Admission Control Policies in Multiservice Cellular Networks: Optimum Configuration and Sensitivity

David García, Jorge Martínez, and Vicent Pla

Universidad Politécnica de Valencia (UPV)
ETSIT, Camino de Vera s/n, 46022 Valencia, Spain
Phone: +34 963879733, fax: +34 963877309
dagarro@doctor.upv.es, (jmartinez, vpla)@dcom.upv.es

Abstract. We evaluate different call admission control policies in various multiservice cellular scenarios. For each of the studied policies we obtain the maximum calling rate that can be offered to the system to achieve a given QoS objective defined in terms of blocking probabilities. We propose an optimization methodology based on a hill climbing algorithm to find the optimum configuration for most policies. The results show that policies of the trunk reservation class outperform policies that produce a product-form solution and the improvement ranges approximately between 5 and 15% in the scenarios studied.

1 Introduction

Call Admission Control (CAC) is a key aspect in the design and operation of multiservice cellular networks that provide QoS guarantees. Different CAC strategies have been proposed in the literature which differ in the amount of information that the decision process has available. Obviously, as more information is fed into the process that decides if a new call is accepted or rejected both the system performance and the implementation complexity increase. We study a class of CAC policies for which the decision to accept a new call, say for example of service r, depends only on either the number of resource units occupied by the calls in progress of service r (which produce a product-form solution) or the number of free resource units in the system (trunk reservation). The physical meaning of a unit of resources will depend on the specific technological implementation of the radio interface.

Two sets of parameters are required to obtain the optimum configuration for these policies: those that describe the services as Markovian processes and those that specify the QoS objective. The QoS objective is defined in terms of the blocking probabilities for both new setup requests and handover requests. In a wireless scenario this distinction is required because a call being forced to terminate due to a handover failure is considered more harmful than the rejection of a new call setup request.

The configuration of a CAC policy specifies the action (accept/reject a new/handover request from each service) that must be taken in each system state in order to maximize the offered calling rate while meeting the QoS objective.

For the class of CAC policies under study, an important question that arises is how the performance of simple policies that produce a product-form solution compare to the performance of the trunk reservation policies and how sensitive these policies are to the tolerance of system parameters and to overloads. Surprisingly, and to the best of our knowledge, no studies have been published so far comparing the performance and sensitiveness of these policies when deployed in cellular access networks.

For a monoservice scenario, it has been shown in [1] that two trunk reservations polices named the *Guard Channel* and the *Fractional Guard Channel*[1] are optimum for common QoS objective functions. More recently, the multiservice scenario has been studied in [2], where using an approximate fluid model the optimum admission policy is also found to be of the trunk reservation class.

We study a representative selection of policies that produce a product-form solution, which include: i) *Complete Sharing*; ii) *Integer Limit* [3]; iii) *Fractional Limit* [4]; iv) *Upper Limit and Guaranteed Minimum* (ULGM) [5]; v) *Fractional Limit and Integer Limit*; vi) *ULGM and Integer Limit*; vii) *ULGM and Fractional Limit*; and, viii) *ULGM with Fractional and Integer Limits*. We also study a representative selection of trunk reservation policies, which include: ix) *Multiple Guard Channel* [6]; and, x) *Multiple Fractional Guard Channel* [7,8]. Details for all these policies are given in the next section.

For each policy and for each possible configuration of them we determine the maximum calling rate that can be offered to the system in order to satisfy the QoS objective. The result of this study is called the solution space and its peak value is called the system capacity for the CAC policy. We obtain the solution space for each policy in five different scenarios.

The main contributions of our work are: i) the determination of the optimum configuration for different multiservice CAC policies in various common scenarios using a novel hill-climbing algorithm; ii) the determination of the system capacity for each scenario (which is attained at the optimum configuration); and iii) the sensitivity analysis of the performance of the different policies to both the tolerance of the values of the configuration parameters and to overloads. The deployment of a hill-climbing algorithm drastically reduces the computational complexity of the process that determines the optimum configuration of a policy and, for example, compares quite favorably to the one proposed in [8] for the Multiple Fractional Guard Channel policy.

We use the theory of Markov Decision Processes (MDP) [10] along with linear programming techniques to obtain the optimum CAC policy and its configuration and we find that the policy is of the Randomized Stationary (RS) type [9]. This policy outperforms previous policies and is considered as a performance upper bound in the comparative study. Clearly the performance of the Complete Sharing policy defines the lower bound.

The remaining of the paper is structured as follows: in Section 2 we describe the model of the system as well as the CAC policies under study. In Section 3 we describe the RS policy in detail, formulating a method for the design of the CAC policy as the solution to a linear program. Section 4 justifies the applicability of the hill

[1] In [1] FGC is referred to as *Limited FGC*.

climbing algorithm for the determination of the optimum configuration of a policy. In Section 5 we compute the system capacity for each of the CAC policies under study. Section 6 discuses how the CAC policies behave under overload conditions. Finally, Section 7 concludes the paper.

