Polynomial Cost Approximations in Markov Decision Theory Based Call Admission Control

Hannu Rummukainen and Jorma Virtamo, *Member, IEEE*

***Abstract*—The problem of call admission control and routing in a multiservice circuit-switched loss network can be solved optimally under certain assumptions by the tools of Markov decision theory. However, in networks of practical size a number of simplifying approximations are needed to make the solution feasible. Assuming link independence, we propose a new method for approximating the state-dependent link costs accurately and relatively efficiently, even on links with extremely large state spaces. The proposed polynomial approximations are optimal in the sense of minimizing the residual in the continuous-time Howard equations of the Markov decision processes associated with the links. Numerical results are presented, and the proposed approximations are found superior to some earlier link-cost approximation methods.**

***Index Terms*—Broadband networks, connection admission control, Markov decision processes, network revenue, piecewise polynomial approximation, telecommunication congestion control, telecommunication network routing.**

I. Introduction

THE PROBLEM of routing and call admission control in multiservice loss networks has been widely studied; see, e.g., [1]. By assuming Poisson call arrival processes and exponential call holding times, a multiservice circuit-switched loss network can be treated as a continuous-time Markov decision process. In this context, a number of related approaches [2]–[10] have been proposed for finding a call admission and routing policy that maximizes the rate of revenue from the network, or equivalently, minimizes the rate of accumulating costs. In all of these approaches, the decision on which route to accept a new call is made by a condition equivalent to comparing the expected costs of accepting the call on different routes, or in other words, the expected revenue loss caused by additional blocked calls if some bandwidth on a particular route is allocated for a new connection. The new call is carried on the route where the expected costs are the smallest, unless the costs exceed the expected revenue of carrying the call, in which case the call is rejected. Unfortunately, it is practically impossible to compute the exact costs in the complete network Markov process, because the number of states in a nontrivial network is astronomical.

To simplify the problem, it is commonly assumed that the state processes of different links in the network are statistically independent, so that the expected costs of accepting an arriving call on a route can be computed by summing costs from individual links on the route. However, when there are more than a few different traffic classes, the number of states even in a single-link model can easily be prohibitive, calling for further approximations in computing the link costs. In the link-cost approximation proposed by Hwang, Kurose, and Towsley [11], as well as in the one by Krishnan and Hübner [5], all traffic classes are effectively coalesced into one, reducing the link-state space into one dimension. Dziong *et al.* [12] compute the link costs of each traffic class from a separate one-dimensional process. Another decomposition into separate processes is given by Liao *et al.* [13] for link blocking evaluation, and this too can be extended for approximating link costs. Because of the heavy model simplifications all these link-cost approximations depend on the exact link state in a very limited manner, and are thus unable to take into account much of the interactions between the different traffic classes. In this paper, we propose a method for computing polynomial approximations for the link costs without simplifying the link model, thus allowing for more accurate link-cost approximations.

Polynomial cost approximations have been proposed before by Schweitzer and Seidmann [14] for the problem of controlling a queueing network. However, the amount of work required by their computational methods is at best proportional to the number of system states, making the methods too expensive to be applicable on large multiservice links. In the context of a multiservice loss network, Marbach *et al.* [15] apply reinforcement learning to estimate the optimal second-degree polynomial link-cost approximation; this requires lengthy simulation runs. The contribution of this paper is to present considerably more efficient computational methods for determining the coefficients of polynomial cost representations.

The rest of the paper is organized as follows. In Section II, we briefly introduce the Markovian single-link model considered. In Section III, we specify the class of polynomial cost approximations to be discussed, and define the linear least-squares problem from which the polynomial coefficients are determined. In Section IV, we study the structure of the normal equations of the least-squares problem, obtaining expressions that are polynomial in link-state space coordinates. In Section V, we develop recursion formulas for computing certain sums of monomials over subsets of the link-state space; these recursion formulas are integral to efficient computation of the link-cost approximations.

In Section VI, we sketch an algorithm for constructing the normal equations and determining the approximations efficiently, and discuss the choice of the approximation basis from the point of view of the time complexity of the algorithm. In

Manuscript received May 1, 2000; revised November 26, 2000; recommended by IEEE/ACM TRANSACTIONS ON NETWORKING Editor G. de Veciana.

H. Rummukainen is with VTT Information Technology, Espoo, Finland.

J. Virtamo is with the Networking Laboratory, Helsinki University of Technology, 02150 Espoo, Finland (e-mail: Hannu.Rummukainen@iki.fi).

Publisher Item Identifier S 1063-6692(01)10549-2.

Section VII, the method is extended for a class of link-control policies that do not need to have product form stationary probabilities. In Section VIII, we discuss results of numerical evaluation of the proposed approximations on a few link models, comparing to three earlier approximation methods. Finally, in Section IX, we summarize our findings and identify areas for future research.

II. Link Model

The independent Markov model of a single link is defined as follows. The link carries connections of K different traffic classes; calls of traffic class k arrive at rate λ_k and have an expected holding time $1/\mu_k$. The call arrival processes of the traffic classes are assumed to be independent Poisson processes, and the call holding times are assumed to be independently exponentially distributed. The expected revenue from a carried class k call is w_k; the control policy optimization criterion is based on these values. Link capacity is divided into C fixed-size bandwidth units called trunks, and all connections of traffic class k take up b_k trunks on the link. The link state is characterized by a vector $\mathbf{i} \in \mathbb{N}^K$, where element i_k indicates the number of connections of traffic class k in progress; note that we include 0 in the set of natural numbers $\mathbb{N}$. Denoting by $\mathbf{b}$ the vector of traffic class trunk requirements, the complete link-state space is defined as

$$\Omega = \left\{ \mathbf{i} \in \mathbb{N}^K \;\middle|\; \mathbf{i}^{\mathrm{T}}\mathbf{b} \leq C \right\} \tag{1}$$

where the constraint ensures that no more than C trunks can be occupied simultaneously. For convenience, we denote by $\Omega(c)$ the subset of states where exactly c trunks are occupied:

$$\Omega(c) = \left\{ \mathbf{i} \in \Omega \;\middle|\; \mathbf{i}^{\mathrm{T}}\mathbf{b} = c \right\}, \qquad c = 0, \ldots, C. \tag{2}$$

The cardinality of the state space Ω is denoted by N; as an example of the magnitude of N, we remark that for $K = 5$, $C = 100$ and $\mathbf{b} = (1\ 2\ 3\ 4\ 5)^{\mathrm{T}}$, the number of states is already over 10^6. In order to simplify the notation, in the sequel we assume that all traffic class trunk requirements b_k, $k = 1, \ldots, K$, are different from each other; this is not a limitation of the approach.

A link-control policy R assigns each state $\mathbf{i} \in \Omega$ a set $R_{\mathbf{i}} \subseteq \{1, \ldots, K\}$ of traffic classes that are admitted in state $\mathbf{i}$. For the most part we will be concerned only with the complete sharing policy, which accepts a call whenever there is sufficient free capacity, but some other kinds of policies are discussed as well. Under a specific control policy, the link state forms a continuous-time Markov process. When the Markov process is in state $\mathbf{i} \in \Omega$, the system accumulates cost from blocked calls at the expected rate

$$r_{\mathbf{i}} = \sum_{k=1}^{K} 1_{k \notin R_{\mathbf{i}}} w_k \lambda_k \tag{3}$$

where we use the notation 1_{C} for the indicator function that is 1 when condition C is true, and 0 when C is false. The expected long-run *average cost rate* of the system under policy R is denoted by g. We define the *relative values* $v_{\mathbf{i}}$, $\mathbf{i} \in \Omega$, of the system states so that $v_{\mathbf{i}}$ plus some state-independent constant indicates the expected costs accumulated *in addition to* the average cost rate, from the time the system visits state $\mathbf{i}$ onwards, when the system is controlled by policy R. The absolute values of these expected costs are inconsequential in the sequel, and thus we fix the state-independent constant component by arbitrarily setting

$$v_{\mathbf{0}} = 0. \tag{4}$$

Denoting by $\mathbf{e}_k$ the unit vector of all zeros except the kth component one, the N by N infinitesimal generator matrix Q of the link-state process under policy R is defined by

$$q_{\mathbf{ij}} = \begin{cases} \lambda_k & \text{if } \mathbf{j} = \mathbf{i} + \mathbf{e}_k \text{ for some } k \in R_{\mathbf{i}} \\ i_k \mu_k & \text{if } \mathbf{j} = \mathbf{i} - \mathbf{e}_k \text{ for some } k = 1, \ldots, K \\ -\left(\displaystyle\sum_{k=1}^{K} i_k \mu_k + \sum_{k \in R_{\mathbf{i}}} \lambda_k \right) & \text{if } \mathbf{j} = \mathbf{i} \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Denoting by $\mathbf{r}$ the N-vector of state cost rates, and by $\mathbf{v}$ the N-vector of relative values, the average cost rate g and the relative value vector are uniquely determined by the Howard equations [16, Ch. 3]

$$\mathbf{r} - g\mathbf{1} + Q\mathbf{v} = \mathbf{0} \tag{6}$$

in conjunction with condition (4).

