Efficient Computation of Optimal Capacity in Multiservice Mobile Wireless Networks*

Vicent Pla, Jorge Martínez and Vicente Casares-Giner
Department of Communications
Universidad Politécnica de Valencia (UPV)
ETSIT Camí de Vera s/n, 46022 Valencia, Spain.
Telephone: +34 963879733, Fax: +34 963877309
E-mail: (vpla,jmartinez,vcasares)@dcom.upv.es

Abstract

In this paper we propose a new algorithm for computing the optimal parameters setting of the Multiple Fractional Guard Channel (MFGC) admission policy in multiservice mobile wireless networks. The optimal parameters setting maximizes the offered traffic that the system can handle while meeting certain QoS requirements. The proposed algorithm is shown to be more efficient than previous algorithms appeared in the literature.

1 Introduction

The enormous growth of mobile telecommunication services, together with the scarcity of radio spectrum has led to reducing the cell size in cellular systems. Smaller cell size entails a higher handoff rate having an important impact on QoS and radio resource management. During the last two decades a considerable number of papers have addressed this topic (see, for instance [1–3]). Moreover, forthcoming 3G networks will establish a new paradigm with a variety of services having different QoS needs and traffic characteristics.

*This work has been supported by the Spanish Government (PGE, 30%) and the European Commission (FEDER, 70%) through the projects TIC2003-08272 and TEC2004-06437-C05-01.

Admission control in the presence of mobility or multiple services is quite well studied. However, this new paradigm where multiservice and mobility meet has not received attention from researchers until very recently.

In [4] Li et al. propose an extension of the well-known *Guard Channel* (GC) [1] mechanism where multiple service types are considered. Bartollini and Chlamtac [5] considered a more general policy than that of [4]. More recently, Heredia et al. [6–8] had proposed an extension of the *Fractional Guard Channel* (FGC) [9] scheme. The structure of optimal admission policies in single service cellular networks under different criteria is studied in [9, 10]. In [5] the authors show that the optimal admission policy (with respect to a certain cost function) in a multiservice cellular network does not belong to any of the types mentioned above; instead it belongs to the wider family of stationary policies [11]. In [12] several types of admission policies for cellular multiservice networks (including MFGC and randomized stationary) have been compared.

In this paper we propose a new algorithm for computing the optimal parameters setting of the of Multiple Fractional Guard Channel (MFGC) admission policy in multiservice mobile wireless networks. As it will be explained later, in MFGC the policy parameters control the amount of system resources that each call type can access. The optimal parameters setting maximizes the offered traffic that the system can handle while meeting certain QoS requirements. To the best of our knowledge only one algorithm for this purpose has been proposed [8] in the literature and its computational performance is substantially improved by the one proposed here. Besides, we observe that a further enhancement of both algorithms is possible by eliminating the iterative procedure of computing the handoff arrival rates.

The remaining of the paper is structured as follows. In Section 2 the system model is described and its mathematical analysis is outlined in Section 3. Section 4 describes in detail the new proposed algorithm. Computational complexity of the algorithm is comparatively evaluated in Section 5. Finally, Section 6 concludes the paper.

2 Model Description

The system has a total of C resource units. The physical meaning of a unit of resources will depend on the specific technological implementation of the radio interface.

The system offers N different classes of services. For each type of service new and handoff call arrivals are distinguished so that there are N types of services and $2N$ types of arrivals. Arrivals are numbered in such manner that for service i new call arrivals are referred to as arrival type i, whereas handoff arrivals are referred to as arrival type $N+i$.

For the sake of mathematical tractability we make the common assumptions of Poisson arrival processes and exponentially distributed random variables for cell residence time and call duration.

The arrival rate for new (handoff) calls of service i is λ_i^n (λ_i^h). A request of service i consumes b_i resource units, $b_i \in \mathbb{N}$.

The call duration of service i is exponentially distributed with rate μ_i^c. The cell residence time of a service i customer is exponentially distributed with rate μ_i^r. Hence, the resource holding time in a cell for service i is exponentially distributed with rate $\mu_i = \mu_i^c + \mu_i^r$.

Recent papers present more accurate modeling of the cell residence time [13], channel holding time [14, 15], arrival processes [16–18] and time within the handoff area [19, 20]. Logically, these models add an extra complexity to the analysis, making it highly intricate or simply infeasible. Some analytical results for the single service case are reported in [21–23]. Notwithstanding, the exponential assumption represents a good performance approximation. Essentially, only the average cell dwell time matters. When the average cell dwell time is small compared to the call duration, there is no expected difference between the exponential assumption and the gamma one. When cell dwell times are large, the difference becomes more noticeable, but the exponential assumption indicates general performance trends [24]. The exponential assumption can also be considered a good approximation for the time in the handoff area [25] and for the interarrival time of handoff requests [26].

Anyhow, the main contribution of this paper is an algorithm to determine the optimal capacity of the system, which relies on a method to compute the system blocking probabilities. Our proposal, however, does not depend on any specific method to find the blocking probabilities and hence it could be substituted — for instance if different assumptions are made for the underlying model — without affecting the proposed algorithm.