2 System Model and Admission Control Policies

We consider a single cell, where a set of *R* services contend for *C* resource units. For any service *r* ($1 \le r \le R$), new setup requests arrive according to a Poisson process with mean rate λ_r^n and request c_r resource units per call. The duration of a service *r* call is exponentially distributed with rate μ_r^c, the cell residence time or dwell time of a service r call is exponentially distributed with rate μ_r^d. Hence, the holding time of the resources occupied by in a service r call is exponentially distributed with rate $\mu_r = \mu_r^c + \mu_r^d$. We will also consider that handover requests arrive according to a Poisson process with mean rate λ_r^h. Although the value of λ_r^h can be determined by a fixed point iteration method that balances the incoming and outgoing handover flows of a cell as described [11], we will suppose it is a known fraction of the value of λ_r^n. The QoS objective is expressed as blocking probabilities for both new setup requests (B_r^n) and handover requests (B_r^h).The handover blocking probabilities are related to the forced termination probabilities of accepted calls through the expression [11]

$$B_r^{ft} = \frac{B_r^h}{\left(\mu_r^c / \mu_r^d\right) + B_r^h}$$

Let the system state vector be $n \equiv (n_1^n, n_1^h, ..., n_R^n, n_R^h)$, where $n_r^{n,h}$ is the number of service r calls in progress in the cell that where initiated as a successful setup request or a handover request respectively. We will denote by $c(n) = \sum_{r=1}^{R} \left(n_r^n + n_r^h\right) \cdot c_r$ the number of busy resource units in state *n*. Let S_P be the state space under policy *P*. For example, for the Complete Sharing policy $S_{CS} = \left\{ n \in N^{2R} \middle| \sum_{r=1}^{R} \left(n_r^n + n_r^h\right) \cdot c_r \le C \right\}$. The stochastic process $n(t)$, which gives the system state at time *t*, is an irreducible finite-state continuous-time Markov chain with a unique steady-state probability vector π.

The definition of each CAC policy under study is as follows:

1. Complete Sharing (CS). A request is admitted provided there are enough free resource units available in the system.
2. Integer Limit (IL). Two parameters are associated with service *r:* l_r^n for new setup requests and l_r^h for handover requests, $l_r^n, l_r^h \in N$. A service *r* request that arrives in state *n* is accepted if $(n_r^{n,h} + 1) \le l_r^{n,h}$ and blocked otherwise.
3. Fractional Limit (FL). Four parameters are assigned to service *r*: $t_r^n, t_r^h \in N$ and $q_r^n, q_r^h \in [0,1]$. A service *r* request is accepted with probability one if $(n_r^{n,h} + 1) \le t_r^{n,h}$, otherwise new setup requests are accepted with probability q_r^n and handover requests with probability q_r^h .
4. Upper Limit and Guaranteed Minimum (ULGM). Service *r* requests have access to two sets of resources: a private set and a shared set. The number of resource units in the private set available for new setup requests is denoted as $(s_r^n \cdot c_r)$ and for

handover requests as $(s_r^h \cdot c_r)$, $s_r^n, s_r^h \in N$. Therefore the size of the shared set is $C - \sum_{r=1}^{R} (s_r^n + s_r^h) \cdot c_r$. A service *r* request is accepted if $(n_r^{n,h}+1) \le s_r^{n,h}$ or if there are enough free resource units in the shared set, otherwise it is blocked.

5. Fractional Limit and Integer Limit (FL+IL). Six parameters are associated with service *r*: $t_r^n, t_r^h, l_r^n, l_r^h \in N$, $q_r^n, q_r^h \in [0,1]$. A service *r* request is accepted with probability one if $(n_r^{n,h}+1) \le t_r^{n,h}$, it is accepted with probability $q_r^{n,h}$ if $t_r^{n,h} < (n_r^{n,h}+1) \le l_r^{n,h}$, and blocked otherwise.
6. ULGM and Integer Limit (ULGM+IL). Four parameters are associated with service *r*: $s_r^n, s_r^h, l_r^n, l_r^h \in N$. A service *r* request is accepted if $(n_r^{n,h}+1) \le s_r^{n,h}$ or if there are enough free resource units in the shared set and $(n_r^{n,h}+1) \le l_r^{n,h}$, and blocked otherwise.
7. ULGM and Fractional Limit (ULGM+FL). Six parameters are associated with service *r*: $s_r^n, s_r^h, t_r^n, t_r^h \in N$ and $q_r^n, q_r^h \in [0,1]$. A service *r* request is accepted with probability one if $(n_r^{n,h}+1) \le s_r^{n,h}$ or if there are enough free resource units in the shared set and $(n_r^{n,h}+1) \le t_r^{n,h}$, it is accepted with probability $q_r^{n,h}$ if there are enough free resource units in the shared set and $(n_r^{n,h}+1) > t_r^{n,h}$, and blocked otherwise.
8. ULGM with Fractional and Integer Limits (ULGM+FL+IL). Eight parameters are associated with service *r*: $s_r^n, s_r^h, t_r^n, t_r^h, l_r^n, l_r^h \in N$ and $q_r^n, q_r^h \in [0,1]$. A service *r* request is accepted with probability one if $(n_r^{n,h}+1) \le s_r^{n,h}$ or if there are enough free resource units in the shared set and $(n_r^{n,h}+1) \le t_r^{n,h}$, it is accepted with probability $q_r^{n,h}$ if there are enough free resource units in the shared set and $t_r^{n,h} < (n_r^{n,h}+1) \le l_r^{n,h}$, and blocked otherwise.
9. Multiple Guard Channel (MGC). Two parameters are associated with service *r:* $l_r^n, l_r^h \in N$. A service *r* request that arrives in state *n* is accepted if $c(n) + c_r \le l_r^{n,h}$ and blocked otherwise.
10. Multiple Fractional Guard Resource unit (MFGC). Four parameters are associated with service *r:* $t_r^n, t_r^h \in N$ and $q_r^n, q_r^h \in [0,1]$. A service *r* request that arrives in state *n* is accepted with probability one if $c(n) + c_r < t_r^{n,h}$, accepted with probability $q_r^{n,h}$ if $c(n) + c_r = t_r^{n,h}$, and blocked otherwise.
11. Randomized Stationary (RS). Each system state is assigned with a set of probabilities $q(n) = \{ q_1^n(n), q_1^h(n), ..., q_R^n(n), q_R^h(n) \}$, $q(n) \in [0,1]^{2R}$. A service *r* request that arrives in state *n* is accepted with probability $q_r^{n,h}(n)$.

