet in the congested cell. In these figures we also plot the 99% confidence intervals. We obtain substantial performance improvements using the sharing scheme. For example, for an overall average waiting time of 0.3 seconds, while the fixed channel scheme can support a normalized packet arrival rate of only 3.3, the sharing scheme can support a (normalized) packet arrival rate of about 4.1 if the reuse factor is 2, and about 5.3 if the reuse factor is 4. These numbers represent improvements of about 24% and 61%, respectively. Similarly, for an average waiting time of 2 seconds in the congested cell, the improvements we obtain for the sharing scheme over the fixed channel scheme is about 22% if the reuse factor is 2, 43% if the reuse factor is 3, and 62% if the reuse factor is 4.

In Figure 7, we compare the percentage utilization of the sharing scheme with that of the FCA scheme. The plots have been normalized with respect to T/r, the total capacity available to a cell under the FCA scheme.

Next, we consider the situation where the minimum number of slots that can be shared is more than one. Practical considerations might force the operator of the system to require that the minimum granularity of the number of channels allocated to a particular cell be a multiple of a certain minimum quantity δ $(\delta \geq 1)$. In our above simulation example, this translates into the requirement that the number of slots allocated to any cell be a multiple of δ $(\delta \geq 1)$ slots in each frame. In Figure 8, we study the impact of this requirement on the packet dropping probability. We compare the packet dropping probability for the case where



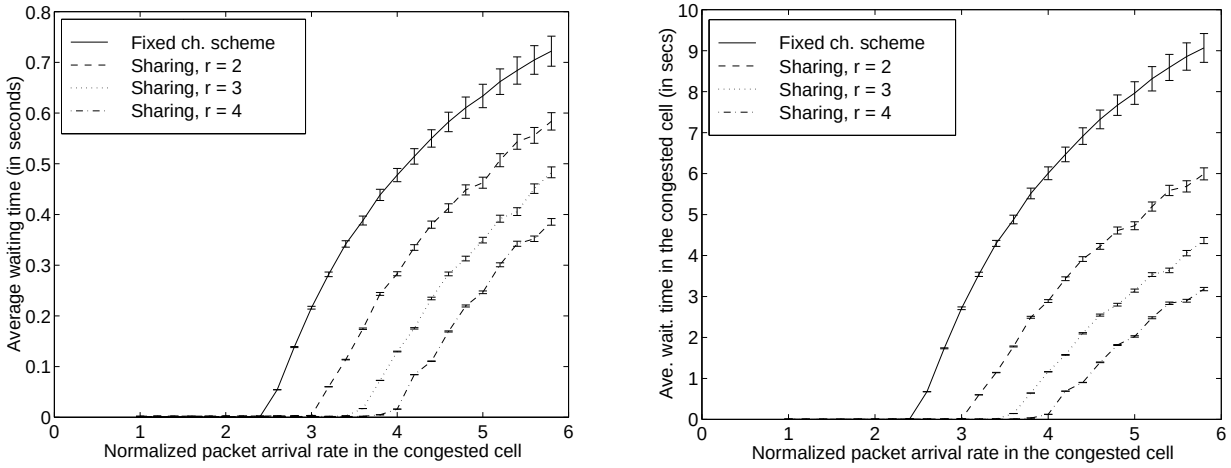
(a) (b)

Figure 5: (a) Comparison of the overall packet dropping probabilities (b) Comparison of packet dropping probabilities in the congested cell

$\delta = 3$ with the case of $\delta = 1$ and with the fixed channel scheme. In this example, we fix the reuse factor at 4. The other parameters are the same as in the earlier simulation. From the figure we conclude that for sufficiently small values of the δ parameter, the channel sharing scheme still yields significant improvements when compared to the fixed channel scheme. Our motivation to consider this situation is to simulate the case where entire carriers have to be allocated to cells instead of single slots. Our investigation here shows that so long as the number of carriers allocated to cells is fairly large, then even under the requirement that entire carriers be allocated to cells instead of single channels, the channel sharing scheme has significant performance gains.

