European Journal of Control (2011)1:89–103

DOI:10.3166/EJC.17.89–103

European Journal of Control

A Reinforcement Learning Approach to Call Admission and Call Dropping Control in Links with Variable Capacity

Antonio Pietrabissa

Computer and System Science Department (DIS), University of Rome "La Sapienza", Rome, Italy

This paper defines a reinforcement learning (RL) approach to call control algorithms in links with variable capacity supporting multiple classes of service. The novelties of the document are the following: i) the problem is modeled as a constrained Markov decision process (MDP); ii) the constrained MDP is solved via a RL algorithm by using the Lagrangian approach and state aggregation. The proposed approach is capable of controlling class-level quality of service in terms of both blocking and dropping probabilities. Numerical simulations show the effectiveness of the approach.

Keywords: Call control, Reinforcement Learning (RL), Markov Decision Processes (MDP), communication networks

1. Introduction

We consider a multiclass network, where link capacity is time-varying, and where calls of different classes compete for bandwidth and are regulated by a Call Admission Control (CAC) algorithm, which decides whether a new call can be safely accepted by the network or not. We are also interested in controlling class-level constraints, expressed in terms of fairness with respect to the blocking probabilities observed by the different classes, and in terms of maximum tolerated dropping probabilities.

The CAC problem has been successfully modeled as a Markov decision process (MDP) ([1]) and solved via reinforcement learning (RL) algorithms ([24]). However, in the MDP and RL formulations introduced so far, the link capacity is considered as a given constant value. On the contrary, as we consider wireless networks this is not true, since the link capacity is time-varying, for instance due to weather conditions, interferences, speed of mobile users, adaptive coding, and modulation schemes; examples of such networks are CDMA networks (e.g., UMTS) ([17]), WiMAX networks ([7]), DVB-S2 satellite networks ([6]). In these networks, if the link capacity decreases the network controller might have to drop one or more calls (*forced dropping*). Call dropping is rather disruptive and should be avoided as far as possible.

To face this problem, in [19] the CAC MDP framework was extended to the case of links with time-varying capacity by integrating it with a Markov chain model of the link; the resulting MDP is solved by using a linear programming (LP) formulation, which allows the enforcement of class-level constraints. The MDP approach of [19] is capable of explicitly controlling both blocking and dropping probabilities by simultaneously computing both an optimal admission policy and an optimal dropping policy. The main drawback of the solution proposed in [19] is that it is not suitable for implementation in real network equipment, due to the scalability problem of the MDP algorithms (the so-called "curse of dimensionality," i.e., the state space explosion as the link capacity and the number of supported classes increase ([1], [19])).

E-mail: pietrabissa@dis.uniroma1.it

Received 12 May 2009; Accepted 4 May 2010
Recommended by J. Lunze, A.J. van der Schaft

In this paper, the MDP framework developed in [19] is used to define a constrained MDP; then, with the objective of addressing the scalability problem, the Lagrangian approach is used to develop a RL admission and dropping control algorithm. Note that, given that RL is a model-free approach, the proposed solution does not require *a priori* knowledge of traffic statistics since it relies on measured statistics.

In the literature, apart from the already cited [19], time-varying link capacity modeling has been faced separately from the MDP and RL-based admission control problems. The most common and effective way to model a link is Markov-chain modeling, where each link state is characterized by a different available capacity ([4], [20], [22], [29]). On the other hand, a plethora of MDP-based admission control algorithms can be found in the literature, both for wireless and wired networks (just to mention a few: [9], [10], [14], [18], [31], [33]); however, none of the cited references explicitly deals with links with time-varying capacity.

Due to the scarce scalability of the MDP approaches, and also due to the fact that they need a perfect knowledge of the model (i.e., in our case, of network and traffic statistical characteristics), RL-based approaches are also widely investigated in the literature: for example, [27] develops a RL CAC algorithm for multiservice networks with Quality of Service constraints, [11] and [30] for cellular networks, [28] for satellite networks, [12] for terrestrial networks, [13] for next-generation networks, etc. As in the MDP formulations introduced until now, in these papers the link capacity is considered as a given constant value.

The paper is organized as follows: Section 2 recalls the definition of constrained MDPs; Section 3 defines the constrained MDP addressing admission and dropping control in links with time-varying capacity; Section 4 describes the Lagrangian approach and the proposed RL algorithm; Section 5 presents the numerical simulations; finally, in Section 6 the conclusions are drawn.

2. Definition of Constrained MDPs

Under the Markovian[1] and stationarity assumptions, an unconstrained MDP is defined by a finite state space S, a finite and nonempty set of available control actions $\mathbf{u} \in A(\mathbf{s})$ associated to each state $\mathbf{s} \in S$, a cost $r(\mathbf{s}, \mathbf{u})$ which is incurred by the system when it is in state $\mathbf{s}$ and action $\mathbf{u}$ is chosen, and the probability $p(\mathbf{s}, \mathbf{s}', \mathbf{u})$ that, in the next stage, the system will be in state $\mathbf{s}'$ when action $\mathbf{u}$ in state $\mathbf{s}$ is chosen. The set of these transition probabilities constitute the transition matrix $\mathbf{T}$. A stationary policy is a function $\pi(\mathbf{s})$ which maps every state $\mathbf{s} \in S$ to a unique control action $\mathbf{u} \in A(\mathbf{s})$. When the system operates under a policy $\pi(\mathbf{s})$, the MDP reduces to a Markov chain, and the following expected average cost is defined:

$$R_\pi = \limsup_{n\to\infty} \frac{1}{n} E_\pi \left\{ \sum_{t=0}^{n-1} r\left[\mathbf{s}(t), \mathbf{u}(t)\right] \right\}, \tag{1}$$

where $\mathbf{s}(t)$ and $\mathbf{u}(t)$ are the state and the action at stage t, and the subscript π specifies that the controller operates under policy π. The MDP problem is to determine the optimal policy π^* minimizing cost (1). Beside standard Dynamic and Linear Programming approaches ([21]), RL algorithms can also be used to solve unconstrained MDPs ([24]); even if RL approaches achieve the optimal solution only under given conditions, such as infinite number of visits of each state, in practice they achieve approximate solutions to problems which are intractable for other methods, as, in this case, for Dynamic and Linear Programming approaches, which are subject to the scalability problems of MDPs.

As shown in the following Section, the admission and dropping control problem can be modeled as a constrained MDP, which is a MDP where another cost $d(\mathbf{s},\mathbf{u})$ is defined with its average cost:

$$D_\pi = \limsup_{n\to\infty} \frac{1}{n} E_\pi \left\{ \sum_{t=0}^{n-1} d\left[\mathbf{s}(t), \mathbf{u}(t)\right] \right\}, \tag{2}$$

and a maximum tolerated constant value K for cost (2) is specified. Then, in the constrained MDP, a policy π is feasible only if it is such that $D_\pi < K$. The constrained MDP, denoted with $\{S, A, \mathbf{T}, r, d, K\}$, is feasible if there exists at least a feasible policy, and the constrained MDP problem is to determine the optimal feasible policy π^* minimizing cost (1).

As mentioned in the introduction, the solution of constrained MDPs can be sought by two main approaches: the LP formulation and the Lagrangian approach. With this latter approach, RL algorithms can be used to solve constrained MDPs, since they formulate the constrained MDP as an unconstrained MDP i) by deleting the constraint on cost (2), and ii) by using a Lagrange multiplier $\omega \in [0, \infty)$ to define a new cost structure:

$$g\left[\mathbf{s}(t), \mathbf{u}(t), \omega\right] = r\left[\mathbf{s}(t), \mathbf{u}(t)\right] + \omega d\left[\mathbf{s}(t), \mathbf{u}(t)\right]. \tag{3}$$

The expected average cost is then defined as:

$$G_\pi(\omega) = \limsup_{n\to\infty} \frac{1}{n} E_\pi \left\{ \sum_{t=0}^{n-1} g\left[\mathbf{s}(t), \mathbf{u}(t), \omega\right] \right\}. \tag{4}$$

[1] A stochastic process has the Markov property if the conditional probability distribution of the next state of the process depends only upon the present state and is conditionally independent of past states.

Different values for the Lagrange multiplier ω determine generally different optimal policies π^*_ω for the unconstrained MDP $\{S, A, \mathbf{T}, g(\omega)\}$. The following theorem holds for stationary MDPs (analogous theorems are demonstrated for example in [23], [2], and [3] under harder assumptions):

Theorem 1: *If the constrained MDP* $\{S, A, \mathbf{T}, r, q, K\}$ *is unichain*[2] *and feasible, and if the costs r and d are nonnegative, then an optimal* ω^* *exists such that the optimal solution of the unconstrained MDP* $\{S, A, \mathbf{T}, g(\omega^q)\}$ *is an optimal (feasible) policy of the constrained MDP* $\{S, A, \mathbf{T}, r, q, K\}$. □

Note that the solution of a constrained MDP is randomized in at least one state ([3]). As discussed in Section 4, the RL algorithm should learn the optimal ω^* from available environment measures.

3. MDP Framework for Links with Variable Capacity

Sections 3.1 and 3.2 briefly describe the MDP framework for links with time-varying capacity developed in [19]; Section 3.3 defines the constrained MDP which takes into account fairness and dropping constraints.

3.1. Link Model

A generic time-varying link modeled as a Markov chain is considered. Each state l of the Markov chain is characterized by a different available capacity, denoted with ξ_l, $l = 1, \ldots, L$, with $\xi_l < \xi_{l+1}$, $l = 1, \ldots, L-1$. The number L of link states is defined based on the link characteristics. For instance, in DVB-S2 satellite networks L is the number of the (finite) available {modulation/coding/symbol rate} combinations, each one leading to a different transmission capacity ([6]); in UMTS networks the link capacity is interference-limited ([17]) and varies continuously (*soft capacity*): in this case, the number of states L is a model parameter driving the trade-off between scalability (which decreases with L) and granularity (which increases with L).

