TEKNILLINEN KORKEAKOULU
Teknillisen fysiikan ja matematiikan osasto
Teknillisen fysiikan ja matematiikan koulutusohjelma

Hannu Rummukainen

On Approximative Markov Control of Multiservice Telecommunication Links

Diplomi-insinöörin tutkintoa varten tarkastettavaksi jätetty diplomityö

Työn valvoja professori Olavi Nevanlinna
Työn ohjaaja professori Jorma Virtamo

Espoo 29. 2. 2000

Tekijä: Hannu Rummukainen

Työn nimi: Markov-päätöksenteon approksimoinnista usean palvelutyypin tietoliikennelinkeillä

English title: On Approximative Markov Control of Multiservice Telecommunication Links

Päivämäärä: 29. 2. 2000 **Sivumäärä:** 86+v

Osasto: Teknillisen fysiikan ja matematiikan osasto

Professuuri: Mat-1 Matematiikka

Työn valvoja: Professori Olavi Nevanlinna

Työn ohjaaja: Professori Jorma Virtamo

Nykyaikainen piirikytkentäinen tietoliikenneverkko tarjoaa päätelaitteille kahden tai useamman pisteen välisiä yhteyksiä. Verkon ohjauksen ongelmana on valita mitä linkkejä pitkin kukin yhteys reititetään, ja päättää mitkä yhteyspyynnöt hylätään ettei verkon kapasiteetti ylittyisi. Eräin oletuksin nämä päätökset voidaan tehdä optimaalisesti Markov-päätöksentekoteorian keinoin; ongelmana kuitenkin on että käytännössä jo yhden linkin Markov-prosessimallin tila-avaruus voi olla suuruusluokkaa 10^{20} tilaa ja tunnetuissa tarkoissa ratkaisumenetelmissä tarvitaan yksi muuttuja kutakin tilaa kohden.

Työssä käydään läpi Markov-päätöksentekoteoriaa äärellisen tila-avaruuden tapauksessa äärettömän aikahorisontin keskikustannuskriteerin optimoinnin kannalta, sekä teleliikenneteorian perusteita lähtien Erlangin perinteisestä puhelinverkon linkkimallista ja edeten mallin yleistykseen nykyaikaisille tietoliikenneverkoille, joissa välitettävät yhteydet jakautuvat useisiin erilaisiin yhteysluokkiin.

Lähemmin tarkastellaan kahta menetelmää optimaalisessa reitityksessä ja yhteydenhyväksynnässä tarvittavien linkkivarjohintojen estimoimiseksi. Ensimmäinen näistä on Krishnanin ja Hübnerin [29] esittämä politiikkaiteraatio yksinkertaistetussa linkkimallissa, jonka osoitetaan olevan erikoistapaus Schweitzerin ja Kindlen [44] yleisille äärellisille Markov-päätöksentekoprosesseille kehittämästä iteratiivisesta aggregaatio–disaggregaatio-algoritmista sovellettuna täydelliseen linkkimalliin. Tämä tulkinta antaa mahdollisuuden käyttää Schweitzerin ja Kindlen menetelmän disaggregaatio-osuutta Krishnanin ja Hübnerin menetelmän tulosten parantamiseen.

Toisena käytännön approksimaationa kehitetään Schweitzerin ja Seidmannin [46] esittämää ajatusta Markov-päätöksentekoprosessin tilojen ns. suhteellisten arvojen sovittamisesta polynomikantafunktioihin pienimmän neliösumman menetelmällä. Reitityksen ja yhteydenhyväksynnän optimoinnissa tarvittavat linkkivarjohinnat saadaan edelleen tilojen suhteellisten arvojen erotuksina. Työssä esitetään rekursiokaavat joiden avulla pienimmän neliösumman menetelmän normaaliyhtälöt voidaan konstruoida tarvitsematta eksplisiittisesti kertoa suuria matriiseja keskenään, ja näin sovitus voidaan tehdä linkeillä joiden tila-avaruus on suurempi kuin on suoraan käsiteltävissä tunnetuilla pienimmän neliösumman menetelmän laskenta-algoritmeilla. Konstruktio esitetään useille eri tyyppisille sovitettaville kantafunktioille. Käytettyjä rekursiokaavoja ja normaaliyhtälöiden konstruktiomenetelmää ei löydy aikaisemmasta kirjallisuudesta.

Lisäksi esitetään muutamia numeerisia esimerkkejä työssä käsiteltyjen menetelmien tuloksista pienillä linkkimalleilla. Paloittain lineaaristen kantafunktioiden pienimmän neliön sovitus havaitaan kehityskelpoiseksi approksimaatiomenetelmäksi.

Avainsanat: Markov-päätöksentekoprosessi, aggregaatio–disaggregaatio, piirikytkentäiset monibittinopeusverkot, linkkivarjohinnat, MDP-reititys, kutsunhyväksyntäongelma, menetysjärjestelmät

Työn sijaintipaikka:

Author: Hannu Rummukainen

Title of thesis: On Approximative Markov Control of Multiservice Telecommunication Links

Finnish title: Markov-päätöksenteon approksimoinnista usean palvelutyypin tietoliikennelinkeillä

Date: 29. 2. 2000 **Pages:** 86+v

Department: Department of Engineering Physics and Mathematics

Chair: Mat-1 Mathematics

Supervisor: Professor Olavi Nevanlinna

Instructor: Professor Jorma Virtamo

A modern circuit-switched telecommunication network offers point-to-point connections. The problem in network control is to choose the links along which each connection is routed, and to decide which connection requests should be denied in order not to exceed the network capacity. With certain assumptions these decision can be made optimally by the methods of Markov decision theory; however in practice the Markov process model of a single link can contain as many as 10^{20} states, and the known exact algorithms for finding the optimal control policy require as many variables as there are system states.

The thesis contains the necessary background on Markov decision theory in the case of a finite state space and the infinite horizon average cost criterion, as well as the basics of teletraffic theory beginning with the traditional Erlang link model for telephone networks, and proceeding to the generalization of the model to modern telecommunication networks where the carried connections can be divided into traffic classes of different characteristics.

Two methods for estimating link shadow prices, which are required in optimal routing and connection admission control, are discussed in detail. The first one is the policy iteration in a simplified link model proposed by Krishnan and Hübner [29]; it is shown in the thesis that the method is a special case of the iterative aggregation-disaggregation algorithm Schweitzer and Kindle [44] developed for general finite Markov decision processes. This interpretation leads to the possibility of using the disaggregation part of the method of Schweitzer and Kindle to improve the results of Krishnan and Hübner.

As another practical approximation, the thesis includes further development on an idea by Schweitzer and Seidmann [46] to fit polynomial basis functions to the relative values of a Markov decision process by the least squares method. The link shadow prices can then be derived as differences of the relative values of specific states. The normal equations of least squares fitting can be constructed without multiplying large matrices, by using novel recursion formulas presented in the thesis. This makes it possible to perform the least squares fitting on links with a state space larger than can be directly handled by known linear least squares algorithms. The construction of the normal equations is presented for several different types of basis functions. The recursion formulas and the method of constructing the normal equations were not found in earlier literature.

Additionally a few numerical examples are presented to demonstrate the approximation methods discussed in the thesis on small link models. Least squares fitting of piecewise linear basis functions is found to be a method worth further development.

Keywords: Markov decision process, aggregation-disaggregation, multi-rate circuit-switched networks, link shadow prices, MDP routing, connection admission control, loss networks

Library code:

Contents

Notation

Vectors are generally typeset in bold face like $\mathbf{a}$, and their elements in ordinary italics like a_i; all vectors are treated as column vectors.

$\mathbb{N}$	the set of natural numbers $0, 1, 2, \ldots$
$[a, b]$	the closed interval of the real line from a to b
$\|S\|$	the number of elements in set S
$S \setminus T$	the set difference, defined as the set of elements $x \in S$ such that $x \notin T$
$\mathrm{P}[\mathrm{E}]$	unconditional probability of event E
$\mathrm{P}[\mathrm{E} \mid \mathrm{C}]$	probability of event E conditioned on event C
$\mathrm{E}[\mathrm{V}]$	unconditional expectation of random variable V
$\mathrm{E}[\mathrm{V} \mid \mathrm{C}]$	expectation of random variable V conditioned on event C
1_{C}	indicator function defined as 1 when condition C is true and 0 otherwise.
$a\|b$	true if integer b is divisible by integer a, ie there is an integer k such that $b = ka$; the notation $a\|b$ is read as "a *divides* b"
$\lfloor x \rfloor$	the floor function, defined as the largest integer less than or equal to x; in particular $\lfloor x \rfloor = x$ for any integer x
$\mathbf{0}$	a vector of all zeros, of dimension appropriate for the context
$\mathbf{1}$	a vector of all ones, of dimension appropriate for the context
$\mathbf{e}_k$	a unit vector of appropriate dimension, with all elements 0 except for kth element which is 1
$(x_i)_{i \in S}$	a vector with elements given by x_i as i iterates over the elements of set S
x_i	element i of vector $\mathbf{x}$
$[MS\mathbf{c}]_i$	element i of vector $MS\mathbf{c}$. This notation is for vector expressions, and simply c_i is used for elements of a single vector $\mathbf{c}$.
$[TCT]_{ij}$	element (i, j) of matrix TCT. This notation is for complicated expressions and simply t_{ij} is used for elements of a single matrix T.
I	the identity matrix
$\mathbf{v} \leq \mathbf{u}$	true if $v_i \leq u_i$ for all i; all other vector inequalities are interpreted similarly elementwise
$(a_1, a_2, \ldots, a_n)$	a tuple (ordered set) of n elements; we commonly use ordered pairs (a, b) as vector element indices

Chapter 1

Introduction

Despite the rapidly increasing importance of telecommunications, the capacity of the available telecommunication networks is not being used as efficiently as theoretically possible. The naive policy of immediately connecting every call as long as there is free capacity left can be surprisingly inefficient when compared to the theoretically optimal control strategy. Unfortunately, in a realistically sized network it is computationally infeasible to optimally decide which connections to carry on which routes, and which connections to refuse in order to avoid overload. The problem is particularly difficult in modern broadband networks which carry connections of several different characteristics simultaneously.

Current practice is mostly based on ad-hoc solutions with limited theoretical justification. Nevertheless, research on finding practically computable near optimal network control strategies is active and there have been some significant developments. For example, the network policy iteration procedure of Dziong and Mason outlined in Section 3.7.3 approximates the network level optimal control problem in terms of independent link level problems, thus markedly reducing the complexity of the problem.

The Dane A. K. Erlang developed the traditional teletraffic theory, including a very successful theoretical model for traditional telephone networks, in the first decades of the 20th century. The original model is no longer adequate in modern broadband networks, but it can be easily generalized to model the plethora of different traffic types in a modern network. The original Erlang link model and its generalization are continuous-time Markov processes on a finite state space, and the problem of their optimal control can be formulated naturally in the context of Markov decision theory. The subject of this thesis is the approximation of optimal control in the generalized multiservice link model.

The reader is assumed to be familiar with the basic theory of linear algebra, linear programming and stochastic processes including Markov processes, although linear programming is not central to the thesis. It has been attempted to keep the treatment above this level reasonably self-contained; for example no previous knowledge of teletraffic theory is required.

The thesis is organized as follows. Chapter 2 briefly reviews results on finite Markov processes and then provides an introduction into the necessary Markov decision theory. The discussion of Markov decision theory is limited to processes on a finite state space, with the infinite horizon average cost criterion. Chapter 3 then introduces concepts of teletraffic theory and discusses the telecommunication models and algorithms of interest in the sequel. Of specific interest are the Kaufman-Roberts recursion which forms the basis for the approximation method of Krishnan and Hübner discussed in Chapter 4, and the convolution algorithm which leads to the least squares fitting methods developed in Chapter 5.

Chapter 4 introduces the approximation method of Krishnan and Hübner and presents a novel connection of the approximation method of Krishnan and Hübner to so-called

aggregation-disaggregation methods for general finite Markov decision processes. Literature on related methods is surveyed. Chapter 5 reviews an article by Schweitzer and Seidmann which does not seem to have been followed up despite proposing a number of interesting approximation ideas for Markov control. Of the ideas in the article, least squares estimation of relative values is then applied on the multiservice Erlang link model, and efficient methods are developed for performing the estimation on links with a larger state space than could be handled by direct linear least squares algorithms. Part of the efficiency is achieved by novel recursion formulas for computing simple sums over regions of the link state space.

Numerical comparisons of the discussed approximation methods on a few relatively small link models are then presented in Chapter 6, and finally Chapter 7 concludes the thesis with a critical summary of the results and suggestions for future work.

Chapter 2

Markov Decision Theory

We briefly review the necessary basic definitions and results on Markov processes, and provide an introduction to relevant topics on Markov decision processes. Should the reader not be familiar with stochastic processes or Markov processes, we refer to [50], [17] and [18] which also contain comprehensive introductions to Markov processes and Markov decision processes. The present treatment is loosely based on [50]; the presentation of Markov decision theory is limited to a specific formulation appropriate to the telecommunication application that is the subject of the thesis.

2.1 Discrete-time Markov processes

Let S be a finite set of system states, the *state space*. Denote the number of states by N. A discrete-time Markov process on the state space S is a sequence of random variables $I(t) \in S$, $t = 0, 1, \ldots$ that satisfies the Markovian property, that the future development of the system depends only on the current state of the system:

$$\mathrm{P}[I(t+1) = i_{t+1} | I(t) = i_t, I(t-1) = i_{t-1}, \ldots, I(0) = i_0] = \mathrm{P}[I(t+1) = i_{t+1} | I(t) = i_t] \quad \text{for all } t = 0, 1, \ldots \text{ and } i_0, i_1, \ldots, i_t \in S. \quad (2.1)$$

We only consider time-homogenous Markov processes, where the right-hand side of (2.1) does not depend on t. We denote the one-step state transition probabilities $\mathrm{P}[I(t+1) = j \mid I(t) = i]$, $i, j \in S$, by p_{ij}, and define the N by N matrix P with the elements p_{ij}, $i, j \in S$. Matrix P is *stochastic*, that is $P\mathbf{1} = \mathbf{1}$.

The n-step transition probabilities $p_{ij}^{(n)} = \mathrm{P}[I(t+n) = j \mid I(t) = i]$ satisfy the recurrence relation

$$p_{ij}^{(n)} = \sum_{j \in S} p_{ij}^{(n-1)} p_{ji}, \quad i, j \in S,\ n = 1, 2, \ldots \quad (2.2)$$

which can be expressed succinctly in terms of the transition matrices $P^{(n)}$ with elements $p_{ij}^{(n)}$, $i \in S$, $j \in S$, as

$$P^{(n)} = P^{(n-1)} P = P^n, \quad n = 1, 2, \ldots, \quad (2.3)$$

where P^n denotes matrix exponentiation. Since $P^0 = I$, the relation $P^{(n)} = P^n$ holds for all $n = 0, 1, \ldots$.

A set of states $C \subseteq S$ is called *closed* if for every $i \in C$ there is no $j \in S \setminus C$ such that $p_{ij} > 0$. A closed set of states is called *irreducible* if it contains no closed set of states

as a proper subset. It follows that the irreducible sets of a process are disjoint. When the complete state space of the process is an irreducible set, the process as well as its transition matrix are called irreducible.

A state $i \in S$ is *recurrent* if the expected number of visits to i is infinite, otherwise the state is *transient*. When the state space of a Markov process is finite there is always at least one recurrent state, and it is equivalent for a state to be recurrent and to belong to an irreducible set. The *period* of a recurrent state i is defined as the greatest common divisor of the indices $n \geq 1$ such that $p_{ii}^{(n)} > 0$. All recurrent states in the same irreducible set have a common period; denoting this period by M, the irreducible set can be divided into M disjoint subsets which the process visits in a repeating sequence, once it has entered the irreducible set. If all recurrent states have a period of 1, the process is said to be *aperiodic*. A periodic process with a single irreducible set, where the period of the recurrent states is M, can be treated as M sequentially alternating aperiodic processes.

An *equilibrium distribution* $\boldsymbol{\pi} = (\pi_i)_{i \in S} \in \mathbb{R}_+^N$ of a Markov process is defined as a probability distribution on S that satisfies the linear system of equations

$$\begin{aligned} P^{\mathrm{T}}\boldsymbol{\pi} &= \boldsymbol{\pi} \\ \boldsymbol{\pi}^{\mathrm{T}}\mathbf{1} &= 1, \end{aligned} \tag{2.4}$$

which means that $\boldsymbol{\pi}$ is a left eigenvector of P with the eigenvalue 1. The name equilibrium distribution refers to the property that if the initial state is a random variable distributed according to an equilibrium distribution, that is $\mathrm{P}[I(0) = i] = \pi_i$ for all $i \in S$, then the state distribution stays constant for all $t = 0, 1, \ldots$. Because of this the equilibrium probabilities π_i, $i \in S$, are also called *stationary probabilities* or *balance probabilities*.

An important property of the transition probability matrix is that $N - \mathrm{rank}(P - I)$ equals the number of irreducible sets of the process. When the process has a single irreducible set the equilibrium distribution $\boldsymbol{\pi}$ is unique. Further, the long-run fraction of time the process is in state $i \in S$ equals π_i with probability 1, implying

$$\lim_{t \to \infty} \frac{1}{t} \sum_{n=0}^{t-1} P^n = \mathbf{1}\boldsymbol{\pi}^{\mathrm{T}}. \tag{2.5}$$

If the process is aperiodic as well as having a single irreducible set, the n-step transition probabilities converge to the balance probabilities regardless of the state of origin:

$$\lim_{n \to \infty} P^n = \mathbf{1}\boldsymbol{\pi}^{\mathrm{T}}, \tag{2.6}$$

and the rate of convergence is geometric. For proofs, see [50, Chapter 2.3]. In a periodic process where the period of the states in the irreducible set is M we can apply (2.6) to each of the M alternating aperiodic processes, each of which has a separate equilibrium distribution.

2.2 Continuous-time Markov processes

For a continuous-time random process $I(t) \in S$, $t \geq 0$, on the state space S, the continuous-time Markov property states that

$$\begin{aligned} \mathrm{P}[I(t) = i \mid I(t_n) = i_n, I(t_{n-1}) = i_{n-1}, \ldots, I(t_1) = i_1] = \mathrm{P}[I(t) = i \mid I(t_n) = i_n] \\ \text{for all } t > t_n > t_{n-1} > \cdots > t_0 \geq 0 \text{ and } i,\ i_n, i_{n-1}, \ldots, i_1 \in S. \end{aligned} \tag{2.7}$$

Unfortunately this property leads to technical complications like the possibility of infinitely many transitions on a finite time interval; see [17, Chapter 8] for details. We avoid the complications by working on a practically useful subset of general continuous-time Markov processes. As in the discrete case, we only consider time-homogenous processes where the right-hand side of (2.7) does not depend on t.

We define a continuous-time Markov process on a finite state space S as a continuous-time random process $I(t) \in S$, $t \geq 0$, that satisfies the following conditions:

1. When the process is in state $i \in S$, it stays there for an exponentially distributed time with the expectation τ_i, independently of the history of the system.

2. When leaving state $i \in S$, the probability p_{ij} of the process making a transition to each state $j \in S$ depends only on i and j and is independent of the history of the system.

Additionally, to avoid ambiguities in the state sojourn times, we require $p_{ii} = 0$ for all $i \in S$. The number of states in S is denoted by N.

It follows from the properties of the exponential distribution that we have for probabilities of transition from any state $i \in S$ on an arbitrary time interval $(t, t + \Delta t)$ that

$$\mathrm{P}[I(t+\Delta t) = j \mid I(t) = i] = \begin{cases} \frac{p_{ij}}{\tau_i}\Delta t + o(\Delta t) & \text{if } j \neq i, \\ 1 - \frac{1}{\tau_i}\Delta t + o(\Delta t) & \text{if } j = i, \end{cases} \tag{2.8}$$

where $o(x)$ denotes any function such that $\lim_{x\to 0} o(x)/x = 0$. The infinitesimal *transition rates* q_{ij}, $i, j \in S$, are defined as

$$q_{ij} = \lim_{\Delta t \to 0_+} \frac{1}{\Delta t}\mathrm{P}[I(t+\Delta t) = j \mid I(t) = i] = \begin{cases} p_{ij}/\tau_i & \text{if } j \neq i, \\ -1/\tau_i & \text{if } j = i. \end{cases} \tag{2.9}$$

In many practical applications the transition rates have more direct physical significance than the transition probabilities p_{ij}. Observe that the set of transition rates q_{ij}, $i, j \in S$, uniquely determines the expected state sojourn times by $\tau_i = 1/\sum_{j\neq i} q_{ij}$ and the transition probabilities by $p_{ij} = q_{ij}\tau_i$ for $j \neq i$. Also note that the sequence of states visited by the process, ignoring the times of transitions, forms a discrete-time Markov process with transition probabilities p_{ij}, $i, j \in S$; this is called the *embedded* process.

The *infinitesimal generator matrix* Q of the process is the N by N matrix with elements q_{ij}, $i \in S$, $j \in S$. The transformation from transition probabilities to transition rates can be expressed in matrix form as $Q = \mathcal{T}^{-1}(P - I)$ where $\mathcal{T}$ is the N by N diagonal matrix with element (i, i) equal to τ_i. Since the matrix P of transition probabilities is stochastic, we have $Q\mathbf{1} = \mathbf{0}$. The matrix of transition probabilities $P(t)$ over a finite time interval, with elements $[P(t)]_{ij} = \mathrm{P}[I(t) = j \mid I(0) = i]$ is given by

$$P(t) = e^{tQ} = \sum_{n=0}^{\infty} \frac{t^n}{n!} Q^n, \quad t \geq 0. \tag{2.10}$$

The concepts of closed and irreducible sets, as well as recurrent and transient states are entirely analogous to the discrete-time case; on the other hand the concepts of state periods and aperiodicity have no relevance in the continuous-time setting. An equilibrium distribution $\boldsymbol{\pi} = (\pi_i)_{i\in S} \in \mathbb{R}_+^N$ of a continuous-time Markov process is defined as a solution of the system of equations

$$\begin{aligned} Q^{\mathrm{T}}\boldsymbol{\pi} &= \mathbf{0}, \\ \boldsymbol{\pi}^{\mathrm{T}}\mathbf{1} &= 1. \end{aligned} \tag{2.11}$$

This means that an equilibrium distribution vector $\boldsymbol{\pi}$ is a left eigenvector of Q with the eigenvalue 0, or in other words belongs to the null space of Q. Considering the transition rates as "probability flow" between states, the equilibrium equations (2.11) can be informally thought of as equating the sum $\sum_{j \neq i} \pi_i q_{ij}$ of probability flows out of each state $i \in S$ with the sum $\sum_{j \neq i} \pi_j q_{ji}$ of incoming probability flows.

Again, if the system has a single irreducible set, it has a unique equilibrium distribution, and the long-run fraction of time the process spends in state $i \in S$ equals π_i with probability 1, implying

$$\lim_{t \to \infty} \frac{1}{t} \int_0^t \mathrm{P}[I(u) = j \mid I(0) = i] \, du = \pi_j \quad \text{for all } i, j \in S. \tag{2.12}$$

In fact the stronger result

$$\lim_{t \to \infty} \mathrm{P}[I(t) = j \mid I(0) = i] = \pi_j \quad \text{for all } i, j \in S, \tag{2.13}$$

holds as well, and the rate of convergence is exponential.

2.2.1 Uniformization

The method of *uniformization* converts a continuous-time Markov process into an essentially equivalent process where the expected sojourn time in every state is equal, denoted by τ. Because of the uniform sojourn times the transition epochs occur as a Poisson process with parameter 1, and we can treat the uniformized process as a discrete-time process with the complication that the discrete time counter at moment t of continuous time is a random variable distributed as $\mathrm{Poisson}(t/\tau)$. The actual transformation is very simple: Let τ be a constant such that $0 < \tau \leq \tau_i$ for all $i \in S$. The transition probabilities of the discrete-time process embedded in the uniformized process are now given by the stochastic matrix $\bar{P} = \tau Q + I$.

The equivalence of the transition probabilities of the original continuous-time process and the uniformized process follows from (2.10) by

$$P(t) = e^{tQ} = e^{t(\bar{P}-I)/\tau} = e^{-t/\tau} e^{t\bar{P}/\tau} = \sum_{n=0}^{\infty} e^{-t/\tau} \frac{(t/\tau)^n}{n!} \bar{P}^n \tag{2.14}$$

where the last form equals the transition probabilities of the uniformized process conditioned on the number of Poisson events up to time t. We note that the balance probabilities of the discrete-time process embedded in the uniformized process are equal to those of the continuous-time process:

$$Q^{\mathrm{T}} \boldsymbol{\pi} = 0 \iff \frac{1}{\tau}(\bar{P}^{\mathrm{T}} - I)\boldsymbol{\pi} = 0 \iff \bar{P}^{\mathrm{T}} \boldsymbol{\pi} = \boldsymbol{\pi}. \tag{2.15}$$

Additionally, the embedded discrete-time process is guaranteed to be aperiodic if τ is chosen strictly smaller than $\min_{i \in S} \tau_i$.

The uniformization process provides a convenient basis for computing some properties of the Markov process as discussed in [50, Section 2.8], but from the point of view of this thesis the importance of the uniformization method is that it provides a remarkable connection between discrete-time and continuous-time Markov decision models.

2.3 Discrete-time Markov decision processes

A Markov decision process is essentially a Markov process with several alternative sets of transition probabilities from which one can choose in each state, but where every choice is associated with a cost. The principal problem in the theory of Markov decision processes is to make the choices in such a way as to minimize the accumulating costs. Alternatively one can also maximize accumulating rewards, but this case is equivalent to setting the costs equal to the additive inverses of the rewards. Markov decision models have applications in many problems of optimal control on topics such as maintenance, inventory control, resource allocation etc.

Let S be the finite set of system states, and let each state $i \in S$ be associated with a finite set $A(i)$ of possible control decisions, *actions*. When the system enters a new state $i \in S$, a control decision $a \in A(i)$ is chosen by arbitrary means; depending on the state i and the action a, a random cost with the expected value $c_i(a)$ is incurred. The action a uniquely determines the set of transition probabilities $p_{ij}(a)$, $j \in S$; like an ordinary Markov process the system makes a transition to a state $j \in S$ by the probability $p_{ij}(a)$ independently of the history of the system.

A *control policy* is a rule for determining the action at each decision epoch. A *stationary policy* R is a control policy that assigns each state $i \in S$ always the same action, which we denote by R_i. In general, control decisions may depend on knowledge of the full history of the system, but when the state space is finite a stationary policy is always sufficient to reach the absolutely minimal long-time average costs; see [18, Chapter 4] for a proof. Hence we only consider stationary control policies from now on.

A Markov decision process run under a stationary policy R can be considered a Markov process with an associated cost structure. We denote the transition probability matrix of the Markov process by $P(R)$, and the vector of incurred costs in each state by $\mathbf{c}(R)$. Let us further define the expected cost function $V_i(t, R)$ where $i \in S$ and $t = 0, 1, \ldots$ as the expected value of the total accumulated cost by time t when the initial state is i and the system is controlled by policy R; we denote the corresponding vector of expected costs in each state by $\mathbf{V}(t, R)$. By conditioning on the state the system visits on each step we find that

$$\mathbf{V}(t, R) = \sum_{n=0}^{t-1} P(R)^n \mathbf{c}(R). \tag{2.16}$$

Now if the associated Markov process has a single irreducible set, it has a unique equilibrium distribution $\boldsymbol{\pi}(R) = (\pi_i(R))_{i \in S}$ and we have by (2.5) that

$$\lim_{t \to \infty} \frac{1}{t} \mathbf{V}(t, R) = \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}} \mathbf{c}(R), \tag{2.17}$$

or in other words the expected long-run average cost per unit time equals the constant $\boldsymbol{\pi}(R)^{\mathrm{T}} \mathbf{c}(R)$ regardless of initial state. In fact it follows from the strong law of large numbers that the actual long-run average cost per unit time equals the expected value $\boldsymbol{\pi}(R)^{\mathrm{T}} \mathbf{c}(R)$ with probability 1; this is shown in [17, Theorem 7-14].

The long-run average cost per unit time under a control policy is such a central concept that it is abbreviated simply as *average cost*; we denote the average cost of policy R by $g(R)$. It should be mentioned that if future costs are of less importance than immediate costs, one may define a *discount factor* $\beta \in (0, 1)$ so that expected costs t steps in the future are weighted by β^t; this variation is referred to as the *discounted cost criterion* as opposed to the *average cost criterion* used in the present work. As another alternative one may only

be interested in the expected costs over a finite time interval. We refer the interested reader to [18] for more information on alternative formulations.

A policy R^* that satisfies $g(R^*) \leq g(R)$ for all stationary policies R is called *average cost optimal*. To avoid unnecessary complications in the development of algorithms for finding an average cost optimal policy, we make the following assumption, which according to [50] is satisfied in most practical applications.

Unichain Assumption. *Under all stationary policies the associated Markov process has a single irreducible set.*

The *relative values* $\mathbf{w}(R) = (w_i(R))_{i \in S}$ characterize the effect in the total expected costs of starting the system in different states; they are defined by

$$\mathbf{w}(R) = \lim_{t \to \infty} \mathbf{V}(t, R) - tg(R)\mathbf{1}. \tag{2.18}$$

The existence of the limit vector and the finiteness of its elements can be proved for aperiodic processes by using the geometric convergence of (2.6) and for periodic processes by treating them as sequentially alternating aperiodic processes.

The relative values play a central role in the algorithms for finding an average cost optimal policy. The relative differences $w_j(R) - w_i(R)$ in relative values between pairs of states $i, j \in S$ measure the benefit of being in state i rather than state j; in order to decide which actions to change in order to improve a policy, it is sufficient to know these relative differences. This is what makes the following theorem useful.

Theorem 2.1. *Let R be a stationary policy for a Markov decision process on the finite state space S, and let the associated Markov process have a single irreducible set. Then all the solutions of the linear system of equations in the variables g and $\mathbf{v} = (v_i)_{i \in S}$*

$$\mathbf{v} = \mathbf{c}(R) - g\mathbf{1} + P(R)\mathbf{v}, \tag{2.19}$$

are given by

$$g = g(R), \tag{2.20}$$

$$\mathbf{v} = \mathbf{w}(R) + b\mathbf{1}, \tag{2.21}$$

where b is an arbitrary constant, $g(R)$ is the average cost of policy R, and $\mathbf{w}(R)$ is the vector of relative values of the Markov decision process run under policy R.

Proof. Under the conditions of the theorem the associated Markov process has a unique equilibrium distribution $\boldsymbol{\pi}(R)$, a unique average cost $g(R) = \boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{c}(R)$ and unique relative values $\mathbf{w}(R)$. We show first that solutions of the form specified in the theorem satisfy the system of equations (2.19):

$$\begin{aligned}
\mathbf{v} = \mathbf{w}(R) + b\mathbf{1} &= \lim_{t \to \infty} \sum_{n=0}^{t-1} P(R)^n \mathbf{c}(R) - tg(R)\mathbf{1} + b\mathbf{1} \\
&= \mathbf{c}(R) - g(R)\mathbf{1} + \lim_{t \to \infty} \sum_{n=1}^{t-1} P(R)^n \mathbf{c}(R) - (t-1)g(R)\mathbf{1} + b\mathbf{1} \\
&= \mathbf{c}(R) - g(R)\mathbf{1} + P(R)\left(\lim_{t \to \infty} \sum_{n=0}^{t-2} P(R)^n \mathbf{c}(R) - (t-1)g(R)\mathbf{1} \right) + b\mathbf{1} \\
&= \mathbf{c}(R) - g(R)\mathbf{1} + P(R)\mathbf{w}(R) + bP(R)\mathbf{1} \\
&= \mathbf{c}(R) - g(R)\mathbf{1} + P(R)\mathbf{v}.
\end{aligned} \tag{2.22}$$

It remains to show that these are the only solutions.

Let g and $\mathbf{v}$ form an arbitrary solution of the linear equations (2.19). Taking the dot product of the equilibrium vector $\boldsymbol{\pi}(R)$ and both sides of the linear system (2.19) we get

$$\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{v} = g(R) - g + \boldsymbol{\pi}(R)^{\mathrm{T}}P(R)\mathbf{v} \tag{2.23}$$

where the terms involving $\mathbf{v}$ cancel out since $P(R)^{\mathrm{T}}\boldsymbol{\pi}(R) = \boldsymbol{\pi}(R)$ by definition, and we find that $g(R) = g$, thus proving that (2.20) holds.

To prove that $\mathbf{v} = \mathbf{w}(R)+b\mathbf{1}$ for some b, we note that by (2.20) and (2.22) the difference $\mathbf{v} - \mathbf{w}(R)$ satisfies

$$\mathbf{v} - \mathbf{w}(R) = P(R)(\mathbf{v} - \mathbf{w}(R)), \tag{2.24}$$

or in other words $\mathbf{v} - \mathbf{w}(R)$ lies in the null space of $P(R) - I$. By the assumption that the associated Markov process has a single irreducible set, $\mathrm{rank}(P(R) - I) = |S| - 1$ and the null space of $P(R) - I$ is one-dimensional. But since $P(R)$ is stochastic, the vector $\mathbf{1}$ lies in the null space of $P(R) - I$ and we conclude that $\mathbf{v} - \mathbf{w}(R)$ must be a scalar multiple of $\mathbf{1}$, thus completing the proof. □

The equations of the linear system (2.19) are known as Howard equations; they can be interpreted as equating the relative value of a state i with the the relative value accumulated in state i plus the expected relative value accumulated in all future states. If we impose an additional constraint setting an arbitrary v_i to a constant value, the combined system of equations has a unique solution. Thus for a stationary policy R we can solve from a single linear system both the average cost $g(R)$ and the relative differences $w_j(R) - w_i(R)$ of the relative values of pairs of states.

We remark that it can be shown that the sum of the relative values of recurrent states is always zero. This means that it is possible to solve the actual relative values via the Howard equations; however, determining the set of recurrent states of a policy may not be trivial, and in practice it is sufficient to know the relative differences of the relative values. In fact the absolute magnitudes of the relative values are of so small practical importance that often any values $\mathbf{v}$ solving the Howard equations are referred to as relative values.

2.3.1 Policy iteration

As the first of the three main algorithms for finding an optimal stationary policy we discuss the policy iteration algorithm. The basic idea of the algorithm is to iteratively construct improved policies until an average cost optimal policy is found. The basis of the algorithm is established in the following proposition.

Proposition 2.2. *Let R be a stationary control policy for a Markov decision process on a finite state space S, and suppose that the unichain assumption is satisfied. Let g be some arbitrary number and $\mathbf{v} \in \mathbb{R}^{|S|}$ be an arbitrary vector. Suppose that*

$$\mathbf{c}(R) - g\mathbf{1} + P(R)\mathbf{v} \leq \mathbf{v}. \tag{2.25}$$

Then the average cost $g(R)$ of R satisfies $g(R) \leq g$. Further, $g(R) < g$ if and only if strict inequality holds in (2.25) *for some state that is recurrent under policy R.*

The above statements are true also with the directions of all the inequality signs reversed.

Proof. Under the unichain assumption the Markov process associated with policy R has unique balance probabilities $\boldsymbol{\pi}(R) = (\pi_i(R))_{i\in S}$, and a unique average cost $g(R) =$

$\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{c}(R)$. Taking the dot product of $\boldsymbol{\pi}(R)$ with both sides of the system (2.25) we get

$$g(R) - g + \boldsymbol{\pi}(R)^{\mathrm{T}} P(R)\mathbf{v} \leq \boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{v} \tag{2.26}$$

with strict inequality if and only if there is strict inequality in (2.25) for some $i \in S$ such that $\pi_i(R) > 0$; but this is equivalent to state i being recurrent under policy R. The terms involving $\mathbf{v}$ in (2.26) cancel each other since $P(R)^{\mathrm{T}}\boldsymbol{\pi}(R) = \boldsymbol{\pi}(R)$, and the equation reduces to $g(R) \leq g$, again with strict inequality if and only if there is strict inequality in (2.25) for some recurrent $i \in S$. Thus the proof for the inequalities explicitly stated in the proposition is complete.