6.2 2-D Case

The results for the one-dimensional case are of importance in its own right, especially in modeling highway cells. But for other systems, it is necessary to extend the channel sharing idea to a 2-D array of cells. In this section, we apply our channel sharing algorithm to a 2-D array of cells, and present numerical results from our simulation. For a 2-D array of cells we denote the reuse factor by R. Meta-cells are pairs of adjacent cells as before. Typical values for the reuse factor in a 2-D hexagonal array of cells are 3, 7, and 12. We consider these three different cases for our simulation. Because we allow sharing of channels between meta-cells we can no longer have the same reuse factor. While the reuse factor increases



(a) (b)

Figure 6: (a) Comparison of the overall average waiting time with 99% confidence intervals (b) Comparison of average waiting time in the congested cell with 99% confidence intervals

from r to $r+1$ in a linear cellular system, the reuse factor increases from R to R' for a 2-D array of hexagonal cells. The way R' is derived from R is described in detail in [11]. For instance, it has been shown that the R' values corresponding to R values of 3, 7, and 12 are 5, 10, and 16, respectively. In Figure 9, we illustrate the reuse scenario for $R = 7$ for a 2-D array of hexagonal cells. The entire set of available channels are used in each group of seven cells shown in bold. To satisfy the minimum reuse distance constraint, the same set of channels get reused only in the corresponding cell in the other reuse groups. For example, the same set of channels are reused in cells A, A', and A''.

The distributed sharing algorithm for a 2-D array of cells will be very similar to the scheme described for a linear cellular system. In the 2-D case, as in the 1-D case, the descent property of the distributed sharing algorithm holds. The proof is similar to the one described for the linear case (Theorem 1). To avoid repetition, we omit the details here. The simulation results show that the descent property of our algorithm is sufficient to guarantee satisfactory performance.

For the simulation, we assume a 2-D array of 37 cells with the cells being assumed to be connected, i.e., we assume that each cell has six neighbors. This assumption has been made to avoid the edge effects. We assume that the packet arrival rate of all except one cell is fixed at 600 packets/sec. We leave the packet arrival rate of that one cell to be a parameter of the simulation. We assume periodic updates of our sharing algorithm, with the period

