Reinforcement Learning for Resource Allocation in LEO Satellite Networks

Wipawee Usaha, *Member, IEEE*, and Javier A. Barria, *Member, IEEE*

Abstract—In this paper, we develop and assess online decision-making algorithms for call admission and routing for low Earth orbit (LEO) satellite networks. It has been shown in a recent paper that, in a LEO satellite system, a semi-Markov decision process formulation of the call admission and routing problem can achieve better performance in terms of an average revenue function than existing routing methods. However, the conventional dynamic programming (DP) numerical solution becomes prohibited as the problem size increases. In this paper, two solution methods based on reinforcement learning (RL) are proposed in order to circumvent the computational burden of DP. The first method is based on an actor–critic method with temporal-difference (TD) learning. The second method is based on a critic-only method, called optimistic TD learning. The algorithms enhance performance in terms of requirements in storage, computational complexity and computational time, and in terms of an overall long-term average revenue function that penalizes blocked calls. Numerical studies are carried out, and the results obtained show that the RL framework can achieve up to 56% higher average revenue over existing routing methods used in LEO satellite networks with reasonable storage and computational requirements.

Index Terms—Call admission control (CAC), low Earth orbit (LEO) satellite network, reinforcement learning (RL), routing, temporal-difference (TD) learning.

I. INTRODUCTION

DYNAMIC resource allocation in low Earth orbit (LEO) satellite systems has been the focus of intensive investigation in recent years. Much research interest has centered on satellite-fixed cells where the traffic served by each satellite changes with time as the satellite footprint moves across regions. The advent of intersatellite links (ISLs)—an optical or radio link established between satellites—enables calls to be routed along satellites in the space segment to provide robust and efficient communication. ISL connectivity changes over time in terms of continuous-time variations of link distance and discrete-time link activation and deactivation as satellites move along their orbits. Therefore, when ISLs are included in LEO, satellite routing decisions, not only changing traffic patterns

between satellites but also the time-varying ISL topology must be considered.

Therefore, most of the existing connection-oriented routing algorithms in LEO satellite networks select the shortest (least cost) and the most stable path (defined as a path whose links have high probability of link availability). The suggested link costs are of several forms, e.g., a constant (minimum hop) [17], [18], [22]; link usage or link occupancy [17]; offered link load [25]; average propagation delay [23]–[25]; and the derivative of the average delay with respect to link flow [6]. Apart from [3] and [8] which account for the rerouted traffic in the near future using traffic estimation and deterministic topology changes, the link costs used in previous works do not account for changes in topology and therefore can result in a high number of forced terminated calls. Even though such benefits are considered in [8], only the single-service scenario was investigated. Furthermore, the proposed future (rerouted) traffic estimator algorithm in [8] is based on the solution of a continuous-time Markov chain and hence does not scale well when link capacity is large. Scalability of the algorithm will further deteriorate if multiple traffic services scenarios need to be considered. The problem studied in [3] supports multiple traffic services, but their algorithm admits all new calls as long as there is residual bandwidth to accommodate them.

In this paper, we propose a selective call admission and routing policy which accounts for the rerouted traffic and selectively admits new calls—even when residual bandwidth is available—aiming for a more efficient use of resources. The effect of rerouted traffic caused by the changing satellite topology is hence here considered so that forced termination of rerouted calls due to insufficient resources is minimized. In [19], it was shown that routing strategies determined from a semi-Markov decision process (SMDP) formulation can minimize the dropping of both new and rerouted traffic. However, the computational complexity of the algorithms proposed in [19] makes the solution impractical for even small size networks. Furthermore, it is well known that solving the SMDP formulation with conventional dynamic-programming (DP) methods can still be too complex to solve—even with simplifications and suitable approximations. Recently, reinforcement learning (RL) methods [36] (also referred to as neuro-dynamic programming [30]) have been suggested to determine nearly optimal policies instead. RL is a method for learning to make a decision (i.e., how to map environment states or situations into actions) in order to, for example, maximize a numerical reward signal. Our motivation to study further applications of RL methods lies on the fact that they can provide nearly optimal solutions to complex DP problems through experience from simulations

Manuscript received January 10, 2006; revised May 23, 2006. This work was supported by the Royal Thai Government and was carried out during the author's stay at Imperial College London. This paper was recommended by Associate Editor S.-F. Su.

W. Usaha is with the School of Telecommunication Engineering, Suranaree University of Technology, Nakorn Ratchasima 30000, Thailand (e-mail: wipawee@sut.ac.th).

J. A. Barria is with the Department of Electrical and Electronic Engineering, Imperial College London, SW7 2BT London, U.K. (e-mail: j.barria@imperial.ac.uk).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TSMCB.2006.886173

and interaction with the environment at a low computational cost. In an RL-based approach, a decision-maker (agent) gets to know a representation of the environment's state and decides to select an action. When the action is taken, a reward signal along with the new environment's state are fed back to the agent. The sequence actions–reward is repeated continually with the aim of maximizing the agent's average reward, which is defined as a specific function of the immediate reward sequence. The vast majority of RL techniques fall into one of the following categories.

- 1) Actor-only methods operate under parameterized family of probabilistic policies [29], [32], [35], [40]. The parameterized components of the actor are updated so as to improve the gradient of some performance measure (with respect to the parameters). A disadvantage of these methods is the large variance of the gradient estimators (estimated directly from simulations) which results in a slow learning rate. Variance reduction techniques have hence been suggested [31], [39].
- 2) Critic-only methods rely on value function approximation¹ and aim at solving for an approximation to the Bellman equation [27], [30], [34], [37], [38]. Then, a greedy policy based on the approximated value function is applied aiming at a nearly optimal policy. Critic-only methods optimize a policy indirectly through learning the value function approximations. Unfortunately, for these methods the policy improvement is not always guaranteed (it is only guaranteed in limited settings [30]), even if good approximations of the value functions are obtained.
- 3) Actor–critic methods combine the strong features of the two methods together [9], [10], [28], [33]. The critic stage attempts to learn value functions from simulation, and uses them to update the actor's policy parameters in the direction of improvement of the performance gradient. Hence, the gradient estimate depends on the value function approximation provided by the critic, and it uses this approximation to improve a randomized policy according to some performance criterion. It is expected that policy improvement is guaranteed as long as the policy is gradient-based (strong feature of actor-only methods) and faster convergence is achievable through variance reduction (from learning the value functions in critic-only methods).

In relation to the underlying problem of this paper, successful applications of RL methods have been reported in packet routing problems [1], [12], [15], and have been employed for bandwidth allocation in differentiated services networks [7]. RL has also been applied to cognitive packet networks [5], which is an alternative packet network architecture where an RL algorithm stored in the packets are executed to make routing decisions. The problem of call admission control (CAC) and routing is formulated in [20] as a continuous-time average-reward finite-state DP problem. RL has also been successfully applied in combined CAC and routing problems [2], [11], [14].

It is worth noting that the CAC and routing schemes developed so far are based mainly on critic-only RL methods which allow an interpretation of shadow prices (or implied costs) in the decision scheme whereby the value functions used for the computation of shadow prices are determined by means of RL methods instead of DP methods. For example the RL method in [11] uses a variant of temporal-difference (TD) learning with optimistic TD(0) policy iteration. However, despite many existing successful applications of optimistic policy iteration (OPI) methods in the literature [11], [13], [16], [26], the theoretical understanding of the convergence properties of such method is still limited. For example, a convergence problem is observed in [11] under some network configuration. Therefore, these approaches rely much on off-line training and trial-and-error initial parameter tuning. More recently in [41], RL methods have been applied to solve route discovery in mobile *ad hoc* networks (MANETs).

Due to the undesirable convergency problems of previously proposed methods, we derive and implement an alternative RL method. The method builds on Konda's actor–critic algorithm for discrete-time MDPs [9], [10] which uses a linear approximation architecture in the critic part. It should be noted that, to the best of our knowledge, this is the first development of this actor–critic algorithm toward the continuous-time domain and the first application of this version to an engineering problem. We also extend the RL method in [11] to solve resource allocation problems in LEO satellite networks and use it as a further benchmark. A comprehensive discussion on the key features of the RL method in [11] which makes it suitable for finding an alternative CAC and routing policy in LEO satellite networks can be found in [21]. The critic-only method developed here caters both multiple service ongoing call arrivals in addition to new call arrivals unlike [2], [11], and [14] where only new call arrivals are considered. Note that in the developed algorithms the additional consideration of the ongoing calls affects the optimal routing policy as more new calls may be rejected to reserve resources for future ongoing calls.

This paper is organized as follows. In the next section, the problem formulation is presented. Section III presents the two proposed RL methods and in Section IV the suitability of RL methods to solve the SMDP problem presented in this paper is discussed and assessed. This paper concludes in Section V.