If we denote by $\boldsymbol{p} = (P_1, \ldots, P_{2N})$ the blocking probabilities for each of the $2N$ arrival streams, the new call blocking probabilities is $P_i^n = P_i$, the handoff failure probability is $P_i^h = P_{N+i}$ and the the forced termination probability of accepted calls under the assumption of homogeneous cell [1] is

$$P_i^{ft} = \frac{P_i^h}{\mu_i^c/\mu_i^r + P_i^h.}$$

The system state is described by an N-tuple $\boldsymbol{x} = (x_1, \ldots, x_N)$, where x_i represents the number of type i calls in the system, that were initiated either as new or handoff calls. Let $b(\boldsymbol{x})$ represent the amount of occupied resources at state $\boldsymbol{x}$, $b(\boldsymbol{x}) = \sum_{i=1}^{N} x_i b_i$.

2.1 Admission Policy (MFGC)

The MFGC policy operates in a manner that the maximum number of resource unit that stream i can dispose of is, on average, t_i. In order to decide on the acceptance of a request of type i, upon its arrival the system compares the amount of resources that will be occupied if it is accepted with the corresponding threshold t_i. The following decisions can be taken

$$b(\boldsymbol{x}) + b_i \begin{cases} \leq \lfloor t_i \rfloor & \text{accept} \\ = \lfloor t_i \rfloor + 1 & \text{accept with probability} \quad t_i - \lfloor t_i \rfloor \\ > \lfloor t_i \rfloor + 1 & \text{reject.} \end{cases}$$

3 Mathematical Analysis

The model of the system is a multidimensional birth-and-death process. The set of feasible states for the process is

$$S := \left\{ \boldsymbol{x} : \; x_i \in \mathbb{N}; \; \sum_{i=1}^{N} x_i b_i \leq C; \; x_i b_i \leq \lceil t_i \rceil, 1 \leq i \leq N \right\}$$

Let $r_{\boldsymbol{xy}}$ be the transition rate from $\boldsymbol{x}$ to $\boldsymbol{y}$ and let $\boldsymbol{e}_i$ denote a vector whose entries are all 0 except the i-th one, which is 1.

$$r_{\boldsymbol{xy}} = \begin{cases} a_i^n(\boldsymbol{x})\lambda_i^n + a_i^h(\boldsymbol{x})\lambda_i^h & \text{if } \boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e}_i \\ x_i\mu_i & \text{if } \boldsymbol{y} = \boldsymbol{x} - \boldsymbol{e}_i \\ 0 & \text{otherwise} \end{cases}$$

The coefficients $a_i^n(\boldsymbol{x})$ and $a_i^h(\boldsymbol{x})$ denote the probabilities of accepting a new and handoff call of service i respectively. Given a policy setting $(t_1, \ldots, t_{2N})$ these coefficients can be determined as follows

$$a_i^n(\boldsymbol{x}) = \begin{cases} 1 & \text{if } b(\boldsymbol{x}) + b_i \le \lfloor t_i \rfloor \\ t_i - \lfloor t_i \rfloor & \text{if } b(\boldsymbol{x}) + b_i = \lfloor t_i \rfloor + 1 \\ 0 & \text{if } b(\boldsymbol{x}) + b_i > \lfloor t_i \rfloor + 1 \end{cases}$$

and

$$a_i^h(\boldsymbol{x}) = \begin{cases} 1 & \text{if } b(\boldsymbol{x}) + b_i \le \lfloor t_i \rfloor \\ t_{N+i} - \lfloor t_{N+i} \rfloor & \text{if } b(\boldsymbol{x}) + b_i = \lfloor t_{N+i} \rfloor + 1 \\ 0 & \text{if } b(\boldsymbol{x}) + b_i > \lfloor t_{N+i} \rfloor + 1 \end{cases}$$

From the above, the global balance equations can be written as

$$p(\boldsymbol{x}) \sum_{\boldsymbol{y} \in S} r_{\boldsymbol{xy}} = \sum_{\boldsymbol{y} \in S} r_{\boldsymbol{yx}} p(\boldsymbol{y}) \qquad \forall \boldsymbol{x} \in S \tag{1}$$

Where $p(\boldsymbol{x})$ is the state $\boldsymbol{x}$ stationary probability. The values of $p(\boldsymbol{x})$ are obtained from (1) and the normalization equation. To obtain the stationary state distribution we used the Gauss-Seidel method. From the values of $p(\boldsymbol{x})$ the blocking probabilities are obtained as

$$P_i = P_i^n = \sum_{\boldsymbol{x} \in S} \big(1 - a_i^n(\boldsymbol{x})\big) p(\boldsymbol{x}) \qquad P_{N+i} = P_i^h = \sum_{\boldsymbol{x} \in S} \big(1 - a_i^h(\boldsymbol{x})\big) p(\boldsymbol{x})$$

If the system is in statistical equilibrium the handoff arrival rates are related to the new call arrival rates and the blocking probabilities (P_i) through the expression [2]

$$\lambda_i^h = \lambda_i^n \frac{1 - P_i^n}{\mu_i^c / \mu_i^r + P_i^h} \tag{2}$$

The blocking probabilities do in turn depend on the handoff arrival rates yielding a system of non-linear equations which can be solved using a fixed point iteration method as described in [1, 2].