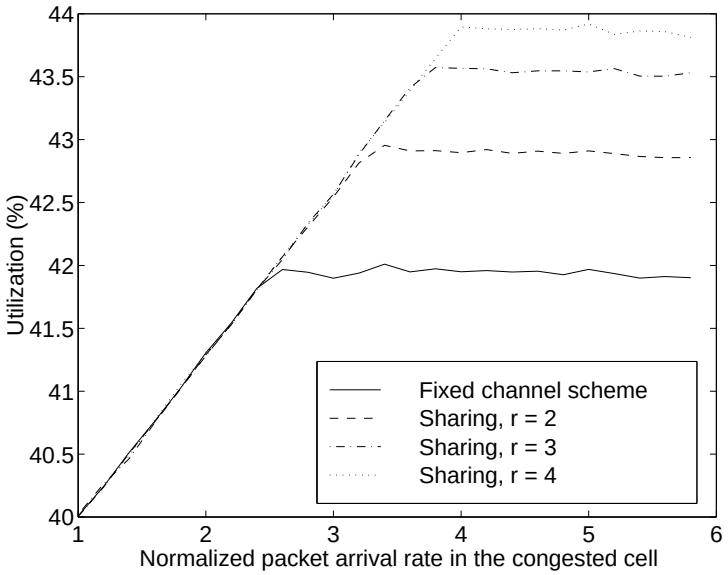


Figure 7: Utilization curves for the sharing and the FCA schemes

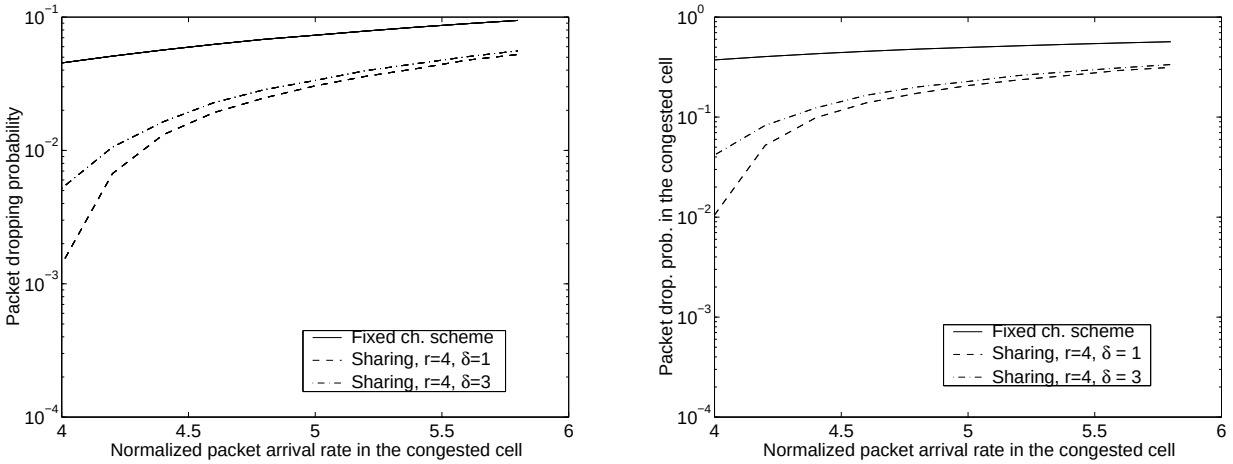
being one second. As in the 1-D case, we normalize the packet arrival rate of the congested cell using the packet arrival rate in the other cells, i.e., 600 packets/sec.

In Figure 10, we compare the packet dropping probability for the FCA scheme and our channel sharing scheme for various reuse factors. We find that our channel sharing scheme performs better than the FCA scheme for all reuse factors. As expected, we also find that the performance improves for higher reuse factors. For example, for a normalized packet arrival rate of 3.5, the performance improvement in the overall packet dropping probability is 73% for a reuse factor of 3, while the improvement for a reuse factor of 12 is 88%. Similar improvements are seen for the packet dropping probability in the congested cell.

In Figures 11(a) and 11(b), we plot the overall average waiting time and the average waiting time in the congested cell, respectively. Again, our channel sharing scheme achieves significant gains when compared to the traditional FCA scheme. In Figure 12, we plot the percentage of the total capacity utilized by the channel sharing and the FCA schemes.

7 Conclusions

We have developed a *channel sharing algorithm* for packet-switched cellular networks that performs significantly better than the fixed channel allocation scheme over a variety of traffic conditions. We formulated our objective in terms of a minimax problem and developed



(a) (b)

Figure 8: Impact of the granularity of the number of channels that can be shared: (a) Comparison of the overall packet dropping probabilities (b) Comparison of packet dropping probabilities in the congested cell

distributed algorithms that achieve this objective. We believe that our distributed sharing algorithm lends itself for easy implementation since a central controller is not required to assign channels to cells. We have shown the convergence of the distributed sharing algorithm to the optimal solution for commonly occurring packet arrival patterns. Simulation results for both the 1-D and 2-D cases show that our distributed channel sharing scheme performs significantly better than the FCA scheme over a variety of traffic conditions even with just a single complete iteration of the sharing algorithm.

References

[1] D. J. Goodman and S. X. Wei, "Efficiency of packet reservation multiple access," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 170–176, February 1991.

[2] M. Zhang and T.-S. Yum, "Comparisons of channel assignment strategies in cellular mobile telephone systems," *IEEE Transactions on Vehicular Technology*, vol. 38, no. 4, pp. 211–215, November 1989.