The fact that the inequality signs can be reversed in the proposition is apparent, since the directions of the inequalities can as easily be reversed in the proof without breaking the chain of conclusions. □

In particular, if the values of g and $\mathbf{v}$ in Proposition 2.2 are computed from the Howard equations (2.19) for another policy R', we get an inequality relation between the average costs of two stationary policies.

We are now ready to formulate the policy iteration algorithm. To begin with, choose an arbitrary policy R as a starting point for the iteration.

1. *(value determination step)* Solve g and $\mathbf{v}$ from the augmented Howard equations

$$\mathbf{c}(R) - g\mathbf{1} + P(R)\mathbf{v} = \mathbf{v}, \tag{2.27}$$

$$v_m = 0, \tag{2.28}$$

   where $m \in S$ is an arbitrary state.

2. *(policy improvement step)* Construct an improved policy R' as follows. For each state $i \in S$, find an action $a \in A(i)$ that minimizes the expression

$$c_i(a) - g + \sum_{j \in S} p_{ij}(a)\, v_j. \tag{2.29}$$

   If the minimal value of the expression is reached by the action R_i of the previous policy, choose that action as the decision of the new policy. Otherwise, choose any $a \in A(i)$ that minimizes the expression.

3. *(convergence test)* If the new policy R' differs from the previous policy R, start over from step 1 with the new policy assigned to R.

We now show that under the unichain assumption the policy iteration algorithm converges to an average cost optimal policy in a finite number of steps. First note that the policy improvement step ensures that the new policy R' and the values g and $\mathbf{v}$ solved in the value determination step for the previous policy R satisfy

$$\mathbf{c}(R') - g\mathbf{1} + P(R')\mathbf{v} \leq \mathbf{v} \tag{2.30}$$

with strict inequality for exactly the states $i \in S$ for which $R'_i \neq R_i$. By Proposition 2.2 this implies for the average costs of policies R' and R that either

a) $g(R') < g(R)$ and (2.30) is a strict inequality for at least one state that is recurrent under policy R', or

b) $g(R') = g(R)$ and (2.30) is an equality for all recurrent states of policy R'.

We shall now prove that in case b) either

b.1) $R' \neq R$ and $\mathbf{w}(R') \leq \mathbf{w}(R)$ with strict inequality for at least one state, or

b.2) $R' = R$ and $\mathbf{w}(R') = \mathbf{w}(R)$.

We use the shorthand $R' = R$ to denote that $R'_i = R_i$ for all $i \in S$, and $R' \neq R$ for the converse of $R' = R$.

By Theorem 2.1 the values $\mathbf{v}$ solved in the value determination step satisfy $\mathbf{v} = \mathbf{w}(R) + b\mathbf{1}$ for some constant b. Subtracting $b\mathbf{1}$ from both sides of (2.30) we get

$$\mathbf{c}(R') - g(R)\mathbf{1} + P(R')\mathbf{w}(R) \leq \mathbf{w}(R) \tag{2.31}$$

where the inequality is strict exactly for the states i such that $R'_i \neq R_i$; in case b) these states must be transient under policy R'. Hence $R'_i = R_i$ for all i that are recurrent under policy R', but this implies that the recurrent set of R is the same as that of R'. Since a recurrent set of states is necessarily closed, the relative values of recurrent states are by definition independent of the policy outside the recurrent set. Thus we have $w_i(R') = w_i(R)$ for all i in the common recurrent set of R and R'.

By Theorem 2.1 we have for the relative values of policy R' that

$$\mathbf{c}(R') - g(R')\mathbf{1} + P(R')\mathbf{w}(R') = \mathbf{w}(R'). \tag{2.32}$$

Since we are studying case b), $g(R') = g(R)$ by assumption and we get by subtracting (2.32) from (2.31) that

$$P(R')\big(\mathbf{w}(R) - \mathbf{w}(R')\big) \leq \mathbf{w}(R) - \mathbf{w}(R'). \tag{2.33}$$

Substituting (2.33) repeatedly into itself, we get

$$\mathbf{w}(R) - \mathbf{w}(R') \geq \cdots \geq P(R')^n\big(\mathbf{w}(R) - \mathbf{w}(R')\big) \quad \text{for all } n = 1, 2, \ldots . \tag{2.34}$$

By summing the inequalities (2.34) over $n = 1, 2, \ldots, t$, dividing by t and letting t tend to infinity, we get

$$\mathbf{w}(R) - \mathbf{w}(R') \geq \lim_{t\to\infty} \frac{1}{t} \sum_{n=0}^{t-1} P(R')^n \big(\mathbf{w}(R) - \mathbf{w}(R')\big) = \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big(\mathbf{w}(R) - \mathbf{w}(R')\big) = \mathbf{0} \tag{2.35}$$

where the last inequality follows from the observations that $w_i(R) - w_i(R') = 0$ for i recurrent under R and $\pi_i(R) = 0$ for i transient under R. We have now established that in case b) $\mathbf{w}(R') \leq \mathbf{w}(R)$ always holds.

To complete the proof that case b) is completely characterized by sub-cases b.1) and b.2), let us suppose that $\mathbf{w}(R') = \mathbf{w}(R)$. Since in case b) $g(R) = g(R')$, inequality (2.31) now states that

$$\mathbf{c}(R') - g(R')\mathbf{1} + P(R')\mathbf{w}(R') \leq \mathbf{w}(R'), \tag{2.36}$$

with strict inequality for states i such that $R'_i \neq R_i$. But by Theorem 2.1 both sides of (2.36) are equal, and we must have $R' = R$. Thus in case b) either b.2) $R = R'$ or b.1) $\mathbf{w}(R') \leq \mathbf{w}(R)$ with strict inequality for some element.

The rest of the proof is straightforward. We have now divided the progress made by the policy iteration algorithm into three distinct cases a), b.1) and b.2). This division indicates that as long as $R' \neq R$, on each iteration of the algorithm either $g(R)$ decreases or $g(R)$

stays constant and $\mathbf{w}(R)^{\mathrm{T}}\mathbf{1}$ decreases. Since there is only a finite number of different stationary control policies, and by the unichain assumption each stationary policy R has a unique average cost $g(R)$ and relative values $\mathbf{w}(R)$, it follows that after a finite number of iterations $R' = R$ and the algorithm terminates. By the construction of R' we have for the final policy R^* that

$$c_i(a) - g(R^*) + \sum_{j \in S} p_{ij}(a)\, v_j(R^*) \geq v_i(R^*) \quad \text{for all } i \in S \text{ and } a \in A(i), \tag{2.37}$$

and by Proposition 2.2 this implies that $g(R^*) \leq g(R)$ for any stationary policy R. Thus the final policy R^* is average cost optimal, and the proof is complete.

As can be seen from the proof, actions of the previous policy must have a special role in the policy improvement step, since otherwise the algorithm could end up oscillating indefinitely between some policies with identical average cost and relative values.

In practice the convergence of policy iteration is extremely fast; according to [50] in most applications convergence typically happens in 3 to 15 iterations regardless of the number of system states; this was certainly found true in the context of the present thesis. It is also asserted in [50] that in many applications the first iteration of the policy iteration algorithm alone is sufficient to produce a near-optimal policy.

The two remaining policy optimization algorithms are not discussed as thoroughly since they have less relevance to the main thesis.

2.3.2 Value iteration

The value iteration algorithm simultaneously computes relative value estimates and improved policies. Unlike policy iteration, value iteration approaches an average cost optimal policy only asymptotically. However, given a pre-specified tolerance ε, value iteration finds in a finite number of steps a policy R such that the relative difference between its average cost $g(R)$ and the optimal average cost is less than ε.

Value iteration does not need the full unichain assumption in order to guarantee convergence; the following weakened version is sufficient.

Weak Unichain Assumption. *Under any average cost optimal policy the associated Markov process has a single irreducible set.*

However, if it is possible for the Markov process associated with an average cost optimal policy to be periodic, value iteration is not directly applicable. In this case one must first apply the following data transformation: Modify the transition probabilities by $p'_{ij}(a) = \tau p_{ij}(a)$ for all $j \neq i$, $i \in S$, and $p'_{ii}(a) = \tau p_{ii}(a) + 1 - \tau$ for all $i \in S$, where $\tau \in (0, 1)$ is a constant, and leave all the other model parameters unchanged. For any stationary policy R, the transformation modifies the transition probability matrix by $P'(R) = \tau P(R) + (1-\tau)I$, and we see that balance probabilities do not change:

$$\begin{aligned} P(R)^{\mathrm{T}}\pi(R) = \pi(R) &\iff \frac{1}{\tau}\big(P'(R) - (1-\tau)I\big)\pi(R) = \pi(R) \\ &\iff P'(R)\pi(R) - \pi(R) + \tau\pi(R) = \tau\pi(R) \iff P'(R)\pi(R) = \pi(R) \end{aligned} \tag{2.38}$$

and consequently the average costs are unchanged as well. The transformation does make sure that $p_{ii} > 0$ for all $i \in S$ so that the Markov process associated with any stationary policy is aperiodic.

The value iteration algorithm works as follows.

1. Let ε be a given tolerance. Set the initial relative value estimates $V_i^{(0)}$, $i \in S$, equal to zero. Initialize the iteration counter n as 1.

2. Compute new value estimates by

$$V_i^{(n)} = \min_{a \in A(i)} \left\{ c_i(a) + \sum_{j \in S} p_{ij}(a)\, V_j^{(n-1)} \right\}, \quad \text{for all } i \in S. \tag{2.39}$$

   Define policy $R(n)$ as follows: The decision in state $i \in S$ is arbitrarily chosen as one of the actions $a \in A(i)$ that reach the minimal value on the right-hand-side of (2.39) for state i.

3. Define the bounds

$$m_n = \min_{i \in S} V_i^{(n)} - V_i^{(n-1)} \quad \text{and} \quad M_n = \max_{i \in S} V_i^{(n)} - V_i^{(n-1)}. \tag{2.40}$$

   If $(M_n - m_n)/m_n \geq \varepsilon$, return to step 2 with the iteration counter n incremented by one. Otherwise, stop the iteration with the policy $R(n)$.

In effect, the 1st step of the value iteration algorithm minimizes the cost of a single step under policy $R(1)$ when each final state $i \in S$ causes a cost $V_i^{(0)}$. Continuing this way, the nth step minimizes the total accumulated costs supposing that the process is stopped after n steps with terminal costs $V_i^{(0)}$, $i \in S$, and that the policies $R(i)$, $1 \leq i \leq n$, are applied on each step in reverse order of their construction. This procedure is essentially an application of dynamic programming [36].

It can be shown [50, Theorem 3.4.2] that if for any average cost optimal policy the associated Markov process is aperiodic, then the bounds m_n and M_n in the value iteration algorithm converge geometrically to the optimal average cost. This property is the reason we need to apply the data transformation when periodicity is possible.

The following proposition justifies the convergence test in step 3 of the algorithm.

Proposition 2.3. *Let $V_i^{(n-1)}$, $i \in S$, be some numbers. Let the policy $R(n)$ be defined as in step 2 of the value iteration algorithm, and the values m_n and M_n be as defined by* (2.40). *Then the average cost $g(R(n))$ of the policy $R(n)$ satisfies*

$$m_n \leq g^* \leq g(R(n)) \leq M_n, \tag{2.41}$$

where g^* denotes the optimal average cost.

Proof. It follows from the construction in step 2 of the value iteration algorithm that

$$c_i(R_n) + \sum_{j \in S} p_{ij}(R(n)_i)\, V_j^{(n-1)} = V_i^{(n)} \quad \text{for all } i \in S. \tag{2.42}$$

Subtracting M_n from the left-hand side and $V_i^{(n)} - V_i^{(n-1)}$ from the right-hand side of the equation, we get the inequality

$$c_i(R_n) - M_n + \sum_{j \in S} p_{ij}(R(n)_i)\, V_j^{(n-1)} \leq V_i^{(n-1)} \quad \text{for all } i \in S. \tag{2.43}$$

By Proposition 2.2 this implies $g(R(n)) \leq M_n$. Analogously, subtracting m_n from the left-hand side of (2.42) and $V_i^{(n)} - V_i^{(n-1)}$ from the right-hand side, we get

$$c_i(R_n) - m_n + \sum_{j \in S} p_{ij}(R(n)_i)\, V_j^{(n-1)} \geq V_i^{(n-1)} \quad \text{for all } i \in S, \tag{2.44}$$

and by Proposition 2.2 this implies $g(R(n)) \geq m_n$.

Since $g^* \leq g(R(n))$ by definition of g^*, it only remains to show that $m_n \leq g^*$. Denoting by R^* an average cost optimal policy, from (2.39) it follows that

$$c_i(R^*) + \sum_{i \in S} p_{ij}(R_i^*)\, V_j^{(n-1)} \geq V_i^{(n)} \quad \text{for all } i \in S. \tag{2.45}$$

Subtracting again m_n from the left-hand side and $V_i^{(n)} - V_i^{(n-1)}$ from the right-hand side, we get

$$c_i(R^*) - m_n + \sum_{i \in S} p_{ij}(R_i^*)\, V_j^{(n-1)} \geq V_i^{(n-1)} \quad \text{for all } i \in S. \tag{2.46}$$

By Proposition 2.2 it follows that $g(R^*) \geq m_n$, and the proof is complete. □

It is worth noting that the bounds of this proposition can also be applied to the average cost rate of any policy produced during the policy iteration algorithm, since the policy improvement step of the policy iteration algorithm chooses the policy via essentially the same minimization as step 2 of the value iteration algorithm.

According to [50] the number of iterations required in the value iteration algorithm varies greatly depending on the problem, and is generally much larger than the number of iterations required in the policy iteration algorithm. However, a policy iteration step requires the solution of a linear system, whereas a value iteration step is a much cheaper operation; thus value iteration is generally a better choice for larger Markov decision problems.

2.3.3 Linear programming

An average cost optimal policy can also be found by linear programming [15], [49]. To keep the problem linear and non-discrete, the actual optimization is performed in the class of stationary randomized policies, which are a generalization of the deterministic stationary policies considered up to this point. A stationary randomized policy specifies for each action $a \in A(i)$ in a state $i \in S$ a probability of choosing that particular action; the random choice of actions is made independently of the history of the system. In the following formulation the decision variable x_{ia} is interpreted as the long-run fraction of time the system spends in state $i \in S$ *and* chooses action $a \in A(i)$, thus capturing a randomized policy and its balance probabilities in the same variables.

The weak unichain assumption needs to be satisfied in order for the linear programming method to work. The method is specified as follows.

1. Solve the linear program in the variables x_{ia}, $i \in S$, $a \in A(i)$

$$\begin{aligned}
&\text{Minimize } \sum_{i \in S} \sum_{a \in A(i)} x_{ia} c_i(a) \\
&\text{subject to} \\
&\sum_{a \in A(i)} x_{ia} = \sum_{j \in S} \sum_{a \in A(j)} x_{ja} p_{ji}(a) \quad \text{for all } i \in S \\
&\sum_{i \in S} \sum_{a \in A(i)} x_{ia} = 1 \\
&x_{ia} \geq 0 \quad \text{for all } i \in S,\ a \in A(i).
\end{aligned} \tag{2.47}$$

2. Construct a deterministic policy R from the optimal x^*_{ia} as follows. For each i such that $\sum_{a \in A(i)} x_{ia} > 0$, choose R_i as an arbitrary a for which $x_{ia} > 0$. Initialize set S_0 with the remaining states:

$$S_0 = \left\{ i \;\middle|\; \sum_{a \in A(i)} x_{ia} = 0 \right\}. \tag{2.48}$$

3. Choose a state $i \in S_0$ such that there is an action $a \in A(i)$ satisfying $p_{ij}(a) > 0$ for some $i \in S \setminus S_0$. Set the action R_i in state i equal to a and remove state i from set S_0. Repeat step 3 until S_0 is empty.

Note that the constraints of the linear program (2.47) are essentially the Markov process equilibrium equations (2.4) applied to the Markov process associated with the randomized policy defined by the x_{ia}'s, and the objective is the average cost of the same policy. Thus the function of step 1 is to find an optimal stationary randomized policy.

The fact that steps 2 and 3 of the algorithm define an average cost optimal deterministic policy can be shown by constructing the dual of the linear program (2.47); we sketch the main ideas of the proof in [50]. The dual linear program can be written in the variables g and v_i, $i \in S$, as

$$\begin{aligned} &\text{Maximize } g \\ &\text{subject to} \\ &c_i(a) - g + \sum_{j \in S} p_{ij}(a)\, v_j \geq v_i \quad \text{for all } i \in S \text{ and } a \in A(i) \\ &g \text{ and } v_i,\ i \in S, \text{ unrestricted.} \end{aligned} \tag{2.49}$$

By Proposition 2.2 we see that the constraints of the dual problem require all stationary policies to have an average cost larger than or equal to g. By duality, the maximum of the dual objective g is equal to the minimum of the primal objective in (2.47), the optimal average cost. By complementary slackness the constraints of the dual problem are tight (equalities) for i and a such that $x_{ia} > 0$. With the help of the weak unichain assumption it is quite straightforward to prove that the initial set S_0 defined in step 2 contains only transient states under the randomized average cost optimal policy found in step 1 as well as under any deterministic policy constructed in steps 2 and 3. It follows that the optimal values of the dual variables satisfy the Howard equations (2.19) in the recurrent set of any deterministic policy constructed by the algorithm, thus guaranteeing average cost optimality.

According to [50] the policy iteration algorithm can be interpreted as a linear programming algorithm that performs pivoting operations on several variables simultaneously, and while the simplex method of linear programming requires more iterations than the policy iteration algorithm, the iterations of the simplex method are computationally cheaper so that the algorithms are in general comparable in performance. However by taking advantage of the specific form of a particular application the policy iteration algorithm can often be made more efficient, and this happens also in the telecommunication application discussed in this thesis (see Section 3.6). On the other hand the linear programming algorithm can easily accommodate additional probabilistic constraints on the constructed policy; for example the frequency of visiting a particular state can be restricted to a specific range.

2.4 Continuous-time Markov decision processes

We define a continuous-time Markov decision process by associating each state i in a finite state space S with a finite set of actions $A(i)$; each time the process enters a new state

$i \in S$ an action $a \in A(i)$ must be chosen. The action a and the state i determine the expected time $\tau_i(a)$ of staying in the state, a steady rate $r_i(a)$ of incurring cost during the sojourn in the state, and probabilities $p_{ij}(a)$ of making a transition to each state $j \in S$ after the sojourn in state i. As in a continuous-time Markov process, the random state sojourn times are exponentially distributed and independent of the history of the system, and the transitions to new states are also independent of the history of the system. Instead of the transition probabilities and state sojourn times, we shall mostly work with the transition rates defined by

$$q_{ij}(a) = \begin{cases} p_{ij}(a)/\tau_i(a) & \text{if } i \neq j, \\ -1/\tau_i(a) & \text{if } i = j, \end{cases} \qquad \text{for all } a \in A(i),\ i, j \in S. \tag{2.50}$$

The above definition of a continuous-time Markov decision process is restricted similarly to the definition of a continuous-time Markov process in Section 2.2 in order to avoid pathological cases arising in more general definitions. The fact that we allow only discrete control, one decision per each state visited, instead of continuous-time control makes the formulation close to so-called semi-Markov decision processes [18, Section 5.1], which essentially generalize the present class of continuous-time Markov decision models by allowing arbitrarily distributed state sojourn times.

Exactly as in the discrete-time case, a stationary control policy is a control rule that specifies the action as a deterministic function of the current state; we denote by R_i the action of policy R in state $i \in S$. We restrict our attention to stationary control policies only.

A continuous-time Markov decision process running under a stationary policy R can be considered a continuous-time Markov process with a cost structure. We denote the state of the Markov process at time $t \geq 0$ by $I^R(t)$, the infinitesimal generator matrix of the Markov process by $Q(R)$ and the vector of cost rates in each state by $\mathbf{r}(R)$.

Suppose that the Markov process associated with policy R has a single irreducible set, and consider an arbitrary realization of the process where the process has spent the times $T_i \geq 0$, $i \in S$, in each state by the time $t = \sum_{i \in S} T_i$. Since the process accumulates cost at a steady rate during each state sojourn, it follows that the total accumulated cost is given by $\sum_{i \in S} T_i r_i(R_i)$. The following proposition is then a direct consequence of the results about long-run behaviour of continuous-time Markov processes discussed in section 2.2.

Proposition 2.4. *Let R be a stationary policy for a continuous-time Markov decision process on the finite state space S, and suppose that the associated Markov process has a single irreducible set. Then the long-run average cost per unit time, briefly the* average cost rate, *is*

$$g(R) = \boldsymbol{\pi}(R)^{\mathrm{T}} \mathbf{r}(R) \quad \textit{with probability 1} \tag{2.51}$$

regardless of the initial state; here $\boldsymbol{\pi}(R)$ is the vector of balance probabilities of the Markov process.

As in the discrete-time case, a policy R^* is called average cost optimal if it satisfies $g(R^*) \leq g(R)$ for all stationary policies R. To be able to take advantage of Proposition 2.4 we shall make use of the continuous-time analogues of the unichain assumption and the weak unichain assumption, as stated on pages 8 and 12.

The development of relative values parallels the discrete-time case. Let us assume that the Markov process associated with policy R has a single irreducible set. We define the expected cost function $V_i(T, R)$, $T \geq 0$, as the expected value of the total accumulated cost by time T when the initial state is i and the system is controlled by policy R, and we denote

the vector of these expected costs by $\mathbf{V}(T,R) = (V_i(T,R))_{i\in S}$. By conditioning on the state visited by the system on each moment of time we get

$$\mathbf{V}(T,R) = \int_0^T e^{tQ(R)}\mathbf{r}(R)\,dt, \quad T > 0. \tag{2.52}$$

By assumption the Markov process has unique balance probabilities $\boldsymbol{\pi}(R)$, and we apply equation (2.12) to deduce

$$\lim_{T\to\infty}\frac{1}{T}\mathbf{V}(T,R) = \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{r}(R) = g(R)\mathbf{1}. \tag{2.53}$$

We define the vector of relative values $\mathbf{w} = (w_i(R))_{i\in S}$ of the system states by

$$\mathbf{w}(R) = \lim_{T\to\infty}\mathbf{V}(T,R) - Tg(R)\mathbf{1}. \tag{2.54}$$

The limit vector exists and its elements are finite by the exponential convergence in (2.13); we do not go through the details here, but the absolute convergence of (2.54) is shown in the course of the proof of Theorem 2.5 below.

We now introduce a transformation based on the uniformization method discussed in Section 2.2 that transforms a continuous-time Markov decision process into a discrete-time Markov decision process with very similar properties. The discrete-time model has the same state space S and the same action choices $A(i)$, $i \in S$, as the continuous-time model. We distinguish the other parameters of the discrete-time model by bars over the symbols. Let τ be a constant such that $0 < \tau < \min_{i\in S,\, a\in A(i)} \tau_i(a)$. The costs and transition probabilities in the discrete-time model are defined as

$$\bar{p}_{ij}(a) = \tau q_{ij}(a), \quad i \neq j,\ i,j \in S,\ a \in A(i), \tag{2.55}$$

$$\bar{p}_{ii}(a) = 1 - \frac{\tau}{\tau_i(a)}, \quad i \in S,\ a \in A(i), \tag{2.56}$$

$$\bar{c}_i(a) = r_i(a), \quad i \in S,\ a \in A(i). \tag{2.57}$$

Observe that the equivalence of the state and action spaces means that any stationary policy of the continuous-time model can be used equally well in the discrete-time model. The transformation applies the uniformization method simultaneously to the continuous-time Markov processes $I^R(t)$ associated with each and every stationary policy R, so that the discrete-time Markov process $\bar{I}^R(t)$ associated with policy R in the transformed discrete-time model is exactly the embedded Markov process of the uniformization of $I^R(t)$. Thus the matrix $\bar{P}(R)$ of transition probabilities in the discrete-time process $\bar{I}^R(t)$ and the infinitesimal generator matrix $Q(R)$ of the continuous-time process $I^R(t)$ are related by

$$\bar{P}(R) = \tau Q(R) + I \quad \text{for any stationary policy } R; \tag{2.58}$$

this is basically a reformulation of equations (2.55)–(2.56) in matrix form. As already shown in Section 2.2, the discrete-time processes and continuous-time processes have exactly the same equilibrium distributions. Note also that the choice of τ as strictly smaller than the minimal $\tau_i(a)$ ensures that the discrete-time process $\bar{I}^R(t)$ is aperiodic.

Intuitively, the transformation compensates for the different state sojourn times of the continuous-time model by proportionally adjusting the probability of self-transitions $\bar{p}_{ii}$ in the discrete-time model, so that the probability of making self-transitions is small in "fast" states and large in "slow" states. The following theorem establishes the importance of the transformation.

Theorem 2.5. *Suppose that R is a stationary control policy for a continuous-time Markov decision process, such that the associated Markov process $I^R(t)$ has a single irreducible set. Then the transformed discrete-time process $\bar{I}^R(t)$ also has a single irreducible set, and the average cost rate $g(R)$ of policy R in the continuous-time process is equal to the average cost $\bar{g}(R)$ of policy R in the discrete-time process. Further, the relative values $\mathbf{w}(R)$ of the continuous-time process are related to the relative values $\bar{\mathbf{w}}(R)$ of the discrete-time process by*

$$\mathbf{w}(R) = \tau \bar{\mathbf{w}}(R). \tag{2.59}$$

Proof. Because of the equivalence of the equilibrium distributions of the processes $I^R(t)$ and $\bar{I}^R(t)$, they both must have a unique equilibrium distribution $\boldsymbol{\pi}(R) = (\pi_i(R))_{i\in S}$. Since a finite-state Markov process that has more than one irreducible set also has more than one equilibrium distribution, both processes must have a single irreducible set.

The equivalence of $g(R)$ and $\bar{g}(R)$ follows from the identity

$$g(R) = \boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{r}(R) = \boldsymbol{\pi}(R)^{\mathrm{T}}\bar{\mathbf{c}}(R) = \bar{\mathbf{g}}(R). \tag{2.60}$$

To relate the relative values $\mathbf{w}(R)$ and $\bar{\mathbf{w}}(R)$ we shall derive (2.59) from the definition (2.54) of relative values in the continuous-time process. By expanding the definition we get

$$\begin{aligned}
\mathbf{w}(R) &= \lim_{T\to\infty} \int_0^T e^{tQ(R)}\mathbf{r}(R)\,dt - T\mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{r}(R) \\
&= \int_0^\infty e^{t\frac{1}{\tau}(P(R)-I)}\mathbf{r}(R) - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{r}(R)\,dt \\
&= \int_0^\infty e^{-t/\tau}\sum_{n=0}^\infty \frac{(t/\tau)^n}{n!}P(R)^n\mathbf{r}(R) - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\mathbf{r}(R)\,dt \\
&= \int_0^\infty \sum_{n=0}^\infty e^{-t/\tau}\frac{(t/\tau)^n}{n!}\big(P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big)\,dt\;\mathbf{r}(R).
\end{aligned} \tag{2.61}$$

To justify swapping the integral and iterated sum, we show that (2.61) converges absolutely. Since the process has a single irreducible set, $P(R)^n$ converges elementwise to $\mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}$ at a geometric rate, ie there are constants $\alpha > 0$ and $0 < \beta < 1$ such that

$$\big|[P(R)^n - \mathbf{1}\pi(R)^{\mathrm{T}}]_{i,j}\big| < \alpha\beta^n \quad \text{for all } i,j \in S \text{ and } n \in \mathbb{N}. \tag{2.62}$$

By this relation we have

$$\begin{aligned}
\int_0^\infty \sum_{n=0}^\infty \left| e^{-t/\tau}\frac{(t/\tau)^n}{n!}\big[P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big]_{i,j}\right| dt &< \int_0^\infty \sum_{n=0}^\infty e^{-t/\tau}\frac{(t/\tau)^n}{n!}\alpha\beta^n\,dt \\
&= \alpha\int_0^\infty e^{-t/\tau}\sum_{n=0}^\infty \frac{(t\beta/\tau)^n}{n!}\,dt = \alpha\int_0^\infty e^{-t/\tau}e^{t\beta/\tau}\,dt \\
&= \alpha\int_0^\infty e^{t(\beta-1)/\tau}\,dt = \alpha\frac{\tau}{\beta-1} \quad \text{for all } i,j \in S,
\end{aligned} \tag{2.63}$$

thus proving that all elements of the integral of the sum of matrices in (2.61) converge absolutely. We continue the development by swapping the integral and iterated sum in (2.61).

$$\begin{aligned}
\mathbf{w}(R) &= \sum_{n=0}^\infty \int_0^\infty e^{-t/\tau}\frac{(t/\tau)^n}{n!}\big(P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big)\,dt\;\mathbf{r}(R) \\
&= \sum_{n=0}^\infty \left(\frac{1}{n!}\int_0^\infty e^{-t/\tau}(t/\tau)^n\,dt\;\big(P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big)\right)\mathbf{r}(R).
\end{aligned} \tag{2.64}$$

Now by making the change of variable $u = t/\tau$ we can evaluate the integral with the help of the gamma function as

$$\int_0^\infty e^{-t/\tau}(t/\tau)^n \, dt = \int_0^\infty e^{-u} u^n \tau \, du = \tau\Gamma(n+1) = \tau n! \tag{2.65}$$

and further simplify (2.64) into

$$\begin{aligned}\mathbf{w}(R) &= \sum_{n=0}^{\infty} \left(\frac{1}{n!}\tau n! \big(P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big) \right) \mathbf{r}(R) \\ &= \tau \sum_{n=0}^{\infty} \big(P(R)^n - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}}\big)\, \mathbf{r}(R) = \tau \lim_{t\to\infty} \sum_{n=0}^{t-1} \big(P(R)^n \mathbf{r}(R) - \mathbf{1}\boldsymbol{\pi}(R)^{\mathrm{T}} \mathbf{r}(R)\big) \\ &= \tau \lim_{t\to\infty} \sum_{n=0}^{t-1} P(R)^n \mathbf{r}(R) - t\mathbf{1}g(R) = \tau \bar{\mathbf{w}}(R) \end{aligned} \tag{2.66}$$

where the last identity is by the definition (2.18) of discrete-time relative values. This completes the proof. □

By Theorem 2.5 we can now apply the policy evaluation and optimization results developed for discrete-time Markov decision processes in Section 2.3 to the transformed discrete-time model, and the results will hold equally well in the continuous-time model. Moreover, as a rule the transformation parameter τ can be eliminated when the discrete-time results are expressed in terms of the continuous-time model parameters; this is because the same results can be derived directly in the continuous-time model by reasoning analogous to that in Section 2.3. The only exception is value iteration, for which the direct continuous-time development leads to a system of differential equations, the solution of which is not straightforward.

To establish the continuous-time analogue of the Howard equations, we restate Theorem 2.1 in the continuous-time model.

Proposition 2.6. *Let R be a stationary policy for a continuous-time Markov decision process on the finite state space S, and let the associated Markov process have a single irreducible set. Then all the solutions of the linear system of equations in the variables g and $\mathbf{v} = (v_i)_{i\in S}$*

$$\mathbf{r}(R) - g\mathbf{1} + Q(R)\mathbf{v} = \mathbf{0} \tag{2.67}$$

are given by

$$g = g(R), \tag{2.68}$$

$$\mathbf{v} = \mathbf{w}(R) + b\mathbf{1}, \tag{2.69}$$

where b is an arbitrary constant, $g(R)$ is the average cost rate of policy R, and $\mathbf{w}(R)$ is the vector of relative values of the Markov decision process run under policy R.

Proof. Let us rewrite (2.67) equivalently in terms of the transformed discrete-time transition probability matrix $\bar{P}(R)$ as

$$\mathbf{r}(R) - g\mathbf{1} + \frac{1}{\tau}\big(\bar{P}(R) - I\big)\mathbf{v} = \mathbf{0}. \tag{2.70}$$

By the change of variable $\bar{\mathbf{v}} = \frac{1}{\tau}\mathbf{v}$, this is equivalent to

$$\mathbf{r}(R) - g\mathbf{1} + \bar{P}(R)\bar{\mathbf{v}} = \bar{\mathbf{v}}, \tag{2.71}$$

but this is the system of Howard equations for the transformed discrete-time model. By Theorem 2.1 we have $g = \bar{g}(R)$ and $\bar{\mathbf{v}} = \bar{\mathbf{w}}(R) + \bar{b}\mathbf{1}$ for an arbitrary constant $\bar{b}$. By Theorem 2.5 equations (2.68) and (2.69) now follow, with the constant b given by $b = \tau\bar{b}$, and by Theorem 2.1 there are no other solutions since (2.71) is equivalent to (2.67). The proof is thus complete. □

Here Theorem 2.5 allowed a proof far simpler than could be attained directly in the continuous-time model by reasoning analogous to the proof of Theorem 2.1.

The continuous-time Howard equations (2.67) can be interpreted physically as follows. Let us first expand the equations elementwise as

$$r_i(R_i) - g + \sum_{j \neq i} q_{ij}(R_i)\,(v_j - v_i) = 0 \quad \text{for all } i \in S. \tag{2.72}$$

Multiplying each equation by $\tau_i(R_i)$, we can express the equations in terms of the transition probabilities p_{ij} of the continuous-time model as

$$r_i(R_i)\tau_i(R_i) - g\tau_i(R_i) + \sum_{j \neq i} p_{ij}(R_i)\,v_j = v_i \quad \text{for all } i \in S. \tag{2.73}$$

Recalling the definition of relative value as the expected future cost in addition to the average cost rate, the continuous-time Howard equations in the form (2.73) can now be interpreted as equating the relative value of state i with the expected cost in addition to the average cost rate incurred during the sojourn in state i, plus the additional relative value accumulated after the first transition.

2.4.1 Policy iteration

The continuous-time equivalent of Proposition 2.2 can be stated as follows.

Proposition 2.7. *Suppose that R is a stationary control policy for a continuous-time Markov process on a finite state space S, and suppose that the Markov process associated with policy R has a single irreducible set. Let g be some arbitrary number and $\mathbf{v} \in \mathbb{R}^{|S|}$ be an arbitrary vector. Suppose that*

$$\mathbf{r}(R) - g\mathbf{1} + Q(R)\mathbf{v} \leq \mathbf{0}. \tag{2.74}$$

Then the average cost $g(R)$ of policy R satisfies $g(R) \leq g$. Further, $g(R) < g$ if and only if strict inequality holds in (2.74) *for some state that is recurrent under policy R.*

The above statements are true also with the directions of all the inequality signs reversed.

We omit the proof, which is obvious given Theorem 2.5 and Proposition 2.2.

When the inequality (2.74) is multiplied by the diagonal matrix $\mathcal{T}(R)$, the diagonal elements of which are given by $\tau_i(R_i)$, we get the equivalent inequality

$$\mathcal{T}(R)\mathbf{r}(R) - g\boldsymbol{\tau}(R) + P(R)\mathbf{v} \leq \mathbf{v}, \tag{2.75}$$

where $P(R)$ is the matrix of transition probabilities in the continuous-time model, and $\boldsymbol{\tau}(R) = (\tau_i(R_i))_{i \in S}$ is the vector of state sojourn times under policy R. The form (2.75) of

the inequalities is analogous to the form (2.73) of Howard equations, and can be interpreted similarly.

For reference we state the policy iteration algorithm in the continuous-time model. To begin with, choose an arbitrary policy R as a starting point for the iteration.

1. *(value determination step)* Solve g and $\mathbf{v}$ from the augmented Howard equations

$$\mathbf{r}(R) - g\mathbf{1} + Q(R)\mathbf{v} = 0, \tag{2.76}$$

$$v_m = 0, \tag{2.77}$$

   where $m \in S$ is an arbitrary state.

2. *(policy improvement step)* Construct an improved policy R' as follows. For each state $i \in S$, find an action $a \in A(i)$ that minimizes the expression

$$r_i(a) - g + \sum_{j \in S} q_{ij}(a)\, v_j. \tag{2.78}$$

   If the minimal value of the expression is reached by the action R_i of the previous policy, choose that action as the decision of the new policy. Otherwise, choose any $a \in A(i)$ that minimizes the expression.