II. PROBLEM FORMULATION

Due to the orbiting nature of LEO satellites, the changing satellite topology can be regarded as snapshots of quasi-stationary satellite topologies that will be visible (available) in a periodical order. In this paper, we assume that all ISLs have equal capacity C , that new call requests arrive into the system according to a Poisson distribution with rate $\lambda_{j,N}$, and that their call holding time are exponentially distributed with parameter $1/\mu_j$ for each one of class- j calls, $j \in J = \{1, \dots, K\}$. As a result, the set of alternative routes predefined for class- j calls also varies as the satellite network topology changes. Note also that in the proposed scheme the final decision as to whether accept or not a call is made by the node o where the call is originated.

¹A value function is an evaluation of the expected future reward to be incurred expressed as a function of the current state.

Fig. 1. Illustration of three possible handovers scenarios in a LEO satellite system.

Given a satellite topology, say the n th topology TP_n , the state space $\mathbf{S}$ of the network is given by

$$\mathbf{S} = \left\{ \mathbf{x} = [y_N(j, \mathbf{r}), y_{HO}(j, \mathbf{r})], \forall j \in J, \forall \mathbf{r} \in W_j : \sum_{j \in J} \sum_{\mathbf{r} \in W_j \cap \{s\}} y(j, \mathbf{r}) b_j \leq C, \forall s \right\}$$

where b_j is the bandwidth requirement of class- j call, $y(j, \mathbf{r}) = y_N(j, \mathbf{r}) + y_{HO}(j, \mathbf{r})$, where $y_N(j, \mathbf{r})$ ($y_{HO}(j, \mathbf{r})$) is the number of new (handover) class- j connections on route $\mathbf{r}$, W_j is the alternative routes set for class- j call, and $W_j \cap \{s\}$ is the set of routes which use link s .

Let Ω denote the finite set of all possible events where

$$\Omega = \left\{ \omega = [\omega_N(1), \dots, \omega_N(K), \omega_{HO}(1), \dots, \omega_{HO}(K)] : \omega \in \{-1, 0, 1\}^{2K} \right\}.$$

An event ω is given by the following description: 1) $\omega_N(j) = 1$ for a new call arrival event of class- j ; 2) $\omega_N(j) = -1$;

$\omega_{HO}(j) = 1$ for an event where a new call of class- j encounters handover for the first time; 3) $\omega_{HO}(j) = 1$ for a subsequent handover call arrival event of class- j ; and 4) $\omega_{N,HO}(j) = -1$ for a new/handover call termination of class- j . Note that in all cases 1)–4) all other elements in ω are 0. Furthermore, we note that in a fixed communication network the updates are performed at epochs of call arrival and call departure at the links which are associated with event ω_k . In contrast, in the LEO satellite system scenario considered here, state transitions that occur due to handovers must also be accounted for. Let m and n be two different satellites then, the following three additional cases need to be included (see also Fig. 1).

- 1) A handover call arrival from an old route to a new route between satellite m and satellite n . In this case, an handover event is treated as a call departure, i.e., the resource along the old route is released and an update is performed at the links associated to call departure. In addition, this handover event is also treated as a call arrival, i.e., a decision must be made to admit and assign a new route to the associated call. If the call can be admitted, the resource along the new route must be reserved. Otherwise, the call is blocked and the state of the links in the new route remains unchanged. Updating is performed on all links in the old and new routes.

- 2) A handover call arrival from an old route between satellite m and satellite n to, for example, satellite n . In this case, the handover event is treated as a call departure. Note that the associated call never actually leaves the system, but all the resources in the old route are released and resources on the updown link of the receiving satellite is used up instead. Updating is performed at the links associated to the call departure.
- 3) A handover call arrival from, for example, satellite m to a new route between satellite m and satellite n . For this case, the handover event is treated as a call arrival. Note that the total number of calls in the system never actually changes and the resources in the updown link are released. A decision must be made to admit and assign a new route to the considered call, and resource along the new route is reserved. Updating is performed on all the links associated to the new route. Otherwise, the handover call is blocked and the link states along the new route remains unchanged. Further details on the updating procedure to accommodate all ongoing call arrival scenarios can be found in [21].

Let $A(\mathbf{x}, \omega)$ denote the set of allowed actions when the current state and event is $(\mathbf{x}, \omega)$ such that $A(\mathbf{x}, \omega) \subset A = \bigcup A_j$ where $A_j = W_j \cup \{0\}$. An action $a \in A(\mathbf{x}, \omega)$ such that $\sum_{j=1}^{2K} \omega(j) > -1$, can be of two types: 1) reject a call and 2) admits the (either new or handover) call to route $\mathbf{r}$ from the set of available routes $\{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}$ where $\mathbf{r}_i \in W_j$ and n_j is the number of routes for class- j of, e.g., topology TP_n . For call termination events, no action is required since all call terminations must be allowed.

Let π be a stationary policy mapping the current event and state of the network to an action. Under policy π , the process $\{\mathbf{x}_k, \omega_k\}$ is an embedded finite state Markov chain evolving in continuous time. Note that in [19], it has been shown that the mean holding time between state transitions no longer needs to be Markovian. Note also that even though the chain evolves in continuous time, we only need to consider the system state at epochs where the events and decisions take place.

Let t_k be the time at which event $\omega_k \in \Omega$ occurs, and let $\mathbf{x}_k$ be the state of the system during interval $[t_{k-1}, t_k)$. Then, following an action a the system transits to the next state $\mathbf{x}'$ and an immediate reward is generated

$$g(\mathbf{x}_k, \omega_k, a_k) = \begin{cases} \zeta_{j,N}, & \text{if } \mathbf{x}' \in \mathbf{S}, \omega_k \leftarrow w_N(j) = 1 \\ \zeta_{j,\text{HO}}, & \text{if } \mathbf{x}' \in \mathbf{S}, \omega_k \leftarrow w_{\text{HO}}(j) = 1 \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

That is, an immediate reward of $\zeta_{j,N} \in \mathcal{R}^+$ ($\zeta_{j,\text{HO}} \in \mathcal{R}^+$) is received for admitting a new (handover) call of class- j . The objective is to find a stationary policy π^* which maximizes the average reward such that $v(\pi^*) \geq v(\pi)$ for every other policy π

$$v(\pi) = \lim_{N \rightarrow \infty} \frac{\sum_{k=0}^{N-1} g(\mathbf{x}_k, \omega_k, a_k)}{t_N} \quad (2)$$

where $a_k = \pi(\mathbf{x}_k, \omega_k)$ and t_N is the completion time of the N th transition. Under the unichain assumption, which states

that under any stationary policy π a state can be reached by any other state under π , the limit in (2) exists and is independent of the initial state. The optimal policy π^* can then be determined by solving Bellman's equation for average reward SMDP

$$h^*(\mathbf{x}) + v(\pi^*)\bar{\tau}(\mathbf{x}, a) = \arg \max_{a \in A(\mathbf{x}, \omega)} \{E_\omega [g(\mathbf{x}, \omega, a) + h^*(\mathbf{x}')]\} \quad (3)$$

for the solution pair $v(\pi^*)$ and h^* , where $v(\pi^*)$ is the optimal average reward, h^* is the vector of $|\mathbf{S}|$ differential reward functions, $|\mathbf{S}|$ is the size of the state space, $\bar{\tau}(x, a)$ is the average transition time corresponding to state-action pair (x, a) , $E_\omega[\cdot]$ is the expectation over the probability of events, and $\mathbf{x}'$ is the next state which is a deterministic function of $\mathbf{x}$, ω , and a . The function $h^*(x)$ has the following interpretation. Under policy π^* , $h^*(x) - h^*(y)$ is the difference in the optimal expected reward over an infinitely long time when starting in the state x rather than state y , where $x, y \in \mathbf{S}$.

III. DETERMINING ROUTING POLICY

A. Actor-Critic Semi-Markov Decision Algorithm (ACSM DP)

Here, we proposed a novel algorithm. In this algorithm policy learning is performed at each (satellite) node and the resulting policy is a randomized policy, that is, given an event and network state, the policy maps to each state a distribution over the set of available actions. The embedded Markov chains $\{\mathbf{x}_k, \omega_k, a_k\}$ in the satellite network evolves within state space $(\mathbf{S} \times \Omega) \times A$ and the distribution of the holding times between state transitions could be nonexponential [19]. Let μ_θ be a randomized policy parameterized by a vector $\theta \in \mathcal{R}^M$ (M is the number of tunable parameters) for a satellite node in a given topology say TP_n . That is μ_θ is a mapping from the current event and state of the network to a distribution of actions. The objective is then to find, for each satellite node, a policy $\mu_\theta(a|\mathbf{x}_k, \omega_k)$ which will translate into an action a_k (accept or reject a call) corresponding to μ_{θ^*} such that $v(\mu_{\theta^*}) \geq v(\mu_\theta)$ for every other μ_θ where $v(\mu_\theta)$ is given by (2).

