The Achievable Region Method in the Optimal Control of Queueing Systems; Formulations, Bounds and Policies

Dimitris Bertsimas

OR 310-95 June 1995

The achievable region method in the optimal control of queueing systems; formulations, bounds and policies

Dimitris Bertsimas
Sloan School of Management and Operations Research Center
Massachusetts Institute of Technology, E53-359
Cambridge, MA 02139

Revised June 1995

Abstract

We survey a new approach that the author and his co-workers have developed to formulate stochastic control problems (predominantly queueing systems) as *mathematical programming problems*. The central idea is to characterize the region of achievable performance in a stochastic control problem, i.e., find linear or nonlinear constraints on the performance vectors that all policies satisfy. We present linear and nonlinear relaxations of the performance space for the following problems: Indexable systems (multiclass single station queues and multiarmed bandit problems), restless bandit problems, polling systems, multiclass queueing and loss networks. These relaxations lead to bounds on the performance of an optimal policy. Using information from the relaxations we construct heuristic nearly optimal policies. The theme in the paper is the thesis that better formulations lead to deeper understanding and better solution methods. Overall the proposed approach for stochastic control problems parallels efforts of the mathematical programming community in the last twenty years to develop sharper formulations (polyhedral combinatorics and more recently nonlinear relaxations) and leads to new insights ranging from a complete characterization and new algorithms for indexable systems to tight lower bounds and nearly optimal algorithms for restless bandit problems, polling systems, multiclass queueing and loss networks.

Keywords: Queueing networks, loss networks, multiarmed bandits, bounds, policies, optimization.

1 Introduction

In its thirty-years history the area of optimal control of stochastic systems (predominantly queueing systems) has addressed with various degrees of success several key problems that arise in areas as diverse as computer and communication networks, manufacturing and service systems. A general characteristic of this body of research is the lack of a unified method of attack for these problems. Every problem seems to require its own formulation and, as a result, its own somewhat ad hoc approach. Moreover, quite often it is not clear how close a proposed solution is to the optimal solution.

In contrast, the field of mathematical programming has evolved around a very small collection of key problem formulations: network, linear, integer and nonlinear programs. In this tradition, researchers and practitioners solve optimization problems by defining decision variables and formulating constraints, thus describing the feasible space of decisions, and applying algorithms for the solution of the underlying optimization problem. When faced with a new deterministic optimization problem, researchers and practitioners have indeed an a priori well-defined plan of attack to solve it: model it as a network, linear, integer or nonlinear program and then use a well-established algorithm for its solution. In this way they obtain feasible solutions which are either provably optimal or with a guarantee for the degree of their suboptimality.

Our goal in this paper is to review an approach to formulate stochastic control problems (predominantly queueing systems) as *mathematical programming problems.* In this way we are able to produce bounds on the performance of an optimal policy and develop optimal or near-optimal policies using information from the formulations. We address the following problems, which, are, in our opinion, among the most important in applications and among the richest in modeling power and structure (detailed definitions of the problems are included in the corresponding sections of the paper):

1. Indexable systems (multiclass queues and multiarmed bandit problems).
2. Restless bandit problems.
3. Polling systems.
4. Multiclass queueing networks.
5. Multiclass loss networks.

In order to motivate the approach we first comment on the power of formulations in mathematical optimization, then describe the approach of the paper in stochastic control in broad terms and put it in a historical perspective.

On the power of formulations in mathematical optimization

Our ability to solve efficiently mathematical optimization problems is to a large degree proportional to our ability to formulate them. In particular, if we can formulate a problem as a *linear optimization problem* (with a polynomial number of variables and constraints, or with a polynomial number of variables

and exponential number of constraints such that we can solve the separation problem in polynomial time) we can solve the problem efficiently (in polynomial time). On the other hand, the general *integer optimization problem* (IP) is rather difficult (the problem is $\mathcal{NP}$-hard). Despite its hardness, the mathematical programming community has developed methods to successfully solve large scale instances. The central idea in this development has been *improved formulations*. Over the last twenty years much of the effort in integer programming research has been in developing sharper formulations using polyhedral methods and more recently techniques from semidefinite optimization (see for example Lovász and Schrijver [39]). Given that linear programming relaxations provide bounds for IP, it is desirable to enhance the formulation of an IP by adding valid inequalities, such that the relaxation is closer and closer to the IP. We outline the idea of improved formulations in the context of IP in Figure 1a.

On the power of formulations in stochastic control

Motivated by the power of improved formulations for mathematical programming problems, we shall now outline in broad terms the approach taken in this paper to formulate stochastic control problems as *mathematical programming problems*. The key idea is the following: Given a stochastic control problem, we define a vector of performance measures (these are typically expectations, but not necessarily first moments) and then we express the objective function as a function of this vector. The critical idea is to characterize *the region of achievable performance* (or performance space), i.e., find constraints on the performance vectors that all policies satisfy. In this way we find a series of relaxations that are progressively closer to the exact region of achievable performance. In Figure 1b we outline the conceptual similarity of this approach to the approach used in integer programming. Interestingly we will see that we can propose both linear and nonlinear relaxations (the latter also based on ideas from semidefinite programming).

The idea of describing the performance space of a queueing control problem has its origin in the work of Coffman and Mitrani [15] and Gelenbe and Mitrani [20], who first showed that the performance space of a multiclass $M/G/1$ queue under the average-cost criterion can be described as a polyhedron. Federgruen and Groenevelt [18], [19] advanced the theory further by observing that in certain special cases of multiclass queues, the polyhedron has a very special structure (it is a *polymatroid*) that gives rise to very simple optimal policies (the $c\mu$ rule). Their results partially extend to some rather restricted multiclass queueing networks, in which it is assumed that all classes of customers have the same routing probabilities, and the same service requirements at each station of the network (see Ross and Yao [47]). Shanthikumar and Yao [50] generalized the theory further by observing that if a system satisfies certain *work conservation laws*, then the underlying performance space is necessarily a polymatroid. They also proved that, when the cost is linear on the performance, the optimal policy is a *static priority rule* (Cox and Smith [16]). Tsoucas [57] derived the region of achievable performance in the problem of scheduling a multiclass nonpreemptive $M/G/1$ queue with Bernoulli feedback, introduced by Klimov [35].

Bertsimas and Niño-Mora [6] characterize the region of achievable performance of a class of stochas-

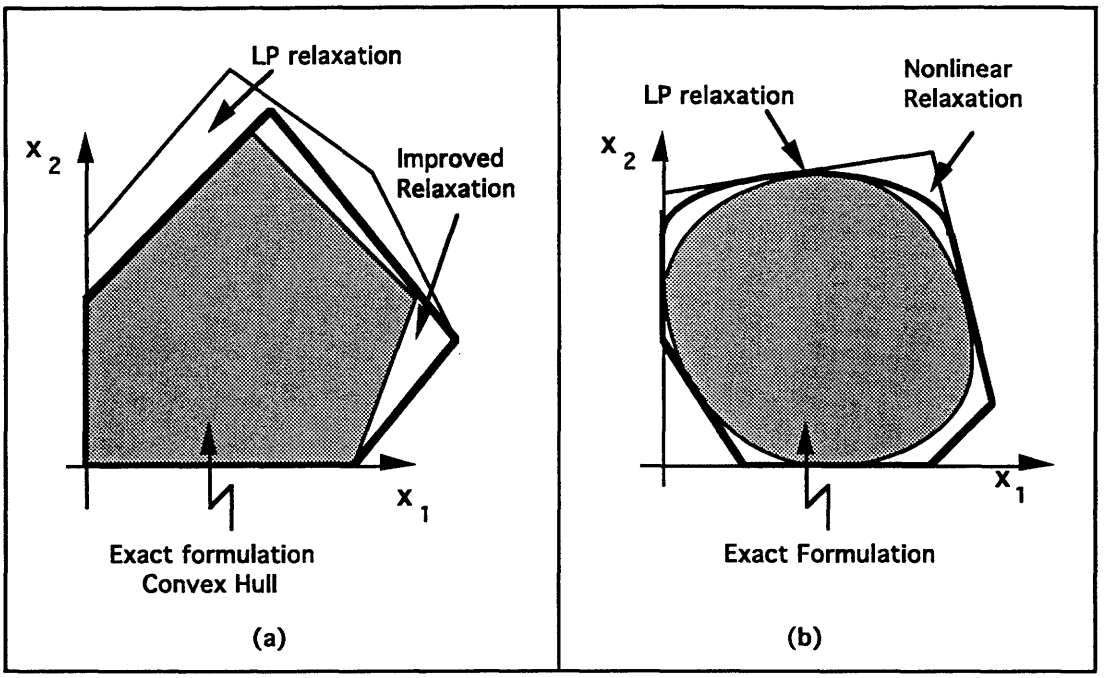

Figure 1: (a) Improved relaxations for integer programming (b) Improved relaxations for stochastic optimization problems.

tic control problems that include all the previous queueing control and multiarmed bandit problems, establishing that the strong structural properties for these problems follow from the corresponding properties of the underlying polyhedra. Interestingly this exact characterization of the achievable region as a polyhedron leads to an exact algorithm that uses linear programming duality and naturally defines indices (Gittins indices) as dual variables. We review these developments in Section 2.

For restless bandit problems Whittle [62] proposes a relaxation that provides a bound for the problem. Bertsimas and Niño-Mora [7] propose a series of increasing more complicated relaxations (the last one being exact) that give increasingly stronger bounds and a heuristic that naturally arises from these relaxations using duality that empirically gives solutions which are close to the optimal one. We review these developments in Section 3.

For polling systems Bertsimas and Xu [10] propose a nonlinear (but convex) relaxation, and a heuristic that uses integer programming techniques, that provides near optimal static policies. We review these developments in Section 4.

For queueing network problems Bertsimas, Paschalidis and Tsitsiklis [8] propose a linear programming relaxation that provides bounds for the optimal solution using a systematic method to generate linear constraints on the achievable region. In this paper we propose stronger nonlinear relaxations that are based on ideas from semidefinite programming. We review these developments in Section 5.

For loss networks Kelly [33] has developed bounds based on nonlinear optimization, while Bertsimas and Crissikou [5] propose a series of increasingly more complex linear relaxations (the last one being

exact) and a heuristic that arises from these relaxations. We review these developments in Section 6.

In Section 7 we present some theoretical limitations of the current approach using some negative complexity results, while in Section 8 we conclude with some open problems.

2 Indexable systems

Perhaps one of the most important successes in the area of stochastic control in the last twenty years is the solution of the celebrated *multiarmed bandit problem*, a generic version of which in discrete time is as follows:

The multiarmed bandit problem: There are K parallel projects, indexed $k = 1, \ldots, K$. Project k can be in one of a finite number of states $j_k \in E_k$. At each instant of discrete time $t = 0, 1, \ldots$ one can work only on a single project. If one works on project k in state $j_k(t)$ at time t, then one receives an immediate reward of $R_{kj_k(t)}$. Rewards are additive and discounted in time by a discount factor $0 < \beta < 1$. The state $j_k(t)$ changes to $j_k(t+1)$ by a homogeneous Markov transition rule, with transition matrix $\mathbf{P}^k = (p^k_{ij})_{i,j \in E_k}$, while the states of the projects one has not engaged remain frozen. The problem is how to allocate one's resources to projects sequentially in time in order to maximize expected total discounted reward over an infinite horizon. More precisely, one chooses a policy u (among a set of policies $\mathcal{U}$) for specifying the project $k(t)$ to work on at each point of time t to achieve:

$$\max_{u \in \mathcal{U}} \mathrm{E}_u \left[\sum_{t=0}^{\infty} \beta^t R_{k(t) j_{k(t)}(t)} \right].$$

The problem has numerous applications and a rather vast literature (see Gittins [23] and the references therein). It was originally solved by Gittins and Jones [21], who proved that to each project k one could attach an *index* $\gamma^k(j_k(t))$, which is a function of the project k and the current state $j_k(t)$ alone, such that the optimal action at time t is to engage a project of largest current index. They also proved the important result that these index functions satisfy a stronger *index decomposition* property: the function $\gamma^k(\cdot)$ only depends on characteristics of project k (states, rewards and transition probabilities), and not on any other project. These indices are now known as Gittins indices, in recognition of Gittins contribution. Since the original solution, which relied on an interchange argument, other proofs were proposed: Whittle [61] provided a proof based on dynamic programming, subsequently simplified by Tsitsiklis [55]. Varaiya, Walrand and Buyukkoc [58] and Weiss [60] provided different proofs based on interchange arguments. Weber [59] and Tsitsiklis [56] outlined intuitive proofs.

The multiarmed bandit problem is a special case of a stochastic *service system* $\mathcal{S}$. In this context, there is a finite set E of job types. Jobs have to be scheduled for service in the system. We are interested in optimizing a function of a performance measure (rewards or taxes) under a class of *admissible* scheduling policies. A policy is called admissible if the decision as to which project to operate at any time t must be based only on information on the current states of the projects.

Definition 1 (Indexable Systems) We say that a stochastic *service system* $\mathcal{S}$ is *indexable* if the following policy is optimal: to each job type i we attach an index γ_i. At each decision epoch select a job with largest current index.

In general the optimal indices γ_i (as functions of the parameters of the system) could depend on characteristics of the entire set E of job types. Consider a partition of set E into subsets E_k, for $k = 1, \ldots, K$. Job types in subset E_k can be interpreted as being part of a common project type k. In certain situations, the optimal indices of job types in E_k depend only on characteristics of job types in E_k and not on the entire set E. Such a property is particularly useful computationally since it enables the system to be decomposed into smaller components and the computation of the indices for each component can be done independently. As we have seen the multiarmed bandit problem has this *decomposition* property, which motivates the following definition:

Definition 2 (Decomposable Systems) An indexable system is called *decomposable* if for all job types $i \in E_k$, the optimal index γ_i of job type i depends only on characteristics of job types in E_k.

In addition to the multiarmed bandit problem, a variety of stochastic control problems has been solved in the last decades by indexing rules (see Table 3 for examples).

Faced with these results, one asks what is the underlying *reason* that these nontrivial problems have very efficient solutions both theoretically as well as practically. In particular, *what is the class of stochastic control problems that are indexable? Under what conditions are indexable systems decomposable?* But most importantly, *is there a unified way to address stochastic control problems that will lead to a deeper understanding of their structural properties?*

In this section we review the work of Bertsimas and Niño-Mora [6]. In Section 2.1 we introduce via an example of a multiclass queue with feedback, which is a special case of the multiarmed bandit problem, the idea of work conservation laws. These physical laws of the system constrain the region of achievable performance and imply that it is a polyhedron of special structure called extended contra-polymatroid. In this way the stochastic control problem can be reduced to a linear programming problem over this polyhedron. In Section 2.2 we propose an efficient algorithm to solve the linear programming problem and observe that the algorithm naturally defines indices. In Sections 2.3 we consider examples of natural problems that can be analyzed using the theory developed, while in Section 2.4 we outline implications of these characterizations.

2.1 Work conservation laws

In order to present the idea of conservation laws we consider the so called Klimov problem with $n = 3$ classes: three Poisson arrival streams with rates λ_i arrive at a single server. Each class has a different service distribution. After service completion, a job of class i becomes a job of class j with probability

p_{ij} and leaves the system with probability $1 - \sum_k p_{ik}$. Let x_i^u be the expected queue length of class i under policy u. What is the space of vectors (x_1^u, x_2^u, x_3^u) as the policies u vary? For every policy

$$A_i^{\{i\}} x_i^u \geq b(\{i\}), \ i = 1, 2, 3$$

which means that the total work class i brings to the system under any policy is at least as much as the work under the policy that gives priority to class i. Similarly, for every subset of classes, the total work that this set of classes brings to the system under any policy is at least as much as the work under the policy that gives priority to this set of classes:

$$A_1^{\{1,2\}} x_1 + A_2^{\{1,2\}} x_2 \geq b(\{1,2\}),$$

$$A_1^{\{1,3\}} x_1 + A_3^{\{1,3\}} x_3 \geq b(\{1,3\}),$$

$$A_2^{\{2,3\}} x_2 + A_3^{\{2,3\}} x_3 \geq b(\{2,3\}).$$

Finally, there is work conservation, i.e., all nonidling policies require the same total work:

$$A_1^{\{1,2,3\}} x_1 + A_2^{\{1,2,3\}} x_2 + A_3^{\{1,2,3\}} x_3 = b(\{1,2,3\}).$$

The performance space in this example consists exactly of the $2^3 - 1 = 7$ constraints. In Figure 2 we present the performance space for this example.