The generic transition frequency ϕ_{lk}, $l, k = 1, \ldots, L$, between two link states can be estimated by link measures, as described in example [4]; the resulting link transition matrix is denoted with $\mathbf{\Phi}$. Note that Markov chains describing network links are ergodic due to physical reasons.

[2] A MDP is unichain if, for each policy, there is a single recurrent class plus one (possibly empty) class of transient states ([20])

3.2. MDP Call Admission and Dropping Control

Let us consider a link supporting C classes of service, each one characterized by a bitrate e_c, $c = 1, \ldots, C$. Let $\mathbf{x}(t)$ be the vector of on-going calls, defined by the number of calls of each class c on-going at time t: $\mathbf{x}(t) = (n_1(t), n_2(t), \ldots, n_C(t))$. The network is represented by a discrete-time system, whose generic state $\mathbf{s}(t)$ is given by the vector $\mathbf{x}(t)$ plus the link capacity at time t, $\xi(t)$: $\mathbf{s}(t) = (\mathbf{x}(t), \xi(t))$. At time t, the load $\eta(t)$ associated to state $\mathbf{s}(t)$ is then $\eta(t) = \eta[\mathbf{s}(t)] = \eta[\mathbf{x}(t)] = \sum_{c=1,\ldots,C} n_c(t) e_c$.

Two cases arise:

1. $\eta(t) \leq \xi(t)$: in this case, if a call request arrives, the admission controller decides whether to admit or reject it based on the current state $\mathbf{s}(t)$. The admission decisions constitute the network admission policy.
2. $\eta(t) > \xi(t)$: in this case, one or more calls must be dropped until the system reaches a state whose load is less than the link capacity. Thus, beside the admission policy, a network dropping policy is required.

Note that, even if the admission policy is such that a given class c call is blocked whenever $\xi(t) + e_c > \eta(t)$, forced dropping cannot be avoided due to the fact that the link capacity is varying. Hereafter, a given state $\mathbf{s}(t)$ will be denoted as *available* at time t if $\eta[\mathbf{s}(t)] \leq \xi(t)$, *unavailable* otherwise.

The system dynamics is function of the state $\mathbf{s}(t)$, of the control action $\mathbf{u}(t)$ and of the disturbance $\mathbf{z}(t)$:

$$\mathbf{s}(t+1) = f(\mathbf{s}(t), \mathbf{u}(t), \mathbf{z}(t)), \tag{5}$$

The disturbance $\mathbf{z}(t)$ represents call attempts and terminations, characterized as follows: for each class c, call attempts are distributed according to a Poisson process with mean arrival frequency λ_c; the call holding time of class c is exponentially distributed with mean termination frequency μ_c[3]. The control $\mathbf{u}(t)$ is relevant at call attempts and when a state becomes unavailable due to the variation of the link capacity: in the former case $\mathbf{u}(t)$ is the admission decision, in the latter it is the dropping decision.

[3] Poisson call attempts and exponential call holding time are widely used in the literature and are adequate at least for voice users, but further research is needed in the area of Markov regenerative decision processes to justify it for the new traffic services ([10], [33]). However, note that traffic statistics are necessary to develop the MDP framework but are not required by the RL approach – see for example [11], whose RL-based admission control succeeded in controlling traffic characterized by a self-similar distribution.

3.2.1. State Space S

Let Ξ be the finite set of feasible on-going call vectors $\mathbf{x}$ whose load $\eta(\mathbf{x})$ is less than the maximum link capacity, and let M be the cardinality of Ξ:

$$\Xi = \left\{ \mathbf{x} = (n_1, \ldots, n_C) \,\middle|\, \sum_{c=1,\ldots,C} n_c e_c \leq \xi_L, \; n_c = 0, 1, 2, \ldots, c = 1, \ldots, C \right\}. \quad (6)$$

For each ξ_l, $l = 1, \ldots, L$, the generic state $\mathbf{s} = (\mathbf{x}, \xi_l)$ is available if $\eta(\mathbf{s}) = \eta(\mathbf{x}) \leq \xi_l$, unavailable otherwise. Then, the set of available states is:

$$S_{AV} = \{\mathbf{s} = (\mathbf{x}, \xi_l) \,|\, \mathbf{x} \in \Xi, \; \eta(\mathbf{s}) \leq \xi_l, \; l = 1, \ldots, L\}, \quad (7)$$

the set of unavailable states is

$$S_{UN} = \{\mathbf{s} = (\mathbf{x}, \xi_l) \,|\, \mathbf{x} \in \Xi, \; \xi_l < \eta(\mathbf{s}) \leq \xi_L, \; l = 1, \ldots, L\}, \quad (8)$$

and the whole state space is $S = S_{AV} \cup S_{UN}$. The total number of states is $L \cdot M$.

3.2.2. Action Space A

In the generic available state $\mathbf{s} = (n_1, \ldots, n_c, \ldots, n_C, \xi_l) \in S_{AV}$, if a call attempt of class c occurs, the controller might decide to block the call (and the state remains the same) or to accept it, provided that $\mathbf{s}' = (n_1, \ldots, n_c + 1, \ldots, n_C, \xi_l) \in S_{AV}$. Such decision is denoted as $u(\mathbf{s}, c)$, and is equal to 1 if the decision is to accept the new call, to 0 if it is to reject it; no decision can be taken on call terminations. By defining $\boldsymbol{\delta}_c$ as a $(C + 1)$ vector of zeros but the element of column c equal to 1, the above defined state $\mathbf{s}'$ is equal to $\mathbf{s} + \boldsymbol{\delta}_c$ and the action space of an available state is defined as follows:

$$A(\mathbf{s}) = \left\{ \mathbf{u}(\mathbf{s}) = \big(u(\mathbf{s}, 1), \ldots, u(\mathbf{s}, C)\big) \,\middle|\, u(\mathbf{s}, c) \in \{0, 1\} \right. \text{ if } \mathbf{s} + \boldsymbol{\delta}_c \in S_{AV}, \; u(\mathbf{s}, c) = 0 \left. \text{ if } \mathbf{s} + \boldsymbol{\delta}_c \notin S_{AV}, \; c = 1, \ldots, C \right\}, \; \forall \mathbf{s} \in S_{AV} \quad (9)$$

In an unavailable state, if a call attempt of class c occurs, the controller has to block the call; moreover, the controller must drop one call to try to reach an available state. The frequency f_{drop} of the dropping action (which determines the time interval between two consecutive dropping actions) depends on implementation constraints but is practically much higher than traffic and link state dynamics. Considering a generic unavailable state $\mathbf{s} = (n_1, \ldots, n_C, \xi_l) \in S_{UN}$, the controller decision is to select the class c of the call to be dropped among all classes c such that $n_c > 0$. The system is then driven to another unavailable if $\mathbf{s} - \boldsymbol{\delta}_c \in S_{UN}$, or to an available state if $\mathbf{s} - \boldsymbol{\delta}_c \in S_{AV}$. Such decision is denoted as $u(\mathbf{s}, c)$, and is equal to 1 if the decision is to drop a class c call, 0 otherwise; furthermore, when the system is in an unavailable state, exactly one dropping decision must be equal to 1. The action space of an unavailable state is then defined as follows:

$$A(\mathbf{s}) = \left\{ \mathbf{u}(\mathbf{s}) = \big(u(\mathbf{s}, 1), \ldots, u(\mathbf{s}, C)\big) \,\middle|\, u(\mathbf{s}, c) \in \{0, 1\} \right. \text{ if } n_c > 0, \; u(\mathbf{s}, c) = 0 \text{ if } n_c = 0, \sum_{c=1}^{C} u(s, c) = 1, \left. c = 1, \ldots, C \right\}, \; \forall \mathbf{s} \in S_{UN}. \quad (10)$$

The controller policy π is defined by setting an action $\mathbf{u}(\mathbf{s}) \in A(\mathbf{s})$ for each state $\mathbf{s} \in S$. The admission policy is defined by the actions $\mathbf{u}(\mathbf{s}) \in A(\mathbf{s})$, $\forall \mathbf{s} \in S_{AV}$. The dropping policy is defined by the actions $\mathbf{u}(\mathbf{s}) \in A(\mathbf{s})$, $\forall \mathbf{s} \in S_{UN}$. The policy space Π is the set of the feasible policies:

$$\Pi = \left\{ \pi : S \to A \,\middle|\, \pi(\mathbf{s}) = \mathbf{u}(\mathbf{s}) \in A(\mathbf{s}), \; \mathbf{s} \in S \right\}. \quad (11)$$

3.2.3. Transition Matrix T

The transition probabilities which define $\mathbf{T}$ are inferred from the link model, from the above-stated assumptions on $\mathbf{z}(t)$, and from the above-defined action space A.