3 Randomized Stationary (RS) Policies

In this section we redefine the notation of the system state vector as $x \equiv (x_1, ..., x_R)$, where $x_r = n_r^n + n_r^h$. In memoryless systems this redefinition does not suppose a loose of generality because once a call of service *r* has been accepted in a cell, the time the resources will be held is a random variable which distribution is independent of the fact that the call had been initiated as new or as handover. Therefore the state space is $S_{RS} = \{ x \in N^R \mid \sum_{r=1}^{R} x_r \cdot c_r \le C \}$. Each state has an associated set of actions $A(x) \in A$, where A is the set of all actions, $A \equiv \{ a = (a_1, ..., a_R) : a_r = 0,1,2 \}$. Element a_r of an action a encodes how service r requests are handled, and its meaning

is as follows: $a_r = 0$ reject both new and handover calls; $a_r = 1$ accept handover calls and reject new calls; and $a_r = 2$ accept both new and handover calls. When an RS policy is applied, one of the possible actions $A(x)$ is chosen at random according to the probability distribution $p_x(a)$, $a \in A(x)$, each time the process visits state x. The transition rate between states depends on the action chosen. Let $r_{xy}(a)$ denote the transition rate between states x and y when action a is chosen. Transitions rates can be expressed as:

associated to arrivals ($y = x + e_r$) associated to departures ($y = x - e_r$)

$$r_{xy}(a) = \begin{cases} 0 & \text{if } a_r = 0 \\ \lambda_r^h & \text{if } a_r = 1 \\ \lambda_r^n + \lambda_r^h & \text{if } a_r = 2 \end{cases} \qquad r_{xy}(a) = x_r \mu_r$$

where r denotes the service type and e_r is a vector whose entries are all 0 except for the r-th one which is 1.

The continuous time process is converted to a discrete time one by applying the uniformization approach. This is possible since a uniform upper bound Γ can be found for the total outgoing rate from each state, where $\Gamma = \sum_{r=1}^{R} (\lambda_r^n + \lambda_r^h + C\mu_r)$. The transition probabilities for the uniformized discrete time Markov chain can be written as:

$$p_{xy}(a) = \begin{cases} r_{xy}(a)/\Gamma & y \neq x \\ 1 - \sum_{z \in S_{RS}} p_{xy}(a) & y = x \end{cases}$$

Let us define the following cost functions:

$$c_r^n(a) = \begin{cases} 1 & \text{if } a_r = 0,\ 1 \\ 0 & \text{if } a_r = 2 \end{cases} \qquad c_r^h(a) = \begin{cases} 1 & \text{if } a_r = 0 \\ 0 & \text{if } a_r = 1,\ 2 \end{cases}$$

Note that the cost associated to any action is independent of the state, i.e. $c_r^{n,h}(x,a) = c_r^{n,h}(a) \forall x$. Cost functions are defined in such a way that their time average equals the corresponding blocking probability, i.e.

$$p_r^{n,h} = \lim_{k \to \infty} \frac{\left(E\left[\sum_{t=0}^{k} c_r^{n,h}\left(x(t), a(t)\right) \right] \right)}{(k+1)}$$

where $x(t)$ and $a(t)$ represent the state and action at time t.

Let $p(x)$ denote the stationary probability of state x. If we introduce $p(x,a) = p(x)p_x(a)$, which denotes the probability of being in state x and choosing action x, then the following equation holds: $p(x) = \sum_{a \in A(x)} p(x,a)$.

3.1 Constrains

We now define several constraint sets that are subsequently used to formulate the design criterion:

S0

$$\sum_{a\in A(x)} p(x,a) = \sum_{\substack{y\in S_{RS}\\ a\in A(y)}} p_{yx}(a)p(y,a), \quad x \in S_{RS}$$

$$\sum_{\substack{x\in S_{RS}\\ a\in A(x)}} p(x,a) = 1 \ , \quad p(x,a) \geq 0$$

Constraints in S0 stem from the associated Markov chain equations.