In the considered model, the state-dependent link shadow price for traffic class k under policy R is defined as

$$p_k(\mathbf{i}) = v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}, \quad \text{for} \quad k = 1, \ldots, K; \quad \mathbf{i} \in \Omega. \tag{7}$$

The link shadow price measures the expected amount of additional cost incurred from time t onwards, if the system is in state $\mathbf{i} + \mathbf{e}_k$ at time t, as compared to being in state $\mathbf{i}$. In other words, $p_k(\mathbf{i})$ is the cost of accepting a call of traffic class k in state $\mathbf{i}$; it is beneficial to accept the call if the cost is less than the value w_k of the arriving call. As noted in the introduction, by assuming link independence one can compute the expected cost on a multilink route, *the route shadow price*, by adding the link shadow prices from all the links of the route; applying the policy improvement procedure of Markov decision theory [16], an arriving call is rejected if the total expected cost on all the available routes exceeds the expected revenue w_k of the call, and otherwise accepted on the route with the least expected cost. Dziong [7, Ch. 5] discusses the details, as well as issues of practical implementation.

III. Approximation

Let R be the link-control policy for which the link shadow prices are to be approximated. We assume that for $c = 0, \ldots, C$, the decisions $R_{\mathbf{i}}$ of the policy are identical for each $\mathbf{i} \in \Omega(c)$. This assumption is satisfied by the complete sharing policy, as well as by so-called trunk reservation policies that accept a call if the number of free trunks is above a traffic class specific limit.

In order to evaluate the link shadow prices, we seek to express the relative value vector $\mathbf{v}$ of policy R as a linear combination of

a modest number of basis vectors $\mathbf{u}_j$, $j = 1, \ldots, J$, the exact form of which we shall discuss shortly. In matrix form, we have

$$\mathbf{v} = \sum_{j=1}^{J} \alpha_j \mathbf{u}_j = U\boldsymbol{\alpha} \tag{8}$$

where $\boldsymbol{\alpha} = (\alpha_1 \cdots \alpha_J)^{\mathrm{T}}$ are the free coefficients of the basis vectors, and U is a N by J matrix with the vectors $\mathbf{u}_j$, $j = 1, \ldots, J$ as columns. Denoting by $[\mathbf{u}_j]_{\mathbf{i}}$ the element of $\mathbf{u}_j$ corresponding to state $\mathbf{i} \in \Omega$, we require that

$$[\mathbf{u}_j]_0 = 0, \qquad \text{for all } j = 1, \ldots, J \tag{9}$$

so that (4) is always satisfied.

Substituting the parametric relative value representation (8) in the Howard equations (6) yields the overdetermined linear system of N equations in J variables

$$\mathbf{r} - g\mathbf{1} + QU\boldsymbol{\alpha} = \mathbf{0}. \tag{10}$$

We solve the system as a linear least-squares problem, locating the parameters $\boldsymbol{\alpha}$ that minimize the Euclidean norm of the left-hand side of (10). The average cost rate g can be computed accurately by independent means when the policy R is of a special form: for the complete sharing policy we can use the Kaufman–Roberts recursion [17], [18], and for product form policies the convolution algorithm [19], [20], [1] is applicable. If the policy R is of such a form that the average cost rate g is not directly computable, it is possible to treat g as another free parameter in the least-squares problem; we defer the details to Section VII.

As shown in any linear algebra textbook (see, e.g., [21, Sec. 5.3]), the coefficient vector $\boldsymbol{\alpha}$ minimizing the Euclidean norm of the left-hand side of (10) can be determined as the solution of the normal equations

$$U^{\mathrm{T}}Q^{\mathrm{T}}QU\boldsymbol{\alpha} = U^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1} - \mathbf{r}). \tag{11}$$

This is a symmetric linear system of J equations in J variables, so that solving it is feasible as long as the number of coefficients J is sufficiently small. It turns out that for simple monomial forms of the basis vectors $\mathbf{u}_j$, $j = 1, \ldots, J$, the coefficient matrix and the right-hand side vector of the system of normal equations can be constructed efficiently, even when the link-state space is so large that handling the matrices Q and U directly is impracticable.

We discuss the details of normal equations construction for the following basis vectors. First, we consider the family of monomial basis vectors $\mathbf{u}(\boldsymbol{\nu})$, $\boldsymbol{\nu} \in \mathbb{N}^K$, with the vector elements defined by

$$[\mathbf{u}(\boldsymbol{\nu})]_{\mathbf{i}} = \prod_{k=1}^{K} i_k^{\nu_k}, \qquad \text{for } \mathbf{i} \in \Omega \tag{12}$$

where 0^0 is taken as 1 so that $\nu_k = 0$ indicates that the kth factor is always unity; this interpretation holds for all potential occurrences of 0^0 in this paper. For simplicity, we assume that the exponent vector $\boldsymbol{\nu}$ does not contain more than two nonzero elements, thus restricting the discussion to the single-coordinate *monomials* $i_k^{\nu_k}$ and the double-coordinate monomials $i_k^{\nu_k} i_l^{\nu_l}$. Note that (9) makes it unnecessary to consider the case $\boldsymbol{\nu} = \mathbf{0}$ in which all elements of $\mathbf{u}(\mathbf{0})$ are equal to 1.

Second, we consider the piecewise monomial basis vectors $\mathbf{u}(\boldsymbol{\nu}, d)$, $\boldsymbol{\nu} \in \mathbb{N}^K$, $d = 1, \ldots, C$, with the vector elements defined by

$$[\mathbf{u}(\boldsymbol{\nu}, d)]_{\mathbf{i}} = 1_{\mathbf{i}\in\Omega(d)} \prod_{k=1}^{K} i_k^{\nu_k}, \qquad \text{for } \mathbf{i} \in \Omega \tag{13}$$

where again 0^0 is taken as 1. Here, we require $\boldsymbol{\nu}$ to have at most one nonzero element, so that the elements of these vectors are either single-coordinate piecewise monomials $1_{\mathbf{i}\in\Omega(d)} i_k^{\nu_k}$ or piecewise constants $1_{\mathbf{i}\in\Omega(d)}$.

We specify the complete basis in terms of the integer parameters D_1, D_2, E_2, and P_1, where $E_2 \geq D_2$, as comprising the single-coordinate monomial vectors up to degree D_1

$$\mathbf{u}(\alpha\mathbf{e}_m), \qquad \text{for } \alpha = P_1 + 1, \ldots, D_1; \quad m = 1, \ldots, K$$

the double-coordinate monomial vectors up to degree $D_2 + E_2$

$$\mathbf{u}(\alpha\mathbf{e}_m + \beta\mathbf{e}_n), \qquad \text{for } \alpha = 1, \ldots, D_2; \quad \beta = 1, \ldots, E_2; \quad m, n = 1, \ldots, K; \quad m \neq n$$

the piecewise constant vectors

$$\mathbf{u}(\mathbf{0}, d), \qquad \text{for } d = 1, \ldots, C$$

and the piecewise single-coordinate monomial vectors up to degree P_1

$$\mathbf{u}(\alpha\mathbf{e}_m, d), \qquad \text{for } \alpha = 1, \ldots, P_1; \quad m = 1, \ldots, K; \quad d = 1, \ldots, C.$$

The basis is symmetric with respect to the different traffic classes, or in other words if a basis vector of a particular family is included for an exponent vector $\boldsymbol{\nu}$, then a basis vector of the same family is included for every permutation of the exponent vector $\boldsymbol{\nu}$. Note that we do not need ordinary single-coordinate monomials of degrees $1, \ldots, P_1$ because for those degrees the piecewise single-coordinate monomials replace them. Also, the piecewise monomials $\mathbf{u}(\boldsymbol{\nu}, d)$ are included for every link occupation level $d = 1, \ldots, C$, and the piecewise constant vectors are always included in the basis. The total number of different basis vectors is now

$$J = D_1K + D_2(2E_2 - D_2)\,\tfrac{1}{2}\,K(K-1) + P_1KC + C. \tag{14}$$

The association of the indices $j = 1, \ldots, J$ with particular basis vectors can be chosen arbitrarily.

In Section VI, we discuss ways to reduce the total number of basis vectors by combining a number of piecewise monomial vectors of type (13) into a single basis vector as $\sum_{d=d_0}^{d_1} \mathbf{u}(\boldsymbol{\nu}, d)$; in the construction of normal equations, such basis vector combinations share the essential properties of the vectors (13), and thus we omit the details.

It is relatively straightforward to extend the treatment for more general monomial basis vectors $\mathbf{u}(\boldsymbol{\nu})$ and $\mathbf{u}(\boldsymbol{\nu}, d)$ with less restrictions on the exponent vectors $\boldsymbol{\nu} \in \mathbb{N}^K$; however, working with products of more than two state-space coordinates would lead to a considerable increase in the amount of computational work required.