3. *(convergence test)* If the new policy R' differs from the previous policy R, start over from step 1 with the new policy assigned to R.

As is apparent from the specification of the algorithm, the final policy R^* will satisfy the so-called average cost optimality inequalities

$$r_i(a) - g(R^*) + \sum_{j \in S} q_{ij}(a)\, w_j(R^*) \geq 0 \quad \text{for all } a \in A(i) \text{ and } i \in S, \tag{2.79}$$

which guarantee the average cost optimality of policy R^* by Proposition 2.7. Of course, by Theorem 2.5 we can apply the convergence proof in the discrete-time case to conclude that under the unichain assumption the policy iteration algorithm converges to an average cost optimal policy in a finite number of steps.

We remark that the continuous-time policy iteration algorithm has an alternative formulation which will also result an average cost optimal policy. By multiplying equation (2.79) by $\tau_i(a)$, we get an alternative equivalent form of the average cost optimality inequalities:

$$r_i(a)\tau_i(a) - g(R^*)\tau_i(a) + \sum_{j \in S} p_{ij}(a)\, w_j(R^*) \geq 0 \quad \text{for all } a \in A(i) \text{ and } i \in S. \tag{2.80}$$

Accordingly, the policy improvement step of the policy iteration algorithm can be modified to minimize the expression

$$r_i(a)\tau_i(a) - g\tau_i(a) + \sum_{j \in S} p_{ij}(a)\, v_j, \tag{2.81}$$

instead of (2.78). Note that the modified algorithm is not identical to the version presented above—the minimization of (2.81) is not as such equivalent to the minimization of (2.78). Nevertheless it is possible to replace $g(R)\mathbf{1}$ by $g(R)\boldsymbol{\tau}(R)$ in the convergence proof of the discrete-time policy iteration to show that the modified algorithm converges in a finite number of iterations, and the form (2.80) of the average cost optimality inequalities ensures the average cost optimality of the final policy. This alternative formulation is more natural in semi-Markov decision models discussed in [50, Section 3.5]; in the telecommunication application of this thesis it is simpler to use our primary formulation of the policy iteration algorithm.

2.4.2 Value iteration

The value iteration algorithm can be applied in continuous-time Markov processes by first transforming the process into a discrete-time process. In terms of the parameters of the continuous-time process and the transformation parameter τ, the algorithm can be specified as follows.

1. Let ε be a given tolerance. Set the initial relative value estimates $V_i^{(0)}$, $i \in S$, equal to zero. Initialize the iteration counter n as 1.

2. Compute new value estimates by

$$V_i^{(n)} = \min_{a \in A(i)} \left\{ \tau r_i(a) + \tau \sum_{j \in S} q_{ij}(a)\, V_j^{(n-1)} + V_i^{(n-1)} \right\}, \quad \text{for all } i \in S. \quad (2.82)$$

   Define policy $R(n)$ as follows: The decision in state $i \in S$ is arbitrarily chosen as one of the actions $a \in A(i)$ that reach the minimal value on the right-hand-side of (2.39) for state i.

3. Define the bounds

$$m_n = \frac{1}{\tau} \min_{i \in S} V_i^{(n)} - V_i^{(n-1)} \quad \text{and} \quad M_n = \frac{1}{\tau} \max_{i \in S} V_i^{(n)} - V_i^{(n-1)}. \quad (2.83)$$

   If $(M_n - m_n)/m_n \geq \varepsilon$, return to step 2 with the iteration counter n incremented by one. Otherwise, stop the iteration with the policy $R(n)$.

The algorithm produces a policy with an average cost rate that differs from the optimal average cost rate g^* by no more than εg^*.

Note that in straightforward application of value iteration in the transformed discrete-time model, instead of equation (2.82) we have

$$\bar{V}_i^{(n)} = \min_{a \in A(i)} \left\{ r_i(a) + \tau \sum_{j \in S} q_{ij}(a)\, \bar{V}_j^{(n-1)} + \bar{V}_i^{(n-1)} \right\}, \quad \text{for all } i \in S. \quad (2.84)$$

Equation (2.82) follows from this by substituting $\bar{V}_i^{(n)} = \frac{1}{\tau} V_i^{(n)}$ for all $i \in S$, and multiplying both sides by τ. The factor $1/\tau$ in (2.83) compensates for this substitution. The effect of the change is that the values $V_i^{(n)}$ approximate the optimal relative values in the continuous-time process as n approaches infinity. To see this, note that when the value iteration algorithm terminates we have for the value vector $\mathbf{V}^{(n)} = (V_i^{(n)})_{i \in S}$ that

$$\mathbf{V}^{(n)} - \mathbf{V}^{(n-1)} = \tau g^* (\mathbf{1} + \boldsymbol{\delta}), \quad (2.85)$$

where $\|\delta\|_\infty \leq \varepsilon$. By substituting $\mathbf{V}^{(n-1)}$ solved from (2.85) into (2.82) and noting that $Q(R(n))\mathbf{1} = \mathbf{0}$, we get

$$\tau \mathbf{r}(R(n)) - \tau g^* \mathbf{1} + \tau Q(R(n)) \mathbf{V}^{(n)} = \big(\tau Q(R(n)) + I\big) g^* \boldsymbol{\delta}. \quad (2.86)$$

As $\|\delta\|_\infty$ approaches zero when n tends to infinity, and $\tau Q(R(n)) + I$ is stochastic for all n, at the limit $\mathbf{V}^{(n)}$ satisfies the Howard equations of the continuous-time process.

As a continuous-time analogue of Proposition 2.3, we state the following proposition which is formulated independently of the value iteration algorithm, so that it can be applied to the results of the policy improvement step of the policy iteration algorithm later on.

Proposition 2.8. *Let $\tilde{v}_i$, $i \in S$, be some numbers. Let Δ_i, $i \in S$, be given by*

$$\Delta_i = \min_{a \in A(i)} \left\{ r_i(a) + \sum_{j \in S} q_{ij}(a)\, \tilde{v}_j \right\} \quad \textit{for all } i \in S, \tag{2.87}$$

and let the policy R be defined so that the action R_i in each state $i \in S$ is one of the actions $a \in A(i)$ reaching the minimum on the right-hand side of (2.87)*. Then the average cost rate $g(R)$ of the policy R satisfies*

$$m_n \le g^* \le g(R) \le M_n, \tag{2.88}$$

where g^* denotes the optimal average cost rate and m_n and M_n are defined by

$$m_n = \min_{i \in S} \Delta_i \quad \textit{and} \quad M_n = \max_{i \in S} \Delta_i. \tag{2.89}$$

Proof. The definitions of the policy R and the bounds m_n and M_n in the proposition are equivalent to those in the value iteration algorithm as specified above. Thus the result follows by applying Proposition 2.3 with the values of the preceding step given by $\bar{V}_i^{n-1} = \frac{1}{\tau}\tilde{v}_i$ for all $i \in S$. □

Chapter 3

Application Background

We introduce the basics of traditional teletraffic theory and a natural generalization for modern broadband networks. For a more comprehensive introduction to issues in broadband networks, including variations and alternatives to the approach followed here, the reader is referred to the book [40] by Ross. Additionally, Dziong [9] provides a more involved treatment of some theoretical and practical issues related to the subject of the thesis.

3.1 Basics of teletraffic theory

We begin in the basic single-service setting in which all calls are considered identical in characteristics; the theory was pioneered by A. K. Erlang (1878–1929) in the beginning of the 20th century, and has been widely and successfully applied to traditional telephone networks.

A telecommunication network is assumed to consist of *nodes* connected by *links*. Each link consists of a fixed number of *trunks*, each of which can carry a single call between the endpoints of the link. To introduce the basic concepts of teletraffic theory, let us concentrate on the behaviour of a single link, disregarding other links in the network for the while.

The arrival of new calls to a link is modelled as a random process. The number of call arrivals since an arbitrary fixed starting epoch is assumed to be a Poisson process; the assumption has been verified to hold very accurately for voice calls in traditional telephone networks on relatively short time scales, but on the scale of hours there is definite variation in the arrival intensity according to business hours etc. The accuracy of the Poisson assumption can be explained by the fact that when there are a large number of individual events each happening by a small probability, the total number of events occurring forms approximately a Poisson process; here the individual events would be potential callers making a call via the particular link. Let λ denote the Poisson parameter. The Poisson assumption implies that the inter-arrival times between subsequent calls are exponentially distributed as $\mathrm{Exp}(\lambda)$ and independent of each other.

In practice calls do not come and go instantaneously but it takes some time for a new connection to be established and for a connection to be removed. In the abstract link model these extra times are combined with the actual call duration to form the *call holding time*, which is treated as a random variable depicting the total length of the time a single call occupies link capacity. For now the holding time of each call is assumed to be exponentially distributed as $\mathrm{Exp}(\mu)$ with the expected holding time $1/\mu$, and independent of all other calls. In practice this assumption is not perfectly accurate but sufficient, and as it happens some key results apply to arbitrary call holding time distributions. The inverse μ of the call holding time can be seen as a *call completion rate*, which is not an entirely accurate char-

acterization considering that the call completions do not form a time-homogenous Poisson process, but this is a convenient point of view in some considerations later on.

The assumptions of Poisson arrivals and exponential call holding times imply that the state of the link system at an arbitrary moment of time is completely characterized by the number of ongoing calls: by the memoryless property of the exponential distribution the elapsed times since the arrivals of the current calls have no bearing on the future development of the system. Let C be the link capacity, ie the maximum number of ongoing calls, and let $I(t) \in \{0, 1, \ldots, C\}$ denote the link state, interpreted as the number of ongoing calls, at time $t \geq 0$ after some arbitrary start epoch. Now suppose that incoming calls are carried whenever possible, that is new calls are accepted when $I(t) < C$ and blocked when $I(t) = C$. Blocked calls are dropped immediately without any kind of queueing, and we do not make any provisions for modelling possible retries by callers whose calls are blocked. A link or a network where blocked calls are immediately lost is called a *loss system*; the opposite, a system where blocked calls are queued until they can be connected is called a *delay system* but these are outside the scope of this treatment.

Under the present assumptions the link state process $I(t)$, $t \geq 0$, is a continuous-time Markov process[1] on the state space $S = \{0, 1, \ldots, C\}$ with the transition rates

$$q_{ij} = \begin{cases} \lambda & \text{if } j = i + 1, \\ i\mu & \text{if } j = i - 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

The transition rates to lower link occupation are as presented because the minimum of i independent $\mathrm{Exp}(1/\mu)$ random variables is distributed as $\mathrm{Exp}(1/i\mu)$. As is easily verified, the balance probabilities of the process are given by

$$\pi_i = \frac{A^i/i!}{\sum_{j=0}^{C} A^j/j!}, \quad i = 0, 1, \ldots, C, \tag{3.2}$$

where $A = \lambda/\mu$.

The quantity *traffic* is defined as the long-run average number of ongoing connections. The quantity is dimensionless, but the unit *erlang* is often used for clarity. *Offered traffic* is the amount of traffic that would be carried were there no capacity restrictions. By Little's formula $A = rs$ where r is the long-run average call arrival rate and s is the long-run average call holding time, and this holds regardless of the distributions of the inter-arrival times and call holding times. In particular, in (3.2) above the quantity $A = \lambda/\mu$ is the offered traffic.

The performance of a link system is commonly measured by different kinds of long-run average blocking probabilities: *Time blocking* is the proportion of time that the system is in any blocking state. *Call blocking* is the proportion of arriving calls that are blocked. *Traffic blocking* is the proportion of the offered traffic that is blocked. In general these blocking probabilities are not equivalent, but when call arrivals form a Poisson process, call blocking and time blocking are equal by the "Poisson arrivals see time averages" property. Technically the blocking probabilities measure *grade of service*, the degree to which requirements on the acceptance of incoming calls are satisfied. A separate issue outside the scope of this thesis is *quality of service*, how well the system satisfies requirements on the quality of individual connections.

In the single link system under consideration time blocking (and consequently call blocking) is simply the balance probability of state C, as given by (3.2). By computing

[1] More precisely, the link state forms a birth-death process.

traffic blocking as $1 - \mathrm{E}[I(t)]/A$ one may verify that it is equal to the time blocking probability as well. This single link blocking probability, considered as a function of the link capacity C and the offered traffic A is called the *Erlang-B* function and denoted by $E(C, A)$. As per (3.2) the function is defined as

$$E(n, A) = \frac{A^n/n!}{\sum_{i=0}^{n} A^i/i!}. \tag{3.3}$$

It can be shown [24] that the Erlang-B function gives the correct time, call and traffic blocking regardless of the distribution of the call holding times, as long as the link functions as a loss system and calls arrive as a Poisson process. This is called the *insensitivity* property.

3.2 Modelling a multiservice link

In modern broadband digital networks, connections may be for example traditional voice calls, telefax calls, real-time video connections, remote terminal access, bulk file transfers, web browser data connections, virtual network links providing a "network in network" for some specific purpose, etc. The different types of calls vary so much in their frequency and network usage characteristics that the Erlang link model of Section 3.1 is as such no longer useful. Research on multiservice models only become popular during the 1980's as new networking technologies began to be deployed; since then the field has expanded considerably. Current practical and research interest is largely focused on the standardized ATM (Asynchronous Transfer Mode) networking technology, but on the level of this treatment technical details are of no consequence.

The single-service Erlang link model can be naturally generalized by supposing a finite set $\mathcal{K}$ of *traffic classes* each with their own characteristics. Not only do the traffic classes have different call arrival rates and call holding times, but also the number of trunks occupied by every connection of a particular class is allowed to be an arbitrary integer smaller than the link capacity. In the multiservice case a trunk most often is an artificial unit of link bandwidth concocted purely for modelling purposes; in fact there are good reasons to want the model to allow more flexible division of bandwidth among connections [40], but such requirements do not fit the model under discussion. We assume that there are no constraints as to which particular trunks be used to carry a multi-trunk connection.

We continue with the basic assumptions that calls of each traffic class arrive as a Poisson process and that call holding times are exponentially distributed and independent of other calls (although the insensitivity property does apply in many cases). In broadband networks these assumptions are not as accurate as in traditional telephone networks, but they are "good enough" on short time scales on the order of minutes, and compared to some alternative approaches the generalized Erlang model does have the advantage of being mathematically quite tractable. In practice one may want to model the call arrival rates and call holding times as functions of link state for a number of reasons [9]. This extension can be accommodated to various degrees in many computational and theoretical methods developed for the basic generalized Erlang model; we shall discuss the necessary modifications on several occasions.

For a traffic class $k \in \mathcal{K}$ we denote the call arrival rate by λ_k, the expected call holding time by $1/\mu_k$, the offered traffic of class k by $A_k = \lambda_k/\mu_k$ and the number of trunks required to carry a connection by b_k. We shall also make use of the vector $\mathbf{b} = (b_k)_{k\in\mathcal{K}}$. By the assumptions of Poisson call arrivals and exponential call holding times it is sufficient to know the number of ongoing calls of each traffic class to completely characterize the link state at an arbitrary moment of time. We identify link states with vectors $\mathbf{i} = (i_k)_{k\in\mathcal{K}} \in$

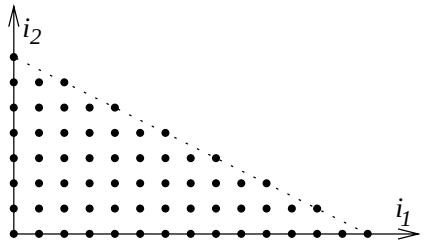

Figure 3.1: The state space of a link of capacity $C = 14$, with two traffic classes. Connections of traffic class 1 require one trunk and connections of traffic class 2 require two trunks. The states on the dotted line have all the 14 trunks occupied.

$\mathbb{N}^K$, where K is the number of traffic classes, so that the number of ongoing connections of traffic class $k \in \mathcal{K}$ in state $\mathbf{i}$ is given by i_k. Denoting the link capacity by C, the complete state space Ω of the multiservice link is

$$\Omega = \big\{\mathbf{i} \in \mathbb{N}^K \;\big|\; \mathbf{i}^{\mathrm{T}}\mathbf{b} \leq C\big\}. \tag{3.4}$$

Observe that the expression $\mathbf{i}^{\mathrm{T}}\mathbf{b}$ gives the number of occupied trunks in state $\mathbf{i}$. To visualize (3.4), Figure 3.1 shows an example of a simple link state space. As apparent from (3.4), the number of system states grows exponentially in the number of traffic classes. In practical link control problems with C on the order of 100 or more and K something like 5 to 10 or even more, the number of system states is enormous.

3.2.1 Link control as a Markov decision problem

On a solitary single-service link it makes no sense to refuse arriving calls before all trunks are in use, but this is not the case on a multi-service link where we have several interwoven arrival processes and blocking some of them in some system states does not prevent the link capacity from being fully utilized, and the average link utilization may in fact be improved. The problem of deciding which calls to accept and which ones to deny is referred to as *connection admission control* (CAC); the problem is a natural application for Markov decision models. We proceed to specify an appropriate continuous-time Markov decision process on the link state space Ω; while some essentially equivalent alternative Markov decision formulations are possible we present the two very similar formulations that have become established in the literature. The link is treated as a loss system.

We identify an action in a state with the subset of traffic classes that are accepted in the state; when the action is chosen, arriving calls of any remaining traffic classes are immediately lost. The set of available actions in state $\mathbf{i}$ is now given by

$$A(\mathbf{i}) = \Big\{a \subseteq \mathcal{K} \;\Big|\; \mathbf{i}^{\mathrm{T}}\mathbf{b} \leq C - b_k \text{ for all } k \in a\Big\}, \quad \mathbf{i} \in \Omega, \tag{3.5}$$

where the constraint ensures that there is sufficient free capacity for any call that is to be accepted. The transition rates under different actions are given by

$$q_{\mathbf{ij}}(a) = \begin{cases} \lambda_k & \text{if } \mathbf{j} = \mathbf{i} + \mathbf{e}_k \text{ for some } k \in a, \\ i_k\mu_k & \text{if } \mathbf{j} = \mathbf{i} - \mathbf{e}_k \text{ for some } k \in \mathcal{K}, \\ -\big(\sum_{k\in\mathcal{K}} i_k\mu_k + \sum_{k\in a}\lambda_k\big) & \text{if } \mathbf{j} = \mathbf{i}, \text{ and} \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

Incurring cost is defined as a weighted sum of the arrival rates of denied traffic classes; the weights are very useful in practical applications of the model [9]. We denote the relative

weight of a connection of traffic class $k \in \mathcal{K}$ by h_k, and define the incurring cost rate under action a in state $\mathbf{i}$ by

$$r_{\mathbf{i}}(a) = \sum_{k \in \mathcal{K} \setminus a} \lambda_k h_k, \quad a \in A(\mathbf{i}),\ \mathbf{i} \in \Omega. \tag{3.7}$$

Note that this depends on the state $\mathbf{i}$ only indirectly to the extent that the action a is state-dependent. By the "Poisson Arrivals See Time Averages" property the long-run average cost computed by Proposition (2.4) for the multiservice link model can be interpreted as arising from the following simple definition of incurring cost: each denied call incurs a cost (revenue loss) of exactly h_k units where k is the traffic class of the denied call.

The alternative established Markov decision formulation is otherwise identical to the above, but instead of minimizing incurred costs, a somewhat different kind of accumulated reward is maximized. In the alternative formulation the reward rate in state $\mathbf{i}$ under any action is usually defined as

$$\sum_{k \in \mathcal{K}} i_k \mu_k h_k, \tag{3.8}$$

that is the weighted average rate of calls completing in state $\mathbf{i}$. This obviously is not simply the negative of the cost rate (3.7), and consequently the relative values in the two formulations are not directly related. Nevertheless, maximizing the average reward rate induced by the rewards (3.8) is equivalent to minimizing the average cost rate induced by the costs (3.7). Informally, the equivalence follows from the observation that under an arbitrary policy, the average reward rate is the rate of accumulating revenue, and the average cost rate is the rate of losing revenue, and the sum of the average reward and cost rates is the offered revenue rate $\sum_{k \in \mathcal{K}} \lambda_k h_k$ which is a constant independent of policy. A formal proof can be constructed by proving from the equilibrium equations that the average reward rate induced by (3.8) is equal to the average reward rate induced by rewards of the form $\sum_{k \in a} \lambda_k h_k$. In the sequel we concentrate on the cost based Markov decision formulation.

Under any stationary control policy the Markov decision process forms an ordinary continuous-time Markov process; we denote the Markov process associated with R by I^R and the state of the Markov process at time $t \geq 0$ after an arbitrary start epoch by $I^R(t) \in \Omega$. We note that state $\mathbf{0}$ with no ongoing connections at all is recurrent regardless of policy, since in all other states there are always transitions to lower link occupations, corresponding to a connection finishing before the next call acceptance epoch. It follows that for all stationary policies R the Markov process I^R has a single irreducible set.

There are some simple policies that come up often enough to be named. The naive policy of always accepting all calls that can possibly be carried is known as the *complete sharing* policy, often abbreviated as the CS policy, and we denote it by R^{CS}. A *trunk reservation* policy accepts calls of traffic class $k \in \mathcal{K}$ if and only if the number of unoccupied trunks is greater than some traffic class specific limit. A *threshold* policy accepts calls of traffic class $k \in \mathcal{K}$ as long as the number of ongoing class k calls stays below a traffic class specific limit z_k. When the vector of limits $\mathbf{z} = (z_k)_{k \in \mathcal{K}}$ belongs to the state space, that is $\mathbf{z} \in \Omega$, a threshold policy is called a *complete partitioning* policy; the name refers to the property that each traffic class has now exclusive use of a fixed part of link capacity, so that the multiservice link degenerates into a group of independent single-service links. Figure 3.2 illustrates these policy types on a small link with two traffic classes.

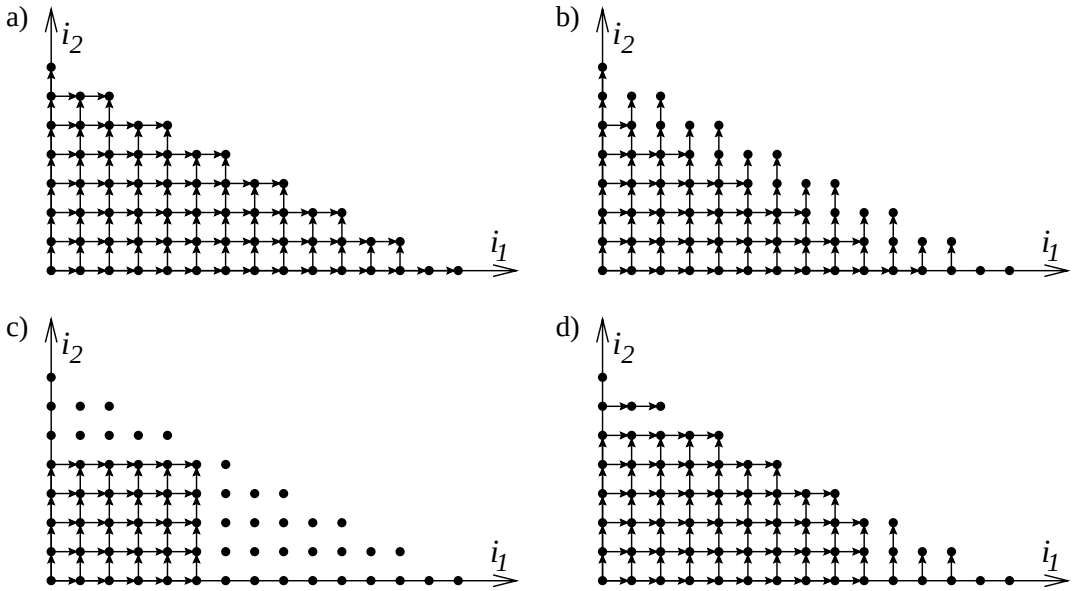

Figure 3.2: Some simple policies on the state space of the link of Figure 3.1. The transitions corresponding to accepted arriving calls are indicated by arrows; the transitions corresponding to completing calls do not vary under different policies and are not explicitly shown. a) The complete sharing policy. a) and b) Trunk reservation policies. c) A complete partitioning policy. a), c) and d) Threshold policies.

3.3 Product form balance

Let us study the balance probabilities of a multiservice link under a stationary policy R. As noted earlier, the Markov process I^R associated with policy R has a single irreducible set, and hence it has a unique equilibrium distribution. The equilibrium equations (2.11) can be written for the process I^R as

$$\pi_{\mathbf{i}}(R)\Big(\sum_{k\in\mathcal{K}} i_k\mu_k + \sum_{k\in R_{\mathbf{i}}}\lambda_k\Big) = \sum_{\substack{k\in\mathcal{K}\\ \mathbf{i}+\mathbf{e}_k\in\Omega}} (i_k+1)\mu_k\pi_{\mathbf{i}+\mathbf{e}_k}(R) + \sum_{\substack{k\in\mathcal{K}\\ k\in R_{\mathbf{i}-\mathbf{e}_k}}} \lambda_k\pi_{\mathbf{i}-\mathbf{e}_k}(R) \quad \text{for all } \mathbf{i}\in\Omega, \tag{3.9}$$

$$\sum_{\mathbf{i}\in\Omega}\pi_{\mathbf{i}}(R) = 1. \tag{3.10}$$

In general no simple alternative is known to solving these equations directly, but the following theorem characterizes the class of policies for which the balance probabilities are given by a so-called *product form* solution analogous to the single-service solution (3.2).

Theorem 3.1. *Let R be a stationary policy for a multiservice link with the state space Ω, and let $\Omega_0 \subseteq \Omega$ be the recurrent set of the associated Markov process I^R. Then the following conditions are equivalent:*

i) For all $\mathbf{i}\in\Omega_0$ and $k\in\mathcal{K}$, calls of traffic class k are accepted in state $\mathbf{i}$ under policy R if and only if $\mathbf{i}+\mathbf{e}_k\in\Omega_0$.

ii) The balance probabilities of I^R are of the product form

$$\pi_{\mathbf{i}}(R) = \frac{1}{G} \prod_{k \in \mathcal{K}} \frac{A_k^{i_k}}{i_k!} \quad \textit{for all } \mathbf{i} \in \Omega_0, \tag{3.11}$$

where $A_k = \lambda_k / \mu_k$ and

$$G = \sum_{\mathbf{i} \in \Omega_0} \prod_{k \in \mathcal{K}} \frac{A_k^{i_k}}{i_k!} \tag{3.12}$$

is a normalization constant.

iii) The balance probabilities of I^R satisfy the local balance equations

$$\lambda_k \pi_{\mathbf{i}-\mathbf{e}_k}(R) = i_k \mu_k \pi_{\mathbf{i}}(R) \quad \textit{for all } \mathbf{i} \in \Omega_0 \textit{ and } k \in \mathcal{K} \textit{ such that } i_k > 0. \tag{3.13}$$

Proof. The proof is presented in four parts: first we show the equivalence of *ii)* and *iii)* by proving that *ii)* $\implies$ *iii)* and *iii)* $\implies$ *ii)*, and then we show the equivalence of *i)* and *iii)* by proving *i)* $\implies$ *iii)* and *iii)* $\implies$ *i)*.

ii) $\implies$ *iii)*: First, note that transitions from any state $\mathbf{i}$ with $i_k > 0$ to state $i - \mathbf{e}_k$ cannot be denied by any policy for any traffic class $k \in \mathcal{K}$. This means that if a state $\mathbf{i}$ is recurrent, then all the states $\mathbf{j}$ such that $\mathbf{j} \leq \mathbf{i}$ are recurrent as well. Hence we have for recurrent $\mathbf{i} \in \Omega_0$ such that $i_k > 0$ for a $k \in \mathcal{K}$ that $\mathbf{i} - \mathbf{e}_k \in \Omega_0$ and thus both $\pi_{\mathbf{i}}(R)$ and $\pi_{\mathbf{i}-\mathbf{e}_k}(R)$ are given by the product form (3.11); the local balance equations (3.13) now trivially follow.

iii) $\implies$ *ii)*: Given any recurrent state $\mathbf{i} \in \Omega_0$, we can construct a finite path of states $\mathbf{i}_0, \mathbf{i}_1, \ldots, \mathbf{i}_n$ such that $\mathbf{i}_0 = i$, $\mathbf{i}_n = \mathbf{0}$ and for all $j = 1, 2, \ldots, n$ we have $\mathbf{i}_j = \mathbf{i}_{j-1} - \mathbf{e}_{k_j}$ for some $k_j \in \mathcal{K}$. By the notes about recurrent states above, all states on the constructed path are recurrent, and we have by (3.13) that

$$\begin{aligned} \pi_{\mathbf{i}}(R) = \frac{\lambda_{k_1}}{i_{k_1}\mu_{k_1}} \pi_{\mathbf{i}_1}(R) = \frac{\lambda_{k_1}\lambda_{k_2}}{i_{k_1} i_{k_2} \mu_{k_1}\mu_{k_2}} \pi_{\mathbf{i}_2}(R) = \cdots \\ = \prod_{k \in \mathcal{K}} \frac{\lambda_k^{i_k}}{i_k!\, \mu_k^{i_k}} \pi_0(R) \quad \text{for all } \mathbf{i} \in \Omega_0, \end{aligned} \tag{3.14}$$

where the final form is as shown regardless of which states $\mathbf{i}_1, \mathbf{i}_2, \ldots, \mathbf{i}_n$ happened to be chosen. When the balance probabilities given by (3.14) are normalized to sum to unity, we get the product form balance (3.11). This completes the equivalence of conditions *ii)* and *iii)*.

i) $\implies$ *iii)*: Since I^R has a unique equilibrium distribution, it is sufficient to show that there are numbers $\pi_{\mathbf{i}}(R)$, $\mathbf{i} \in \Omega$, satisfying *iii)* that also satisfy the equilibrium equations (3.9) and the normalization equations (3.10). We already know that the numbers $\pi_{\mathbf{i}}(R)$, $\mathbf{i} \in \Omega$, given by (3.11) satisfy *iii)* and are normalized. Noting that $\pi_{\mathbf{i}}(R) = 0$ for all $\mathbf{i} \notin \Omega_0$, and using the local balance equations (3.13) we can write the equilibrium equations (3.9) in recurrent states as

$$\pi_{\mathbf{i}}(R) \Big(\sum_{k \in \mathcal{K}} i_k \mu_k + \sum_{k \in R_{\mathbf{i}}} \lambda_k \Big) = \sum_{\substack{k \in \mathcal{K} \\ \mathbf{i} + \mathbf{e}_k \in \Omega_0}} \lambda_k \pi_{\mathbf{i}}(R) + \sum_{\substack{k \in \mathcal{K} \\ k \in R_{\mathbf{i}-\mathbf{e}_k}}} i_k \mu_k \pi_{\mathbf{i}}(R) \quad \text{for all } \mathbf{i} \in \Omega_0, \tag{3.15}$$

where we define $R_{\mathbf{x}} = \emptyset$ for vectors $\mathbf{x} \notin \Omega$. Now by *i)* both sides of (3.15) contain exactly the same terms, and we conclude that the equilibrium equations are satisfied in recurrent

states. In transient states the equilibrium equations are trivially satisfied, and we conclude that *i)* $\Longrightarrow$ *iii)*.

iii) $\Longrightarrow$ *i)*: First note that equations (3.15) were derived without assuming *i)*; they are simply the equilibrium equations (3.9) simplified by the local balance equations (3.13). Since $\pi_{\mathbf{i}}(R) > 0$ for all $\mathbf{i} \in \Omega_0$, we can cancel out the common factor $\pi_{\mathbf{i}}(R)$ from each equation in (3.15), and we proceed to eliminate common terms from both sides of the equations, noting that $k \in R_{\mathbf{i}} \implies \mathbf{i} + \mathbf{e}_k \in \Omega_0$, to derive

$$\sum_{\substack{k \in \mathcal{K} \\ k \notin R_{\mathbf{i}-\mathbf{e}_k}}} i_k \mu_k = \sum_{\substack{k \in \mathcal{K} \setminus R_{\mathbf{i}} \\ \mathbf{i}+\mathbf{e}_k \in \Omega_0}} \lambda_k \quad \text{for all } \mathbf{i} \in \Omega_0, \tag{3.16}$$

where again $R_{\mathbf{x}} = \emptyset$ for vectors $\mathbf{x} \notin \Omega$. Both sides of (3.16) are clearly nonnegative; we shall prove that they are in fact zero.

Suppose that the right-hand side of (3.16) is strictly positive for some $\mathbf{i}_0 \in \Omega_0$; this means that there is a $k_0 \in \mathcal{K} \setminus R_{\mathbf{i}_0}$ such that state $\mathbf{i}_1 = \mathbf{i}_0 + \mathbf{e}_{k_0}$ is recurrent. Let us now study the equation (3.16) for state $\mathbf{i}_1$. Since $k_0 \notin R_{\mathbf{i}_1 - \mathbf{e}_{k_0}}$, we have a nonzero term $i_{k_0}\mu_{k_0}$ on the left-hand side. This implies that the right-hand side is strictly positive, and thus there is a $k_1 \in \mathcal{K} \setminus R_{\mathbf{i}_1}$ such that state $\mathbf{i}_2 = \mathbf{i}_1 + \mathbf{e}_{k_1}$ is recurrent. By continuing in this manner we can construct an arbitrarily long chain of *distinct* recurrent states $\mathbf{i}_0, \mathbf{i}_1, \mathbf{i}_2, \ldots$, but this is clearly impossible since the state space is finite. We conclude that the right-hand-side of (3.16) must be zero for all $\mathbf{i} \in \Omega_0$.

Consequently all terms on the left-hand sides of (3.16) are zero, and it follows that for all $k \in \mathcal{K}$ and $\mathbf{i} \in \Omega_0$ we have either $i_k = 0$ or $k \in R_{\mathbf{i}-\mathbf{e}_k}$. This means that calls of traffic class $k \in \mathcal{K}$ are accepted in all states $\mathbf{i}$ such that $\mathbf{i} + \mathbf{e}_k \in \Omega_0$. It remains to verify that no other calls are accepted in recurrent states, but this is obvious considering that if a call of a traffic class $k \in \mathcal{K}$ is accepted in a recurrent state $\mathbf{i}$, then state $\mathbf{i} + \mathbf{e}_k$ is necessarily recurrent as well, and thus the proof is complete. □

The insensitivity property, ie that the call holding time distribution can be arbitrary without affecting the balance probabilities, holds in the multiservice case exactly when the control policy is of the form specified in Theorem 3.1; see [4], [24].

Equations (3.13) are called local balance equations because they balance the "probability flow" between adjacent pairs of states; this requirement is a lot stricter than the general balance condition (3.9) which only considers the total of all probability flows into and out of a state. The concepts of local balance and product form balance arise in many kinds of problems in queuing theory [24], [38]; they are closely connected with the notion of *time reversibility*, the property of a system that its development followed backward in the time variable is statistically indistinguishable from its development forward in time.

Control policies that satisfy part *i)* of the theorem and deny all calls in states outside Ω_0 are called *coordinate convex* policies; the name refers to the shape of the region Ω_0 of the state space where all calls are accepted. Correspondingly, a set $\Omega_0 \subseteq \mathbb{N}^K$ is called coordinate convex if for any $\mathbf{i} \in \Omega_0$ and k such that $i_k > 0$ also $\mathbf{i} - \mathbf{e}_k \in \Omega_0$. There is a one-to-one correspondence between coordinate convex policies and nonempty coordinate convex subsets of the state space Ω. Of the policies mentioned, all threshold policies including the complete sharing policy and the complete partitioning policies are coordinate convex, provided that the call acceptances in transient states are defined appropriately, but a trunk reservation policy is coordinate convex only when it degenerates to the complete sharing policy.

One can derive new coordinate convex sets from a coordinate convex set $\Omega_0 \subseteq \Omega$ by choosing any subset of states $\mho \subseteq \Omega \setminus \{\mathbf{0}\}$ that satisfies

$$\mathbf{j} \geq \mathbf{i} \implies \mathbf{j} \in \mho \quad \text{for all } \mathbf{i} \in \mho \text{ and } \mathbf{j} \in \Omega, \tag{3.17}$$

and then removing the states $\mho$ from set Ω_0. Since transient states are immaterial from the point of view of long-run system behaviour, one could as well completely remove the states $\mho$ from the state space of the system. Obviously this kind of state space truncation does not invalidate the product form balance of any coordinate convex policies applicable in the truncated state space.