In particular, the following transition frequencies are defined:

(a) Transitions between available states when a class c call terminates:

$$t(\mathbf{s}, \mathbf{s} - \boldsymbol{\delta}_c) = \begin{cases} n_c(\mathbf{s})\mu_c & \text{if } \mathbf{s} - \boldsymbol{\delta}_c \in S_{AV} \\ 0 & \text{otherwise} \end{cases}, \quad \mathbf{s} \in S_{AV}, \; c = 1, \ldots, C, \quad (12)$$

where, with slight abuse of notation, $n_c(\mathbf{s})$ is the number of on-going calls of class c when the system is in state $\mathbf{s}$;

(b) Transitions between available states when a class c call is accepted:

$$t(\mathbf{s}, \mathbf{s} + \boldsymbol{\delta}_c) = \begin{cases} u(\mathbf{s}, c)\lambda_c & \text{if } \mathbf{s} + \boldsymbol{\delta}_c \in S_{AV} \\ 0 & \text{otherwise} \end{cases}, \quad \mathbf{s} \in S_{AV}, \; c = 1, \ldots, C \quad (13)$$

(note that these transition frequencies depend on the admission policy);

(c) Transitions from unavailable states when a class c call is dropped or terminates[4]:

$$t(\mathbf{s}, \mathbf{s} - \boldsymbol{\delta}_c) = \begin{cases} f_{drop} u(\mathbf{s}, c) + n_c(\mathbf{s})\mu_c & \text{if } \mathbf{s} - \boldsymbol{\delta}_c \in S \\ 0 & \text{otherwise} \end{cases},$$

$$\mathbf{s} \in S_{UN},\ c = 1, \ldots, C; \tag{14}$$

(note that these transition frequencies depends on the dropping policy);

(d) Transitions due to link state changes:

$$t(\mathbf{s}, \mathbf{s}') = \phi_{lk}, \mathbf{s} = (\mathbf{x}, \xi_l) \in S,\ \mathbf{s}' = (\mathbf{x}, \xi_k) \in S,$$

$$l, k = 1, \ldots, L,\ l \neq k; \tag{15}$$

(e) All the other transition frequencies are null.

To define the transition probabilities, a standard *uniformization* procedure ([19]) is considered:

(1) Divide the transition frequencies by the constant γ, which has to be greater than the maximum total output frequency among all the states:

$$p(\mathbf{s}, \mathbf{s}') = t(\mathbf{s}, \mathbf{s}')/\gamma, \text{ with } \gamma > \max_{\mathbf{s} \in S} \left\{ \sum_{\mathbf{s}' \in S} t(\mathbf{s}, \mathbf{s}') \right\},$$

$$\mathbf{s} \neq \mathbf{s}',\ \mathbf{s}, \mathbf{s}' \in S; \tag{16}$$

(2) Add self-transitions to let the sum of the transitions leaving a state equal to 1:

$$p(\mathbf{s}, \mathbf{s}) = 1 - \sum_{\substack{\mathbf{s}' \in S \\ \mathbf{s}' \neq \mathbf{s}}} p(\mathbf{s}, \mathbf{s}'),\ \mathbf{s} \in S. \tag{17}$$

3.2.4. Cost Function r

The objective of minimizing the total blocking probability is pursued by associating the following cost to the couples available state-action:

$$r[\mathbf{s}, \mathbf{u}(\mathbf{s})] = \sum_{c=1}^{C} [1 - u(\mathbf{s}, c)] = \sum_{c=1}^{C} P_b(\mathbf{s}, c),$$

$$\forall \mathbf{u}(\mathbf{s}) \in A_{AV},\ \mathbf{s} \in S_{AV}, \tag{18}$$

where $P_b(\mathbf{s}, c) = 1 - u(\mathbf{s}, c)$ is the probability of blocking a class c call when the system is in state $\mathbf{s}$.

[4] In this case, $\mathbf{s} - \delta_c$ can be either available or unavailable.

In the unavailable states, each incoming call is blocked; the associated cost is thus:

$$r[\mathbf{s}, \mathbf{u}(\mathbf{s})] = \sum_{c=1}^{C} P_b(\mathbf{s}, c) = C,\ \forall \mathbf{u}(\mathbf{s}) \in A(\mathbf{s}),\ \forall \mathbf{s} \in S_{UN}. \tag{19}$$

Costs (18) and (19) entail that the total cost minimized by the MDP under policy π is the sum of the per class stationary blocking probabilities, denoted with $P_{b,\pi}(c)$, $c = 1, \ldots, C$:

$$R_\pi = \sum_{\mathbf{s} \in S} r[\mathbf{s}, \mathbf{u}(\mathbf{s})] P_\pi(\mathbf{s}) = \sum_{c=1,\ldots,C} P_{b,\pi} c), \tag{20}$$

where $P_\pi(\mathbf{s})$ is the stationary probability that the system is in state $\mathbf{s}$ under policy π.

Lemma 1: *The MDP* $\{S, A, \mathbf{T}, r\}$ *is unichain for any policy* π. □

Proof: The MDP is unichain, since blocking decisions set the associated transition probability to zero (see equation (13)), whereas the transitions associated to call termination, call dropping, and link state changes are always positive (see equations (12), (14) and (15)). In fact, a given policy might determine one or more sets of states such that all the transitions toward these sets are associated to blocked call arrivals and are null. However, there are always nonnull transitions departing from these sets which are associated to call terminations, call dropping, and link state changes. Thus, these sets identify the set of transient states, and the remaining states constitute the single recurrent class. The recurrent class always exists: at least, a set of nonnull transitions (due to call terminations, call dropping and link state changes) always exists from every state to the empty states $\mathbf{s} = (\mathbf{x}, \xi_l)$, $l = 1, \ldots, L$, such that $\mathbf{x} = (0, \ldots, 0)$. □

3.3. Constrained MDP Modeling Fairness and Dropping Constraints

This Section develops the constrained MDP which addresses the objectives of controlling class-level constraints in terms of fairness and dropping probabilities.

3.3.1. State Augmentation to Address Fairness

The state-dependent stationary blocking probability of class c in state $\mathbf{s}$, denoted with $P_b(\mathbf{s}, c)$, is defined as the probability of blocking a class c call when the system is in state $\mathbf{s}$. A fair call control policy is such that the blocking probabilities of the different classes are not "excessively" different. To achieve fairness, the policy should aim at

minimizing the worst (i.e., largest) blocking probability among the supported classes (as in [19]):

$$\min_{\pi\in\Pi}\sum_{\mathbf{s}\in S}\left[1-u(\mathbf{s},c)\right]P_\pi(\mathbf{s})=\min_{\pi\in\Pi}\sum_{\mathbf{s}\in S}P_b(\mathbf{s},c)P_\pi(\mathbf{s})$$
$$=\min_{\pi\in\Pi}P_{b,\pi}(c),\ c=\arg\max_{c'=1,\ldots,C}\left\{P_{b,\pi}(c')\right\}. \quad (21)$$

To directly take into account this enforcement in the cost function, we should add a new term $P_{b,\pi}(\mathbf{s},c)$ to equation (18) in a multiobjective optimization fashion:

$$r\left[\mathbf{s},\mathbf{u}(\mathbf{s})\right]=\sum_{c'=1}^{C}\left[1-u(\mathbf{s},c')\right]+w\left[1-u(\mathbf{s},c)\right]$$
$$=\sum_{c'=1}^{C}P_b(\mathbf{s},c')+wP_b(\mathbf{s},c),\ \mathbf{u}(\mathbf{s})\in A(\mathbf{s}),$$
$$\mathbf{s}\in S_{AV},\ c=\arg\max_{c'=1,\ldots,C}\left\{P_{b,\pi}(c')\right\} \quad (22)$$

where w is a weight which has to be tuned to specify the relevance of the second term, aimed at obtaining fair blocking probabilities, with respect to the first term, aimed at minimizing the total blocking probability.

The class c which has the worst stationary blocking probability $P_{b,\pi}(c)$ depends on the policy π. Hereafter, the worst class will be denoted with $c_b=\arg\max_{c=1,\ldots,C}\left\{P_{b,\pi}(c)\right\}$. To restore a state-dependent cost, we define the following augmentation of the available state space:

$$\bar{S}_{AV}=\left\{\bar{\mathbf{s}}=(\mathbf{s},c)\middle|\mathbf{s}\in S_{AV},\ c=1,\ldots,C\right\}. \quad (23)$$

The meaning is that the system visits the available state $\bar{\mathbf{s}}=(\mathbf{s},c)=(\mathbf{x},\xi_l,c)\in\bar{S}_{AV}$ when the number of on-going calls is given by $\mathbf{x}$, the link is in state l and $c=c_b$. The action space $\bar{A}(\bar{\mathbf{s}})$ of an augmented available state is the same as the action space of the corresponding "original" state, defined by (9): $\bar{A}(\bar{\mathbf{s}})=A(\mathbf{s})$, with $\mathbf{s}$ such that $\bar{\mathbf{s}}=(\mathbf{s},c),\ \forall\bar{\mathbf{s}}\in\bar{S}_{AV}$. In the augmented state space, the cost function is state-dependent:

$$\bar{r}\left[\bar{\mathbf{s}},\mathbf{u}(\bar{\mathbf{s}})\right]=\sum_{c'=1}^{C}\left[1-u(\bar{\mathbf{s}},c')\right]+w\left[1-u(\bar{\mathbf{s}},c)\right],$$
$$\mathbf{u}(\bar{\mathbf{s}})\in\bar{A}(\bar{\mathbf{s}}),\ \forall\bar{\mathbf{s}}=(\mathbf{s},c)\in\bar{S}_{AV}. \quad (24)$$

Remark 1: *It is important to underline that no hard constraints on dropping probabilities should be enforced. In fact, if a maximum tolerated blocking probability were enforced the constrained MDP would not admit feasible solutions as the traffic load grows (e.g., see [16], [32]).* □

3.3.2. State Augmentation to Address Dropping Constraints

The stationary dropping probability $P_{d,\pi}(c)$ of class c calls under policy π is defined as the probability that an already accepted call of class c is dropped. We define the call c ingress probability as the birth probability of class c, λ_c/γ, times the acceptance probability of class c calls, computed as $[1-P_{b,\pi}(c)]$. The state-dependent dropping cost $C_d(\mathbf{s},c)$ of class c in state $\mathbf{s}$ is defined as the probability that a class c call is dropped when the system is in state $\mathbf{s}$:

$$C_d(\mathbf{s},c)=u(\mathbf{s},c)\frac{f_{drop}}{\gamma},\ \mathbf{s}\in S_{UN},\ c=1,\ldots,C. \quad (25)$$

By summing the dropping costs (25) over the unavailable states, weighted by the stationary probabilities and divided by the ingress probability of class c, the class c dropping probability $P_{d,\pi}(c)$ is obtained:

$$P_{d,\pi}(c)=\frac{\sum_{\mathbf{s}\in S_{UN}}C_d(\mathbf{s},c)P_\pi(\mathbf{s})}{\frac{\lambda_c}{\gamma}\left[1-P_{b,\pi}(c)\right]}$$
$$=\frac{\sum_{\mathbf{s}\in S_{UN}}u(\mathbf{s},c)\frac{f_{drop}}{\gamma}P_\pi(\mathbf{s})}{\frac{\lambda_c}{\gamma}\left[1-P_{b,\pi}(c)\right]},\ c=1,\ldots,C. \quad (26)$$

In fact, the numerator of equation (26) represents the probability that a class c call is dropped, whereas the denominator represents the probability that a class c enter the system. As explained hereafter, probability (26) is well-posed by construction. Note that, in a generic realization of the MDP, the system can visit an unavailable state with $n_c\geq 1$ on-going calls of class c, $c=1,\ldots,C$, only if at least n_c class c calls were previously accepted. Thus, in all the realizations of the MDP, the (nonnegative) total number of dropped class c calls cannot exceed the (nonnegative) total number of accepted class c calls. Consequently, considering the whole MDP, the total probability that a class c call is dropped (represented by the numerator of equation (26)) cannot be greater than the acceptance probability of class c calls (represented by the denominator of equation (26)); thus, it follows that $P_{d,\pi}(c)\in[0,1]$.

