Distributed frequency allocation for cellular networks: Trade-offs and tuning strategies

MARINA PAPATRIANTAFILOU DAVID RUTTER PHILIPPAS TSIGAS
Comp. Science Department, Chalmers University of Technology, Göteborg, Sweden
<ptrianta, rutter, tsigas>@cs.chalmers.se

ABSTRACT
Frequency allocation in cellular networks has a number of optimisation criteria and a number of interesting trade-offs involved in trying to meet them. Analysing them is particularly important for applying frequency allocation algorithms in practice. However, this can be prohibitively complex, even in a sequential setting, and especially in a distributed setting. The latter implies a larger number of parameters to consider, but is more suitable for dynamic solutions. This, besides analytical evaluation, necessitates the use of experimentation in tuning the algorithms' performance.

In [4], it was shown how to analytically measure and provide worst-case guarantees regarding *request satisfiability* and how to provide *fully dynamic* frequency assignment (i.e. *on-demand*) with low communication and time overhead, which is particularly important for adapting to temporal changes in frequency demands in each cell.

In applying these methods, the *trade-off* between connection set-up time and request satisfiability had to be considered. Our conjecture was that, by designing appropriate *dynamic* tuning strategies, the trade-offs involved could be balanced in a way so that the gains are substantial, while the losses remain small. The key results of our study, confirming the above conjecture, are the following: We propose strategies for applying and fine tuning the performance of the methods in [4], mainly addressing the above decision and without affecting the worst-case guarantees. The proposed strategies are *dynamic*, i.e. the amount of frequencies *allocated* at each base station follows the *offered load* at that station. Next we study the performance of the original algorithms and their tunable versions. The set of experiments model a variety of situations, including dynamic, non-uniform *load* and base station *failures*, having the main variable parameter actually be the *temporal-change-of-traffic-distribution*; it measures an extensive range of performance metrics (including *utilisation, frequency reuse, fault-tolerance*). Also, our study provides a way to measure the down-side of the trade-off and get some new insights into the issues involved in attempting to optimise key constraints.

KEY WORDS
Cellular Networks, Dynamic Distributed Frequency Allocation

1. Introduction

A network that supports mobile communication is typically divided into *cells*, each one having a *base station*. The base stations support broadcast communication within the cell, and are connected by a fixed network - usually a wire network (although this is not assumed in this study), with communication between base stations usually based on point-to-point message transmission. In order to communicate, a *mobile host* (MH) must first send a request to the base station for its current cell. The base station will then, subject to availability, assign a number of *communication channels*, or *frequencies*, for the MH to use, depending on the bandwidth required for the communication session. The same frequency cannot be used by cells within a certain broadcast range, as the broadcast ranges of two neighbouring base stations may overlap, causing *co-channel interference*. However, if the cells are geographically separated then the same channel can be used. This is known as *frequency reuse*. (It should be pointed out that the total number of frequencies available, or the *spectrum*, is limited by a number of factors [5, 6].) If a communicating mobile host changes cell, it might be possible that it keeps the same frequency in the new cell or not, in which case a new frequency must be assigned; this is known as *inter-cell hand-off*. Since mobile hosts generally operate on a limited battery power supply, power consuming operations such as network communications, CPU operations and disk accesses should be kept to a minimum. A mobile host's involvement in the session request process should therefore be minimal. Also, since the cost of breaks in mobile data communications can be high, unnecessary breaks in communication, such as those caused by *intra-cell hand-offs*, should be avoided.

A cellular network generally services a large geographic area, with the base stations at varying distances from each other. In current networks, it is usually the case that there is a central node that allocates the frequencies to the base stations. This causes two problems: The first is that the communication delay increases with the distance of a base station to the central node. The second is that the failure of the central node affects the entire network, while the failure of any base station along a path to the central node also affects all base stations whose communication paths include the failed node. A solution should then provide a fault tolerance that is at least better than the scenario described above - i.e. minimises the *failure locality* [3]. The failure locality is the size of the network affected by a faulty node in the network, measured by the distance in nodes.