Let $\mathcal{W} = (\mathbf{x}, \omega) \in \mathbf{S} \times \Omega$. Provided that certain assumptions are satisfied to ensure existence of the gradient (see [10] and [21]), the parameterized policy μ_{θ^*} can be found from improving the gradient of the average reward for SMDP $\nabla v(\mu_\theta)$ given by

$$\begin{aligned} \nabla v(\mu_\theta) &= \frac{\sum_{\forall \mathcal{W} \in \mathbf{S} \times \Omega} \sum_{\forall a \in A(\mathcal{W})} p_\theta(\mathcal{W}) \mu_\theta(a|\mathcal{W}) \psi^\theta(\mathcal{W}, a) Q^\theta(\mathcal{W}, a)}{\sum_{\forall \mathcal{W} \in \mathbf{S} \times \Omega} p_\theta(\mathcal{W}) \sum_{\forall a \in A(\mathcal{W})} \mu_\theta(a|\mathcal{W}) \bar{\tau}(\mathcal{W}, a)} \end{aligned} \quad (4)$$

where ∇ denotes the gradient with respect to θ and

$$\begin{aligned} \psi^\theta(\mathcal{W}, a) &= [\psi_0^\theta(\mathcal{W}, a), \dots, \psi_{M-1}^\theta(\mathcal{W}, a)]^T \\ \psi_j^\theta(\mathcal{W}, a) &= \frac{\partial}{\partial \theta(j)} \mu_\theta(a|\mathcal{W}) \\ &= \frac{\mu_\theta(a|\mathcal{W})}{\mu_\theta(a|\mathcal{W})}, \quad j = 0, \dots, M-1 \end{aligned}$$

$$Q^\theta(\mathcal{W}, a) = g(\mathcal{W}, a) - v(\theta)\bar{\tau}(\mathcal{W}, a) + \sum_{\forall \mathcal{W}'} p_{\mathcal{W}\mathcal{W}'}(a) h^\theta(\mathcal{W}')$$

$$\bar{\tau}(\mathcal{W}, a) = \sum_{\forall \mathcal{W}'} \int_0^{\infty} \tau q_{\mathcal{W}\mathcal{W}'}(d\tau, a)$$

$$q_{\mathcal{W}\mathcal{W}'}(\tau, a) = \Pr[\tau_{k+1} \leq \tau, \mathcal{W}_{k+1} = \mathcal{W}' | \mathcal{W}_k = \mathcal{W}, a_k = a]$$

where $\tau_{k+1} = t_{k+1} - t_k$ is the time between the transition $\mathcal{W}_k$ to $\mathcal{W}_{k+1}$. The derivation of $\nabla v(\mu_\theta)$ is an extension for the SMDP case and can be found in Appendix C.

The function $Q^\theta(\mathcal{W}, a)$ is the action value function of starting in state-event action tuple $(\mathcal{W}, a)$ and following policy μ_θ thereafter. However, since the exact model of the system is assumed unknown, $Q^\theta(\mathcal{W}, a)$ must then be estimated from simulation. Let the estimated action value function $\tilde{Q}_u^\theta(\mathcal{W}, a)$ be defined as

$$\tilde{Q}_u^\theta(\mathcal{W}, a) = \sum_{i=0}^{K-1} u(i) \phi_i^\theta(\mathcal{W}, a) \quad (5)$$

where $\mathbf{u} = [u(0), \dots, u(K-1)]^T$ is a parametric vector, $\phi^\theta(\mathcal{W}, a) = [\phi_0^\theta(\mathcal{W}, a), \dots, \phi_{K-1}^\theta(\mathcal{W}, a)]^T$ is the feature vector of state-action pair $(\mathcal{W}, a) \in (\mathbf{S} \times \Omega) \times A$.

For the multiservice network scenario such as the one considered in this paper, we use TD(λ) critic learning with $0 \leq \lambda < 1$ and linear function approximation so that the critic can be updated at every time step. TD methods do not require a model of the environment and hence they can naturally be implemented in online applications. TD(λ) methods seamlessly integrate TD and Monte Carlo methods ($\lambda = 0$ represents the most extreme form of bootstrapping, and $\lambda = 1$ represents no bootstrapping (Monte Carlo methods [36]). Note also that the feature structure $\phi^\theta(\mathcal{W}, a)$ depends on θ and the TD(λ) critic type and should be chosen in such a way that it satisfies certain assumptions on stepsize and critic features (critic convergence analysis can be found in [10] and [21]).

1) *ACSM DP—Policy $\mu_\theta(a|\mathcal{W})$ and Feature Structure:* Let μ_θ be a randomized policy parameterized by $\theta \in \mathcal{R}^M$ for a satellite node in a given topology say TP_n . The objective is then to find, for each satellite node, a policy corresponding to μ_{θ^*} such that $v(\mu_{\theta^*}) \geq v(\mu_\theta)$ for every other μ_θ where $v(\mu_\theta)$ is given by (2). If the set of routes available is $\{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}$ where $\mathbf{r}_i \in W_j$, then the node o where the call was originated (event ω_k) admits the call to some route with some probability $\mu_\theta(a|\mathcal{W})$ where $a \in \{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}$, or rejects the call with probability $\mu_\theta(0|\mathcal{W})$. If ω_k corresponds to a call completion, the event is accepted with probability 1. If ω_k corresponds to a call arrival, but the node is not where the call was originated, the event is accepted with probability 1—meaning that any subsequent node that is dealing with the request must accept any decision to ω_k that node o takes. The actions just described correspond to the policy $\mu_\theta(a|\mathcal{W})$ given by

$$\mu_\theta(a|\mathcal{W}) : \begin{cases} a = \text{accept with} \\ \text{probability } p_\theta(\mathcal{W}, a), & \text{if } \omega_k \leftarrow \omega_N(j) = 1 \\ a = \text{accept}, & \text{if } \omega_k \leftarrow \omega_{\text{HO}}(j) = 1 \\ a = \text{accept}, & \text{if } \omega_k \leftarrow \omega_{N, \text{HO}}(j) = -1. \end{cases} \quad (6)$$

TABLE I
EVALUATION OF DIFFERENT STATE REPRESENTATIONS $B(x, a)$, $a > 0$

$B(\mathbf{x}, a), a > 0$	$G(\pi)$ (8)
$\frac{\sum_{\forall s \in \mathbf{r}} \sum_{\forall j \in J} x_j^s b_j}{ \mathbf{r} }$, $ \mathbf{r} = \text{number of links in route } \mathbf{r}$	328.51
$\text{resid}(\mathbf{r})$	336.02
$\text{resid}(\mathbf{r}) + \mathbb{I}(\text{resid}(\mathbf{r}) < 10)$	334.80
$\text{resid}(\mathbf{r}) + \mathbb{I}(\text{resid}(\mathbf{r}) < 15)$	333.82

The function $p_\theta(\mathcal{W}, a)$ is the probability of selecting action a according to parameter vector $\theta \in \mathcal{R}^M$. In this paper, the distribution $p_\theta(\mathcal{W}, a)$ in the form (7) is chosen as it defines the probability of selecting an action which has a continuous differentiable function of θ , and the scalar function $B(\mathbf{x}, a)$ is the state representation that reflects the current network state

$$p_\theta(\mathcal{W}, a) = \frac{\exp(s_\theta(a))}{\sum_{\forall u \in A} \exp(s_\theta(u))} \quad (7)$$

$$s_\theta(a) = B(\mathbf{x}, a) + \theta(\omega, a).$$

We assessed several forms of $B(\mathbf{x}, a)$ as shown in Table I using the net revenue (8) as criteria

$$G(\pi) = \sum_{\forall j} \zeta_{j,N} \bar{\lambda}_{j,N} (1 - \bar{B}_{j,\text{HO}}) \quad (8)$$

where $\bar{\lambda}_{j,N}$ is the average accepted rate of new class- j calls, $\zeta_{j,N}$ is the reward for class- j new calls, and $\bar{B}_{j,\text{HO}}$ is the handover blocking probability class- j calls. The considered forms of $B(\mathbf{x}, a)$ for $a > 0$ include the average bandwidth occupation, the residual route bandwidth, the residual route bandwidth with indicator functions when it falls below a threshold value. The form (9) gave the highest net revenue over all other considered forms

$$\begin{aligned} B(\mathbf{x}, a) &= \begin{cases} \text{resid}(\mathbf{r}), & \text{if } a = \mathbf{r} \in \{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\} \\ \max_{\mathbf{r} \in \{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}} \{\text{resid}(\mathbf{r})\}, & \text{if } a = 0 \end{cases} \\ \text{resid}(\mathbf{r}) &= \min_{s \in \mathbf{r}} \left\{ C - \sum_{\forall j \in J} x_j^s b_j \right\} \times \frac{1}{C}. \end{aligned} \quad (9)$$

Note that under this form $B(\mathbf{x}, a)$ reflects the congestion status of the network associated to the call arrival event—a significant feature when making routing decisions. Hence, for a given action $a = \mathbf{r} \in \{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}$, $B(\mathbf{x}, a)$ represents the residual bandwidth on each route $\mathbf{r}$ and $B(\mathbf{x}, 0)$ is the maximum residual bandwidth among all available routes. Note that for all forms in Table I, $B(\mathbf{x}, 0)$, is the same as in (9).