S1

$$\sum_{\substack{x\in S_{RS}\\ a\in A(x)}} p(x,a)\cdot c_r^J(x,a) \leq B_r^J \ , \quad J = n,h, \quad r = 1,...,R$$

Parameters $B_r^{n,h}$ represent the maximum allowed values for the blocking probabilities.

3.2 Design Criterion.

The design criterion considered here is made up of an objective function, which is to be minimized, plus the constraint sets defined above. As both objective functions and constraints are linear, the design problem can be formulated as a linear program. Thus the simplex method or other well-known algorithms can be used to solve the linear program. Different design criterion can be used [9] but in this work we only use the following:

Minimize

$$\sum_{\substack{r=\\ x\in S_{RS}\\ a\in A(x)}}^{R} p(x,a)\left(c_r^n(x,a) + c_r^h(x,a)\right)$$

subject to: S0 and S1

The solution of the linear program yields the values of $p(x,a)$ from which the steady-state probabilities $p(x)$ and the CAC policy configuration $p_x(a)$ are obtained; note that the values of $p_x(a)$ can be readily mapped to the RS policy description given in Section 2. In this way, the blocking probability for each service is minimized and upper bounded by the QoS objective. From a practical perspective it is worth noting that there may not exist a feasible solution if the value of C is not high enough. Finding the minimum value of C so that a feasible solution exists or, equivalently, finding the maximum offered traffic so that a feasible solution exists for a given value of C, are typical problems at the planning phase that can be solved by applying this design criterion.

4 Determination of the Optimum Policy Configuration

The common approach to carry out the CAC synthesis process in multiservice systems is by iteratively executing an analysis process. We refer to as synthesis process a process that having as inputs the values of the system parameters ($\lambda_r^{n,h}, \mu_r, c_r$ and C) and the QoS objective ($B_r^{n,h}$), produces as output the optimum configuration (the thresholds $l_r^{n,h}, t_r^{n,h}, s_r^{n,h}$) for a given CAC policy. In contrast the analysis process is a process that having as inputs the value of the system parameter and the configuration

for a given CAC policy produces as output the blocking probabilities for the different services.

Given that in general, the blocking probabilities are non-monotonic functions both of the offered load and the thresholds that specify most policy configurations; the common approach is to carry out a multidimensional search using for example meta-heuristics like genetic algorithms which are able to find a *good* configuration in a reasonable amount of time. It should also be pointed out that each execution of the analysis process requires solving the associated continuous-time Markov chain, for which we use the Gauss-Seidel algorithm when the solution has not a product-form. As shown before, the first eight policies are created by defining multiple thresholds with multiple weights for each traffic stream. It is shown [12] that these policies result in product-form solutions.

In this respect, formulating the problem of finding the optimum CAC policy by the theory of MDPs has as advantage that both the value of the system parameters and the QoS objective become part of the inputs and as output we obtain the optimum configuration. Therefore no additional search is required.

The study of the optimum configuration for each policy is done for five different scenarios (A, B, C, D and E) that are defined in Table 1, where the $B_r^{n,h}$ are specified as percentages. The parameter f_r represents the percentage of service r new call requests, i.e., we suppose that the aggregated new call request rate is $\lambda = \sum_{r=1}^{R} \lambda_r^n$, $\lambda_r^n = f_r \lambda$. This simplification reduces the complexity of the multidimensional search and seems to be a common approach among operators. The parameters in Table 1 have been selected to explore possible trends in the numerical results, i.e., taking scenario A as a reference, scenario B represents the case where the ratio c_1 / c_2 is smaller, scenario C where f_1 / f_2 is smaller, scenario D where B_1 / B_2 is smaller and scenario E where B_1 and B_2 are equal.

Table 1. Definition of the scenarios under study.

	A	B	C	D	E
c_1	1	1	1	1	1
c_2	2	4	2	2	2
f_1	0.8	0.8	0.2	0.8	0.8
f_2	0.2	0.2	0.8	0.2	0.2
B_1^n %	5	5	5	1	1
B_2^n %	1	1	1	2	1
	A, B, C, D, E				
B_r^h %	$0.1 B_r^n$				
λ_r^n	$f_r \lambda$				
λ_r^h	$0.5 \lambda_r^n$				
μ_1	1				
μ_2	3				

In order to illustrate the algorithm we have chosen a simple example with only two service classes but without their associated handover streams. This allows us to repre-

sent the solution space in only three dimensions. Fig. 1 and Fig. 2 show the solution spaces when policies IL and MGC are deployed in scenario A with $C = 10$ resource units (recall that $c_1 = 1$, $c_2 = 2$, $B_1 = 5\%$ and $B_2 = 1\%$).

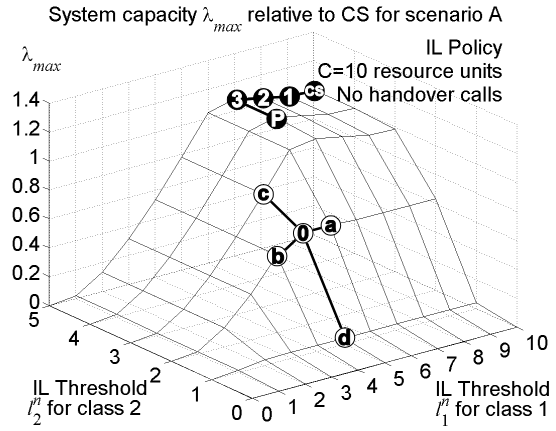

Fig. 1. Example of the use of a hill-climbing algorithm to determine the optimum configuration for the IL policy.