In summary, a good solution for frequency allocation in cellular networks should:

- Avoid co-channel interference
- Maximise frequency reuse and satisfiable requests
- Minimise connection setup time
- Minimise total communication overhead
- Minimise the impact of failures in the network
- Minimise/avoid the involvement of the MH
- Avoid intra-cell hand-offs

It can be observed that in order to achieve the goals of dynamic channel assignment and minimal failure locality, a *distributed solution* is required, where the base stations "negotiate" between themselves for the set of frequencies, instead of depending on some central stations which may be some distance apart. A local distributed solution can reduce the failure locality, as there is no central point of control, which in the event of failure can cause the entire network to fail.

Prakash, Shivaratri and Singhal, in [9], introduced such a distributed solution (DIST_DCA). The algorithm, during a primary phase, distributes the free frequencies to cells on demand, so that no conflicts occur. Later, to respond to temporal changes of load in different cells, it employs a borrowing scheme to allow the allocation to be dynamic. While the algorithm is expected to perform adequately under stable load, under dynamically changing load it may introduce arbitrarily long waiting (dependency) chains among stations waiting for replies; this may result in long waiting times and poor failure locality.

Garg, Papatriantafilou and Tsigas, in [4], introduced two distributed solutions, that allocate frequencies to base stations on-demand (and hence are fully dynamic), based on generalised distributed list colouring of the conflict graph describing potential co-channel interference. The first solution (DET_DLC) achieves the equivalent of the best known sequential upper bound for request satisfiability [2]. By employing a powerful synchronisation mechanism due to Choy and Singh [3], it achieves low time and communication overhead, as well as low failure locality. The second solution in [4] (RAND_DLC) is a first approach towards exploring the achievable bounds in request satisfiability without use of extra synchronisation and employs randomisation. The worst-case guarantee for request satisfiability is not as high as the deterministic solution, but the failure locality is minimal.

The challenge was to see how these guarantees reflect when applying these methods. In doing so we had to address the following issue: Aiming at maximising the number of satisfiable requests, the best strategy is to assign frequencies to base stations on demand, leaving the unused frequencies to satisfy calls in different base stations. On the other hand, considering minimising the overhead per connection, it is best for each base station to reserve some of the frequencies for satisfying as many requests as possible locally. Our conjecture was that, by designing appropriate *dynamic* tuning strategies, the trade-offs involved could be balanced in a way so that the gains are substantial, while the losses remain small.

The primary goal of our investigation has been to design strategies for *fine tuning* the performance of the solutions, mainly addressing the above decision and without affecting the worst-case guarantees. The strategies should be *dynamic*, aiming at retaining the dynamic nature of the aforementioned algorithms: the amount of frequencies *allocated* at each base station should follow the *offered load* at that station (or a *fair* proportion of it, if the total offered load in a neighbourhood of base stations is higher than the load that can be served in that neighbourhood).

To thoroughly study the performance of the original and the modified algorithms beyond worst-case guarantees shown in [4], we designed a set of experiments to: (i) model a variety of situations, including dynamic, non-uniform *load* and base station *failures*, having the main variable parameter actually be the *temporal-change-of-traffic-distribution*; (ii) measure an extensive range of performance metrics (including *utilisation, frequency reuse, fault-tolerance*). It should be mentioned that, to the best of our knowledge, neither of the above was done before in the experimental study of algorithms in this field. This modelling provides also a way to measure the down-side of the trade-offs and get new insights into the issues involved in optimising key constraints. The above, besides forming a method to test-under-pressure tuning techniques, gives a new perspective by suggesting a new, arguably hard evaluation procedure for dynamic frequency allocation protocols, and by pointing to new questions that follow from this study and deserve further analytical investigation.

To summarise some indicative figures of the experiments: using the proposed strategies, we could achieve 6-7 times faster response, 3-5 times reduced communication complexity, at the cost of approximately 1 unit, in the per-cent scale, of requests that could not be met and of bandwidth utilisation, thus confirming our conjecture. Furthermore, as we expected, the randomised approach performs significantly better than the worst-case guarantee proven in [4]: in particular, its utilisation and frequency reuse figures are very close to those of the deterministic one. Moreover, the results reveal the great importance of small failure locality and the very small influence of failures on the utilisation, which seems only slightly affected compared with the deterministic.