IV. Structure of the Normal Equations

Observe that the columns of the matrix QU are exactly the vectors $Q\mathbf{u}_j$, $j = 1, \ldots, J$, and consequently the elements of the right-hand side vector of the normal equations are given by

$$\begin{aligned}\left[U^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1}-\mathbf{r})\right]_j &= \sum_{\mathbf{i}\in\Omega}[QU]_{\mathbf{i},j}\,[g\mathbf{1}-\mathbf{r}]_{\mathbf{i}} \\ &= \sum_{\mathbf{i}\in\Omega}[Q\mathbf{u}_j]_{\mathbf{i}}\,(g-r_{\mathbf{i}}) \\ &\qquad \text{for } j = 1, \ldots, J \end{aligned} \tag{15}$$

and the elements of the matrix $U^{\mathrm{T}}Q^{\mathrm{T}}QU$ are

$$\begin{aligned}\left[U^{\mathrm{T}}Q^{\mathrm{T}}QU\right]_{hj} &= \mathbf{u}_h^{\mathrm{T}}Q^{\mathrm{T}}Q\mathbf{u}_j \\ &= \sum_{\mathbf{i}\in\Omega}[Q\mathbf{u}_h]_{\mathbf{i}}\,[Q\mathbf{u}_j]_{\mathbf{i}} \\ &\qquad \text{for } h, j = 1, \ldots, J. \end{aligned} \tag{16}$$

In order to further deconstruct the matrix structures, let us take advantage of the sparsity of Q as defined by (5), so as to express an element of $Q\mathbf{u}_j$ as

$$\begin{aligned}[Q\mathbf{u}_j]_{\mathbf{i}} = &\sum_{k\in R_{\mathbf{i}}}\lambda_k\left([\mathbf{u}_j]_{\mathbf{i}+\mathbf{e}_k}-[\mathbf{u}_j]_{\mathbf{i}}\right) \\ &-\sum_{k=1}^{K}\mu_k i_k\left([\mathbf{u}_j]_{\mathbf{i}-\mathbf{e}_k}-[\mathbf{u}_j]_{\mathbf{i}}\right). \end{aligned} \tag{17}$$

When the basis vectors $\mathbf{u}_j$, $j = 1, \ldots, J$ are piecewise monomials in state space coordinates, it follows that the elements (17) of $Q\mathbf{u}_j$ are piecewise polynomials in state space coordinates, and we can derive piecewise polynomial expressions for the elements (15) and (16) of the normal equations. This is the reason for the use of monomial basis vectors specifically.

Let us now derive explicit forms for the elements of the vectors $Q\mathbf{u}(\boldsymbol{\nu})$ and $Q\mathbf{u}(\boldsymbol{\nu}, d)$, where $\boldsymbol{\nu}$ is restricted to the simple forms considered. Of particular interest in the sequel are the numbers of terms in the resulting polynomial expressions. To begin with, for single-coordinate monomial basis vectors $\mathbf{u}(\alpha\mathbf{e}_m)$, $\alpha > 0$, the element differences in (17) can be expanded as

$$\begin{aligned}&[\mathbf{u}(\alpha\mathbf{e}_m)]_{\mathbf{i}\pm\mathbf{e}_k}-[\mathbf{u}(\alpha\mathbf{e}_m)]_{\mathbf{i}} \\ &\quad = (i_m \pm 1_{k=m})^{\alpha} - i_m^{\alpha} \\ &\quad = 1_{k=m}\sum_{\theta=0}^{\alpha-1}\binom{\alpha}{\theta}(\pm 1)^{\alpha-\theta}i_m^{\theta} \end{aligned} \tag{18}$$

where the upper signs correspond to each other, and the lower signs to each other; substituted in (17), this gives

$$\begin{aligned}&[Q\mathbf{u}(\alpha\mathbf{e}_m)]_{\mathbf{i}} \\ &\quad = \sum_{k\in R_{\mathbf{i}}}\lambda_k 1_{k=m}\sum_{\theta=0}^{\alpha-1}\binom{\alpha}{\theta}i_m^{\theta} \\ &\qquad + \sum_{k=1}^{K} i_k\mu_k 1_{k=m}\sum_{\theta=0}^{\alpha-1}\binom{\alpha}{\theta}(-1)^{\alpha-\theta}i_m^{\theta} \\ &\quad = 1_{m\in R_{\mathbf{i}}}\lambda_m - \mu_m\alpha i_m^{\alpha} \\ &\qquad + \sum_{\theta=1}^{\alpha-1}\left(1_{m\in R_{\mathbf{i}}}\lambda_m\binom{\alpha}{\theta} + \mu_m\binom{\alpha}{\theta-1}(-1)^{\alpha-\theta+1}\right)i_m^{\theta}. \end{aligned} \tag{19}$$

The elements of $Q\mathbf{u}(\alpha\mathbf{e}_m)$ are seen to consist of exactly α single-coordinate monomial terms and at most one constant term.

For double-coordinate monomial basis vectors $\mathbf{u}(\alpha\mathbf{e}_m + \beta\mathbf{e}_n)$, where $m \neq n$ and $\alpha, \beta > 0$, similar manipulations yield

$$\begin{aligned}&[Q\mathbf{u}(\alpha\mathbf{e}_m+\beta\mathbf{e}_n)]_{\mathbf{i}} \\ &\quad = 1_{m\in R_{\mathbf{i}}}\lambda_m i_n^{\beta} + 1_{n\in R_{\mathbf{i}}}\lambda_n i_m^{\alpha} - (\mu_m\alpha+\mu_n\beta)i_m^{\alpha}i_n^{\beta} \\ &\qquad + \sum_{\theta=1}^{\alpha-1}\left(1_{m\in R_{\mathbf{i}}}\lambda_m\binom{\alpha}{\theta} + \mu_m\binom{\alpha}{\theta-1}(-1)^{\alpha-\theta+1}\right)i_m^{\theta}i_n^{\beta} \\ &\qquad + \sum_{\theta=1}^{\beta-1}\left(1_{n\in R_{\mathbf{i}}}\lambda_n\binom{\beta}{\theta} + \mu_n\binom{\beta}{\theta-1}(-1)^{\beta-\theta+1}\right)i_m^{\alpha}i_n^{\theta}. \end{aligned} \tag{20}$$

This is a sum of at most two single-coordinate monomial terms and $\alpha+\beta-1$ double-coordinate monomial terms.

For piecewise constant basis vectors $\mathbf{u}(\mathbf{0}, d)$, $d = 1, \ldots, C$, differences of adjacent elements can be expanded as

$$\begin{aligned}[\mathbf{u}(\mathbf{0}, d)]_{\mathbf{i}\pm\mathbf{e}_k} - [\mathbf{u}(\mathbf{0}, d)]_{\mathbf{i}} &= 1_{\mathbf{i}\pm\mathbf{e}_k\in\Omega(d)} - 1_{\mathbf{i}\in\Omega(d)} \\ &= 1_{\mathbf{i}^{\mathrm{T}}\mathbf{b}=d\mp b_k} - 1_{\mathbf{i}^{\mathrm{T}}\mathbf{b}=d} \end{aligned} \tag{21}$$

and consequently, the elements of $Q\mathbf{u}(\mathbf{0}, d)$ are given by

$$[Q\mathbf{u}(\mathbf{0}, d)]_{\mathbf{i}} = \begin{cases} 1_{k\in R_{\mathbf{i}}}\lambda_k, & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d - b_k \text{ for some } k \\ -\sum_{k\in R_{\mathbf{i}}}\lambda_k - \sum_{k=1}^{K}\mu_k i_k, & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d \\ \mu_k i_k, & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d + b_k \text{ for some } k \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

For a fixed $\mathbf{i} \in \Omega$, the vectors $\mathbf{u}(\mathbf{0}, d)$, $d = 1, \ldots, C$, together contain at most $3K + 1$ terms in the state space coordinates; of these, at most $K + 1$ are constant terms, and at most $2K$ are single-coordinate monomial terms.

Finally, for single-coordinate piecewise monomial basis vectors $\mathbf{u}(\alpha \mathbf{e}_m)_{,d}$, $\alpha > 0$, $d = 1, \ldots, C$, we get

$$
[Q\mathbf{u}(\alpha\mathbf{e}_m, d)]_{\mathbf{i}} = \begin{cases} 1_{k\in R_{\mathbf{i}}}\lambda_k i_m^{\alpha}, & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d - b_k \text{ for } k \neq m, \\ 1_{m\in R_{\mathbf{i}}}\left(\lambda_m + \sum_{\theta=1}^{\alpha}\lambda_m\binom{\alpha}{\theta} i_m^{\theta}\right) & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d - b_m, \\ -\left(\sum_{l\in R_{\mathbf{i}}}\lambda_l\right) i_m^{\alpha} & \\ \quad -\sum_{\substack{l=1\\ l\neq m}}^{K}\mu_l i_l i_m^{\alpha} - \mu_m i_m^{\alpha+1} & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d, \\ \sum_{\theta=0}^{\alpha}\mu_m\binom{\alpha}{\theta}(-1)^{\alpha-\theta} i_m^{\theta+1} & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d + b_m, \\ \mu_k i_k i_m^{\alpha} & \text{if } \mathbf{i}^{\mathrm{T}}\mathbf{b} = d + b_k \text{ for } k \neq m, \\ 0 & \text{otherwise.} \end{cases} \tag{23}
$$

For a fixed $\mathbf{i} \in \Omega$, the vectors $\mathbf{u}(\alpha\mathbf{e}_m, d)$, $d = 1, \ldots, C$, together contain at most $3K + 1 + 2\alpha$ terms in state space coordinates; of these, at most $2K - 2$ are double-coordinate monomial terms, at most $2\alpha + K + 2$ are single-coordinate monomial terms, and at most one is a constant term.