[3] S. Tekinay and B. Jabbari, "Handover and channel assignment in mobile cellular networks," *IEEE Communications Magazine*, vol. 29, no. 11, pp. 42–46, November 1991.

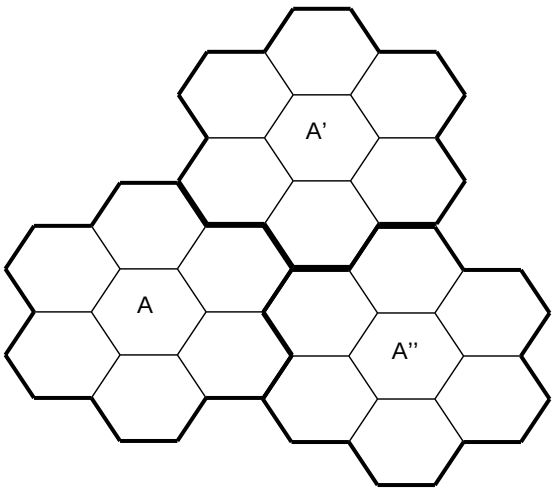


Figure 9: Illustration of a 2-D array of hexagonal cells for $R = 7$

[4] S. Kalyanasundaram, J. Li, E. K. P. Chong, and N. B. Shroff, "Channel sharing scheme for packet-switched cellular networks," in *IEEE INFOCOM '99*, (New York), pp. 609–616, March 1999.

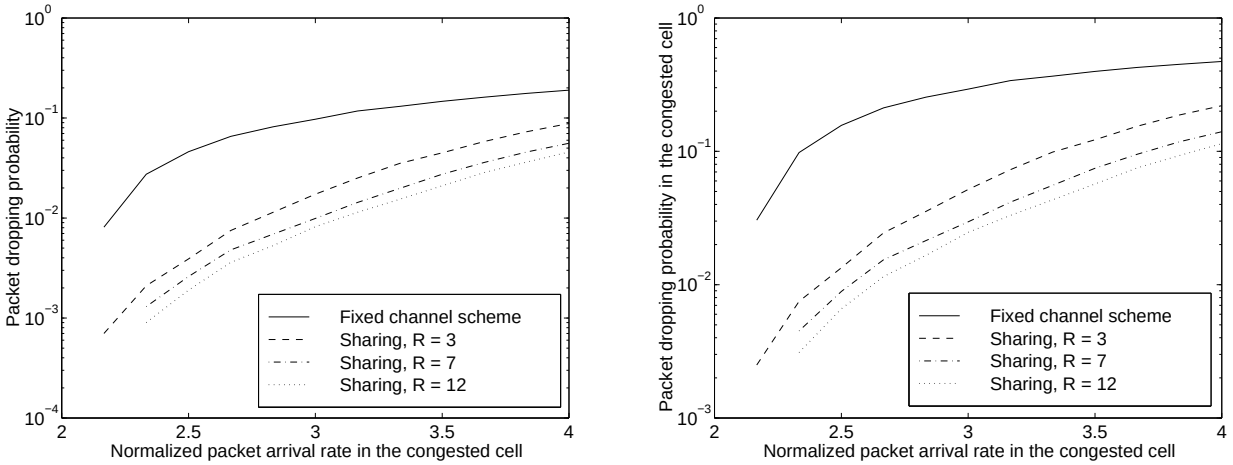
[5] C.-J. Chang, P.-C. Huang, and T.-T. Su, "Channel borrowing scheme in a cellular radio system with guard channels and finite queues," in *IEEE International Communications Conference*, vol. 2, (Dallas, TX), pp. 1168–1172, 1996.

[6] H. Jiang and S. Rappaport, "Prioritized channel borrowing without locking: A channel sharing strategy for cellular communications," *IEEE/ACM Transactions on Networking*, vol. 4, no. 2, pp. 163–171, April 1996.