Comparing DET_DLC and RAND_DLC with DIST_DCA, we observe, as expected, much better guarantees in request satisfiability, with comparable results in bandwidth utilisation. Regarding fault tolerance, the difference is significant: 3 times lower number of dropped calls, 8 (this is 15 for RAND_DCA, which has minimum failure locality) times lower number of affected stations. The comparison in response times and communication complexity in the absence of failures is negative, however, we contend that as the network size grows, the long waiting chains expected of DIST_DCA will show their impact, making the advantages of DET_DLC and RAND_DLC, in which the waiting chains depend only on local measures, even more prominent.

2. Measures and Experiment Parameters

Below we define the measures we are using to evaluate the algorithms. First, some notation: v_i denotes a base station, $N(i)$ denotes its neighbourhood, i.e. itself and the nearby base stations with which the potential for co-channel interference exists, and $Used_i$ denotes the number of frequencies it actually uses each time in order to serve requests.

Response Time: the implied overhead for the connection setup time. Both the mean response time for queued requests, and the mean for all requests are measured.

Direct Connections: the frequency requests that can be serviced locally, without the need for the base station to acquire more, thus implying a faster response time. This, combined with the response time gives a more complete picture of the connection setup time, as a direct connection will have a significantly faster response time than a queued request.

Dropped Calls: the number of requests that could not be satisfied.

Total Messages: the total number of messages (or communication complexity) over the whole network for the entire experiment execution.

Frequency Reuse: the ratio of the number of calls served in a neighbourhood over the number of frequencies that are used for them, i.e. $\frac{\sum_{j\in N(i)} |Used_j|}{|Used|}$, where $Used = \bigcup_{j\in N(i)} Used_j$. It is measured when a call is dropped.

Bandwidth Utilisation: the total number of frequencies in use, divided by the spectrum size, i.e. $\frac{|Used|}{Spectrum}$, where $Used = \bigcup_{j\in N(i)} Used_j$, providing an indication of the effectiveness of the tuning strategy used, a higher value indicating that the algorithm is effectively estimating the number of frequencies to acquire and retain. It is measured when a call is dropped.

Number of Affected Base Stations: the number of base stations which have to wait indefinitely due to the failure of others.

For the experiments we used DIAS, an event-based distributed algorithm simulator, based on the message passing model, which is part of Lydian[1]. Having as our main aim to simulate, the traffic for a real cellular network, the key parameter we used for capturing this is the use of "*hot-cell configurations*" which is a distribution of the load across the network. In each configuration a cell may be *hot, normal,* or *cold*, corresponding to having high, moderate or low load, respectively. The configurations changed throughout the execution, reflecting non-uniform, dynamically-changing load. The simulation parameters are:
Network size: 49 cells (7x7 cellular grid; internal and border cells have 6 and 2-3 neighbours, respectively)
Spectrum size: 500 frequencies
Types of cells: hot, normal, and cold.
Changes in hot-cell configurations, modelling dynamic and non-uniform load: the experiments were carried out for 1 to 10 changes in hot-cell configurations.
Network (message) delay: 1 to 3 milliseconds.
Process (base-station) step time delay: 1 to 2 microseconds.
Mean call duration: exponentially distributed with a mean $1/\mu = 3$ minutes
Mean call arrivals rate λ: Poisson distribution, in hot-cells: $\lambda = 85/min$, normal cells: $\lambda = 45/min$, cold cells: $\lambda = 20/min$.
Length of simulated execution: 100,000 requests, each one requesting one channel (frequency)
Failures: Up to three crash-failures occur at arbitrary stations at arbitrary points in the execution.

3. Algorithms and Tuning Strategies

A brief description of the algorithms studied is given here; the pseudo-code presentation of their protocols can be found in the respective references. The algorithms are based on the modelling of the cellular network as an *interference graph*: vertices denote cells (base stations), and an edge between two cells denotes the potential for co-channel interference.