Generally, nonpiecewise monomial basis vectors are transformed by the infinitesimal generator matrix Q into polynomials where both the polynomial degree and the number of different coordinates in individual monomial terms stay unchanged. On the other hand, piecewise monomial basis vectors are transformed into piecewise polynomials that are nonzero in as much as $2K + 1$ state sets $\Omega(c)$, $c = 0, \ldots, C$, and both the polynomial degree and the maximum number of different coordinates in individual monomial terms are increased by one. Because of these properties, the considered set of basis vectors is the largest family of monomial and piecewise monomial basis vectors that, when multiplied by the matrix Q, results in piecewise monomials with no more than two different coordinates. This is essential in keeping the computational complexity of the developed approximation algorithm manageable.

We arrive now at the key observation that allows arranging the vector elements (15) and the matrix elements (16) in a practically computable form. Under the assumption that the decisions $R_{\mathbf{i}}$ of the policy being evaluated are identical for each $\mathbf{i} \in \Omega(c)$, all coefficients of the polynomial expressions for $Q\mathbf{u}_j$, where $\mathbf{u}_j$ is a monomial or piecewise monomial basis vector, stay unchanged over each state set $\Omega(c)$. Denoting by $R_{(c)}$ the common decision $R_{\mathbf{i}}$ in states $\mathbf{i} \in \Omega(c)$, we can express the elements of $Q\mathbf{u}_j$ in the generic polynomial form

$$
[Q\mathbf{u}_j]_{\mathbf{i}} = \sum_{\boldsymbol{\nu}\in E_{cj}} \zeta_{cj}(\boldsymbol{\nu}) \prod_{k=1}^{K} i_k^{\nu_k}, \qquad \text{for } \mathbf{i} \in \Omega(c), \quad c = 0, \ldots, C \tag{24}
$$

where $E_{cj} \subset \mathbb{N}^K$, $c = 0, \ldots, C$, are finite sets of exponent vectors, and $\zeta_{cj}(\boldsymbol{\nu})$ is the coefficient of the monomial $\prod_{k=1}^{K} i_k^{\nu_k}$ within the set of states $\Omega(c)$. Let T_{cj} denote the number of exponent vectors in the set E_{cj}, or in other words the number of terms in the polynomial (24) for specific c and j. For the kinds of basis vectors under consideration, the exponent vectors E_{cj} and the coefficients $\zeta_{cj}(\boldsymbol{\nu})$ for $c = 0, \ldots, C$ and $j = 1, \ldots, J$ follow directly from (19), (20), (22), and (23), and in developing these equations, we gave simple upper bounds for the numbers of terms.

From the definition (3) of the state-specific cost rates, we see that for the assumed kinds of policies $r_{\mathbf{i}}$ is also constant within any set of states $\Omega(c) \ni \mathbf{i}$; we denote the common values by $r_{(c)}$, $c = 0, \ldots, C$. Now by substituting (24), we can rearrange (15) as

$$
\begin{aligned}
&\left[U^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1} - \mathbf{r})\right]_j \\
&\quad = \sum_{\mathbf{i}\in\Omega} [Q\mathbf{u}_j]_{\mathbf{i}}\,(g - r_{\mathbf{i}}) \\
&\quad = \sum_{c=0}^{C} \sum_{\mathbf{i}\in\Omega(c)} \sum_{\boldsymbol{\nu}\in E_{cj}} \zeta_{cj}(\boldsymbol{\nu}) \prod_{k=1}^{K} i_k^{\nu_k} (g - r_{(c)}) \\
&\quad = \sum_{c=0}^{C} \sum_{\boldsymbol{\nu}\in E_{cj}} (g - r_{(c)})\zeta_{cj}(\boldsymbol{\nu}) S(c, \boldsymbol{\nu})
\end{aligned} \tag{25}
$$

and (16) as

$$
\begin{aligned}
&\left[U^{\mathrm{T}}Q^{\mathrm{T}}QU\right]_{hj} \\
&\quad = \sum_{\mathbf{i}\in\Omega} [Q\mathbf{u}_h]_{\mathbf{i}}\,[Q\mathbf{u}_j]_{\mathbf{i}} \\
&\quad = \sum_{c=0}^{C} \sum_{\mathbf{i}\in\Omega(c)} \sum_{\boldsymbol{\nu}\in E_{ch}} \zeta_{ch}(\boldsymbol{\nu}) \prod_{k=1}^{K} i_k^{\nu_k} \sum_{\boldsymbol{\nu}'\in E_{cj}} \zeta_{cj}(\boldsymbol{\nu}') \prod_{k=1}^{K} i_k^{\nu'_k} \\
&\quad = \sum_{c=0}^{C} \sum_{\boldsymbol{\nu}\in E_{ch}} \sum_{\boldsymbol{\nu}'\in E_{cj}} \zeta_{ch}(\boldsymbol{\nu})\zeta_{cj}(\boldsymbol{\nu}') S(c, \boldsymbol{\nu} + \boldsymbol{\nu}')
\end{aligned} \tag{26}
$$

where $S(c, \boldsymbol{\nu})$ is defined as the sum of monomials

$$
S(c, \boldsymbol{\nu}) = \sum_{\mathbf{i}\in\Omega(c)} \prod_{k=1}^{K} i_k^{\nu_k}, \quad \text{for } c = 0, 1, \ldots, C, \ \boldsymbol{\nu} \in \mathbb{N}^K. \tag{27}
$$

Observe that (25) is a linear combination of at most $\sum_{c=0}^{C} T_{cj}$ monomial sums $S(c, \boldsymbol{\nu})$, and (26) is a linear combination of at most $\sum_{c=0}^{C} T_{ch}T_{cj}$ monomial sums $S(c, \boldsymbol{\nu})$. Provided that the

link capacity C and the numbers of terms T_{cj}s are not prohibitively large, (25) and (26) thus express the elements of the matrix $U^{\mathrm{T}}Q^{\mathrm{T}}QU$ and the vector $U^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1}-\mathbf{r})$ as practically computable functions of the monomial sums $S(c, \boldsymbol{\nu})$, $c = 0, \ldots, C$, $\boldsymbol{\nu} \in \mathbb{N}^K$. In Section V, we proceed to consider the practical computation of $S(c, \boldsymbol{\nu})$.

It should be noted that we can accommodate state-dependent call arrival rates λ_k and holding times $1/\mu_k$, as long as these too are constant over each state set $\Omega(c)$. With these extensions, it still holds that the polynomial expressions (24) of $[Q\mathbf{u}_j]_{\mathbf{i}}$, $j = 1, \ldots, J$, stay unchanged over $\mathbf{i} \in \Omega(c)$ for any $c = 0, \ldots, C$, and our general treatment applies with minimal modifications. Moreover, if the arrival rates λ_k are defined by

$$\lambda_k(\mathbf{i}) = \lambda_k(0) + \xi_k i_k, \qquad \text{for } \mathbf{i} \in \Omega \quad \text{and} \quad k = 1, \ldots, K \tag{28}$$

where $\lambda_k(0)$ and ξ_k, $k = 1, \ldots, K$, are arbitrary constants that may depend on the state set $\Omega(c)$ containing $\mathbf{i}$, then the number of terms in the polynomial expressions (24) of $[Q\mathbf{u}_j]_{\mathbf{i}}$ increase somewhat, but the degrees of the considered polynomials stay unchanged, thus only increasing the computational complexity of the developed approximation method by a small constant factor. In particular, the latter extension allows using a finite-source arrival model for some subset of the traffic classes.

V. Computing Sums of Monomials

While the monomial sums $S(c, \boldsymbol{\nu})$ defined in (27) could be computed by the convolution algorithm [19], [20], [1], we develop more efficient recursion formulas for these specific forms of sums. The derivation is a novel extension of Buzen's ideas in his treatment [22] of the convolution algorithm.