4 Optimal Capacity: Algorithm

We pursue the goal of computing the system capacity, i.e. the maximum offered traffic that the network can handle while meeting certain QoS requirements. These QoS requirements are given in terms of upper-bounds for the new call blocking probabilities (B_i^n) and the forced termination probabilities (B_i^{ft}). Let $\lambda^T = \sum_{1 \le i \le N} \lambda_i^n$ be the aggregated call arrival rate and let f_i $(0 \le f_i < 1, \sum_{1 \le i \le N} f_i = 1)$ represent the fraction of λ^T that correspond to service i, i.e. $\lambda_i^n = f_i \lambda^T$, the capacity optimization problem can be formally stated as follows

Given: C, b_i, f_i, μ_i^c, μ_i^r, B_i^n, B_i^{ft} ; $i = 1, \ldots, N$

Maximize: λ^T

by finding the appropriate MFGC parameters t_i; $i = 1, \ldots, 2N$

Subject to: $P_i^n \le B_i^n$, $P_i^{ft} \le B_i^{ft}$; $i = 1, \ldots, N$

We propose an algorithm to work out this capacity optimization problem. Our algorithm has a main part (Algorithm 1 `capacity`) from which the procedure `solveMFGC` (see Algorithm 2) is called. The procedure `solveMFGC` does, in turn, calls another procedure (`MFGC`) that calculates the blocking probabilities. For the sake of notation simplicity we introduce the 2N-tuple $\boldsymbol{p_{max}} = (B_1^n, \ldots, B_N^n, B_1^h, \ldots, B_N^h)$ as the upper-bounds vector for the blocking probabilities, where the value of B_i^h is given by

$$B_i^h = \frac{\mu_i^c}{\mu_i^r} \frac{B_i^{ft}}{1 - B_i^{ft}} \tag{3}$$

Following the common convention bold-faced fonts were used font to represent array variables in the pseudo-code of the algorithms.

The algorithm `capacity` is essentially a binary search of λ_{max}^T that calls `solveMFGC` at each iteration to find out whether, for the tested value of the aggregated new call arrival rate λ^T, there exists a policy configuration ($\boldsymbol{t}$) that fulfills the QoS constraints ($\boldsymbol{p_{max}}$). If it exists (`solveMFGC` returns possible=TRUE) the lower limit of the interval that encloses λ_{max}^T is increased as $L := \lambda$; and otherwise (`solveMFGC` returns possible=FALSE) the upper limit of the interval is decreased as $U := \lambda$.

Algorithm 1 $(\lambda^{T}_{max}, \boldsymbol{t}_{opt})$=`capacity`$(\boldsymbol{p_{max}}, \boldsymbol{f}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C)$

```
$\varepsilon_1 :=<$ desired precision $>$
$L := 0$
$U :=<$ high value $>$
(possible, $\boldsymbol{t}$) := solve_MFGC($\boldsymbol{p_{max}}, U\boldsymbol{f}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)
atLeastOnce:=FALSE;

while possible do
  $L := U$
  $\boldsymbol{t}_L := \boldsymbol{t}$
  atLeastOnce:=TRUE
  $U := 2U$
  (possible, $\boldsymbol{t}$) := solve_MFGC($\boldsymbol{p_{max}}, U\boldsymbol{f}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)
end while{it makes sure that $U > \lambda^{T}_{max}$}

repeat
  $\lambda := (L + U)/2$
  (possible, $\boldsymbol{t}$) := solve_MFGC($\boldsymbol{p_{max}}, \lambda\boldsymbol{f}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)
  if possible then
    $L := \lambda$
    $\boldsymbol{t}_L := \boldsymbol{t}$
    atLeastOnce:=TRUE;
  else
    $U := \lambda$
  end if
until $(U - L)/L \leq \varepsilon_1$ AND atLeastOnce
$\lambda^{T}_{max} := L$
$\boldsymbol{t} := \boldsymbol{t}_L$
```

Algorithm 2 `(possible,`$\boldsymbol{t}$`)=solveMFGC(`$\boldsymbol{p_{max}}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$`)` (calculates MFGC parameters)

INPUTS: $\boldsymbol{p_{max}}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$

OUTPUTS: possible, $\boldsymbol{t}$

```

$\varepsilon_2$ :=< desired precision >
$\delta$ :=< small value >
$\boldsymbol{t} := (\delta, \delta, \ldots, \delta)$
$\boldsymbol{p}$ := MFGC($\boldsymbol{t}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)

repeat
  canConverge:=TRUE;
  $i := 1$;

  repeat
    if $\boldsymbol{p}(i) > \boldsymbol{p_{max}}(i)$ then
      $\boldsymbol{t'} := \boldsymbol{t}; \boldsymbol{t'}(i) := C$
      $\boldsymbol{p'}$ := MFGC($\boldsymbol{t'}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)

      if $\boldsymbol{p'}(i) > \boldsymbol{p_{max}}(i)$ then
        canConvege:=FALSE;
      else
        $L := \boldsymbol{t}(i); U := C$
        repeat
          $\boldsymbol{t}(i) := (L + U)/2$
          $\boldsymbol{p}$ := MFGC($\boldsymbol{t}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C$)
          if $\boldsymbol{p}(i) > \boldsymbol{p_{max}}(i)$ then
            $L := \boldsymbol{t}(i)$
          else
            $U := \boldsymbol{t}(i)$
          end if
        until $(1 - \varepsilon_2)\boldsymbol{p_{max}}(i) \leq \boldsymbol{p}(i) \leq \boldsymbol{p_{max}}(i)$
      end if

    end if
    $i := i + 1$
  until ($i > 2N$) OR ( NOT(canConverge))

  if canConverge then
    if $\boldsymbol{p}(i) \leq \boldsymbol{p_{max}}(i) \quad \forall i$ then
      possible:=TRUE; exit:=TRUE;
    else
      exit:=FALSE;
    end if
  else
    possible:=FALSE; exit:=TRUE;
  end if

until exit
```