Deterministic Distributed List Colouring: The DET_DLC solution, proposed by Garg et al. [4], models the problem as a generalised list-colouring problem: each node (base station) of a graph has a list of colours (free frequencies) and is requested to colour itself with a number of colours (the number of pending requests) without any conflicts with its neighbours. The solution achieves the equivalent of the best known sequentially achievable upper bound for request satisfiability, implied by a theorem due to Alon and Tarsi [2]. The solution assumes an initial acyclic orientation of the graph (it can be obtained by a vertex colouring and orienting each edge towards, w.l.o.g. its lower-coloured adjacent vertex) and uses it for resolving priorities among competing base stations. It is based on a mutual exclusion approach - where only one base station in a neighbourhood is able to select available frequencies at any given point in time. It employs a double doorway mechanism [3] to ensure synchronisation among the neighbours with small delay and no starvation. The solution has a failure locality of 4, worst case response time of $O(\Delta)$ and a messaging complexity of $O(\Delta^2)$, where Δ is the degree of the graph (the maximum degree among all vertices in the graph).

Randomised Distributed List Colouring: The RAND_DLC solution by Garg et al. [4] does not rely on a mutual exclusion approach. Instead, each base station is able to select a random subset of the available frequencies, and then, through a negotiation phase with its neighbours, determine which of the frequencies can be used. The solution ensures that no frequency can be concurrently used by any two neighbours. The expected number of frequencies that a station can acquire is a fraction of $1/(4\Delta)$ of the free frequencies at the station. The message complexity of the solution is 3Δ, and the failure locality is 1.

Distributed Dynamic Channel Allocation: In evaluating the performance of the algorithms in [4] and the dynamic tuning strategies proposed here, we compare with a solution by Prakash et al [9] (DIST_DCA), which: (i) is distributed (based on a successful centralised technique), and (ii) has earlier been shown experimentally ([9]) to perform well under stable traffic (in particular, under traffic whose distribution varies in time only little compared to the needs which implied a frequency assignment made during the initialisation phase). DIST_DCA is based on a multiple mutual exclusion scheme: a base station attempts to acquire one frequency at a time, initially getting them from the local set of available frequencies, and, later, when that is exhausted, borrowing them from neighbouring stations. The mutual exclusion mechanism is based on a priority scheme that uses timestamps based on Lamport's logical clocks [8], to order the base stations requests. This method of ordering however has a number of negative properties, one being the length of the base stations waiting chain, which can grow to the size of the network. This also implies a failure locality of the same size, i.e. the size of the network. However, in a "friendly" environment, where the load is uniform and does not change frequently, and where there are no faults, the algorithm performs reasonably well, this being the reason that we consider it as a benchmark, as mentioned above.

3.1 Tuning Strategies

Frequency Acquisition and Retention

In the default behaviour of DET_DLC, a base station i will select a subset S of the free frequencies, such that $|S| = min(|Free_i|, r_i)$, where r_i is the number of frequency requests for process/base station i. However, in certain situations a frequency acquisition strategy can take into account that some acquisition/retention of extra frequencies can pay-off. The challenge is to get the benefits from such a policy without sacrificing the *dynamic* properties of DET_DLC.

In the default behaviour of the RAND_DLC, a fixed fraction of the free frequencies are chosen. The expected number of frequencies to be picked without conflicts with neighbours that are choosing concurrently is maximised using an $1/(\Delta + 1)$-fraction. In practical terms, this value ensures that a process will select a set of frequencies inversely proportional to the number of neighbours it has. Under uniform load, this is ideal, as the frequencies will be distributed almost equally amongst the base stations in the network. However, if the load is not evenly distributed this may be not the best choice; for example a cold cell (one which has a light load) may acquire frequencies which could better be used by one of its neighbours which has a higher load. Since frequencies are only made available when no longer required by a user (in the *release*() function), it may be some time until these unused frequencies are made available again. This calls for the investigation of acquisition strategies that are able to respond to changes in the load of the network. In doing so we must maintain the *locality* of the solution.