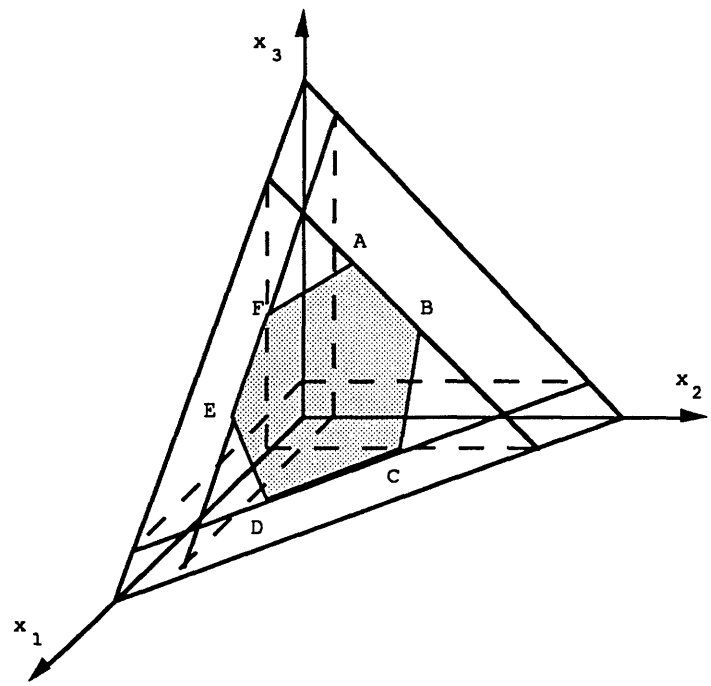

Figure 2: An extended contra-polymatroid in dimension 3.

The polytope has $3! = 6$ extreme points corresponding to the 6 permutations of a set of 3 elements. Extreme point A corresponds to the performance vector of the policy that gives highest priority to class

1, then to class 2 and last priority to class 3. The vector corresponding to point A is the unique solution of the triangular system:

$$A_1^{\{1\}} x_1 = b(\{1\}),$$

$$A_1^{\{1,2\}} x_1 + A_2^{\{1,2\}} x_2 = b(\{1,2\}),$$

$$A_1^{\{1,2,3\}} x_1 + A_2^{\{1,2,3\}} x_2 + A_3^{\{1,2,3\}} x_3 = b(\{1,2,3\}).$$

Polyhedra of this type are called extended contra-polymatroids and have the property that their extreme points correspond to the performance vectors of complete priority policies.

In order to present these ideas more generally we consider next a generic stochastic *service system.* There are n job types, which we label $i \in E = \{1, \ldots, n\}$. Jobs have to be scheduled for service in the system. Let us denote $\mathcal{U}$ the class of *admissible* scheduling policies. Let x_i^u be a performance measure of type i jobs under admissible policy u, for $i \in E$. We assume that x_i^u is an expectation. Let $\mathbf{x}_\pi$ denote the performance vector under a *static priority rule* (i.e., the servicing priority of a job does not change over time) that assigns priorities to the job types according to permutation $\pi = (\pi_1, \ldots, \pi_n)$ of E, where type π_n has the highest priority.

Definition 3 (Conservation Laws) The performance vector $\mathbf{x}$ is said to satisfy *conservation laws* if there exist a set function $b(\cdot) : 2^E \to \Re_+$ and a matrix $\mathbf{A} = (A_i^S)_{i \in E, S \subseteq E}$ satisfying

$$A_i^S > 0, \quad \text{for} \quad i \in S \qquad \text{and} \qquad A_i^S = 0, \quad \text{for} \quad i \in S^c, \qquad \text{for all } S \subseteq E,$$

such that:

(a)

$$b(S) = \sum_{i \in S} A_i^S x_i^\pi \quad \text{for all } \pi : \{\pi_1, \ldots, \pi_{|S|}\} = S \quad \text{and} \quad S \subseteq E; \tag{1}$$

(b) For all admissible policies $u \in \mathcal{U}$,

$$\sum_{i \in S} A_i^S x_i^u \geq b(S) \quad \text{for all } S \subset E \qquad \text{and} \qquad \sum_{i \in E} A_i^E x_i^u = b(E), \tag{2}$$

or respectively,

$$\sum_{i \in S} A_i^S x_i^u \leq b(S) \quad \text{for all } S \subset E \qquad \text{and} \qquad \sum_{i \in E} A_i^E x_i^u = b(E). \tag{3}$$

Condition (1) means that all policies which give priority to the set $E \setminus S$ over the set S have the same performance irrespective of the rule used within S or within $E \setminus S$. Moreover, all such policies have the same performance $b(S)$ (conservation law). Condition (2) means that all other policies have performance bigger than $b(S)$, while for $S = E$ all policies have the same performance.

The following theorem characterizes the performance space for a performance vector $\mathbf{x}$ satisfying conservation laws as a polyhedron.

Theorem 1 *Assume that performance vector* $\mathbf{x}$ *satisfies conservation laws* (1) *and* (2) *(respectively* (1) *and* (3)). *Then*

(a) The performance space for the vector $\mathbf{x}$ *is exactly the polyhedron*

$$\mathcal{B}_c(\mathbf{A},\mathbf{b}) = \{\mathbf{x} \in \Re_+^n : \sum_{i\in S} A_i^S x_i \geq b(S), \quad \textit{for } S \subset E \quad \textit{and} \quad \sum_{i\in E} A_i^E x_i = b(E)\}, \tag{4}$$

or respectively

$$\mathcal{B}(\mathbf{A},\mathbf{b}) = \{\mathbf{x} \in \Re_+^n : \sum_{i\in S} A_i^S x_i \leq b(S), \quad \textit{for } S \subset E \quad \textit{and} \quad \sum_{i\in E} A_i^E x_i = b(E)\}. \tag{5}$$

(b) The vertices of $\mathcal{B}_c(\mathbf{A},\mathbf{b})$ *(respectively* $\mathcal{B}(\mathbf{A},\mathbf{b})$*) are the performance vectors of the static priority rules.*

We can explicitly characterize the performance vectors corresponding to priority rules as follows. Let $\pi = (\pi_1, \ldots, \pi_n)$ be a permutation of E corresponding to the static priority rule π. We let $\mathbf{x}_\pi = (x_{\pi_1}, \ldots, x_{\pi_n})'$ be the performance vector corresponding to the static priority rule π. We also let

$$\mathbf{b}_\pi = (b(\{\pi_1\}), b(\{\pi_1,\pi_2\}), \ldots, b(\{\pi_1,\ldots,\pi_n\}))'.$$

Let $\mathbf{A}_\pi$ denote the following lower triangular submatrix of $\mathbf{A}$:

$$\mathbf{A}_\pi = \begin{pmatrix} A_{\pi_1}^{\{\pi_1\}} & 0 & \ldots & 0 \\ A_{\pi_1}^{\{\pi_1,\pi_2\}} & A_{\pi_2}^{\{\pi_1,\pi_2\}} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{\pi_1}^{\{\pi_1,\ldots,\pi_n\}} & A_{\pi_2}^{\{\pi_1,\ldots,\pi_n\}} & \ldots & A_{\pi_n}^{\{\pi_1,\ldots,\pi_n\}} \end{pmatrix}.$$

Then, the performance vector $\mathbf{x}_\pi$ is the unique solution of the linear system

$$\sum_{i=1}^{j} A_{\pi_i}^{\{\pi_1,\ldots,\pi_j\}} x_{\pi_i} = b(\{\pi_1,\ldots,\pi_j\}), \quad j = 1,\ldots,n \tag{6}$$

or, in matrix notation:

$$\mathbf{x}_\pi = \mathbf{A}_\pi^{\leftarrow 1}\mathbf{b}_\pi. \tag{7}$$

It is not obvious that the vector $\mathbf{x}_\pi \in \mathcal{B}_c(\mathbf{A},\mathbf{b})$ found in (7). This is a consequence of the fact that the performance vectors satisfy conservation laws. This motivates the following definition based on Bhattacharya et. al. [12]:

Definition 4 (Extended Polymatroid) We say that polyhedron $\mathcal{B}(\mathbf{A},\mathbf{b})$ is an *extended polymatroid* with ground set E and correspondingly $\mathcal{B}_c(\mathbf{A},\mathbf{b})$ is an *extended contra-polymatroid* if for every permutation π of E, $\mathbf{x}_\pi \in \mathcal{B}(\mathbf{A},\mathbf{b})$ or correspondingly $\mathbf{x}_\pi \in \mathcal{B}_c(\mathbf{A},\mathbf{b})$.

The connection of conservation laws and extended polymatroids is that if a performance vector $\mathbf{x}$ satisfies conservation laws (1) and (2) (resp. (1) and (3)), then its performance space $\mathcal{B}_c(\mathbf{A},\mathbf{b})$ in (4) (resp. $\mathcal{B}(\mathbf{A},\mathbf{b})$ in (5)) is an extended polymatroid (resp. contra-polymatroid).

Remark: If $A_i^S = 1$ for all $i \in S$, $\mathbf{x}_\pi \in \mathcal{B}(\mathbf{A}, \mathbf{b})$ if and only if the set function $b(S)$ is submodular, i.e., for all $S, T \subset E$,

$$b(S) + b(T) \leq b(S \cap T) + b(S \cup T).$$

This case corresponds to the usual polymatroids introduced in Edmonds [17].

2.2 An algorithm for optimization of indexable systems

Suppose that we want to find an admissible policy $u \in \mathcal{U}$ that maximizes a linear reward function on the performance $\sum_{i \in E} R_i x_i^u$, where the vector $\mathbf{x}$ satisfies conservation laws (1) and (2). This optimal control problem can be expressed as

$$(LP_{\mathcal{U}}) \qquad \max\left\{ \sum_{i \in E} R_i x_i^u : u \in \mathcal{U} \right\}. \tag{8}$$

By Theorem 1 this optimal control problem can be transformed into the following linear programming problem over an extended contra-polymatroid.

$$(LP_c) \qquad \max\left\{ \sum_{i \in E} R_i x_i : \mathbf{x} \in \mathcal{B}_c(\mathbf{A}, \mathbf{b}) \right\}. \tag{9}$$

The theory over extended polymatroids is completely analogous. Given that the extreme points of the polytope $\mathcal{B}_c(\mathbf{A}, \mathbf{b})$ correspond to priorities (permutations), the problem reduces to finding which priority is optimal.

Tsoucas [57] and Bertsimas and Niño-Mora [6] presented the following *adaptive greedy* algorithm. The algorithm constructs an optimal solution $\overline{\mathbf{y}}$ to the linear programming dual of problem (LP_c). It first decides, which class will have the highest priority π_n by calculating the maximum $\frac{R_i}{A_i^E}$, $i \in E$ (this is the greedy part of the algorithm). This value is assigned to x_{π_n}. It then modifies the rewards (this is the adaptive part of the algorithm) and picks the next class by performing a similar maximum ratio test. The algorithm also calculates numbers γ_i, which define indices. We next formally present the algorithm.

Algorithm $\mathcal{AG}$

Input: $(\mathbf{R}, \mathbf{A})$.

Output: $(\gamma, \pi, \overline{\mathbf{y}}, \nu, \mathcal{S})$, where $\pi = (\pi_1, \ldots, \pi_n)$ is a permutation of E, $\overline{\mathbf{y}} = \left(\overline{y}^S\right)_{S \subseteq E}$, $\nu = (\nu_1, \ldots, \nu_n)$, and $\mathcal{S} = \{S_1, \ldots, S_n\}$, with $S_k = \{\pi_1, \ldots, \pi_k\}$, for $k \in E$.

Step 0. Set $S_n = E$. Set $\nu_n = \max\left\{ \frac{R_i}{A_i^E} : i \in E \right\}$;

pick $\pi_n \in \operatorname{argmax}\left\{ \frac{R_i}{A_i^E} : i \in E \right\}$;

set $\gamma_{\pi_n} = \nu_n$.

Step k. For $k = 1, \ldots, n-1$:

Set $S_{n-k} = S_{n-k+1} \setminus \{\pi_{n-k+1}\}$; set $\nu_{n-k} = \max\left\{ \frac{R_i - \sum_{j=0}^{k-1} A_i^{S_{n-j}} \nu_{n-j}}{A_i^{S_{n-k}}} : i \in S_{n-k} \right\}$;

pick $\pi_{n-k} \in \operatorname{argmax}\left\{ \frac{R_i - \sum_{j=0}^{k-1} A_i^{S_{n-j}} \nu_{n-j}}{A_i^{S_{n-k}}} : i \in S_{n-k} \right\}$;

set $\gamma_{\pi_{n-k}} = \gamma_{\pi_{n-k+1}} + \nu_{n-k}$.

Problem	Feasible space	Indices	Optimal solution
$(LP)\min_{\mathbf{x}\in\mathcal{B}(\mathbf{A},\mathbf{b})}\mathbf{Cx}$	$\mathcal{B}(\mathbf{A},\mathbf{b})^a:\begin{cases}\sum_{i\in S}A_i^S x_i\le b(S), S\subseteq E\\ \sum_{i\in E}A_i^E x_i=b(E)\\ \mathbf{x}\ge\mathbf{0}\end{cases}$	$(\mathbf{C},\mathbf{A})\overset{\mathcal{AG}}{\longmapsto}\gamma$	$\gamma_{\pi_1}\le\cdots\le\gamma_{\pi_n}$ $\overline{\mathbf{x}}_\pi=\mathbf{A}_\pi^{-1}\mathbf{b}_\pi$
$(LP_c)\max_{\mathbf{x}\in\mathcal{B}_c(\mathbf{A},\mathbf{b})}\mathbf{Rx}$	$\mathcal{B}_c(\mathbf{A},\mathbf{b})^b:\begin{cases}\sum_{i\in S}A_i^S x_i\ge b(S), S\subseteq E\\ \sum_{i\in E}A_i^E x_i=b(E)\\ \mathbf{x}\ge\mathbf{0}\end{cases}$	$(\mathbf{R},\mathbf{A})\overset{\mathcal{AG}}{\longmapsto}\gamma$	$\gamma_{\pi_1}\le\cdots\le\gamma_{\pi_n}$ $\overline{\mathbf{x}}_\pi=\mathbf{A}_\pi^{-1}\mathbf{b}_\pi$

Table 1: Linear programming over extended polymatroids

[a] Extended polymatroid
[b] Extended contra-polymatroid

Step n. For $S\subseteq E$ set

$$\overline{y}^S=\begin{cases}\nu_j, & \text{if } S=S_j \text{ for some } j\in E;\\ 0, & \text{otherwise.}\end{cases}$$

Bertsimas and Niño-Mora [6] show that outputs γ and $\overline{\mathbf{y}}$ of algorithm $\mathcal{AG}$ are uniquely determined by the input $(\mathbf{R},\mathbf{A})$. Moreover, $\overline{\mathbf{y}}$ is an optimal solution to the linear programming dual of (LP_c). The above algorithm runs in $O(n^3)$. They also show, using linear programming duality, that linear program (LP_c) has the following indexability property.

Definition 5 (Generalized Gittins Indices) Let $\overline{\mathbf{y}}$ be the optimal dual solution generated by algorithm $\mathcal{AG}$. Let

$$\gamma_i^*=\sum_{S:\ E\supseteq S\ni i}\overline{y}^S,\quad i\in E.$$

We say that $\gamma_1^*,\ldots,\gamma_n^*$ are the *generalized Gittins indices* of linear program (LP_c).

Theorem 2 (Indexability of LP over extended polymatroids) *Linear program (LP_c) is optimized by the vector $\mathbf{x}_\pi$, where π is any permutation of E such that*

$$\gamma_{\pi_1}\le\cdots\le\gamma_{\pi_n}. \tag{10}$$

Let $\gamma_1,\ldots,\gamma_n$ be the generalized Gittins indices of linear program (LP_c). A direct consequence of the previous discussion is that the control problem $(LP_{\mathcal{U}})$ is solved by an index policy, with indices given by $\gamma_1,\ldots,\gamma_n$.

Theorem 3 (Indexability of Systems Satisfying Conservation Laws) *A policy that selects at each decision epoch a job of currently largest generalized Gittins index is optimal for the control problem.*

Bertsimas and Niño-Mora [6] provide closed form expressions for the generalized Gittins indices, which can be used for sensitivity analysis. Optimization over an extended polymatroid is analogous to the extended contra-polymatroid case (see Table 1).

In the case that matrix $\mathbf{A}$ has a certain special structure, the computation of the indices of (LP_c) can be simplified. Let E be partitioned as $E = \bigcup_{k=1}^{K} E_k$. Let $\mathbf{A}^k$ be the submatrix of matrix $\mathbf{A}$ corresponding to subsets $S \subseteq E_k$. Let $\mathbf{R}^k = (R_i)_{i \in E_k}$. Assume that the following condition holds:

$$A_i^S = A_i^{S \cap E_k} = (A^k)_i^{S \cap E_k}, \quad \text{for } i \in S \cap E_k \quad \text{and } S \subseteq E. \tag{11}$$

Let $\{\gamma_i^k\}_{i \in E_k}$ be the generalized Gittins indices obtained from algorithm $\mathcal{AG}$ with input $(\mathbf{R}^k, \mathbf{A}^k)$.

Theorem 4 (Index Decomposition) *Under condition (11), the generalized Gittins indices corresponding to* $(\mathbf{R}, \mathbf{A})$ *and* $(\mathbf{R}^k, \mathbf{A}^k)$ *satisfy*

$$\gamma_i = \gamma_i^k, \quad \textit{for } i \in E_k \quad \textit{and } k = 1, \ldots, K. \tag{12}$$

Theorem 4 implies that the reason for decomposition to hold is (11). A useful consequence is the following:

Corollary 1 *Under the assumptions of Theorem 4, an optimal solution of problem* (LP_c) *can be computed by solving* K *subproblems: Apply algorithm* $\mathcal{AG}$ *with inputs* $(\mathbf{R}^k, \mathbf{A}^k)$, *for* $k = 1, \ldots, K$.