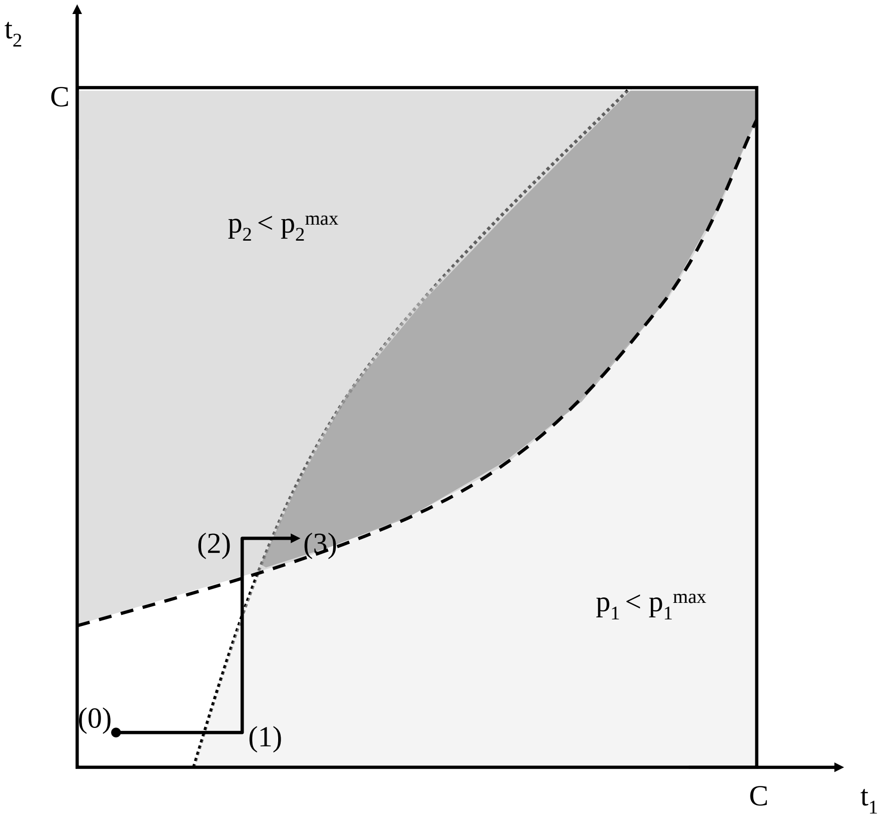

Figure 1: Graphical traces of a `solveMFGC` run, $\lambda_1^T \leq \lambda_{max}^T$.

In order to find a policy configuration that fulfills the QoS constraints, or decide that such configuration does not exist, the algorithm `solveMFGC` proceeds as follows. All t_i's are initialized with a small value $(\delta)^1$. Then the algorithm cyclically checks for each stream $i = 1, \ldots, 2N$ whether its QoS constraint $(\boldsymbol{p}(i) \leq \boldsymbol{p_{max}}(i))$ is met for the current policy setting, and if not $(\boldsymbol{p}(i) > \boldsymbol{p_{max}}(i))$ the value of t_i is increased so that $(1 - \varepsilon)\boldsymbol{p_{max}}(i) \leq \boldsymbol{p}(i) \leq \boldsymbol{p_{max}}(i)$. This process continues until either the QoS goal is achieved ($\boldsymbol{p}(i) \leq \boldsymbol{p_{max}}(i) \quad \forall i$ and the algorithm returns possible=TRUE); or the algorithm gives up as it realizes that the QoS goal is unattainable (then the algorithm returns possible=FALSE). The algorithm decides that the QoS goal is achieved if after a complete cycle $(i = 1, \ldots, 2N)$ the QoS constraint was met for all streams without requiring to increment the corresponding thresholds t_i. The algorithm decides that the QoS goal is unattainable if it happens that for a stream the QoS constraint can not be met even if the

[1] According to the philosophy of the algorithm an initial value of zero should have been used. However, a non-zero value was used due to implementation reasons.