The second observation we can make of both DET_DLC and RAND_DLC, is that a base station v_i releasing a frequency after it has been used, makes it available for other neighbours, but

Parameter	Description
Δ_i	degree (# of neighbours) of v_i)
$Busy_i$	set of frequencies held by v_i
λ_i	mean request-arrival rate at v_i
T	mean call duration
r_i	actual number of requests in waiting queue at v_i
$Little'sLaw$	mean number of requests estimated at v_i ($\lambda_i T$)
$free\_ratio$	fraction of frequencies to be left in neighbourhood for fairness
min_ratio_i	$min(\lambda_i T) \neq 0$ observed by v_i
LittlesLaw Strategy	$\lambda_i T$ - $\|Busy_i\|$
$QueueRatio_i$	$r_i(1{+}1/(\Delta{+}1))(1{-}free\_ratio)$
QueueRatio Strategy	$max(QueueRatio_i, min\_ratio_i)$

Table 1. Definitions Used in Tuning Strategies

it (v_i) may have queued requests that require the use of the frequency. The net effect of this is that the call is queued and v_i must acquire another frequency using the respective algorithm. The challenge here is to develop a fair retain strategy *without starving* the low-load cells.

Parameters for the tuning strategies

We consider two strategies for retention and acquisition of frequencies, outlined in this subsection. The related definitions can be found in Table 1.

The first and most straightforward strategy is to estimate the number of frequencies required based on *Little's Law* [7]. This method gave significant improvements in time and communication overhead per request, at a very small cost in dropped calls and bandwidth utilisation compared with the default DET_DLC and RAND_DLC solutions studied; the results and the improvements are discussed in the next section.

In our effort of further investigating where the best balance in this trade-off lies, we introduced a more refined strategy, which is based on the current waiting queue size r_i for a station v_i and attempts to fine-tune the overall performance of the solutions through two parameters. The first parameter, called the *free_ratio*, reduces the number of frequencies acquired (or retained), in an attempt to additionally ensure fairness in frequency acquisition (in addition to fairness in the competition synchronisation, which is taken care of by the algorithms studied) and see the effects in the overall algorithm performance. The second parameter, the*minimum ratio*, is used when the queue size at a station is very small. The minimum ratio is calculated as the minimum non-zero Little's-Law estimate encountered so far for the base station. This attempts to ensure that some frequencies are acquired or retained, so that cold cells are able to service their calls directly, without incurring additional communication cost. The assumption is made that the trade-off is small in this case (frequency availability vs. response time). These two parameters are used in order to compute the *QueueRatio* and the estimate of frequencies to be acquired/retained by a base station, according to Table 1.

In the case of RAND_DLC, when a frequency is required, then we attempt to acquire one by picking a subset from the set of free frequencies known to station v_i. If none are available, we can either drop the call or retry. Retrying immediately is likely to produce the same result, so it is worthwhile to *back-off* and try again after a set time period. This time period must of course be determined by the time that a mobile user is prepared to wait for a connection. For this reason, in this study, one retry attempt was used, and the back-off time was chosen randomly between 100 and 500 milliseconds.

4. Experiment Results

The results for all algorithms are summarised graphically in the end of the paper, in an attempt to give a complete overview of them. These were derived from graphs such as those in Figure 1, by using the values where the measures seem to converge after 10 changes of hot-cell configurations. Figure 1 shows some key results regarding improvements and trade-offs implied by the tuning strategies to DET_DLC.

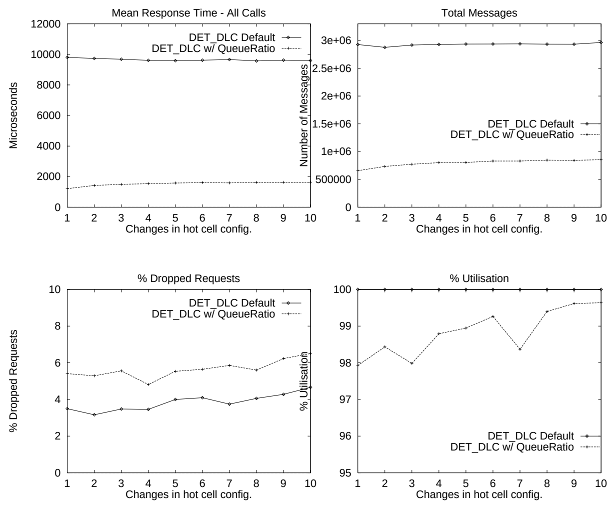


Figure 1. Indicative figures showing the improvements and trade-offs using the QueueRatio strategy for DET_DLC