From equation (26), the following C constraints are defined:

$$P_{d,\pi}(c)=\frac{\sum_{\mathbf{s}\in S_{UN}}C_d(\mathbf{s},c)P_\pi(\mathbf{s})}{\frac{\lambda_c}{\gamma}\left[1-P_{b,\pi}(c)\right]}<P_{d,MAX}(c),$$
$$c=1,\ldots,C, \quad (27)$$

expressing the objective of limiting the dropping probabilities under the maximum tolerated dropping probabilities $P_{d,MAX}(c)$, $c = 1, \ldots, C$.

To make use of the Lagrangian framework developed in [26] and [27], a single constraint has to be considered. Thus, we consider the following constraint, which is equivalent to constraints (27):

$$\sum_{\mathbf{s}\in S_{UN}} C_d(\mathbf{s}, c) P_\pi(\mathbf{s}) < P_{d,MAX}(c) \frac{\lambda_c}{\gamma} \left[1 - P_{b,\pi}(c)\right],$$

$$c = \arg\max_{c'=1,\ldots,C} \left\{ \frac{P_{d,\pi}(c')}{P_{d,MAX}(c')} \right\}. \tag{28}$$

The corresponding cost d (i.e., the cost which takes into account the constraints in constrained MDPs, see Section 2) is then defined as follows:

$$d\left[\mathbf{s}, \mathbf{u}(\mathbf{s})\right] = C_d(\mathbf{s}, c),\ \mathbf{s} \in S_{UN},\ \mathbf{u}(\mathbf{s}) \in A(\mathbf{s}),$$

$$c = \arg\max_{c'=1,\ldots,C} \left\{ \frac{P_{d,\pi}(c')}{P_{d,MAX}(c')} \right\}. \tag{29}$$

The class c which has the worst normalized stationary dropping probability depends on the policy π. Hereafter, the worst class c will be denoted with $c_d = \arg\max_{c=1,\ldots,C} \left\{P_{d,\pi}(c)/P_{d,MAX}(c)\right\}$. Similarly to Section 3.3.1, to restore a state-dependent cost we define the following augmentation of the unavailable state space:

$$\bar{S}_{UN} = \left\{\bar{\mathbf{s}} = (\mathbf{s}, c) \middle| \mathbf{s} \in S_{UN},\ c = 1, \ldots, C\right\}. \tag{30}$$

The meaning is that the system visits the unavailable state $\bar{\mathbf{s}} = (\mathbf{s}, c) = (\mathbf{x}, \xi_l, c) \in \bar{S}_{UN}$ when the number of ongoing calls is given by $\mathbf{x}$, the link is in state l and $c = c_d$. The action space $\bar{A}(\bar{\mathbf{s}})$ of an augmented unavailable state is the same as the action space of the corresponding "original" state, defined by (10): $\bar{A}(\bar{\mathbf{s}}) = A(\mathbf{s})$, with $\mathbf{s}$ such that $\bar{\mathbf{s}} = (\mathbf{s}, c)$, $\forall \bar{\mathbf{s}} \in \bar{S}_{UN}$. Equations (23) and (30) define the augmented state-space $\bar{S} = \bar{S}_{AV} \bigcup \bar{S}_{UN}$. Accordingly, $\bar{\pi}$ denotes a policy in the augmented policy space:

$$\bar{\Pi} = \left\{\bar{\pi} : \bar{S} \to \bar{A} \middle| \bar{\pi}(\bar{\mathbf{s}}) = \mathbf{u}(\bar{\mathbf{s}}) \in \bar{A}(\bar{\mathbf{s}}), \bar{\mathbf{s}} \in \bar{S}\right\}. \tag{31}$$

In the augmented state space, cost (29) becomes state-dependent:

$$d\left[\bar{\mathbf{s}}, \mathbf{u}\right] = C_d(\bar{\mathbf{s}}, c),\ \bar{\mathbf{s}} = (\mathbf{s}, c) \in \bar{S}_{UN},$$

$$\mathbf{u}(\bar{\mathbf{s}}) \in \bar{A}(\bar{\mathbf{s}}),\ c = 1, \ldots, C, \tag{32}$$

where $C_d(\bar{\mathbf{s}}, c)$ is defined analogously to equation (25).

3.3.3. *Constrained MDP Addressing Fairness and Dropping Probability Constraint*

This Section completes the definition of the constrained MDP.

The following transition frequencies outgoing from the augmented states are defined:

(a) By fixing the value of c, the transition frequencies between available states due to call terminations, call acceptances and link state changes are defined as in equations (12), (13), and (15).

(b) Similarly, by fixing the value of c, the transition frequencies between unavailable states due to call terminations, call droppings, and link state changes are defined as in equations (14) and (15).

(c) For each couple c and c', the transition frequencies between available states and unavailable states due to link state changes are the following:

$$t(\bar{\mathbf{s}}, \bar{\mathbf{s}}') = \begin{cases} \phi_{lk} & \text{if } c = c_b \text{ and } c' = c_d \\ 0 & \text{otherwise} \end{cases},$$

$$\bar{\mathbf{s}} = (\mathbf{x}, \xi_l, c) \in \bar{S}_{AV},\ \bar{\mathbf{s}}' = \left(\mathbf{x}, \xi_k, c'\right) \in \bar{S}_{UN};$$

$$c, c' = 1, \ldots, C;\ l, k = 1, \ldots, L;\ l \neq k \tag{33}$$

(note that only one of transitions (33) is enabled at a time, depending on the worst classes c_b and c_d).

(d) Similarly, for each couple c and c', $c, c' = 1, \ldots, C$, the transition frequencies between unavailable states and available states due to link state changes are the following:

$$t(\bar{\mathbf{s}}', \bar{\mathbf{s}}) = \begin{cases} \phi_{lk} & \text{if } c = c_b \text{ and } c' = c_d \\ 0 & \text{otherwise} \end{cases},$$

$$\bar{\mathbf{s}}' = \left(\mathbf{x}, \xi_k, c'\right) \in \bar{S}_{UN},\ \bar{\mathbf{s}} = (\mathbf{x}, \xi_l, c) \in \bar{S}_{AV};$$

$$c, c' = 1, \ldots, C;\ l, k = 1, \ldots, L;\ l \neq k \tag{34}$$

(note that only one of transitions (34) is enabled at a time, depending on the worst classes c_b and c_d).

(e) The transition frequencies from unavailable states to available states due to a class c call dropping or termination are the following:

$$t(\bar{\mathbf{s}}', \bar{\mathbf{s}}'') = \begin{cases} f_{drop} u(\bar{\mathbf{s}}', c) + n_c(\bar{\mathbf{s}}')\mu_c, \\ \quad \text{if } c' = c_d,\ c'' = c_b \text{ and} \\ \quad\quad \bar{\mathbf{s}}'' = (\mathbf{s} - \boldsymbol{\delta}_c, c'') \in \bar{S}_{AV} \\ 0, \quad \textit{otherwise} \end{cases}, \tag{35}$$

$$\bar{\mathbf{s}}' = (\mathbf{s}, c') \in \bar{S}_{UN},\ c = 1, \ldots, C.$$

Remark 2: *From definitions a)–e), it follows that, under policy $\bar{\pi}$, the alive transitions are: 1) transitions between couples of available states $(\mathbf{s}, c_b)$ and $(\mathbf{s}', c_b)$; 2) transitions between couples of unavailable states $(\mathbf{s}, c_d)$ and $(\mathbf{s}', c_d)$; 3) transitions between copule of available states $(\mathbf{s}, c_b)$ and unavailable states $(\mathbf{s}', c_b)$.* □

By following the uniformization procedure outlined in Section 3.2.3 (equations (16) and (17)), the augmented transition matrix $\bar{\mathbf{T}}$ is obtained. To cope with the possibility that, in a given realization, the system under policy $\bar{\boldsymbol{\pi}}$ starts in an available (unavailable) state $(\mathbf{s}, c)$, with $c \neq c_b$ ($c \neq c_d$), where, according to Remark 4, no transition is alive but the self-transition, the following transition probabilities are added to the transition matrix (and the self-transitions are modified accordingly as in equation (17)):

$$t(\bar{\mathbf{s}}', \bar{\mathbf{s}}'') = \begin{cases} 1 & \text{if } c' \neq c_b \text{ and } c'' = c_b \\ 0 & \text{otherwise} \end{cases},$$

$$\bar{\mathbf{s}}' = (\mathbf{s}, c') \in \bar{S}_{AV},\ \bar{\mathbf{s}}'' = (\mathbf{s}, c'') \in \bar{S}_{AV},\ c' \neq c'', \tag{36}$$

$$t(\bar{\mathbf{s}}', \bar{\mathbf{s}}'') = \begin{cases} 1 & \text{if } c' \neq c_d \text{ and } c'' = c_d \\ 0 & \text{otherwise} \end{cases},$$

$$\bar{\mathbf{s}}' = (\mathbf{s}, c') \in \bar{S}_{UN},\ \bar{\mathbf{s}}'' = (\mathbf{s}, c'') \in \bar{S}_{UN},\ c' \neq c''. \tag{37}$$

Cost (24) is associated to each available augmented state, whereas the cost associated to each unavailable augmented state is still defined as in equation (19):

$$\bar{r}\left[\bar{\mathbf{s}}, \mathbf{u}(\bar{\mathbf{s}})\right] = C,\ \forall \mathbf{u}(\bar{\mathbf{s}}) \in \bar{A}(\bar{\mathbf{s}}),\ \forall \bar{\mathbf{s}} \in \bar{S}_{UN}, \tag{38}$$

To take into account dropping constraints, the cost d defined in equation (32) is associated to each unavailable state, whereas the cost d associated to the available states is null, since no dropping occurs.