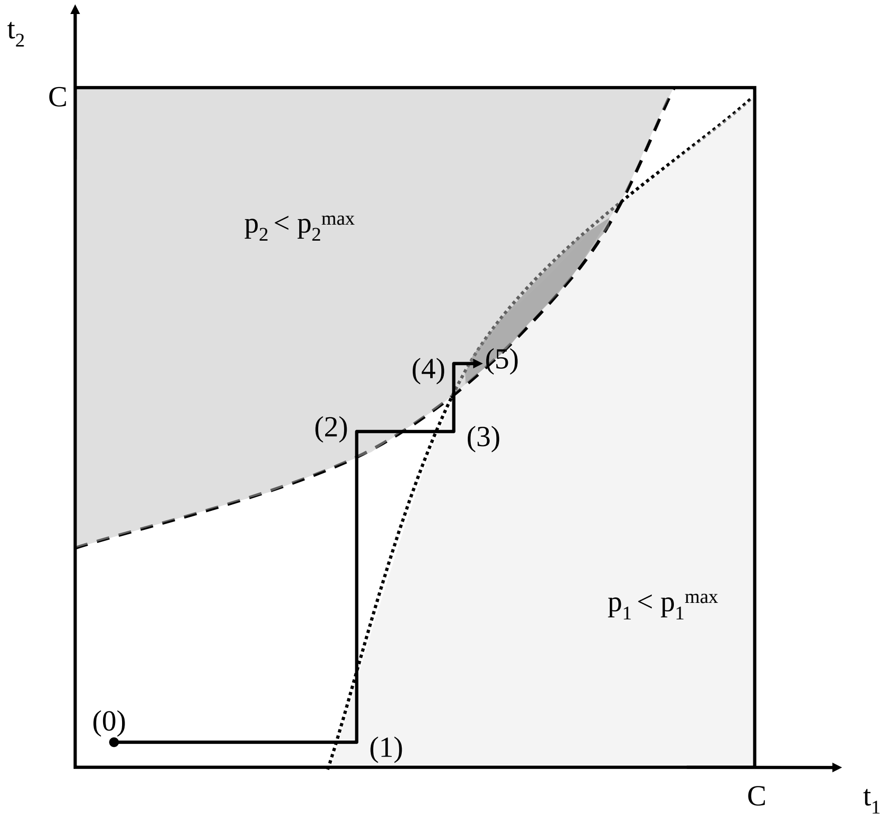

Figure 2: Graphical traces of a `solveMFGC` run, $\lambda_1^T \leq \lambda_2^T \leq \lambda_{max}^T$.

corresponding threshold is set to its maximum value $t_i = C$.

Figures 4 through 4 show an example illustrating the basic behavior of our algorithm. We used a rather simple case with only one type of service (two types of arrivals) in order to represent it graphically. Each figure represents one execution of the algorithm `solveMFGC` with a fixed value of λ^T. Figure 4 shows an example where the value λ^T of was relatively low and then the possible solutions of $\boldsymbol{t}$ was rather wide. Figure 4 shows another run of `solveMFGC` using a higher value of λ^T; again a policy setting that fulfills the QoS constraints can be found. Note, however, that increasing λ^T had the effect of shrinking the solution region. Finally, Fig. 4) shows an example where $\lambda^T > \lambda_{max}^T$ and then no feasible solution for $\boldsymbol{t}$ exists.

4.1 On the procedure `MFGC`

The procedure `MFGC`, which is invoked in the inner-most loop of our algorithm, is used to obtain the blocking probabilities ($\boldsymbol{p} :=$ MFGC$(\boldsymbol{t}, \boldsymbol{\lambda_n}, \boldsymbol{\mu_c}, \boldsymbol{\mu_r}, \boldsymbol{b}, C)$). For this computation

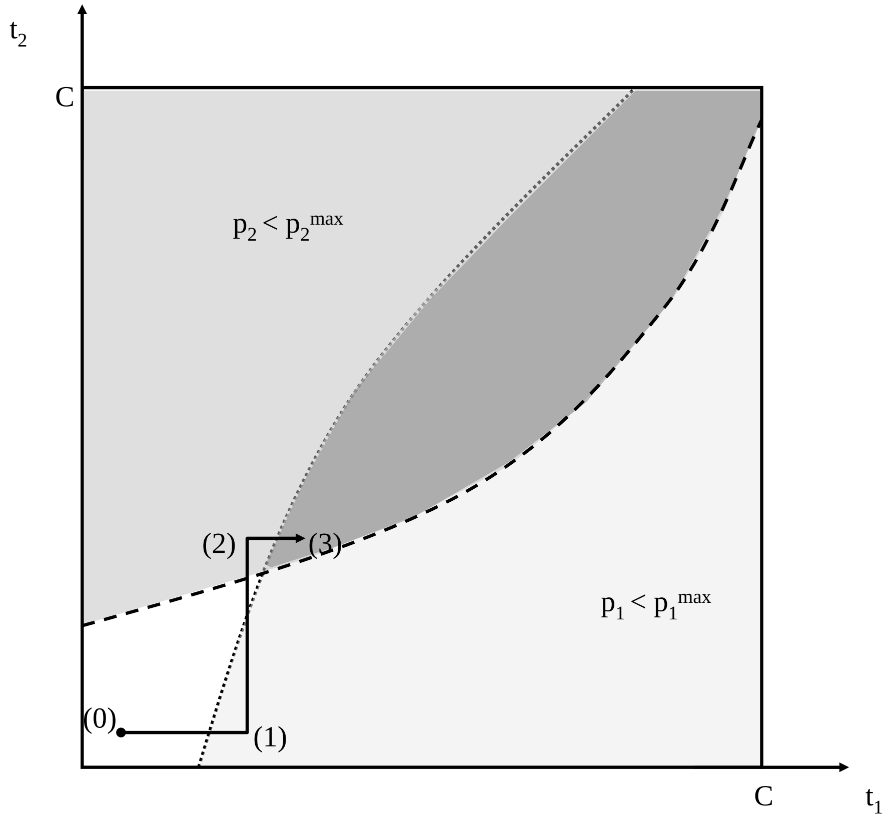

Figure 3: Graphical traces of a `solveMFGC` run, $\lambda_3^T > \lambda_{max}^T$.

an iterative procedure is required in order to obtain the value of the handoff request rates (see the end of Section 3). At each iteration a multidimensional birth-and-death process is solved. Solving this process, that in general will have a large number of states, constitutes the most computationally expensive part of the algorithm.

The following observation can be used to speed up the algorithm since it permits to eliminate the above mentioned iterative procedure. Each run of `solveMFGC` tries to find $\boldsymbol{t}$ so that $\boldsymbol{p} = \boldsymbol{p_{max}}$ (within tolerance limit). Thus, instead of using (2) to compute λ_i^h we use the expression

$$\lambda_i^h = \lambda_i^n \frac{1 - B_i^n}{\mu_i^c / \mu_i^r + B_i^h} \tag{4}$$

Although (2) and (4) look very similar there is a substantial difference between the two. In Eq. (4), λ_i^h is explicitly defined whereas in (2) it is not as P_i^n and P_i^h depend on λ_i^h.