2.3 Examples

In order to apply Algorithm $\mathcal{AG}$ in concrete problems, one has to define appropriate performance vectors, calculate the matrix $\mathbf{A}$, the set function $b(.)$ and the reward vector $\mathbf{R}$. In this section we illustrate how to calculate the various parameters in the context of *branching bandit problems* introduced by Weiss [60], who observed that it can model a large number of stochastic control problems.

There is a finite number of project types, labeled $k = 1, \ldots, K$. A type k project can be in one of a finite number of states $i_k \in E_k$, which correspond to *stages* in the development of the project. It is convenient in what follows to combine these two indicators into a single label $i = (k, i_k)$, the state of a project. Let $E = \{1, \ldots, n\}$ be the finite set of possible states of all project types.

We associate with state i of a project a random time v_i and random arrivals $\mathbf{N}_i = (N_{ij})_{j \in E}$. Engaging the project keeps the system busy for a duration v_i (the duration of stage i), and upon completion of the stage the project is replaced by a nonnegative integer number of new projects N_{ij}, in states $j \in E$. We assume that given i, the durations and the descendants $(v_i; (N_{ij})_{j \in E})$ are random variables with an arbitrary joint distribution, independent of all other projects, and identically distributed for the same i. Projects are to be selected under an admissible policy $u \in \mathcal{U}$: Nonidling (at all times a project is being engaged, unless there are no projects in the system), nonpreemptive (work cannot be interrupted until the engaged project stage is completed) and nonanticipative (decisions are based only on past and present information). The decision epochs are $t = 0$ and the instants at which a project stage is completed and there is some project present.

We shall refer to a project in state i as a *type i job*. In this section, we will define two different performance measures for a branching bandit process. The first one will be appropriate for modelling

a discounted reward-tax structure. The second will allow us to model an undiscounted tax structure. In each case we demonstrate what data is needed, what the performance space is and how the relevant parameters are computed.

Discounted Branching Bandits

Data: Joint distribution of $v_i, N_{i1}, \ldots, N_{in}$ for each i:

$$\Phi_i(\theta, z_1, \ldots, z_n) = \mathrm{E}\Big[e^{-\theta v_i} z_1^{N_{i1}} \ldots z_n^{N_{in}} \Big],$$

$$\Psi_i(\theta) = \Phi_i(\theta, 1, \ldots, 1),$$

m_i, the number of type i bandits initially present, R_i, an instantaneous reward received upon completion of a type i job, C_i, a holding tax incurred continuously while a type i job is in the system and $\alpha > 0$, the discount factor.

Performance measure: Using the indicator

$$I_j(t) = \begin{cases} 1 & \text{if a type } j \text{ job is being engaged at time } t; \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

the performance measure is

$$x_j^u(\alpha) = \mathrm{E}_u \left[\int_0^\infty e^{-\alpha t} I_j(t)\, dt \right]. \tag{14}$$

Performance Space: The performance vector for branching bandits $\mathbf{x}^u(\alpha)$ satisfies conservation laws
(a) $\sum_{i \in S} A^S_{i,\alpha} x_i^u(\alpha) \geq b_\alpha(S)$, for $S \subset E$, with equality if policy u gives complete priority to S^c-jobs.
(b) $\sum_{i \in E} A^E_{i,\alpha} x_i^u(\alpha) = b_\alpha(E)$.
Therefore, the performance space for branching bandits corresponding to the performance vector $\mathbf{x}^u(\alpha)$ is the extended contra-polymatroid base $\mathcal{B}_c(A_\alpha, b_\alpha)$. The matrix $\mathbf{A}_\alpha$ and set function $b_\alpha(\cdot)$ are calculated as follows:

$$A^S_{i,\alpha} = \frac{1 - \Psi_i^{S^c}(\alpha)}{1 - \Psi_i(\alpha)}, \quad i \in S, \quad S \subseteq E; \tag{15}$$

$$b_\alpha(S) = \frac{1}{\alpha} \prod_{j \in S^c} [\Psi_j^{S^c}(\alpha)]^{m_j} - \frac{1}{\alpha} \prod_{j \in E} [\Psi_j^E(\alpha)]^{m_j}, \quad S \subseteq E \tag{16}$$

and the functions $\Psi_i^S(\theta)$ are calculated from the following fixed point system:

$$\Psi_i^S(\theta) = \Phi_i\Big(\theta, \big(\Psi_j^S(\theta)\big)_{j \in S}, 1_{S^c}\Big), \quad i \in E.$$

Objective function:

$$\max_u V^{(\mathbf{R},\mathbf{C})}_{u,\alpha}(m) = \sum_{i \in E} \hat{R}_{i,\alpha} x_i^u(\alpha) - \sum_{i \in E} m_i C_i = \sum_{i \in E} \Big\{ R_i + C_i - \sum_{j \in E} \mathrm{E}[N_{ij}] C_j \Big\} \frac{\alpha \Psi_i(\alpha)}{1 - \Psi_i(\alpha)} x_i^u(\alpha) - \sum_{i \in E} m_i C_i \tag{17}$$

Undiscounted Branching Bandits

We consider the case that the busy period of the branching bandit process is finite with probability 1.

Data: $\mathrm{E}[v_i]$, $\mathrm{E}[v_i^2]$, $\mathrm{E}[N_{ij}]$. Let $E[\mathbf{N}]$ denote the matrix of $E[N_{ij}]$.

Performance measure:

$$I_j(t) = \begin{cases} 1, & \text{if a type } j \text{ job is being engaged at time } t; \\ 0, & \text{otherwise,} \end{cases}$$

$$Q_j(t) = \text{ the number of type } j \text{ jobs in the system at time } t.$$

$$\theta_j^u = \mathrm{E}_u\left[\int_0^\infty I_j(t)t\,dt\right] \quad j \in E.$$

$$W_j^u = \int_0^\infty \mathrm{E}_u[Q_j(t)]\,dt, \quad j \in E.$$

Stability: The branching bandits process is stable (its busy period has finite first two moments) if and only if $E[\mathbf{N}] < \mathbf{I}$ (the eigenvalues of the matrix $E[\mathbf{N}]$ are smaller than one).

Performance space:

I. The performance vector θ^u satisfies the following conservation laws:

(a) $\sum_{i\in S} A_i^S \theta_i^u \le b(S)$, for $S \subset E$, with equality if policy u gives complete priority to S^c-jobs.

(b) $\sum_{i\in E} A_i^E \theta_i^u = b(E)$.

The matrix $\mathbf{A}$ and set function $b(\cdot)$ are calculated as follows:

$$A_i^S = \frac{\mathrm{E}[T_i^{S^c}]}{\mathrm{E}[v_i]}, \quad i \in S, \tag{18}$$

and

$$b(S) = \frac{1}{2}\mathrm{E}[(T_m^E)^2] - \frac{1}{2}\mathrm{E}[(T_m^{S^c})^2] + \sum_{i\in S} \frac{\mathrm{E}[\nu_i]\,\mathrm{E}[v_i^2]}{2}\left(\frac{\mathrm{E}[T_i^{S^c}]}{\mathrm{E}[v_i]} - \frac{\mathrm{E}[(T_i^{S^c})^2]}{\mathrm{E}[v_i^2]}\right), \tag{19}$$

where we obtain $\mathrm{E}[T_i^S]$, $\mathrm{Var}[T_i^S]$ (and thus $\mathrm{E}[(T_i^S)^2]$) and $\mathrm{E}[\nu_j]$ by solving the following linear systems:

$$\mathrm{E}[T_i^S] = \mathrm{E}[v_i] + \sum_{j\in S} \mathrm{E}[N_{ij}]\,\mathrm{E}[T_j^S], \quad i \in E;$$

$$\mathrm{Var}[T_i^S] = \mathrm{Var}[v_i] + \left(\mathrm{E}[T_j^S]\right)_{j\in S}^T \mathrm{Cov}\Big[(N_{ij})_{j\in S}\Big]\left(\mathrm{E}[T_j^S]\right)_{j\in S} + \sum_{j\in S} \mathrm{E}[N_{ij}]\,\mathrm{Var}[T_j^S], i \in E;$$

$$\mathrm{E}[\nu_j] = m_j + \sum_{i\in E} \mathrm{E}[N_{ij}]\,\mathrm{E}[\nu_i], \quad j \in E.$$

Finally,

$$\mathrm{E}[T_m^S] = \sum_{i\in S} m_i\,\mathrm{E}[T_i^S],$$

$$\mathrm{Var}[T_m^S] = \sum_{i\in S} m_i\,\mathrm{Var}[T_i^S].$$

II. The performance vector $\mathbf{W}^u$ satisfies the following conservation laws:

(a) $\sum_{i\in S} A'^S_i W_i^u \ge b'(S)$, for $S \subset E$, with equality if policy u gives complete priority to S-jobs.

(b) $\sum_{i\in E} A_i^E W_i^u = b'(E)$, where

$$A_i'^S = \mathrm{E}[T_i^S], \quad i \in S, \tag{20}$$

Problem	Performance measure	Performance space	Algorithm
$\max_{u\in\mathcal{U}} V_{u,\alpha}^{(\mathbf{R},\mathbf{C})}$	$x_j^u(\alpha) = \int_0^\infty \mathrm{E}_u[I_j(t)]e^{-\alpha t}\,dt$	$\mathcal{B}_c(\mathbf{A}_\alpha, \mathbf{b}_\alpha)^a$ $\mathbf{A}_\alpha$, $\mathbf{b}_\alpha(\cdot)$: see (15), (16)	$(\hat{\mathbf{R}}_\alpha, \mathbf{A}_\alpha) \overset{\mathcal{AG}}{\longmapsto} \gamma(\alpha)$ $\hat{\mathbf{R}}_\alpha$: see (17)
$\min_{u\in\mathcal{U}} V_u^{(\mathbf{0},\mathbf{C})}$ (finite busy period case)	$\theta_j^u = \int_0^\infty \mathrm{E}_u[I_j(t)]t\,dt]$	$\mathcal{B}(\mathbf{A}, \mathbf{b})^b$ $\mathbf{A}$, $\mathbf{b}(\cdot)$: see (18), (19)	$(\hat{\mathbf{C}}, \mathbf{A}) \overset{\mathcal{AG}}{\longmapsto} \gamma$ $\hat{\mathbf{C}}$: see (22)
	$W_j^u = \int_0^\infty \mathrm{E}_u[Q_j(t)]\,dt]$	$\mathcal{B}_c(\mathbf{A}', \mathbf{b}')$ $\mathbf{A}'$, $\mathbf{b}'(\cdot)$: see (20), (21)	$(\mathbf{C}, \mathbf{A}') \overset{\mathcal{AG}}{\longmapsto} \gamma$

Table 2: Modelling Branching Bandit Problems.

[a] Extended contra-polymatroid base
[b] Extended polymatroid base

$$b'(S) = b(E) - b(S^c) + 2\sum_{i\in S} h_i \mathrm{E}[T_j^S], \tag{21}$$

where row vector $\mathbf{h} = (h_i)_i \in E$ is given by

$$\mathbf{h} = \mathbf{m}(\mathbf{I} - \mathrm{E}[\mathbf{N}])^{-1}\mathrm{Diag}((\frac{(E[v_i])^2 - Var[v_i]}{E[v_i]})_{i\in E})(\mathbf{I} - \mathrm{E}[\mathbf{N}]),$$

and Diag(x) denotes a diagonal matrix with diagonal entries corresponding to vector x.

Objective function:

$$\min_u V_u^{(\mathbf{0},\mathbf{C})} = \sum_{i\in E} \hat{C}_i\theta_i^u + 2\sum_{i\in E} C_i h_i = \sum_{i\in E} \frac{1}{\mathrm{E}[v_i]}\left(C_i - \sum_{j\in E} \mathrm{E}[N_{ij}]C_j\right)\theta_i^u + 2\sum_{i\in E} C_i h_i \tag{22}$$

In Table 2 we summarize the problems we considered, the performance measures used, the conservation laws, the corresponding performance space, as well as the input to algorithm $\mathcal{AG}$.

Given that branching bandits is a general problem formulation that encompasses a variety of problems, it should not be surprising that the theory developed encompasses a large collection of problems. In Table 3 below we illustrate the stochastic control problem from the literature, the objective criterion, where the original indexability result was obtained and where the performance space was characterized. For explicit derivations of the parameters of the extended polymatroids involved the reader is referred to Bertsimas and Niño-Mora [6].

2.4 Implications of the theory

In this section we summarize the implications of characterizing the performance space as an extended polymatroid as follows:

1. An independent algebraic proof of the indexability of the problem, which translates to a very efficient solution. In particular, Algorithm $\mathcal{AG}$ that computes the indices of indexable systems is as fast as the fastest known algorithm (Varaiya, Walrand and Buyukkoc [58]).

System	Criterion	Indexability	Performance Space
Batch of jobs	LC[a]	Smith [51]: D[b] Rothkopf [49]	Queyranne [43]: D, P[c] Bertsimas & Niño-Mora [6]: P
	DC[d]	Rothkopf [48]: D Gittins & Jones [21]	Bertsimas & Niño-Mora [6]: P
Batch of jobs with out-tree prec. constraints	LC	Horn [31]: D Meilijson & Weiss [40]	Bertsimas & Niño-Mora [6]: EP[e]
	DC	Glazebrook [24]	Bertsimas & Niño-Mora [6]: EP
Multiclass $M/G/1$	LC	Cox & Smith [16]	Coffman & Mitrani [15]: P Gelenbe & Mitrani [20]: P
	DC	Harrison [26, 27]	Bertsimas & Niño-Mora [6]: EP
Multiclass [f] $M/G/c$	LC	Federgruen & Groenevelt [18] Shanthikumar & Yao [50]	Federgruen & Groenevelt [19]: P Shanthikhumar & Yao [50]: P
Multiclass $G/M/c$	LC	Federgruen & Groenevelt [18] Shanthikumar & Yao [50]	Federgruen & Groenevelt [18]: P Shanthikumar & Yao [50]: P
Multiclass Jackson network [g]	LC	Ross & Yao [47]	Ross & Yao [47]: P
Multiclass $M/G/1$ with feedback	LC	Klimov [35]	Tsoucas [57]: EP
	DC	Tcha & Pliska [54]	Bertsimas & Niño-Mora [6]: EP
multiarmed bandits	DC	Gittins & Jones [21]	Bertsimas & Niño-Mora [6]: EP
Branching bandits	LC	Meilijson & Weiss [40]	Bertsimas & Niño-Mora [6]: EP
	DC	Weiss [60]	Bertsimas & Niño-Mora [6]: EP

Table 3: Indexable problems and their performance spaces.

[a] Linear cost
[b] Deterministic processing times
[c] Polymatroid
[d] Discounted linear reward-cost
[e] Extended polymatroid
[f] Same service time distribution for all classes
[g] Same service time distribution and routing probabilities for all classes (can be node dependent)

2. An understanding of whether or not the indices decompose. For example, in the classical multiarmed bandits problem Theorem 4 applies and therefore, the indices decompose, while in the general branching bandits formulation they do not.

3. A general and practical approach to sensitivity analysis of indexable systems, based on the well understood sensitivity analysis of linear programming (see Bertsimas and Niño-Mora [6] and Glazebrook [25]).

4. A new characterization of the indices of indexable systems as sums of dual variables corresponding to the extended polymatroid that characterizes the feasible space. This gives rise to new interpretations of the indices as prices or retirement options. For example, we can obtain a new interpretation of indices in the context of branching bandits as retirement options, thus generalizing the interpretation of Whittle [61] and Weber [59] for the indices of the classical multiarmed bandit problem.

5. The realization that the algorithm of Klimov for multiclass queues and the algorithm of Gittins for multiarmed bandits are examples of the same algorithm.

6. Closed form formulae for the performance of the optimal policy that can be used a) to prove structural properties of the problem (for example a result of Weber [59] that the objective function value of the classical multiarmed bandit problem is submodular) and b) to show that the indices for some stochastic control problems do not depend on some of the parameters of the problem.

Most importantly, this approach provides a unified treatment of several classical problems in stochastic control and is able to address in a unified way their variations such as: discounted versus undiscounted cost criterion, rewards versus taxes, preemption versus nonpreemption, discrete versus continuous time, work conserving versus idling policies, linear versus nonlinear objective functions.