[7] H. Jiang and S. Rappaport, "A channel borrowing scheme for TDMA cellular communications systems," in *IEEE Vehicular Technology Conference*, vol. 1, (Chicago, IL), pp. 97–101, 1995.

[8] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunication systems: A comprehensive survey," *IEEE Personal Communications Magazine*, pp. 10–31, June 1996.

[9] H. Jiang and S. Rappaport, "CBWL: A new channel assignment and sharing method for cellular communication systems," *IEEE Transactions on Vehicular Technology*, vol. 43, no. 2, pp. 313–322, May 1994.



(a) (b)

Figure 10: (a) Comparison of overall packet dropping probability (b) Comparison of packet dropping probability in the most congested cell

[10] J. Li, N. B. Shroff, and E. K. P. Chong, "A channel sharing scheme to improve system capacity and quality of service in wireless cellular networks," in *Proceedings of the Third IEEE Symposium on Computers and Communications*, (Athens, Greece), pp. 700–704, 1998.

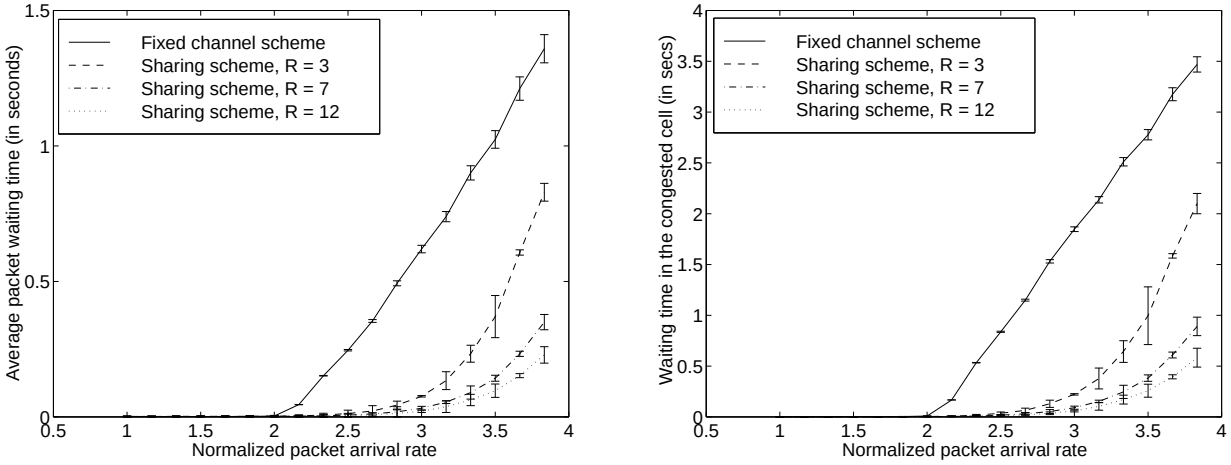
[11] J. Li, N. B. Shroff, and E. K. P. Chong, "A study of a channel sharing scheme in wireless cellular networks including handoffs," in *IEEE INFOCOM '99*, (New York), pp. 1179–1186, March 1999.

[12] J. Li, N. B. Shroff, and E. K. P. Chong, "A new localized channel sharing scheme for cellular networks," *ACM/Baltzer Wireless Networks*, vol. 5, no. 6, pp. 503–517, 1999.

[13] K. L. Yeung and T.-S. P. Yum, "Cell group decoupling analysis of a dynamic channel assignment strategy in microcellular radio systems," *IEEE Transactions on Communications*, vol. 43, no. 2–4, pp. 1289–1292, February–April 1995.

[14] C. Xu and F. C. M. Lau, *Load Balancing in Parallel Computers: Theory and practice.* Kluwer Academic Publishers, 1997.

[15] V. F. Dem'yanov and V. N. Malozemov, *Introduction to Minimax.* John Wiley and Sons, 1974.