5 Numerical Evaluation

In this section we evaluate the numerical complexity of our algorithm. To this end we used the algorithm proposed by Heredia et al. in [6–8] as a reference. Henceforth we refer to this algorithm as HCO after its authors' initials.

The HCO algorithm requires the optimal *prioritization order* as input, i.e. a list of call types sorted by their relative priorities [8]. If $\boldsymbol{t}$ is the policy setting for which the maximum capacity is achieved, the optimal prioritization order is the permutation $\boldsymbol{\sigma}^* \in \boldsymbol{\Sigma}$, $\boldsymbol{\Sigma} := \{(\sigma_i, \ldots, \sigma_{2N}) : \ \sigma_i \in \mathbb{N}, 1 \leq \sigma_i \leq 2N\}$, such that $t(\sigma_1^*) \leq t(\sigma_2^*) \leq \ldots \leq t(\sigma_{2N}^*) = C$. Selecting the optimal prioritization order is a complicated task as it depends on both QoS constraints and system characteristics as pointed out in [8]. In general there are a total of $(2N)!$ different prioritization orders. In [8] the authors give some guidelines to construct a partially sorted list of prioritization orders according to their likelihood of being the optimal ones. Then a trial and error process is followed using successive elements of the list until the optimal prioritization order is found. For each element the HCO algorithm is run and if after a large number of iterations it did not converged, another prioritization order is tried.

Our algorithm does not require any *a priori* knowledge. Indeed, after obtaining the policy setting $\boldsymbol{t}$ for which the maximum capacity is achieved, the optimal prioritization order is automatically determined as a by-product of our algorithm. This constitutes by itself a significant advantage of our algorithm over the HCO algorithm. Moreover, in what follows we show through numerical examples that our algorithm is still more efficient than the HCO algorithm when the latter is provided with the optimal prioritization order.

For the numerical examples we considered a system with two services ($N = 2$). Unless otherwise indicated, the values of the parameters are $\boldsymbol{b} = (1, 2)$, $\boldsymbol{f} = (0.8, 0.2)$, $\boldsymbol{\mu_c} = (1/180, 1/300)$, $\boldsymbol{\mu_r} = (1/900, 1/1000)$, $\boldsymbol{B}^n = (0.02, 0.02)$, $\boldsymbol{B}^{ft} = (0.002, 0.002)$; all tolerances have been set to $\varepsilon = 10^{-2}$. By (3), $\boldsymbol{B}^h \approx (0.01002, 0.00668)$ and then $\boldsymbol{p_{max}} \approx (0.02, 0.02, 0.01002, 0.00668)$.

A comparison of the number of floating point operations (flops) required by the HCO algorithm and our algorithm is shown in Table 1 and in Fig. 5. Both algorithms were tested with and without the speed-up technique (see Section 4.1) yielding a total of four

Table 1: Comparison of the HCO algorithm (with known prioritization order) and our algorithm with and without speed-up technique (figures in Mflops)

	HCO		our algorithm	
C	—	speed-up	—	speed-up
5	5.70	2.00	1.17	0.39
10	60.20	20.00	13.80	4.53
20	438.00	156.00	145.00	46.60

cases. The speed-up technique divides the flop count by a factor of about three.

To asses the impact of mobility on computational complexity, different scenarios were considered with varying mobility factors (μ_i^r/μ_i^c) for each service. The rest of the parameters have the same values as the ones used in the previous example, except μ_i^r which is varied to obtain four different mobility factor combinations: A) $\mu_1^r = 0.2\mu_1^c$, $\mu_2^r = 0.2\mu_2^c$; B) $\mu_1^r = 0.2\mu_1^c$, $\mu_2^r = 1\mu_2^c$; C) $\mu_1^r = 1\mu_1^c$, $\mu_2^r = 0.2\mu_2^c$; D) $\mu_1^r = 1\mu_1^c$, $\mu_2^r = 1\mu_2^c$. Computational cost results are displayed in Table 2 and aggregated costs across scenarios are plotted in Fig. 5. Again, our algorithm performs better than the HCO algorithm provided with the optimal prioritization order, and with the speed-up technique. The gain factor ranges from 1.4 to 7.8 with an average of 3.8, and in general it decreases when the number of resource units (C) increases.

It is worth noting that, as expected, the disagreement among the values obtained for the optimal capacity computed using the different methods was within tolerance in all tested cases. The same can be said for the policy setting $\boldsymbol{t}$.

6 Conclusions

We proposed a new algorithm for computing the optimal parameters setting of the of Multiple Fractional Guard Channel (MFGC) admission policy in multiservice mobile wireless networks. The optimal parameters setting maximizes the offered traffic that the system can handle while meeting certain QoS requirements. Compared to a recently published algorithm (HCO) ours offers the advantage of not needing a call prioritization order as input. We observed that a further enhancement of both algorithms is possible by eliminating the iterative procedure for computing the handoff call arrival rates. Numerical

Table 2: Comparison of the HCO algorithm (with known prioritization order) and our algorithm with speed-up technique for different mobility factors (figures in Mflops)

		C						
		5	10	15	20	25	30	**Total**
HCO (speed-up)	A	2.08	17.54	45.78	74.33	267.04	407.74	814.51
	B	2.67	14.06	50.25	147.13	266.41	487.41	967.93
	C	1.12	24.54	54.56	110.41	309.38	410.93	910.94
	D	2.24	16.86	53.39	121.39	106.49	462.12	762.49
	Total	8.11	73.00	203.98	453.26	949.32	1768.20	3455.8
our algorithm (speed-up)	A	0.35	4.42	18.51	53.64	119.46	199.69	396.07
	B	0.34	3.87	16.49	43.01	83.60	172.73	320.04
	C	0.38	3.93	17.59	47.95	119.33	191.66	380.84
	D	0.31	3.93	15.33	45.92	95.25	172.58	333.32
	Total	1.39	16.15	67.92	190.51	417.64	736.66	1430.3

examples show that our algorithm is faster than the HCO algorithm even if the latter is provided with the optimal prioritization order and is enhanced with the above mentioned observation.