Before we treat the general monomial sums (27), let us consider simpler sums of the form

$$s(c, k) = \sum_{\substack{\mathbf{i} \in \mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)} = c}} 1, \quad \text{for } c = 0, \ldots, C; \quad k = 1, \ldots, K \tag{29}$$

where $\mathbf{b}^{(k)} = (b_1\ b_2\ \cdots\ b_k)^{\mathrm{T}}$ is a truncated vector of traffic class trunk requirements. Observe that $S(c, \mathbf{0}) = s(c, K)$, and moreover, since this equals the number of states in the set $\Omega(c)$, the total number of system states is given by

$$N = \sum_{c=0}^{C} s(c, K). \tag{30}$$

When $c > 0$ and $k > 1$, we find a recursive formulation for $s(c, k)$ by the following algebraic manipulations:

$$\begin{aligned} s(c, k) &= \sum_{\substack{\mathbf{i} \in \mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)} = c}} 1 = \sum_{\substack{\mathbf{i} \in \mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)} = c \\ i_k = 0}} 1 + \sum_{\substack{\mathbf{i} \in \mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)} = c \\ i_k > 0}} 1 \\ &= \sum_{\substack{\mathbf{i} \in \mathbb{N}^{k-1} \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k-1)} = c}} 1 + \sum_{\substack{\mathbf{j} \in \mathbb{N}^k \\ \mathbf{j}^{\mathrm{T}}\mathbf{b}^{(k)} = c - b_k}} 1 \\ &= s(c, k-1) + s(c - b_k, k). \end{aligned} \tag{31}$$

The change of variable from $\mathbf{i}$ to $\mathbf{j}$ is based on the identity $\mathbf{i} = \mathbf{j} + \mathbf{e}_k$. To initiate the recursion, we set

$$s(0, k) = 1, \qquad \text{for } k = 1, \ldots, K, \tag{32}$$

$$s(c, k) = 0, \qquad \text{for } c < 0; \quad k = 1, \ldots, K, \tag{33}$$

and

$$s(c, 1) = 1_{b_1 | c}, \qquad \text{for } c > 0 \tag{34}$$

where $b_1|c$ stands for the condition that c is divisible by b_1. Equations (32)–(34) follow directly from the definition (29) of $s(c, k)$. It is easy to see that the work of computing $s(c, k)$ for all $c = 0, \ldots, C$ and $k = 1, \ldots, K$ consists of less than $C(K-1)$ additions. As we have use only for the values $S(c, \mathbf{0}) = s(c, K)$, the values $s(c, k)$ with $k < K$ can be discarded as soon as they are no longer needed. By performing the recursive computation of $s(c, k)$ with k as the outer and c as the inner loop index, at any particular stage storage is needed for no more than C numbers.

Let us now proceed to express $S(c, \boldsymbol{\nu})$ with $\boldsymbol{\nu} \neq \mathbf{0}$ recursively. To begin with, fix an exponent vector $\boldsymbol{\nu} \in \mathbb{N}^K$ such that $\boldsymbol{\nu} \neq \mathbf{0}$, and a $c > 0$. Let m be an index such that $\nu_m > 0$; there clearly is one since $\boldsymbol{\nu} \neq \mathbf{0}$. By reducing on the value of i_m, we can rewrite $S(c, \boldsymbol{\nu})$ as

$$\begin{aligned} S(c, \boldsymbol{\nu}) &= \sum_{\substack{\mathbf{i} \in \mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b} = c}} \prod_{k=1}^{K} i_k^{\nu_k} = \sum_{\substack{\mathbf{i} \in \mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b} = c \\ i_m > 0}} \prod_{k=1}^{K} i_k^{\nu_k} \\ &= \sum_{\substack{\mathbf{i} \in \mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b} = c - b_m}} (i_m + 1)^{\nu_m} \prod_{\substack{k=1 \\ k \neq m}}^{K} i_k^{\nu_k} \\ &= \sum_{\substack{\mathbf{i} \in \mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b} = c - b_m}} \left(\sum_{\theta=0}^{\nu_m} \binom{\nu_m}{\theta} i_m^{\theta} \right) \prod_{\substack{k=1 \\ k \neq m}}^{K} i_k^{\nu_k} \\ &= \sum_{\theta=0}^{\nu_m} \binom{\nu_m}{\theta} \sum_{\substack{\mathbf{i} \in \mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b} = c - b_m}} i_m^{\theta} \prod_{\substack{k=1 \\ k \neq m}}^{K} i_k^{\nu_k} \\ &= \sum_{\theta=0}^{\nu_m} \binom{\nu_m}{\theta} S(c - b_m, \boldsymbol{\nu} + (\theta - \nu_m)\mathbf{e}_m). \end{aligned} \tag{35}$$

On each reduction by this recursion formula, the first argument of S is decreased, and no element of the exponent vector is increased. To provide the ground cases of the recursion, we get from the definition (27) of $S(c, \boldsymbol{\nu})$ that

$$S(c, \boldsymbol{\nu}) = 0, \qquad \text{for } c \leq 0; \quad \boldsymbol{\nu} \in \mathbb{N}^K \tag{36}$$

along with

$$S(c, \mathbf{0}) = s(c, K), \qquad \text{for } c = 0, \ldots, C. \tag{37}$$

To compute the monomial sum $S(c, \boldsymbol{\nu})$ for some fixed arguments c and $\boldsymbol{\nu}$, we need most of the sums $S(c', \boldsymbol{\nu}')$ where $c' < c$ and $\nu_1' \leq \nu_1, \ldots, \nu_K' \leq \nu_K$. In principle this makes the recursive computation inefficient when $\sum_{k=1}^{K} \nu_k$ is not small; fortunately, in the proposed link-cost approximation procedures, we have use for all the recursively referenced sums in any case,

so that the amount of work in computing the necessary monomial sums is of the order $O(nC\nu_{\max})$, where n is the number of monomial sums needed and $\nu_{\max}$ is the largest individual exponent in all the monomials to be summed.

VI. Computational Considerations

The treatment of Section IV can be developed into a practical algorithm by using appropriate data structures as follows. The nonzero elements of the matrix QU are represented symbolically as polynomials in state space coordinates, as in (24), storing for each term the coefficient $\zeta_{cj}(\boldsymbol{\nu})$ along with the exponent vectors $\boldsymbol{\nu} \in E_{cj}$. Note that we have chosen such basic vectors that a single term can have no more than two nonzero exponents ν_k, $k = 1, \ldots, K$; thus it is prudent to store only the nonzero elements ν_k and their indices k for each exponent vector $\boldsymbol{\nu}$. A row of QU is represented sparsely by storing only the nonzero elements, and their column indices. This row representation is identical for all the rows corresponding to a single state set $\Omega(c)$, and so it suffices to store the row representation only once for each $c = 0, \ldots, C$. Accordingly, the complete matrix QU is represented by a total of $\sum_{c=0}^{C} \sum_{j=1}^{J} T_{cj}$ term data structures. Denoting the maximum number of terms on a row of QU by

$$T = \max_{c=0,\ldots,C} \sum_{j=1}^{J} T_{cj} \tag{38}$$

the storage requirement is at worst proportional to $(C+1)T$ when the type of basis is fixed.

By gathering the observations we made about the numbers of terms in (19), (20), (22), and (23), and multiplying these by the numbers of basis vectors of each type, we can bound T by

$$T \le D_1K(D_1+1)+D_2(2E_2-D_2)\tfrac{1}{2}K(K-1)(D_2+E_2+1) + (3K+1)+P_1K(3K+1+2P_1). \tag{39}$$

In estimating the complexity of the approximation algorithm, we regard the parameters D_1, D_2, E_2, P_1 of the basis as fixed and only the link model parameters as varying; thus, we state simply that T is of the order $O(K^2)$.

Before constructing the normal equations, the necessary monomial sums $S(c, \boldsymbol{\nu})$ should be precomputed by the recursion formulas of Section V. As can be seen from (25) and (26), we need values of $S(c, \boldsymbol{\nu})$ for all $c = 0, \ldots, C$ and for such exponent vectors $\boldsymbol{\nu}$ that either $\boldsymbol{\nu} \in E_{cj}$ for some c and j, or $\boldsymbol{\nu} = \mathbf{n} + \mathbf{m}$ with $\mathbf{n} \in E_{ch}$ and $\mathbf{m} \in E_{cj}$ for some c, h and j. Since any exponent vector in the sets E_{cj}, $c = 0, \ldots, C$, $j = 1, \ldots, J$, can have no more than two nonzero elements, and these two nonzero elements can be at arbitrary positions in the exponent vectors, and the actual values of these nonzero elements are bounded by the type of basis, it follows that there are $O(K^2)$ different exponent vectors in the sets E_{cj} and these exponent vectors can be combined into $O(K^4)$ different sums. Consequently, all the necessary monomial sums can be computed in $O(K^4C)$ operations.

Once all the necessary monomial sums $S(c, \boldsymbol{\nu})$ are available, the vector on the right-hand side of the normal equations can be computed by the following algorithm based on (25).

Algorithm 1.

Compute the vector $\mathbf{a} = U^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1} - \mathbf{r})$.
Initialize the J-vector $\mathbf{a}$ with zeros.
Initialize the J-vector $\boldsymbol{z}$ by $z_j = g - r_{(c)}$, $j = 1, \ldots, J$.
Loop for $c = 0, \ldots, C$
  Loop for j over such indices $1, \ldots, J$ that $E_{cj} \neq \emptyset$
    Loop for $\boldsymbol{\nu}$ in E_{cj}
      Add $z_j\zeta_{cj}(\boldsymbol{\nu})S(c, \boldsymbol{\nu})$ to a_j.
    End loop
  End loop
End loop.