In addition, for each class j , there are $n_j + 1$ available actions (i.e., admitting the call to one of n_j routes and call rejection). Hence, for all $\mathcal{K}$ classes, the number of tunable actor parameters is $M = \mathcal{K} + \sum_{\forall j \in \mathcal{K}} n_j$. Bearing this in mind and

the fact that we have chosen to use the TD(λ) critic, the actor and critic feature vectors take the following forms:

$$\psi_i^\theta(\mathcal{W}, a) = \frac{\partial}{\partial \theta(i)} \mu_\theta(a|\mathcal{W}), \quad \text{for } i = 0, \dots, M-1 \quad (10)$$

$$\phi_i^\theta(\mathcal{W}, a) = \begin{cases} \psi_i^\theta(\mathcal{W}, a), & \text{for } i = 0, \dots, M-1 \\ \phi_i^\theta(\mathcal{W}), & \text{for } i = M \end{cases} \quad (11)$$

where

$$\phi_i^\theta(\mathcal{W}) = B(\mathbf{x}, 0) = \max_{\mathbf{r} \in \{\mathbf{r}_1, \dots, \mathbf{r}_{n_j}\}} \{\text{resid}(\mathbf{r})\}.$$

Note that with this choice of policy structure and features, the number of tunable actor parameters is $|\theta| = M$ and the number of tunable critic parameters is $|\mathbf{u}| = |\theta| + 1$. These parameters are updated as presented in the ACSMDP algorithm in Appendix B.

B. Optimistic Policy Iteration (OPI)

The central idea of this approach is to approximate $v(\pi^*)$ by a scalar $\tilde{v}$ and $h^*(x)$ by a vector $\tilde{h}_q$ parameterized by some vector $\mathbf{q} \in \mathcal{R}^K$ obtained through some training process (e.g., via simulation or interacting with the environment directly). The approximate optimal policy is then obtained from the greedy policy

$$\pi(\mathbf{x}, \omega) = \arg \max_{a \in A(\mathbf{x}, \omega)} \{g(\mathbf{x}, \omega, a) + \tilde{h}_q(\mathbf{x}')\} \quad (12)$$

where $\mathbf{x}'$ is the next network state resulting from taking action a at $(\mathbf{x}, \omega)$. In this method, the following linear approximation architecture is employed:

$$\tilde{h}_q(\mathbf{x}) = \sum_{i=0}^{K-1} q(i) \phi_i(\mathbf{x}) \quad (13)$$

where $\mathbf{q} = [q(0), \dots, q(K-1)]^T$ is a parametric vector and $\phi(\mathbf{x}) = [\phi_0(\mathbf{x}), \dots, \phi_{K-1}(\mathbf{x})]^T$ is the feature vector for state $\mathbf{x} \in \mathbf{S}$. Let the parametric and feature vector be defined such that

$$\begin{aligned} \tilde{h}_q(\mathbf{x}) = \sum_{\forall s \in \mathcal{L}} & \left[q(0, s) + \sum_{j=1}^{\mathcal{K}} q(j, s) x_N(j, s) \right. \\ & + \sum_{j=1}^{\mathcal{K}} q(j + \mathcal{K}, s) x_{\text{HO}}(j, s) \\ & + \sum_{j=1}^{\mathcal{K}} q(j + 2\mathcal{K}, s) (x_N(j, s))^2 \\ & + \sum_{j=1}^{\mathcal{K}} q(j + 3\mathcal{K}, s) x_N(j, s) x_{\text{HO}}(j, s) \\ & \left. + \sum_{j=1}^{\mathcal{K}} q(j + 4\mathcal{K}, s) (x_{\text{HO}}(j, s))^2 \right] \quad (14) \end{aligned}$$

where $\mathbf{q} = [q(0, s), \dots, q(5\mathcal{K}, s)]^T$, $\forall s \in \mathcal{L}$, is the parametric vector for, e.g., topology TP_n . $x_N(j, s)(x_{\text{HO}}(j, s))$ is the number of new (handover) calls of class- j on routes using link s , and $\mathcal{L}$ is the set of available links in topology TP_n . Note that similar quadratic features (14) have been used in [11]. For this approximation, the number of parameters required for tuning in topology TP_n is $|\mathcal{L}|(1 + 5\mathcal{K})$.

If a new call of class- j is accepted the number of class- j new calls becomes $x_N(j, s) + 1$ on link $s \in \mathbf{r}$. The net gain of admitting a class- j new/handover (N/HO) call to some route $\mathbf{r}$ is the gain obtained from admitting the call rather than rejecting it and is given by $\zeta_{j, \varrho} + \tilde{h}_q(\mathbf{x}') - \tilde{h}_q(\mathbf{x})$ where $\mathbf{x}'$ is the network state after admitting the call and $\varrho = N, \text{HO}$ [19]. From the quadratic form of the feature vector $\phi(\mathbf{x})$ in (14), the following net-gain result can be obtained in terms of the link net gains [11]:

$$\zeta_{j, \varrho} + \tilde{h}_q(\mathbf{x}') - \tilde{h}_q(\mathbf{x}) = \zeta_{j, \varrho} - \sum_{\forall s \in \mathbf{r}} \tilde{p}_j^s(\mathbf{x}, \pi) \quad (15)$$

where $\tilde{p}_j^s(\mathbf{x}, \pi)$ is the approximated link s shadow price (or implied cost) and $\zeta_{j, \varrho} \in \mathcal{R}^+$ is the immediate reward received for admitting a new/handover call ($\varrho = N, \text{HO}$) of class- j .

The greedy policy (12) can be reinterpreted in terms of link shadow price as follows (the greedy policy for handover calls is determined in a similar manner). A class- j new call is rejected if $\zeta_{j, N} < \sum_{\forall s \in \mathbf{r}} \tilde{p}_j^s(\mathbf{x}, \pi)$; otherwise it is admitted to route $\mathbf{r}^*$ which gives the highest positive net gain

$$\mathbf{r}^* = \arg \max_{\mathbf{r} \in W_j} \left\{ \zeta_{j, N} - \sum_{\forall s \in \mathbf{r}} \tilde{p}_j^s(\mathbf{x}, \pi) \right\}. \quad (16)$$

Note that the notion of link shadow price obtained here—which reflects the state-dependent cost of admitting a call in addition to the reward earned by admitting it—differs in two respects from [19]: 1) handover calls also constitute to the link shadow price whereby the cost of admitting handover calls includes the blocking of future (high reward) new calls and 2) the link shadow price in this paper is deduced from a specific choice of approximation architecture (14) whereas in [19], the link shadow price is derived from explicit network decomposition into independent link processes. Finally, the OPI method decomposes the network into the link level, and to improve the learning rate, we distributed the learning process at the link level as in [11]. The training procedure of this algorithm is given in Appendix A.

C. Exploration Versus Exploitation

To achieve decisions that produce high rewards, the learning schemes must favor or exploit good actions that have been found to produce high rewards in the past. However, the learning schemes must also explore alternative actions to discover better actions in the future. Therefore, the learning schemes progress by trying out a variety of actions and gradually favoring actions that appear to be best. In this paper, the ACSMDP algorithm inherits an on-policy behavior whereby it exploits the policy it has learned so far, and explores other actions

Fig. 2. Schematic diagram of the test network.

probabilistically under the same (randomized) policy. The OPI algorithm, on the other hand, can employ a ϵ -greedy exploration where the best action is chosen with high probability (ϵ), and other nongreedy actions are selected with small probability ($1 - \epsilon$). It is worth noting that the study of the suitable amount of interplay between the exploitation and exploration in RL is an area which warrants future investigation [36].

IV. NUMERICAL RESULTS

In this section, we first compare the performance, in terms of average revenue (8) with the DP-based SMDP algorithm proposed in [19] and the MAXMIN routing method employed in [3] which aims at maximizing the minimum residual capacity in the network. These methods were selected because they also consider rerouted traffic of ongoing calls. We then test the robustness of the proposed methods in the presence of varying incoming traffic demand that emulates the stochastic nature of the traffic as the satellites sweeps across different regions on Earth. Finally, an assessment of the computational time and storage space requirement for all methods is carried out.