References

[1] D. Hong and S. S. Rappaport, "Traffic model and performance analysis for cellular mobile radio telephone systems with prioritized and nonprioritized handoff procedures," *IEEE Transactions on Vehicular Technology*, vol. VT-35, no. 3, pp. 77–92, Aug. 1986, see also: CEAS Technical Report No. 773, June 1, 1999, College of Engineering and Applied Sciences, State University of New York, Stony Brook, NY 11794, USA.

[2] Y.-B. Lin, S. Mohan, and A. Noerpel, "Queueing priority channel assignment strategies for PCS hand-off and initial access," *IEEE Transactions on Vehicular Technology*, vol. 43, no. 3, pp. 704–712, Aug. 1994.

[3] F. Barceló, "Comparison of handoff resource allocation strategies through the state dependent rejection scheme," in *Proceedings of ITC 17*, 2001, pp. 323–334.

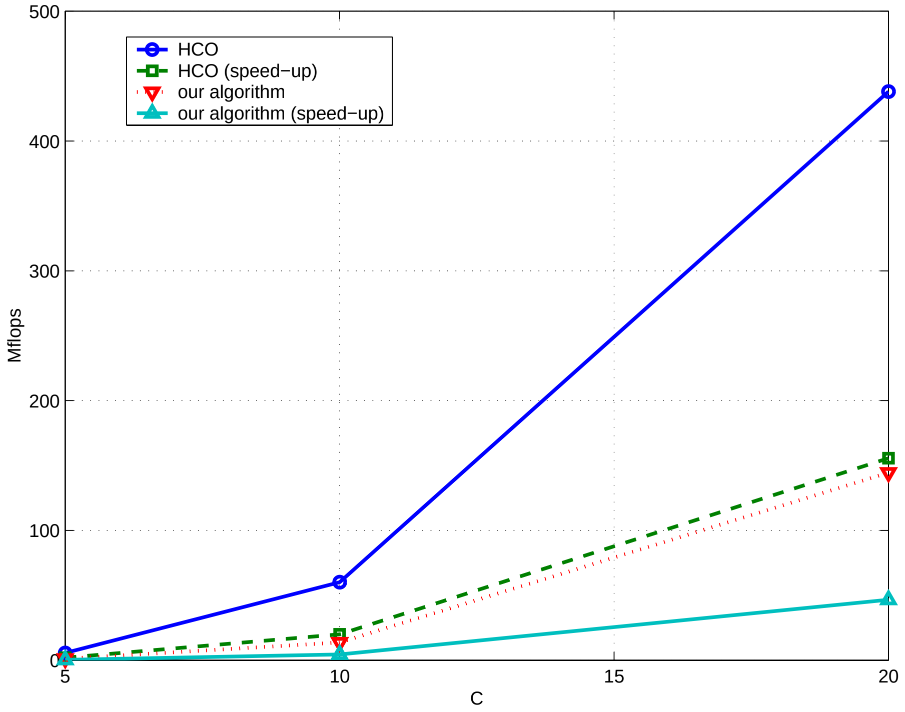

Figure 4: Comparison of the HCO algorithm and our algorithm with and without speed-up technique.

[4] B. Li, C. Lin, and S. T. Chanson, "Analysis of a hybrid cutoff priority scheme for multiple classes of traffic in multimedia wireless networks," *Wireless Networks Journal (WINET)*, vol. 4, no. 4, pp. 279–290, 1998.

[5] N. Bartolini and I. Chlamtac, "Call admission control in wireless multimedia networks," in *Proceedings of IEEE PIMRC*, 2002.

[6] H. Heredia-Ureta, F. A. Cruz-Pérez, and L. Ortigoza-Guerrero, "Multiple fractional channel reservation for optimum system capcity in multi-service cellular networks," *Electronics Letters*, vol. 39, no. 1, pp. 133–134, Jan. 2003.

[7] ——, "Multiple fractional channel reservation for multi-service cellular networks," in *Proceedings of IEEE ICC*, 2003, pp. 964 – 968.

[8] ——, "Capacity optimization in multiservice mobile wireless networks with multiple fractional channel reservation," *IEEE Transactions on Vehicular Technology*, vol. 52, no. 6, pp. 1519 – 1539, Nov. 2003.