Since a sparse row representation is used for QU, the number of indices j such that $E_{cj} = \emptyset$ does not affect the operation count of Algorithm 1, and the total number of operations is hence proportional to $\sum_{c=0}^{C} \sum_{j=1}^{J} T_{cj}$, which is of the order $O(CT) = O(K^2C)$.

Similarly, the matrix on the left-hand side of the normal equations is constructed by the following algorithm based on (26).

Algorithm 2.

Compute the matrix $A = U^{\mathrm{T}}Q^{\mathrm{T}}QU$.
Initialize the J by J matrix A with zeros.
Loop for $c = 0, \ldots, C$
  Loop for h over such indices $1, \ldots, J$ that $E_{ch} \neq \emptyset$
    Loop for j over such indices $1, \ldots, J$ that $E_{cj} \neq \emptyset$
      Loop for $\boldsymbol{\nu}$ in E_{ch}
        Loop for $\boldsymbol{\nu}'$ in E_{cj}
          Add $\zeta_{ch}(\boldsymbol{\nu})\zeta_{cj}(\boldsymbol{\nu}')S(c, \boldsymbol{\nu}+\boldsymbol{\nu}')$ to A_{hj}.
        End loop
      End loop
    End loop
  End loop
End loop.

The total number of operations in Algorithm 2 is proportional to $\sum_{c=0}^{C} \sum_{h=1}^{J} \sum_{j=1}^{J} T_{ch}T_{cj}$, and is thus of the order $O(CT^2) = O(K^4C)$. Note that in practice Algorithm 2 should take advantage of the symmetry of the matrix $U^{\mathrm{T}}Q^{\mathrm{T}}QU$ so as to halve the constant factor of the computational complexity.

Once the normal equations have been constructed, they can be solved via Cholesky factorization in $O(J^3)$ operations, as discussed in [23]. Since linear dependency of the vectors $Q\mathbf{u}$ can be hard to avoid in some cases, it is useful to apply the variant of Cholesky factorization with pivoting and an appropriate termination test; on the subject of the termination test Higham [24] gives more up-to-date advice than is found in [23]. In terms of the link parameters, the solution of the normal equations requires on the order of $O(K^6 + K^3C^3)$ operations; thus, we find that the solution stage actually dominates the complexity of the complete approximation algorithm for the discussed types of bases.

When C is considerably larger than K, the term K^3C^3 dominates the complexity of the algorithm, and may grow prohibitively large for moderate values of K and C. Noting that the term K^3C^3 arises as the cube of the number of piecewise single-coordinate monomial basis vectors, we reduce the computational complexity of the approximation by replacing in the basis the piecewise single-coordinate basis vectors $\mathbf{u}(\alpha\mathbf{e}_m, d)$, $d = 1, \ldots, C-e$, by the combinations

$$\mathbf{u}(\alpha\mathbf{e}_m, 1, C-e) = \sum_{d=0}^{C-e} \mathbf{u}(\alpha\mathbf{e}_m, d),$$

$$\text{for } \alpha = 1, \ldots, P_1; \quad m = 1, \ldots, K \quad (40)$$

where $e \in \mathbb{N}$ is a new basis parameter. The piecewise single-coordinate monomial basis vectors $\mathbf{u}(\alpha\mathbf{e}_m, d)$, $d = C-e+1, \ldots, C$ are kept separate. In numerical experiments, good results were achieved by setting e as the largest e such that the cost rate $r_{(C-e+1)}$ on link occupation level $C-e+1$ is nonzero; in other words, we allow different piecewise terms in the relative value vector only on the topmost e link occupation levels where elements of the vector $g\mathbf{1}-\mathbf{r}$ in the Howard equations (6) have values different from zero. For the complete sharing policy, this means that $e = b_{\max} = \max_{k=1,\ldots,K} b_k$. Then the total number of basis vectors reduces to

$$J = D_1K + D_2(2E_2 - D_2)\,\tfrac{1}{2}K(K-1) + P_1K(e+1) + C \quad (41)$$

and the approximation can be computed in $O(K^6 + K^4C + K^3e^3)$ operations.

We note that the complexity of the approximation algorithm can be decreased considerably by dropping all double-coordinate and piecewise single-coordinate monomial basis vectors, that is, by setting $D_2 = E_2 = P_1 = 0$; however, it was observed in numerical experiments that on some link models no satisfactory fit could be found in less extensive bases.

VII. Extension for More General Link-Control Policies

The proposed approximation is applicable also when the link-control policy R is of a form not allowing easy evaluation of the average cost rate g, as long as the assumption holds that the set $R_{\mathbf{i}}$ of accepted traffic classes in state $\mathbf{i}$ depends only on the number $\mathbf{i}^{\mathrm{T}}\mathbf{b}$ of trunks occupied. We discuss the necessary modifications to the algorithm discussed above.

Let us rearrange the Howard equations (6) in the form

$$\begin{pmatrix} -\mathbf{1} & QU \end{pmatrix} \begin{pmatrix} g \\ \boldsymbol{\alpha} \end{pmatrix} = -\mathbf{r}. \quad (42)$$

Considering this as an overdetermined linear system of N equations in $J+1$ variables, the parameters that minimize the Euclidean norm of the residual vector can be determined from the normal equations

$$\begin{pmatrix} N & -\mathbf{1}^{\mathrm{T}}QU \\ -U^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{1} & U^{\mathrm{T}}Q^{\mathrm{T}}QU \end{pmatrix} \begin{pmatrix} g \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \mathbf{1}^{\mathrm{T}} \\ -U^{\mathrm{T}}Q^{\mathrm{T}} \end{pmatrix} \mathbf{r}. \quad (43)$$

Compared to the earlier form of normal equations (11), here the left-hand side matrix is basically the same as earlier, but with a row and a column augmented, and also the right-hand side vector is of similar structure as the right-hand side of (11).

The normal equations (43) are constructed largely as discussed in Section VI for the earlier form of normal equations, but with the following additional considerations. First, the number of states N in the top left corner of the matrix of (43) is given by (30), and can be computed in $O(C)$ operations from the monomial sums $S(c, \mathbf{0})$, $c = 0, \ldots, C$, which must be precomputed for the matrix construction in any case. Second, the element $\mathbf{1}^{\mathrm{T}}\mathbf{r}$ on the right-hand side of (43) can be computed by

$$\mathbf{1}^{\mathrm{T}}\mathbf{r} = \sum_{\mathbf{i}\in\Omega} r_{\mathbf{i}} = \sum_{c=0}^{C} r_{(c)}S(c, \mathbf{0}) \quad (44)$$

in $O(C)$ operations. It remains to consider the computation of the vectors $U^{\mathrm{T}}Q^{\mathrm{T}}(-\mathbf{1})$ and $U^{\mathrm{T}}Q^{\mathrm{T}}(-\mathbf{r})$. These are both of the form $U^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{y}$ with an N-vector $\mathbf{y}$ that is constant within each state class $\Omega(c)$; but this means that they can be computed by Algorithm 1 of Section VI in $O(K^2C)$ operations simply by initializing $\mathbf{z}$ differently in the second step of the algorithm. All these extra steps are cheap in comparison to the construction of the matrix $U^{\mathrm{T}}Q^{\mathrm{T}}QU$, and neither does the one extra free parameter noticeably increase the complexity of the solution of the normal equations; thus we conclude that the average cost rate g can be treated as a free parameter without changing the total complexity estimates provided in the previous section.

In principle this extension can be used for estimating the average cost rates of link-control policies that do not have product form stationary probabilities; however, in practice the average cost rate estimates are often considerably more inaccurate than the produced link shadow price estimates.

VIII. Numerical Results

Numerical results are presented for three variants of the proposed approximation method on three different links. For comparison, the link shadow prices were also evaluated by the method of Krishnan and Hübner [5], and by the class-oriented transformation method of Dziong *et al.* [12]. The approximation methods were compared by two criteria corresponding to different uses of the methods.

We remark that the method of Krishnan and Hübner can be seen as computing a relative value vector that is a combination of the piecewise constant basis vectors $\mathbf{u}(\mathbf{0}, d)$, $d = 1, \ldots, C$, discussed in the present paper; furthermore, it is possible to interpret the method as a projection method applied to the Howard equations (6) with a weighted inner product. No such interpretation is possible for the method of Dziong *et al.* which uses K different one-dimensional link models to compute the link shadow prices separately for each traffic class.