A. General Settings

We consider a 36-node LEO satellite network illustrated in Fig. 2. In this test network, there are five different topologies—topologies TP₁–TP₄ have certain links turned off and TP₅ has all links are turned on. The number of paths per topologies is 3792, 3312, 3312, 3792, and 4332, respectively. The visibility

TABLE II
MULTISERVICE PARAMETERS: CASES 1–3

	Class 1	Class 2	Class 3
Case 1, Mean arr. rate, $\lambda_{j,N}$	63	63	63
Case 2, Mean arr. rate, $\lambda_{j,N}$	63	126	63
Case 3, Mean arr. rate, $\lambda_{j,N}$	126	31.5	63
Bandwidth requirement, b_j	1	1	2
Mean call holding time, $1/\mu_j$	5.0	20.0	2.0
Reward for new calls, $\zeta_{j,N}$	1.0	8.0	2.0
Reward for HO calls, $\zeta_{j,HO}$	50	50	50

TABLE III
MULTISERVICE PARAMETERS: CASES 4–6

	Class 1	Class 2	Class 3
Mean arr. rate, $\lambda_{j,N}$	94.5	63	31.5
Case 4, Bandwidth requirement, b_j	1	2	2
Case 5, Bandwidth requirement, b_j	1	2	3
Case 6, Bandwidth requirement, b_j	1	2	4
Mean call holding time, $1/\mu_j$	5.0	20.0	2.0
Reward for new calls, $\zeta_{j,N}$	1.0	8.0	2.0
Reward for HO calls, $\zeta_{j,HO}$	50	50	50

TABLE IV
MULTISERVICE PARAMETERS: CASES 7–9

	Class 1	Class 2	Class 3
Mean arr. rate, $\lambda_{j,N}$	100.8	50.4	25.2
Bandwidth requirement, b_j	1	1	1
Mean call holding time, $1/\mu_j$	2.0	5.0	20.0
Case 7, Reward for new calls, $\zeta_{j,N}$	2.0	5.0	10.0
Case 8, Reward for new calls, $\zeta_{j,N}$	4.0	8.0	12.0
Case 9, Reward for new calls, $\zeta_{j,N}$	0.5	1.25	2.5
Reward for HO calls, $\zeta_{j,HO}$	50	50	50

time is 10 min, and the target probability for stable topology is 0.9 [19]. The occurrence of changing topology is cyclic, i.e., in the sequence of TP₁, TP₅, TP₂, TP₅, ..., TP₄, TP₅, TP₁ ... and so on. The first and the last node of each orbit are interconnected as illustrated by the thick dotted lines. In the following study, we assume that call blocking or dropping can occur only due to insufficient bandwidth in the ISL segment of the network.

New call requests from the traffic sources in the service areas arrive into the system according to a Poisson process (in calls per minute). The duration of the calls in each service (in minutes) is governed by an exponential distribution and any blocked calls are assumed lost. In order to highlight the selective rejection ability of the proposed RL methods, nine cases were investigated. In cases 1–3 (Table II), we vary the mean call arrival rate; in cases 4–6 (Table III), we modify the bandwidth requirement, and in cases 7–9 (Table IV), we vary the reward structure (1). In all reported tests $\zeta_j^N < \zeta_j^{HO}$ which implies that the decision gives higher priority to handover calls than new calls. Further tests can be found in [21]. After having completed the training process, the parameters obtained from the OPI and ACSMDP methods were evaluated in a

Fig. 3. Percentage of calls routed on minimum hop paths (OPI, ACSMDP, MINMAX).

Fig. 4. Service types blocking probability ($C = 10$ units, $C = 30$ units, and $C = 50$ units).

simulation run of 2×10^6 time steps. Using this runlength, the measured results are within an 98% interval of confidence. For comparison, the MAXMIN and SMDP methods were also evaluated under the same runlength.

B. Net Revenue

Figs. 3 and 4 illustrate the routing behavior and blocking probabilities for the network with link capacity of $C = 10$ units, $C = 30$ units, and $C = 50$ units. For convenience, the new and handover of each service type are referred to as 1N, 1H, 2N, 2H, 3N, and 3H.

Fig. 3 depicts the percentage of calls routed on a minimum hop path. The OPI method shows that less 1N, 2N, and 3N calls are routed on minimum hop paths than 1H, 2H, and 3H meaning that more 1N, 2N, and 3N calls are routed on longer alternative paths. This allows more resources to accommodate 1N, 2N, and 3N calls and permit 1H, 2H, and 3H calls to find alternative shortest path easier. The ACSMDP method has a tendency to route traffic uniformly over the set of available routes. Furthermore, minimum hop paths are used in the least proportion over other routing methods. This can be explained by the usage of (9) as state representation in the actor's feature. Such state representation can only distinguish the network state

TABLE V
AVERAGE NET REVENUE (8) OBTAINED FROM THE SIMULATION STUDY UNDER DIFFERENT TRAFFIC SCENARIOS

	10 units link capacity				30 units link capacity			50 units link capacity		
	SMDP	OPI	ACSMDP	MAXMIN	OPI	ACSMDP	MAXMIN	OPI	ACSMDP	MAXMIN
Case 1-3	160.61	145.80	114.00	98.70	362.45	306.42	247.78	485.73	446.18	362.26
Case 4-6	173.98	133.58	75.68	64.40	376.27	215.82	177.36	530.18	368.05	270.64
Case 7-9	281.86	235.95	120.45	102.37	466.71	389.94	359.60	584.43	526.28	488.90
Average	205.48	171.78	103.38	88.49	401.81	304.06	261.58	533.44	446.84	373.94

in an highly aggregated manner (unlike the OPI approach which can discriminate individual link states). As a result, the choices of routes appear similar to the actor, and a uniform path selection behavior as shown in Fig. 3 is obtained. Despite the higher blocking probabilities, the MAXMIN tends to consistently use minimum hop paths for all accepted connections. Such uniform routing behavior (similar to the ACSMDP method) is expected from this method since routing decisions are made irrespective of the call rewards.

Fig. 4 shows that the OPI method performs selective rejection on 1N calls to accommodate higher revenue calls. That is, the OPI method rejects more 1N calls than 2N and 3N calls despite $1/\mu_1 < 1/\mu_2$ and $b_1 \leq b_3$. Furthermore, the blocking probabilities at 1H and 3H are very low. The ACSMDP method is also able to selectively reject lower reward calls in accordance to the randomized policy (6). Unlike the other two methods, MAXMIN does not consider the reward for each service type. Instead, it seeks to balance the traffic to the route with maximal minimum residual bandwidth, and as a result, calls of any type are accepted as long as the capacity constraint is still met. Call blocking occurs in this method mainly due to insufficient capacity. Hence, calls with less bandwidth requirement (1N, 1H, 2N, 2H) experience lower call-blocking probability than calls with higher bandwidth requirement (3N, 3H), regardless of their rewards. The result is that more calls with low bandwidth requirement and low reward are accepted than calls with high bandwidth requirement and high reward.

Table V reports the comparison of net revenue (8) of the OPI and ACSMDP method with the DP-based SMDP and MAXMIN methods in a multiple service network with $C = 10$ units, $C = 30$ units, and $C = 50$ units of link capacity in each link. Note that for $C > 20$ units, the DP-based SMDP approach becomes too computationally exhaustive to implement (see Section IV-D). From the result reported here, it can be seen that ACSMDP revenue outcome on average is 30.6%–7.1% higher than MAXMIN. We also note that as the system grows (from $C = 10$ units to $C = 50$ units), the ACSMDP method and OPI method tend to agree more in respect to their achievable average revenue outcome aggregates, and both of them outperform MAXMIN. These results suggest that the proposed RL methods can perform nearly as good as the DP-based SMDP approach and achieve up to 56.6% higher average revenue over existing routing algorithms. These and further results obtained in [21] would indicate that RL methods could become even more relevant as the system dimensions grow, and that RL-based solutions are a viable alternative when DP-based SMDP solutions become too computational expensive and/or the systems under analysis is large.

Fig. 5. Robustness to different degrees of traffic variation. Performance relative to the MAXMIN approach.

C. Robustness to Traffic Variation

So far, the parameters in the OPI and ACSMDP methods have been trained and evaluated under a stationary traffic demand environment. However, as the satellite sweeps across different regions on Earth, stochastic incoming traffic patterns are expected. In this test, we use the associated parameters already trained (see Section IV-A). To examine robustness the traffic variation is modeled by multiplying the arrival rate by a factor Δ which is uniformly selected from the range $[1 - \xi_V, 1 + \xi_V]$. The parameter ξ_V accounts for the traffic variation, and it is varied from 0 to 1 with small increments (say, $\xi_V \sim 0.2$) indicating little variation where as $\xi_V = 1$ amounts to a total traffic variation from 0–126 calls/min.