(a) (b)

Figure 11: (a) Comparison of overall average waiting time with 99% confidence intervals (b) Comparison of average waiting time in the congested cell with 99% confidence intervals

Appendix

A Solution to the minimax problem

To solve the minimax problem in Eq. (3), we use the following theorem from [15].

Theorem A *Let $f_i(\boldsymbol{x}), i = 1, 2, \ldots, M$, be continuously differentiable functions on an open set Ω'. Let*

$$\Phi(\boldsymbol{x}) = \max_{1 \le i \le M} f_i(\boldsymbol{x})$$

and

$$R(\boldsymbol{x}) = \{i \in \{1, 2, \ldots, M\} : f_i(\boldsymbol{x}) = \Phi(\boldsymbol{x})\}.$$

Let Ω be a convex closed subset of Ω'. Then a necessary condition for a point $\boldsymbol{x}^* \in \Omega$ to be a minimum point of $\Phi(\boldsymbol{x})$ on Ω is that

$$\inf_{\boldsymbol{z} \in \Omega} \max_{i \in R(\boldsymbol{x}^*)} \left\langle \frac{\partial f_i(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle = 0 \tag{38}$$

where $\langle a, b \rangle$ denotes the inner product of a and b. If $\Phi(\boldsymbol{x})$ is convex, this condition is also sufficient.

In our case, $\Omega' = \{[x_1, x_2, \ldots, x_{M-1}] : 0 < x_i < N,\ i = 1, 2, \ldots, (M-1)\}$. It is clear that, for any $\boldsymbol{x} \in \Omega$, the quantity on the LHS of Eq. (38) is at most zero. We can see this by choosing $\boldsymbol{z} = \boldsymbol{x}$, which results in the inner product of zero being taken with $\partial f_i / \partial \boldsymbol{x}$,

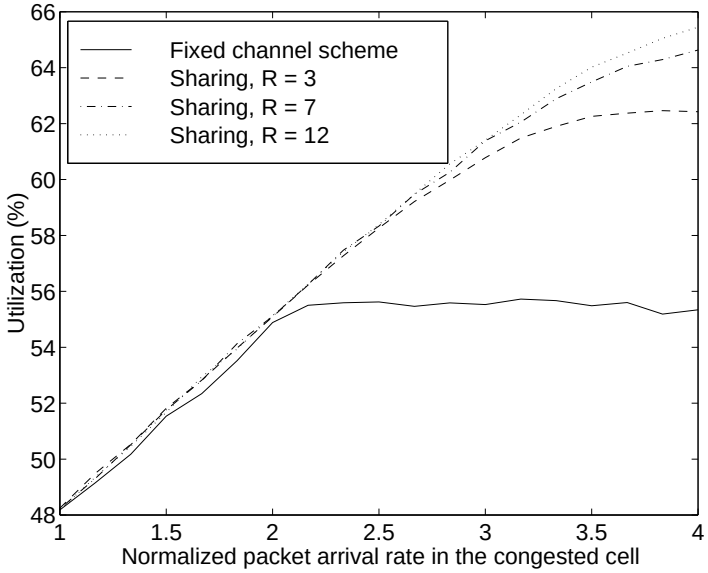


Figure 12: Utilization curves for the sharing and the FCA schemes

$i = 1, 2, \ldots, M$. Hence, to prove that an $\boldsymbol{x}$ satisfies the necessary condition for a minimizer, we have to prove that the quantity on the LHS of Eq. (38) cannot be negative.

We solve the minimax problem (stated in Eq. (3)) by first guessing the solution and then showing that it satisfies the conditions for the minimizer.