The product form balance can be generalized to state-dependent arrival rates and call holding times, provided that the products computed in part *iii* $\implies$ *ii* of the proof of Theorem 3.1 are independent of the chosen path. A simple way to guarantee this is to require that the arrival rates and call holding times of class k calls depend only on i_k, the number of class k calls in progress. In fact, the balance probabilities corresponding to state-dependent transition rates can be an arbitrary probability distribution on the link state space, and still satisfy the local balance conditions; to see this, suppose that $\pi_{\mathbf{i}-\mathbf{e}_k}(R)$ and $\pi_{\mathbf{i}}(R)$ in (3.13) are taken from an arbitrary probability distribution over Ω and note that for each $\mathbf{i} \in \Omega$ the local balance is satisfied by $\lambda_k(\mathbf{i}-\mathbf{e}_k) = \alpha(\mathbf{i}) i_k \pi_{\mathbf{i}}(R)$ and $\mu_k(\mathbf{i}) = \alpha(\mathbf{i})\pi_{\mathbf{i}-\mathbf{e}_k}$ for all values of $\alpha(\mathbf{i})$.

3.4 Kaufman-Roberts recursion

Let us consider the problem of evaluating the time blocking for a traffic class under the complete sharing policy. The time blocking of traffic class $k \in \mathcal{K}$ is equal to the sum of the balance probabilities of the states where calls of traffic class k are blocked, specifically the states $\mathbf{i} \in \Omega$ such that $\mathbf{i} + \mathbf{e}_k \notin \Omega$. Since the complete sharing policy is coordinate convex, we have the explicit expression (3.11) for its balance probabilities, provided that the normalization constant (3.12) can be computed. The difficulty is that even the number of blocking states alone can easily be excessive for practical computation.

The following simple solution was first invented by Fortet and Grandjean in 1964 but it only became widely known after its independent reinventions by Kaufman [23] and Roberts [39] in 1981, and consequently the algorithm is commonly called Kaufman-Roberts recursion.

Let us first partition the state space into $C+1$ disjoint subsets, so that each subset is specified by the number of occupied trunks:

$$\Omega(c) = \big\{ \mathbf{i} \in \Omega \;\big|\; \mathbf{i}^{\mathrm{T}}\mathbf{b} = c \big\}, \quad c = 0, 1, \ldots, C, \tag{3.18}$$

Observe that the set of states where calls of traffic class $k \in \mathcal{K}$ are blocked under the complete sharing policy is now given by $\Omega(C - b_k + 1) \cup \Omega(C - b_k + 2) \cup \cdots \cup \Omega(C)$.

The Kaufman-Roberts recursion provides an effective method for computing the long-run probability that the Markov process $I^{R^{\mathrm{CS}}}$ realized by a multiservice link run under the complete sharing policy is found in set $\Omega(c)$, denoted by $\hat{\pi}_c$:

$$\hat{\pi}_c = \mathrm{P}\Big[I^{R^{\mathrm{CS}}} \in \Omega(c) \Big] = \sum_{\mathbf{i} \in \Omega(c)} \pi_{\mathbf{i}}(R^{\mathrm{CS}}), \quad c = 0, 1, \ldots, C. \tag{3.19}$$

Here $\pi_{\mathbf{i}}(R^{\mathrm{CS}})$, $\mathbf{i} \in \Omega$, are the product form balance probabilities of the complete sharing policy as given by (3.11). In terms of the aggregated balance probabilities (3.19) the blocking probability of traffic class $k \in \mathcal{K}$ is simply $\hat{\pi}_{C-b_k+1} + \hat{\pi}_{C-b_k+2} + \cdots + \hat{\pi}_C$.

Proposition 3.2 (Kaufman-Roberts recursion). *On a multiservice link under the complete sharing policy the aggregated balance probabilities $\hat{\pi}_c$ of the state sets $\Omega(c)$, $c = 0, 1, \ldots, C$, satisfy the recursion formula*

$$c\hat{\pi}_c = \sum_{k \in \mathcal{K}} \frac{\lambda_k}{\mu_k} b_k \hat{\pi}_{c-b_k} \quad \textit{for all } c = 0, 1, \ldots, C, \tag{3.20}$$

where $\hat{\pi}_c$ is considered zero for $c < 0$.

Proof. By Theorem 3.1 the balance probabilities $\pi_{\mathbf{i}}(R^{\mathrm{CS}})$, $\mathbf{i} \in \Omega$, of the complete sharing policy satisfy the local balance conditions (3.13). We define $\pi_{\mathbf{x}}(R^{\mathrm{CS}}) = 0$ for vectors $\mathbf{x}$ outside the state space Ω so that the local balance equations can be reformulated for the complete sharing policy as

$$i_k \pi_{\mathbf{i}}(R^{\mathrm{CS}}) = \frac{\lambda_k}{\mu_k} \pi_{\mathbf{i}-\mathbf{e}_k}(R^{\mathrm{CS}}) \quad \text{for all } \mathbf{i} \in \Omega \text{ and } k \in \mathcal{K}. \tag{3.21}$$

Summing (3.21) over a state class $\Omega(c)$ we get

$$\sum_{\mathbf{i} \in \Omega(c)} i_k \pi_{\mathbf{i}}(R^{\mathrm{CS}}) = \sum_{\mathbf{i} \in \Omega(c)} \frac{\lambda_k}{\mu_k} \pi_{\mathbf{i}-\mathbf{e}_k}(R^{\mathrm{CS}}) \quad \text{for all } k \in \mathcal{K} \text{ and } c = 0, 1, \ldots, C. \tag{3.22}$$

In order to simplify this further we note that nonzero terms on the right-hand side have $i_k > 0$ since otherwise $\mathbf{i} - \mathbf{e}_k \notin \Omega$, and the equivalence

$$\begin{aligned} \big(\mathbf{i} \in \Omega(c) \text{ and } i_k > 0\big) &\iff \big(\mathbf{i}^{\mathrm{T}}\mathbf{b} = c \text{ and } \mathbf{i} \geq \mathbf{e}_k\big) \\ &\iff \big((\mathbf{i} - \mathbf{e}_k)^{\mathrm{T}}\mathbf{b} = c - b_k \text{ and } \mathbf{i} - \mathbf{e}_k \geq \mathbf{0}\big) \iff \mathbf{i} - \mathbf{e}_k \in \Omega(c - b_k) \end{aligned} \tag{3.23}$$

shows that we can simplify (3.22) into the form

$$\sum_{\mathbf{i} \in \Omega(c)} i_k \pi_{\mathbf{i}}(R^{\mathrm{CS}}) = \frac{\lambda_k}{\mu_k} \sum_{\mathbf{j} \in \Omega(c-b_k)} \pi_{\mathbf{j}}(R^{\mathrm{CS}}) \quad \text{for all } k \in \mathcal{K} \text{ and } c = 0, 1, \ldots, C. \tag{3.24}$$

Now multiplying (3.24) by b_k and summing the equations over $k \in \mathcal{K}$, we get

$$\sum_{\mathbf{i} \in \Omega(c)} \pi_{\mathbf{i}}(R^{\mathrm{CS}}) \sum_{k \in \mathcal{K}} i_k b_k = \sum_{k \in \mathcal{K}} \frac{\lambda_k}{\mu_k} b_k \sum_{\mathbf{j} \in \Omega(c-b_k)} \pi_{\mathbf{j}}(R^{\mathrm{CS}}) \quad \text{for all } c = 0, 1, \ldots, C. \tag{3.25}$$

which readily yields the desired result. That $\hat{\pi}_c$ must be considered zero for $c < 0$ follows from our definition of $\pi_{\mathbf{x}}(R^{\mathrm{CS}})$ as zero for vectors $\mathbf{x}$ outside Ω. The proof is thus complete. □

In practice one uses the recursion formula (3.20) to first compute unnormalized aggregated probabilities by setting the unnormalized probability for $c = 0$ to unity; this specific choice gives the product form normalization constant (3.12) as the sum of the unnormalized aggregated probabilities. Including the normalization of the probabilities, computation of the aggregated balance probabilities $\hat{\pi}_c$, $c = 0, 1, \ldots, C$, by Kaufman-Roberts recursion requires $O(KC)$ arithmetic operations, where K is the number of traffic classes.

We conclude this section in a corollary which was already noted by Kaufman [23]; this result will prove useful in the sequel.

Corollary 3.2. *The average rate of calls of traffic class $k \in \mathcal{K}$ completing in states of the set $\Omega(c)$ is given by*

$$\mathrm{E}\Big[I_k^{R^{\mathrm{CS}}} \mu_k \;\Big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \Big] = \frac{\lambda_k \hat{\pi}_{c-b_k}}{\hat{\pi}_c} \quad \textit{for all } c = 0, 1, \ldots, C, \tag{3.26}$$

where $\hat{\pi}_c$ is considered zero for $c < 0$.

Proof. Once the expected call completion rate is expanded in terms of the individual state balance probabilities, the result follows directly from equation (3.24):

$$\begin{aligned} \mathrm{E}\Big[I_k^{R^{\mathrm{CS}}} \mu_k \;\Big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \Big] &= \frac{\sum_{\mathbf{i} \in \Omega(c)} i_k \mu_k \pi_{\mathbf{i}}(R^{\mathrm{CS}})}{\sum_{\mathbf{i} \in \Omega(c)} \pi_{\mathbf{i}}(R^{\mathrm{CS}})} \\ &= \frac{\lambda_k \sum_{\mathbf{j} \in \Omega(c-b_k)} \pi_{\mathbf{j}}(R^{\mathrm{CS}})}{\sum_{\mathbf{i} \in \Omega(c)} \pi_{\mathbf{i}}(R^{\mathrm{CS}})} = \frac{\lambda_k \hat{\pi}_{c-b_k}}{\hat{\pi}_c} \quad \text{for all } c = 0, 1, \ldots, C. \end{aligned} \tag{3.27}$$

□

3.5 The convolution algorithm

The first convolution-like method for queuing networks is generally credited to Buzen [5]. Applications of convolution methods for computing performance measures in teletraffic models were introduced by Iversen [20], [22] and independently by Tsang and Ross [52].

We discuss a generic convolution algorithm for effectively computing sums of the form

$$\sum_{\mathbf{i} \in \Omega(c)} \prod_{k \in \mathcal{K}} f_k(i_k), \quad c = 0, 1, \ldots, C, \tag{3.28}$$

where f_k, $k \in \mathcal{K}$, are arbitrary functions defined for integers $0, 1, \ldots, \lfloor \frac{C}{b_k} \rfloor$. The ability to practically compute these sums over the enormous link state space is a surprisingly versatile tool. The only known alternative family of methods of comparable generality is Monte Carlo sampling; both convolution and Monte Carlo methods are discussed in the book [40] by Ross.

The acyclic convolution $\mathbf{x} * \mathbf{y}$ of two $(C+1)$-vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as a $(C+1)$-vector with elements given by

$$[\mathbf{x} * \mathbf{y}]_n = \sum_{j=0}^{n} x_j y_{n-j}, \quad n = 0, 1, \ldots, C. \tag{3.29}$$

As an operation convolution is commutative and associative, as is apparent from the definition. Computing the convolution directly according to the definition takes $O(C^2)$ arithmetic operations. There is a multitude of more and less well-known transformations like the FFT (Fast Fourier Transform) or various number theoretic transforms that allow computing a convolution in $O(C \log C)$ operations [8]; however Tsang and Ross [52] found a simple FFT method to give inaccurate results for large systems.

Let us assign indices $1, 2, \ldots, K$ for the elements of the set $\mathcal{K}$ of traffic classes.

Proposition 3.3. *Suppose that the vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^K \in \mathbb{R}^{C+1}$ are defined elementwise by*

$$v_j^k = \begin{cases} f_k(\frac{j}{b_k}) & \textit{if } b_k | j, \\ 0 & \textit{otherwise,} \end{cases} \quad \textit{for all } j = 0, 1, \ldots, C \textit{ and } k = 1, 2, \ldots, K. \tag{3.30}$$

Then the elements of the convolution $\mathbf{s} = \mathbf{v}^1 * \mathbf{v}^2 * \cdots * \mathbf{v}^K$ *satisfy*

$$s_c = \sum_{\mathbf{i}\in\Omega(c)} \prod_{k=1}^{K} f_k(i_k) \quad \textit{for all } c = 0, 1, \ldots, C. \tag{3.31}$$

Proof. Let us for notational simplicity assume that $f_k(x)$ is defined for non-integral x, so that we can write $v_j^k = 1_{b_k|j} f_k(\frac{j}{b_k})$. With this notation, we can expand element c of the convolution vector $\mathbf{s}$ by the definition (3.29) of convolution as follows:

$$\begin{aligned}
s_c &= \sum_{j_1=0}^{n} v_{j_1}^1 [\mathbf{v}^2 * \mathbf{v}^3 * \cdots * \mathbf{v}^K]_{c-j_1} \\
&= \sum_{j_1=0}^{c} 1_{b_1|j_1} f_1(\tfrac{j_1}{b_1}) \sum_{j_2=0}^{c-j_1} 1_{b_2|j_2} f_2(\tfrac{j_2}{b_2}) [\mathbf{v}^3 * \mathbf{v}^4 * \cdots * \mathbf{v}^K]_{c-j_1-j_2} \\
&= \sum_{j_1=0}^{c} 1_{b_1|j_1} f_1(\tfrac{j_1}{b_1}) \sum_{j_2=0}^{c-j_1} 1_{b_2|j_2} f_2(\tfrac{j_2}{b_2}) \sum_{j_3=0}^{c-j_1-j_2} 1_{b_3|j_3} f_3(\tfrac{j_3}{b_3}) \quad \cdots \\
&\quad \cdots \sum_{j_{K-1}=0}^{c-j_1-j_2-\ldots-j_{K-2}} 1_{b_{K-1}|j_{K-1}} f_{K-1}(\tfrac{j_{K-1}}{b_{K-1}}) 1_{b_K|j_K} f_K(\tfrac{j_K}{b_K}) \Big|_{j_K = c-j_1-j_2-\ldots-j_{K-1}}.
\end{aligned} \tag{3.32}$$

It is clear that nonzero terms must have $b_k|j_k$ for all $k = 1, 2, \ldots, K$; accordingly we make the change of variable $i_k = j_k/b_k$ and rewrite (3.32) to only iterate over potentially nonzero terms:

$$\begin{aligned}
s_c &= \sum_{i_1=0}^{\lfloor \frac{c}{b_1} \rfloor} f_1(i_1) \sum_{i_2=0}^{\lfloor \frac{c-i_1 b_1}{b_2} \rfloor} f_2(i_2) \sum_{i_3=0}^{\lfloor \frac{c-i_1 b_1 - i_2 b_2}{b_3} \rfloor} f_3(i_3) \quad \cdots \\
&\quad \cdots \sum_{i_{K-1}=0}^{\lfloor \frac{c-i_1 b_1 - i_2 b_2 - \ldots - i_{K-2} b_{K-2}}{b_{K-1}} \rfloor} f_{K-1}(i_{K-1}) 1_{b_K|i_K} f_K(i_K) \Big|_{i_K = \frac{c-i_1 b_1 - i_2 b_2 - \ldots - i_{K-1} b_{K-1}}{b_K}}.
\end{aligned} \tag{3.33}$$

The equivalence of (3.33) and (3.31) now follows by noting that (3.33) iterates over all vectors $(i_1, i_2, \ldots, i_K)$ with nonnegative integer elements such that $\sum_{k=1}^{K} i_k b_k = c$; this is essentially equivalent to the definition (3.18) of $\Omega(c)$. The proof is thus complete. □

From the form (3.33) we see that element n of the convolution $\mathbf{v}^k * \mathbf{v}^{k+1} * \cdots * \mathbf{v}^K$ consists of terms where

$$i_k b_k + i_{k+1} b_{k+1} + \cdots + i_K b_K = n = c - i_1 b_1 - i_2 b_2 - \cdots - i_{k-1} b_{k-1}. \tag{3.34}$$

Hence if we compute the convolutions "backwards", beginning with $\mathbf{v}^{K-1} * \mathbf{v}^K$, and multiply the intermediate result vector $\mathbf{v}^k * \mathbf{v}^{k+1} * \cdots * \mathbf{v}^K$ by the diagonal matrix defined by some numbers $g(n)$, $n = 0, 1, \ldots, C$, then the elements of the ultimate result vector will give the sums

$$\sum_{\mathbf{i}\in\Omega(c)} g(i_k b_k + i_{k+1} b_{k+1} + \cdots + i_k b_K) \prod_{l=1}^{K} f_l(i_l), \quad c = 0, 1, \ldots, C. \tag{3.35}$$

Of course, the numbering of the traffic classes can be chosen as necessary to provide for g any argument of the form $\sum_{l\in\mathcal{L}} i_l b_l$ where $\mathcal{L}$ is a subset of $\mathcal{K}$.

More generally, we can extend the convolution method to compute sums $\sum_{\mathbf{i}\in\Omega(c)} F(\mathbf{i})$ where F is of the form

$$F(\mathbf{i}) = \sum_{j=1}^{n} \prod_{k=1}^{K} f_{jk}(i_k) \prod_{k=1}^{K} g_{jk}\Big(\sum_{l\in\mathcal{L}_{jk}} i_l b_l\Big), \quad \mathbf{i} \in \Omega, \tag{3.36}$$

where the f_{jk}'s and g_{jk}'s are arbitrary functions, and the sets $\mathcal{L}_{jk} \subseteq \mathcal{K}$ are for each j strictly ordered as $\mathcal{L}_{j1} \subseteq \mathcal{L}_{j2} \subseteq \cdots \subseteq \mathcal{L}_{jK}$. Here it is necessary to apply the convolution method separately for each j in (3.36), so that the effort of computing the sums $\sum_{\mathbf{i}\in\Omega(c)} F(\mathbf{i})$ for each $c = 0, 1, \ldots, C$ comprises a total of $n(K-1)$ convolutions.

3.5.1 On computing performance measures by convolution

We briefly discuss how to compute blocking probabilities by the convolution method for threshold policies. Under a threshold policy the recurrent set Ω_0 is of the form $\Omega_0 = \{\mathbf{i} \in \Omega \mid \mathbf{i} \leq \mathbf{z}\}$ for some $\mathbf{z} \in \mathbb{N}^K$, so that the balance probabilities (3.11) can be written simply as

$$\pi_{\mathbf{i}}(R) = \frac{1}{G} \prod_{k=1}^{K} 1_{i_k \leq z_k} \frac{A_k^{i_k}}{i_k!} \quad \text{for all } \mathbf{i} \in \Omega, \tag{3.37}$$

which is a product of functions of i_k; to use the equation in practice we need to evaluate the normalization constant G. Let us define the sums

$$s(c) = \sum_{\mathbf{i}\in\Omega(c)} \prod_{k=1}^{K} 1_{i_k \leq z_k} \frac{A_k^{i_k}}{i_k!}, \quad c = 0, 1, \ldots, C, \tag{3.38}$$

which are clearly of the form (3.31) computable by the convolution method; all the sums can be computed by $K-1$ convolutions as shown in Proposition 3.3. In terms of these sums the normalization constant is given by $G = \sum_{c=0}^{C} s(c)$ and moreover, the time blocking of traffic class $k \in \mathcal{K}$ equals

$$\sum_{c=C-b_k+1}^{C} \sum_{\mathbf{i}\in\Omega(c)} \pi_{\mathbf{i}}(R) = \frac{1}{G} \sum_{c=C-b_k+1}^{C} s(c). \tag{3.39}$$

As in the single-service case, the assumption of time-homogenous Poisson arrival processes implies that also the call and traffic blockings of traffic class k equal (3.39).

One of the strengths of the convolution method is that it can as easily accommodate state-dependent transition rates, when for each $k \in \mathcal{K}$ the arrival rate λ_k and the call completion rate μ_k are functions of the number i_k of class-k calls in progress. In this case the call and traffic blockings are no longer equal to the time blocking, but they can still be computed by convolutions. Also, the convolution method can be applied to certain tree-shaped network topologies. We refer the interested reader to the books [40] by Ross and [21] by Iversen for the details.

3.6 Policy optimization and relative values

The tremendous size of the link state space in practical problems makes it impossible to directly use any of the standard policy optimization procedures described in Chapter 2. Methods of practically computable policy optimization are discussed in later chapters, but we make some basic observations here.

The Howard equations can be written for a policy R on the multiservice link as

$$\sum_{k\in\mathcal{K}\setminus R_{\mathbf{i}}} \lambda_k h_k - g + \sum_{k\in\mathcal{K}} i_k \mu_k (v_{\mathbf{i}-\mathbf{e}_k} - v_{\mathbf{i}}) + \sum_{k\in R_{\mathbf{i}}} \lambda_k (v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}) = 0 \quad \text{for all } \mathbf{i} \in \Omega. \tag{3.40}$$

There is no known general solution for these equations except in some degenerate cases of minor practical interest; for the sake of completeness we shall discuss these cases shortly.

As per Proposition 2.4, the average cost rate $g(R)$ can be expressed in terms of the equilibrium probabilities $\pi_{\mathbf{i}}(R)$, $\mathbf{i} \in \Omega$, of policy R by

$$g(R) = \sum_{k\in\mathcal{K}} \sum_{c=C-b_k+1}^{C} \sum_{\mathbf{i}\in\Omega(c)} h_k \lambda_k \pi_{\mathbf{i}}(R). \tag{3.41}$$

This can be computed efficiently whenever $\sum_{\mathbf{i}\in\Omega(c)} \pi_{\mathbf{i}}(R)$ is effectively computable, which happens for certain coordinate convex policies as discussed in the preceding sections. As noted earlier, costs in the link model can be interpreted so that each denied call incurs an immediate cost of h_k units, where k is the traffic class of the denied call. Correspondingly the relative values $w_{\mathbf{i}}(R)$, $\mathbf{i} \in \Omega$, measure the expected costs of calls blocked in the future, in addition to the average cost rate, when the system starts from state $\mathbf{i}$. When all the weights h_k equal unity, the average cost rate is actually the average number of calls blocked per unit time, and the relative value of a state tells the expectation of how many calls more or less than dictated by the average rate will be blocked.

An important property of the multiservice Markov decision model is that the minimizations inherent in the policy iteration and value iteration algorithms can be performed separately for each traffic class. In the policy improvement step of the policy iteration algorithm the following expression is to be minimized over all $a \in A(\mathbf{i})$ for some fixed $\mathbf{i} \in \Omega$:

$$\sum_{k\in\mathcal{K}\setminus a} \lambda_k h_k + \sum_{k\in\mathcal{K}} i_k \mu_k (v_{\mathbf{i}-\mathbf{e}_k} - v_{\mathbf{i}}) + \sum_{k\in a} \lambda_k (v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}). \tag{3.42}$$

By transforming the expression into the form

$$\sum_{k\in\mathcal{K}} \lambda_k \left(1_{k\notin a} h_k + 1_{k\in a} (v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}})\right) + \sum_{k\in\mathcal{K}} i_k \mu_k (v_{\mathbf{i}-\mathbf{e}_k} - v_{\mathbf{i}}), \tag{3.43}$$

we see that to minimize the expression over all $a \in A(\mathbf{i})$ it is sufficient to check for each traffic class $k \in \mathcal{K}$ that is possibly admissible, that is $\mathbf{i} + \mathbf{e}_k \in \Omega$, whether $v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}$ is less than h_k; if so, then k must be included in the set of admitted traffic classes a, and otherwise k is excluded from a.

The comparison of $v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}$ with h_k has an intuitive physical interpretation in terms of the following two quantities. The state-dependent *link shadow price* is defined as

$$p_k(\mathbf{i}) = v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}} \quad \text{for } k \in \mathcal{K} \text{ and } \mathbf{i} \in \Omega \text{ such that } \mathbf{i} + \mathbf{e}_k \in \Omega, \tag{3.44}$$

where the dependence on the old policy R for which the values $v_{\mathbf{i}}$, $\mathbf{i} \in \Omega$, were evaluated is implicit. Suppose that the link is in state $\mathbf{i}$ and a call of traffic class k arrives. By the definition of relative values, the link shadow price measures the expected amount of additional costs caused by proceeding from state $\mathbf{i} + \mathbf{e}_k$ instead of state $\mathbf{i}$, or in other words the expected additional costs caused by blocked calls in the future when the free link capacity is reduced by b_k trunks now. The link shadow price is an exact measure of expected future costs if the old policy R is followed from the moment of call acceptance onward.

The state-dependent *link net gain* is defined as

$$G_k(\mathbf{i}) = h_k - p_k(\mathbf{i}) \quad \text{for } k \in \mathcal{K} \text{ and } \mathbf{i} \in \Omega \text{ such that } \mathbf{i} + \mathbf{e}_k \in \Omega, \tag{3.45}$$

where again the dependence on policy is implicit. Since h_k can be interpreted as the immediate cost of rejecting an arriving call of traffic class k, the link net gain $G_k(\mathbf{i})$ can be thought of as the total decrease in costs, or increase in revenue, induced by accepting a call of traffic class k in state $\mathbf{i}$. Strictly speaking, the link net gain is simply the quantity minimized in the policy improvement step and its significance is that when the net gain is positive, accepting calls of traffic class k in state $\mathbf{i}$ will improve the policy.

A link controller can perform policy improvement in real time by computing the link net gain at the arrival of each new call and deciding accordingly whether to deny or accept the call. Of course the link controller would have to have immediate access to the relative values of some (hopefully close to optimal) policy, which is nontrivial to arrange when the link state space is excessively large.

Since the minimization of (3.43) is the only stage of the policy iteration algorithm where one needs to iterate over all actions, it is of no practical consequence that the number of different actions in the model grows exponentially with the number of traffic classes. The same conclusion applies to value iteration which minimizes an expression very similar to (3.42). Similar simplifications are also possible in the linear programming approach.

Policy optimization on a multiservice link can lead to the situation that calls of some traffic classes are denied completely, or served only very seldom. The concept of how equal some measure of the level of service is between the traffic classes is called access *fairness*, and it can be treated by methods of cooperative game theory as discussed in [9, Chapter 8]. We ignore the issue of fairness and assume that it is handled separately, for example by adjusting the reward parameters h_k, $k \in \mathcal{K}$, when needed.

3.6.1 Known solutions to Howard equations

When there is only one traffic class, the multiservice link Howard equations (3.40) degenerate into simple one-dimensional difference equations. Assuming the complete sharing policy which is the only sensible policy in the basic Erlang single-link model, then dropping the traffic class subscripts as meaningless and changing to a scalar state variable, the Howard equations reduce to

$$1_{i=C} h\lambda - g + i\mu(v_{i-1} - v_i) + 1_{i<C}\lambda(v_{i+1} - v_i) = 0 \quad \text{for all } i = 0, 1, \ldots, C. \tag{3.46}$$

By noting that this gives $v_{i+1} - v_i$ recursively in terms of $v_i - v_{i-1}$, and using the equations for $i = 0$ and $i = C$ as boundary conditions, it is straightforward to solve the equations algebraically for the link shadow prices $v_{i+1} - v_i$ in each state $i = 0, 1, \ldots, C-1$. While we could further express the value v_i for each i as an iterated sum of these differences, in practical network level policy optimization ([28]; see also Section 3.7) it suffices to know the link shadow prices, for essentially the same reason as explained above in the context of single link policy optimization: the link shadow price $v_{i+1} - v_i$ tells the expected additional costs caused by blocked future calls on the link when a new call is established in link state i.

We do not present the purely algebraic solution, but a more complicated but insightful method of solution based on probabilistic reasoning. For the most part this is how Krishnan [28] arrived at the solution, but with additional insight due to J. Virtamo. First, observe that state C is recurrent, and it is the only state where costs are incurred. Thus the average cost rate of the process is $g = h\lambda E(C, A)$ where $E(\cdot, \cdot)$ is the stationary probability of state C as given by the Erlang-B function and $A = \lambda/\mu$. Let us consider the growth in time of the total

accumulated cost, equal to h times the total number of blocked calls, when the process starts in state $i < C$. No costs are incurred before the moment of first transition to state $i+1$, and afterwards the statistical development of the accumulated cost is indistinguishable from that if the process were started in state $i+1$. Thus conditioning the expected total accumulated cost on the moment of first transition to state $i+1$, we obtain for the relative values of states i and $i+1$ that

$$v_i = v_{i+1} - gM_i, \tag{3.47}$$

where M_i is the expected time of first transition from state i to state $i+1$. Before the first transition to state $i+1$, the process can be considered a truncated system that is otherwise identical to the complete link but where the number of trunks is only i; since we begin from state i, the expected time of first transition to state $i+1$ is equal to the expected time between blocking events in the truncated system. The expected rate of blocking in the truncated system is $\lambda E(i, A)$ by the "Poisson Arrivals See Time Averages" property, and thus the expected time between blocking events is $1/\lambda E(i, A)$, and this is equal to M_i. We conclude that the link shadow price in state i is

$$v_{i+1} - v_i = gM_i = h\frac{E(C, A)}{E(i, A)} \quad \text{for all } i = 0, 1, \ldots, C-1. \tag{3.48}$$

It is easy to verify that this satisfies the single-service Howard equations (3.46). While the single-service case can be handled easily, in the general case the kind of reasoning used here leads to matrix-geometric considerations [37] which do not allow computing link shadow prices without solving the relative values of all the system states.

Another special case in solving the multiservice link Howard equations (3.40) arises when the policy R is a complete partitioning policy. Since a complete partitioning policy assigns each traffic class exclusive use of a fixed part of the link capacity, the link state process can be treated as a group of independent single-service processes, as long as the link state is in the recurrent set of the policy. Consequently, when the states $\mathbf{i}$ and $\mathbf{i} + \mathbf{e}_k$ are both in the recurrent set of the policy, the relative value difference $v_{\mathbf{i}+\mathbf{e}_k} - v_{\mathbf{i}}$ can be computed from the single-service solution (3.48) where the parameters are taken as $A = A_k = \lambda_k/\mu_k$, $h = h_k$, $i = i_k$, and C is the number of trunks in exclusive use of traffic class k. The result cannot be applied in transient states, ie when there are more ongoing calls of some traffic class than the capacity allocation of that traffic class would allow; however Kolarov and Hui [26] provide a relatively expensive recurrence for computing the relative values of the transient states when there are only two traffic classes.

3.7 Extending the model to networks

In addition to connection admission control, at network level one needs to consider the question of *routing*—which links should a connection between some arbitrary pair of nodes use? There is a wealth of different routing strategies, both ad hoc methods and ones with more theoretical basis; Dziong [9, Chapter 5] gives a general classification of extant approaches. We discuss only network extensions of the single-link Markov decision model of Section 3.2.

For the most part we continue with the notation used for a multiservice link, making modifications and extensions as we proceed. We denote the set of links in the network by $\mathcal{L}$, the number of links by L and the number of nodes in the network by N. We require that for each pair of nodes there is at least one path of links connecting them; this implies that $L \geq N - 1$. A network where each and every pair of nodes is connected by a direct link is called *fully connected*. In a fully connected network $L = N(N-1)/2$.

3.7.1 Fixed routing

Let us first suppose a fixed routing where a call requesting a connection between a pair of nodes is given a route as a deterministic function of the connection endpoint nodes. Only then is a separate connection admission decision made, and if the call is denied, then it is lost and no alternative routes are attempted. The multiservice link model can be extended for a fixed routing network in a very natural way by extending the concept of traffic class to include the route, so that there are a number of independent Poisson arrival processes for each route.

The same model applies also when the route of a call is determined at random from a set of possible routes without using any information about the state of the network. This is because the arrival process on each route choice will be Poisson as long as the initial arrival process being partitioned is Poisson. This kind of randomized route selection, known as *load balancing*, is useful when there is a lot of traffic between a particular pair of nodes and some of the alternative routes between those nodes are underutilized.

As in the single-link case, by assumptions of Poisson arrivals and exponential call holding times the network state is completely described by a vector of K integers indicating how many calls of each traffic class are in progress; the difference to the single-link case is in the way the state vector is constrained. Let us extend the trunk requirements into an L by K matrix B where element (l, k) tells how many trunks of link $l \in \mathcal{L}$ a connection of traffic class $k \in \mathcal{K}$ requires.[2] Thus the route a traffic class uses is encoded in the matrix B. Defining $\mathbf{C}$ as a vector where element l gives the number of trunks on link l, we can express as $B\mathbf{i} \leq \mathbf{C}$ the requirement that the connections carried on each link take up no more trunks than there is capacity on the link. Correspondingly the network state space is defined as

$$\Omega = \big\{\mathbf{i} \in \mathbb{N}^K \;\big|\; B\mathbf{i} \leq \mathbf{C}\big\}, \tag{3.49}$$

and the set of available actions in state $\mathbf{i} \in \Omega$ is

$$A(\mathbf{i}) = \Big\{a \subseteq \mathcal{K} \;\Big|\; B(\mathbf{i} + \mathbf{e}_k) \leq \mathbf{C} \text{ for all } k \in a\Big\}. \tag{3.50}$$

The transition rates $q_{\mathbf{ij}}(a)$ and cost rates $r_{\mathbf{i}}(a)$ are defined exactly as in the single-link multiservice case. The Markov decision model is now complete; were the state space of practical size, standard policy optimization procedures could now be used to find an optimal connection admission policy for the fixed routing network.

The only essential change in this extension of the multiservice model is that the shape of the state space is somewhat more complicated. In practice a far greater difficulty is the size of the state space—even in the network topologies with the least connections the number of states grows exponentially with N, and in better connected networks the exponent is of the order N^2. Thus the number of states is infeasibly large in all except the most trivial of multiservice networks. On the other hand, if all calls are identical except for their route, and the network is sufficiently small, then methods developed for policy optimization on a multiservice link can be applied to optimization of connection admission policy for fixed network routing, on the condition that the method can be generalized for a state space of the form (3.49).

[2] In practice it is quite possible that a connection requires a different number of trunks on different links; the reason for this is that the actual bandwidth requirements of connections vary over time, and the constant *equivalent bandwidth* on the basis of which the trunk requirement number is assigned depends not only on link capacity but also on the characteristics of the traffic to be carried on the link.

3.7.2 State-dependent routing

More generally, we can extend the fixed routing model to make state-dependent routing decisions. Let each connection class $k \in \mathcal{K}$ be assigned a set W_k of possible routes, and define an action as a mapping from the set of traffic classes to the set of routes, so that action a associates each traffic class $k \in \mathcal{K}$ with either a route $a(k) \in W_k$ or a special token $a(k) = s_{\text{nyet}}$ indicating that calls of traffic class k are to be blocked.

The state space is defined similarly to the fixed routing model: We identify a state with an integer vector $\mathbf{i} = (i_{(k,s)})_{k \in \mathcal{K},\, s \in W_k}$ where $i_{(k,s)}$ tells the number of class k connections in progress on route s. The number of trunks used on each link is limited by a constraint of the form $B\mathbf{i} \leq \mathbf{C}$ where matrix B is defined so that element $(l, (k, s))$ of B tells the number of trunks a call of traffic class $k \in \mathcal{K}$ routed on route $s \in W_k$ occupies on link $l \in \mathcal{L}$, and $\mathbf{C}$ is defined as in the fixed routing case. Thus the complete network state space is of the form

$$\Omega = \left\{ \mathbf{i} \in \mathbb{N}^D \;\middle|\; B\mathbf{i} \leq \mathbf{C} \right\}, \tag{3.51}$$

where $D = \sum_{k \in \mathcal{K}} |W_k|$.