Fig. 5 illustrates the relative net revenue with respect to the MAXMIN method. The relative net revenue with respect to the MAXMIN method is defined by $[\text{Net_revenue_other} / \text{Net_revenue_MAXMIN}]$ and is obtained for the multiple service network with $C = 10$ units, $C = 30$ units, and $C = 50$ units link capacity. From the figure, it is found that the OPI, ACSMDP, and DP-based SMDP methods are robust against random traffic variations: their relative net revenue with respect to the MAXMIN method differs only slightly as the parameter ξ_V is increased. These results suggest that for the test here performed, the parameters trained off-line under a stationary traffic scenario, can be used over a wide terrestrial region, with the associated reduction in the number of parameters stored onboard the satellite.

Fig. 6. Computational time as a function of link capacity (Logarithmic scale).

D. Computational Time and Storage Requirements

Here, we examine the computational time of the MAXMIN, SMDP, ACSMDP, and OPI methods as a function of link capacity.² The computational time refers to the time required to complete the simulation run of 2×10^6 time steps (events). The capacity of all the network links are identical for each case of link capacity. The obtained results are shown in Fig. 6 and are given as an average of all cases. Note that the SMDP method is only evaluated up to $C = 20$ units link capacity—beyond which results cannot be obtained within a reasonable amount of time (say, two days). From this figure, the simulation with online decisions show that the computational time of the SMDP method is over one order of magnitude greater than the MAXMIN, ACSMDP, and OPI methods at $C = 10$ units link capacity, and over two orders of magnitude at $C = 20$ units link capacity. The gains in computational time are due to the off-line training procedures of the decision parameters and compact representation—enabling less computation for online decision making.

The last experiment focuses on the online decision-making computational time and examines the computational time as a function of the number of nodes in the network. The link capacity is $C = 30$ units throughout the network. Five different sizes of networks are studied comprising of 4, 9, 16, 25, and 36 nodes, respectively. Each network undergoes topology changes according to Fig. 2. The resultant computational time is shown in Fig. 7. The OPI method shows the highest computational time followed by the ACSMDP and MAXMIN methods, respectively. In terms of the number of iterations, the amount of computation needed to compute one online decision is $O(Kn_r n_l)$ for the case of the OPI method, $O((n_r + 1)n_l)$ for the ACSMDP method and $O(n_r n_l)$ for the MAXMIN method, where K is the number of features of the tunable vector

Fig. 7. Computational time as a function of the number of nodes on the network. The results shown are for link capacity $C = 30$ units.

$\mathbf{q} \in \mathcal{R}^K$, n_r is the maximum number of alternative routes for one class, and n_l is the number of links in the longest route.

In respect to the storage requirements, for the OPI method the storage requirement in one satellite node for online decision making is $\sum_{\forall n} K|\mathcal{L}|$ where $|\mathcal{L}|$ is the number of links in topology TP_n and K is the dimension of the link parameter vector $\mathbf{q} \in \mathcal{R}^K$ such that $K \ll |X_s|$, $|X_s|$ is the size of the link state space. The storage requirement for online decision making in the ACSMDP algorithm is $\sum_{\forall n} M_n$ parameters where M_n is the dimension of parameter vector $\theta \in \mathcal{R}^{M_n}$ such that $M_n = \mathcal{K} + \sum_{\forall j \in \mathcal{K}} n_j$, $\mathcal{K}$ is the number of classes, and n_j is the number of routes for class- j call in topology TP_n . We note that the ACSMDP method will generally demand less onboard storage for its decision parameters than the OPI method. This is because M_n is dependent on the number of satellite nodes, which is significantly fewer than the number of links in the network.

V. FINAL REMARKS

In this paper, we develop and assess two RL approaches to solve an SMDP-based call admission and routing algorithm for LEO satellite networks. The focus is placed on employing RL approaches to learn a nearly optimal decision policy to circumvent the computational burden faced when computing an exact solution by means of conventional DP techniques as proposed in [19]. This leads to two distinct RL-based call admission and routing methods which are significantly less demanding in terms of computational time, computational and storage requirements than the DP-based SMDP approach. Numerical comparison between the RL-based methods and the MAXMIN routing method [3] are also carried out. In respect to the two developed RL-based algorithms, as the systems' capacity increases the outcome of the ACSMDP algorithm tend to agree more with the average revenue obtained by the OPI algorithm. In terms of computational time, the ACSMDP is faster than the OPI, and ACSMDP would generally demand less onboard storage than the OPI method. Finally, the OPI method

²The simulation code is compiled in C and is run on Fujitsu Life Book with Intel Pentium M processor 1.20-GHz 512-MB RAM.

may have convergence problems and hence would rely much on off-line training and trial-and-error initial parameter tuning. For the ACSMDP approach, a proof of convergence has been derived in [21]. All the evidence gathered so far suggests that RL-based methods solutions become more relevant as the system dimension grows and hence are worth further investigation in aspects like, for example, the state representation of actor feature.

The actual implementation of the proposed algorithm requires the gathering and storing of link state information at each ISL in the network. This can be performed as follows: 1) Each node sends link state information updates of its outgoing links to all other nodes in the network either periodically or when a significant change on its outgoing ISL is detected (see, e.g., [4]). 2) Each node stores the tuned parameters for every link and every topology in the network. If the current topology is TP_n , each node uploads the parameters associated with the current topology. 3) Upon a call request of class- j at a source node for a destination node, the source node performs the computation involved in admitting and assigning the call to one of the available paths using the information from 1) and 2). Parameter training can be performed within a reasonable time thus allowing training in a timely manner in case of any link or satellite node failures. During the training time, a simpler contingency routing method can be invoked instead. Finally, we have assumed that the link state information is always up-to-date and completely observable to every node in the network. However, in practice, link state updates are not always received simultaneously—e.g., there would be time delay before each update propagates throughout the network. We are currently extending the proposed RL algorithms to cope with large scale networks and network scenarios where the link states (the number of occupied resources) and link availability (link existence) are subject to unexpected changes.

APPENDIX A OPI ALGORITHM

The OPI algorithm is used to train the parametric vector $\mathbf{u} \in \mathcal{R}^K$. In this algorithm, γ_k, η_k are small stepsize parameters and that $\nabla_{\mathbf{u}} \tilde{h}_{\mathbf{u}}(\mathbf{x}) = \phi(\mathbf{x})$ since $\tilde{h}_{\mathbf{u}}(\mathbf{x}) = \mathbf{u}^T \phi(\mathbf{x})$. Note also that the parametric vector $\mathbf{u}$ is trained separately for each topology.

Algorithm OPI. *OPI based on TD(0). Arbitrarily initialize $\mathbf{r}_0 \in \mathcal{R}^K, \mathbf{x}_0 \in \mathbf{S}$, and $\tilde{v}_0$.*

- 1) **for** $k = 1$ **to** T **do**
- 2) At t_k , an event ω_k is generated at state $\mathbf{x}_k$.
- 3) $\tau_k = t_k - t_{k-1}$
- 4) Perform updates:
 - a) $d_k = g(\mathbf{x}_{k-1}, \omega_{k-1}, a_{k-1}) - \tilde{v}_{k-1}\tau_k + \tilde{h}_{\mathbf{u}_{k-1}}(\mathbf{x}_k) - \tilde{h}_{\mathbf{u}_{k-1}}(\mathbf{x}_{k-1})$.
 - b) $\mathbf{u}_k = \mathbf{u}_{k-1} + \gamma_k d_k \nabla_{\mathbf{u}} \tilde{h}_{\mathbf{u}_{k-1}}(\mathbf{x}_{k-1})$.
 - c) $\tilde{v}_k = \tilde{v}_{k-1} + \eta_k (g(\mathbf{x}_{k-1}, \omega_{k-1}, a_{k-1}) - \tilde{v}_{k-1}\tau_k)$.
- 5) Select action
 - a) $a_k = \arg \max_{a \in A(\mathbf{x}_k, \omega_k)} \{g(\mathbf{x}_k, \omega_k, a) + \tilde{h}_{\mathbf{u}_k}(\mathbf{x}')\}$.
- 6) Get reward $g(\mathbf{x}_k, \omega_k, a_k)$ and network changes to next state $\mathbf{x}_{k+1}$.
- 7) **end for** k ■

APPENDIX B ACSM DP ALGORITHM

The ACSMDP algorithm is used to train the parametric vectors $\mathbf{u} \in \mathcal{R}^{M+1}$ and $\theta \in \mathcal{R}^M$. In this algorithm, γ_k, η_k are small stepsize parameters and $\tilde{Q}_{\mathbf{u}}^{\theta}(\mathbf{x}, \omega, a) = \mathbf{u}^T \phi^{\theta}(\mathbf{x}, \omega, a)$, $\nabla_{\mathbf{u}} \tilde{Q}_{\mathbf{u}}^{\theta}(\mathbf{x}, \omega, a) = \phi^{\theta}(\mathbf{x}, \omega, a)$. Note also that the parametric vectors $\mathbf{u}, \theta$ are trained separately for each node and topology. The convergence results of the algorithm is provided in [21].