3 Restless bandits

In this section we address the restless bandit problem defined as follows: There is a collection of N projects. Project $n \in \mathcal{N} = \{1, \ldots, N\}$ can be in one of a finite number of states $i_n \in E_n$, for $n = 1, \ldots, N$. At each instant of discrete time $t = 0, 1, 2, \ldots$, exactly $M < N$ projects must be operated. If project n, in state i_n, is in operation, then an *active* reward $R^1_{i_n}$ is earned, and the project state changes into j_n with an active transition probability $p^1_{i_n j_n}$. If the project remains idle, then a *passive* reward $R^0_{i_n}$ is received, and the project state changes into j_n with a passive transition probability $p^0_{i_n j_n}$. Rewards are discounted in time by a discount factor $0 < \beta < 1$. Projects are to be selected for operation according to an *admissible* scheduling policy u: the decision as to which M projects to operate at any time t must be based only on information on the current states of the projects. Let $\mathcal{U}$ denote the class of admissible scheduling policies. The goal is to find an admissible scheduling policy that maximizes the total expected discounted reward over an infinite horizon, i.e.,

$$Z^* = \max_{u \in \mathcal{U}} E_u \left[\sum_{t=0}^{\infty} \left(R^{a_1(t)}_{i_1(t)} + \cdots + R^{a_N(t)}_{i_N(t)} \right) \beta^t \right], \tag{23}$$

where $i_n(t)$ and $a_n(t)$ denote the state and the action (active or passive), respectively, corresponding to project n at time t. We assume that the initial state of project n is i_n with probability α_{i_n}, independently of all other projects.

The restless bandit problem was introduced by Whittle [62], as an extension of the multiarmed bandit problem studied in the previous section. The latter corresponds to the special case that exactly one project must be operated at any time (i.e., $M = 1$), and passive projects are frozen: they do not change state ($p^0_{i_n i_n} = 1$, $p^0_{i_n j_n} = 0$ for all $n \in N$ and $i_n \neq j_n$). Whittle mentions applications in the areas of clinical trials, aircraft surveillance and worker scheduling.

In this section we review the work of Bertsimas and Niño-Mora [7], who build upon the work of Whittle [62] and propose a series of N linear relaxations (the last being exact) of the achievable region,

using the theory of discounted Markov decision processes (MDP) (see Heyman and Sobel [29]). They also propose an indexing heuristic that arises from the first of these relaxations.

In order to formulate the restless bandit problem as a linear program we introduce the indicators

$$I^1_{i_n}(t) = \begin{cases} 1, & \text{if project } n \text{ is in state } i_n \text{ and active at time } t; \\ 0, & \text{otherwise,} \end{cases}$$

and

$$I^0_{i_n}(t) = \begin{cases} 1, & \text{if project } n \text{ is in state } i_n \text{ and passive at time } t; \\ 0, & \text{otherwise.} \end{cases}$$

Given an admissible scheduling policy $u \in \mathcal{U}$ we define performance measures

$$x^1_{i_n}(u) = E_u\left[\sum_{t=0}^{\infty} I^1_{i_n}(t)\beta^t\right], \text{ and } x^0_{i_n}(u) = E_u\left[\sum_{t=0}^{\infty} I^0_{i_n}(t)\beta^t\right].$$

Notice that performance measure $x^1_{i_n}(u)$ (resp. $x^0_{i_n}(u)$) represents the total expected discounted time that project n is in state i_n and active (resp. passive) under scheduling policy u. Let

$$P = \left\{ \mathbf{x} = \left(x^{a_n}_{i_n}(u)\right)_{i_n \in E_n, a_n \in \{0,1\}, n \in \mathcal{N}} \;\middle|\; u \in \mathcal{U} \right\},$$

the corresponding performance space. As the restless bandit problem can be viewed as a discounted MDP (in the state space $E_1 \times \cdots \times E_N$), the performance space P is a polyhedron. The restless bandit problem can thus be formulated as the linear program

$$Z^* = \max_{\mathbf{x} \in P} \sum_{n \in \mathcal{N}} \sum_{i_n \in E_n} \sum_{a_n \in \{0,1\}} R^{a_n}_{i_n} x^{a_n}_{i_n}. \tag{24}$$

For the special case of the multiarmed bandit problem P has been fully characterized as an extended contra-polymatroid. For the restless bandit problem we construct approximations of P that yield polynomial-size linear relaxations of the linear program (24).

3.1 A First-order Linear Programming Relaxation

The restless bandit problem induces a *first-order MDP* over each project n in a natural way: The state space of this MDP is E_n, its action space is $\mathcal{A}^1 = \{0, 1\}$, and the reward received when action a_n is taken in state i_n is $R^{a_n}_{i_n}$. Rewards are discounted in time by discount factor β. The transition probability from state i_n into state j_n, given action a_n, is $p^{a_n}_{i_n j_n}$. The initial state is i_n with probability α_{i_n}.

Let

$$Q^1_n = \left\{ \mathbf{x}_n = \left(x^{a_n}_{i_n}(u)\right)_{i_n \in E_n, a_n \in \mathcal{A}^1} \mid u \in \mathcal{U} \right\}.$$

From a polyhedral point of view, Q^1_n is the projection of the restless bandit polyhedron P over the space of the variables $x^{a_n}_{i_n}$ for project n. From a probabilistic point of view, Q^1_n is the performance space of the first-order MDP corresponding to project n. In order to see this, we observe that as policies u for the restless bandit problem range over all admissible policies $\mathcal{U}$, they induce policies u_n for the first-order

MDP corresponding to project n that range over all admissible policies for that MDP. From the theory of discounted MDPs we obtain

$$Q_n^1 = \left\{ \mathbf{x}_n \geq \mathbf{0} \mid x_{j_n}^0 + x_{j_n}^1 = \alpha_{j_n} + \beta \sum_{i_n \in E_n} \sum_{a_n \in \{0,1\}} p_{i_n j_n}^{a_n} x_{i_n}^{a_n}, \quad j_n \in E_n \right\}. \tag{25}$$

Whittle [62] proposes a relaxation of the problem by relaxing the requirement that exactly M projects must be active at any time and imposing that the total expected discounted number of active projects over the infinite horizon must be $M/(1-\beta)$. This condition on the discounted average number of active projects can be written as

$$\sum_{n \in \mathcal{N}} \sum_{i_n \in E_n} x_{i_n}^1(u) = \sum_{t=0}^{\infty} E_u \left[\sum_{n \in \mathcal{N}} \sum_{i_n \in E_n} I_{i_n}^1(t) \right] \beta^t = \sum_{t=0}^{\infty} M \beta^t = \frac{M}{1-\beta}.$$

Therefore, the first-order relaxation can be formulated as the linear program

$$\begin{aligned} Z^1 \quad = \quad & \max \sum_{n \in \mathcal{N}} \sum_{i_n \in E_n} \sum_{a_n \in \{0,1\}} R_{i_n}^{a_n} x_{i_n}^{a_n} \\ & \text{subject to} \\ & x_{j_n}^0 + x_{j_n}^1 = \alpha_{j_n} + \beta \sum_{i_n \in E_n} \sum_{a_n \in \{0,1\}} p_{i_n j_n}^{a_n} x_{i_n}^{a_n}, \quad j_n \in E_n, \qquad n \in \mathcal{N}, \\ & \sum_{n \in \mathcal{N}} \sum_{i_n \in E_n} x_{i_n}^1 = \frac{M}{1-\beta}. \end{aligned} \tag{26}$$

We will refer to the feasible space of linear program (26) as the first-order approximation of the restless bandit polyhedron, and will denote it as P^1. Notice that linear program (26) has $O(N|E_{\max}|)$ variables and constraints, where $|E_{\max}| = \max_{n \in \mathcal{N}} |E_n|$.

3.2 A kth-order Linear Programming Relaxation

We can strengthen the relaxation by introducing new decision variables corresponding to performance measures associated with kth-order project interactions. For each k-tuple of projects $1 \leq n_1 < \cdots < n_k \leq N$, the admissible actions that can be taken at a corresponding k-tuple of states $(i_1, \ldots, i_k)$ range over

$$\mathcal{A}^k = \left\{ (a_1, \ldots, a_k) \in \{0,1\}^k \mid a_1 + \cdots + a_k \leq M \right\}.$$

Given an admissible scheduling policy u for the restless bandit problem, we define kth-order performance measures, for each k-tuple $1 \leq n_1 < \cdots < n_k \leq N$ of projects, by

$$x_{j_1 \ldots j_k}^{a_1 \ldots a_k}(u) = E_u \left[\sum_{t=0}^{\infty} I_{j_1}^{a_1}(t) \cdots I_{j_k}^{a_k}(t) \beta^t \right], \quad j_1 \in E_{n_1}, \ldots, j_k \in E_{n_k}. \tag{27}$$

The restless bandit problem induces a kth-order MDP over each k-tuple of projects $n_1 < \cdots < n_k$ as follows: The state space of the MDP is $E_{n_1} \times \cdots \times E_{n_k}$, the action space is $\mathcal{A}^k$, and the reward corresponding to state $(i_{n_1}, \ldots, i_{n_k})$ and action $(a_{n_1}, \ldots, a_{n_k})$ is $R_{i_{n_1}}^{a_{n_1}} + \cdots + R_{i_{n_k}}^{a_{n_k}}$. Rewards are discounted in time

by discount factor β. The transition probability from state $(i_{n_1},\ldots,i_{n_k})$ into state $(j_{n_1},\ldots,j_{n_k})$, given action $(a_{n_1},\ldots,a_{n_k})$, is $p_{i_{n_1}j_{n_1}}^{a_{n_1}}\cdots p_{i_{n_k}j_{n_k}}^{a_{n_k}}$. The initial state is $(i_{n_1},\ldots,i_{n_k})$ with probability $\alpha_{i_{n_1}}\cdots\alpha_{i_{n_k}}$. As in the first order relaxation all performance vectors in the k-th order relaxation satisfy

$$\sum_{(a_1,\ldots,a_k)\in\mathcal{A}^k} x_{j_1\ldots j_k}^{a_1\ldots a_k} = \alpha_{j_1}\cdots\alpha_{j_k} + \beta \sum_{\substack{i_1\in E_{n_1},\ldots,i_k\in E_{n_k}\\(a_1,\ldots,a_k)\in\mathcal{A}^k}} p_{i_1j_1}^{a_1}\cdots p_{i_kj_k}^{a_k}\, x_{i_1\ldots i_k}^{a_1\ldots a_k}, \quad j_1\in E_{n_1},\ldots,j_k\in E_{n_k}, \tag{28}$$

If u is an admissible scheduling policy, then

$$\sum_{1\le n_1<\cdots<n_k\le N}\ \sum_{i_1\in E_{n_1},\ldots,i_k\in E_{n_k}}\ \sum_{\substack{(a_1,\ldots,a_k)\in\mathcal{A}^k:\\ a_1+\cdots+a_k=r}} x_{i_1\ldots i_k}^{a_1\ldots a_k}(u) = \frac{\binom{M}{r}\binom{N-M}{k-r}}{1-\beta}, \tag{29}$$

for

$$\max(0,k-(N-M)) \le r \le \min(k,M). \tag{30}$$

Conservation law (29) follows since at each time the number of k-tuples of projects that contain exactly r active projects is $\binom{M}{r}\binom{N-M}{k-r}$, for r in the range given by (30).

In terms of the original performance vectors

$$x_i^a(u) = \sum_{\substack{(a_1,\ldots,a_k)\in\mathcal{A}^k\\ i_1\in E_{n_1},\ldots,i_k\in E_{n_k}:\\ i_r=i,a_r=a}} x_{i_1\ldots i_k}^{a_1\ldots a_k}(u). \tag{31}$$

We now define the kth-order relaxation of the restless bandit problem as the linear program

$$Z^k = \max \sum_{n\in\mathcal{N}}\sum_{i_n\in E_n}\sum_{a_n\in\{0,1\}} R_{i_n}^{a_n} x_{i_n}^{a_n}$$

subject to (28), (29), (31) and nonnegativity constraints on the variables. We define the kth-order approximation to the restless bandit polyhedron P as the projection of the feasible space of the above linear program into the space of the first-order variables, x_i^a, and denote it as P^k. It is easy to see that the sequence of approximations is monotone, in the sense that

$$P^1 \supseteq P^2 \supseteq \cdots \supseteq P^N = P.$$

Notice that the kth-order relaxation has $O(N^k|E_{\max}|^k)$ variables and constraints, for k fixed. Therefore, the kth-order relaxation has polynomial size, for k fixed.

The last relaxation of the sequence is exact, i.e., $Z^N = Z^*$, since it corresponds to the linear programming formulation of the restless bandit problem modeled as a MDP in the standard way.

3.3 A Primal-dual Heuristic for the Restless Bandit Problem

The dual of the linear program (26) is

$$Z^1 = \min \sum_{n\in\mathcal{N}} \sum_{j_n\in E_n} \alpha_{j_n}\lambda_{j_n} + \frac{M}{1-\beta}\lambda \tag{32}$$

subject to

$$\lambda_{i_n} - \beta \sum_{j_n\in E_n} p^0_{i_n j_n}\lambda_{j_n} \geq R^0_{i_n}, \quad i_n \in E_n, \quad n \in \mathcal{N},$$

$$\lambda_{i_n} - \beta \sum_{j_n\in E_n} p^1_{i_n j_n}\lambda_{j_n} + \lambda \geq R^1_{i_n}, \quad i_n \in E_n, \quad n \in \mathcal{N},$$

Let $\{\overline{x}^{a_n}_{i_n}\}$, $\{\overline{\lambda}_{i_n}, \overline{\lambda}\}$, $i_n \in E_n$, $n \in \mathcal{N}$ be an optimal primal and dual solution to the first-order relaxation (26) and its dual. Let $\{\overline{\gamma}^{a_n}_{i_n}\}$ be the corresponding optimal reduced cost coefficients, i.e.,

$$\overline{\gamma}^0_{i_n} = \overline{\lambda}_{i_n} - \beta \sum_{j_n\in E_n} p^0_{i_n j_n}\overline{\lambda}_{j_n} - R^0_{i_n},$$

$$\overline{\gamma}^1_{i_n} = \overline{\lambda}_{i_n} - \beta \sum_{j_n\in E_n} p^1_{i_n j_n}\overline{\lambda}_{j_n} + \overline{\lambda} - R^1_{i_n},$$

which are nonnegative. It is well known (see Murty [41], pp. 64-65), that the optimal reduced costs have the following interpretation:

$\overline{\gamma}^1_{i_n}$ is the *rate of decrease* in the objective-value of linear program (26) *per unit increase* in the value of the variable $x^1_{i_n}$.

$\overline{\gamma}^0_{i_n}$ is the rate of decrease in the objective-value of linear program (26) per unit increase in the value of the variable $x^0_{i_n}$.

The proposed heuristic takes as input the vector of current states of the projects, $(i_1, \ldots, i_N)$, an optimal primal solution to (26), $\{\overline{x}^{a_n}_{j_n}\}$, and the corresponding optimal reduced costs, $\{\overline{\gamma}^{a_n}_{j_n}\}$, and produces as output a vector with the actions to take on each project, $(a^*(i_1), \ldots, a^*(i_N))$. An informal description of the heuristic, with the motivation that inspired it, is as follows:

The heuristic is structured in a primal and a dual stage. In the primal stage, projects n whose corresponding active primal variable $\overline{x}^1_{i_n}$ is strictly positive are considered as candidates for active selection. The intuition is that we give preference for active selection to projects with positive $\overline{x}^1_{i_n}$ with respect to those with $\overline{x}^1_{i_n} = 0$, which seems natural given the interpretation of performance measure $x^1_{i_n}(\cdot)$ as the total expected discounted time spent selecting project n in state i_n as active. Let p represent the number of such projects. In the case that $p = M$, then all p candidate projects are set active and the heuristic stops. If $p < M$, then all p candidate projects are set active and the heuristic proceeds to the dual stage that selects the remaining $M - p$ projects. If $p > M$ none of them is set active at this stage and the heuristic proceeds to the dual stage that finalizes the selection.

In the dual stage, in the case that $p < M$, then $M - p$ additional projects, each with current active primal variable zero ($\overline{x}^1_{i_n} = 0$), must be selected for active operation among the $N - p$ projects, whose

actions have not yet been fixed. As a heuristic index of the undesirability of setting project n in state i_n active, we take the active reduced cost $\overline{\gamma}^1_{i_n}$. This choice is motivated by the interpretation of $\overline{\gamma}^1_{i_n}$ stated above: the larger the *active index* $\gamma^1_{i_n}$ is, the larger is the rate of decrease of the objective-value of (26) per unit increase in the active variable $x^1_{i_n}$. Therefore, in the heuristic we select for active operation the $M-p$ additional projects with smallest active reduced costs.

In the case that $p > M$, then M projects must be selected for active operation, among the p projects with $\overline{x}^1_{i_n} > 0$. Recall that by complementary slackness, $\overline{\gamma}^1_{i_n} = 0$ if $\overline{x}^1_{i_n} > 0$. As a heuristic index of the desirability of setting project n in state i_n active we take the passive reduced cost $\overline{\gamma}^0_{i_n}$. The motivation is given by the interpretation of $\overline{\gamma}^0_{i_n}$ stated above: the larger the *passive index* $\gamma^0_{i_n}$ is, the larger is the rate of decrease in the objective-value of (26) per unit increase in the value of the passive variable $x^0_{i_n}$. Therefore, in the heuristic we select for active operation the M projects with largest passive reduced costs.