In specifying the transition rates we need to take into account which route the action specifies for each connection class:

$$q_{\mathbf{ij}}(a) = \begin{cases} \lambda_k & \text{if } a(k) \neq s_{\text{nyet}} \text{ and } \mathbf{j} = \mathbf{i} + \mathbf{e}_{(k,a(k))} \text{ for some } k \in \mathcal{K}, \\ i_{(k,s)}\mu_k & \text{if } \mathbf{j} = \mathbf{i} - \mathbf{e}_{(k,s)} \text{ for some } k \in \mathcal{K} \text{ and } s \in W_k, \\ -\sum_{\mathbf{n} \neq \mathbf{i}} q_{\mathbf{in}}(a) & \text{if } \mathbf{j} = \mathbf{i}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.52}$$

Like in the earlier models, the cost rate in state $\mathbf{i}$ under action a is defined as a weighted sum of the arrival rates of the blocked traffic classes:

$$r_{\mathbf{i}}(a) = \sum_{k \in \mathcal{K}} 1_{a(k) = s_{\text{nyet}}} \lambda_k h_k. \tag{3.53}$$

While this Markov decision model does in principle provide the definitive solution for finding the optimal state-dependent routing and connection admission policy, the size of the model state space is beyond excessive on any practical network. Despite the impracticability of the model, let us extend some of the discussion in Section 3.6 to the network model.

Path shadow prices and net gains

In the general network Markov decision model with state-dependent routing, the policy improvement step of the policy iteration algorithm minimizes over a the expression

$$\sum_{k \in \mathcal{K}} 1_{a(k) = s_{\text{nyet}}} \lambda_k h_k + \sum_{k \in \mathcal{K}} 1_{a(k) \neq s_{\text{nyet}}} \lambda_k (v_{\mathbf{i} + \mathbf{e}_{(k,a(k))}} - v_{\mathbf{i}}) + \sum_{k \in \mathcal{K}} \sum_{s \in W_k} i_{(k,s)} \mu_k (v_{\mathbf{i} - \mathbf{e}_{(k,s)}} - v_{\mathbf{i}}), \tag{3.54}$$

and the minimization is performed separately for each $\mathbf{i} \in \Omega$. Like in the single-link case, the minimized expression reduces to a separable form

$$\sum_{k \in \mathcal{K}} \lambda_k \left(1_{a(k) = s_{\text{nyet}}} h_k + 1_{a(k) \neq s_{\text{nyet}}} (v_{\mathbf{i} + \mathbf{e}_{(k,a(k))}} - v_{\mathbf{i}}) \right) + \sum_{k \in \mathcal{K}} \sum_{s \in W_k} i_{(k,s)} \mu_k (v_{\mathbf{i} - \mathbf{e}_{(k,s)}} - v_{\mathbf{i}}), \tag{3.55}$$

where the minimizing a can be determined as follows. For each $k \in \mathcal{K}$, find the minimum of $v_{\mathbf{i}+\mathbf{e}_{(k,s)}} - v_{\mathbf{i}}$ over $s \in W_k$, and denote the minimizing s by $s_{\min}$. If h_k is smaller than $v_{\mathbf{i}+\mathbf{e}_{(k,s_{\min})}} - v_{\mathbf{i}}$, set $a(k) = s_{\text{nyet}}$ to deny calls of traffic class k in state $\mathbf{i}$; otherwise set $a(k) = s_{\min}$ to choose route $s_{\min}$ for calls of traffic class k in state $\mathbf{i}$.

The state-dependent *path shadow price* $p_k^s(\mathbf{i})$ is defined as

$$p_k^s(\mathbf{i}) = v_{\mathbf{i}+\mathbf{e}_{(k,s)}} - v_{\mathbf{i}} \quad \text{for } k \in \mathcal{K},\ s \in W_k \text{ and } \mathbf{i} \in \Omega, \tag{3.56}$$

where the dependence on policy is implicit. The path shadow price measures the expected amount of additional costs caused by proceeding from state $\mathbf{i} + \mathbf{e}_{(k,s)}$ instead of state $\mathbf{i}$, that is by accepting a call of traffic class k on route s in state $\mathbf{i}$. Correspondingly the state-dependent *path net gain* $G_k^s(\mathbf{i})$ is defined as

$$G_k^s(\mathbf{i}) = h_k - p_k^s(\mathbf{i}) \quad \text{for } k \in \mathcal{K},\ s \in W_k \text{ and } \mathbf{i} \in \Omega. \tag{3.57}$$

Intuitively, the path net gain measures the value of accepting a call of traffic class k on route s in state $\mathbf{i}$. In the policy improvement step of the policy iteration algorithm one essentially chooses the route with the maximum path net gain if the maximum is positive, and blocks calls of traffic class k if the net gain is less than or equal to zero.

3.7.3 The link independence assumption

In the literature the standard approach to reducing the complexity of network models is to assume that the arrival processes on each link are statistically independent. This assumption is accurate when each link carries traffic of a large number of different connection classes and specifically of different routes. There is a case when the links in fact are statistically independent: this happens with *direct routing*, a special case of fixed routing where all routes contain exactly one link. In a fully connected network direct routing can be a very reasonable policy. Using direct routing and the link independence assumption as the basis for decomposing Markov decision process based network policy optimization was first discussed by Krishnan and Ott [30].

Dziong and Mason [12], [13] have developed a general method of network policy iteration based on the link independence assumption. Since they use the alternative Markov decision formulation based on reward maximization, as mentioned in Section 3.2.1, the treatment here differs from the original in some details. The basic idea of the method is to define a separate Markov decision model for each link, and then compute estimates of link shadow prices in the link models. Under the link independence assumption the path shadow price is the sum of link shadow prices, thus allowing computation of path net gains and network level policy iteration.

The individual link models are multiservice link models of the form discussed earlier on, with state-dependent arrival rates. Let us now specify the parameters of the multiservice link model for link $l \in \mathcal{L}$. The set of traffic classes $\mathcal{K}^l$ on link l is the subset of network traffic classes that can be routed via link l. The trunk requirement b_k^l of traffic class k on link l is an element $(l,(k,s))$ of the matrix B where $s \in W_k$ is any route containing link l; all such elements of B should arguably be equal. The link state space Ω^l is the multiservice link state space determined by the link capacity and traffic class trunk requirements. By the link independence assumption, the network state is completely characterized by the states of the individual links, and correspondingly we define the network state space Ω as the cartesian product of the link state spaces. For a network state $\mathbf{i} \in \Omega$, let us denote the corresponding state of link l by $\mathbf{i}^l \in \Omega^l$.

The call holding times of traffic classes are naturally the same in the link models as in the network model; however, the arrival rates and cost parameters of the traffic classes

present more difficulties. The difficulties do not apply when the policy being improved on is a direct routing policy: in this case the network level arrival rate and cost parameter of traffic class $k \in \mathcal{K}$ are already specific to individual links, and no additional work is needed to derive link specific arrival rates and cost parameters.

In practical network control, the arrival rates and holding times are frequently re-estimated from connection data gathered in real time, so one can as easily gather data for link specific call arrival and completion processes. Under a state-dependent routing policy the arrival rates depend on the network state. However, to preserve link independence and to keep the arrival rate estimation manageable, we assume that the arrival rates depend only on particular aspects of link state or even that they are state independent; the assumptions have to be adjusted according to how well they can be supported by the computational methods used on the link models. Analytical estimation of link state dependent arrival rates for purposes of network design and planning is discussed in [14].

As defined above, the number of traffic classes in the link models would grow with the number of nodes in the network. However, this can be largely circumvented by combining within each link model traffic classes with equal trunk requirements and similar call holding times, which is accomplished by adding together their arrival rates. Since a Poisson arrival process can be seen as arising from a large number of independent sources, each triggering by a small probability, this aggregation of similar traffic classes that differ in their origin and destination nodes and arrival rates should only improve the accuracy of the assumption of Poisson arrivals in the link models.

For each path $s \in W_k$ on which calls of traffic class $k \in \mathcal{K}$ can be routed, the cost h_k of blocking a call of traffic class k has to be distributed on the links of path s. Provided that the path choices are disjoint, this can be written as $h_k = \sum_{l \in s} h_k^l$ for all $s \in W_k$, where h_k^l is the cost of blocking a call of traffic class k in the link process of link l. According to Dziong and Mason [13] the division rule has little effect on the performance of the method, so that it is sufficient to divide h_k evenly among the links of each path; nevertheless Dziong and Mason also discuss some more sophisticated division rules.

Let R denote the network level routing and connection admission policy which we set out to improve by policy iteration. By the link independence assumption the costs incurred on each link are independent, and consequently the relative values of network states are given by

$$w_{\mathbf{i}}(R) = \sum_{l \in \mathcal{L}} w_{\mathbf{i}^l}^l(R) \quad \text{for all } \mathbf{i} \in \Omega, \tag{3.58}$$

where $w_{\mathbf{j}}^l(R)$ is the relative value of state $\mathbf{j} \in \Omega_l$ in the individual link process of link l. This implies further that the state-dependent path shadow price of traffic class $k \in \mathcal{K}$ on path s in network state $\mathbf{i} \in \Omega$ can be expressed in terms of link shadow prices as

$$p_k^s(\mathbf{i}) = \sum_{l \in s} p_k(\mathbf{i}^l, l). \tag{3.59}$$

Now the path net gains can be computed by their definition (3.57), and used in policy improvement. The policy improvement can be performed in real time by computing the path net gains for all route choices at the arrival of each new call, and then choosing the route with the maximum positive path net gain. This is doable in practice provided that the link shadow prices are immediately available.

To provide the link shadow prices, we only need to compute the relative values separately in each link model, using whatever computational method deemed appropriate. Note that the network level policy is encoded in the state-dependent arrival rates in the link models; from the point of view of the link models the complete sharing policy is applied. However, it is possible to perform policy improvement in the individual link models and then

compute the link relative values for an improved policy, even though this was not done by Dziong and Mason. In effect, policy improvement on an individual link chooses a better connection admission policy for the independent arrival process of that link, and this also improves the network level connection admission policy by the link independence assumption; since the previous policy is present in the network policy improvement stage only implicitly through link shadow prices, it is only beneficial if the link shadow prices are for an improved policy.

In practice information on the states of the links should be frequently distributed around the network, but nevertheless routing decisions cannot be based on fully up-to-date information about the state of distant links. Some of the implementation issues are discussed by Dziong [9, Section 5.5]. The errors caused by inaccurate state information are outside the scope of this thesis.

Dziong and Mason found the network policy iteration to converge in one or two steps when the network level arrival rates were kept constant, and the method provided better performance than some simpler routing strategies. In the rest of this thesis, we concentrate on Markov control of individual multiservice links, assuming that the network level control is performed by the network policy iteration of Dziong and Mason, or some similar method.

Chapter 4

The Krishnan-Hübner method

In telecommunications literature a number of simplified link models have been proposed in order to make it feasible to estimate blocking probabilities on multiservice links; see for example [10], [11], [19], [29]. In principle any Markovian link model can be used to estimate the link shadow prices as well; however we shall discuss closer only the model Krishnan and Hübner [29] propose specifically for link shadow price estimation.

The available literature on general approximative Markov control tends to assume that the size N of the state space is small enough that operations taking $O(N)$ time are feasible, but of course this is not the case in the multiservice Erlang link model. However, by taking advantage of the structure of the link model it is possible to perform one iteration of a so-called aggregation-disaggregation method; this leads to a class of approximations generalizing the Krishnan-Hübner method. We begin the treatment by reviewing the literature on aggregation-disaggregation.

4.1 Aggregation-disaggregation

Aggregation-disaggregation algorithms are a fairly popular technique for solving the Howard equations and especially the Markov process equilibrium equations. Generally these algorithms work iteratively by converting the system to be solved into a smaller one using the current solution estimates (aggregation), then solving the smaller system and mapping the result back into the full state space to provide an improved solution estimate (disaggregation).

Aggregation-disaggregation methods were first proposed for linear programming in the Soviet literature; see the survey [53] by Vakhutinsky. However conditions of convergence when found were restrictive; a method that converges globally for Markov control problems with the discounted cost criterion was provided by Mendelssohn [35] but no generalization of this method for the undiscounted case has been found. In the present thesis we concentrate in aggregation-disaggregation for the linear system of Howard equations. Early work on aggregation-disaggregation for linear systems of equations is surveyed by Chatelin [6], and a good survey of later approaches is given by Schweitzer [43]. Unfortunately most theoretical and practical successes in aggregation-disaggregation have been limited to the following cases:

- Linear programming and linear equations where the matrix of the system is a contraction, which in the case of Markov control problems means that one must be using the discounted cost criterion. In this case the bounds of Whitt [54] for approximations of dynamic programs apply; the aggregation of Howard equations is treated by Schweitzer, Puterman and Kindle [45].

- The problem of determining the stationary balance probabilities (that is, the left eigenvector corresponding to the eigenvalue 0) of a stochastic matrix: see the surveys by Chatelin [6] and Schweitzer [42], [43]; tight bounds for a single iteration are provided by Courtois [7].

- Problems with special structure. A particularly tractable case is formed by problems where the matrix of the system is nearly block-diagonal, so that the system can be seen as a weakly coupled collection of separate Markov processes. The classic reference for this so-called "near completely decomposable" is the 1961 article [47] by Simon and Ando. Aldhaheri and Khalil [1] discuss policy iteration in the near completely decomposable context. On another note, Kim and Smith [25] define a general class of Markov processes where aggregation-disaggregation can be performed exactly.

Mandel and Sekerka [32] showed the local convergence (that is, convergence when the iteration starts in some neighbourhood of the solution) of a simple aggregation-disaggregation method when the matrix of the system is a contraction. Recently there has been some new progress in proving convergence for aggregation-disaggregation methods: for general contractive linear operators as well as Markov operators, Marek and Szyld [34] show local convergence. Similar results for the computation of the stationary balance vector of a Markov process are shown by Krieger [27], who interprets aggregation-disaggregation as an algebraic multigrid method. Finally, Marek and Mayer [33] show the global convergence of aggregation-disaggregation in stationary balance computation, when a sufficiently large (but constant) number of successive approximation steps are performed in the full problem dimension between aggregation steps.

As regards applying aggregation-disaggregation to the multiservice link control problem, the known methods do not extend to general linear systems with non-contractive matrices, nor do the Howard equations of the multiservice link control problem have structure of the kinds exploited in the literature. However, Schweitzer and Kindle [44] discuss a heuristic aggregation-disaggregation method for the Howard equations of a general semi-Markov problem with the average cost criterion,

$$v_i = c_i - g\tau_i + \sum_{j \in S} p_{ij} v_j \quad \text{for all } i \in S. \tag{4.1}$$

These are essentially the form (2.73) of continuous-time Howard equations from Chapter 2, with the policy dependencies omitted and c_i denoting $r_i(R_i)\tau_i(R_i)$. It is assumed that the Markov process associated with the transition probabilities p_{ij}, $i, j \in S$, has a single irreducible set, so that the Howard equations (4.1) augmented with the constraint

$$v_m = a \tag{4.2}$$

where $m \in S$ is an arbitrary state and a is a constant, have a unique solution in the variables g and v_i, $i \in S$.

Let $\mathcal{D}$ be a partitioning of the state space S into disjoint subsets, so that $\cup_{D \in \mathcal{D}} D = S$ and $D, E \in \mathcal{D},\ D \neq E \implies D \cap E = \emptyset$, where it is required that there is a $D_m \in \mathcal{D}$ such that $D_m = \{m\}$. Let every system state $i \in S$ be assigned a weight $w_i > 0$, such that $\sum_{i \in D} w_i = 1$ for all $D \in \mathcal{D}$. Now by multiplying each equation in (4.1) by the corresponding w_i and summing separately over each state class $D \in \mathcal{D}$, we get the reduced set of equations

$$\sum_{i \in D} w_i v_i = \sum_{i \in D} w_i c_i - g \sum_{i \in D} w_i \tau_i + \sum_{i \in D} \sum_{j \in S} w_i p_{ij} v_j \quad \text{for all } D \in \mathcal{D}, \tag{4.3}$$

which can be manipulated into the lower dimensional Howard equations analogue

$$\hat{v}_D = \hat{c}_D - g\hat{\tau}_D + \sum_{E\in\mathcal{D}} \hat{p}_{DE}\hat{v}_E \quad \text{for all } D \in \mathcal{D}, \tag{4.4}$$

where the aggregated variables and parameters are defined by

$$\hat{v}_D = \sum_{i\in D} w_i v_i, \quad D \in \mathcal{D}, \tag{4.5}$$

$$\hat{c}_D = \sum_{i\in D} w_i c_i, \quad D \in \mathcal{D}, \tag{4.6}$$

$$\hat{\tau}_D = \sum_{i\in D} w_i \tau_i, \quad D \in \mathcal{D}, \tag{4.7}$$

and

$$\hat{p}_{DE} = \frac{\sum_{i\in D}\sum_{j\in E} w_i p_{ij} v_j}{\sum_{j\in E} w_j v_j}, \quad D, E \in \mathcal{D}. \tag{4.8}$$

The constraint (4.2) implies for the aggregated variables $\hat{v}_D$ that

$$\hat{v}_{D_m} = a. \tag{4.9}$$

To coincide the aggregated equations (4.4), we get from the Howard equations (4.1) by simple manipulations the disaggregation equations

$$v_i - \sum_{j\in D} p_{ij} v_j = c_i - g\tau_i + \sum_{\substack{E\in\mathcal{D}\\ E\neq D}} p^{\#}_{iE}\hat{v}_E \quad \text{for all } i \in D,\ D \in \mathcal{D}, \tag{4.10}$$

where

$$p^{\#}_{iE} = \frac{\sum_{j\in E} p_{ij} v_j}{\sum_{j\in E} w_j v_j}, \quad i \in S,\ E \in \mathcal{D}. \tag{4.11}$$

In their paper Schweitzer and Kindle show that the aggregated equations (4.4) in combination with the constraint (4.9) and the disaggregation equations (4.10) are equivalent to the original Howard equations (4.1) and (4.2), provided that

1. the denominators in the coefficient definitions (4.8) and (4.11) are nonzero, which can be guaranteed by setting the constant a sufficiently large in the constraints (4.2) and (4.9),

2. the aggregated equations (4.4) and (4.9) are nonsingular; this too is satisfied for a sufficiently large as shown by Schweitzer and Kindle,

3. the recurrent set S_0 of the Markov process is not a subset of one of the state classes $D \in \mathcal{D}$, that is there are $D, E \in \mathcal{D}$, $D \neq E$, such that $D \cap S_0 \neq \emptyset$ and $E \cap S_0 \neq \emptyset$. This condition can be circumvented by modifying the disaggregation equations as shown in Appendix B of the original paper.

When all these conditions are satisfied, the equivalence of the linear systems can be proved by elementary means.

The algorithm. Ignoring technical details and convergence tests, we can specify a preliminary version of the iterative aggregation-disaggregation algorithm as follows:

1. Compute the aggregation parameters (4.6)–(4.8) and the disaggregation parameters (4.11) in terms of the current relative value estimates v_i, $i \in S$, as computed in the previous iteration.

2. Solve g and $\hat{v}_D$, $D \in \mathcal{D}$, from the aggregated system (4.4), (4.9), using the aggregation parameters computed in step 1. If the aggregated system turns out to be singular or $\hat{v}_D < 0$ for some $D \in \mathcal{D}$, increase by an appropriate amount both a in constraint (4.9) and the current relative value estimates v_i, $i \in S$, and then return to step 1.

3. For each state class $D \in \mathcal{D}$, solve new relative value estimates from the disaggregation equations (4.10), using the disaggregation parameters computed in step 1. If negative relative values are found, increase both a and the relative value estimates by an appropriate amount.

Unfortunately the outlined algorithm, while beneficial in many practical cases, is not globally convergent and does fail in practice. Because of this Schweitzer and Kindle specify the algorithm so that successive approximation (Gauss-Jacobi iterative solution of the Howard equations) is used as a fall-back method to ensure convergence. On the other hand for problems where there is a natural grouping of states with similar characteristics (state sojourn times, cost rates and sums of transition rates to other groups) aggregation-disaggregation was found to converge rapidly without using the fall-back method.

As regards the choice of the state weights w_i, $i \in S$, Schweitzer and Kindle suggest the equal weighting $w_i = 1/|D|$ for all $i \in D$ and $D \in \mathcal{D}$. They also show that by using the stationary balance probabilities of the states as weights condition 2 above for the unique solvability of the aggregated system will hold without need to increase a, although a still has to be increased sufficiently to ensure condition 1. Schweitzer and Kindle also discuss additional conditions that have to be satisfied for the aggregation-disaggregation method to work in principle if state weights are allowed to be equal to zero; however in their computational tests they found it harmful to set too many state weights to zero in practice.

While the aggregation-disaggregation method as such requires $O(|S|)$ or more operations in the computation of the aggregated parameters, the disaggregation stage, and the fall-back successive approximation, it is possible to perform a single aggregation-disaggregation iteration very efficiently in the multiservice link control problem. We discuss the details in Section 4.3.

Bertsekas and Castañon [3] develop a related aggregation-disaggregation method, where they use aggregation to cancel errors in the residuals of successive approximation results. Like Schweitzer and Kindle, they were unable to ensure convergence without falling back to pure successive approximation whenever aggregation-disaggregation fails to help. The reliance on successive approximation makes the method of Bertsekas and Castañon too expensive to be useful in the multiservice link control problem.

4.2 The Krishnan-Hübner method

In their paper [29] Krishnan and Hübner consider the random process formed by the number of occupied trunks on a multiservice link under the complete sharing policy. Their method for relative value estimation is based on the assumption that this process is Markovian. Of course, actually the number of occupied trunks at time t does not alone determine the

statistical future development of the number of occupied trunks, since the transition rates of the system vary depending on the exact link state. The process can be seen as an aggregated system, the states of which are identified with the state classes $\Omega(c)$, $c = 0, 1, \ldots, C$, of the full multiservice link model. Correspondingly we denote the states of the aggregated system by the integers $0, 1, \ldots, C$.

The transition rates $\hat{q}_{cd}$ of the aggregated system are derived from the transition rates $q_{\mathbf{ij}}(R^{\mathrm{CS}})$ of the full link model as expectations under the complete sharing policy. For simplicity of notation we assume that all the trunk requirements b_k, $k \in \mathcal{K}$, are different from each other; if this is not the case, one can simply add up all the transition rates of the traffic classes with equal trunk requirements in the following.

$$\begin{aligned}\hat{q}_{cd} &= \mathrm{E}\Big[\sum_{\mathbf{j} \in \Omega(d)} q_{I^{R^{\mathrm{CS}}}\mathbf{j}}(R^{\mathrm{CS}}) \;\Big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \Big] \\ &= \begin{cases} \lambda_k & \text{if } d = c + b_k \text{ for some } k \in R_c^{\mathrm{CS}}, \\ \mu_k \mathrm{E}\big[I_k^{R^{\mathrm{CS}}} \;\big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \big] & \text{if } d = c - b_k \text{ for some } k \in \mathcal{K}, \\ -\Big(\sum_{k \in R_c^{\mathrm{CS}}} \lambda_k + \sum_{k \in \mathcal{K}} \mu_k \mathrm{E}\big[I_k^{R^{\mathrm{CS}}} \;\big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \big] \Big) & \text{if } d = c, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{4.12}$$

Here R_c^{CS} denotes the common action of the complete sharing policy in all states of set $\Omega(c)$. Observe that the aggregated transition rates $\hat{q}_{cd}$, $c, d \in \{0, 1, \ldots, C\}$ form a proper infinitesimal generator matrix, since $\sum_{d=0}^{C} q_{cd} = 0$ for all c. It is straightforward to show that the infinitesimal generator matrix has a unique equilibrium distribution which corresponds exactly to the stationary probabilities of the link being in each state class $\Omega(c)$.

Krishnan and Hübner noticed that a result by Kaufman, stated as Corollary 3.2 in this thesis, presents a relatively simple and efficient method of computing the expectations required for the downward transition rates in (4.12) from the aggregated probabilities given by the Kaufman-Roberts recursion. Finally, the cost rates $\hat{r}_c$ in the aggregated model are defined analogously by

$$\hat{r}_c = \mathrm{E}\big[r_{I^{R^{\mathrm{CS}}}} \;\big|\; I^{R^{\mathrm{CS}}} \in \Omega(c) \big], \tag{4.13}$$

and their aggregation is trivial since under the complete sharing policy $r_{\mathbf{i}}$ is constant over each state class $\Omega(c)$. Now we can write aggregated Howard equations for the class-wise relative values $\hat{v}_c$ as

$$\hat{r}_c - g + \sum_{d=0}^{C} \hat{q}_{cd} \hat{v}_d = 0 \quad \text{for all } c = 0, 1, \ldots, C. \tag{4.14}$$

In combination with an additional constraint $\hat{v}_0 = 0$ these equations can be solved for g and $\hat{v}_c$, $c = 0, 1, \ldots, C$. These equations have a unique solution since they are the Howard equations for a Markov decision process satisfying the unichain assumption.

On the basis of the probabilistic similarity of the aggregated system and the full multiservice link model Krishnan and Hübner proceed to estimate link shadow prices by

$$p_k(\mathbf{i}) = \hat{v}_{\mathbf{b}^{\mathrm{T}}\mathbf{i} + b_k} - \hat{v}_{\mathbf{b}^{\mathrm{T}}\mathbf{i}}. \tag{4.15}$$

These link shadow price estimates can then be used for real-time single-link policy improvement as discussed in Section 3.6, and by making the link independence assumption, in network policy improvement of a direct routed initial policy as discussed in Section 3.7.3.

4.3 The Krishnan-Hübner method as aggregation-disaggregation

To begin with, let us apply the aggregation-disaggregation method of Schweitzer and Kindle described in Section 4.1 in a Markov decision model that has been uniformized according to the transformation given in Section 2.2.1. Again, we do not explicitly denote the policy R that is being evaluated. Let $\tau \in (0, \min_{i\in S} \tau_i(R_i))$ be the uniformization parameter, in terms of which the transition probability matrix of the uniformized process is $\bar{P} = \tau Q + I$. The parameters of the uniformized process are indicated by a bar above the parameter symbol.

Our motivation for the uniformization is, that in the multiservice Erlang link model uniformization leads to transition probabilities of the following simple form, which is linear in state space coordinates:

$$\bar{p}_{\mathbf{ij}}(R_{\mathbf{i}}) = \begin{cases} \tau\lambda_k & \text{if } \mathbf{j} = \mathbf{i} + \mathbf{e}_k \text{ for some } k \in R_{\mathbf{i}}, \\ \tau i_k \mu_k & \text{if } \mathbf{j} = \mathbf{i} - \mathbf{e}_k \text{ for some } k \in \mathcal{K}, \\ 1 - \tau\big(\sum_{k\in\mathcal{K}} i_k\mu_k + \sum_{k\in R_{\mathbf{i}}} \lambda_k\big) & \text{if } \mathbf{j} = \mathbf{i}, \text{ and} \\ 0 & \text{otherwise.} \end{cases} \tag{4.16}$$

Without uniformization the transition probabilities given by $p_{ij} = 1_{i\neq j} q_{ij}(R_i) / \sum_{j\neq i} q_{ij}(R_i)$ are rational functions of state space coordinates, making it considerably more complicated to sum transition probabilities over subsets of the state space.

We set the cost rates $\bar{\mathbf{r}}$ in the uniformized process as equal to those of the original process, and by definition the expected state times $\bar{\tau}_i$ in the uniformized process are identical to unity. Thus the semi-Markov form (4.1) of Howard equations in the uniformized process is

$$\bar{\mathbf{v}} = \bar{\mathbf{r}} - \bar{g}\mathbf{1} + \bar{P}\bar{\mathbf{v}}, \tag{4.17}$$

and again we add the constraint

$$\bar{v}_m = a \tag{4.18}$$

analogous to (4.2). The following proposition expresses aggregation-disaggregation on the uniformized model in terms of the parameters and variables of the original model.

Proposition 4.1. *The aggregation-disaggregation equations* (4.4)–(4.11) *of Schweitzer and Kindle, applied on the uniformized Markov decision model are equivalent to the aggregated equations*

$$\hat{r}_D - g + \sum_{E\in\mathcal{D}} \hat{q}_{DE}\hat{v}_E = 0 \quad \textit{for all } D \in \mathcal{D}, \tag{4.19}$$

where

$$\hat{v}_D = \sum_{i\in D} w_i v_i \quad D \in \mathcal{D}, \tag{4.20}$$

$$\hat{r}_D = \sum_{i\in D} w_i r_i, \quad D \in \mathcal{D}, \tag{4.21}$$

and

$$\hat{q}_{DE} = \frac{\sum_{i\in D}\sum_{j\in E} w_i q_{ij} v_j}{\sum_{j\in E} w_j v_j}, \quad D, E \in \mathcal{D}, \tag{4.22}$$

with the constraint

$$\hat{v}_{D_m} = \tau a \tag{4.23}$$

and the disaggregation equations

$$-\sum_{j\in D} q_{ij}v_j = r_i - g + \sum_{\substack{E\in\mathcal{D}\\E\neq D}} q^{\#}_{iE}\hat{v}_E, \quad \textit{for all } i\in D,\ D\in\mathcal{D}, \tag{4.24}$$

where

$$q^{\#}_{iE} = \frac{\sum_{j\in E} q_{ij}v_j}{\sum_{j\in E} w_j v_j} \quad i\in S,\ E\in\mathcal{D}. \tag{4.25}$$

Proof. Observe first that the semi-Markov form (4.17) of the Howard equations of the uniformized process is equivalent to

$$\mathbf{0} = \mathbf{r} - g\mathbf{1} + \tau Q\bar{\mathbf{v}}, \tag{4.26}$$

which follows by substituting the definitions $\bar{P} = \tau Q + I$ and $\bar{\mathbf{r}} = \mathbf{r}$, as well as $\bar{g} = g$ which is a consequence of the fact that uniformization does not alter stationary balance probabilities. By *defining* $\hat{v}_{D_m}$ as in (4.23), it follows that the relative value vectors in the uniformized and original processes are related by $\bar{\mathbf{v}} = \frac{1}{\tau}\mathbf{v}$.

In the following, the aggregation-disaggregation parameters derived for the uniformized model are indicated by a bar above the symbol, similarly to the parameters of the uniformized model; an exception is made for the aggregation weights w_i, $i\in S$, which are identical in the uniformized and original models.

Equation (4.8) yields

$$\begin{aligned}\hat{\bar{p}}_{DE} = \frac{\sum_{i\in D}\sum_{j\in E} w_i\bar{p}_{ij}\bar{v}_j}{\sum_{j\in E} w_j\bar{v}_j} &= \frac{\sum_{i\in D}\sum_{j\in E} w_i(1_{i=j}+\tau q_{ij})\frac{1}{\tau}v_j}{\sum_{j\in E} w_j\frac{1}{\tau}v_j}\\ &= 1_{D=E}\frac{\sum_{i\in D} w_i\frac{1}{\tau}v_j}{\sum_{j\in E} w_j\frac{1}{\tau}v_j} + \tau\frac{\sum_{i\in D}\sum_{j\in E} w_i q_{ij}v_j}{\sum_{j\in E} w_j v_j}\\ &= 1_{D=E} + \tau q_{DE} \quad \text{for all } D,E\in\mathcal{D},\end{aligned} \tag{4.27}$$

where q_{DE} is defined by (4.22). From equation (4.5) we get

$$\hat{\bar{v}}_D = \sum_{i\in D} w_i\bar{v}_i = \sum_{i\in D} w_i\frac{1}{\tau}v_i = \frac{1}{\tau}\hat{v}_D \quad \text{for all } D\in\mathcal{D}, \tag{4.28}$$

where $\hat{v}_D$ is defined by (4.20), and we have trivially $\hat{\bar{c}}_D = \hat{r}_D$ and $\hat{\bar{\tau}}_D = 1$ for all $D\in\mathcal{D}$, since $\bar{\tau}_i = 1$ for all $i\in S$.

Substituting the above results in the aggregation equations (4.4) of the uniformized model, we get

$$\frac{1}{\tau}\hat{v}_D = \hat{r}_D - g + \sum_{E\in\mathcal{D}}(1_{D=E} + \tau\hat{q}_{DE})\frac{1}{\tau}\hat{v}_E \quad \text{for all } D\in\mathcal{D}, \tag{4.29}$$

which is readily found equivalent to (4.19). It remains to show the equivalence of the disaggregation aspect.

Equations (4.11) and (4.25) are connected by

$$\bar{p}^{\#}_{iE} = \frac{\sum_{j\in E} \bar{p}_{ij}\bar{v}_j}{\sum_{j\in E} w_j\bar{v}_j} = \frac{\sum_{j\in E}(1_{i=j}+\tau q_{ij})\frac{1}{\tau}v_j}{\sum_{j\in E} w_j\frac{1}{\tau}v_j} = \tau\frac{\sum_{j\in E} q_{ij}v_j}{\sum_{j\in E} w_j v_j} = \tau q^{\#}_{iE}$$
$$\text{for all } i \in D \in \mathcal{D} \text{ and } E \in \mathcal{D} \text{ such that } E \neq D. \quad (4.30)$$

Substituting the above results in the disaggregation equations (4.10), we get

$$\frac{1}{\tau}v_i - \sum_{j\in D}(1_{i=j}+\tau q_{ij})\frac{1}{\tau}v_j = r_i - g + \sum_{\substack{E\in\mathcal{D}\\E\neq D}} \tau q^{\#}_{iE}\frac{1}{\tau}\hat{v}_E \quad \text{for all } i \in D,\ D \in \mathcal{D}, \quad (4.31)$$

which is equivalent to (4.24). This completes the proof. □

Observe that the uniformization parameter τ was eliminated everywhere except in the constraint (4.23).

The proof of Proposition 4.1 only uses the fact that the relative value vectors are related by $\bar{\mathbf{v}} = \frac{1}{\tau}\mathbf{v}$. Thus when the the aggregation parameters $\hat{r}_D$, $\hat{q}_{DE}$, and $q^{\#}_{iE}$ are computed from the current relative value estimate in the iterative aggregation-disaggregation algorithm, the resulting inexact forms of the aggregated equations (4.4) and (4.19), and the disaggregation equations (4.10) and (4.24) will be equivalent, provided that the relative value estimates satisfy the same relation $\bar{\mathbf{v}} = \frac{1}{\tau}\mathbf{v}$. It follows that on the uniformized process the aggregation-disaggregation algorithm of Schweitzer and Kindle can be equivalently formulated in terms of the alternative aggregated equations (4.19) and disaggregation equations (4.24).

We are now all set to define the special case of the iterative aggregation-disaggregation algorithm that is equivalent to the Krishnan-Hübner method. Let $v_i^{(0)}$, $i \in S$, denote the initial relative value estimates in terms of which the aggregation parameters $\hat{q}_{DE}$ and $q^{\#}_{iE}$ are evaluated on the first iteration of the iterative aggregation-disaggregation algorithm. Suppose that the initial relative value estimates are constant on the state classes $\mathcal{D}$, that is $i, j \in D \in \mathcal{D}$ implies $v_i^{(0)} = v_j^{(0)}$. Then the relative values cancel out in the aggregation parameters, yielding

$$\hat{q}_{DE} = \sum_{i\in D} w_i \sum_{j\in E} q_{ij}, \quad (4.32)$$

and

$$q^{\#}_{iE} = \sum_{j\in E} q_{ij}. \quad (4.33)$$

Let us now apply the alternative formulation of the iterative aggregation-disaggregation method on the multiservice link model to evaluate the relative values and the average cost rate of the complete sharing policy R^{CS}, defining the state partitioning by

$$\mathcal{D} = \{\Omega(c) \mid c = 0, 1, \ldots, C\}, \quad (4.34)$$

and using the stationary balance probabilities of the complete sharing policy as the aggregation weights, that is $w_i = \pi_i(R^{\mathrm{CS}})/\sum_{i\in D}\pi_i(R^{\mathrm{CS}})$ for $i \in \Omega(c)$. Let the initial relative value estimates be state-class-wise constant, so that equations (4.32)–(4.33) apply. Then the aggregated transition rates (4.32) can be written as

$$\hat{q}_{DE} = \sum_{\mathbf{i}\in D} \mathrm{P}[I^{R^{\mathrm{CS}}} = \mathbf{i} \mid I^{R^{\mathrm{CS}}} \in D] \sum_{\mathbf{j}\in E} q_{\mathbf{ij}}$$
$$= \mathrm{E}\Big[\sum_{\mathbf{j}\in D} q_{I^{R^{\mathrm{CS}}}\mathbf{j}}(R^{\mathrm{CS}}) \,\Big|\, I^{R^{\mathrm{CS}}} \in E\Big] \quad \text{for all } D, E \in \mathcal{D}, \quad (4.35)$$

and the aggregated cost rates (4.21) become

$$\hat{r}_D = \sum_{\mathbf{i} \in D} \mathrm{P}[I^{R^{\mathrm{CS}}} = \mathbf{i} \mid I^{R^{\mathrm{CS}}} \in D] r_{\mathbf{i}} = \mathrm{E}\big[r_{I^{R^{\mathrm{CS}}}} \;\big|\; I^{R^{\mathrm{CS}}} \in \Omega(c)\big]. \tag{4.36}$$

These are equivalent to the transition rates (4.12) and the cost rates (4.13) of the Krishnan-Hübner aggregated system, and we conclude that the aggregation step of the first iteration of the iterative aggregation-disaggregation algorithm in fact solves the aggregated Howard equations (4.14) of the Krishnan-Hübner method.