Theorem 3 *If the system of equations*

$$\rho_1 = \rho_2 = \cdots = \rho_M$$

results in a solution $\boldsymbol{x}^* = [x_1^*, x_2^*, \ldots, x_{M-1}^*]$ *that satisfies the constraints*

$$0 < x_i^* < N, \ i = 1, 2, \ldots, (M-1),$$

then $\boldsymbol{x}^*$ *is the unique optimal solution to the optimization problem given in Eq. (3) (without the integer constraints).*

Proof: We first prove that $\boldsymbol{x}^*$ satisfies the condition in Eq. (38). The proof is by contradiction. Assume that

$$\inf_{\boldsymbol{z} \in \Omega} \max_{i \in R(\boldsymbol{x}^*)} \left\langle \frac{\partial \rho_i(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - x^* \right\rangle < 0. \tag{39}$$

Because $\boldsymbol{x}^*$ was obtained by equating all the ρ, we have that

$$R(\boldsymbol{x}^*) = \{1, 2, \ldots, M\}.$$

Eq. (39) implies that there exists a j and $\boldsymbol{z} \in \Omega$ such that

$$\left\langle \frac{\partial \rho_j(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle < 0, \tag{40}$$

and

$$\left\langle \frac{\partial \rho_i(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle \leq \left\langle \frac{\partial \rho_j(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle \text{ for all } i \neq j. \tag{41}$$

Eq. (40) and Eq. (41) together imply that there exists $\boldsymbol{z} \in \Omega$ such that

$$\left\langle \frac{\partial \rho_i(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle < 0, \text{ for all } i. \tag{42}$$

We will now show that this cannot happen. From Eq. (2), we have

$$\begin{aligned}
\left\langle \frac{\partial \rho_1(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle &= \frac{-\lambda_1(z_1 - x_1^*)}{(N/2 + x_1^*)^2} < 0 \\
\left\langle \frac{\partial \rho_2(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle &= \frac{\lambda_2(z_1 - x_1^*)}{(N - x_1^* + x_2^*)^2} - \frac{\lambda_2(z_2 - x_2^*)}{(N - x_1^* + x_2^*)^2} < 0 \\
&\vdots \\
\left\langle \frac{\partial \rho_i(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle &= \frac{\lambda_i(z_{i-1} - x_{i-1}^*)}{(N - x_{i-1}^* + x_i^*)^2} - \frac{\lambda_i(z_i - x_i^*)}{(N - x_{i-1}^* + x_i^*)^2} < 0 \\
&\vdots \\
\left\langle \frac{\partial \rho_M(\boldsymbol{x}^*)}{\partial \boldsymbol{x}}, \boldsymbol{z} - \boldsymbol{x}^* \right\rangle &= \frac{\lambda_M(z_{M-1} - x_{M-1}^*)}{(3N/2 - x_{M-1}^*)^2} < 0.
\end{aligned} \tag{43}$$

Since $\lambda_i > 0$, for all i, we have

$$\begin{aligned}
-(z_1 - x_1^*) &< 0 \\
(z_1 - x_1^*) - (z_2 - x_2^*) &< 0 \\
(z_2 - x_2^*) - (z_3 - x_3^*) &< 0 \\
&\vdots \\
(z_{M-2} - x_{M-2}^*) - (z_{M-1} - x_{M-1}^*) &< 0 \\
(z_{M-1} - x_{M-1}^*) &< 0.
\end{aligned} \tag{44}$$

Adding all the above equations, we get $0 < 0$, which gives us our contradiction. Therefore, $\boldsymbol{x}^*$ satisfies the necessary conditions for a minimizer.

We next note that $\Phi(\boldsymbol{x}) = \max_{1 \leq i \leq M} \rho_i(\boldsymbol{x})$ is convex, because $\rho_i(\boldsymbol{x})$ for each i is convex [15]. Therefore, $\boldsymbol{x}^*$ satisfies the sufficient conditions for a minimizer. Further, because our minimax optimization problem is a convex optimization problem, the set of all global minimizers is convex. But, in our problem, any small change in the optimal solution results in a change in the objective function value. Therefore, the minimizer is unique over Ω. $\square$