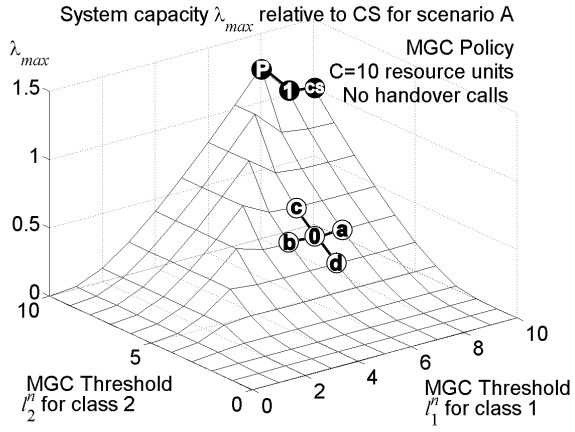

Fig. 2. Example of the use of a hill-climbing algorithm to determine the optimum configuration for the MGC policy.

The configuration of both policies is defined by two parameters l_1 and l_2. Remember that when deploying the IL policy a call setup request from service *r* is accepted only if $(n_r + 1) \le l_r$. On the other hand, when deploying the MGC policy a call setup request from service *r* is accepted only if $c(n) + c_r \le l_r$. These two policies were

selected because they have solution spaces with shapes that can be considered as representing two different families (with and without product form solution respectively). The MGC has been chosen instead of the MFGC policy, because it has the same number of configuration parameter as the IL policy.

The form of the solution spaces shown in Fig.1 and Fig.2 suggests that a hill-climbing algorithm could be an efficient approach to obtain the optimum configuration for most CAC policies. The hill-climbing algorithm works as follows. Given an starting point in a *k*-dimensional discrete search space (for example, point 0 in both figures), the hill-climbing algorithm begins by computing the value of the function (the system capacity λ_{max}) for the two adjacent neighbors in each of the *k* dimensions (points a, b, c and d in both figures). Then the algorithm selects as the new starting point the adjacent point with the largest function value (point c in both figures) and the process repeats iteratively until a local maximum is found. In this way the algorithm makes a number of successive unitary steps along each dimension of the search space and stops when it reaches the peak.

As defined in Section 1, each point of the solution space defines the maximum aggregated call request rate ($\lambda = \lambda_1 + \lambda_2$, $\lambda_r = f_r \lambda$) that can be offered to the system in order to satisfy the QoS objective. The plotted surfaces are obtained as follows. At each point, λ_{max} is computed by a binary search process which has as input the value of the system parameters μ_r, c_r, C and the thresholds l_r, and produces as output the blocking probabilities p_r. The binary search process stops when it finds the λ_{max} that meets the QoS objective B_r, r=1,…, R.

It should be noted that the system capacity is expressed as a relative value to the capacity obtained for the CS policy. When the solution space is continuous, for example for the MFGC policy, a gradual refinement process is used to reduce the size of the step once a promising region has been found, which is possibly close to the optimum.

A further reduction of the computation complexity can be obtained by observing that the optimum configuration (point P) for any policy is near the CS configuration (point CS), and therefore it is a good idea to select it as the starting point. In Fig. 1 points 1 to 3 and in Fig.2 point 1 illustrate a typical progression of the algorithm starting from the CS configuration and ending at the peak.

To gain additional insight into the problem we show the solution spaces for the IL and MGC policies in the scenario A, now considering their associated handover streams. Given that each policy has now a four-dimensional solution space in the scenario under study, an appropriate representation is required. We have chosen to represent slices of the solution space: variation of system capacity λ_{max} (expressed as a relative value to the capacity obtained for the CS policy) as a function of each configuration parameter (l_1^n, l_1^h, l_2^n and l_2^h) while keeping the others constant at their optimum values.

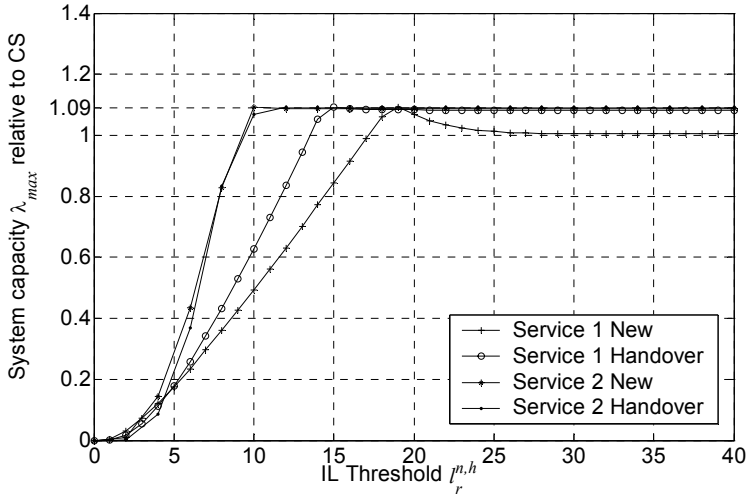

Fig. 3. Two-dimensional representation of the solution space for the IL policy in the scenario A with C=40.