Assuming that the optimal solution of (26) is nondegenerate we can interpret the primal-dual heuristic as an index heuristic as follows:

1. Given the current states $(i_1, \ldots, i_N)$ of the N projects, compute the indices

$$\delta_{i_n} = \overline{\gamma}^1_{i_n} - \overline{\gamma}^0_{i_n}.$$

2. Set active the projects that have the M smallest indices. In case of ties, set active projects with $\overline{x}^1_{i_n} > 0$.

In contrast with the Gittins indices for usual bandits, notice that the indices δ_{i_n} for a particular project depend on characteristics of all other projects. Bertsimas and Niño-Mora [7] report computational results that show that the primal-heuristic and the bound given by the second order relaxation are very close, proving that both the relaxation and the heuristic are very effective.

4 Polling systems

Polling systems, in which a single server in a multiclass queueing system serves several classes of customers incurring changeover times when he serves different classes, have important applications in computer, communication, production and transportation networks. In these application areas several users compete for access to a common resource (a central computer in a time sharing computer system, a transmission channel in a communication system, a machine in a manufacturing context or a vehicle in transportation applications). As a result, the problem has attracted the attention of researchers across very different disciplines. The name polling systems comes primarily from the communication literature. Motivated by its important applications, polling systems have a rather large literature, which for the most part addresses the performance of specific policies rather than the optimal design of the polling

system. For an extensive discussion of the research work on polling systems, we refer to the survey papers by Levy and Sidi [37] and Takagi [52], [53].

Consider a system consisting of N infinite capacity stations (queues), and a single server which serves them one at a time. The arrival process to station i ($i = 1, 2, \ldots, N$) is assumed to be a Poisson process with rate λ_i. The overall arrival rate to the system is $\lambda = \sum_{i=1}^{N} \lambda_i$. Customers arriving to station i will be referred to as class-i customers and have a random service requirement X_i with finite mean x_i and second moment $x_i^{(2)}$. The actual service requirement of a specific customer is assumed to be independent of other system variables. The cost of waiting for class-i customers is c_i per unit time. There are changeover time d_{ij} whenever the server changes from serving class-i customers to class-j customers. The offered traffic load at station i is equal to $\rho_i = \lambda_i x_i$, and the total traffic load is equal to $\rho = \sum_{i=1}^{N} \rho_i$.

The natural performance measure in polling systems is the mean delay time between the request for service from a customer and the delivery of the service by the server to that customer. The optimization problem in polling systems is to decide which customer should be in service at any given time in order to minimize the weighted expected delay of all the classes. Let $\mathcal{U}$ be the class of nonpreemptive, non-anticipative and stable policies. Within $\mathcal{U}$ we further distinguish between static ($\mathcal{U}_{static}$) and dynamic ($\mathcal{U}_{dynamic}$) policies. At each decision epoch static policies do not take into account information about the state of stations in the system other than the one occupied by the server and are determined a priori or randomly. In particular the server's behavior when in station i is independent of the state of the other stations in the system (i.e., the queue lengths and the interarrival times of the customers). Examples of static policies include randomized policies, in which the next station to be visited by the server is determined by an a priori probability distribution, and routing table policies, in which the next station to be visited is predetermined by a routing table. A special case of the routing table policies is the cyclic policy, where the stations are visited by the server in a cyclic order.

Dynamic policies take into account information about the current state of the network. For example, a threshold policy or a policy that visits the most loaded station, is a dynamic policy, because the decision on which customer to serve next by the server depends on the current queue lengths at various stations in the system. In certain applications it might be impractical or even impossible to use a dynamic policy. For example in a transportation network, the vehicle might not know the overall state of the network. As a result, although static policies are not optimal, they can often be the only realistic policy. Moreover, when there are no changeover times, the policy that minimizes the mean weighted delay is an indexing policy (the $c\mu$ rule), a static policy.

While the literature on performance analysis of polling systems is huge there are very few papers addressing optimization. Hofri and Ross [30] and Reiman and Wein [46] address the optimization problem over dynamic policies for polling systems with two stations. Boxma et. al. [13] propose heuristic polling table strategies. In this section we review the work in Bertsimas and Xu [10] in which a nonlinear (but

convex) optimization problem is proposed, whose solution provides a lower bound on an arbitrary static policy. Using information from the lower bounds and *integer programming* techniques, static policies (routing table policies) are constructed that are very close (within 0-3%) to the lower bound.

4.1 Lower bounds on achievable performance

We develop in this section lower bounds on the weighted mean waiting time for polling systems for nonpreemptive, nonanticipative and stable policies. We call these policies admissible. We present bounds for both static and dynamic policies.

Let $E[W_i]$ be the average waiting time of class-i customers. The goal is to find a static policy $u \in \mathcal{U}_{static}$ to minimize the weighted mean delay $E[W]$ for the polling system:

$$\min_{u \in \mathcal{U}_{static}} E[W] = \frac{1}{\lambda} \sum_{i=1}^{N} c_i \lambda_i E[W_i]. \tag{33}$$

Within a particular class, we assume that the server uses a First-In-First-Out (FIFO) discipline. We denote with d_{ij} the changeover time when the server changes from serving class-i customers to class-j customers $(i, j = 1, 2, \ldots, N)$.

Theorem 5 *The optimal weighted mean delay in a polling system under any static and admissible policy is bounded from below by Z_{static}, the solution of the following convex programming problem:*

$$Z_{static} = \quad \min \ \frac{1}{2\lambda} \sum_{i=1}^{N} \frac{c_i \lambda_i^2 x_i^{(2)}}{1-\rho_i} + \frac{1}{2\lambda} \sum_{i=1}^{N} \frac{c_i \lambda_i (1-\rho_i)}{(\sum_{j=1}^{N} m_{ji})} \tag{34}$$

$$\text{subject to} \quad \sum_{j=1}^{N} m_{ij} - \sum_{k=1}^{N} m_{ki} = 0, \qquad i = 1, 2, \ldots, N \tag{35}$$

$$\sum_{i,j=1}^{N} d_{ij} m_{ij} \leq 1 - \rho \tag{36}$$

$$m_{ii} = 0, \ m_{ij} \geq 0 \qquad \forall \ i, j,$$

The m_{ij} in the above convex optimization problem represent the steady state average number of visits from station i to station j per unit time. Constraint (35) represents conservation of flow, while (36) is the stability condition. These constraints are still valid even under dynamic policies. The fact that the objective function in (34) represents a lower bound for the optimal solution value, only holds for static policies. The idea of the proof is that under a static policy, the expected waiting time of class i customers decomposes to

$$E[W_i] = \frac{\lambda_i x_i^{(2)}}{2(1-\rho_i)} + \frac{v_i^{(2)}}{2v_i},$$

where v_i, $v_i^{(2)}$ are the first and second moments of the vacation times observed at station i. Using the inequality $v_i^{(2)} \geq v_i^2$ and expressing the expected vacation times v_i in terms of m_{ij} we obtain the bound. For details see Bertsimas and Xu [10]. Notice that while the feasible space of (34) is a network flow problem with a side constraint, the objective function is a nonlinear, but convex function.

In order to acquire further insight on the bound of the previous theorem, the flow conservation constraints (35) are relaxed to obtain a closed form formula for the lower bounds.

Theorem 6 *a) The weighted mean delay in a polling system under any static and admissible policy is bounded from below by:*

$$Z_{closed} = \frac{1}{2\lambda}\sum_{i=1}^{N}\frac{c_i\lambda_i^2 x_i^{(2)}}{1-\rho_i} + \frac{(\sum_{i=1}^{N}\sqrt{c_i\lambda_i(1-\rho_i)d_i^*})^2}{2\lambda(1-\rho)}, \tag{37}$$

where $d_i^* = d_{j(i),i} = \min_j\{d_{ji}\}$.

b) For a homogeneous polling system ($c_i = 1,\ x_i = x$, *for all* $i = 1, 2, \ldots, N$), *under static and admissible policies, the weighted mean delay is bounded from below by:*

$$\frac{\lambda x^2}{2(1-\rho)} + \frac{(\sum_{i=1}^{N}\sqrt{\lambda_i(1-\rho_i)d_i^*})^2}{2\lambda(1-\rho)}. \tag{38}$$

Notice that the bound from (38) is stronger than (37) for the special case of a homogeneous system. For arbitrary policies including dynamic ones we can prove the following.

Theorem 7 *The optimal weighted mean delay in a polling system under any admissible policy is bounded from below by*

$$Z_{dynamic} = \max\left\{\frac{1}{\lambda}\left[\sum_{i=1}^{N}\frac{\lambda_i x_i^{(2)}}{2}\right]\left[\sum_{i=1}^{N}\frac{c_i\lambda_i}{(1-\sigma_{i-1})(1-\sigma_i)}\right],\right.$$
$$\left.\frac{1}{1-\rho}\min_{j\in N}\{\sum_{i=1}^{N}\frac{c_i\lambda_i}{\lambda}d_{ji}\} + \frac{\sum_{i=1}^{N}c_i\lambda_i}{\lambda}\frac{\sum_{j=1}^{N}\lambda_j x_j^{(2)}}{2(1-\rho)}\right\},$$

where $\sigma_i = \sum_{j=1}^{i}\rho_j$ *and* $\rho = \sum_{j=1}^{N}\rho_j$ *is the system utilization.*

The first term in the lower bound represents the optimal weighted mean delay with no changeover costs (the optimal policy corresponds to the $c\mu$ rule), while the second term incorporates changeover times in the lower bound.

4.2 Design of effective static policies

The goal of this subsection is to provide a technique to construct near optimal policies using integer programming. As already mentioned the m_{ij} are interpreted as the steady state average number of visits from station i to station j per unit time. Let $e_{ij} = m_{ij}/\sum_{k,l} m_{kl}$ and $\mathbf{E} = [e_{ij}]$ be the corresponding changeover ratio matrix. Intuitively, in order for the performance of a policy u to be close to the lower bound, it is desirable that the proportion of changeovers from station i to station j under the policy u is close to e_{ij}. We refer to this requirement as the *closeness* condition. We consider two classes of policies that satisfy the closeness condition approximately.

Randomized policies:

Under this class of policies the server after serving exhaustively all customers at station i moves to station j with probability p_{ij}. Kleinrock and Levy [34] consider randomized policies, in which the next station visited will be station j with probability p_j, independent of the present station.

Given the values of m_{ij} from the lower bound calculation, we would like to choose the probabilities p_{ij} so that the closeness condition is satisfied. An obvious choice is to pick $p_{ij} = e_{ij}/\sum_{k=1}^{N} e_{ik}$. $\mathbf{P} = [p_{ij}]$ is the corresponding changeover probability matrix. We note, however, that this choice of p_{ij} does not necessarily represent the optimal randomized policy.

Routing table policies

Under this class of policies the server visits stations in an a priori periodic sequence. For example the server visits a three station system using the cyclic sequence (1,2,3,1,2,3,...) or the sequence (1,2,1,2,3,1,2,1,2,3,...), i.e., stations 1 and 2 are visited twice as often as station 3. Boxma et. al. [13] use heuristic rules to construct routing table policies.

We use integer programming methods to construct routing tables that satisfy the closeness condition. Let h_{ij} be the number of changeovers from station i to station j in an optimal routing table. $\mathbf{H} = [h_{ij}]$ is the changeover matrix. Note that unlike m_{ij}, h_{ij} should be integers. Notice that $\sum_{i,j} h_{ij}$ is the length of the routing table, i.e., the total number of changeovers in the periodic sequence. Moreover, $e_{ij}\sum_{k,l} h_{kl}$ is the desired number of changeovers from station i to station j in the routing table under the closeness condition. In order to satisfy the closeness condition, a possible objective in selecting a routing table is to minimize the maximum difference between the number of changeovers h_{ij} from station i to station j in the optimal routing table and the desired number of changeovers determined by $e_{ij}\sum_{k,l} h_{kl}$. i.e.,

$$\min_{h}\{\max_{i,j}\{|h_{ij} - e_{ij}\sum_{k,l} h_{kl}|\}\}. \tag{39}$$

In addition, the flow conservation at each station requires that

$$\sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, \qquad i = 1, 2, \ldots, N, \tag{40}$$

i.e., the number of visits by the server to station i should equal to the number of visits by the server from station i to other stations. The h_{ij}'s should also form an Eulerian tour. Let I be the set of all stations and G be the subset of stations in the network. Since the Eulerian tour should be connected, we require that for all subsets G of the stations

$$\sum_{i\in G,\ j\in\bar{G}} h_{ij} \geq 1, \quad \forall\, G \subset I,\ G \neq \phi. \tag{41}$$

In summary, the problem becomes

$$\begin{array}{lll} (P_{Eulerian}) & \min_h\{\max_{i,j}\{|h_{ij} - e_{ij}\sum_{k,l} h_{kl}|\}\} & \\ \text{subject to :} & \sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, & i = 1, 2, \ldots, N \\ & \sum_{i\in G,\ j\in\bar{G}} h_{ij} \geq 1, & \forall\, G \subset I,\ G \neq \phi \\ & h_{ij} \geq 0,\ \textit{integer}, & i, j = 1, 2, \ldots, N \end{array} \tag{42}$$

Equation (42) can be easily converted to a pure integer programming problem. Since our goal is only to obtain an approximate solution, we approximate the problem by relaxing the connectivity constraints in

equation (41). But if equation (41) is relaxed, $h_{ij} = 0,\ \forall\ i, j$ will be a feasible solution and will minimize the objective function in (42). In order to exclude this infeasible solution to (42), we impose a lower limit on the length of the routing table. Since each of the stations should be visited at least once in any feasible routing table, the length of any routing table should be at least N. Moreover, we place an upper bound L_{max} on the length of the routing table to make the integer programming solvable:

$$\begin{array}{lll} (P_{approx}) & \min_h\{\max_{i,j}\{|h_{ij} - e_{ij}\sum_{k,l} h_{kl}|\}\} & \\ \text{subject to} & \sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, & i = 1, 2, \ldots, N \\ & N \leq \sum_{i,j} h_{ij} \leq L_{max} & \\ & h_{ij} \geq 0,\ integer, & i, j = 1, 2, \ldots, N \end{array}$$

The previous formulation can be reformulated as a pure integer programming problem as follows:

$$\begin{array}{lll} (P_{approx}) & \min\ y & \\ \text{subject to} & y - h_{ij} + e_{ij}\sum_{k,l} h_{kl} \geq 0, & i, j = 1, 2, \ldots, N \\ & z + h_{ij} - e_{ij}\sum_{k,l} h_{kl} \geq 0, & i, j = 1, 2, \ldots, N \\ & \sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, & i = 1, 2, \ldots, N \\ & N \leq \sum_{i,j} h_{ij} \leq L_{max} & \\ & h_{ij} \geq 0,\ integer, & i, j = 1, 2, \ldots, N \end{array} \tag{43}$$

Note that there are many feasible routing tables that will be consistent with the h_{ij}'s obtained from the solution of (P_{approx}). We will select a Eulerian tour that spaces the visits to the stations as equally as possible. Although it is possible to formulate this requirement precisely as another integer programming problem, we found numerically that the added effort is not justified from the results it produces.

4.3 Performance of proposed policies

Based on extensive simulation results, Bertsimas and Xu [10] arrive at the following conclusions:

1. The routing policies constructed outperform all other static routing policies reported in the literature and they are at most within 3% from the lower bounds for the cases studied. Moreover, randomized policies are outperformed by routing table policies.

2. The performance of the routing table policy improves compared against other static routing policies as the change-over times or the system utilization increases.

3. For lower change-over times and system utilizations dynamic policies outperform static policies by a significant margin. But as the change-over times or system utilization increase, static policies are equally or more effective. This is a rather interesting fact, since at least in the cases that optimal dynamic policies are known (two stations), they are rather complicated threshold policies (Hofri and Ross [30]).

4. Based on the intuition from the proof of the static lower bound and the numerical results, Bertsimas and Xu [10] conjecture that the static lower bounds developed in this paper are valid for dynamic policies also under heavy traffic conditions or for large changeover costs.

These results suggest that as far as static policies are concerned the routing table policies constructed through integer programming are adequate for practical problems.

5 Multiclass queueing networks

A *multiclass queueing network* is one that services multiple types of customers which may differ in their arrival processes, service requirements, routes through the network as well as costs per unit of waiting time. In open networks we are interested to determine an optimal policy for sequencing and routing customers in the network that minimizes a linear combination of the expected sojourn times of each customer class, while in closed networks we are interested in i the maximization of throughput. There are both *sequencing* and *routing* decisions involved in these optimization problems. A *sequencing policy* determines which type of customer to serve at each station of the network, while a *routing policy* determines the route of each customer. There are several important applications of such problems: packet-switching communication networks with different types of packets and priorities, job shop manufacturing systems, scheduling of multi-processors and multi-programmed computer systems, to name a few.

In this section we present a systematic way introduced in Bertsimas, Paschalidis and Tsitsiklis [8] based on a potential function to generate constraints (linear and nonlinear) that all points in the achievable space of a stochastic optimization problem should satisfy.