Algorithm ACSMDP. *Konda's actor-critic algorithm for SMDP with TD(λ) critic. Arbitrarily initialize $\mathbf{u}_0, \mathbf{z}_0 \in \mathcal{R}^{M+1}, \theta_0 \in \mathcal{R}^M, \mathbf{x}_0 \in \mathbf{S}$, and $\tilde{v}_0$.*

- 1) **for** $k = 1$ **to** T **do**
- 2) At t_k , an event ω_k is generated at state $\mathbf{x}_k$.
- 3) $\tau_k = t_k - t_{k-1}$
- 4) Generate $a_k \in A(\mathbf{x}, \omega)$ from $\mu_{\theta_{k-1}}$.
- 5) Get reward $g(\mathbf{x}_k, \omega_k, a_k)$ and network changes to next state $\mathbf{x}_{k+1}$.
- 6) Perform updates:
 - a) $d_k = g(\mathbf{x}_{k-1}, \omega_{k-1}, a_{k-1}) - \tilde{v}_{k-1}\tau_k + \tilde{Q}_{\mathbf{u}_{k-1}}^{\theta_{k-1}}(\mathbf{x}_k, \omega_k, a_k) - \tilde{Q}_{\mathbf{u}_{k-1}}^{\theta_{k-1}}(\mathbf{x}_{k-1}, \omega_{k-1}, a_{k-1})$.
 - b) $\mathbf{z}_k = \lambda \mathbf{z}_{k-1} + \nabla_{\mathbf{u}} \tilde{Q}_{\mathbf{u}_{k-1}}^{\theta_{k-1}}(\mathbf{x}_k, \omega_k, a_k)$.
 - c) $\mathbf{u}_k = \mathbf{u}_{k-1} + \gamma_k d_k \mathbf{z}_k$.
 - d) $\tilde{v}_k = \tilde{v}_{k-1} + \eta_k (g(\mathbf{x}_k, \omega_k, a_k) - \tilde{v}_{k-1}\tau_k)$.
 - e) $\theta_k = \theta_{k-1} + \beta_k \Gamma(\mathbf{r}_{k-1}) \tilde{Q}_{\mathbf{u}_{k-1}}^{\theta_{k-1}}(\mathbf{x}_k, \omega_k, a_k) \psi^{\theta_{k-1}}(\mathbf{x}_k, \omega_k, a_k)$.
- 7) **end for** k ■

The updates of $d_k, \mathbf{z}_k, \mathbf{u}_k, \tilde{v}_k$, and θ_k are performed at every epoch of call arrival and departure. Note that for all other nodes, a single decision is taken ($a = \text{accept}$) at which $\psi^{\theta_{k-1}}(\mathbf{x}_k, \omega_k, a_k) = [0, \dots, 0]^T$, and θ_k remains unchanged for these nodes. Similarly, for a call departure event, θ_k remains unchanged.

APPENDIX C GRADIENT ESTIMATION FOR THE ACSMDP

Recall that for each pair (x, a) , the policy $\mu_{\theta}(a|x)$ denotes the probability that action a is taken when the state is x where $\sum_{a \in A(x)} \mu_{\theta}(a|x) = 1$ and $A(x)$ is the set of allowed actions at state x .

Let $q_{xx'}(\tau, a)$ denote the transition distributions of the SMDP

$$q_{xx'}(\tau, a) = \Pr[\tau_k \leq \tau, x_{k+1} = x' | x_k = x, a_k = a]. \quad (17)$$

Note that $q_{xx'}(\tau, a)$ is independent of θ and the dependence of the process $\{x_k, a_k, \tau_k\}$ on θ is only through the execution of policy μ_{θ} . Note also that the relationship between $q_{xx'}(\tau, a)$ and the transition probabilities $p_{xx'}(a)$ from the discrete-time Markov chain follows

$$\lim_{\tau \rightarrow \infty} q_{xx'}(\tau, a) = \Pr[x_{k+1} = x' | x_k = x, a_k = a] = p_{xx'}(a).$$

Let $\bar{\tau}(x, a)$ be the average transition time corresponding to state-action pair (x, a) . Assume that

$$\bar{\tau}(x, a) = \sum_{\forall x' \in X} \int_0^\infty \tau q_{xx'}(d\tau, a) < \infty \quad (18)$$

for all x, x' and a . Let $g : X \times A \rightarrow \mathcal{R}$ be an expected immediate reward function.

Gradient Estimation

Let $h^\theta(x) : X \rightarrow \mathcal{R}$ be a differential reward function defined as

$$h^\theta(x) = \sum_{\forall a \in A(x)} \mu_\theta(a|x) Q^\theta(x, a) \quad (19)$$

$$Q^\theta(x, a) = g(x, a) - v(\theta) \bar{\tau}(x, a) + \sum_{\forall x' \in X} p_{xx'}(a) h^\theta(x'). \quad (20)$$

By differentiating (19) with respect to $\theta(j)$ for $0 \leq j \leq M-1$, the following term is obtained:

$$\begin{aligned} \frac{\partial}{\partial \theta(j)} h^\theta(x) = \sum_{\forall a \in A(x)} \left(\frac{\partial}{\partial \theta(j)} \mu_\theta(a|x) Q^\theta(x, a) \right. \\ \left. + \mu_\theta(a|x) \frac{\partial}{\partial \theta(j)} Q^\theta(x, a) \right). \quad (21) \end{aligned}$$

After substituting (20) into the latter term on the right-hand side of (21) and rearranging the terms, we have

$$\begin{aligned} \sum_{\forall a \in A(x)} \mu_\theta(a|x) \frac{\partial}{\partial \theta(j)} v(\theta) \bar{\tau}(x, a) \\ = \sum_{\forall a \in A(x)} \frac{\partial}{\partial \theta(j)} \mu_\theta(a|x) Q^\theta(x, a) + \sum_{\forall a \in A(x)} \mu_\theta(a|x) \\ \times \frac{\partial}{\partial \theta(j)} g(x, a) + \sum_{\forall a \in A(x)} \mu_\theta(a|x) \sum_{\forall x' \in X} p_{xx'}(a) \\ \times \frac{\partial}{\partial \theta(j)} h^\theta(x') - \frac{\partial}{\partial \theta(j)} h^\theta(x). \quad (22) \end{aligned}$$

The second term on the right-hand side of (22) becomes zero since $g(x, a)$ is, by definition, independent of θ . Therefore, (22) is reduced to

$$\begin{aligned} \sum_{\forall a \in A(x)} \mu_\theta(a|x) \frac{\partial}{\partial \theta(j)} v(\theta) \bar{\tau}(x, a) = \sum_{\forall a \in A(x)} \frac{\partial}{\partial \theta(j)} \mu_\theta(a|x) \\ \times Q^\theta(x, a) + \sum_{\forall x' \in X} p_{xx'}^\theta \frac{\partial}{\partial \theta(j)} h^\theta(x') - \frac{\partial}{\partial \theta(j)} h^\theta(x). \quad (23) \end{aligned}$$

Summing over the steady-state probabilities $p_\theta(x)$ on both sides of (23), we then have

$$\begin{aligned} \frac{\partial}{\partial \theta(j)} v(\theta) \sum_{\forall x \in X} p_\theta(x) \sum_{\forall a \in A(x)} \mu_\theta(a|x) \bar{\tau}(x, a) \\ = \sum_{\forall x \in X} p_\theta(x) \sum_{\forall a \in A(x)} \frac{\partial}{\partial \theta(j)} \mu_\theta(a|x) Q^\theta(x, a). \end{aligned}$$

Thus, the gradient of the average reward for the SMDP with respect to $\theta(j)$ for $0 \leq j \leq M-1$ is given by

$$\frac{\partial}{\partial \theta(j)} v(\theta) = \frac{\sum_{\forall x \in X} \sum_{\forall a \in A(x)} p_\theta(x) \mu_\theta(a|x) \psi_j^\theta(x, a) Q^\theta(x, a)}{\sum_{\forall x \in X} p_\theta(x) \sum_{\forall a \in A(x)} \mu_\theta(a|x) \bar{\tau}(x, a)} \quad (24)$$

where

$$\psi_j^\theta(x, a) = \frac{\frac{\partial}{\partial \theta(j)} \mu_\theta(a|x)}{\mu_\theta(a|x)}.$$

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their helpful and constructive comments.