For the IL policy, Fig. 3 plots the slices of the solution space in scenario A with C=40. As it could be expected, the values of the configuration parameters at which the peak for λ_{max} (1.09) is achieved are different. An interesting observation is that the peak value seems insensitive to the values of some configuration parameters in a quite large region. Unfortunately, the peak value seems sensitive to the value of l_1^n, that is, to the threshold defined for new calls of service 1.

This can be intuitively explained noting that if the system is loaded with λ_{max} and we deploy the CS policy then the blocking probability achieved for the new calls of service 1 (p_1^n) is far below its objective, while the blocking probabilities achieved by the other traffic streams are much closer to their objectives. In this scenario we can increase the capacity of the system by restricting the access to the new calls of service 1 (by decreasing its threshold if deploying the IL policy) and offering the new available capacity to the other traffic streams. Therefore, if a value larger than the optimum one is chosen for l_1^n then the performance of the IL policy approximates the performance of the CS policy, but if a value smaller than the optimum one is chosen for l_1^n then the performance of the IL policy can be even worse than the performance of the CS policy.

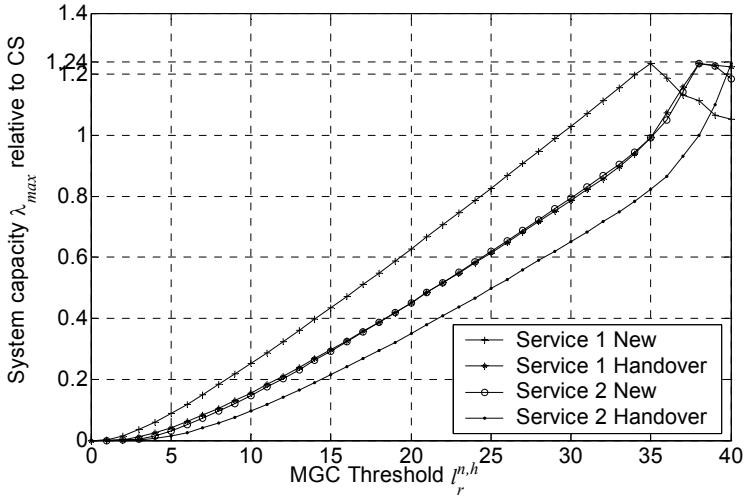

Fig. 4. Two-dimensional representation of the solution space for the MGC policy in the scenario A with C=40.

Figure 4 plots the slices of the solution space for the MGC policy. As it could be expected, the values of the configuration parameters at which the peak for λ_{max} (1.24) is achieved are different. For this policy, there is no plateau but a clear half-pyramidal or near-half-pyramidal shape with the maximum located at the apex. In this case, the position of the apex is of capital significance since not far away from this point the system capacity is poor compared to the one obtained for the CS policy. This solution space suggests that the definition of the configuration parameters requires more precision, unless we are willing to accept a degradation of the system capacity. Nevertheless, the slopes of the curves are now less steep. It should also be noted that, as mentioned before, the optimum configuration is close to the CS configuration (all thresholds set to *C*).

5 System Capacity

In this section we obtain the system capacity that can be expect when deploying one of the CAC policies under study. As defined before, the system capacity is the maximum arrival rate of new calls ($\lambda = \sum_{r=1}^{R} \lambda_r^n$) that can be offered to the system while meeting the QoS requirements, i.e. that produces blocking probabilities lower or equal than the $B_r^{n,h}$.

Table 2. System capacity for the CAC policies under study.

	C	CS	IL	ULGM	FL	FL +IL	ULGM +FL	ULGM +IL	ULGM +FL+IL	MGC	FMGC	RS
A	10	1.54	1.13	1.07	1.13	1.14	1.18	1.17	1.18	1.23	1.33	1.34
	20	5.61	1.11	1.06	1.12	1.13	1.15	1.14	1.16	1.26	1.31	1.31
	40	15.76	1.09	1.07	1.10	1.10	1.14	1.13	1.14	1.24	1.25	1.26
B	10	0.37	1.05	1.00	1.18	1.18	1.18	1.05	1.19	1.10	1.15	1.20
	20	2.78	1.03	1.08	1.11	1.11	1.12	1.09	1.13	1.21	1.25	1.25
	40	10.39	1.06	1.04	1.07	1.10	1.09	1.06	1.10	1.21	1.22	1.23
C	10	1.37	1.07	1.05	1.07	1.09	1.10	1.08	1.10	1.11	1.21	1.22
	20	5.77	1.05	1.05	1.08	1.08	1.07	1.06	1.09	1.20	1.21	1.21
	40	17.62	1.05	1.04	1.07	1.07	1.07	1.06	1.07	1.15	1.16	1.16
D	10	1.74	1.03	1.00	1.06	1.06	1.04	1.03	1.06	1.13	1.16	1.17
	20	6.05	1.04	1.00	1.05	1.05	1.05	1.04	1.05	1.13	1.15	1.15
	40	16.54	1.03	1.00	1.04	1.04	1.00	1.03	1.04	1.10	1.11	1.11
E	10	1.54	1.05	1.00	1.06	1.06	1.05	1.05	1.06	1.13	1.17	1.17
	20	5.62	1.04	1.02	1.05	1.05	1.06	1.05	1.06	1.13	1.15	1.16
	40	15.7	1.03	1.02	1.04	1.04	1.05	1.04	1.05	1.10	1.10	1.10

The study is done for the five scenarios described in Table 1. Results for the capacity are displayed in Table 2. They are expressed as relative values in relation to the capacity obtained for the CS policy, while for this policy we display absolute values. The maximum traffic that can be offered by each service can be easily determined from the parameters in Table 1 once the system capacity has been obtained.