Expected Trade-offs and Results

If we take the perspective of trying to maximise the number of direct connection requests, we can see that for DET_DLC this can be improved dramatically, by as much as 77%, as the default algorithm does not meet any requests directly. There are trade-offs involved in acquiring extra frequencies and retaining frequencies, the most important being an increase in the number of dropped connection requests of 1.8% for the QueueRatio strategy and 1.2% for the Little's Law based strategy. Here we see evidence of the trade-off discussed previously. The utilisation results also conform to this, as the results for the QueueRatio and Little's Law strategies are lower (by 1-1.5%) than the default algorithm, which is 100%. There are some other benefits in trying to maximise the number of requests we can service directly, such as a great reduction in the number of messages, from $2.9 * 10^6$ to $7 * 10^5$ for the QueueRatio strategy and $1.1 * 10^6$ for the Little's Law strategy. The increase in direct connections also results in a lower response time, an improvement by around $2ms$ for queued requests, and $8ms$ overall, which is actually approximately 6.5

times lower. Using the tuning strategies we can get results comparable to those of DIST_DLC —keeping in mind that this is a relatively “friendly” environment, i.e. no faults.

For RAND_DLC, the benefits are not so pronounced. This is because in the default algorithm the number of frequencies acquired can in many cases be larger than the number of requests, implying availability of frequencies to service requests immediately. Using the tuning strategies we can increase the number of direct connections by 17% for the QueueRatio strategy and 10% for Little’s Law, with the trade-off being that less requests can be satisfied in total. As with DET_DLC, this reduction in satisfiability is reflected in the results for the utilisation and frequency reuse, with the QueueRatio strategy faring marginally better than the Little’s Law based strategy. One negative result is the increase in response time, both for queued requests and overall. This is due to the action of the back-off strategy. The reduction in the availability of frequencies requires that a base station retry the acquisition process more often, resulting in more back-off attempts, and as a result the response time is higher. We can also note that the utilisation and frequency reuse is lower for RAND_DLC (for the default strategy the utilisation is 0.70% lower), this also playing an important role in request satisfiability.

Fault Tolerance
The presence of faults in the network highlights the benefits of using not only DET_DLC and RAND_DLC, but also the proposed tuning strategies, especially compared to DIST_DCA.

As indicated in the description of the algorithms, the failure locality of DET_DLC and RAND_DLC is 4 and 1, respectively, whereas DIST_DCA has a failure locality that can potentially be the diameter of the network. These analytically proven bounds for the algorithms failure locality and its impacts are reflected in the experimental results, as it is clear that the number of direct connections and dropped calls, as well as the frequency reuse and bandwidth utilisation are all adversely affected to a significantly greater extent than DET_DLC and RAND_DLC. The number of affected stations for DIST_DCA is very high, for the case of three failures the number affected is 43.8, or 89%, being close to that figure even in the case of one failure. For DET_DLC the numbers of affected base stations were 18.8 (40%) and 38 (78%) for the Little’s Law strategy and QueueRatio strategy for one and three failures, respectively. For RAND_DLC the lowest and highest number were just 2.45 (5%) and 12.9 (26%) for the Little’s Law strategy and the default solution, for one and three failures, respectively. Other important measures, such as request satisfiability are affected, with the dropped requests being as high as 58%, compared to 18% for RAND_DLC (default) and 47% for DET_DLC (default).

It should be noted that the effect of failures on the number of messages is a result of there being less base stations in operation. We have not estimated the *response-time figures for the case of failures*, as, similarly, due to the less contention, an algorithm that serves (significantly) fewer requests is bound to have (significantly) lower average response time, as computed over the set of answered requests; if we consider that the response time for the dropped calls in the affected base stations is actually infinite then it can be seen why, either way, the figures in this case could not accurately describe the performance of the algorithms.

An interesting result for DET_DLC and RAND_DLC is the positive effect of the tuning strategies on the number of affected base stations and dropped calls. It is also interesting to note that for the Little’s Law strategy the number of dropped calls is relatively unaffected by failures. For RAND_DLC this is the case; for DET_DLC, there is an increase in dropped calls for the first failure, and then the number is stable following this. The QueueRatio strategy has a positive effect as well in most cases, with the exception of the affected base stations for DET_DLC. This positive effect of the tuning strategies can be attributed to the base stations making less acquisition attempts. If we note that a base station can only be affected by a failure if it must send a message to an affected base station, then if less requests are made, then also less base stations should be affected, and thus less dropped calls should be the result. We contend that in a larger network, the long waiting chains expected of DIST_DCA will have a greater probability of forming, making the advantages of using DET_DLC and RAND_DLC even more prominent.