Lea and Ke [31] discuss similar simple state aggregation in the single- and multidimensional Erlang models, but they compute the aggregated model parameters directly as weighted sums over the states in each aggregated set, thus restricting their method to small state spaces. Their aggregation equations are essentially equivalent to equations (4.32)–(4.33), with the state weights given by the stationary balance probabilities of the complete sharing policy. Thus the method of Lea and Ke can also be interpreted as the aggregation part of the aggregation-disaggregation method of Schweitzer and Kindle, with the initial relative value estimates constant over the state classes. Lea and Ke propose several different kinds of state partitions, but of these the only kind applicable to multidimensional state spaces is the same aggregation by the number of occupied trunks as used by Krishnan and Hübner. Lea and Ke do note that to make the aggregated Howard equations yield exact results, the relative values of states in each aggregated set must be equivalent.

4.3.1 Extensions to the Krishnan-Hübner method

The connection of the method of Krishnan and Hübner with the aggregation-disaggregation algorithm of Schweitzer and Kindle suggests the possibility of using the disaggregation equations (4.24) to improve the Krishnan-Hübner relative value estimates. Since the state partitioning (4.34) implies that there are no in-class transitions between states of the same class, the iterated sum on the left-hand side of (4.24) is zero, and the disaggregation equations reduce to

$$v_{\mathbf{i}} = \tau_{\mathbf{i}} \Big(r_{\mathbf{i}} - g + \sum_{k \in \mathcal{K}} \mu_k i_k \hat{v}_{D(\mathbf{i} - \mathbf{e}_k)} + \sum_{k \in R_{\mathbf{i}}^{\mathrm{CS}}} \lambda_k \hat{v}_{D(\mathbf{i} + \mathbf{e}_k)} \Big) \quad \text{for } \mathbf{i} \in \Omega \tag{4.37}$$

where $D(\mathbf{i})$ denotes the state class $D \in \mathcal{D}$ such that $\mathbf{i} \in D$. Clearly the disaggregated value estimates (4.37) can be computed independently for each $\mathbf{i} \in \Omega$, and thus they can be used in the real-time policy improvement procedure. A simpler interpretation of equation (4.37) is to see it as solving $v_{\mathbf{i}}$ from one of the Howard equations, in terms of the surrounding states, and then substituting the aggregated values for the relative values of the surrounding states.

Another possible improvement on the Krishnan-Hübner method is to use different state weights in computing the transition rates of the aggregated system. While this modification precludes the use of Kaufman-Roberts recursion, the convolution algorithm allows practical computation of the aggregated transition rates for arbitrary product form state weights, although the computational complexity of the method does increase. Computing the aggregated transition rates via the Kaufman-Roberts recursion as proposed by Krishnan and Hübner takes $O(KC)$ operations, while using the convolution algorithm naively requires as much as $O(K^2C^2)$ operations. The complexity of the convolution approach can be reduced to $O(K \log(K) C \log C)$ operations by using a transformation method for the individual convolutions and by arranging the computations for the different traffic classes in a binary tree so as to be able to reuse the results of previous convolutions; these kinds of optimizations of the convolution algorithm are discussed by Tsang and Ross [52]. However, when

the states in each state class are weighted equally as suggested by Schweitzer and Kindle, the recursion formulas developed in Section 5.3 allow computing the aggregated transition rates in $O(KC)$ operations.

The relative value estimates provided by the disaggregation equations (4.37), as well as the variant of Krishnan-Hübner method with equal weighting, with and without disaggregation, were evaluated numerically in a few link models and the results are recounted in Chapter 6. An entirely different extension is to continue the policy iteration procedure in the aggregated Markov model beyond the first iteration; this is also considered in Chapter 6.

4.4 On the accuracy of the Krishnan-Hübner method

In their article Krishnan and Hübner do not treat the accuracy of the method. Moreover, no results giving useful error bounds for a single iteration of aggregation-disaggregation of linear systems were found in the literature. However, we can make some basic observations on the subject.

Let us first express the Krishnan-Hübner aggregation in matrix notation. The N by $C+1$ matrix X with elements

$$x_{\mathbf{i}c} = 1_{\mathbf{i}\in\Omega(c)} \quad \mathbf{i} \in \Omega,\ c = 0, 1, \ldots, C, \tag{4.38}$$

expands aggregated values into the full state space so that $X\hat{\mathbf{v}}$ is the N-vector of class-wise constant relative value estimates corresponding to the $(C+1)$-vector of aggregated values $\hat{\mathbf{v}}$. Second, the $C+1$ by N matrix T with elements

$$t_{c\mathbf{i}} = \mathrm{P}[I^{R^{\mathrm{CS}}} = \mathbf{i} \mid I^{R^{\mathrm{CS}}} \in \Omega(c)], \quad c = 0, 1, \ldots, C,\ \mathbf{i} \in \Omega, \tag{4.39}$$

is the actual aggregation mapping that transforms the Howard equations into a space of a tractable number of dimensions. In terms of these matrices the aggregated Howard equations (4.14) can be written as

$$T\mathbf{r} - g\mathbf{1} + TQX\hat{\mathbf{v}} = \mathbf{0}. \tag{4.40}$$

Suppose now that the relative values of a multiservice link are given by

$$\mathbf{v} = X\check{\mathbf{v}} + \boldsymbol{\delta}, \tag{4.41}$$

where $\boldsymbol{\delta}$ is in some sense small, which may imply that the relative values are nearly constant on the state classes $\Omega(c)$. Then the Howard equations yield

$$T\mathbf{r} - g\mathbf{1} + TQ(X\check{\mathbf{v}} + \boldsymbol{\delta}) = \mathbf{0}. \tag{4.42}$$

The values of g in both (4.40) and (4.42) are equal which follows from the fact that the stationary balance probabilities of TQX correspond exactly to the probabilities of the system being in each aggregated set $\Omega(c)$. Now subtracting (4.42) from (4.40) gives

$$TQX(\hat{\mathbf{v}} - \check{\mathbf{v}}) = TQ\boldsymbol{\delta}. \tag{4.43}$$

From this it follows that if $\boldsymbol{\delta}$ is close to the null space of TQ, then $\hat{\mathbf{v}} - \check{\mathbf{v}}$ must also be close to the null space of TQX. Since TQX is an infinitesimal generator matrix with a single irreducible set, its null space is the span of $\mathbf{1}$. In particular, if $\boldsymbol{\delta} = \mathbf{0}$, then the Krishnan-Hübner method gives the exact relative values up to a constant, as already noted by Lea and Ke [31]. Obviously $\boldsymbol{\delta} = \mathbf{0}$ will be true when the traffic classes are indistinguishable; it has also been verified that this is possible for choice parameter values in some other special cases.

4.5 Bounds on the average cost rate of the produced policy

Proposition 2.8 gives bounds for the average cost rate of the policy produced in policy improvement, using arbitrary relative value estimates. These bounds can be computed efficiently when the relative value estimates are constant on each state class $\Omega(c)$, as is the case for the policy produced by the original Krishnan-Hübner method.

Suppose that the single link connection admission policy R is produced in the policy improvement step of the policy iteration algorithm, using relative value estimates $\tilde{\mathbf{v}}$ such that $\tilde{v}_{\mathbf{i}} = \hat{v}_c$ for all $\mathbf{i} \in \Omega(c)$ and $c = 0, 1, \ldots, C$. By Proposition 2.8 the average cost rate $g(R)$ of the policy is bounded from above by

$$\begin{aligned} M_n &= \max_{\mathbf{i}\in\Omega} \min_{a\in A(\mathbf{i})} r_{\mathbf{i}}(a) + \sum_{\mathbf{j}\in\Omega} q_{\mathbf{ij}}(a)\tilde{v}_{\mathbf{j}} \\ &= \max_{c=0,\ldots,C} \max_{\mathbf{i}\in\Omega(c)} \sum_{k\in\mathcal{K}} i_k\mu_k(\tilde{v}_{\mathbf{i}-\mathbf{e}_k} - \tilde{v}_{\mathbf{i}}) + \sum_{\substack{k\in\mathcal{K} \\ \mathbf{i}+\mathbf{e}_k\notin\Omega}} h_k\lambda_k + \sum_{\substack{k\in\mathcal{K} \\ \mathbf{i}+\mathbf{e}_k\in\Omega}} \lambda_k \min\left\{h_k,\ \tilde{v}_{\mathbf{i}+\mathbf{e}_k} - \tilde{v}_{\mathbf{i}}\right\}. \end{aligned} \tag{4.44}$$

By noting that only the state space coordinates i_k vary within each state class $\Omega(c)$ while the rest of the maximized expression stays constant, we can rearrange this as

$$M_n = \max_{c=0,\ldots,C} \left\{ \sum_{\substack{k\in\mathcal{K} \\ c+b_k>C}} h_k\lambda_k + \sum_{\substack{k\in\mathcal{K} \\ c+b_k\le C}} \lambda_k \min\left\{h_k,\ \hat{v}_{c+b_k} - \hat{v}_c\right\} + \max_{\mathbf{i}\in\Omega(c)} \sum_{k\in\mathcal{K}} i_k\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) \right\}. \tag{4.45}$$

The inner maximization here is maximizing a linear function of state space coordinates on a plane of the state space; the objective value can be easily bounded from above by considering the same maximization on the corresponding plane of $\mathbb{R}^K$, where the maximum will be reached in one of the extremal points of the domain:

$$\begin{aligned} \max_{\mathbf{i}\in\Omega(c)} \sum_{k\in\mathcal{K}} i_k\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) &= \max_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c}} \sum_{k\in\mathcal{K}} i_k\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) \\ &\le \max_{\substack{\mathbf{i}\in\mathbb{R}^K \\ \mathbf{i}\ge\mathbf{0} \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c}} \sum_{k\in\mathcal{K}} i_k\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) = \max_{k\in\mathcal{K}} \frac{c}{b_k}\mu_k(\hat{v}_{c-b_k} - \hat{v}_c). \end{aligned} \tag{4.46}$$

Substituting this into (4.45), we get the efficiently computable upper bound

$$M_n = \max_{c=0,\ldots,C} \left\{ \sum_{\substack{k\in\mathcal{K} \\ c+b_k>C}} h_k\lambda_k + \sum_{\substack{k\in\mathcal{K} \\ c+b_k\le C}} \lambda_k \min\left\{h_k,\ \hat{v}_{c+b_k} - \hat{v}_c\right\} + \max_{k\in\mathcal{K}} \frac{c}{b_k}\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) \right\} \tag{4.47}$$

for the average cost rate of the single link admission control policy produced by the Krishnan-Hübner method. Entirely analogously we can transform the lower bound of Proposition 2.8 into the practically computable form

$$m_n = \min_{c=0,\ldots,C} \left\{ \sum_{\substack{k\in\mathcal{K} \\ c+b_k>C}} h_k\lambda_k + \sum_{\substack{k\in\mathcal{K} \\ c+b_k\le C}} \lambda_k \min\left\{h_k,\ \hat{v}_{c+b_k} - \hat{v}_c\right\} + \min_{k\in\mathcal{K}} \frac{c}{b_k}\mu_k(\hat{v}_{c-b_k} - \hat{v}_c) \right\}. \tag{4.48}$$

In principle these bounds make it possible to produce a connection admission policy that is guaranteed to be at least as good as the complete sharing policy, since the Krishnan-Hübner method computes the average cost rate of the complete sharing policy exactly and one can then revert to the complete sharing policy whenever the upper bound (4.47) exceeds the average cost rate of the complete sharing policy. Unfortunately for these bounds to be even close to tight in practice, the relative value estimates have to be more accurate than is usually achieved by the Krishnan-Hübner method; this is discussed further in Chapter 6.

Chapter 5

Least-squares estimation of relative values

The problem of finding such a representation of the relative values of a policy that can be stored in a modest amount of memory can be considered as a choice among a class of parametrized representations. The choice can be made in a sense optimally by least squares fitting. This approach has not appeared in the telecommunications literature, but Schweitzer and Seidmann [46] discuss least squares fitting in queuing networks, with the implicit assumption that the state space is small enough to allow operations whose time requirement is linear in the number of states. The contribution of this thesis is that we present efficient methods for performing least squares fitting on multiservice links.

5.1 Review of previous work

In queuing networks the system states are characterized by vectors $\mathbf{i} \in \mathbb{N}^K$ like in multiservice links. Schweitzer and Seidmann consider representing the relative values of a policy in the form

$$v_{\mathbf{i}}(\boldsymbol{\alpha}) = \sum_{j=1}^{J} \alpha_j f_j(\mathbf{i}) \quad \text{for all } \mathbf{i} \in S, \tag{5.1}$$

where the J basis functions f_j are given and the coefficients α_j, $j = 1, 2, \ldots, J$, are to be estimated so as to approximate the optimal relative values, and S is the system state space. Specifically they recommend second and third order polynomials in state space coordinates, that is the basis functions

$$f_0(\mathbf{i}) = 1, \tag{5.2}$$

$$f_k(\mathbf{i}) = i_k, \quad k = 1, 2, \ldots, K, \tag{5.3}$$

$$f_{kl}(\mathbf{i}) = i_k i_l, \quad k = 1, 2, \ldots, K,\ l = k, k+1, \ldots, K, \tag{5.4}$$

and optionally also

$$f_{klm}(\mathbf{i}) = i_k i_l i_m, \quad k = 1, 2, \ldots, K,\ l = k, \ldots, K,\ m = l, \ldots, K, \tag{5.5}$$

which they found to give good a posteriori fits to optimal relative values, with relative errors of only a few per cent. Schweitzer and Seidmann discuss using the relative value representation (5.1) in three policy optimization methods, in both discounted and undiscounted

Markov control problems. We only show the equations of the undiscounted (infinite horizon) case.

As the first approximate method, Schweitzer and Seidmann replace the relative values in the dual program (2.49) of the standard linear programming approach by the approximations (5.1) and then develop a modified primal program in N variables with J constraints, where N is the number of system states. Unfortunately a linear program of even this size is infeasibly large for application on multiservice links.

As another alternative, the authors discuss global least squares fitting of the approximation (5.1) to the optimal relative values by

$$\min_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^J \\ g\in\mathbb{R}}} \sum_{\mathbf{i}\in S} w_{\mathbf{i}} \left\{ \min_{a\in A(\mathbf{i})} c_{\mathbf{i}}(a) - g\tau_{\mathbf{i}}(a) + \sum_{\mathbf{j}\in S} p_{\mathbf{ij}}(a) v_{\mathbf{j}}(\boldsymbol{\alpha}) - v_{\mathbf{i}}(\boldsymbol{\alpha}) \right\}^2 \tag{5.6}$$

where $c_{\mathbf{i}}(a) = r_{\mathbf{i}}(a)\tau_{\mathbf{i}}(a)$ and $w_{\mathbf{i}}$, $\mathbf{i} \in S$, are arbitrary weights. The inner minimization corresponds to the policy improvement stage of the policy iteration algorithm, and the outer minimization is optimizing the weighted sum of squared errors in the fit. The objective function of the outer minimization problem is a piecewise quadratic function of the coefficients $\boldsymbol{\alpha}$ and average cost rate g; Schweitzer and Seidmann develop a projected gradient algorithm for finding coefficients α for a local minimum. Again, this requires far too many operations on each iteration and does not appear useful for the problem of optimal control on multiservice links.

The third method proposed is policy iteration where the value determination stage is replaced by the linear least-squares problem

$$\min_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^J \\ g\in\mathbb{R}}} \sum_{\mathbf{i}\in S} w_{\mathbf{i}} \Big(c_{\mathbf{i}}(R_{\mathbf{i}}) - g\tau_{\mathbf{i}}(R_{\mathbf{i}}) + \sum_{\mathbf{j}\in S} p_{\mathbf{ij}}(R_{\mathbf{i}}) v_{\mathbf{j}}(\boldsymbol{\alpha}) - v_{\mathbf{i}}(\boldsymbol{\alpha}) \Big)^2 \tag{5.7}$$

where R is the policy to be improved, and again $w_{\mathbf{i}}$, $\mathbf{i} \in S$, are arbitrary weights. This is simply minimizing the weighted residual error in a form of the Howard equations. Schweitzer and Seidmann show that the minimization problem has a unique solution if the unichain assumption holds and if the only linear combination of the basis functions that forms a vector with all components equal is the case $\boldsymbol{\alpha} = \mathbf{0}$. Unlike standard policy iteration, this method is not guaranteed to converge and the number of iterations has to be artificially limited. In the sequel we discuss how a slightly modified form of the first policy iteration of this approximation can be made sufficiently efficient to be practical on large multiservice links.

As an alternative to least squares fitting the authors also propose a Galerkin approach, where the basic assumption is that for a good fit of the relative value estimates the Howard equations are approximately satisfied:

$$v_{\mathbf{i}}(\boldsymbol{\alpha}) \approx c_{\mathbf{i}}(R_{\mathbf{i}}) - g + \sum_{\mathbf{j}\in S} p_{\mathbf{ij}}(R_{\mathbf{i}}) v_{\mathbf{j}}(\boldsymbol{\alpha}) \quad \text{for all } \mathbf{i} \in S. \tag{5.8}$$

Then the parameters $\boldsymbol{\alpha}$ can be estimated by requiring that the dot products of both sides of this linear system with the weighted basis vectors $(w_{\mathbf{i}} f_j(\mathbf{i}))_{\mathbf{i}\in S}$ be equal for all $j = 1, 2, \ldots, J$. This approach is feasible on multiservice links, but is not pursued in this thesis; however the methods used in least squares fitting are applicable to efficient computation of the dot products as well.

For evaluating the accuracy of the relative value estimation methods after the fact, Schweitzer and Seidmann refer to a result similar to Proposition 2.8. Unfortunately it does not appear to be practical to compute these bounds for other kinds of relative value estimates

than the state-class-wise constant estimates for which an effective computation method was developed in Section 4.5.

Curiously, no further development on these ideas on polynomial approximation and least squares fitting was found in the literature.

5.2 Construction of normal equations

Let us now proceed to the details of least squares fitting of relative values on large multiservice links. Applying the form (5.7) of relative value fitting directly on large link models appears to be impracticable because the state times $\tau_{\mathbf{i}}(a)$ are rational functions of the state space coordinates. To avoid this difficulty we develop the details of the relative value fitting for the form of Howard equations based on transition rates instead of transition probabilities; the resulting method is equivalent to applying (5.7) after performing uniformization of the Markov decision model.

For convenience, let us express the relative value representation (5.9) in matrix form as

$$\mathbf{v} = X\boldsymbol{\alpha}, \tag{5.9}$$

where the N by J expansion matrix X is defined elementwise by $x_{\mathbf{i},j} = f_j(\mathbf{i})$, and $\boldsymbol{\alpha}$ is the J-vector of basis function coefficients. We note that for the relative value estimates to be useful in real-time policy improvement, it is desirable that the number of parameters J is not too large and that the functions f_j are fast to compute for any $\mathbf{i} \in \Omega$.

Let R be a connection admission policy for a multiservice link with K traffic classes. In order to simplify the notation, we do not explicitly indicate the dependency of the cost rate vector $\mathbf{r}$ and the infinitesimal generator matrix Q on the policy. Substituting the relative value vector (5.9) in the Howard equations yields

$$\mathbf{r} - g\mathbf{1} + QX\boldsymbol{\alpha} = \mathbf{0}, \tag{5.10}$$

where we have an additional free variable, the average cost rate g. Let us for the moment suppose that g is known; for example for certain kinds of policies it could be independently and relatively efficiently computed by Kaufman-Roberts recursion or the convolution algorithm. To determine the coefficients $\boldsymbol{\alpha}$, we require that the approximated relative values satisfy the Howard equations as accurately as possible in the least squares sense.

As shown in any linear algebra textbook (see eg [51, Lecture 11]), the coefficient vector $\boldsymbol{\alpha}$ minimizing the Euclidean norm of the left-hand side of (5.10) can be determined as the solution of the normal equations

$$X^{\mathrm{T}}Q^{\mathrm{T}}QX\boldsymbol{\alpha} = X^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1} - \mathbf{r}). \tag{5.11}$$

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 relatively small. It turns out that for many simple forms of the basis functions f_j, $j = 1, 2, \ldots, J$, there are effective algorithms for computing the coefficient matrix and right-hand side vector of the system of normal equations, even when the link state space is so large that handling the matrices Q and X directly is impracticable.

Let us now discuss the problem of finding a least squares solution to the Howard equations when the average cost rate g is not known. When (5.10) is rewritten as

$$\begin{pmatrix} -\mathbf{1} & QX \end{pmatrix} \begin{pmatrix} g \\ \boldsymbol{\alpha} \end{pmatrix} = -\mathbf{r}, \tag{5.12}$$

it is clear that this is an analogous least squares problem with $J+1$ free parameters. The normal equations for this form of the system are

$$\begin{pmatrix} N & -\mathbf{1}^{\mathrm{T}}QX \\ -X^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{1} & X^{\mathrm{T}}Q^{\mathrm{T}}QX \end{pmatrix} \begin{pmatrix} g \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} \mathbf{1}^{\mathrm{T}} \\ -X^{\mathrm{T}}Q^{\mathrm{T}} \end{pmatrix} \mathbf{r}. \tag{5.13}$$

The system contains $J+1$ equations in $J+1$ variables, and the coefficient matrix and the right-hand-side vector of the system can be computed efficiently by the same methods as for the system (5.11).

To numerically solve (5.11) or (5.13), we need to compute the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ and the vectors $X^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{y}$ where $\mathbf{y}$ is $\mathbf{1}$, $\mathbf{r}$, or $g\mathbf{1}-\mathbf{r}$. Constructing these for specific forms of the basis functions f_j is treated in the following sections, but first we shall develop the basic recursion formulas in terms of which the elements of the coefficient matrices and right-hand sides of the systems of normal equations will be expressed.

5.3 Recursion formulas

In most of the approximation methods discussed it is necessary to compute sums of the form $s(c) = \sum_{\mathbf{i}\in\Omega(c)} f(\mathbf{i})$ where f is of some simple form. A relatively general and effective method is provided by the convolution algorithm discussed in Section 3.5, but it requires effort of the order of at least $O(KC\log C)$ operations, where C is the link capacity. Fortunately it is possible to develop more efficiently computable recursion formulas for some particularly simple forms of the summed function f. The following is an extension of Buzen's ideas in his treatment [5] of two special cases of the convolution algorithm.

Let us first consider sums of the form

$$s(c,k) = \sum_{\substack{\mathbf{i}\in\mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)}=c}} \prod_{l=1}^{k} a_l^{i_l} \quad \text{for } c = 0,1,\ldots,C \text{ and } k = 1,2,\ldots,K, \tag{5.14}$$

where $\mathbf{b}^{(k)} = (b_1\ b_2\ \cdots\ b_k)^{\mathrm{T}}$ is a truncated vector of traffic class trunk requirements, and a_k, $k = 1,2,\ldots,K$, are arbitrary nonzero numbers. Note that $s(c,k)$ can be interpreted as the sum of a simple product form function over the state class $\Omega(c)$ on a multiservice link with k traffic classes. This case was already discussed by Buzen [5] for queueing networks where $b_k = 1$ for all k, and the generalization for multiservice links with varying trunk requirements is straightforward. By algebraic manipulations we can develop a recursive formulation for (5.14) as follows:

$$\begin{aligned} s(c,k) &= \sum_{\substack{\mathbf{i}\in\mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)}=c}} \prod_{l=1}^{k} a_l^{i_l} = \sum_{\substack{\mathbf{i}\in\mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)}=c \\ i_k=0}} \prod_{l=1}^{k} a_l^{i_l} + \sum_{\substack{\mathbf{i}\in\mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)}=c \\ i_k>0}} \prod_{l=1}^{k} a_l^{i_l} \\ &= \sum_{\substack{\mathbf{i}\in\mathbb{N}^{k-1} \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k-1)}=c}} \prod_{l=1}^{k-1} a_l^{i_l} + a_k \sum_{\substack{\mathbf{j}\in\mathbb{N}^k \\ \mathbf{j}^{\mathrm{T}}\mathbf{b}^{(k)}=c-b_k}} \prod_{l=1}^{k} a_l^{j_l} = s(c,k-1) + a_k s(c-b_k,k). \end{aligned} \tag{5.15}$$

The change of variable from $\mathbf{i}$ to $\mathbf{j}$ is treated more rigorously in the proof of the Kaufman-Roberts recursion formula on page 33. To initiate the recursion, we set $s(0,k) = 1$ for all k and $s(c,1) = 1_{b_1|c}\, a_1^{c/b_1}$ for all c; these follow directly from the definition (5.14) of $s(c,k)$. When the values $s(c,k)$ are arranged in a $C+1$ by K table, as illustrated in Table

	1	2	3	$\cdots$	$k-1$	k	$\cdots$	K
0	1	1	1	$\cdots$	1	1	$\cdots$	1
1	$1_{b_1=1}a_1$	$1_{b_1=1}a_1+$ $1_{b_2=1}a_2$						$s(1,K)$
2	$1_{b_1\|2}a_1^{b1/2}$		$\ddots$					$\vdots$
$\vdots$	$\vdots$							
$c-b_k$						$s(c-b_k,k)$		
$\vdots$	$\vdots$					$a_k\cdot\downarrow$ $\vdots$		$\vdots$
c	$1_{b_1\|c}a_1^{b1/c}$				$s(c,k-1)\rightarrow$	$s(c,k)$		$s(c,K)$
$\vdots$	$\vdots$							$\vdots$
C	$1_{b_1\|C}a_1^{b1/C}$	$\cdots$					$\cdots$	$s(C,K)$

Table 5.1: Recursive computation of $s(c,k)$ in tabular form. The table can be filled in any order, for example from left to right and top to bottom, as long as the elements above and to the left of a new element are computed before the new element.

5.1, the value of $s(c,k)$ is the sum of the value directly on the left, and the value b_k rows up. Computing the sums $s(c,k)$ for all $c = 0,1,\ldots,C$ and $k = 1,2,\ldots,K$ requires less than $C(K-1)$ additions and the same number of multiplications. Usually we are only interested in a particular multiservice link with a fixed number of traffic classes K, and it is not necessary to store the values $s(c,k)$ for $k < K$; when the tabular computation is done from left to right, that is by using k as the outer and c as the inner loop index, then at any particular stage of the computation space is needed for no more than C numbers.

In this thesis we only have use for the above procedure in computing sums of the form

$$\gamma_0(c,k) = \sum_{\substack{\mathbf{i}\in\mathbb{N}^k \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}^{(k)}=c}} 1 \quad \text{for } c = 0,1,\ldots,C \text{ and } k = 1,2,\ldots,K, \tag{5.16}$$

which is a special case of (5.14) with the parameters a_k all equal to unity. Observe that $\gamma_0(c,K)$ is the cardinality of the set $\Omega(c)$ of multiservice link states, and thus the total number of system states is given by $N = \sum_{c=0}^{C}\gamma_0(c,K)$. Since there is no need to multiply by the a_k's, computing $\gamma_0(c,k)$ for all $c = 0,1,\ldots,C$ and $k = 1,2,\ldots,K$ requires less than $C(K-1)$ additions and no multiplications.

It turns out that a similar recursive algorithm can be developed for sums of the form

$$\gamma_{(\nu_1,\nu_2,\ldots,\nu_K)}(c) = \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c}} \prod_{l=1}^{K} i_l^{\nu_l} \quad \text{for } c = 0,1,\ldots,C \text{ and } \nu_1,\nu_2,\ldots,\nu_K \in \mathbb{N}, \tag{5.17}$$

where 0^0 is taken as 1, so that an exponent $\nu_k = 0$ indicates that the kth factor is 1 in every product form term of the sum. The recursion for (5.17) does not appear to be known in the telecommunications or queuing network literature, which is unsurprising since the usefulness and effectiveness of the recursion is limited to very specialized applications like the fitting procedures developed in this thesis.

Let m be a fixed index k such that $\nu_m > 0$; we can assume that such an index exists since $\gamma_{(0,0,\ldots,0)}(c)$ equals $\gamma_0(c,K)$ for which we already have an efficient computational

algorithm. Now (5.17) can be rewritten as

$$\gamma_{(\nu_1,\nu_2,\ldots,\nu_K)}(c) = \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c}} \prod_{l=1}^{K} i_l^{\nu_l} = \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c \\ i_m>0}} \prod_{l=1}^{K} i_l^{\nu_l} = \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c-b_m}} (i_m+1)^{\nu_m} \prod_{\substack{l=1 \\ l\neq m}}^{K} i_l^{\nu_l}$$

$$= \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c-b_m}} \Bigl(\sum_{j=0}^{\nu_m} \tbinom{\nu_m}{j} i_m^j \Bigr) \prod_{\substack{l=1 \\ l\neq m}}^{K} i_l^{\nu_l} = \sum_{j=0}^{\nu_m} \tbinom{\nu_m}{j} \sum_{\substack{\mathbf{i}\in\mathbb{N}^K \\ \mathbf{i}^{\mathrm{T}}\mathbf{b}=c-b_m}} i_m^j \prod_{\substack{l=1 \\ l\neq m}}^{K} i_l^{\nu_l}$$

$$= \sum_{j=0}^{\nu_m} \tbinom{\nu_m}{j} \gamma_{(\nu_1,\nu_2,\ldots,\nu_{m-1},j,\nu_{m+1},\ldots,\nu_K)}(c-b_m). \quad (5.18)$$

This recursive formula expresses (5.17) in terms of itself, so that on the right-hand side the argument c is replaced by a smaller number $c - b_k$, and the maximal ν_m is replaced by a number j less than or equal to ν_m. To provide the ground cases of the recursion, we set $\gamma_{(0,0,\ldots,0)}(c) = \gamma_0(c, K)$ for all c, and when ν_m is nonzero we set $\gamma_{(\nu_1,\nu_2,\ldots,\nu_K)}(c) = 0$ for $c < b_m$.

Computing the sum (5.17) for some fixed arguments $\nu_1^*, \nu_2^*, \ldots, \nu_K^*$ and c^* requires computing the sums for all $\nu_1 \le \nu_1^*, \nu_2 \le \nu_2^*, \ldots, \nu_K \le \nu_K^*$ and most $c < c^*$, which makes the recursion very inefficient when $\sum_{k=1}^{K} \nu_k$ is not small. Fortunately the recursive computations do not present any extra burden in the fitting procedures developed in this chapter since all the recursively referenced sums are needed in any case. Once values of $\gamma_{(\nu_1,\nu_2,\ldots,\nu_K)}(c)$ are known for all $c = 0, 1, \ldots, C$ and all $\nu_1 \le \nu_1^*, \nu_2 \le \nu_2^*, \ldots, \nu_K \le \nu_K^*$, computing $\gamma_{(\nu_1^*,\nu_2^*,\ldots,\nu_K^*)}(c)$ for all $c = 0, 1, \ldots, C$ takes less than $C(1 + \max_k \nu_k^*)$ additions and equally many multiplications, ignoring the computation of the binomial coefficients which can presumably be looked up in a small pre-computed table.

Since all the values in these recursion formulas are pure integers, the formulas can be implemented completely accurately as long as the numbers fit the machine number type used. However, even 64-bit integers can easily be insufficient for sums of the form (5.17) when the link capacity and the number of traffic classes are large.

5.4 Quadratic relative value functions

Schweitzer and Seidmann [46] found second and third order polynomials in the state space coordinates $(i_1, i_2, \ldots, i_K)$ to provide good fits to relative values in queuing networks. We discuss the fitting of the following polynomial basis functions:
quadratic polynomials without cross-terms

$$v_{\mathbf{i}} = \sum_{k=1}^{K} \alpha_k i_k + \sum_{k=1}^{K} \alpha_{kk} i_k^2, \quad (5.19)$$

and quadratic polynomials with cross-terms

$$v_{\mathbf{i}} = \sum_{k=1}^{K} \alpha_k i_k + \sum_{k=1}^{K} \sum_{l=k}^{K} \alpha_{kl} i_k i_l. \quad (5.20)$$

In the former case we have $2K$ parameters, and in the latter $\frac{1}{2}K(K+3)$ parameters. The polynomials do not have constant terms since the constant would not be determined by the Howard equations; in effect we are thus adding the constraint $v_{\mathbf{0}} = 0$ to keep the system

determined. Higher order polynomials could be treated analogously to the following, but as will be seen the resulting complexity of constructing the system of normal equations grows very rapidly if cross-terms are included, and without cross-terms the advantage is dubious. We develop the system of normal equations for the class (5.20) of quadratic polynomials with cross-terms, which of course includes the class (5.19).

Let us express the elements of the expansion matrix X corresponding to the quadratic approximation with cross terms as

$$x_{\mathbf{i},(k,l)} = i_k i_l \quad \text{for } \mathbf{i} \in \Omega,\ k = 1, 2, \ldots, K,\ l = 0, 1, \ldots, K, \tag{5.21}$$

where i_0 is taken as 1 for all $\mathbf{i} \in \Omega$. Using the definition (3.6) of transition rates on a multiservice link we can express elements of the matrix QX as

$$\begin{aligned}
[QX]_{\mathbf{i},(k,l)} &= \sum_{\mathbf{j}\in\Omega} q_{\mathbf{ij}} x_{\mathbf{j},(k,l)} \\
&= \sum_{n\in\mathcal{K}} i_n \mu_n x_{\mathbf{i}-\mathbf{e}_n,(k,l)} + \sum_{n\in R_{\mathbf{i}}} \lambda_n x_{\mathbf{i}+\mathbf{e}_n,(k,l)} - \Big(\sum_{n\in\mathcal{K}} i_n\mu_n + \sum_{n\in R_{\mathbf{i}}} \lambda_n\Big) x_{\mathbf{i},(k,l)} \\
&= \sum_{n\in\mathcal{K}} i_n\mu_n\big((i_k - 1_{k=n})(i_l - 1_{l=n}) - i_k i_l\big) + \sum_{n\in R_{\mathbf{i}}} \lambda_n\big((i_k + 1_{k=n})(i_l + 1_{l=n}) - i_k i_l\big).
\end{aligned} \tag{5.22}$$

This is simplified in two separate cases according to the values of k and l. When $k = l$, equation (5.22) yields

$$\begin{aligned}
[QX]_{\mathbf{i},(k,k)} &= i_k\mu_k\big((i_k-1)^2 - i_k^2\big) + 1_{k\in R_{\mathbf{i}}}\lambda_k\big((i_k+1)^2 - i_k^2\big) \\
&= i_k\mu_k(-2i_k + 1) + 1_{k\in R_{\mathbf{i}}}\lambda_k(2i_k + 1) \\
&= (-2\mu_k)i_k^2 + (\mu_k + 1_{k\in R_{\mathbf{i}}}2\lambda_k)i_k + 1_{k\in R_{\mathbf{i}}}\lambda_k \quad \text{for } k = 1, 2, \ldots, K,
\end{aligned} \tag{5.23}$$

and when $k \neq l$, we get

$$\begin{aligned}
[QX]_{\mathbf{i},(k,l)} &= i_k\mu_k\big((i_k-1)i_l - i_k i_l\big) + 1_{l\neq 0} i_l\mu_l\big(i_k(i_l-1) - i_k i_l\big) \\
&\quad + 1_{k\in R_{\mathbf{i}}}\lambda_k\big((i_k+1)i_l - i_k i_l\big) + 1_{l\in R_{\mathbf{i}}}\lambda_l\big(i_k(i_l+1) - i_k i_l\big) \\
&= (-\mu_k - 1_{l\neq 0}\mu_l) i_k i_l + 1_{k\in R_{\mathbf{i}}}\lambda_k i_l + 1_{l\in R_{\mathbf{i}}}\lambda_l i_k \\
&\qquad \text{for } k = 1, 2, \ldots, K \text{ and } l = k+1, k+2, \ldots, K.
\end{aligned} \tag{5.24}$$

These two cases are both of the general form

$$[QX]_{\mathbf{i},(k,l)} = \sum_{j_1=0}^{1}\sum_{j_2=0}^{1} \beta_{(k,l)}(j_1, j_2, \mathbf{i}) i_k^{j_1} i_l^{j_2}, \tag{5.25}$$

where the coefficients $\beta_{(k,l)}(j_1, j_2, \mathbf{i})$ are state-dependent indirectly via the action $R_{\mathbf{i}}$ and possibly also via the call arrival and completion rates if they are modelled as state-dependent. Note that for fixed k and l, only three of the four coefficients in (5.25) can be nonzero. When cross-terms are not included, $k \neq l$ implies $l = 0$ and by definition $i_l = 1$ for all $\mathbf{i}$; hence (5.25) reduces to a second-order polynomial in i_k.