The parameters of the link models on which the numerical tests were run are shown in Tables I and II. The links in Table I have on the order of 30 000 states, to keep it feasible to compute the exact link shadow prices for comparison. The link model parameters are quite varied, except for the call holding times μ_k^{-1} which are proportional to the traffic class trunk requirements. Note that on link L5 all traffic classes are weighted equally, whereas on link L6 the traffic class weights are proportional to b_k/μ_k, making the optimization criterion equivalent to maximizing the average number of occupied trunks. We remark that in the gain scheduled routing approach of [25] equal weighting

TABLE I
PROPERTIES OF THE THREE SMALLER LINK MODELS L3, L5 AND L6 IN THE NUMERICAL TESTS: LINK CAPACITY, NUMBER OF TRAFFIC CLASSES, TRAFFIC CLASS TRUNK REQUIREMENTS, ARRIVAL RATES, HOLDING TIMES, WEIGHTS, AND THE RATIO OF TOTAL OFFERED TRAFFIC TO LINK CAPACITY

	L3	L5	L6
C	100	70	60
K	3	5	6
b_k	1,2,3	1,2,5,8,15	1,2,5,7,11,14
λ_k	20,20,5	$12,3,\frac{1}{2},\frac{1}{5},\frac{1}{12}$	$9,4,\frac{1}{2},\frac{1}{3},\frac{1}{10},\frac{1}{14}$
μ_k^{-1}	1,2,3	1,2,5,8,15	1,2,5,7,11,14
w_k	1,2,2	1,1,1,1,1	1,4,25,49,121,196
offered	1.45	0.97	1.33

TABLE II
PARAMETERS OF A LINK OF 600 TRUNKS, TESTED UNDER THREE DIFFERENT SETS OF ARRIVAL RATES AS CASES L10H, L10M, AND L10L: TRAFFIC CLASS TRUNK REQUIREMENTS, HOLDING TIMES, WEIGHTS, AND THE DIFFERENT SETS OF ARRIVAL RATES

b_k	2,	2,	4,	4,	6,	6,	8,	8,	10,	10
μ_k^{-1}	1,	1,	1.25,	1.25,	1.67,	1.67,	2.5,	2.5,	5,	5
w_k	2,	1.4,	5,	2.5,	10,	4,	20,	7,	5,	16
L10H:λ_k	15,	15,	12,	12,	10,	10,	6,	6,	4,	4
L10M:λ_k	15,	15,	10,	10,	7,	7,	3,	3,	1.8,	1.8
L10L:λ_k	16,	16,	12,	12,	7,	7,	2.4,	2.4,	1.1,	1.1

is of particular importance. The link parameters in Table II are from [15]; here, the number of link states is approximately $1.9 \cdot 10^{14}$, making direct evaluation of the relative values impossible. The only difference between the parameters in the cases L10H, L10M, and L10L is the amount of offered traffic, which equals, respectively, 1.7, 1.0, and 0.88 times the link capacity.

We compare the following approximation methods: Method KH is the approximation proposed by Krishnan and Hübner, and method COT is the class-oriented transformation method of Dziong *et al.* Method A is the least-squares approximation with few basis vectors of all the considered types, and where the piecewise single-coordinate basis vectors are combined with $e = b_{\max}$, as discussed in Section VI; the parameters specifying the basis are $D_1 = 2$, $D_2 = E_2 = 1$, and $P_1 = 1$. Method B is otherwise the same as method A, but there are more basis vectors; the parameters specifying the basis are $D_1 = 3$, $D_2 = 1$, $E_2 = 2$, $P_1 = 2$, and again $e = b_{\max}$. Method C is the least-squares approximation with no combined piecewise single-coordinate basis vectors, but otherwise only a few of the basis vectors of the different types; the parameters specifying the basis are $D_1 = 0$, $D_2 = E_2 = 1$, $P_1 = 2$, and $e = C$. Note that the bases used in methods B and C are strict supersets of the basis of method A, whereas both contain basis vectors not in the other one. In all our tests the link shadow prices were evaluated for the complete sharing policy.

As the first evaluation criterion, the link shadow prices were compared to the exact link shadow prices solved from the complete Howard equations, and the accuracy was measured by the scaled average error

$$\epsilon_p = \sum_{\mathbf{i}\in\Omega} \sum_{\substack{k=1 \\ \mathbf{i}+\mathbf{e}_k\in\Omega}}^{K} \left| \frac{\tilde{p}_k(\mathbf{i}) - p_k(\mathbf{i})}{w_k} \right| \Bigg/ \sum_{\mathbf{i}\in\Omega} \sum_{\substack{k=1 \\ \mathbf{i}+\mathbf{e}_k\in\Omega}}^{K} 1 \qquad (45)$$

TABLE III
SCALED AVERAGE ERRORS IN LINK SHADOW PRICES

	L3	L5	L6
KH	0.325	1.208	1.106
COT	0.126	4.033	0.068
A	0.006	0.303	0.015
B	0.006	0.225	0.010
C	0.001	0.285	0.012

where $\tilde{p}_k(\mathbf{i}) = [U\boldsymbol{\alpha}]_{\mathbf{i}+\mathbf{e}_k} - [U\boldsymbol{\alpha}]_{\mathbf{i}}$ is the estimated link shadow price of traffic class k in state $\mathbf{i}$, and $p_k(\mathbf{i})$ is the exact link shadow price. The reason for the scaling by the traffic class weights w_k is, that in network level policy improvement the meaningful range of $p_k(\mathbf{i})$ is the interval $[0, w_k]$: if $p_k(\mathbf{i})$ exceeds w_k then calls of class k are denied in state $\mathbf{i}$ regardless of how much larger than w_k the link shadow price is. Thus accuracy in all the traffic classes is assessed as equally important. By directly measuring the accuracy of the estimated link shadow prices we intend to assess the suitability of the method for computing route shadow prices as sums of link shadow prices over the links of a route.

The scaled average errors could only be computed for the smaller links L3, L5 and L6, and are shown in Table III. It is clear that by this measure the methods of Krishnan and Hübner and Dziong *et al.* are inferior to the least-squares fitting methods. The errors in the results of the least-squares fitting methods are very small except on link L5; based on a number of tests reported in [26] and not repeated here, it appears that the scaled average error is consistently worse on links where the traffic class weights are specified as $w_k = 1$ for all $k = 1, \ldots, K$. On links L5 and L6, it appears that the extension of basis from A to B is more beneficial than from A to C, while on link L3 the situation is the opposite; generally method B does better when K is larger and method C does better when K is small.

As the second evaluation criterion, each set of link shadow price estimates was used as the basis of a single policy iteration step, creating an improved policy that accepts calls of class k in state $\mathbf{i}$ exactly when the link shadow price estimate $\tilde{p}_k(\mathbf{i})$ is less than w_k; then the average cost rate of the resulting single-link connection admission control policy was computed. For the smaller links L3, L5, and L6, the average cost rate was computed from the Howard equations, and for the larger links the average cost rate was evaluated by a simulation of 10^6 steps. The results are shown in Table IV, which also includes the average cost rates of the following policies: the complete sharing policy, the policy produced by the first policy iteration based on the exact relative values of the complete sharing policy, and the policies produced by the methods of Marbach *et al.* [27], [15]. Unfortunately, there appear to be some problems with the single-link results in [15] and [27]; the numbers presented here have been computed from the blocking percentages reported in [27].

Measured by the average cost rate, the methods of Krishnan and Hübner and Dziong *et al.* are more competitive with the least-squares fitting methods, but they are inconsistent: each of the older methods fails on one of the test links, producing a policy worse than the complete sharing policy. In the larger test cases, method C could not be used due to computer memory limitations, but otherwise all the least-squares fitting methods perform equally well, and on the small test links the results are very close to those of the first policy iteration on the exact relative

TABLE IV
AVERAGE COST RATES OF THE SINGLE-LINK CONNECTION ADMISSION CONTROL POLICIES RESULTING FROM THE RELATIVE VALUE ESTIMATES, ALONG WITH THE AVERAGE COST RATES OF THE COMPLETE SHARING POLICY AND OF THE POLICY PRODUCED BY THE FIRST EXACT POLICY ITERATION, AND FINALLY THE RESULTS OF THE METHODS PROPOSED BY MARBACH *et al.* [15]

	L3	L5	L6	L10H	L10M	L10L
complete sharing	20.82	0.71	30.13	198	24.4	6.93
1st policy iter.	15.67	0.23	30.13	–	–	–
KH	15.80	0.25	38.74	177	23.2	6.86
COT	15.67	0.78	30.13	142	16.2	6.50
A	15.67	0.21	30.13	144	19.1	5.65
B	15.67	0.21	30.13	144	19.1	5.65
C	15.67	0.21	30.13	–	–	–
TD(0):MLP	–	–	–	176	23.4	6.93
TD(0):Quadratic	–	–	–	186	35.3	6.93

TABLE V
CPU TIMES (IN SECONDS) REQUIRED TO COMPUTE THE APPROXIMATIONS, AND TO SOLVE THE RELATIVE VALUES DIRECTLY FROM THE HOWARD EQUATIONS

	L3	L5	L6	L10H	L10M	L10L
KH	0.01	0.01	0.01	0.36	0.37	0.38
COT	0.00	0.00	0.00	0.01	0.01	0.01
A	0.38	1.1	1.7	29	28	30
B	1.3	7.4	13	260	270	270
C	19	36	37	–	–	–
direct	28	30	35	–	–	–

values. Of the approximate methods of Marbach *et al.*, the multilayer perceptron based method TD(0) : MLP performed similarly to the method of Krishnan and Huebner, but cannot compete with the methods of Dziong *et al.* and ours. The method TD(0) : Quadratic which uses a simple quadratic relative value approximation performed even worse in this case; however, it should be noted that the single-link results of Marbach *et al.* were computed with a discounted cost criterion and are thus not fully comparable.