5. Conclusions and Future Work

Perhaps the least obvious property of many frequency allocation solutions is the manner in which the frequencies are selected. An algorithm that uses the frequency reuse information, while maintaining the failure locality and the guarantees of the previous solutions, is the subject of our planned research. Another interesting direction is the continuation of this study together with priority and/or or frequency reservation schemes for accommodating hand-offs, in combination with QoS parameters.

References

[1] Lydian - An Educational Animation Environment for Distributed Algorithms and Protocols. Available at: http://www.cs.chalmers.se/~lydian/ (2001)

[2] Alon, N., Tarsi, M.: Colourings and Orientations of Graphs. In Combinatorica 12 (2), Springer-Verlag (1992) 125-134.

[3] Choy, M., Singh, A.: Efficient Fault Tolerant Algorithms for Resource Allocation in Distributed Systems. In ACM Transactions on Programming Languages and Systems Vol 17 (3) May 1995, ACM Press New York (1995), 535-559.

[4] Garg, N., Papatriantafilou, M., Tsigas, P.: Distributed Long-Lived List Colouring: How to Dynamically Allocate Frequencies in Cellular Networks. To appear in ACM/Baltzer Wireless Networks journal (WINET). Prel. version in Proceedings of the 8th IEEE Symposium on Parallel and Distributed Processing, IEEE New Jersey (1996) 18-25.

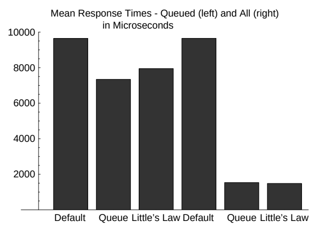
[5] Hild, S.G.: A Brief History of Mobile Telephony. Technical Report No. 372, University of Cambridge, Computer Laboratory. (1995)

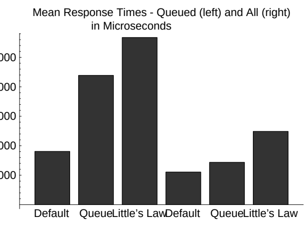
[6] Ioannidis, G.: Mobile Computing. PhD Thesis. Columbia University (1992)

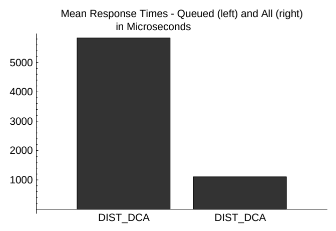
[7] Kleinrock, L.: Queueing Systems : Volume 1 Theory. Wiley (1976)

[8] Lamport, L.: Time, Clocks and the Ordering of Events in a Distributed System. In Communications of the ACM Vol 21(7) July 1978, ACM Press New York, 558-565.

[9] Prakash, R., Shivaratri, N.G., Singhal, M.: Distributed Dynamic Fault-Tolerant Channel Allocation for Mobile Computing. In IEEE Transactions on Vehicular Technology Vol 48 (6) November 1999, IEEE New Jersey, 1874-1888