We consider initially an open multiclass queueing network involving only sequencing decisions (routing is given) with N single server stations (nodes) and R different job classes. The class of a job summarizes all relevant information on the current characteristics of a job, including the node at which it is waiting for service. In particular, jobs waiting at different nodes are by necessity of different classes and a job changes class whenever it moves from one.node to another. Let $\sigma(r)$ be the node associated with class r and let C_i be the set of all classes r such that $\sigma(r) = i$. When a job of class r completes service at node i, it can become a job of class s, with probability p_{rs}, and move to server $\sigma(s)$; it can also exit the network, with probability $p_{r0} = 1 - \sum_{s=1}^{R} p_{rs}$. There are R independent Poisson arrival streams, one for each customer class. The arrival process for class r customers has rate λ_{0r} and these customers join station $\sigma(r)$. The service time of class r jobs is assumed to be exponentially distributed with rate μ_r. Note that jobs of different classes associated with the same node can have different service requirements. We assume that service times are independent of each other and independent of the arrival process.

Whenever there is one or more customers waiting for service at a node, we can choose which, if any, of these customers should be served next. (Notice, that we are not restricting ourselves to work-conserving

policies.) In addition, we allow for the possibility of preemption. A rule for making such decisions is called a *policy*. Let $n_r(t)$ be the number of class r customers present in the network at time t. The vector $\mathbf{n}(t) = (n_1(t), \ldots, n_R(t))$ will be called the *state* of the system at time t. A policy is called *Markovian* if each decision it makes is determined as a function of the current state. It is then clear that under a Markovian policy, the queueing network under study evolves as a continuous-time Markov chain.

For technical reasons, we will only study Markovian policies satisfying the following assumption:
Assumption A: a) The Markov chain $\mathbf{n}(t)$ has a unique invariant distribution.
b) For every class r, we have $E[n_r^2(t)] < \infty$, where the expectation is taken with respect to the invariant distribution.

Let n_r be the steady-state mean of $n_r(t)$, and x_r be the mean response time (waiting plus service time) of class r customers. We are interested in determining a scheduling policy that minimizes a linear cost function of the form $\sum_{r=1}^{R} c_r x_r$. As before we approach this problem by trying to determine the set X of all vectors $(x_1, \ldots, x_R)$ that are obtained by considering different policies that satisfy Assumption A. By minimizing the cost function $\sum_{r=1}^{R} c_r x_r$ over the set X, we can then obtain the cost of an optimal policy.

The set X is not necessarily convex and this leads us to considering its convex hull X'. Any vector $\mathbf{x} \in X'$ is the performance vector associated with a, generally non-Markovian, policy obtained by "time sharing" or randomization of finitely many Markovian policies. Note also that if the minimum over X' of a linear function is attained, then it is attained at some element of X. We will refer to X' as the *region of achievable performance.*

5.1 Sequencing of Multiclass Open Networks: Approximate Polyhedral Characterization

The traffic equations for our network model take the form

$$\lambda_r = \lambda_{0r} + \sum_{r'=1}^{R} \lambda_{r'} p_{r'r}, \qquad r = 1, \ldots, R. \tag{44}$$

We assume that the inequality

$$\sum_{r \in C_i} \frac{\lambda_r}{\mu_r} < 1$$

holds for every node i. This ensures that there exists at least one policy under which the network is stable.

Let us consider a set S of classes. We consider a potential function of the form $(R^S(t))^2$ where

$$R^S(t) = \sum_{r \in S} f_S(r) n_r(t), \tag{45}$$

and where $f_S(r)$ are constants to be referred to as *f-parameters*. For any set S of classes, we will use a different set of *f-parameters*, but in order to avoid overburdening our notation, the dependence on S will not be shown explicitly.

We will impose the following condition on the *f-parameters.* Although it may appear unmotivated at this point, the proof of Theorem 8 suggests that this condition leads to tighter bounds. We assume that for each S we have:

For any node i, the value of the expression

$$\mu_r \left[\sum_{r' \in S} p_{rr'}(f(r) - f(r')) + \sum_{r' \notin S} p_{rr'} f(r) \right] \tag{46}$$

is nonnegative and the same for all $r \in C_i \cap S$, and will be denoted by f_i. If $C_i \cap S$ is empty, we define f_i to be equal to zero.

Bertsimas, Paschalidis and Tsitsiklis [8] prove the following theorem. We present its proof, since it is instructive of how potential function arguments can be used in general to construct polyhedral approximations of the achievable region in stochastic control problems.

Theorem 8 *For any set S of classes, for any choice of the* f-parameters *satisfying the restriction (46), and for any policy satisfying Assumption A, the following inequality holds:*

$$\sum_{r \in S} \lambda_r f(r) x_r \geq \frac{N'(S)}{D'(S)} \tag{47}$$

where :

$$\begin{aligned} N'(S) \;=\; & \sum_{r \in S} \lambda_{0r} f^2(r) + \sum_{r \notin S} \lambda_r \sum_{r' \in S} p_{rr'} f^2(r') + \\ & \sum_{r \in S} \lambda_r \left[\sum_{r' \in S} p_{rr'} (f(r) - f(r'))^2 + \sum_{r' \notin S} p_{rr'} f^2(r) \right] \end{aligned}$$

$$D'(S) = 2 \left[\sum_{i=1}^{N} f_i - \sum_{r \in S} \lambda_{0r} f(r) \right]$$

S being a subset of the set of classes and x_r the mean response time of class r.

Proof We first uniformize the Markov chain so that the transition rate at every state is equal to

$$\nu = \sum_r \lambda_{0r} + \sum_r \mu_r$$

The idea is to pretend that every class is being served with rate μ_r, but a service completion is a fictitious one unless a customer of class r is being served in actuality. Without loss of generality we scale time so that $\nu = 1$. Let τ_k be the sequence of transition times for the uniformized chain. Let $B_r(t)$ be the event that node $\sigma(r)$ is busy with a class r customer at time t. Let $\overline{B}_r(t)$ be its complement. Let $1\{.\}$ the indicator function. We assume that the process $\mathbf{n}(t)$ is right-continuous.

We have the following recursion for $R(\tau_k)$

$$
\begin{aligned}
&E[R^2(\tau_{k+1}) \mid \mathbf{n}(\tau_k)] = \\
&\quad \sum_{r\in S} \lambda_{0r}(R(\tau_k)+f(r))^2 + \sum_{r\notin S} \lambda_{0r} R^2(\tau_k) + \\
&\quad \sum_{r\in S} \mu_r 1\{B_r(\tau_k)\} \left[\sum_{r'\in S} p_{rr'}(R(\tau_k)-f(r)+f(r'))^2 + \sum_{r'\notin S} p_{rr'}(R(\tau_k)-f(r))^2 \right] + \\
&\quad \sum_{r\in S} \mu_r 1\{\overline{B}_r(\tau_k)\} R^2(\tau_k) + \\
&\quad \sum_{r\notin S} \mu_r 1\{B_r(\tau_k)\} \left[\sum_{r'\in S} p_{rr'}(R(\tau_k)+f(r'))^2 + \sum_{r'\notin S} p_{rr'} R^2(\tau_k) \right] + \\
&\quad \sum_{r\notin S} \mu_r 1\{\overline{B}_r(\tau_k)\} R^2(\tau_k)
\end{aligned}
$$

In the above equation, we use the convention that the set of classes $r' \notin S$ also contains the case $r' = 0$, which corresponds to the external world of the network. (Recall that p_{r0} is the probability that a class r customer exits the network after completion of service.) We now use the assumption that the *f-parameters* satisfy (46). Then, the term

$$
2\sum_{r\in S} \mu_r 1\{B_r(\tau_k)\} \left[\sum_{r'\in S} p_{rr'} R(\tau_k)(f(r)-f(r')) + \sum_{r'\notin S} p_{rr'} R(\tau_k) f(r) \right]
$$

can be written as

$$
\sum_{i=1}^{N} f_i R(\tau_k) 1\{\text{server } i \text{ busy from some class } r \in S \cap C_i \text{ at } \tau_k\}.
$$

(Recall that we defined $f_i = 0$ for those stations i having $C_i \cap S$ empty.) To bound the above term, we use the fact that the indicator is at most 1. It should now be apparent why we selected *f-parameters* satisfying (46). By doing so, we were able to aggregate certain indicator functions before bounding them by 1.

In addition, to bound the term

$$
\sum_{r\notin S} 2\mu_r 1\{B_r(\tau_k)\} \sum_{r'\in S} p_{rr'} R(\tau_k) f(r')
$$

we use the inequality $1\{B_r(\tau_k)\} \geq 0$.

We apply all of these bounds to our recursion for $R(\tau_k)$. We then take expectations of both sides with respect to the invariant distribution (these expectations are finite due to Assumption A) and we can replace $E[R(\tau_k)]$ by $E[R(t)]$. After some elementary algebra and rearrangements, using (46) and the relation (valid in steady-state) $E[1\{B_r(\tau_k)\}] = \lambda_r/\mu_r$, we finally obtain (47). □

Remarks : In order to apply Theorem 8, we must choose some *f-parameters* that satisfy (46). We do not know whether there always exists a choice of the *f-parameters* that provides dominant bounds.

But, even if this were the case, it would probably be difficult to determine these "best" *f-parameters*. Later in this section, we show that finding the best *f-parameters* is not so important because there is a nonparametric variation of this bounding method that yields tighter bounds. The situation is analogous with polyhedral combinatorics, where from a given collection of valid inequalities we would like to select only facets.

Let us now specify one choice of the *f-parameters* that satisfies (46). For a set S of classes, (46) yields

$$f_i = \mu_r f(r) \sum_{r' \in S} p_{rr'} - \mu_r \sum_{r' \in S} p_{rr'} f(r') + \mu_r f(r) \sum_{r' \notin S} p_{rr'}, \qquad \forall r \in C_i \cap S$$

which implies

$$\frac{f_i}{\mu_r} = f(r) - \sum_{r' \in S} p_{rr'} f(r'), \qquad \forall r \in C_i \cap S$$

Thus, due to (46), in order to explicitly determine the *f-parameters*, it suffices to select nonnegative constants f_i, for each station i with $C_i \cap S$ nonempty. One natural choice of these f_i's that appears to provide fairly tight bounds is to let $f_i = 1$, for all stations i with $C_i \cap S$ nonempty. This leads to $f_S(r)$ being equal to the expected remaining processing time until a job of class r exits the set of classes S. With this choice, the parameters $f_S(r)$ can be determined by solving the system of equations

$$f_S(r) = \frac{1}{\mu_r} + \sum_{r' \in S} p_{rr'} f_S(r'), \qquad r \in S. \tag{48}$$

Moreover this choice of the *f-parameters* causes the denominator of (47) to be of form $1 - \sum_{r \in S} \lambda_r / \mu_r$, which is the natural heavy traffic term; this is a key reason why we believe that it leads to tight bounds. Our claim is also supported by the fact that in indexable problems (Section 2), this choice of the *f-parameters* yields an exact characterization.

We next present a *nonparametric method* for deriving constraints on the achievable performance region. This variation has also been derived independently in Kumar and Kumar [36]. We use again a function of the form

$$R(t) = \sum_{r=1}^{R} f(r) n_r(t) \tag{49}$$

where $f(r)$ are scalars that we call *f-parameters*. We introduce $B_{0i}(t)$ to denote the event that node i is idle at time t. We then define

$$I_{rr'} = E[1\{B_r(\tau_k)\} n_{r'}(\tau_k)] \tag{50}$$

and

$$N_{ir'} = E[1\{B_{0i}(\tau_k)\} n_{r'}(\tau_k)], \tag{51}$$

where $1\{\cdot\}$ is the indicator function and the expectations are taken with respect to the invariant distribution.

Theorem 9 *For every scheduling policy satisfying Assumption A, the following relations hold:*

$$2\mu_r I_{rr} - 2\sum_{r'=1}^{R} \mu_{r'} p_{r'r} I_{r'r} - 2\lambda_{0r}\lambda_r x_r = \lambda_{0r} + \lambda_r(1 - p_{rr}) + \sum_{r' \neq r} \lambda_{r'} p_{r'r} \qquad r = 1, \ldots, R \tag{52}$$

and

$$\mu_r I_{rr'} + \mu_{r'} I_{r'r} - \sum_{w=1}^{R} \mu_w p_{wr} I_{wr'} - \sum_{w=1}^{R} \mu_w p_{wr'} I_{wr} - \lambda_{0r}\lambda_{r'} x_{r'} - \lambda_{0r'}\lambda_r x_r = -\lambda_r p_{rr'} - \lambda_{r'} p_{r'r} \qquad \forall r, r' \text{ such that } r > r'. \tag{53}$$

$$\sum_{r \in C_i} I_{rr'} + N_{ir'} = \lambda_{r'} x_{r'} \tag{54}$$

$$I_{rr'} \geq 0, \; N_{ir'} \geq 0, \; x_i \geq 0$$

Proof We uniformize as in Theorem 8 and proceed similarly to obtain the recursion

$$\begin{aligned} &E[R^2(\tau_{k+1}) \mid \mathbf{n}(\tau_k)] = \\ &\sum_{r=1}^{R} \lambda_{0r}(R(\tau_k) + f(r))^2 + \\ &\sum_{r=1}^{R} \mu_r 1\{B_r(\tau_k)\} \left[\sum_{r'=1}^{R} p_{rr'}(R(\tau_k) - f(r) + f(r'))^2 + p_{r0}(R(\tau_k) - f(r))^2 \right] + \\ &\sum_{r=1}^{R} \mu_r 1\{\overline{B}_r(\tau_k)\} R^2(\tau_k) \end{aligned}$$

Rearranging terms and taking expectations with respect to the invariant distribution, we obtain

$$\begin{aligned} 2\sum_{r=1}^{R} \mu_r \left[\sum_{r'=1}^{R} p_{rr'}(f(r) - f(r')) + p_{r0} f(r) \right] E[1\{B_r(\tau_k)\} R(\tau_k)] - 2\sum_{r=1}^{R} \lambda_{0r} f(r) E[R(\tau_k)] \\ = \sum_{r=1}^{R} \lambda_{0r} f^2(r) + \sum_{r=1}^{R} \lambda_r \left[\sum_{r'=1}^{R} p_{rr'}(f(r) - f(r'))^2 + p_{r0} f^2(r) \right] \end{aligned} \tag{55}$$

Moreover, it is seen from (49) and (50) that

$$E[1\{B_r(\tau_k)\} R(\tau_k)] = \sum_{r'=1}^{R} f(r') I_{rr'}.$$

Let us define the vector $\mathbf{f} = (f(1), \ldots, f(R))$. We note that both sides of (55) are quadratic functions of f. In particular, (55) can be written in the form

$$\mathbf{f}'\mathbf{Q}\mathbf{f} = \mathbf{f}'\mathbf{Q}_0\mathbf{f}, \tag{56}$$

for some symmetric matrices $\mathbf{Q}$, $\mathbf{Q}_0$. Since (56) is valid for all choices of $\mathbf{f}$, we must have $\mathbf{Q} = \mathbf{Q}_0$. It only remains to carry out the algebra needed in order to determine the entries of the matrices $\mathbf{Q}$ and $\mathbf{Q}_0$. From (55), equality of the rth diagonal entries of $\mathbf{Q}$ and $\mathbf{Q}_0$ yields (52), and equality of the off-diagonal terms yields (53). Due to symmetry, it suffices to consider $r > r'$.

Finally, since the events $B_r(\tau_k)$ = "server i busy from class r at τ_k", $r \in C_i$, and $B_{0i}(\tau_k)$ ="server i idle at τ_k" are mutually exclusive and exhaustive we obtain (54). □

Remark: An alternative derivation of the equalities of the previous theorem is as follows: we consider a test function g and write

$$E\{E[g(\mathbf{n}(\tau_{k+1}))|\ \mathbf{n}(\tau_k)]\} = E[g(\mathbf{n}(\tau_k))],$$

as long as the indicated expectations exist. By rewriting the previous equality in terms of the instantaneous transition rate matrix for the Markov chain $\mathbf{n}(t)$, and by introducing some new variables, we obtain certain relations between the variables. In particular, (52) can be derived by considering test functions of the form $g(\mathbf{n}(t)) = n_r^2(t)$, while (53) can be derived by considering the test functions of the form $g(\mathbf{n}(t)) = n_r(t)n_{r'}(t)$. Intuitively, we expect that these quadratic test functions capture some of the interaction among different customer classes.

By minimizing $\sum_{r=1}^{R} c_r x_r$ subject to the constraints of Theorem 9 we obtain a lower bound which is no smaller than the one obtained using Theorem 8. In addition, the linear program in Theorem 9 only involves $O(R^2)$ variables and constraints. This should be contrasted with the linear program associated to our nonparametric variation of the method which involved R variables but $O(2^R)$ constraints.

5.2 Indexable systems: Polynomial reformulations

When applied to the systems we studied in Section 2, the parametric method gives as the performance space an extended polymatroid. In particular, for undiscounted branching bandits Bertsimas, Paschalidis and Tsitsiklis [9] obtain exactly the polyhedron (extended polymatroid) outlined in Section 2. The nonparametric method gives a new reformulation of the extended polymatroid using $O(N^2)$ variables and constraints, where N is the number of bandit types. This is interesting because a) it makes it very easy to solve indexable systems with side constraints, using linear programming techniques (see Bertsimas et. al. [9]) and b) it has been conjectured in mathematical programming that whenever linear optimization problems are solvable in polynomial time, they have formulations with a polynomial number of variables and constraints. No such polynomial formulation has been known for polymatroids and extended polymatroids. The nonparametric method produces polynomial reformulations, thus proving the conjecture for this special case. It is interesting that a purely stochastic method (potential function method) proves a nontrivial result in mathematical programming.