The CPU times required to compute the reported results are shown in Table V, including the time required to directly solve the Howard equations on the line labeled "direct." All the tests were run on a PC with a 350-MHz AMD K6-2 processor. The bulk of the time in the approximation methods was spent in computing the coefficients of the normal equations in relatively high level Common Lisp code, whereas the exact relative values were computed by a sparse matrix iterative package implemented in the C language. With the exception of method C, the least-squares fitting methods computed the values considerably faster than the exact relative values could be computed; thus the combination of basis vectors discussed in Section VI is quite justified. The methods of Krishnan and Hübner and Dziong *et al.* were several orders of magnitude faster than the proposed methods.

IX. CONCLUSION

We have proposed a practically computable method for estimating the state-dependent link shadow prices on an individual link model for a class of link-control policies that includes the complete sharing policy and trunk reservation policies. The approximation is based on least-squares fitting of polynomial relative values to the Howard equations of the Markov decision process associated with the link. The approximation can be applied to routing and connection admission control on network level, as well as single-link connection admission control. Numerical tests on single-link connection admission control problems indicate that the results are more accurate and consistent than the earlier approximations proposed by Krishnan and Hübner [5], and Dziong *et al.* [12].

Compared to the neurodynamic programming methods of Marbach *et al.* [27], [15], our approach performed better in the single-link numerical test cases. However, their results were computed with a discounted cost criterion, and they used a simpler relative value representation than we did, even though this is not a limitation of their approach. Moreover, the methods of Marbach *et al.* generalize directly to the network routing problem whereas our methods need the assumption of link independence, and have to be embedded in the routing optimization framework of Dziong and Mason [2], [7] in order to be used for routing. On the other hand, Marbach *et al.* report very long computation times despite the simpler relative value representation.

There remain several areas of further study to bring our method closer to practice. In the absence of analytical results on the accuracy of the approximation, more extensive testing on different kinds of link models will be needed. In particular, the method should be incorporated as a part of a network routing and connection admission control optimization procedure in order to evaluate its performance in a simulation environment.

In [8]–[10], the Markov decision-theory-based call admission and routing approach is extended to handle issues of queueing, pre-emption, and variable bit-rate calls. We expect that our approximation approach can be applied to these extended models, although partial blocking of service as treated in [10] will cause additional difficulties.

A problem with our approach is that the amount of work required grows fast with the number of traffic classes K; thus when K approaches a dozen or more, it becomes necessary to consider new ways to simplify the approximation. For example, one could relax the requirement of symmetry of the basis vectors with respect to all the traffic classes, or simplify the link model in some way, perhaps similarly to the approaches in [12], [13], [7].

ACKNOWLEDGMENT

The authors would like to thank S. Aalto and the anonymous referees for their comments.

REFERENCES

[1] K. W. Ross, *Multiservice Loss Models for Broadband Telecommunication Networks*. New York: Springer-Verlag, 1995.
[2] Z. Dziong and L. G. Mason, "Call admission and routing in multiservice loss networks," *IEEE Trans. Commun.*, vol. 42, pp. 2011–2022, Feb. 1994.
[3] A. Kolarov and J. Hui, "Least cost routing in multiple-service networks," in *IEEE INFOCOM*, 1994, vol. 3, pp. 1482–1489.
[4] K. R. Krishnan and T. J. Ott, "State-dependent routing for telephone traffic: Theory and results," in *Proc. 25th IEEE Conf. Decision and Control*, Athens, Greece, 1986, pp. 2124–2128.

[5] K. R. Krishnan and F. Hübner-Szabo de Bucs, "Admission control and state-dependent routing for multirate circuit-switched traffic," in *Proc. 15th ITC*, V. Ramaswami and P. E. Wirth, Eds: Elsevier Science B. V., 1997, pp. 1043–1055.
[6] K. W. Ross and D. H. K. Tsang, "Optimal circuit access policies in an ISDN environment: A Markov decision approach," *IEEE Trans. Commun.*, vol. 37, pp. 934–939, Sept. 1989.
[7] Z. Dziong, *ATM Network Resource Management*. New York: McGraw-Hill, 1997.
[8] E. Nordström, S. Blaabjerg, and G. Fodor, "Admission control of CBR/VBR and ABR/UBR call arrival streams: A Markov decision approach," in *Proc. IEEE ATM Workshop*, 1997, pp. 113–122.
[9] E. Nordström, "Near-optimal link allocation of blockable narrow-band and queueable wide-band call traffic in ATM networks," in *Proc. 15th ITC*, V. Ramaswami and P. E. Wirth, Eds. Washington, D.C.: Elsevier Science, 1997, pp. 987–996.
[10] E. Nordström and J. Carlström, "Call admission control and routing for integrated CBR/VBR and ABR services: A Markov decision approach," in *Proc. IEEE ATM Workshop*, Kochi, Japan, May 1999, pp. 71–76.
[11] R.-H. Hwang, J. F. Kurose, and D. Towsley, "State dependent routing for multirate loss networks," in *Proc. IEEE GLOBECOM*, 1992, vol. 1, pp. 565–570.
[12] Z. Dziong, K.-Q. Liao, L. Mason, and N. Tetreault, "Bandwidth management in ATM networks," in *Proc. 13th Int. Teletraffic Congress*, A. Jensen and V. B. Iversen, Eds., 1991, pp. 821–827.
[13] K.-Q. Liao, Z. Dziong, and L. G. Mason, "Dynamic link bandwidth allocation in an integrated services digital network," in *Proc. IEEE ICC*, Boston, MA, 1989, pp. 977–981.
[14] P. J. Schweitzer and A. Seidmann, "Generalized polynomial approximations in Markovian decision processes," *J. Mathemat. Anal. Applicat.*, vol. 110, pp. 568–582, 1985.
[15] P. Marbach, O. Mihatsch, and J. N. Tsitsiklis, "Call admission control and routing in integrated services networks using neuro-dynamic programming," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 197–208, Feb. 2000.
[16] H. C. Tijms, *Stochastic Models: An Algorithmic Approach*. Chichester, U.K.: Wiley, 1994.
[17] J. S. Kaufman, "Blocking in a shared resource environment," *IEEE Trans. Commun.*, vol. 29, pp. 1474–1481, Oct. 1981.
[18] J. W. Roberts, "A service system with heterogenous user requirements—Application to multiservices telecommunications systems," in *Performance of Data Communication Systems and Their Applications*, G. Pujolle, Ed. Amsterdam: North-Holland, 1981, pp. 423–431.
[19] V. B. Iversen, "A simple convolution algorithm for the exact evaluation of multiservice loss systems with heterogeneous traffic flows and access control," in *NTS-7, Det sjunde nordiska teletrafikseminariet*. Lund, Sweden: Lund Tekniska Högskola, Aug. 1987, p. IX 3.
[20] D. H. K. Tsang and K. W. Ross, "Algorithms to determine exact blocking probabilities for multirate tree networks," *IEEE Trans. Commun.*, vol. 38, pp. 1266–1271, Aug. 1990.
[21] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1989.
[22] J. P. Buzen, "Computational algorithms for closed queueing networks with exponential servers," *Commun. ACM*, vol. 16, pp. 527–531, Sept. 1973.
[23] J. J. Dongarra, C. B. Moler, J. R. Bunch, and G. W. Stewart, *LINPACK User's Guide*. Philadelphia, PA: SIAM, 1979.
[24] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*. Philadelphia, PA: SIAM, 1996.
[25] J. Carlström and E. Nordström, "Gain scheduled routing in multiservice networks," Dept. of Computer Sci. and Information Technol., Uppsala Univ., Uppsala, Sweden, Tech. Rep. 2000-009, 2000.
[26] H. Rummukainen, "On approximative Markov control of multiservice telecommunication links," Laboratory of Telecommunications Technology, Helsinki Univ. of Technology, Espoo, Finland, Tech. Rep. 2/2000, 2000.
[27] P. Marbach and J. N. Tsitsiklis, "A neuro-dynamic programming approach to call admission control in integrated service networks: The single-link case," Massachusetts Inst. of Technology, Cambridge, MA, Tech. Rep. Lab. Inform. Decision Syst. Rep. LIDS-P-2402. [Online.] Available: http://web.mit.edu/jnt/www/publ.html, Nov. 1997.

Hannu Rummukainen received the M.Sc. degree in mathematics from Helsinki University of Technology, Espoo, Finland, in 2000.

Since then, he has been working on vehicle routing algorithms with VTT Information Technology, Espoo, Finland.

Jorma Virtamo (M'00) received the M.Sc. (Tech) degree in engineering physics and D.Sc. (Tech) degree in theoretical physics from the Helsinki University of Technology, Espoo, Finland, in 1970 and 1976, respectively.

In 1986, he joined the Technical Research Center of Finland, VTT Information Technology, where he led a teletraffic research group, and became a Research Professor in 1995. Since 1997, he has been a Professor in the Networking Laboratory of the Helsinki University of Technology. His current research interests include queueing theory and performance analysis and control of the Internet, optical networks and *ad hoc* networks.