REFERENCES

- [1] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in *Advances in Neural Information Processing Systems 6*. San Mateo, CA: Morgan Kaufmann, 1994.
- [2] J. Carlstrom, "Reinforcement learning for admission control and routing," Ph.D. dissertation, Uppsala Univ., Uppsala, Sweden, 2000.
- [3] O. Ercetin, S. Krishnamurthy, S. Dao, and L. Tassiulas, "Provision of guaranteed services in broadband LEO satellite networks," *Comput. Netw.*, vol. 39, no. 1, pp. 61–77, May 2002.
- [4] L. Franck and G. Maral, "Signalling for inter-satellite link routing in broadband non-GEO satellite systems," *Comput. Netw.*, vol. 39, no. 1, pp. 79–92, May 2002.
- [5] E. Gelenbe, R. Lent, and A. Nunez, "Self-aware networks and QoS," *Proc. IEEE*, vol. 92, no. 9, pp. 1478–1489, Sep. 2004.
- [6] I. Gragopoulos, E. Papapetrou, and F. Pavlidou, "Performance study of adaptive routing algorithms for LEO satellite constellations under self-similar and Poisson traffic," *Space Commun.*, vol. 16, no. 1, pp. 15–22, Jan. 2000.
- [7] T. C.-K. Hui and C.-K. Tham, "Adaptive provisioning of differentiated services networks based on reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 33, no. 4, pp. 492–501, Nov. 2003.
- [8] B. W. Kim, S. L. Min, H. S. Yang, and C. S. Kim, "Predictive call admission control scheme for low Earth orbit satellite networks," *IEEE Trans. Veh. Technol.*, vol. 49, no. 6, pp. 2320–2335, Nov. 2000.
- [9] V. R. Konda, "Actor-critic algorithms," Ph.D. dissertation, MIT, Cambridge, MA, 2002.
- [10] V. R. Konda and J. N. Tsitsiklis, "On actor-critic algorithms," *SIAM J. Control Optim.*, vol. 42, no. 4, pp. 1143–1166, Aug. 2003.
- [11] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis, "Call admission control and routing in integrated services networks using Neuro-dynamic programming," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 2, pp. 197–208, Feb. 2000.
- [12] L. Peshkin and V. Savova, "Reinforcement learning for adaptive routing," in *Proc. Int. Joint Conf. Neural Netw.*, 2002, pp. 1825–1830.
- [13] S. Singh and D. P. Bertsekas, "Reinforcement learning for dynamic channel allocation in cellular telephone systems," in *Advances in Neural Information Processing Systems 10*. Cambridge, MA: MIT Press, 1997, pp. 974–980.
- [14] H. Tong and T. X. Brown, "Adaptive call admission control under quality-of-service constraints: A reinforcement learning solution," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 2, pp. 209–220, Feb. 2000.
- [15] N. Tao, J. Baxter, and L. Weaver, "A multi-agent, policy-gradient approach to network routing," in *Proc. 18th Int. Mach. Learn. Conf.*, 2001, pp. 553–560.
- [16] G. Tesauro, "Practical issues in temporal difference learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 257–277, May 1992.
- [17] P. Tam, J. Lui, H. W. Chan, C. Sze, and C. N. Sze, "An optimized routing scheme and a channel reservation strategy for a low Earth orbit satellite system," in *Proc. IEEE VTC*, Sep. 1999, vol. 5, pp. 2870–2874.
- [18] H. Uzunalioglu, I. F. Akyildiz, and M. Bender, "A routing algorithm for connection-oriented low Earth orbit (LEO) satellite networks with dynamic connectivity," *Wireless Netw.*, vol. 6, no. 3, pp. 181–190, May 2000.

- [19] W. Usaha and J. Barria, "Markov decision theory framework for resource allocation in LEO satellite constellations," *Proc. Inst. Electr. Eng.—Commun.*, vol. 149, no. 5, pp. 270–276, Oct. 2002.
- [20] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 1995.
- [21] W. Usaha, "Resource allocation in networks with dynamic topology," Ph.D. dissertation, Imperial College London, London, U.K., 2004.
- [22] H. Uzunalioglu, "Probabilistic routing protocol for low Earth orbit satellite networks," in *Proc. ICC*, Jun. 1998, vol. 1, pp. 89–93.
- [23] M. Werner, C. Delluchi, H. Vogel, G. Maral, and J. De Ridder, "ATM-based routing in LEO/MEO satellite networks with intersatellite links," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 1, pp. 69–82, Jan. 1997.
- [24] M. Werner, "A dynamic routing concept for ATM-based satellite personal communication network," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 8, pp. 1636–1648, Oct. 1997.
- [25] M. Werner, C. Mayer, G. Maral, and M. Holzbock, "A neural network base approach to distributed adaptive routing of LEO intersatellite link traffic," in *Proc. IEEE VTC*, May 1998, vol. 2, pp. 1498–1502.
- [26] W. Zhang and T. G. Dietterich, "High performance job-shop scheduling with a time delay TD(λ) network," in *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press, 1996, pp. 1024–1030.
- [27] J. Abounadi, D. P. Bertsekas, and V. S. Borkar, "Learning algorithms for Markov decision processes," Lab. Inf. Decision Syst., MIT, Cambridge, MA, Rep. LIDS-P-2434, 1998.
- [28] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 5, pp. 834–846, Sep./Oct. 1983.
- [29] J. Baxter and P. L. Bartlett, "Infinite-horizon policy gradient estimation," *J. Artif. Intell. Res.*, vol. 15, no. 4, pp. 319–350, Jul.–Dec. 2001.
- [30] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.
- [31] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," in *Advances in Neural Information Processing Systems 14*. Vancouver, BC, Canada: MIT Press, 2001.
- [32] T. Jaakkola, S. P. Singh, and M. I. Jordan, "Reinforcement learning algorithm for partially observable Markov decision problems," in *Advances in Neural Information Processing Systems 7*. San Mateo, CA: Morgan Kaufmann, 1995, pp. 345–352.
- [33] V. R. Konda and V. S. Borkar, "Actor-critic like learning algorithms for Markov decision processes," *SIAM J. Control Optim.*, vol. 38, no. 1, pp. 94–123, 1999.
- [34] S. Mahadevan, "Average reward reinforcement learning: Foundations, algorithms and empirical results," *Mach. Learn.*, vol. 22, no. 1–3, pp. 159–196, 1996.
- [35] P. Marbach and J. N. Tsitsiklis, "Simulation-based optimization of Markov reward processes," *IEEE Trans. Autom. Control*, vol. 46, no. 2, pp. 191–209, Feb. 2001.
- [36] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.
- [37] J. N. Tsitsiklis and B. Van Roy, "An analysis of temporal-difference learning with function approximation," *IEEE Trans. Autom. Control*, vol. 42, no. 5, pp. 674–690, May 1997.
- [38] —, "Average cost temporal-difference learning," *Automatica*, vol. 35, no. 11, pp. 1799–1808, Nov. 1999.
- [39] L. Weaver and N. Tao, "The optimal reward baseline for gradient based reinforcement learning," in *Proc. 17th Conf. Uncertain. Artif. Intell.*, Seattle, WA, Aug. 2001, pp. 538–545.
- [40] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 229–256, May 1992.
- [41] W. Usaha and J. Barria, "A reinforcement learning ticket-based probing path discovery scheme for MANETs," *Ad Hoc Netw.*, vol. 2, no. 3, pp. 319–334, 2004.

Wipawee Usaha (M'04) received the B.Eng. degree from the Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani, Thailand, in 1996, and the M.Sc. and Ph.D. degrees from Imperial College London, University of London, London, U.K., in 1998 and 2004, respectively, all in electrical engineering.

Currently, she is with the School of Telecommunication Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand. Her research interests include distributed resource allocation and

QoS provision in MANETs, satellite, cellular systems, and network monitoring using reinforcement-learning techniques.

Javier A. Barria (M'02) received the B.Sc. degree in electronic engineering from the University of Chile, Santiago, in 1980, and the Ph.D. and M.B.A. degrees from Imperial College London, London, U.K., in 1992 and 1998, respectively.

From 1981 to 1993, he was a System Engineer and Project Manager (network operations) with the Chilean Telecommunications Company. Currently, he is a Reader in the Intelligent Systems and Networks Group, Department of Electrical and Electronic Engineering, Imperial College London. His

research interests include communication networks monitoring strategies using signal and image processing techniques, distributed resource allocation in dynamic topology networks, and fair and efficient resource allocation in IP environments. He has been a joint holder of several FP5 and FP6 European Union project contracts all concerned with aspects of communication systems design and management.

Dr. Barria was a British Telecom Research Fellow from 2001 to 2002.