As observed, the trunk reservation policies perform better than those that produce a product-form solution and, in the scenarios studied, the improvement ranges between 5 and 15% approximately, although in some configurations can be a bit lower and in others a bit higher. However, the relative gain diminishes when the number of resource units C increases.

In general, the implementation complexity is not an issue in these types of policies because at the most they only require storing a reduced set of parameters per service. For the RS algorithm, it can be shown that the maximum number of variables that need to be stored are the number of states plus $4R$ [9].

6 Sensitivity to Overloads

We study how the achieved handover blocking probabilities p_r^h of all services increase with different degrees of overload (1% to 500%), which is defined as the ratio of the offered aggregated calling rate of new calls to the λ_{max} that allows the system to meet the QoS requirements. It should the noted that the components of the offered aggregated calling rate are determined by $\lambda_r^n = f_r\lambda$ as shown in Table 1, and there-

fore, as we increase the offered aggregated calling rate λ their relative ratios are maintained.

Given a policy and a configuration, the limiting service (LS) is the one for which its achieved p_r^h or p_r^n prohibits to increase λ_{max}. When deploying any of the policies that produce a product-form solution and the system is loaded with λ_{max}, typically, the achieved p_r^h of the LS is the one closest to its objective, while for some very few particular cases it is the p_r^n of the LS which is the one closest to its objective. When overload occurs, typically, the p_r^h of the LS goes above its objective, while for the very few particular cases mentioned before it is the p_r^n of the LS which goes above its objective, while all the p_r^h remain below their objectives for a certain range of overload.

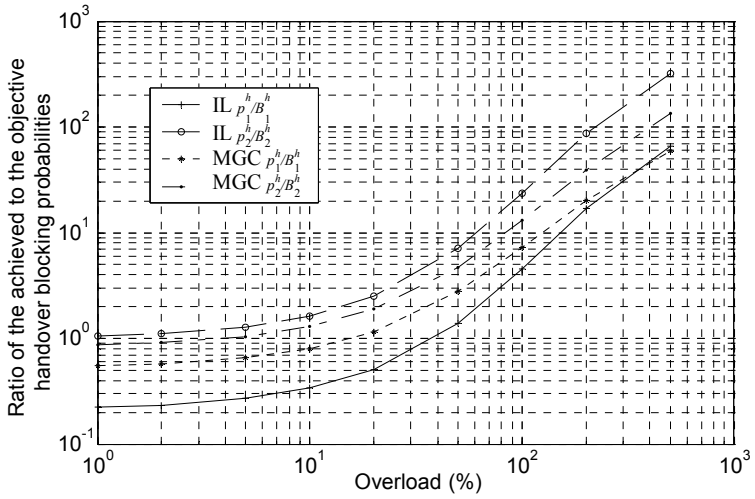

Fig. 5. Ratio of the achieved to the objective handover blocking probabilities in the scenario B with *C*=20

When deploying any of the last three policies and the system is loaded with λ_{max}, all the achieved blocking probabilities $p_r^{n,h}$ are quite close to their objectives, and this is specially true for the last two. Under a certain degree of overload, the p_r^h achieved by the LS is typically smaller than the p_r^h achieved by the LS of any of the policies that produce a product-form solution, except for the very few particular cases just described.

The increase ratio for the p_r^h with a given increase of the overload ratio tends to be similar for all policies. All the comments of this section are especially applicable in scenarios with C=20 or 40 bandwidth units.

Fig. 5 shows the ratio of the achieved to the objective handover blocking probabilities for each service class and degree of overload. For example, for the IL policy and a 10% of overload the achieved handover blocking probability of service class 1, p_1^h, is 33.75% of its objective ($B_1^h = 5\%$).

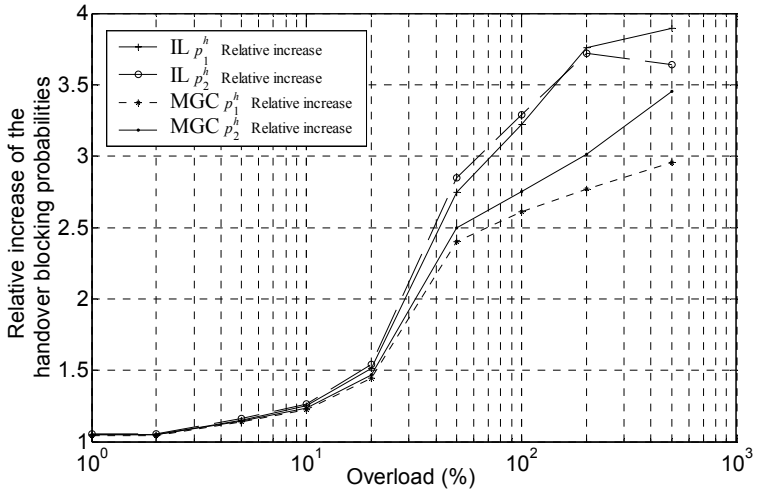

Fig. 6. Relative increase of the handover blocking probabilities in the scenario B with C=20