The constrained MDP is then $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, \bar{r}, d, K\}$.

The following theorem holds:

Theorem 2: *The constrained MDP $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, \bar{r}, d, K\}$ has following properties:*

(i) the constrained MDP is unichain under any policy $\bar{\pi}$;
(ii) the constrained MDP is feasible for any $P_{d,MAX}(c) \in [0, 1]$, $c = 1, \ldots, C$;
(iii) the expected average cost evaluates the total blocking probability and the fairness with respect to the blocking probabilities;
(iv) by setting $0 < K \leq P_{d,MAX}(c_d)\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]$, the dropping probabilities $P_{d,\bar{\pi}}(c)$ are constrained by the maximum dropping probabilities $P_{d,MAX}(c)$, $c = 1, \ldots, C$. □

Proof: See Annex 1. □

There is still a problem: the maximum tolerated value of the constant K is not known in advance, since it depends on the policy. This problem will be addressed in the RL framework by estimating this value on-line (see Section 4).

4. Lagrangian Approach and RL Algorithm

As described in Section 4.2, RL algorithms can be used to approximate the solution of constrained MDPs under the Lagrangian approach introduced in Section 4.1.

4.1. Lagrangian Approach

As in equation (3), under the Lagrangian approach the following cost function is obtained:

$$g\left[\bar{\mathbf{s}}, \mathbf{u}(\bar{\mathbf{s}}), \omega\right] = \bar{r}\left[\bar{\mathbf{s}}, \mathbf{u}(\bar{\mathbf{s}})\right] + \omega d\left[\bar{\mathbf{s}}, \mathbf{u}(\bar{\mathbf{s}})\right] =$$

$$\begin{cases} \sum_{c=1}^{C} u(\bar{\mathbf{s}}, c) + wP_{b,\bar{\pi}}(\bar{\mathbf{s}}, c),\ \mathbf{u}(\bar{\mathbf{s}}) \in \bar{A}(\bar{\mathbf{s}}),\ \bar{\mathbf{s}} = (\mathbf{s}, c) \in \bar{S}_{AV} \\ C + \omega C_{d,\bar{\pi}}(\bar{\mathbf{s}}, c),\ \mathbf{u}(\bar{\mathbf{s}}) \in A(\bar{\mathbf{s}}),\ \bar{\mathbf{s}} = (\mathbf{s}, c) \in \bar{S}_{UN} \end{cases}. \tag{39}$$

Equation (39) defines the unconstrained MDP $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, g(\omega)\}$. The following theorem holds:

Theorem 3: *An optimal ω^* exists such that an optimal solution of the unconstrained MDP $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, g(\omega^*)\}$ is an optimal (randomized) policy of the constrained MDP $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, \bar{r}, d, K\}$.* □

Proof: The theorem follows from the fact that Theorem 1 holds, since i) Theorem 2 states that the constrained MDP $\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, \bar{r}, d, K\}$ is unichain; ii) Theorem 2 shows that the constrained MDP is feasible; iii) the costs $\bar{r}$ and d (equations (24), (32) and (38)) are nonnegative. □

With the RL approach, ω^* must be approximated by on-line environment measures, as specified in the following Section. As shown in [2], in unichain constrained MDPs the optimal policy is a randomized policy at least in 1 state. However, for the sake of implementation complexity, we will only seek deterministic suboptimal policies: simulation results will evaluate how the deterministic policy manages to control blocking and dropping probabilities.

4.2. RL Algorithm

Once the problem is defined as an unconstrained MDP, standard RL algorithms can be applied. The proposed

algorithm is based on the RL CAC developed in [26] for the constant capacity case. For the sake of simplicity, and since the specific structure of the RL algorithm is not a focus of this article, the popular Watkins' Q-Learning algorithm is considered ([24]), which is based on the estimation of the action-value function Q (that gives the expected cost of selecting a given action in a given state and following a fixed policy thereafter). At stage t, the estimation step of Q-learning is the following:

$$Q(\bar{\mathbf{s}}, \bar{\mathbf{u}}; t+1) = (1-\alpha) Q(\bar{\mathbf{s}}, \bar{\mathbf{u}}; t) + \alpha \left[g(\bar{\mathbf{s}}, \bar{\mathbf{u}}, \omega) + \rho(t) \max_{\bar{\mathbf{u}}' \in \bar{A}(\bar{\mathbf{s}}')} Q(\bar{\mathbf{s}}', \bar{\mathbf{u}}'; t) \right], \tag{40}$$

where $\rho(t) \in (0,1]$ is the learning rate and $\alpha \in (0,1)$ is the discount factor. If $\rho(t)$ is such that $\lim_{n\to\infty} \sum_{t=0}^{n-1} \left|\rho(t)\right|^2 < \infty$ and $\lim_{n\to\infty} \sum_{t=0}^{n-1} \rho(t) = \infty$, Q converges to the optimal Q^* for $t \to \infty$ ([24]).

Then, the deterministic optimal policy is retrieved as:

$$\mathbf{u}^*(\bar{\mathbf{s}}) = \arg\max_{\mathbf{u} \in \bar{A}(\bar{\mathbf{s}})} Q^*(\bar{\mathbf{s}}, \mathbf{u}),\ \bar{\mathbf{s}} \in \bar{S}. \tag{41}$$

Equation (41) shows that the estimation of the action-value function has to be performed in each state. Thus, an effective approximation of the optimal action-value function can be achieved only if each state is visited a sufficient number of times. To favor the state space exploration, at each stage t the standard ε-greedy approach ([24]) is used to choose the controller action:

$$\bar{\pi}(\bar{\mathbf{s}}; t) = \begin{cases} \mathbf{u}^*(\bar{\mathbf{s}}), & \text{if } rnd(t) < (1-\varepsilon) \\ \mathbf{u}(\bar{\mathbf{s}}) \neq \mathbf{u}^*(\bar{\mathbf{s}}), & \text{otherwise} \end{cases},\ \bar{\mathbf{s}} \in \bar{S}, \tag{42}$$

where $rnd(t) \in [0,1]$ is a uniform random number and $\varepsilon \in [0,1]$ is a predefined real number which sets the trade-off between the need of exploration (required to obtain a rich environment model) and the convergence speed.

A key feature of RL algorithms is that they do not rely on statistics but on measurements. In particular, the proposed call control algorithm requires updating the following measures at every stage and for any class $c = 1, \ldots, C$: number of call births, number of blocked calls, number of dropped calls, which allow the computation of measured blocking probabilities and measured dropping probability. Note that if $c' = c_b$ ($c' = c_d$) and the system is in a given available (unavailable) state $\bar{\mathbf{s}} = (\mathbf{s}, c')$, as the worst measured blocking probability (the worst normalized measured dropping probability) changes to c'', the system immediately transitions to the state $\bar{\mathbf{s}} = (\mathbf{s}, c'')$, according to equation (36) (equation (37)).

To obtain a practical algorithm suitable for implementation, the scalability of the Q function and the on-line estimation of the Lagrangian multiplier and of the value K of the maximum tolerated cost will be addressed in Sections 4.2.1 and 4.2.2.

4.2.1. State Aggregation

The state aggregation is based on the one introduced for the MDP approach in [18] (where the aggregation model is discussed in detail). As analyzed in [18] for the MDP approach, state aggregation leads to suboptimal policies, and the controller performance worsens with the aggregation level. Differently, with the RL approach the state space exploration even benefits from some level of state aggregation, since the RL algorithm learns to control the blocking/dropping probabilities of a given state only after having visited that state a sufficient number of times: without aggregation, the number of states rapidly grows due to the poor scalability of MDP models, leading to poor exploration capability and, consequently, to poor algorithm performance. The granularity H of state aggregation, defined hereafter, is another parameter which has to be tuned (see the simulations in Section 5).

The states are aggregated based on their load. By defining H load thresholds $\eta_{th}(i)$, $i = 1, \ldots, H$, such that i) $\eta_{th}(i+1) > \eta_{th}(i)$, $i = 1, \ldots, H-1$, ii) $\eta_{th}(1) = 0$, iii) $\eta_{th}(H) = \eta_L, H-1$ aggregated available states are obtained, for each value of c:

$$\hat{\mathbf{s}}_{i,c} = \left\{ \bar{\mathbf{s}} \in \bar{S}_{AV} \middle| \bar{\mathbf{s}} = (\mathbf{s}, c) \text{ and } \eta_{th}(i) < \eta(\bar{\mathbf{s}}) \leq \eta_{th}(i+1) \right\}, \quad i = 1, \ldots, H-1,\ c = 1, \ldots, C, \tag{43}$$

and $H-1$ aggregated unavailable states are obtained:

$$\hat{\mathbf{s}}_{i,c} = \left\{ \bar{\mathbf{s}} \in \bar{S}_{UN} \middle| \bar{\mathbf{s}} = (\mathbf{s}, c) \text{ and } \eta_{th}(i) < \eta(\bar{\mathbf{s}}) \leq \eta_{th}(i+1) \right\}, \quad i = 1, \ldots, H-1,\ c = 1, \ldots, C. \tag{44}$$

The number of aggregate states is therefore $2(H-1)L$.