In terms of (5.25), the elements of the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ can be expressed as

$$\begin{aligned}[X^{\mathrm{T}}Q^{\mathrm{T}}QX]_{(k,l),(m,n)} &= \sum_{\mathbf{i}\in\Omega}[QX]_{\mathbf{i},(k,l)}[QX]_{\mathbf{i},(m,n)} \\ &= \sum_{\mathbf{i}\in\Omega}\sum_{j_1=0}^{1}\sum_{j_2=0}^{1}\beta_{(k,l)}(j_1,j_2,\mathbf{i})\,i_k^{j_1}\,i_l^{j_2}\sum_{j_3=0}^{1}\sum_{j_4=0}^{1}\beta_{(m,n)}(j_3,j_4,\mathbf{i})\,i_m^{j_3}\,i_n^{j_4} \\ &= \sum_{j_1=0}^{1}\sum_{j_2=0}^{1}\sum_{j_3=0}^{1}\sum_{j_4=0}^{1}\sum_{\mathbf{i}\in\Omega}\beta_{(k,l)}(j_1,j_2,\mathbf{i})\,\beta_{(m,n)}(j_3,j_4,\mathbf{i})\,i_k^{j_1}\,i_l^{j_2}\,i_m^{j_3}\,i_n^{j_4}. \quad (5.26)\end{aligned}$$

At most nine of the sixteen coefficient products $\beta_{(k,l)}(j_1,j_2,\mathbf{i})\beta_{(m,n)}(j_3,j_4,\mathbf{i})$ in (5.26) can be nonzero. When cross-terms are excluded, the product $i_k^{j_1}\,i_l^{j_2}\,i_m^{j_3}\,i_n^{j_4}$ of four elements of $\mathbf{i}$ in (5.26) reduces to a product of two elements of $\mathbf{i}$ with exponents ranging from 0 to 2.

When the actions of the policy R as well as the call arrival rates λ_k and completion rates μ_k are constant over large regular regions of the state space Ω, also the coefficients β will be constant over these same regions, and the elements of the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ as given by equation (5.26) can be feasibly computable. For example, suppose that the action $R_{\mathbf{i}}$ as well as the call arrival rates λ_k and completion rates μ_k are constant over the state classes $\Omega(c)$, $c = 0, 1, \ldots, C$; then (5.26) yields

$$\begin{aligned}&[X^{\mathrm{T}}Q^{\mathrm{T}}QX]_{(k,l),(m,n)} \\ &\quad= \sum_{j_1=0}^{1}\sum_{j_2=0}^{1}\sum_{j_3=0}^{1}\sum_{j_4=0}^{1}\sum_{c=0}^{C}\beta_{(k,l)}(j_1,j_2,c)\,\beta_{(m,n)}(j_3,j_4,c)\sum_{\mathbf{i}\in\Omega(c)}i_k^{j_1}\,i_l^{j_2}\,i_m^{j_3}\,i_n^{j_4}, \quad (5.27)\end{aligned}$$

where $\beta_{(k,l)}(j_1,j_2,c)$ denotes the common value of $\beta_{(k,l)}(j_1,j_2,\mathbf{i})$ over $\mathbf{i}\in\Omega(c)$. Observe that the innermost sum $\sum_{\mathbf{i}\in\Omega(c)} i_k^{j_1}\,i_l^{j_2}\,i_m^{j_3}\,i_n^{j_4}$ is of a form efficiently computable by the recursion formulas of Section 5.3, and that the outer sums iterate over a practically manageable number of terms.

As discussed in Section 5.2, for least-squares fitting we also need the vectors $X^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{y}$ where $\mathbf{y}$ is $\mathbf{1}$, $\mathbf{r}$, or $g\mathbf{1}-\mathbf{r}$. Since $r_{\mathbf{i}}$ depends on the state $\mathbf{i}$ only through the policy action $R_{\mathbf{i}}$ and possibly also the λ_k's, the matrix-vector products are practically computable whenever the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ is. Again, we show the development for the case when $R_{\mathbf{i}}$, λ_k and μ_k are constant over $\mathbf{i}\in\Omega(c)$; for the product with $\mathbf{r}$ we get

$$\begin{aligned}[X^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{r}]_{(k,l)} = \sum_{\mathbf{i}\in\Omega}[QX]_{\mathbf{i},(k,l)}r_{\mathbf{i}} &= \sum_{j_1=0}^{1}\sum_{j_2=0}^{1}\sum_{\mathbf{i}\in\Omega}\beta_{(k,l)}(j_1,j_2,\mathbf{i})\,i_k^{j_1}\,i_l^{j_2}r_{\mathbf{i}} \\ &= \sum_{j_1=0}^{1}\sum_{j_2=0}^{1}\sum_{c=0}^{C}\beta_{(k,l)}(j_1,j_2,c)\,r_c\sum_{\mathbf{i}\in\Omega(c)}i_k^{j_1}\,i_l^{j_2}. \quad (5.28)\end{aligned}$$

The innermost sum is again efficiently computable by the recursion formulas of Section 5.3; in fact the same values of the innermost sum are already required for computing (5.27). The computation of the other matrix-vector products is entirely analogous.

Constructing all elements of the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ by (5.27) requires computing the sum $\sum_{\mathbf{i}\in\Omega(c)} i_k^{j_1}\,i_l^{j_2}\,i_m^{j_3}\,i_n^{j_4}$ for all choices of k,l,m,n such that $1\le k\le l\le m\le n\le K$ and all $j_1,j_2,j_3,j_4\in\{0,1\}$. There are $\binom{K}{4}$ such choices of the indices k,l,m,n; by the recursion formulas of Section 5.3 all the required sums can be computed in $\binom{K}{4}4C+O(K^3C) = O(K^4C)$ arithmetic operations, and the storage requirement for all the sums is of the order

$\binom{K}{4}C + O(K^3C)$. After the innermost sums of (5.27) are computed by recursion formulas, evaluating all the elements of $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ in terms of the sums takes another $O(K^4C)$ operations, and thus the total work of constructing $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ is on the order $O(K^4C)$. The remaining vectors of the system of normal equations require less work, but solving the system by direct methods takes on the order of $(K^2)^3$ operations. The total work of fitting the relative value function parameters is thus of the order $O(K^4 \max(C, K^2))$.

In case the basis functions without cross-terms are used, the complexity of the approach is significantly smaller. The work and storage requirements of computing the innermost sums of (5.27) are only of the order $O(K^2C)$ and the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ has only $(2K)^2$ elements, so that constructing it in terms of the sums takes only $O(K^2C)$ operations. Solving the system of normal equations requires then $O(K^3)$ operations, so that the total work is of the order $O(K^2C)$, provided that $K < C$. The asymptotic time complexity changes only by a constant factor when higher order polynomials without cross-terms are used. On the other hand, when fitting mth order polynomials *with* cross-terms, elements of QX will contain products of m different state space coordinates and subsequently elements of $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ will be sums of products of $2m$ state space coordinates, resulting in complexity of the order of $K^{2m}C$ for constructing the system of normal equations alone.

The discussed condition of the coefficients $\beta_{(k,l)}(j_1, j_2, \mathbf{i})$ being constant over $\mathbf{i} \in \Omega(c)$ is one of the simplest useful cases where the system of normal equations can be constructed efficiently; this case covers the complete sharing policy and all trunk reservation policies, with call arrival and completion rates that may depend on the number of occupied trunks $\mathbf{b}^{\mathrm{T}}\mathbf{i}$. When more involved state dependencies are introduced, the complexity of the approach grows rapidly. For example, when R is a general threshold policy the state space is divided into 2^K regions, each with a different action $R_{\mathbf{i}}$; handling this case may be viable by using the general convolution method instead of recursion formulas.

5.4.1 Link shadow prices from polynomial relative values

When the relative values are estimated by simple polynomial functions, the link shadow prices are polynomial functions of lower order. For the class of quadratic polynomial relative values with cross-terms (5.20) the link shadow prices reduce to

$$\begin{aligned} p_n(\mathbf{i}) &= v_{\mathbf{i}+\mathbf{e}_n} - v_{\mathbf{i}} \\ &= \sum_{k=1}^{K} \alpha_k \big((i_k + 1_{k=n}) - i_k\big) + \sum_{k=1}^{K}\sum_{l=k}^{K} \alpha_{kl} \big((i_k + 1_{k=n})(i_l + 1_{l=n})\big) - i_k i_l \\ &= \sum_{k=1}^{K} \alpha_k 1_{k=n} + \sum_{k=1}^{K}\sum_{l=k}^{K} \alpha_{kl}(1_{k=n} i_l + 1_{l=n} i_k + 1_{k=n} 1_{l=n}) \\ &= \alpha_n + \alpha_{nn} + \sum_{k=1}^{n} \alpha_{kn} i_k + \sum_{l=n}^{K} \alpha_{nl} i_l, \end{aligned} \tag{5.29}$$

which is a linear function of the state space coordinates. Consequently when doing policy improvement on the estimated relative values, the traffic class n acceptance condition of link net gain being positive, that is $G_n(\mathbf{i}) = h_n - p_n(\mathbf{i}) > 0$, yields the simple linear constraint

$$\sum_{k=1}^{n-1} \alpha_{kn} i_k + 2\alpha_{nn} i_n + \sum_{k=n+1}^{K} \alpha_{nk} i_k < h_n - \alpha_n - \alpha_{nn}. \tag{5.30}$$

When the cross-terms are omitted, this reduces to

$$i_n < \frac{h_n - \alpha_n}{2\alpha_{nn}} - \frac{1}{2}, \tag{5.31}$$

and we find that policy improvement on the quadratic relative values without cross-terms leads to a threshold policy.

The acceptance condition for traffic class n analogous to (5.30) for mth order polynomial basis functions without cross-terms is of the general form

$$\sum_{j=1}^{m-1} \alpha'_{j+1}\, i_n^j < h_n - \alpha'_1 \tag{5.32}$$

where α'_j, $j = 1, 2, \ldots, m$, are constants derived from the fitted parameters. By considering the shape of the recurrent set of the improved policy given by the acceptance condition (5.32), we find that when cross-terms are omitted, policy improvement results in a threshold policy regardless of the order of the polynomial basis functions.

5.5 Piecewise linear relative value functions

As an alternative, we consider the class of state-class-wise linear functions

$$v_{\mathbf{i}} = \alpha_{(0,c)} + \sum_{k \in \mathcal{K}} \alpha_{(k,c)} i_k \quad \text{for } \mathbf{i} \in \Omega(c),\ c = 0, 1, \ldots, C. \tag{5.33}$$

Observe that the class of value functions defined by (5.33) subsumes the piecewise constant representation produced by the Krishnan-Hübner method. The representation (5.33) has $K + 1$ parameters for each state class; however when c is sufficiently small, we have for some or even all $k \in \mathcal{K}$ that $i_k = 0$ for all $\mathbf{i} \in \Omega(c)$, and consequently some of the parameters are superfluous. To avoid getting under-determined parameters in the normal equations we drop the superfluous parameters, so that the total number of parameters is

$$J = \sum_{c=0}^{C} \left(1_{|\Omega(c)| > n_k(c)} + n_k(c)\right), \tag{5.34}$$

where $n_k(c) = \sum_{k \in \mathcal{K}} 1_{c \geq b_k}$ tells the number of linear coefficients needed, and the constant term is included only when there are more states in the class than available linear coefficients. We need the state class sizes $|\Omega(c)|$ later on in any case, so they can as well be computed before determining which variables are necessary. Note that J is always somewhat less than $(K + 1)(C + 1)$.

The expansion matrix X corresponding to (5.33) is defined by

$$x_{\mathbf{i},(k,c)} = 1_{\mathbf{i} \in \Omega(c)}(1_{k=0} + 1_{k>0} i_k), \tag{5.35}$$

and consequently the elements of matrix QX are

$$[QX]_{\mathbf{i},(k,c)} = \sum_{\mathbf{j}\in\Omega} q_{\mathbf{ij}} x_{\mathbf{j},(k,c)}$$

$$= \sum_{d=0}^{C} \sum_{\mathbf{j}\in\Omega(d)} q_{\mathbf{ij}} 1_{\mathbf{j}\in\Omega(c)} (1_{k=0} + 1_{k>0} j_k) = \sum_{\mathbf{j}\in\Omega(c)} q_{\mathbf{ij}} (1_{k=0} + 1_{k>0} j_k)$$

$$= \begin{cases} \lambda_l \big(1_{k=0} + 1_{k>0}(i_k + 1_{k=l})\big) & \text{if } c - b_l \geq 0 \text{ and } \mathbf{i} \in \Omega(c - b_l) \text{ for some } l \in R_{\mathbf{i}}, \\ i_l \mu_l \big(1_{k=0} + 1_{k>0}(i_k - 1_{k=l})\big) & \text{if } c + b_l \leq C \text{ and } \mathbf{i} \in \Omega(c + b_l) \text{ for some } l \in \mathcal{K}, \\ -\big(\sum_{l\in\mathcal{K}} i_l \mu_l + \sum_{l\in R_{\mathbf{i}}} \lambda_l\big)\big(1_{k=0} + 1_{k>0} i_k\big) & \text{if } \mathbf{i} \in \Omega(c), \\ 0 & \text{otherwise;} \end{cases}$$

$$\text{for all } \mathbf{i} \in \Omega,\ k \in \mathcal{K},\ c = 0, 1, \ldots, C. \quad (5.36)$$

In terms of (5.36) the elements of matrix $X^{\mathrm{T}} Q^{\mathrm{T}} Q X$ are

$$[X^{\mathrm{T}} Q^{\mathrm{T}} Q X]_{(k,c),(l,d)} = \sum_{\mathbf{i}\in\Omega} [QX]_{\mathbf{i},(k,c)} [QX]_{\mathbf{i},(l,d)}$$

$$= \sum_{e=0}^{C} \sum_{\mathbf{i}\in\Omega(e)} [QX]_{\mathbf{i},(k,c)} [QX]_{\mathbf{i},(l,d)} \quad \text{for all } k, l \in \mathcal{K}, \text{ and } c, d = 0, 1, \ldots, C. \quad (5.37)$$

We denote the inner sum in the final form of (5.37) by

$$\xi^e_{(k,c),(l,d)} = \sum_{\mathbf{i}\in\Omega(e)} [QX]_{\mathbf{i},(k,c)} [QX]_{\mathbf{i},(l,d)}, \quad k, l \in \mathcal{K},\ c, d, e = 0, 1, \ldots, C. \quad (5.38)$$

In order to continue the development, we assume that $[QX]_{\mathbf{i},(k,c)}$ is constant over state classes $\mathbf{i} \in \Omega(d)$ where $d = 0, 1, \ldots, C$; this means that the policy actions $R_{\mathbf{i}}$ as well as the call arrival rates λ_k, $k \in \mathcal{K}$, and the call completion rates μ_k, $k \in \mathcal{K}$, are constant over the state classes $\Omega(d)$. We shall now proceed to express (5.38) in terms of the state class sums

$$\sum_{\mathbf{i}\in\Omega(c)} \prod_{n=1}^{m} i_{k_n}, \quad c = 0, 1, \ldots, C,\ m \in \{0, 1, 2, 3, 4\},\ k_1, \ldots, k_m \in \mathcal{K}. \quad (5.39)$$

which are efficiently computable by the recursion formulas of Section 5.3. To construct the complete matrix $X^{\mathrm{T}} Q^{\mathrm{T}} Q X$, all of the sums (5.39) must be computed; this is not cheap but quite feasible for small K. Evaluating all the sums by the recursion formulas takes $O(K^4 C)$ arithmetic operations and requires storage for $(1 + K + K^2 + K^3 + K^4)C$ numbers.

Let us now consider $\xi^e_{(k,c),(l,d)}$ as defined in (5.38), with the index e fixed. For $\xi^e_{(k,c),(l,d)}$ to be nonzero, both $[QX]_{\mathbf{i},(k,c)}$ and $[QX]_{\mathbf{i},(l,d)}$ must be nonzero. By equation (5.36), $[QX]_{\mathbf{i},(k,c)}$ is nonzero for $\mathbf{i} \in \Omega(e)$ in exactly the following cases:

$[QX]_{\mathbf{i},(k,c)}$	conditions	max. elements	max. terms
λ_j	$k = 0,\ c = e + b_j$ for some $j \in R_{(e)}$	K	K
$\lambda_j i_k + 1_{k=j}\lambda_j$	$k > 0,\ c = e + b_j$ for some $j \in R_{(e)}$	K^2	$K^2 + K$
$\mu_j i_j$	$k = 0,\ c = e - b_j$ for some $j \in \mathcal{K}$	K	K
$\mu_j i_k i_j - 1_{k=j}\mu_j i_j$	$k > 0,\ c = e - b_j$ for some $j \in \mathcal{K}$	K^2	$K^2 + K$
$-\sum_{j\in\mathcal{K}} \mu_j i_j - \sum_{j\in R_{(e)}} \lambda_j$	$k = 0,\ c = e$	1	$K + 1$
$-\sum_{j\in\mathcal{K}} \mu_j i_k i_j - \big(\sum_{j\in R_{(e)}} \lambda_j\big) i_k$	$k > 0,\ c = e$	K	$K^2 + K$

Here $R_{(e)}$ denotes the common action $R_{\mathbf{i}}$ for $\mathbf{i} \in \Omega(e)$. The column "max. elements" indicates the maximum number of pairs (k, c) for which each condition can be fulfilled; row $\mathbf{i}$ of the matrix QX can contain no more than this number of nonzero elements of the form shown on the corresponding line of the table. The column "max. terms" indicates the maximum number of terms in components of the state vector $\mathbf{i}$ that the nonzero elements on row $\mathbf{i}$ of QX can together contain. For example, on the last row of the table the expression $-\sum_{j\in\mathcal{K}} \mu_j i_k i_j - \big(\sum_{j\in R_{(e)}} \lambda_j\big) i_k$ consists of $K+1$ terms in components of $\mathbf{i}$, and elements of this form can appear at most K times on a row of the matrix QX; thus elements of this form together have at most $K \cdot (K+1) = K^2 + K$ terms in components of $\mathbf{i}$.

Altogether, a row of QX can contain at most $2K^2 + 3K + 1$ nonzero elements. It follows that $\xi^e_{(k,c),(l,d)}$ can be nonzero for at most $(2K^2+3K+1)^2$ values of k, c, l, d when e is fixed, and the total number of nonzero elements in $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ can be no more than

$$(2K^2 + 3K + 1)^2(C+1) = (4K^4 + 12K^3 + 13K^2 + 6K + 1)(C+1). \tag{5.40}$$

By the assumption that $[QX]_{\mathbf{i},(k,c)}$ is constant over state classes $\mathbf{i} \in \Omega(d)$, we can write $\xi^e_{(k,c),(l,d)}$ in the form

$$\xi^e_{(k,c),(l,d)} = \sum_{i\in\Omega(e)} [QX]_{\mathbf{i},(k,c)}[QX]_{\mathbf{i},(l,d)} = \sum_{i\in\Omega(e)} \Pi^e_{(k,c),(l,d)}(\mathbf{i}), \tag{5.41}$$

where $\Pi^e_{(k,c),(l,d)}(\mathbf{i})$ is a polynomial in the components of $\mathbf{i}$, such that the coefficients only depend on k, c, l, d, and e. In the table above we see that for a fixed e elements of QX can contain at most $3K^2+6K+1$ terms in components of $\mathbf{i}$, and thus $\Pi^e_{(k,c),(l,d)}(\mathbf{i})$ can contain no more than $(3K^2+6K+1)^2$ terms. By performing the sum over $\Omega(e)$ in equation (5.41) separately for each term of $\Pi^e_{(k,c),(l,d)}$, we can express $\xi^e_{(k,c),(l,d)}(\mathbf{i})$ as a linear sum of at most $(3K^2+6K+1)^2$ terms of the form (5.39). Finally, by taking into account the $C+1$ possible values of e, the total number of terms of the form (5.39) required for constructing the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ is less than

$$(3K^2 + 6K + 1)^2(C+1) = (9K^4 + 36K^3 + 42K^2 + 12K + 1)(C+1). \tag{5.42}$$

Since the highest order terms in elements of QX are quadratic, $\Pi^e_{(k,c),(l,d)}$ contains terms of up to fourth order. Moreover, as elements of QX contain all terms of the forms i_k and $i_k i_l$, where $k, l \in \mathcal{K}$, as well as constant terms, it follows that each and every one of the sums (5.39) is necessary in constructing $X^{\mathrm{T}}Q^{\mathrm{T}}QX$.

The above treatment can be developed into an algorithm for constructing $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ in $O(K^4C)$ operations: the basic idea is to begin with the matrix storage zeroed, and then perform the outermost loop over $e = 0, 1, \ldots, C$, adding on each iteration at most $(3K^2 + 6K + 1)^2$ terms to appropriate locations all over the matrix storage. The sums (5.39) should be computed in advance for quick reference during the matrix construction. Both the advance computation of the sums, and the actual matrix construction loop require $O(K^4C)$ operations. In practice one can also take advantage of the fact that $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ is symmetric to nearly halve the amount of work.

In contrast to the matrix construction, the elements of the vector $X^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1} - \mathbf{r})$ are

significantly easier to compute:

$$
\begin{aligned}
[X^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1}-\mathbf{r})]_{(k,c)} &= \sum_{\mathbf{i}\in\Omega}[QX]_{\mathbf{i},(k,c)}[g\mathbf{1}-\mathbf{r}]_{\mathbf{i}} \\
&= \sum_{\substack{l\in\mathcal{K}\\ c-b_l\geq 0}} 1_{l\in R_{(c-b_l)}} \sum_{\mathbf{i}\in\Omega(c-b_l)} \lambda_l\big(1_{k=0}+1_{k>0}(i_k+1_{k=l})\big)(g-r_{\mathbf{i}}) \\
&\quad + \sum_{l\in\mathcal{K}} \sum_{\mathbf{i}\in\Omega(c+b_l)} i_l\mu_l\big(1_{k=0}+1_{k>0}(i_k-1_{k=l})\big)(g-r_{\mathbf{i}}) \\
&\quad - \sum_{\mathbf{i}\in\Omega(c)} \Big(\sum_{l\in\mathcal{K}} i_l\mu_l + \sum_{l\in R_{\mathbf{i}}}\lambda_l\Big)\big(1_{k=0}+1_{k>0}i_k\big)(g-r_{\mathbf{i}}) \\
&\qquad\qquad \text{for all } k\in\mathcal{K} \text{ and } c=0,1,\ldots,C. \quad (5.43)
\end{aligned}
$$

Here $R_{(c)}$ denotes the common action $R_{\mathbf{i}}$ for $\mathbf{i}\in\Omega(c)$. By the assumption that $R_{\mathbf{i}}$ and the λ_k's are constant over the state classes $\Omega(c)$, also the cost rates $r_{\mathbf{i}}$ are constant over each state class. Thus an element of $X^{\mathrm{T}}Q^{\mathrm{T}}(g\mathbf{1}-\mathbf{r})$ is a linear sum of at most $2+(K+1)+(K+1)$ terms of the form (5.39), and to construct the whole vector we need less than $(2K+4)(C+1)(K+1)=O(K^2C)$ terms of the form (5.39). The other matrix-vector products required for the normal equations with g as a free parameter can be computed analogously.

Using direct methods to solve the normal equations requires $O(J^3)=O(K^3C^3)$ operations. Since in practice $C\gg K$, the solution of the normal equations may in fact dominate the total time requirements of the method on large links, depending on the constant factors of the algorithms used.

5.5.1 Piecewise constant relative value functions

As a special case of the above we can fit state-class-wise constant relative value functions of the form

$$v_{\mathbf{i}}=\alpha_c \quad \text{for } \mathbf{i}\in\Omega(c),\ c=1,2,\ldots,C. \tag{5.44}$$

When the elements of the system of normal equations corresponding to the linear terms of the representation (5.33) are omitted, the complexity of the outlined construction procedure reduces to $O(K^2C)$. To see this, observe that when the columns (k,c) of QX where $k>0$ are left out, a row of QX contains only $3K+1$ terms in components of $\mathbf{i}$. Thus the number of terms in $\Pi^e_{(k,c),(l,d)}$ which can be nonzero with $k=l=0$, is only $(3K+1)^2$ and the total number of terms of the form (5.39) required for the matrix $X^{\mathrm{T}}Q^{\mathrm{T}}QX$ decreases to

$$(3K+1)^2(C+1)=(9K^2+6K+1)(C+1). \tag{5.45}$$

Also, all the needed terms of the form (5.39) are of at most second order, so that the necessary sums can be computed in $O(K^2C)$ time by the recursion formulas of Section 5.3. The complexity of constructing the matrix-vector products $X^{\mathrm{T}}Q^{\mathrm{T}}\mathbf{y}$ for various $\mathbf{y}$ is similarly reduced.

Since policy improvement on piecewise constant relative values leads to a trunk reservation policy, with the actions $R_{\mathbf{i}}$ constant on each state class $\Omega(c)$, it is possible to perform several policy iteration steps in hope of finding a near optimal trunk reservation policy. However as noted by Schweitzer and Seidmann, this kind of approximate policy iteration is not guaranteed to converge.

5.6 Numerical considerations

Usually when solving a linear least squares problem numerically, constructing the normal equations directly is not recommended because of ill-conditioning [51, page 142]; however in the present case speed considerations preclude the use of better conditioned methods. For example, when fitting piecewise linear functions on a link where the actual relative values are very close to constant on each state class, the determination of the (small) coefficients of the linear terms becomes unstable as rounding errors swamp the errors caused by the linear terms.

Extreme ill-conditioning of the system of normal equations can be circumvented by not solving for those parameters that cannot be stably determined. The problem of determining a subset of variables that are sufficiently weakly dependent is known as *subset selection*, and is discussed by Golub and Van Loan [16, Section 12.2] in the context of solving the linear least squares problem directly. Their approach is not directly applicable for solving the normal equations.

As an additional complication, when fitting specifically state-class-wise linear values, it must be ensured that no state class is left without at least one parameter, so that we can compute a relative value estimate for every system state. Also, if the average cost rate is being estimated as a free parameter, it should be included in the determined subset of parameters. By the same reasoning, when fitting piecewise constant value estimates we need to solve all the parameters in any case. For the polynomials spanning the complete state space there is no need to restrict the choice of parameter subset determined, except to ensure that the average cost rate is in the determined subset if appropriate.

Satisfactory results were achieved in numerical experiments by treating the normal equations as a linear least squares problem, and using the rank-deficient least squares solution procedure provided in the LAPACK library [2], which performs complete QR factorization with column pivoting and allows specifying the subset of parameters that should be determined regardless of ill-conditioning difficulties. Strictly speaking this approach is incorrect, since it merely disregards a number of columns of the system matrix, whereas a least squares problem in terms of a subset of the variables should ignore a subset of both rows and columns of the system. However, this is not disastrous since the rows which should be ignored are nearly linearly dependent on the other rows, and the extraneous elements in the residual error vector are linear combinations of the significant elements. Hence the Euclidean norm of the residual vector with extraneous elements can be assumed to be small whenever the norm of the residual corresponding to the chosen parameter subset is small.

Chapter 6

Computational experiments

The methods developed in this thesis along with the original Krishnan-Hübner method were tested numerically on a few multiservice link models with state spaces small enough that the results could be verified accurately, and compared by a few different criteria.

6.1 The methods compared

Several variants of both least squares fitting and the Krishnan-Hübner method were compared. In the following we indicate in parentheses the somewhat mnemonic abbreviations by which the methods are identified in the results tables.

In addition to the original Krishnan-Hübner method (K&H), the following modifications discussed in Section 4.3.1 were evaluated: Instead of the state weights derived from the product form balance probabilities of the complete sharing policy, the states in each state class were balanced equally as suggested by Schweitzer and Kindle (EWA). Viewing the Krishnan-Hübner method as the first policy iteration in a Markov model, the policy iteration algorithm was continued to convergence ($n\times$K&H). As another alternative the number of policy iterations was limited to two to get an impression of the changes in successive iterations ($2\times$K&H). Additionally, Schweitzer-Kindle disaggregation was applied to the aggregated values produced by the original Krishnan-Hübner method as well as all the variants (indicated by appending +DA).

Least squares fitting was performed for all the basis functions discussed in detail in Chapter 5: quadratic polynomials without cross-terms (LS-QS) and with cross-terms (LS-QX), piecewise linear polynomials (LS-PL), and piecewise constant values (LS-PC). All these basis functions were fit to the relative values of the complete sharing policy, as well as to the relative values of the trunk reservation policy produced by the original Krishnan-Hübner method (indicated by prepending K&H+). In the former case, the average cost rate was fixed by computing it by Kaufman-Roberts recursion (indicated by the suffix /g), and in the latter case the average cost rate was included in the fitted parameters, since there is no known method for efficiently computing the exact average cost rate of a general trunk reservation policy.

The approximation methods for which we present results are summarized in Table 6.1, along with their abbreviations in the results tables. For completeness we also provided results for least squares fitting of cubic polynomials without cross-terms (LS-PC); these were implemented simply and inefficiently by constructing the system of normal equations as an explicit product of very large matrices.

All computations were performed in IEEE double precision floating point numbers.

abbreviation	approximation method
K&H	the original Krishnan-Hübner method
K&H+DA	the Krishnan-Hübner method with disaggregation
$2\times$K&H	up to two policy iterations in the Krishnan-Hübner model
$2\times$K&H+DA	up to two policy iterations in the Krishnan-Hübner model, with disaggregation
$n\times$K&H	policy iteration until convergence in the Krishnan-Hübner model
$n\times$K&H	policy iteration until convergence in the Krishnan-Hübner model, with disaggregation
EWA	Equal-weight aggregation
EWA+DA	Equal-weight aggregation with disaggregation
LS-PC/g	Least-squares fitting of piecewise constant values, with the average cost rate fixed
LS-PL/g	Least-squares fitting of piecewise linear values, with the average cost rate fixed
LS-QS/g	Least-squares fitting of quadratic polynomials without cross-terms, with the average cost rate fixed
LS-QX/g	Least-squares fitting of quadratic polynomials with cross-terms, with the average cost rate fixed
LS-CS/g	Least-squares fitting of cubic polynomials without cross-terms, with the average cost rate fixed
K&H+LS-PC	Least-squares fitting of piecewise constant values to the relative values of the Krishnan-Hübner policy
K&H+LS-PL	Least-squares fitting of piecewise linear values to the relative values of the Krishnan-Hübner policy
K&H+LS-QS	Least-squares fitting of quadratic polynomials without cross-terms to the relative values of the Krishnan-Hübner policy
K&H+LS-QX	Least-squares fitting of quadratic polynomials with cross-terms to the relative values of the Krishnan-Hübner policy

Table 6.1: The approximation methods presented, and their abbreviations.

6.2 Comparison criteria

The approximation methods were compared by three criteria corresponding to different purposes of using the methods. First, supposing that the approximated relative values were used for real-time policy improvement on a single link as per Section 3.6, the exact average cost of the resulting single link connection admission policy was evaluated. This does not however directly measure the suitability of the methods to network level policy improvement, where the accuracy of the link shadow prices is of most importance.

In the reviewed literature the suitability of approximate relative value estimation methods to network level policy improvement is evaluated by comparing the average cost rates of the resulting policies in a network simulation. However, this is computationally expensive and requires fixing the network level policy iteration method. As a more direct and simpler alternative, we compare the link shadow prices computed from the estimated and true relative values by a novel error measure, in which we take into account the intended use of the link shadow prices in network level policy improvement. This error measure, designated

"mean scaled error", is defined as

$$\sum_{\mathbf{i}\in\Omega}\sum_{\substack{k\in\mathcal{K}\\ \mathbf{i}+\mathbf{e}_k\in\Omega}}\Big|L\big(\frac{\tilde{v}_{\mathbf{i}+\mathbf{e}_k}-\tilde{v}_{\mathbf{i}}}{h_k}\big)-L\big(\frac{v_{\mathbf{i}+\mathbf{e}_k}-v_{\mathbf{i}}}{h_k}\big)\Big|\Big/\sum_{\mathbf{i}\in\Omega}\sum_{\substack{k\in\mathcal{K}\\ \mathbf{i}+\mathbf{e}_k\in\Omega}}1, \tag{6.1}$$

where $\tilde{v}_{\mathbf{i}}$ denotes the estimated and $v_{\mathbf{i}}$ the true relative value of state $\mathbf{i}\in\Omega$, and the function $L(x)=\min\{1,\max\{0,x\}\}$ forces its argument onto the interval $[0,1]$. The rationale for this specific form is as follows: The fraction $f=(v_{\mathbf{i}+\mathbf{e}_k}-v_{\mathbf{i}})/h_k$ is the link shadow price scaled so that 1 is the point at which it becomes profitable to deny calls of traffic class k in state $\mathbf{i}$. Consider now network level policy improvement where the cost parameters h_k are equal on each link—when the link shadow prices of a route choice are added up, it is sufficient for the fraction f to be larger than 1 on any one of the links, for calls of traffic class k to be denied from the route. Thus on the interval $[0,1]$ the accuracy of the link shadow prices is crucial, whereas all link shadow prices larger than 1 on this scale are functionally equivalent; this is why we use $L(\cdot)$ to limit the range to $[0,1]$. The bottom limit 0 is there merely to ensure that inaccurate relative value estimates never lead to negative link shadow prices. The rest of expression (6.1) simply averages the absolute errors in these limited fractions over the link state space and the traffic classes. Finally, if the mean scaled link shadow price error is less than ϵ/n, the sum of link shadow prices over a route of n identical links can be expected to be less than ϵh_k off from the correct path shadow price.

As the third comparison criterion, the accuracy of the average cost rate estimates provided by the methods was evaluated. The intention was to see whether the methods could be used to compute link performance measures for policies that do not have product form balance probabilities, and consequently also lack efficient performance measurement algorithms.

The exact relative values and average cost rates needed for evaluating the discussed comparison criteria were computed by solving the complete Howard equations by the stabilized biconjugate gradient (BICGSTAB) method with symmetric successive over-relaxation (SSOR) preconditioning [41, Chapter 7], which was found to provide results quicker and more accurately than the Bertsekas-Castañon iterative aggregation-disaggregation method [3]. The implementation of the BICGSTAB procedure in the LASPACK library by Tomáš Skalický [48] was used. It should be noted that the BICGSTAB procedure is no panacea to the solution of moderately large Howard equations—on some links it would fail to converge, or run into a floating point overflow.

6.3 Sample link models

The relevant properties of the link models for which computational results are presented, can be seen in Table 6.2. The column "offered" measures the traffic load via the proportion

$$\Big(\sum_{k\in\mathcal{K}} b_k\frac{\lambda_k}{\mu_k}\Big)\Big/C, \tag{6.2}$$

where the offered traffics of traffic classes are multiplied by the number of trunks required by connections of each traffic class and added up, thus producing the expected number of occupied trunks on the link were its capacity not limited, and this number is then normalized by dividing by the link capacity. Hence if the ratio (6.2) is below unity, one can expect most of the traffic to be carried; the larger the ratio is, the more traffic must necessarily be denied, and this is where the role of effective optimization of connection admission policy is most important.