Fig. 6 shows the relative increase of the p_r^h in relation to the previous degree of overload. For example, for the IL policy the achieved handover blocking probability of service class 1 for an overload of 10% is 25.23% higher (1.2523) than the achieved handover blocking probability of service class 1 for an overload of 5%.

Finally, two main conclusions can be drawn from the sensitivity analysis to overloads. First, in general, deploying an admission control policy is convenient because it introduces a certain degree of fairness in sharing among the different services the penalty due to the increase of the blocking probabilities during overloads. Second, in general, the last three policies tend to handle the overload similarly to (in the low to medium overload region) or better than (in the high overload region) those which produce a product-form solution, in the sense that the relative increase of the achieved handover blocking probabilities for the last three is lower than the relative increase for the policies that produce a product-form solution.

7 Conclusions

We have determined the maximum system capacity that can be expected when using different CAC policies. Due to the multidimensionality and non-monotonic behavior of the system under study the determination of the optimum configuration becomes difficult and computationally costly. The form of the solution spaces suggested us the use of a hill-climbing algorithm to reduce the computational complexity of the procedure that obtains the optimum configuration for each admission policy.

For the system capacity, numerical results show that policies of the trunk reservation class outperform policies that produce a product-form solution and the improve-

ment ranges approximately between 5 and 15% in the scenarios studied. However, the relative gain diminishes when the number of resource units C increases. In addition, the shape of the solutions space of the trunk reservation policies shows that higher precision is required when determining the optimum configuration with respect to the product-form policies. Given that in practice the system parameters must be estimated and are non-stationary, trunk reservation policies could become less attractive.

We have also studied the sensitivity of the policies to overloads and found that, in general, trunk reservation policies handle the overload better.

It is clear that the results of our study have been obtained in quite idealistic conditions that might be difficult to achieve in a real operating system, nevertheless the study provides novel insights to a difficult problem.

Future work should address the study of the solution space for the RS policy, although intuition suggests that it could have a shape similar to the one found for the MFGC policy but probably with a more pronounced apex. We also plan to study adaptive policies which performance is insensitive to system parameters.

Acknowledgements

The authors are grateful to the anonymous reviewers for their helpful reviews and suggestions. This work has been supported by the Spanish *Ministerio de Ciencia y Tecnología* under projects TIC2003-08272 and TIC2001-0956-C04-04, and by the *Generalitat of Valencia* under grant CTB/PRB/2002/267.

References

1. Ramachandran Ramjee, Ramesh Nagarajan, Don Towsley, "On Optimal Call Admission Control in Cellular Networks," ACM/Baltzer Wireless Networks Journal, vol. 3, no. 1, pp 29-41, 1997.
2. E. Altman, T. Jimenez, G. Koole, “On optimal call admission control in resource-sharing system,” IEEE Transactions on Communications, vol. 49, no. 9, pp.1659-1668, Sep. 2001.
3. V.B. Iversen, “The Exact Evaluation of Multi-Service Loss Systems with Access Control”, in Proceedings of the Seventh Nordic Teletraffic Seminar (NTS-7), Aug. 1987, Lund, Sweden.
4. Chin-Tau Lea and Anwar Alyatama, “Bandwidth Quantization and States Reduction in the Broadband ISDN,” IEEE/ACM Trans. on Networking, vol.3, no.3, pp.352-360, Jun. 1995.
5. G.L. Choudhury, K.K. Leung, and W. Whitt, “Efficiently Providing Multiple Grades of Service with Protection Against Overloads in Shared Resources”, AT&T Technical Journal, vol. 74, no. 4, pp. 50-63, 1995.
6. B. Li, C. Lin, and S. T. Chanson, “Analysis of a hybrid cutoff priority scheme for multiple classes of traffic in multimedia wireless networks,” ACM/Baltzer Wireless Networks Journal, vol. 4, no. 4, pp. 279-290, 1998.
7. H. Heredia-Ureta, F. A. Cruz-Pérez, and L. Ortigoza-Guerrero, “Multiple fractional channel reservation for optimum system capacity in multi-service cellular networks,” Electronic Letters, vol. 39, pp. 133-134, Jan. 2003.

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.
9. Vicent Pla and Vicente Casares-Giner, "Optimal Admission Control Policies in Multiservice Cellular Networks," in Proceedings of the International Network Optimization Conference (INOC2003), Paris, France, Oct. 2003.
10. S. M. Ross, Applied Probability Models with Optimization Applications. Holden-Day, 1970.
11. 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.
12. C.-T. Lea and A. Alyatama, “Bandwidth quantization and states reduction in the broadband ISDN,” IEEE/ACM Transactions on Networking, vol. 3, no. 3, pp. 352–360, June 1995.