We briefly review the application of the method to the undiscounted branching bandit problem described in Section 2. Let τ_k be the sequence of service completions. Let $\chi_i(\tau_k)$ be 1 if at time τ_k a class i customer starts being served. Let $N_r(t)$ be the number of class r customers present in the system at time t. Any nonpreemptive policy satisfies the following formula that describes the evolution of the system:

$$N_i(\tau_{k+1}) = N_i(\tau_k) + \sum_{j=0}^{R} \chi_j(\tau_k)(N_{ji} - \delta_{ij}), \tag{57}$$

We first observe that the value of $\rho_i^+ = E[\chi_i(\tau_k)]$ is the same for all policies and can be obtained as the unique solution of the system of equations

$$\sum_{j=0}^{R} \rho_j^+ E[N_{ji}] = \rho_i^+, \qquad i = 1, \ldots, R, \tag{58}$$

which is obtained by taking expectations of both sides of (57) with respect to the stationary distribution. Moreover,

$$\sum_{i=0}^{R} \rho_i^+ = 1, \tag{59}$$

which follows from the definition of ρ_i^+.

By considering the potential function

$$R(t) = \sum_{i \in E} f_i N_i(t) \tag{60}$$

and applying the evolution equation (57) for $t = \tau_{k+1}$ and using the parametric potential function method we obtain that the performance vector $n_i^+ = E[N_i(\tau_k)]$ satisfies conservation laws, and thus its performance space is an extended polymatroid (see Bertsimas et. al. [9]).

More interestingly, consider the auxiliary variables

$$z_{ji} = E[\chi_j(\tau_k) N_i(\tau_k)], \qquad i, j \in E,$$

Let $\mathbf{z}$ stand for the $R(R+1)$–dimensional vector with components z_{ij}. Notice that $z_{ji} = 0$ if and only if $N_i(\tau_k) > 0$ implies $\chi_j(\tau_k) = 0$; that is, if and only if class i has priority over class j. In particular, a policy is nonidling if and only if $z_{0i} = 0$ for all $i \neq 0$.

Then by applying the nonparametric method we obtain

Theorem 10 *The achievable space* $(\mathbf{n}^+, \mathbf{z})$ *for the undiscounted branching bandit problem is exactly the polyhedron*

$$\sum_{j=0}^{R} \rho_j^+ E[(N_{ji} - \delta_{ji})^2] + 2 \sum_{j=0}^{R} z_{ji} E[N_{ji} - \delta_{ji}] = 0, \ i \in E,$$

$$\sum_{j=0}^{R} z_{jr} E[N_{jr'} - \delta_{jr'}] + \sum_{j=0}^{R} z_{jr'} E[N_{jr} - \delta_{jr}] + \sum_{j=0}^{R} \rho_j^+ E[(N_{jr} - \delta_{jr})(N_{jr'} - \delta_{jr'}] = 0, \ r, r' \in E.$$

$$n_i^+ = \sum_{j \in E} z_{ji}, \ i \in E,$$

$$n_i^+, z_{ij} \geq 0.$$

Notice that the previous characterization involves only a quadratic number of constraints at the expense of a quadratic number of new variables. Having characterized the performance space at service completions Bertsimas et. al. [9] show how to pass to the performance space at arbitrary times. The extended polymatroid that results is identical with the one obtained in Section 2. A characterization corresponding to Theorem 10 is obtained as well.

5.3 Extensions: Routing and Closed Networks

In this section we briefly describe several generalizations to the methods introduced in the previous section. In particular, we treat networks where routing is subject to optimization and closed networks.

Routing and Sequencing

The framework and the notation is exactly as in Section 5.1. Instead of the routing probabilities $p_{rr'}$ being given, we control whether a customer of class r becomes a customer of class r'. For this reason, we introduce $p_{rr'}(\tau_k)$ to denote the probability (which is under our control) that class r becomes r' at time τ_{k+1}, given that we had a class r service completion at time τ_{k+1}. For each class r, we are given a set F_r of classes to which a class r customer can be routed to. (If F_r is a singleton for every r, the problem is reduced to the class with no routing decisions allowed.)

The procedure for obtaining the approximate achievable region is similar to the proof of Theorem 8 except that the constants $p_{rr'}$ are replaced by the random variables $p_{rr'}(\tau_k)$ in the main recursion. We also need to define some additional variables. Similarly with equations (50) and (51), these variables will be expectations of certain products of certain random variables; the routing random variables $p_{rr'}(\tau_k)$ will also appear in such products.

An important difference from open networks is that the application of the nonparametric method to $R(t)$ also yields the traffic equations for the network; these traffic equations are now part of the characterization of the achievable region because they involve expectations of the decision variables $p_{rr'}(\tau_k)$, whereas in Section 5.1 they involved the constants $p_{rr'}$. Application of the method to $R^2(t)$ provides more equations that with some definitional relations between variables, similar to (54), complete the characterization.

Closed Networks

Consider a closed multiclass queueing network with N single server stations (nodes) and R different job classes. There are K customers always present in the closed network. We use the same notation as for open networks except that there are no external arrivals ($\lambda_{0r} = 0$) and the probability that a customer exits the network is equal to zero ($p_{r0} = 0$). We do not allow routing decisions, although routing decisions can be included in a manner similar to the case of open networks.

As in open networks, we only consider sequencing decisions and policies satisfying Assumption A(a); Assumption A(b) is automatically satisfied. We seek to maximize the weighted throughput

$$\sum_{r=1}^{R} c_r \lambda_r,$$

where $\lambda_r = \mu_r E[1\{B_r(\tau_k)\}]$ is the throughput of class r. The derivation of the approximate achievable region is routine and we omit the details.

Although we presented the method for systems with Poisson arrivals and exponential service time distributions, the method can be easily extended to systems with phase-type distributions by introducing additional variables. Moreover, one can use the method to derive bounds on the performance of particular

policies. This is done by introducing additional constraints that capture as many features of a particular policy as possible.

5.4 Higher Order Interactions and Nonlinear Characterizations

The methodology developed so far leads to a *polyhedral* set that contains the achievable region and takes into account *pairwise* interactions among classes in the network. We can extend the methodology and its power as follows:

1. We take into account *higher order interactions* among various classes by extending the potential function technique developed thus far.

2. We obtain *nonlinear* characterizations of the achievable region in a systematic way by using ideas from the powerful methodology of semidefinite programming.

In particular, we show how to construct a sequence of progressively more complicated nonlinear approximations (relaxations) which are progressively closer to the exact achievable space.

Higher Order Interactions

The results so far have made use of the function $R(t) = \sum_{r=1}^{R} f(r)n_r(t)$ and were based essentially on the equation

$$E\big[E[R^2(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]\big] = E[R^2(\tau_k)].$$

By its nature, this method takes into account only pairwise interactions among the various classes. For example, the nonparametric method introduces variables $E[1\{B_r(\tau_k)\}n_j(\tau_k)]$, taking into account the interaction of classes r and j.

We now describe a generalization that aims at capturing higher order interactions. Consider again an open queueing network of the form described in Section 5.1, where there are no routing decisions to be made. We apply the nonparametric method by deriving an expression for $E[R^3(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]$ and then collecting terms; alternatively, we can use test functions $g(\mathbf{n}(t)) = n_r(t)n_j(t)n_k(t)$. (We need to modify Assumption A(b) and assume that $E[n_r^3(t)] < \infty$.) In addition to the variables $I_{rj} = E[1\{B_r(\tau_k)\}n_j(\tau_k)]$, we introduce some new variables, namely,

$$H_{rjk} = E[1\{B_r(\tau_k)\}n_j(\tau_k)n_k(\tau_k)] \tag{61}$$

and

$$M_{jk} = E[n_j(\tau_k)n_k(\tau_k)]$$

The recursion for $E[R^3(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]$ leads to a set of linear constraints involving the variables $\{(n_r, I_{rj}, H_{rjk}, M_{jk})$

The new variables we introduced take into account interactions among three customer classes and we expect that they lead to tighter constraints. Another advantage of this methodology is that we can

now obtain lower bounds for more general objective functions involving the *variances* of the number of customers of class r, since the variables $M_{jj} = E[n_j^2(\tau_k)]$ are now in the augmented space.

Naturally, we can continue with this idea and apply the nonparametric method to $E[R^i(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]$ for $i \geq 4$. In this way, we take into account interactions among i classes in the system. There is an obvious tradeoff between accuracy and tractability in this approach. If we denote by P_i the set obtained by applying the nonparametric method to $E[R^i(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]$, the approximate performance region that takes into account interactions of up to order i is $\cap_{l=1}^{i} P_l$. The dimension of this set and the number of constraints is $O(R^i)$, which even for moderate i can be prohibitively large.

The explicit derivation of the resulting constraints is conceptually very simple but is algebraically involved and does not yield any additional insights. In fact this derivation is not hard to automate. Bertsimas et. al [8] have used the symbolic manipulation program Maple to write down the recursion for $E[R^i(\tau_{k+1}) \mid \mathbf{n}(\tau_k)]$, and generate equality constraints by collecting the coefficients of each monomial in the f–parameters.

Nonlinear Interactions

From higher order interactions we find linear equalities on joint moments of the type

$$s(j_1, \ldots, j_R) = E[n_1^{j_1}(\tau_k) \ldots n_R^{j_R}(\tau_k)].$$

Clearly these quanitities are related. We next outline that their relation can be captured with the condition that a certain matrix is positive semidefinite. The general problem is as follows:
Under what conditions the numbers $s(j_1, \ldots, j_R)$ represent the joint moments $E[Y_1^{j_1} \ldots Y_R^{j_R}]$, where $Y_1, \ldots, Y_R$ is a collection of random variables?

We consider the vector

$$\mathbf{y} = (1, y_1, \ldots, y_R, y_1^2, y_1 y_2, \ldots, y_1 y_R, y_2^2, y_2 y_3, \ldots y_2 y_R, \ldots, y_R^2, y_1^3, \ldots)',$$

in which we include all the monomials of degree 1, 2, ... If the numbers $s(j_1, \ldots, j_R)$ represent joint moments, then there exists a joint density $f(.)$ such that

$$s(j_1, \ldots, j_R) = \int y_1^{j_1} \ldots y_R^{j_R} f(y_1, \ldots, y_R) dy_1 \ldots dy_R.$$

We consider the matrix

$$\mathbf{Z}(\mathbf{y}) = \mathbf{y}\mathbf{y}'.$$

Clearly the matrix $\mathbf{Z}(\mathbf{y})$ is symmetric and positive semidefinite. Therefore, the matrix

$$\mathbf{Z} = \int Z(\mathbf{y}) f(\mathbf{y}) dy$$

is also symmetric positive semidefinite. Notice that the entries of the matrix $\mathbf{Z}$ are the numbers $s(j_1, \ldots, j_R)$. Therefore, a linking constraints for the variables $s(j_1, \ldots, j_R)$ is that the matrix $\mathbf{Z}$ is symmetric and positive semidefinite. Clearly these are necessary conditions. If one is only given

$s(i,j) = E[Y_i Y_j]$, the condition that the matrix $\mathbf{Z}$ is semidefinite corresponds exactly to the requirement that the covariance matrix is semidefinite.

In summary we can strengthen the bounds we obtain from linear relaxations, by obtaining linear constraints on higher order moments and adding the constraint that the matrix $\mathbf{Z}$ is symmetric positive semidefinite. This is an instance of a positive semidefinite optimization problem that can be solved efficiently.

5.5 Performance of the bounds

In this section we summarize the principal insights form the extensive computational results reported in Bertsimas et. al. [8] regarding the performance of these methods:

1. The lower bound obtained by the nonparametric variation of the method is at least as good as the lower bound obtained by the parametric method as expected. In fact in more involved networks it is strictly better. The reason is that the nonparametric method takes better into account the interactions among various classes.

2. The strength of the lower bounds obtained by the potential function method is comparable to the strength of the "pathwise bound" derived in Ou and Wein [42].

3. The bounds are very good in imbalanced traffic conditions and the strength of the bounds increases with the traffic intensity. A plausible explanation is that in imbalanced conditions the behavior of the system is dominated by a single bottleneck station and for single station systems we know that the bounds are exact.

4. In balanced traffic conditions, the bounds also behave well, especially when the traffic intensity is not very close to one.

5. In certain routing examples, the method is asymptotically exact (as $\rho \to 1$), which is very encouraging.

6. In closed network examples, the bounds were extremely tight.

7. When applied to indexable systems considered in Section 2 the parametric variation of the method produces the exact performance region (extended polymatroid), while the nonparametric variation leads to a reformulation of the achievable region with a quadratic (as opposed to exponential) number of constraints. This is particularly important when side constraints are present, because we can now apply linear programming methods on a polynomial size formulation of the problem.

The techniques in this section are generalizable to an arbitrary stochastic control problem as follows:

1. Given a stochastic control problem, first write the equations that describe the evolution of the system (see for example (57)).

2. Define a potential function of the type (60) and apply the nonparametric variation of the method.

3. By defining appropriate auxiliary variables, find a relaxation of the achievable region.

4. By exploring higher order interactions, obtain stronger relaxations.

Overall, the power of the method stems from the fact that it takes into account higher order interactions among various classes. The first order method is as powerful as conservation laws since it leads to exact characterizations in indexable systems. As such, this approach can be seen as the natural extension of conservation laws.

6 Loss networks

In this section we consider the problem of routing calls/messages in a telephone/communication network. The important difference with the problem of the previous section is that this is a loss network, in the sense that calls that can not be routed are lost as opposed to being queued. Because of its importance the problem has attracted the attention of many researchers (for a comprehensive review see Kelly [32]).

We will describe the problem in terms of a telephone network, which is represented by a directed graph $G = (V, A)$ with $N = |V|$ nodes and $N(N-1)$ ordered pairs of nodes (i,j). Calls of type (i,j) need to be routed from node i to node j and carry a reward $w(i,j)$. Arriving calls of type (i,j) may be routed either directly on link (i,j) or on a route $r \in R(i,j)$ (a path in G), where $R(i,j)$ is the set of alternative routes for calls of type (i,j). Let C_{ij} be the capacity of the direct link (i,j) ($C_{ij} = 0$ if the link (i,j) is missing). Let $S(i,j) = \{(i,j)\} \cup R(i,j)$ be the set of routes for calls of type (i,j). When a call of type (i,j) arrives at node i, it can be routed through r only if there is at least one free circuit on each link of the route. If it is accepted, it generates a revenue of $w(i,j)$ and simultaneously holds one circuit on each link of the route r for the holding period of the call. Incoming calls of type (i,j) arrive at the network according to a Poisson process of rate λ_{ij}, while its holding period is assumed to be exponentially distributed with rate μ_{ij} and independent of earlier arrivals and holding times. The problem is to find an acceptance/rejection policy to maximize the total expected reward in steady state.

Our goal in this section is to illustrate that modeling the problem as a MDP and aggregating the state space leads to an upper bound on the expected reward and near optimal policies. We report results obtained in Bertsimas and Cryssikou [5]. Kelly [33] has developed an approach based on nonlinear optimization to obtain bounds. We illustrate the methodology first for the case of one link and two classes of arriving calls and then for a network.

Single Link

Consider a single link of capacity C, which is offered two different types of calls. Let λ_1 and λ_2 be the arrival rates, μ_1 and μ_2 be the service rates and w_1 and w_2 be the reward generated by acceptance of

a type 1 and 2 call, respectively. We are interested in maximizing the expected reward $\sum_{i=1}^{2} w_i E[n_i]$, where n_i is the number of calls of type i in steady-state.

We can formulate the problem exactly as a MDP (see Heyman and Sobel [29], p. 183) by introducing $2C$ variables

$$x_{ij} = P\{A_1 = 1 |\ n_1 = i, n_2 = j\},\ y_{ij} = P\{A_2 = 1 |\ n_1 = i, n_2 = j\},$$

where A_r is the decision $(1,0)$ of whether a call of type r is accepted (rejected) respectively. The balance equations lead to a formulation of the problem as a linear programming problem in the variables x_{ij}, y_{ij}. The optimal solution of this linear program completely specifies the optimal policy. In contrast Kelly [33] uses dynamic programming to solve the single link problem optimally.

Network

Let n_{ij}^r be the number of calls of type (i,j) routed through path $r \in S(i,j)$ in steady-state. We consider *Markovian* policies, in the sense that a decision is only based on the current state. Let A_{ij}^r denote the $(0,1)$ decision of whether an incoming call (i,j) is routed through route $r \in S(i,j)$. We can model the problem as an MDP in an exponential state-space. Instead we will consider relaxations of the exact MDP.