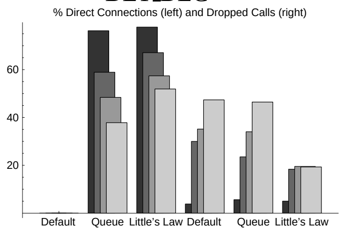


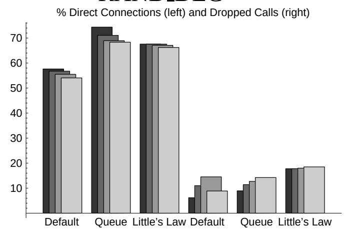


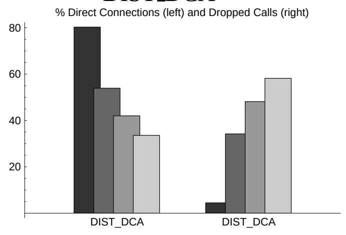
DET_DLC

RAND_DLC

DIST_DCA



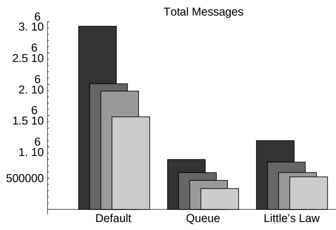


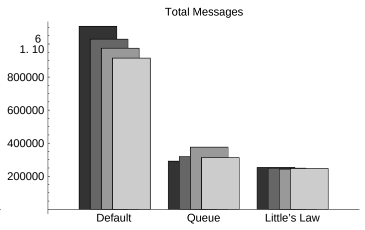


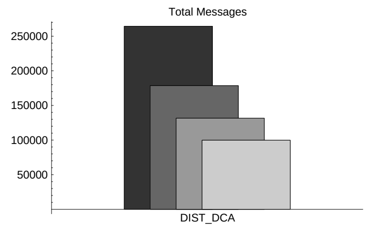
DET_DLC

RAND_DLC

DIST_DCA



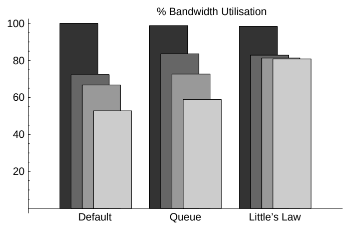


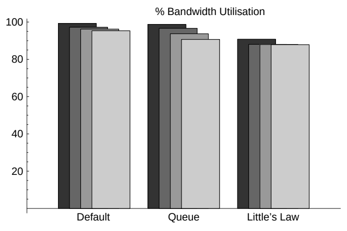


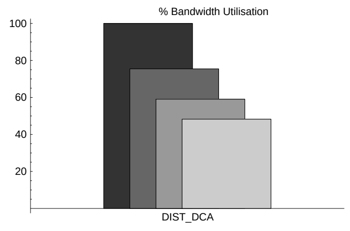
DET_DLC

RAND_DLC

DIST_DCA



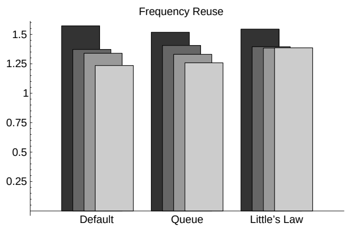


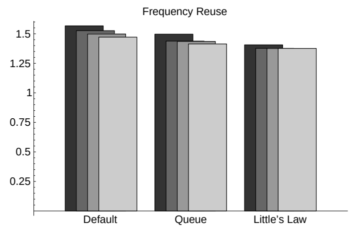


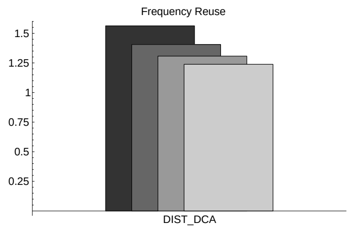
DET_DLC

RAND_DLC

DIST_DCA



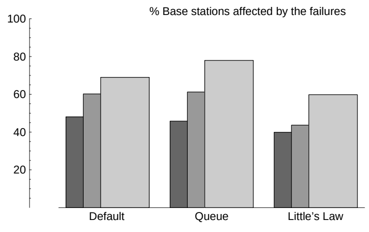


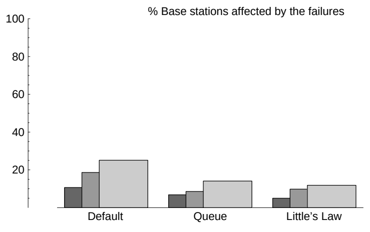


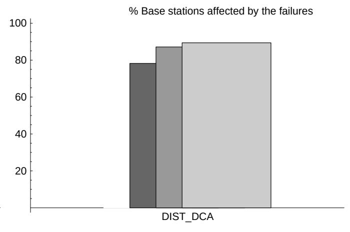
DET_DLC

RAND_DLC

DIST_DCA







DET_DLC

RAND_DLC

DIST_DCA

⬛ No Failures ▩ 1 Failure ▨ 2 Failures ⬜ 3 Failures