	C	K	b_k	λ_k	μ_k^{-1}	h_k	$\lvert\Omega\rvert$	offered
L3U	100	3	1,2,3	20,10,4	b_k	1	30787	0.96
L3W	100	3	1,2,3	20,10,4	b_k	$b_k\mu_k^{-1}$	30787	0.96
M3U	100	3	1,2,3	20,20,5	b_k	1	30787	1.45
M3W	100	3	1,2,3	20,20,5	b_k	1,2,2	30787	1.45
H3U	100	3	1,2,7	20,20,5	1,2,3	1	13962	2.05
H3W	100	3	1,2,7	20,20,5	1,2,3	$b_k\mu_k^{-1}$	13962	2.05
M4U	100	4	1,2,6,17	99,6,2,1	$\frac{1}{3}$,1,3,2	1	33462	1.15
M4W	100	4	1,2,6,17	99,6,2,1	$\frac{1}{3}$,1,3,2	$b_k\mu_k^{-1}$	33462	1.15
L5U	70	5	1,2,5,8,15	12,3,$\frac{1}{2}$,$\frac{1}{5}$,$\frac{1}{12}$	b_k	1	31499	0.97
L5W	70	5	1,2,5,8,15	12,3,$\frac{1}{2}$,$\frac{1}{5}$,$\frac{1}{12}$	b_k	$b_k\mu_k^{-1}$	31499	0.97
M5U	70	5	1,3,5,9,12	12,3,$\frac{1}{2}$,$\frac{1}{5}$,$\frac{1}{12}$	b_k	1	22808	1.14
M5W	70	5	1,3,5,9,12	12,3,$\frac{1}{2}$,$\frac{1}{5}$,$\frac{1}{12}$	b_k	$b_k\mu_k^{-1}$	22808	1.14
H5U	65	5	1,3,6,7,8	33,2,1,$\frac{1}{2}$,$\frac{1}{5}$	b_k	1	23347	1.91
H5W	65	5	1,3,6,7,8	33,2,1,$\frac{1}{2}$,$\frac{1}{5}$	b_k	$b_k\mu_k^{-1}$	23347	1.91
M6U	60	6	1,2,5,7,11,14	9,4,$\frac{1}{2}$,$\frac{1}{3}$,$\frac{1}{10}$,$\frac{1}{14}$	b_k	1	32423	1.33
M6W	60	6	1,2,5,7,11,14	9,4,$\frac{1}{2}$,$\frac{1}{3}$,$\frac{1}{10}$,$\frac{1}{14}$	b_k	$b_k\mu_k^{-1}$	32423	1.33

Table 6.2: Properties of the tested links. From left to right: link capacity, number of traffic classes, then the trunk requirements, arrival rates, call holding times, and weights of the traffic classes; the number of states in the link model, and the proportion of link capacity covered by the offered traffic (see text).

Two kinds of cost parameters h_k are used, corresponding to optimization of either the number of carried calls, or the number of trunks utilized. As the first alternative, all the cost parameters h_k are set equally to unity, so that all calls are considered of identical worth with the effect that the optimization maximizes the number of carried calls and minimizes call blocking. The second alternative tested is to set $h_k = b_k\mu_k^{-1}$ for all k, so that the worth of a call equals the product of the number of trunks it occupies and the expected call holding time, with the effect that the optimization maximizes the average number of occupied trunks. We refer to the former as the unweighted and the latter the weighted case. All link models are tested with both kinds of cost parameters. The cost parameters of link M3W do not follow the rule; this is because the links M3U and M3W are precisely the single link examples used by Krishnan and Hübner in their paper [29].

Ignoring the differences in cost parameters h_k, there are only eight link models tested. Three of the links are variations of the single-link examples of Krishnan and Hübner, with different amounts of offered traffic, and additionally in links H3U and H3W the trunk requirement b_k of the third traffic class has been increased. Then there are three somewhat different link models with five traffic classes and varying amounts of offered traffic. In addition there is one link with four traffic classes, and one with six traffic classes.

The call holding times μ_k^{-1} of the traffic classes are generally equivalent to the trunk requirements b_k, that is wide-band connections are presumed to last longer than narrow-band ones; an exception is made for the links H3U and H3W to keep them closer to the examples of Krishnan and Hübner, and for the links M4U and M4W simply to get another alternative case.

The shorthand link identifiers consist of a letter roughly indicating the magnitude of the offered ratio, a digit indicating the number of traffic classes, and a letter indicating whether the cost parameters are all unity or not, that is whether the link is “unweighted” or “weighted”.

6.4 Results and conclusions

Obviously the selection of links tested is not wide enough to make far-reaching conclusions, but we can note some issues in the presented results.

6.4.1 Average cost rates

To compare the applicability of the approximation methods to the optimization of single link connection admission policy, the average cost rates of the policies obtained by policy improvement on approximated relative values are shown in Table 6.3.

On most weighted links, the complete sharing policy is a near-optimal single link connection admission policy; this is understandable considering that the weights were chosen so as to maximize the link utilization. While this makes it unnecessary to use any approximation methods to improve the single link connection admission policy, in network level policy improvement it is still necessary to have good estimates of link shadow prices available.

There is a remarkable contrast between the weighted and unweighted cases in the performance of the methods based on the Krishnan-Hübner method. On one hand, except for the link M3W which does not follow the usual weighting rule, in every weighted case all methods based on the Krishnan-Hübner method produce a policy worse than the initial complete sharing policy. On the other hand, in all the unweighted cases these methods produce policies markedly better than the complete sharing policy.

It should also be noted that in many cases continuing the policy iteration in the aggregated model after the second iteration worsens the average cost rate, and only on link M3U is a slight improvement made. In contrast, only on links M3U and M3W (that is, the examples of Krishnan and Hübner), and L3U does the second policy iteration produce a worse average cost rate than the first iteration.

The good results in the unweighted cases might be interpreted as indicating that in these cases the aggregated Krishnan-Hübner model would be a good match to the actual random process formed by the number of occupied trunks, particularly since the policy iterations in the Krishnan-Hübner model consistently improve the average cost rate, and the first iteration often performs better than the first iteration in the complete multiservice link model. However, the situation is complicated by the fact that as random processes the weighted links are identical to the unweighted links, and only the optimization criteria changes—yet policy iteration on the aggregated models of the unweighted links leads to wildly inferior results.

The disaggregation step quite consistently makes a small improvement on the basic Krishnan-Hübner or equal-weight aggregation methods, and the few exceptions are in the results of continuing the policy iteration in the aggregated model, where the aggregation-disaggregation connection does not apply. Unfortunately in the unweighted cases the improvements made by disaggregation are negligible to the point of not affecting the digits visible in the table.

As regards the choice of aggregation weights, the equal-weight aggregation (EWA) performs approximately as well as the Krishnan-Hübner method, while in individual cases there are relatively large differences in favour of each method; however in some of the unweighted cases equal-weight aggregation falls quite far short of the Krishnan-Hübner method.

Let us now move on to discuss the least squares fitting methods on the relative values of the complete sharing policy. By virtue of approximating the relative values of the complete sharing policy more rigorously than the Krishnan-Hübner method, in terms of the average

	L3U	L3W	M3U	M3W	H3U	H3W	M4U	M4W
CS policy	2.16	8.68	12.15	20.82	13.83	109.33	5.50	30.73
1st policy it.	1.85	8.51	8.04	15.67	8.05	109.33	3.00	30.57
K&H	1.58	12.09	8.05	15.80	8.04	118.58	1.91	47.03
K&H+DA	1.58	12.02	8.00	15.80	8.04	117.39	1.89	45.83
2×K&H	1.61	9.19	8.15	16.11	8.00	114.39	1.64	33.40
2×K&H+DA	1.61	9.19	8.15	16.11	8.00	114.45	1.64	33.46
n×K&H	1.61	9.19	8.12	16.11	8.00	115.66	1.65	36.30
n×K&H+DA	1.61	9.19	8.12	16.11	8.00	114.82	1.65	35.27
EWA	1.70	9.47	8.17	16.35	8.04	116.78	1.79	45.82
EWA+DA	1.70	9.39	8.17	16.35	8.04	115.75	1.76	45.53
LS-PC/g	1.97	8.51	8.05	17.17	8.40	118.58	2.90	34.18
LS-PL/g	1.85	8.74	8.04	15.67	8.04	109.37	3.00	30.72
LS-QS/g	2.35	8.68	11.95	20.75	13.60	110.24	3.00	30.73
LS-QX/g	3.03	8.68	8.34	15.67	12.28	109.34	2.79	30.73
LS-CS/g	2.83	8.68	12.22	20.48	13.60	110.10	3.00	30.73
K&H+LS-PC	1.97	8.51	10.43	19.77	9.46	110.96	3.61	30.54
K&H+LS-PL	1.58	8.51	8.00	15.64	8.00	109.39	3.00	30.58
K&H+LS-QS	2.08	8.68	13.07	22.56	13.60	109.33	1.94	30.73
K&H+LS-QX	1.98	8.68	10.18	19.60	8.40	109.33	2.01	30.73
	L5U	L5W	M5U	M5W	H5U	H5W	M6U	M6W
CS policy	0.71	15.05	1.35	21.65	7.18	64.06	1.50	30.13
1st policy it.	0.23	15.05	0.49	21.61	2.15	64.06	0.57	30.13
K&H	0.25	26.38	0.54	30.55	2.63	70.83	0.62	38.74
K&H+DA	0.25	24.78	0.52	29.28	2.63	70.38	0.62	38.35
2×K&H	0.23	17.46	0.52	24.53	2.36	67.05	0.56	33.09
2×K&H+DA	0.22	17.23	0.52	25.08	2.36	67.46	0.56	32.99
n×K&H	0.23	18.47	0.53	24.53	2.41	67.86	0.56	34.17
n×K&H+DA	0.22	17.75	0.52	25.08	2.41	67.74	0.56	33.90
EWA	0.32	21.72	0.52	28.59	3.70	69.25	0.55	38.13
EWA+DA	0.32	20.54	0.52	28.51	3.70	69.47	0.55	37.05
LS-PC/g	0.61	15.56	1.08	23.16	2.15	68.39	1.09	32.81
LS-PL/g	0.69	15.09	0.78	22.01	2.11	64.06	0.90	30.20
LS-QS/g	0.78	15.05	1.29	21.65	3.43	64.06	1.00	30.13
LS-QX/g	0.25	15.05	0.62	21.69	4.37	64.06	0.59	30.13
LS-CS/g	0.78	15.05	2.36	21.65	4.18	64.06	1.00	30.13
K&H+LS-PC	0.61	15.11	1.10	22.05	2.12	65.58	1.21	30.42
K&H+LS-PL	0.52	15.14	0.76	22.10	2.94	65.41	0.88	30.44
K&H+LS-QS	0.52	15.05	0.54	21.65	2.19	64.06	0.74	30.13
K&H+LS-QX	0.24	15.05	0.52	21.65	2.67	64.06	0.59	30.13

Table 6.3: The average cost rates of single link connection admission policies which were created by policy improvement on the relative value estimates produced by the approximation methods. Also included are the complete sharing (CS) policy and the policy created on the first iteration of the policy iteration algorithm using accurate relative values.

cost rate of the produced policy these least squares fitting methods only perform as well as the first policy iteration in the complete model and often fall more or less short. Of the more complete sets of basis functions, that is the piecewise linear functions (LS-PL/g) and quadratic polynomials with cross-terms (LS-QX/g), both do match the first policy iteration in the complete model quite often, but in some cases one or the other fails badly. The fact that the more complete sets of basis functions sometimes fare worse than subsets of their parameters (witness links L3U and L3W) suggests that the subset selection (selection of parameters to be determined in face of ill-conditioning) needs improvement.

It is interesting that in the weighted cases the plain quadratic polynomials without cross-terms (LS-QS/g) perform quite well. On the other hand with few exceptions mainly on the weighted links, the fitting of piecewise constant functions (LS-PC/g) performs worse than the variants of the Krishnan-Hübner method. This suggests that the unweighted minimization of residual in the Howard equations may not be the optimal fitting criterion at least when the fits are not very close.

The least squares fitting methods on the relative values of the Krishnan-Hübner policy, especially with piecewise linear functions (K&H+LS-PL) and quadratic polynomial with cross-terms (K&H+LS-QX) as bases, seem competitive with the variants of the Krishnan-Hübner method in terms of the average cost rate of the produced policy; however the fitting methods suffer again from the occasional bad fit.

6.4.2 Link shadow prices

To compare the applicability of the approximation methods to network level policy optimization, Table 6.4 shows the mean scaled errors in the link shadow price estimates computed from the relative value estimates. The estimated link shadow prices are always compared with the exact link shadow prices under the policy whose relative values the method is estimating; for example completed policy iteration in the aggregated Krishnan-Hübner model ($n\times$K&H+DA) estimates the relative values of the second-to-last policy in the policy iteration procedure.[1]

The least squares fitting of piecewise linear functions (LS-PL/g and K&H+LS-PL) consistently produces good link shadow price estimates, seldom producing mean scaled errors above 0.1, but by comparing LS-PL/g and K&H+LS-PL it is apparent that the relative values of the Krishnan-Hübner policy are harder to fit by piecewise linear functions. The quadratic polynomials with cross-terms (LS-QX/g and K&H+LS-QX) produce occasionally even better fits, but also several markedly worse fits than the piecewise linear basis functions.

The disaggregation step of Schweitzer and Kindle generally provides a small improvement to the link shadow price accuracy, but in some cases it also causes an even smaller deterioration.

In terms of the mean scaled error, equal-weight aggregation (EWA) performs slightly better than the Krishnan-Hübner method (K&H). This is explained by the fact that the error measure is also equally weighted over the state space, without taking into account the balance probabilities; this is intentional since in network level real-time policy improvement the balance probabilities of the link level policy do not necessarily reflect the actual observed link state probabilities which depend on network level routing decisions.

Again, least squares fitting of quadratic polynomials without cross-terms performs relatively well on weighted links. As another repeating theme, least squares fitting of piecewise constant functions (LS-PC/g and K&H+LS-PC) performs worse than variants of the

[1] Incidentally, because of the way the convergence test of the policy iteration algorithm works, the second-to-last policy is actually the same as the final policy produced by the iteration.

	L3U	L3W	M3U	M3W	H3U	H3W	M4U	M4W
K&H	0.103	0.238	0.165	0.257	0.185	0.113	0.232	0.328
K&H+DA	0.101	0.235	0.163	0.254	0.183	0.113	0.227	0.324
2×K&H	0.082	0.228	0.143	0.251	0.147	0.105	0.169	0.323
2×K&H+DA	0.080	0.231	0.135	0.249	0.144	0.091	0.168	0.314
n×K&H	0.082	0.228	0.140	0.251	0.141	0.119	0.171	0.371
n×K&H+DA	0.080	0.231	0.137	0.249	0.138	0.119	0.170	0.355
EWA	0.100	0.244	0.163	0.253	0.168	0.113	0.232	0.319
EWA+DA	0.099	0.242	0.162	0.251	0.168	0.114	0.227	0.310
LS-PC/g	0.121	0.329	0.161	0.250	0.180	0.113	0.251	0.405
LS-PL/g	0.029	0.054	0.020	0.021	0.042	0.008	0.182	0.032
LS-QS/g	0.112	0.250	0.122	0.186	0.204	0.080	0.267	0.079
LS-QX/g	0.099	0.169	0.078	0.087	0.234	0.019	0.250	0.025
LS-CS/g	0.119	0.253	0.120	0.181	0.206	0.080	0.274	0.079
K&H+LS-PC	0.078	0.330	0.157	0.257	0.164	0.105	0.210	0.433
K&H+LS-PL	0.023	0.024	0.026	0.034	0.032	0.009	0.234	0.081
K&H+LS-QS	0.128	0.169	0.191	0.289	0.147	0.052	0.283	0.122
K&H+LS-QX	0.105	0.082	0.121	0.216	0.111	0.041	0.222	0.085
	L5U	L5W	M5U	M5W	H5U	H5W	M6U	M6W
K&H	0.199	0.141	0.234	0.106	0.445	0.060	0.258	0.086
K&H+DA	0.191	0.141	0.227	0.107	0.433	0.062	0.249	0.088
2×K&H	0.138	0.152	0.170	0.141	0.329	0.165	0.191	0.127
2×K&H+DA	0.135	0.121	0.171	0.122	0.322	0.157	0.183	0.111
n×K&H	0.142	0.106	0.175	0.141	0.339	0.079	0.197	0.071
n×K&H+DA	0.139	0.096	0.173	0.122	0.331	0.075	0.188	0.069
EWA	0.166	0.141	0.233	0.106	0.413	0.060	0.249	0.086
EWA+DA	0.159	0.142	0.226	0.108	0.405	0.061	0.241	0.089
LS-PC/g	0.172	0.207	0.255	0.128	0.514	0.093	0.254	0.086
LS-PL/g	0.109	0.040	0.091	0.029	0.062	0.003	0.096	0.021
LS-QS/g	0.214	0.104	0.187	0.076	0.461	0.041	0.234	0.058
LS-QX/g	0.117	0.032	0.140	0.021	0.445	0.011	0.152	0.019
LS-CS/g	0.215	0.104	0.193	0.077	0.482	0.041	0.238	0.059
K&H+LS-PC	0.156	0.523	0.216	0.345	0.417	0.336	0.240	0.463
K&H+LS-PL	0.158	0.041	0.144	0.042	0.126	0.030	0.178	0.055
K&H+LS-QS	0.232	0.094	0.202	0.061	0.271	0.060	0.261	0.104
K&H+LS-QX	0.159	0.068	0.178	0.046	0.131	0.030	0.189	0.070

Table 6.4: Mean scaled errors in link shadow prices (see text).

Krishnan-Hübner method, providing more evidence for the suspicion that the unweighted least squares fitting is unfitting for inaccurate fits.

Unlike in the comparison of the average cost rates of produced policies, here the least squares fitting methods that use a superset of the basis functions of another method performed worse than the subsumed method only in a few isolated cases.

6.4.3 Average cost rate estimates

Table 6.5 shows the average cost rate estimates provided by the approximation methods. In the aggregation-based methods, that is Krishnan-Hübner and equal-weight aggregation, we consider the average cost rate computed in the aggregated model as the average cost rate estimate. As in the case of link shadow price estimation, the average cost rate estimates are compared with whatever policy it is that the method is approximating.

To provide more comparison material, in Table 6.5 the least squares fitting methods on the relative values of the complete sharing policy have been replaced by the variants which treat the average cost rate as a free parameter. Of course, the average cost rate of the complete sharing policy is a solved problem: it can be computed exactly by the Kaufman-Roberts recursion, and the exact value is also provided by the Krishnan-Hübner method. Thus the most interesting average cost rate estimates are those computed for the Krishnan-Hübner policy, which usually is not of a product form.

As is apparent from Table 6.5, the average cost rate estimates are not very accurate, and while all methods except the further iterations in the aggregated Krishnan-Hübner model ($2\times$K&H and $n\times$K&H) occasionally provide an accurate estimate, most of the time they are nowhere near the correct value. We can safely conclude that none of these approximation methods are as such any good for link performance evaluation, unless it can be somehow a priori determined that a method will provide a very good fit.

6.4.4 Average cost rate bounds

Numerical tests were also performed for the error bounds of Section 4.5 on the average cost rates of policies constructed from piecewise constant relative values by policy improvement. Unfortunately it was found that in all cases tested the upper bounds were 2 to 20 times the magnitude of the average cost rate of the complete sharing policy, and the lower bounds were negative, making the bounds useless in practice. It was verified that computing the bounds of Proposition 2.8 exactly did not provide significant improvement on the fast but more loose bounds of Section 4.5.

	L3U	L3W	M3U	M3W	H3U	H3W
K&H	0	0	0	0	0	0
2×K&H	+20.73%	−39.37%	+28.76%	+23.25%	+30.96%	−29.08%
n×K&H	+18.95%	−20.22%	+26.23%	+20.89%	+30.17%	−30.82%
EWA	−64.00%	−62.07%	−28.02%	−26.74%	−22.37%	−13.74%
LS-PC	−51.22%	−49.57%	−45.39%	−44.43%	−32.06%	−23.62%
LS-PL	+16.24%	+13.43%	+0.34%	+0.31%	+1.18%	−0.03%
LS-QS	+53.54%	+46.51%	+11.76%	+11.85%	+10.50%	−0.93%
LS-QX	+36.57%	+37.67%	+7.86%	+9.70%	+5.99%	+1.41%
K&H+LS-PC	−20.20%	−46.49%	−8.40%	−15.41%	−5.55%	−30.78%
K&H+LS-PL	+21.98%	−0.48%	+5.21%	+4.65%	+4.87%	−0.70%
K&H+LS-QS	+99.09%	+45.31%	+69.76%	+53.31%	+66.44%	+3.53%
K&H+LS-QX	+38.24%	+20.94%	+42.72%	+37.49%	+17.48%	−1.43%

	M4U	M4W	L5U	L5W	M5U	M5W
K&H	0	0	0	0	0	0
2×K&H	+57.78%	−45.49%	+87.07%	−58.05%	+80.00%	−58.35%
n×K&H	+75.08%	−70.42%	+100.02%	−72.52%	+81.05%	−48.12%
EWA	+54.76%	+36.71%	−60.23%	−34.65%	−14.42%	−2.94%
LS-PC	+17.32%	+9.49%	+13.18%	+15.38%	−5.23%	−0.67%
LS-PL	+78.97%	+6.82%	+8.37%	+2.84%	+21.49%	+4.04%
LS-QS	+46.88%	+5.73%	−14.72%	−0.51%	+46.01%	+8.07%
LS-QX	+10.08%	+2.05%	−27.46%	−1.63%	+15.61%	+0.83%
K&H+LS-PC	+276.79%	+15.89%	+378.81%	+42.29%	+215.56%	+7.36%
K&H+LS-PL	−168.60%	+3.77%	−212.04%	+2.34%	−53.73%	−2.26%
K&H+LS-QS	−51.96%	+5.30%	−273.72%	−4.37%	+63.96%	+2.18%
K&H+LS-QX	−353.52%	+1.08%	−331.14%	−4.79%	−177.01%	−0.11%

	H5U	H5W	M6U	M6W
K&H	0	0	0	0
2×K&H	+81.19%	−53.07%	+67.86%	−52.86%
n×K&H	+96.86%	−68.11%	+80.70%	−69.58%
EWA	+111.96%	+39.00%	−19.32%	−4.91%
LS-PC	+61.69%	+17.07%	−7.23%	−0.76%
LS-PL	+21.88%	+0.39%	+17.64%	+1.42%
LS-QS	+47.08%	+0.91%	+33.82%	+2.39%
LS-QX	+47.69%	+0.12%	+10.66%	−0.24%
K&H+LS-PC	+265.43%	−21.09%	+186.19%	+23.48%
K&H+LS-PL	−281.98%	−0.48%	−120.17%	+0.71%
K&H+LS-QS	+84.88%	+3.47%	−73.38%	−1.81%
K&H+LS-QX	−164.85%	+1.52%	−173.51%	−1.32%

Table 6.5: The relative error in the average cost rate estimates provided by the approximation methods. Note that not all methods estimate the average cost rate at all, and this table contains some different fitting methods than the other results tables.

Chapter 7

Conclusion and future work

In the thesis an interpretation of the Krishnan-Hübner method via a previously known aggregation-disaggregation method has been presented. Unfortunately the interpretation does not lead to any useful convergence or error bound results, and the disaggregation formula it suggests as an improvement over the Krishnan-Hübner method does not have a very significant effect in the few computational tests presented. Thus the interpretation has value mostly as a mathematical curiosity, unless future work on the convergence of the aggregation-disaggregation variant of Schweitzer and Kindle can prove useful conditions for the method to be globally convergent. However at present this seems unlikely considering that no related convergence results apply to linear systems with a stochastic system matrix.

In the Krishnan-Hübner method, the partitioning of the link state space by the number of occupied trunks is natural and leads to perhaps the most easily handled aggregation formulas, but it would be interesting to study the advantages of choosing different state partitions, for example of the type $\mathbf{i}^{\mathrm{T}}\mathbf{d} = \textit{constant}$ for various vectors $\mathbf{d}$. Ideally one would be able to choose the state partition that leads to the most accurate approximation on a given set of link model parameters; even heuristic rules would be of use. Different state partitions should also be considered for the piecewise constant and piecewise linear basis functions used in least squares fitting.

The numerical results cover only a few special cases, and it has been noticed in other numerical tests, not included in the thesis in order to keep the amount of numbers in the results tables manageable, that the relative performance of the approximation methods can be quite different if link parameters are chosen at random without regard to the plausibility of their appearing in practice.[1]

A novel method was developed for least squares fitting of various basis functions to the relative values of trunk reservation policies. The method, in particular with the piecewise linear relative values, shows promise as a relatively efficient and tolerably accurate way of estimating link shadow prices. However more research is necessary before the method can be considered ready for practical applications. For example, even in the few links tested there were occasional inaccurate fits, which should be at least detected if not avoided. More extensive testing on both large links and in network policy iteration should be carried out.

Currently only an ad-hoc solution in the form of a LAPACK library routine has been proposed for the subset selection problem, that is determining which parameters to ignore in the normal equations, in order to work around ill-conditioning of the normal equations. The subset selection problem should be properly analysed in the context of the present fitting problem, to be able to justify the cutoff point at which the condition number of the system

[1] Also, the author has an unscientific tendency to refer to data not shown.

is large enough to begin dropping parameters, and to make a reasoned choice as to which parameters to drop.

The computational tests were run on link models with the number of system states on the order of tens of thousands, and the presented recursion formulas for computing integer sums over portions of link state space did not produce integers larger than can be presented accurately in IEEE double precision floating point numbers. However, to apply the recursion formulas on links with truly large state spaces, one either has to resort to less efficient 128-bit or even larger integer types, or properly analyse the errors caused by floating point arithmetic in the recursion formulas. Another area of study is whether it is possible to extend the recursion formulas to network state spaces without sacrificing their computational efficiency.

It may be feasible to extend the least squares fitting methods for fitting the relative values of threshold policies, by using the convolution algorithm in constructing the system of normal equations. This would extend the range of state-dependent transition rates that could be handled. Another advantage would be the possibility of performing policy iteration with the relative values represented as quadratic polynomials without cross-terms.

As discussed by Schweitzer and Seidmann and reviewed in Section 5.1, in least squares fitting of relative values it is possible to assign different weights to the residual errors in different states; the advantages of weight choices should be investigated. Also, the choice of weights in aggregation-disaggregation bears further study.

As also noted in Section 5.1, the same methods as used in constructing the normal equations for least squares fitting, can be used for fitting relative values to basis functions via the Galerkin approach; this alternative should be studied and compared to least squares fitting.

In general, it would be useful to have tight error bounds for the link shadow price estimates computed by the discussed approximation methods. Especially if these bounds could be computed a priori on the basis of link model parameters alone, it would be possible to choose alternative network control methods whenever a particular approximative method would be too inaccurate. Since in practice call arrival rates and holding times are estimated from past calls, one should also be able to bound the errors in the link shadow price estimates in terms of error bounds on the call arrival rates and holding times. Additionally in network level policy optimization another level of uncertainty is caused by not quite up-to-date information on the states of far away links; thus one should also have error bounds for the link shadow price on a distant link given that the state of that link a few moments ago is known.

Bibliography

[1] Rabah W. Aldhaheri and Hassan K. Khalil. Aggregation of the policy iteration method for nearly completely decomposable Markov chains. *IEEE Transactions on Automatic Control*, 36(2):178–187, February 1991.

[2] E. Anderson, Z. Bai, C. Bischof, et al. *LAPACK Users' Guide*. SIAM, Philadelphia, third edition, 1999.

[3] Dimitri P. Bertsekas and David A. Castañon. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Transactions on Automatic Control*, 34(6):589–598, June 1989.

[4] D. Y. Burman, J. P. Lehoczky, and Y. Lim. Insensitivity of blocking probabilities in a circuit-switching network. *Journal of Applied Probability*, 21(4):850–859, 1984.

[5] Jeffrey P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, 16(9):527–531, September 1973.

[6] Françoise Chatelin. Iterative aggregation / disaggregation methods. In *Mathematical Computer Performance and Reliability (Pisa, 1983)*, pages 199–207. North-Holland, 1984.

[7] P. J. Courtois and P. Semal. Error bounds for the analysis by decomposition of non-negative matrices. In *Mathematical Computer Performance and Reliability (Pisa, 1983)*, pages 209–224. North-Holland, 1984.

[8] Richard E. Crandall. *Topics in advanced scientific computation*. TELOS/Springer-Verlag, Santa Clara, California, 1996.

[9] Zbigniew Dziong. *ATM Network Resource Management*. McGraw-Hill, New York, 1997.

[10] Zbigniew Dziong and Ke-Qiang Liao. Reward maximization as a common basis for routing, management and planning in ISDN. In L. Lada, editor, *Network Planning in the 1990's*, pages 123–130. North-Holland, 1989.

[11] Zbigniew Dziong, Ke-Qiang Liao, Lorne Mason, and Nicole Tetreault. Bandwidth management in ATM networks. In A. Jensen and V. B. Iversen, editors, *Proceedings of the 13th International Teletraffic Congress*, pages 821–827, 1991.

[12] Zbigniew Dziong and Lorne Mason. An analysis of near optimal call admission and routing model for multi-service loss networks. In *Proceedings of the Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM'92*, volume 1, pages 141–152, 1992.

[13] Zbigniew Dziong and Lorne G. Mason. Call admission and routing in multi-service loss networks. *IEEE Transactions on Communications*, 42(2):2011–2022, 1994.

[14] Zbigniew Dziong, Josée Mignault, and Catherine Rosenberg. Blocking evaluation for networks with reward maximization routing. In *Proceedings of the Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM'93*, volume 2, pages 593–601. IEEE, 1993.

[15] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Numerical linear algebra and optimization*, volume 1. Addison-Wesley, Redwood City, California, 1990.

[16] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, second edition, 1989.

[17] Daniel P. Heyman and Matthew J. Sobel. *Stochastic Models in Operations Research Volume I: Stochastic Processes and Operating Characteristics*. McGraw-Hill, 1982.

[18] Daniel P. Heyman and Matthew J. Sobel. *Stochastic Models in Operations Research Volume II: Stochastic Optimization*. McGraw-Hill, 1984.

[19] Ren-Hung Hwang, James F. Kurose, and Don Towsley. State dependent routing for multirate loss networks. In *GLOBECOM'92*, volume 1, pages 565–570. IEEE, 1992.

[20] Villy 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*, page IX 3, Lund, Sweden, August 1987. Lund Tekniska Högskola.

[21] Villy B. Iversen. *Data- og teletrafikteori*. Den Private Ingeniørfond, Denmark, 1999. An English version is forthcoming.

[22] Villy B. Iversen and Yun Liu. The performance of convolution algorithms for evaluating the total load in an isdn system. In *NTS-9, Det niende nordiske teletrafikkseminaret*, Kjeller, Norway, August 1990. Teledirektoratets forskningavdeling.

[23] Joseph S. Kaufman. Blocking in a shared resource environment. *IEEE Transactions on Communications*, 29(10):1474–1481, October 1981.

[24] F. P. Kelly. *Reversibility and stochastic networks*. John Wiley & Sons, 1979.

[25] David S. Kim and Robert L. Smith. An exact aggregation/disaggregation algorithm for large scale Markov chains. *Naval Research Logistics*, 42(7):1115–1128, 1995.

[26] Aleksandar Kolarov and Joseph Hui. On computing Markov decision theory-based cost for routing in circuit-switched broadband networks. *Journal of Network and Systems Management*, 3(4):405–426, 1995.

[27] Udo R. Krieger. On a two-level multigrid solution method for finite Markov chains. *Linear Algebra and its Applications*, (223/224):415–438, 1995.

[28] K. R. Krishnan. Markov decision algorithms for dynamic routing. *IEEE Communications Magazine*, pages 66–69, October 1990.

[29] K. R. Krishnan and F. Hübner-Szabo de Bucs. Admission control and state-dependent routing for multirate circuit-switched traffic. In V. Ramaswami and P. E. Wirth, editors, *Proceedings of the 15th ITC*, pages 1043–1055. Elsevier Science B. V., 1997.

[30] K. R. Krishnan and T. J. Ott. State-dependent routing for telephone traffic: Theory and results. In *Proceedings of the 25th Conference on Decision and Control*, pages 2124–2128, Athens, Greece, 1986. IEEE.

[31] Chin-Tau Lea and Kai-Wei Ke. Quantization and cost computation of MDP-based admission and routing. In *Proceedings of the 15th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM'96*, volume 3, pages 1004–1011, 1996.

[32] Jan Mandel and Bohuslav Sekerka. A local convergence proof for the iterative aggregation method. *Linear Algebra and its Applications*, 51:163–172, 1983.

[33] Ivo Marek and Petr Mayer. Convergence analysis of an iterative aggregation/disaggregation method for computing stationary probability vectors of stochastic matrices. *Numerical Linear Algebra with Applications*, 5:253–274, 1998.

[34] Ivo Marek and Daniel B. Szyld. Local convergence of the (exact and inexact) iterative aggregation method for linear systems and Markov operators. *Numerische Mathematik*, 69(1):61–82, 1994.

[35] Roy Mendelssohn. An iterative aggregation procedure for Markov decision processes. *Operations Research*, 30(1):62–73, 1982.

[36] George L. Nemhauser. *Introduction to Dynamic Programming*. John Wiley & Sons, 1966.

[37] Marcel F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. The Johns Hopkins University Press, Baltimore, 1981.

[38] Thomas G. Robertazzi. *Computer Networks and Systems: Queuing Theory and Performance Evaluation*. Springer-Verlag, 1990.

[39] James W. Roberts. A service system with heterogenous user requirements—application to multi-services telecommunications systems. In G. Pujolle, editor, *Performance of Data Communication Systems and their Applications*, pages 423–431. North–Holland, 1981.

[40] Keith W. Ross. *Multiservice Loss Models for Broadband Telecommunication Networks*. Springer, 1995.

[41] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, Massachusetts, 1996.

[42] Paul J. Schweitzer. Aggregation methods for large Markov chains. In *Mathematical Computer Performance and Reliability (Pisa, 1983)*, pages 275–286. North-Holland, 1984.

[43] Paul J. Schweitzer. A survey of aggregation-disaggregation in large Markov chains. In W. J. Stewart, editor, *Numerical Solution of Markov Chains*, pages 63–88. Marcel Dekker, New York, 1991.

[44] Paul J. Schweitzer and Kyle W. Kindle. Iterative aggregation for solving undiscounted semi-Markovian reward processes. *Communications in Statistics: Stochastic Models*, 2(1):1–41, 1986.

[45] Paul J. Schweitzer, Martin L. Puterman, and Kyle W. Kindle. Iterative aggregation-disaggregation procedures for discounted semi-Markov reward processes. *Operations Research*, 33(3):589–605, May–June 1985.

[46] Paul J. Schweitzer and Abraham Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110:568–582, 1985.

[47] Herbert A. Simon and Albert Ando. Aggregation of variables in dynamic systems. *Econometrica*, 29(2):111–138, April 1961.

[48] Tomáš Skalický. LASPACK reference manual. Dresden University of Technology, August 1995. Available online at http://www.tu-dresden.de/mwism/skalicky/laspack/laspack.html and http://www.netlib.org/linalg/laspack.tgz. .

[49] Hamdy A. Taha. *Operations research: an introduction*. Prentice-Hall, Upper Saddle River, New Jersey, sixth edition, 1997.

[50] Henk C. Tijms. *Stochastic Models: An Algorithmic Approach*. John Wiley & Sons, Chichester, 1994.

[51] Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[52] Danny H. K. Tsang and Keith W. Ross. Algorithms to determine exact blocking probabilities for multirate tree networks. *IEEE Transactions on Communications*, 38(8), August 1990.

[53] I. Ya. Vakhutinsky, L. M. Dudkin, and A. A. Ryvkin. Iterative aggregation—a new approach to the solution of large-scale problems. *Econometrica*, 47(4):821–840, July 1979.

[54] Ward Whitt. Approximations of dynamic programs. *Mathematics of Operations Research*, 3(3):231–243, August 1978.