Analogously to the ideas we introduced in Section 3 on restless bandits we introduce a first order relaxation by introducing the variables

$$P\{A_{ij}^r = 1 |\ n_{pq}^s = l\}$$

and a second order relaxation of the MDP by introducing the variables

$$P\{A_{ij}^r = 1 |\ n_{pq}^s = l,\ n_{uv}^f = k\}.$$

Notice that the second order relaxation for the single link case is exact. Both relaxations lead to linear programming problems that give an upper bound on the maximum reward.

Kelly [33] considers a nonlinear relaxation using the variables

$$x_{ij} = E[n_{ij}^{(i,j)}],\ y_{ij} = E[\sum_{r \neq (i,j)} n_{ij}^r].$$

In terms of the strength of the relaxation Bertsimas and Cryssikou [5] report that for symmetric networks and symmetric data the second order relaxation and the relaxation of Kelly [33] are close, while under asymmetry either in the topology or the data, the second relaxation is stronger.

We next outline how we can define heuristic policies from these relaxations. An optimal policy should specify the probability of accepting a call through a particular route given the number of calls in each link. Given that the relaxations only compute $P\{A_{ij}^r = 1 |\ n_{pq}^s = l\}$ (the first order relaxation) or $P\{A_{ij}^r = 1 |\ n_{pq}^s = l,\ n_{uv}^f = k\}$ (the second order relaxation), we can propose an approximation as follows. From the first or second order relaxation we accept a call of type (i,j) through route r with

probability $P\{A_{ij}^r = 1 | \ n_{ij}^r = l\}$, i.e., we use only limited information from the relaxations. In several computational experiments Bertsimas and Cryssikou [5] report that this policy behaves within $0 - 3\%$ from the upper bound. In other words, both the bound and the policyare nearly optimal.

7 Limitations of the proposed approach

The strategy we have presented through examples to obtain bounds and policies for stochastic control problems consists of the following steps:

1. We first define appropriate decision variables.
2. Using the potential function method or using particular properties of the control problem we generate a relaxation (or a series of relaxations) of the achievable space.
3. If the relaxation is exact we have found an exact solution of the problem (for example indexable systems). If not, optimizing over the relaxation we obtain a bound on the achievable performance.
4. By using information from the bounds we construct near optimal policies (for example restless bandits, polling systems and loss networks) using linear and integer programming.

Two questions that naturally arise:

1. Having solved the relaxation of the stochastic control problem, how does one construct policies?
2. Can we explicitly characterize the achievable space of the stochastic control problem or are there limitations to our ability to describe the achievable region?

The answer to the first question is not yet complete. For restless bandits, polling systems and loss networks we proposed empirically effective heuristics from the relaxations, but the heuristics were somewhat ad hoc. For queueing networks we did not propose policies. Avram, Bertsimas and Ricard [2] formulate optimization models for fluid models of multiclass networks and using the maximum principle and infinite linear programming duality find optimum threshold policies that can then be simulated for the queueing network control problem.

Regarding the second question, complexity theory provides insights to our ability to fully characterize the achievable region. There was an important distinction among the stochastic control problems we addressed. While indexable systems are solvable very efficiently by an essentially greedy method (an indexing rule), the research community has had significant difficulties with the other problem classes. It is fair to say that there do not exist general methods that solve large scale versions of these problems efficiently. One naturally wonders whether these problems are inherently hard or we could perhaps solve them if we understand them at a deeper level.

The following brief discussion is informal with the goal of explaining the fundamental distinction in complexity of these problems (for formal definitions see Papadimitriou [44]). Let $\mathcal{P}$, $\mathcal{PSPACE}$, $\mathcal{EXPTIME}$ be the class of problems solvable by a computer (a Turing machine) in polynomial time, polynomial space (memory) and exponential time respectively. Let $\mathcal{NP}$ the class of problems which, if given a polynomial size certificate for the problem, we can check in polynomial time whether the certificate is valid. A significant tool to assess the difficulty of a problem is the notion of *hardness*. For example if we show that a problem is $\mathcal{EXPTIME}$-hard, it means that it *provably* does not have a polynomial (efficient) algorithm for its solution, since we know that $\mathcal{EXPTIME} \neq \mathcal{P}$. If we prove that a problem is $\mathcal{PSPACE}$-hard it means that if we find a polynomial algorithm for the problem, then all problems in $\mathcal{PSPACE}$ have efficient algorithm. It is widely believed (but not yet proven) that $\mathcal{PSPACE} \neq \mathcal{P}$; therefore, proving that a problem is $\mathcal{PSPACE}$-hard is a very strong indication (but not yet a proof) of the inherent hardness of the problem.

Papadimitriou and Tsitsiklis [45] have recently shown that the multiclass queueing network problem is $\mathcal{EXPTIME}$-hard and the restless bandit problem is $\mathcal{PSPACE}$-hard. It is immediate that the polling optimization problem is $\mathcal{NP}$-hard, since it has as a special case the traveling salesman problem. Modifications of the proof methods in [45] show that the loss network problem is also $\mathcal{EXPTIME}$-hard, and the polling system problem is $\mathcal{PSPACE}$-hard. Finally, *indexable* systems have efficient algorithms and, therefore they belong in the class $\mathcal{P}$. In Figure 3 we summarize this discussion and classify these problems in terms of their computational complexity. The difficulty the research community has had in developing efficient methods for the stochastic control problems is a reflection of their complexity.

In view of the complexity of these problems, we won't be able to characterize completely as a convex region with a polynomial number of variables and constraints the performance space of queueing and loss networks. Moreover, it is unlikely we would be able to characterize the achievable space for restless bandits and polling systems.

8 Open Problems

Our previous discussion left open the following questions:

1. Find a systematic technique to propose policies from relaxations.

2. Develop techniques to bound the closeness of heuristic policies to relaxations. Techniques developed in the field of approximability of combinatorial problems might be useful here (see for example Bertsimas and Vohra [11]).

3. Investigate computationally the power of semidefinite relaxations for multiclass queueing networks.

In closing, we hope that the results we reviewed will be of interest to applied probabilists, as they provide new interpretations, proofs, algorithms, insights and connections to important problems in

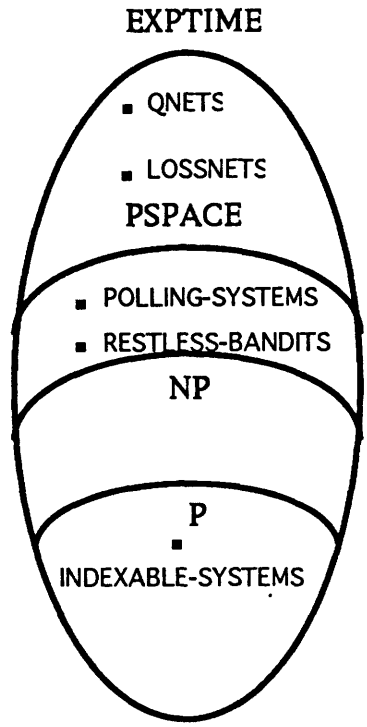

Figure 3: Classification of the complexity of the stochastic control problems we addressed.

stochastic control, as well as to mathematical optimizers, since they reveal a new and important area of application.

Acknowledgments

This research was supported in part by a Presidential Young Investigator Award DDM-9158118 with matching funds from Draper Laboratory. I would like to express my appreciation to my Ph.D students Thalia Cryssikou, David Gamarnik, Haiping Xu, Andrew Luo, Gina Mourtzinou, Jose Niño-Mora, Yiannis Paschalidis, Michael Ricard and Chungpiaw Teo and my colleagues Florin Avram of Northeastern University, John Tsitsiklis and Larry Wein of MIT for contributing to my understanding of the area of stochastic control. I would also like to thank a reviewer of the paper and the editor of the special issue Professor Shaler Stidham for very insightful comments that benefited the paper substantially.

References

[1] E. Anderson and P. Nash, (1987), *Linear Programming in Infinite-dimensional spaces; theory and applications*, John Wiley.

[2] F. Avram, D. Bertsimas and M. Ricard (1994), "Optimization of multiclass fluid queueing networks: a linear control approach", to appear in *Stochastic networks; proceedings of the IMA*, (F. Kelly and R. Williams, editors).

[3] D. Atkins and H. Chen, (1993), "Dynamic scheduling control for a network of queues", to appear.

[4] R.E. Bellman, (1957), *Dynamic Programming*, Princeton University Press, Princeton.

[5] D. Bertsimas and T. Cryssikou, (1995), "Bounds and policies for loss networks", in preparation.

[6] D. Bertsimas and J. Niño-Mora, (1993), "Conservation laws, extended polymatroids and multiarmed bandit problems; a unified approach to indexable systems", to appear in *Mathematics of Operations Research.*

[7] D. Bertsimas and J. Niño-Mora, (1994), "Restless bandits, linear programming relaxations and a primal-dual heuristic", submitted for publication.

[8] D. Bertsimas, I. Paschalidis and J. Tsitsiklis, (1994), "Optimization of multiclass queueing networks: polyhedral and nonlinear characterizations of achievable performance", *Annals of Applied Probability*, 4, 1, 43-75.

[9] D. Bertsimas, I. Paschalidis and J. Tsitsiklis, (1994), "Branching Bandits and Klimov's Problem: Achievable Region and Side Constraints", submitted for publication.

[10] D. Bertsimas and H. Xu, (1993), "Optimization of polling systems and dynamic vehicle routing problems on networks", submitted for publication.

[11] D. Bertsimas and R. Vohra, (1994) "Linear programming relaxations, approximation algorithms and randomization: a unified approach to covering problems", submitted for publication.

[12] P. P. Bhattacharya, L. Georgiadis and P. Tsoucas, (1991), "Extended polymatroids: Properties and optimization", *Proceedings of International Conference on Integer Programming and Combinatorial Optimization (Carnegie Mellon University).* Mathematical Programming Society, 298-315.

[13] O.J. Boxma, H. Levy and J.A. Weststrate, (1990), "Optimization of Polling Systems", in *Performance '90*, eds. P. King, I. Mitrani, R. Pooley, North-Holland, Amsterdam, 349-361.

[14] H. Chen and D. Yao, (1989), "Optimal scheduling control of a multiclass fluid network", to appear in *Operations Research.*

[15] E. Coffman and I. Mitrani, (1980), "A characterization of waiting time performance realizable by single server queues", *Operations Research*, 28, 810-821.

[16] D. R. Cox and W. L. Smith, (1961), *Queues*, Methuen (London) and Wiley (New York).

[17] J. Edmonds, (1970), "Submodular functions, matroids and certain polyhedra", in *Combinatorial Structures and Their Applications*, 69-87. R. Guy *et al.* (eds.), Gordon and Breach, New York.

[18] A. Federgruen and H. Groenevelt, (1988a), "Characterization and optimization of achievable performance in general queueing systems", *Operations Research*, 36, 733-741.

[19] A. Federgruen and H. Groenevelt, (1988b), "$M/G/c$ queueing systems with multiple customer classes: Characterization and control of achievable performance under nonpreemptive priority rules", *Management Science*, 34, 1121-1138.

[20] E. Gelenbe and I. Mitrani, (1980), *Analysis and Synthesis of Computer Systems*, Academic Press, New York.

[21] J. C. Gittins and D. M. Jones, (1974), "A dynamic allocation index for the sequential design of experiments". In J. Gani, K. Sarkadi and I. Vince (eds.), *Progress in Statistics European Meeting of Statisticians 1972*, vol. 1. Amsterdam: North-Holland, 241-266.

[22] J. C. Gittins, (1979), "Bandit processes and dynamic allocation indices", *Journal of the Royal Statistical Society Series*, B 14, 148-177.

[23] J. C. Gittins, (1989), *Bandit Processes and Dynamic Allocation Indices*, John Wiley.

[24] K. D. Glazebrook, (1976), "Stochastic scheduling with order constraints", *Int. Journal Systems Science*, 657-666.

[25] K. D. Glazebrook, (1987), "Sensitivity analysis for stochastic scheduling problems", *Mathematics of Operations Research*, 12, 205-223.

[26] J. M. Harrison, (1975a), "A priority queue with discounted linear costs", *Operations Research*, 23, 260-269.

[27] J. M. Harrison, (1975b), "Dynamic scheduling of a multiclass queue: discount optimality", *Operations Research*, 23, 270-282.

[28] J. M. Harrison and L.M. Wein, "Scheduling network of queues: Heavy traffic analysis of a simple open network", *Queueing Systems Theory and Applications*, 5, 265-280.

[29] Heyman, D. P. and Sobel, M. J. (1984). *Stochastic Models in Operations Research, vol. II: Stochastic Optimization*, McGraw-Hill, New York.

[30] M. Hofri and K.W. Ross, (1988), "On the optimal control of two queues with server set up times and its analysis", *SIAM J. on Computing*, 16, 399-419.

[31] W. A. Horn, (1972), "Single-machine job sequencing with treelike precedence ordering and linear delay penalties. *SIAM J. Appl. Math.*, 23, 189-202.

[32] F. Kelly, (1991), "Loss networks", *Annals of Applied Probability*, 1, 319-378.

[33] F. Kelly, (1994), "Bounds on the performance of dynamic routing schemes for highly connected networks", *Mathematics of Operations Research*, 19, 1-20.

[34] L. Kleinrock and H. Levy, (1988), "The analysis of random polling systems", *Operations Research*, 36, 716-732.

[35] G. P. Klimov, (1974), "Time sharing service systems I", *Theory of Probability and Applications*, 19, 532-551.

[36] S. Kumar and P.R. Kumar, (1993), "Performance bounds for queueing networks and scheduling policies", preprint.

[37] H. Levy and M. Sidi, (1990), "Polling systems: applications, modelling and optimization", *IEEE Transactions on Communications*, 38, 10, 1750-1760.

[38] H. Levy, M. Sidi and O.J. Boxma, "Dominance relations in polling systems", *Queueing Systems*, Vol.6, No.2, pp155-171, 1990.

[39] L. Lovász and A. Schrijver, (1990), "Cones of matrices and set functions and 0-1 optimization", *SIAM Jour. Opt.*, 166-190.

[40] I. Meilijson, and G. Weiss, (1977), "Multiple feedback at a single-server station. *Stochastic Process. Appl.*, 5, 195-205.

[41] Murty, K. G. (1983). *Linear Programming.* Wiley, New York.

[42] Z. Ou and L. Wein, (1992), "Performance bounds for scheduling queueing networks", *Annals of Applied Probability*, 2, 460-480.

[43] Queyranne, M. (1993), "Structure of a Simple Scheduling Polyhedron", *Math. Programming*, 58, 263-285.

[44] C. Papadimitriou, (1994), *Computational Complexity*, Addison-Wesley.

[45] C. Papadimitriou and J. Tsitsiklis, (1993), "Complexity of queueing network problems", extended abstract.

[46] M. Reiman and L. Wein (1994), "Dynamic scheduling of a two-class queue with setups", submitted for publication.

[47] K. W. Ross and D. D. Yao (1989), "Optimal dynamic scheduling in Jackson Networks", *IEEE Transactions on Automatic Control*, 34, 47-53.

[48] M. H. Rothkopf, (1966a), "Scheduling independent tasks on parallel processors", *Management Science*, 12, 437-447.

[49] M. H. Rothkopf, (1966b), "Scheduling with Random Service Times", *Management Science*, 12, 707-713.

[50] J. G. Shanthikumar and D. D. Yao, (1992), "Multiclass queueing systems: Polymatroidal structure and optimal scheduling control", *Operations Research*, 40, Supplement 2, S293-299.

[51] W. E. Smith, (1956), "Various optimizers for single-stage production", *Naval Research Logistics Quarterly*, 3, 59-66.

[52] H. Takagi, (1986), *Analysis of Polling Systems*, MIT press.

[53] H. Takagi, (1988), *Queueing Analysis of Polling Systems, ACM Comput. Surveys*, 20, 5-28.

[54] D. W. Tcha and S. R. Pliska, (1977), "Optimal control of single-server queueing networks and multi-class $M/G/1$ queues with feedback", *Operations Research*, 25, 248-258.

[55] J. N. Tsitsiklis, (1986), "A lemma on the multiarmed bandit problem", *IEEE Transactions on Automatic Control*, 31, 576-577.

[56] J. N. Tsitsiklis, (1994), "A short proof of the Gittins index theorem", *Annals of Applied Probability*, to appear.

[57] P. Tsoucas, (1991), "The region of achievable performance in a model of Klimov", Technical Report RC16543, IBM T. J. Watson Research Center.

[58] P. P. Varaiya, J. C. Walrand and C. Buyukkoc, (1985), "Extensions of the multiarmed bandit problem: The discounted case", *IEEE Transactions on Automatic Control*, 30, 426-439.

[59] R. Weber, (1992), "On the Gittins index for multiarmed bandits", *The Annals of Applied Probability*, 2, 1024-1033.

[60] G. Weiss, (1988), "Branching bandit processes", *Probability in the Engineering and Informational Sciences*, 2, 269-278.

[61] P. Whittle (1980), "Multiarmed bandits and the Gittins index", *Journal of the Royal Statistical Society*, 42, 143-149.

[62] P. Whittle, (1988), "Restless Bandits: activity allocation in changing world", in *Celebration of Applied Probability*, ed. J. Gani, *Journal of Applied Probabiltity*, 25A, 287-298.