The action space and the cost functions after the state aggregation are easily derived from the action space and cost functions defined in Section 3. State-dependent blocking and dropping probabilities are computed exactly as with the nonaggregated states. Practical rules are implemented in case states with different available actions are

in the same state aggregate: if the estimated best action associated to the aggregate is not available for the system (i.e., the best action is to accept a call, but with this decision the resulting system load would become greater than the link load; the best action is to drop a class c call, but no class c call is on-going), then the controller selects the second best action (or the c-th best action, if the first $(c-1)$ ones are not available).

4.2.2. *Estimation of the Lagrange Multiplier and of the Value of the Maximum Tolerated Cost*

The following simple estimate of the Lagrange multiplier ω, analogous to the one used in [5], was implemented in the simulations:

$$\omega(t+1) = \begin{cases} 0, & t \le t_{expl} \\ \max\left\{0, \omega(t) + \dfrac{k_\omega}{t - t_{expl}} \left[\max\limits_{c=1,\ldots,C} \dfrac{P_{d,MEAS}(c,t)}{P_{d,MAX}(c)} - 1\right]\right\}, & t > t_{expl} \end{cases} \tag{45}$$

where $P_{d,MEAS}(c,t)$ is the measured dropping probability of class c at stage t, k_ω is a positive constant and t_{expl} is the number of stages used to explore the environment before starting to update the Lagrange multiplier: in fact, the simulation starts with no on-going connections and the measurements becomes significant only when the network is sufficiently loaded. This algorithm converges thanks to the fact that the gain of the variation step is $k_\omega/(t - t_{expl})$, which is driven to 0 as $t \to \infty$, whereas the quantity between the square brackets remains limited.

The main problems of this estimate, which are not addressed in this paper, are i) that the algorithm efficiency is sensitive to the gain constant k_ω; ii) that it cannot follow variations of traffic statistical characteristics (for instance, rules to re-start the estimate when the system performances are not satisfactory could be implemented).

The value of the maximum tolerated cost (see Theorem 2) is computed at every stage t as:

$$K(t) = P_{d,MAX}(c_d)\frac{\lambda_c}{\gamma}\left[1 - P_{b,MEAS}(c_d,t)\right], \quad \text{with } c_d = \arg\max_{c=1,\ldots,C}\left\{P_{d,MEAS}(c,t)/P_{d,MAX}(c)\right\}, \tag{46}$$

where $P_{b,MEAS}(c,t)$ is the measured blocking probability of class c at stage t. Note that, in a well-behaving communication network, the blocking probability is kept small – usually well below 10% – and, therefore, in practice, both the values of K in Theorem 2 and of $K(t)$ in equation (46) are close to the policy-independent value $P_{d,MAX}(c_d)[\lambda_c/\gamma]$. Nonetheless, the validity of this approach must be evaluated by simulation.

5. Numerical Simulation Results

Numerical simulations were performed with the aim of evaluating the effectiveness of the proposed approach. Table 1 summarizes the link model parameters, whereas Table 2 collects the parameters of the three classes supported by the considered link; σ is a parameter used to vary the offered load.

Five simulation sets were considered; 10 simulation runs per set were performed. To evaluate the overall algorithm performance in different traffic conditions, the five simulation sets were characterized by different values of σ, leading to different values of offered traffic load η_{OFF}, computed as $\eta_{OFF} = \sum_{c=1}^{C} \frac{\lambda_c}{\mu_c} e_c$. Table 3 shows the parameter setting selected for all the simulation sets.

For each run, the link, class, and simulation parameters were used to generate an event list; the events can be call births/terminations and link state variations. At each event, the RL algorithm is updated. At each call birth event, the admission controller uses the admission policy computed by the RL algorithm to decide whether to accept the call or not; at each link state variation, if the new link state

Table 1. Values of the link model

Link state load [Mbps]	Transition frequency matrix [min^{-1}]
$\xi_1 = 1$	$\Phi = \begin{bmatrix} 0 & 0.2641 & 0.1208 & 0.0302 & 0 & 0 \\ 0.0058 & 0 & 0.1968 & 0.2290 & 0.0526 & 0 \\ 0.0029 & 0.0049 & 0 & 0.5427 & 0.1715 & 0 \\ 0 & 0.0019 & 0.0107 & 0 & 0.5349 & 0.1647 \\ 0 & 0.0010 & 0.0049 & 0.0175 & 0 & 0.29331 \\ 0 & 0 & 0.0010 & 0.0117 & 0.0370 & 0 \end{bmatrix}$
$\xi_2 = 1.6$	
$\xi_3 = 2.2$	
$\xi_4 = 2.8$	
$\xi_5 = 3.4$	
$\xi_6 = \xi_L = 4$	

capacity ξ_l is lower than the current state capacity $\eta(t)$, the controller uses its dropping policy to decide which call to drop; if necessary, other call droppings are performed with rate f_{drop}.

Each simulation run was executed three times: the first time, referred to as *greedy*, a heuristic policy was implemented, which, in a greedy fashion, always accepts the calls whenever enough capacity is available, and always drops the on-going call with the highest bitrate; the second time, referred to as RL1, the RL algorithm was used, without dropping constraints and without considering fairness criteria; the third time, referred to as RL2, the RL algorithm was used with dropping constraints and fairness. To evaluate the obtained fairness, the Jain's fairness index ([8]) was considered, widely used in the literature, which rates the fairness with values between 0 and 1 (the greater the index, the fairer the policy):

$$FI(t) = \left(\sum_{c=1,\ldots,C} P_{bMEAS}(c,t) \right)^2 \Bigg/ \left(C \sum_{c=1,\ldots,C} (P_{bMEAS}(c,t))^2 \right). \quad (47)$$

Table 2. Class parameters

Class of service	Class 1	Class 2	Class 3
e_c [kbps]	64	150	300
λ_c [min^{-1}]	$2.125 \cdot \sigma$	$1.25 \cdot \sigma$	$0.5 \cdot \sigma$
μ_c [min^{-1}]	0.333	0.2	0.2

Table 3. Simulation parameters

Parameter	Value
f_{drop} [min^{-1}]	600
w	1
$P_{d,MAX}(c)$, $c = 1, 2, 3$	0.01
H	20
Number of aggregate states ($2(H-1)L$)	228
t_{expl} [stages]	100
k_ω	500
$\rho(t)$	$1/t$
α	0.95
ε	0.02
Number of runs per simulation	10
Length of each sim. run [h]	12

The first noteworthy result is that, thanks to the state aggregation procedure, the RL approach avoids the scalability problem of MDPs. In fact, the number of aggregate states is $2(H-1)L = 228$. To estimate the number of states *ML* required by the MDP described in Section 3.2, we consider that M grows as $\frac{1}{C!} \prod_{c=1}^{C} (N_c + 1)$, where $N_c = \lfloor \xi_L / e_c \rfloor$ is the maximum number of on-going class c calls ([15]): with the values of Tables 1–3, we obtain $ML \sim 1.8 \cdot 10^4$, which, considering the subsequent large action space and large number of transitions, is already too much from the implementation viewpoint. Note also that the RL algorithm could not be implemented without an aggregation policy, since the state augmentation described in Section 3.3 even worsens the scalability problem.

The average results of the simulation runs of each simulation set are summarized in Table 4 and in Figure 1, which show that:

– The performance of the *greedy* heuristic almost matches the RL1 policy, aimed at minimizing the total blocking probability; in this case, implementing the RL algorithm would not be convenient.

Table 4. Offered load η_{OFF}, average total blocking probability $P_{b,TOT}$, average load η_{AVG}, average fairness index *FI* and average maximum dropping probability $P_{d,MAX}$ for *greedy* policy, RL1, and RL2

Simulation set	1	2	3	4
η_{OFF} [Mbps]	2.42	2.64	2.86	3.08
$\eta_{AVG,\ greedy}$ [Mbps]	2.4185	2.4865	2.687	2.7989
$\eta_{AVG,\ RL1}$ [Mbps]	2.4145	2.4826	2.6761	2.7987
$\eta_{AVG,\ RL2}$ [Mbps]	2.4062	2.4688	2.6559	2.7568
$P_{b,\ TOT\ greedy}$ [%]	1.9754	2.22	3.2203	4.8182
$P_{b,\ TOT\ RL1}$ [%]	1.8386	2.1204	2.9454	4.5443
$P_{b,\ TOT,\ RL2}$ [%]	3.115	3.4296	5.2887	7.5324
FI_{greedy} [%]	79.215	77.8097	74.5456	77.4063
FI_{RL1} [%]	78.2469	75.6998	72.8562	75.1384
FI_{RL2} [%]	97.1701	95.7164	98.168	96.8869
$P_{d,\ MAX,\ greedy}$ [%]	1.1891	2.1713	2.4472	2.7987
$P_{d,\ MAX,\ RL1}$ [%]	0.97357	1.5183	1.5148	2.3527
$P_{d,\ MAX,\ RL2}$ [%]	0.9037	1.0133	0.9326	1.088

– The multiobjective cost function of RL2 manages to enforce fairness among classes without significantly affecting the average throughput: in all the simulations the fairness index was about 20%–25% higher than with RL1 and *greedy*.
– The dropping probability constraint is effectively enforced by RL2: the maximum dropping probability is always about 1% (which is the maximum value $P_{d,MAX}(c)$, $c = 1, \ldots, C$) whereas in RL1 and *greedy* it is uncontrolled and grows with the offered load beyond 1%.
– As the offered load increases, fairness and dropping constraint are enforced at the price of increasing the blocking probability and, consequently, of lowering the average load. The effect due to the dropping constraint becomes more evident as the offered load increases: at low loads, the RL algorithm evenly distributes the dropping probabilities among the classes to lower the maximum one, without significantly affecting the total blocking probability; at higher loads, the dropping probabilities must be further reduced by decreasing the acceptance probabilities (i.e., the denominator of equation (26)).