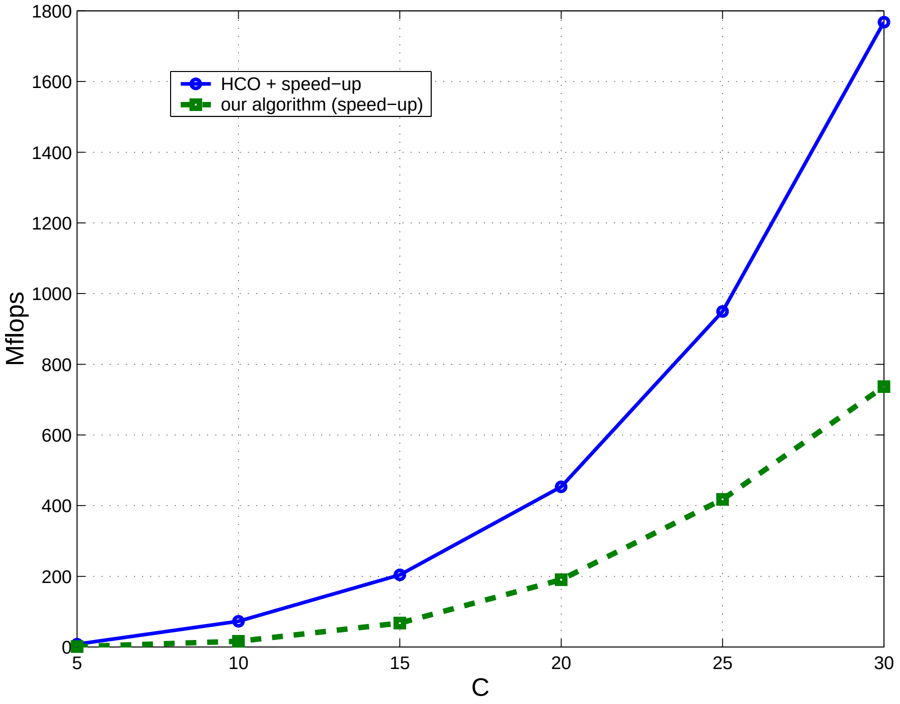

Figure 5: Comparison of the HCO algorithm and our algorithm with speed-up technique;aggregate values for scenarios A,B,C and D.

[9] R. Ramjee, R. Nagarajan, and D. Towsley, "On optimal call admission control in cellular networks," *Wireless Networks Journal (WINET)*, vol. 3, no. 1, pp. 29–41, 1997.

[10] N. Bartolini, "Handoff and optimal channel assignment in wireless networks," *Mobile Networks and Applications (MONET)*, vol. 6, no. 6, pp. 511–524, 2001.

[11] K. W. Ross, *Multiservice Loss Models for Broadband Telecommunication Networks.* Springer Verlag;, 1995.

[12] V. Pla and V. Casares-Giner, "Optimal admission control policies in multiservice cellular networks," in *Proceedings of the International Network Optimization Conference (INOC)*, 2003, pp. 466–471.

[13] M. M. Zonoozi and P. Dassanayake, "User mobility modeling and characterization of mobility patterns," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, pp. 1239–1252, Sept. 1997.

[14] F. Barceló and J. Jordán, “Channel holding time distribution in public telephony systems (PAMR and PCS),” *IEEE Transactions on Vehicular Technology*, vol. 49, no. 5, pp. 1615–1625, Sept. 2000.

[15] Y. Fang and I. Chlamtac, “Teletraffic analysis and mobility modeling of PCS networks,” *IEEE Transactions on Communications*, vol. 47, no. 5, pp. 1062–1072, July 1999.

[16] M. Rajaratnam and F. Takawira, “Handoff traffic characterization in cellular networks under nonclassical arrivals and service time distributions,” *IEEE Transactions on Vehicular Technology*, vol. 50, no. 4, pp. 954–970, July 2001.

[17] M. Sidi and D. Starobinski, “New call blocking versus handoff blocking in cellular networks,” *Wireless Networks Journal (WINET)*, vol. 3, no. 1, pp. 15–27, Mar. 1997.

[18] E. Chlebus and W. Ludwin, “Is handoff traffic really poissonian ?” in *Proceedings of ICUPC’95*, 1995, pp. 348–353.

[19] M. Ruggieri, F. Graziosi, and F. Santucci, “Modeling of the handover dwell time in cellular mobile communications systems,” *IEEE Transactions on Vehicular Technology*, vol. 47, no. 2, pp. 489–498, May 1998.

[20] V. Pla and V. Casares-Giner, “Analytical-numerical study of the handoff area sojourn time,” in *Proceedings of IEEE GLOBECOM*, Nov. 2002.

[21] M. Rajaratnam and F. Takawira, “Nonclassical traffic modeling and performance analysis of cellular mobile networks with and without channel reservation,” *IEEE Transactions on Vehicular Technology*, vol. 49, no. 3, pp. 817–834, May 2000.

[22] A. Alfa and W. Li, “PCS networks with correlated arrival process and retrial phenomenon,” *IEEE Transactions on Wireless*, vol. 1, no. 4, pp. 630–637, Oct. 2002.

[23] S. Dharmaraja, K. Trivedi, and D. Logothetis, “Performance modeling of wireless networks with generally distributed handoff interarrival times,” *Computer Communications*, vol. 26, pp. 1747–1755, 2003.

[24] F. Khan and D. Zeghlache, "Effect of cell residence time distribution on the performance of cellular mobile networks," in *Proceedings of VTC'97*. IEEE, 1997, pp. 949 – 953.

[25] V. Pla and V. Casares-Giner, "Effect of the handoff area sojourn time distribution on the performance of cellular networks," in *Proceedings of IEEE MWCN*, Sept. 2002.

[26] P. V. Orlik and S. S. Rappaport, "On the handoff arrival process in cellular communications," *Wireless Networks Journal (WINET)*, vol. 7, no. 2, pp. 147–157, March/April 2001.