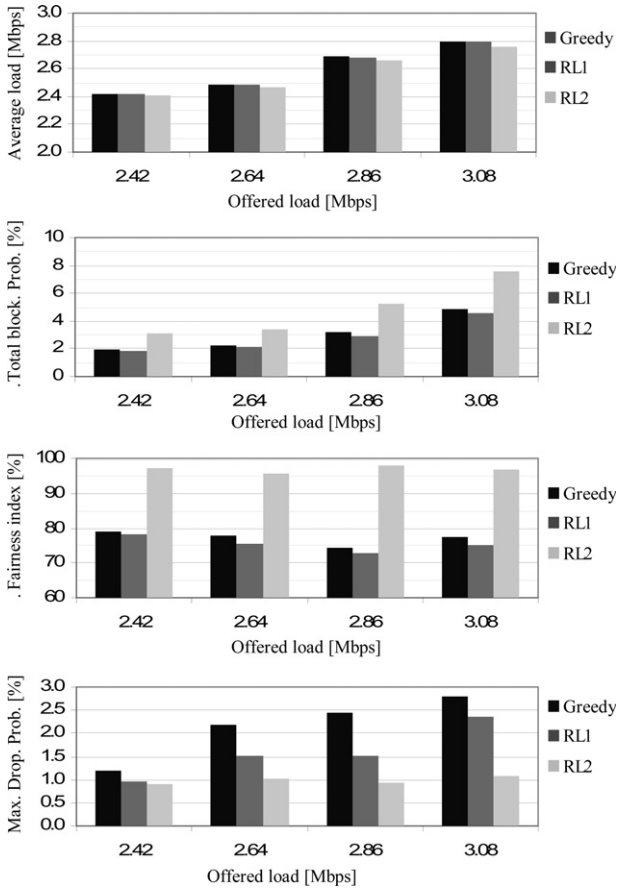

Fig. 1. Average load, average fairness index, and maximum dropping probability

Finally, Table 5 and Figure 2 show the effect of varying the state aggregation granularity. The scenario is the one of simulation set number 3 with RL2 policy. As expected, the table shows that a small number of aggregate states does not allow the controller to efficiently operate (in particular, it does not manage to increase the fairness index).

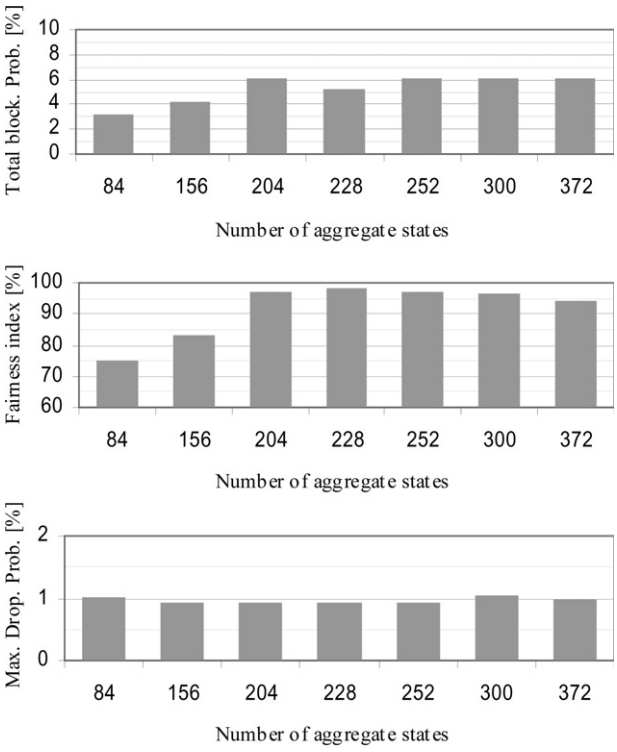

Fig. 2. Effect of the aggregation granularity: total blocking probability ($P_{b,TOT}$); fairness index (FI); maximum dropping probability ($P_{d,MAX}$)

Table 5. Effect of the aggregation granularity: total blocking probability ($P_{b,TOT}$); fairness index (FI); maximum dropping probability ($P_{d,MAX}$)

H	8	14	18	20	22	26	32
Aggregates $(2(H-1)L)$	84	156	204	228	252	300	372
$P_{b,TOT,RL2}$ [%]	3.25	4.16	6.07	5.29	6.07	6.14	6.14
FI_{RL2} [%]	74.92	83.40	96.84	98.17	96.84	96.76	94.09
$P_{d,MAX,RL2}$ [%]	1.00	0.92	0.91	0.93	0.91	1.04	0.99

6. Conclusions and Future Work

This article introduces a practical and innovative approach to deal with networks with time-varying link capacity, such as CDMA networks, DVB-S2 satellite networks, and WiMAX networks. The innovative approach consists in i) exploiting the Markov Decision Process (MDP) framework developed in [19] to define a constrained MDP; ii) defining an unconstrained MDP via the Lagrangian approach; iii) developing a Reinforcement Learning algorithm to solve on-line the MDP. Beside the standard admission policy, the algorithm returns a dropping policy to control the call dropping due to the variable link capacity; numerical simulations confirm the effectiveness of the proposed approach in controlling both blocking and dropping probabilities.

On-going work is aimed at achieving an estimation of the optimal Lagrange multiplier value adaptively with respect to variations of traffic and link statistics. In this respect, further research is considered within the ε-MDP framework ([25]).

References

1. Altman E. Applications of Markov decision processes in communication networks—-A survey. Tech. Rep, INRIA, Available from http://www.inria.fr/RRRT/RR-3984.html, 2000.
2. Beutler FJ, Ross KW. Optimal policies for controlled Markov chains with a constraint. *J Math Anal Appl* 1985; 112: 236–252.
3. Butler FJ, Ross KW. Time-average optimal constrained semi-Markov decision processes. *Adv Appl Prob* 1986; 18(2): 341–359.
4. Castanet L, Deloues T, Lemorton J. Channel modelling based on N-state Markov chains for satcom systems simulation. In: Twelfth International Conference on Antennas and Propagation (ICAP 2003), 2003; 1: 119–122.
5. Djonin DV, Krishnamurthy V. Q-learning algorithms for constrained markov decision processes with randomized monotone policies: application to MIMO transmission control. *IEEE Trans Signal Process* 2007; 55(5): 2170–2181.
6. ETSI Standard TR 102 376 V1.1.1: Digital Video Broadcasting (DVB) User guidelines for the second generation system for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications (DVB-S2), 2005.
7. IEEE standard for Local and Metropolitan Area Networks Part 16: Air interface for fixed broadband wireless access systems. IEEE Standard 802.16, 2004.
8. Jain RK, Chiu D-MW, Hawe WR. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Digital Equipment Corporation, Tech. Rep, 1984.
9. Kalyanasundaram S, Chong EKP, Shroff NB. Admission control schemes to provide class-level QoS in multiservice networks. *Comput Netw* 2001; 35: 307–326.
10. Kalyanasundaram S, Chong EKP, Shroff NB. Optimal resource allocation in multi-class networks with user-specified utility functions. *Comput Netw* 2002; 38: 613–630.
11. Lilith N, Dogancay K. Reinforcement learning-based dynamic guard channel scheme with maximum packing for cellular telecommunications systems. International Conference on Wireless Communications, Networking and Mobile Computing (WiCom 2007), 2007; 1967–1970.
12. Marbach P, Mihatsch O, Tsitsiklis JN. Call admission control and routing in integrated services networks using reinforcement learning. In Proceedings of the 37th IEEE Conference on Decision and Control, 1998; 1: 563–568.
13. Mignanti S, Di Giorgio A, Suraci V. A model based RL admission control algorithm for next generation networks. The Second International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST '08), 2008; 303–308.
14. Ni J, Tsang DHK, Tatikonda S, Bensaou B. Optimal and structured call admission control policies for resource-sharing systems. *IEEE Trans Commun* 2007; 55(1): 158–170.
15. Nordström E, Carlström J. Communication networks—A new reward model for MDP state aggregation with application to CAC and Routing. *Eur Trans Telecommun* 2005; 16: 495–508.
16. Park J-S, Huang L, Lee DC, Kuo C-CJ. Optimal code assignment and call admission control for OVSF-CDMA systems constrained by blocking probabilities. In Proceedings of IEEE GIobecom 2004; 3290–3294.
17. Perez-Romero J, Sallent O, Agusti R, Diaz-Guerra MA. *Radio Resource Management Strategies in UMTS*. John Wiley & Sons, 2005.
18. Pietrabissa A. Admission control in UMTS networks based on approximate dynamic programming. *Eur J Control* 2008; 14(1): 62–75.
19. Pietrabissa A. Optimal call admission and call dropping control in links with variable capacity. *Eur J Control* 2009; 15(1): 56–67.
20. Pimentel C, Falk T, Lisboâ L. Finite-state Markov modeling of correlated Rician-fading channels. *IEEE Trans Veh Technol* 2004; 53(5): 1491–1501.
21. Puterman ML. *Markov Decision Processes*. New Jersey, John Wiley & Sons, 1994.
22. Sanchez-Salas DA, Cuevas-Ruiz JL. N-states channel model using Markov chains. In Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007), 2007; 342–347.
23. Sennott LI. Computing average optimal constrained policies in stochastic dynamic programming. *Probab Engineer Inform Sci* 2001; 15(1): 103–133.
24. Sutton R, Barto AG. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 1998.
25. Szita I, Takacs B, Lörincz A. ε-MDPs: Learning in varying environments. *J Mach Learn Res* 2003; 3(1): 145–174.
26. Tong H, Brown TX. Adaptive call admission control under quality-of-service constraints: A reinforcement learning solution. *IEEE J Sel Areas Commun* 2000; 18(2): 209–220.
27. Tong H, Brown TX. Reinforcement learning for call admission control and routing under quality of service constraints in multimedia networks. *Mach Learn* 2002; 49: 111–139.
28. Usaha W, Barria J. Reinforcement Learning for resource allocation in LEO satellite networks. *IEEE Trans Syst Man Cybern B, Cybern* 2007; 37(3): 515–527.

Q1

29. Vucetic B, Du J. Channel Modeling and simulation in satellite mobile communication systems. *IEEE J Sel Areas Commun* 1992; 10(8): 1209–1218.
30. Yang X, Bigham J. A call admission control scheme using NeuroEvolution algorithm in cellular networks. In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI07), 2007; 186–191.
31. Yang X, Feng G. Optimizing admission control for multi-service wireless networks with bandwidth asymmetry between uplink and downlink. *IEEE Trans Veh Technol* 2007; 56(2): 907–917.
32. Yeung KL, Nanda S. Channel management in microcell/ macrocell cellular radio systems. *IEEE Trans Veh Technol* 1996; 45(4): 601–612.
33. Yu F, Krishnamurthy V, Leung VCM. Cross-layer optimal connection admission control for variable bit rate multimedia traffic in packet wireless CDMA networks. *IEEE Trans Signal Process* 2006; 54(2): 542–555.

Annex 1

Proof of Theorem 2: Let $\bar{\Pi}_{c',c''}$ be the subset of policies such that $c' = c_b$ and $c'' = c_d$:

$$\bar{\Pi}_{c',c''} = \left\{\bar{\pi} \in \bar{\Pi} \middle| c' = c_b, c'' = c_d\right\}, \quad c', c'' \in \{1, \ldots, C\}, \tag{A.1}$$

which induces a partition of the policy space, that is, $\bar{\Pi} = \bigcup_{c',c'' \in \{1,\ldots,C\}} \bar{\Pi}_{c',c''}$, and $\bar{\Pi}_{c',c''} \bigcap \bar{\Pi}_{c''',c'^{\nu}} = \emptyset$ if $(c', c'') \neq (c''', c'^{\nu})$. Accordingly, we further introduce the subsets $\bar{S}_{c',c''}$ defined as:

$$\bar{S}_{c',c''} = \left\{\bar{\mathbf{s}} \in \bar{S}_{AV} \middle| \bar{\mathbf{s}} = (\mathbf{s}, c')\right\} \bigcup \left\{\bar{\mathbf{s}} \in \bar{S}_{UN} \middle| \bar{\mathbf{s}} = (\mathbf{s}, c'')\right\}, \quad c', c'' \in \{1, \ldots, C\}, \tag{A.2}$$

which induces a partition of the augmented state space, that is, $\bar{S} = \bigcup_{c',c'' \in \{1,\ldots,C\}} \bar{S}_{c',c''}$, and $\bar{S}_{c',c''} \bigcap \bar{S}_{c''',c'^{\nu}} = \emptyset$ if $(c', c'') \neq (c''', c'^{\nu})$. Let also $\bar{A}_{c',c''}$ be the action space (10) defined over the subset $\bar{S}_{c',c''}$, and $\bar{\mathbf{T}}_{c',c''}$ be the matrix of the transitions between the states in $\bar{S}_{c',c''}$.

From definitions (23) and (30), it is clear that there is a one-to-one correspondence between each subset $\bar{S}_{c',c''}$ and the state space S. As noted in Remark 2, under a given policy $\bar{\pi}$, the only alive transitions are: 1) the ones between the available states $\bar{\mathbf{s}} = (\mathbf{s}, c_b)$, $\bar{\mathbf{s}} \in \bar{S}_{AV}$ and the unavailable states $\bar{\mathbf{s}} = (\mathbf{s}, c_d)$, $\bar{\mathbf{s}} \in \bar{S}_{UN}$; 2) the transition $t(\bar{\mathbf{s}}'\bar{\mathbf{s}}'')$ from $\bar{\mathbf{s}}' = (\mathbf{s}, c) \in \bar{S}_{AV}$ with $c \neq c_b$ to $\bar{\mathbf{s}} = (\mathbf{s}, c_b) \in \bar{S}_{AV}$, in case the system initial state is $\bar{\mathbf{s}}'$; 3) the transition $t(\bar{\mathbf{s}}'\bar{\mathbf{s}})$ from $\bar{\mathbf{s}}' = (\mathbf{s}, c) \in \bar{S}_{UN}$ with $c \neq c_d$ to $\bar{\mathbf{s}} = (\mathbf{s}, c_d) \in \bar{S}_{UN}$, in case the system initial state is $\bar{\mathbf{s}}'$. Since these last two transitions occur with probability 1, it follows that the states characterized by $\bar{\mathbf{s}} = (\mathbf{s}, c) \in \bar{S}_{AV}$ with $c \neq c_b$ and $\bar{\mathbf{s}} = (\mathbf{s}, c) \in \bar{S}_{UN}$ with $c \neq c_d$ are transient. Thus, for each subset of policies (A.1), the MDP $\left\{\bar{S}, \bar{A}, \bar{\mathbf{T}}, \bar{r}, d, K\right\}$ under a given policy $\bar{\pi} \in \bar{\Pi}_{c',c''}$ reduces to the MDP $\left\{\bar{S}_{c',c''}, \bar{A}_{c',c''}, \bar{\mathbf{T}}_{c',c''}, \bar{r}, d, K\right\}$ under policy $\bar{\pi}$, which is equivalent to the MDP $\{S, A, T, \bar{r}, d, K\}$ under a policy π such that $\pi(\mathbf{s}) = \begin{cases} \bar{\pi}(\mathbf{s}, c'), & \text{if } \mathbf{s} \in S_{AV} \\ \bar{\pi}(\mathbf{s}, c''), & \text{if } \mathbf{s} \in S_{UN} \end{cases}$, with $\bar{r}$ and d computed for c' and c'', respectively. Since Lemma 1 states that the MDP $\{S, A, T, \bar{r}, d, K\}$ is unichain (different cost functions does not affect the unichain property), Property i) follows.

Properties iii), iv) and ii) straightforwardly follows:

(iii) From equations (24) and (38) it follows that the expected cost is equal to:

$$\begin{aligned} R_{\bar{\pi}} &= \lim_{n\to\infty} \frac{1}{n} E_{\bar{\pi}} \left\{ \sum_{t=0}^{n-1} \bar{r}\left[\bar{\mathbf{s}}(t), \mathbf{u}(t)\right] \right\} \\ &= \sum_{\bar{\mathbf{s}} \in \bar{S}} \bar{r}\left[\bar{\mathbf{s}}, \mathbf{u}\right] P_{\bar{\pi}}(\bar{\mathbf{s}}) = \sum_{\bar{\mathbf{s}} \in \bar{S}_{c_b,c_d}} \bar{r}\left[\bar{\mathbf{s}}, \mathbf{u}\right] P_{\bar{\pi}}(\bar{\mathbf{s}}) \\ &= \sum_{c=1}^{C} P_{b,\bar{\pi}}(c) + w P_{b,\bar{\pi}}(c_b), \end{aligned} \tag{A.3}$$

The first term of (A.3) evaluates the sum of the blocking probabilities, whereas the second term, weighted by the parameter w, evaluates the maximum blocking probability among the classes of service and, thus, enforces fairness among classes (see equation (21)).

(iv) Considering the cost function (32), by computing the expected cost and by dividing it by the admission probability of class c_d, $\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]$, the following equation is obtained (see equation (27):

$$\begin{aligned} \frac{D_{\bar{\pi}}}{\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]} &= \frac{\lim_{n\to\infty} \frac{1}{n} E_{\bar{\pi}} \left\{ \sum_{t=0}^{n-1} d\left[\bar{\mathbf{s}}(t), \mathbf{u}(t)\right] \right\}}{\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]} \\ &= \frac{\sum_{\bar{\mathbf{s}} \in \bar{S}_{UN}} d(\bar{\mathbf{s}}, \mathbf{u}) P_{\bar{\pi}}(\bar{\mathbf{s}})}{\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]} \\ &= \frac{\sum_{\bar{\mathbf{s}} \in \bar{S}_{c_b,c_d} \cap \bar{S}_{UN}} u(\bar{\mathbf{s}}, \mathbf{u}) \frac{f_{drop}}{\gamma} P_{\bar{\pi}}(\bar{\mathbf{s}})}{\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]} \\ &= P_{d,\bar{\pi}}(c_d), \end{aligned} \tag{A.4}$$

Enforcing the single constraint $D_{\bar{\pi}} < K \leq P_{d,MAX}(c_d)\frac{\lambda_c}{\gamma}\left[1 - P_{b,\bar{\pi}}(c_d)\right]$ is then equivalent to enforcing constraint (28) (or constraints (27)).

(ii) Finally, to prove point ii), it is sufficient to show that there is always at least one feasible policy: the policy $\bar{\pi}'$ which blocks all the calls is always feasible: in fact, in this case the recurrent states are the empty states $\bar{\mathbf{s}} = (\mathbf{s}, c) = (\mathbf{x}, \xi_l, c)$ such that $\mathbf{x} = (0, \ldots, 0)$ (see Lemma 1), which are available. Since cost (32) is null for the available states, we have that $D_{\bar{\pi}'} = 0 < K$. □