A Mathematical Programming Approach to Stochastic and Dynamic Optimization Problems

Dimitris Bertsimas

WP# - 3668-94 MSA March, 1994

A mathematical programming approach to stochastic and dynamic optimization problems

Dimitris Bertsimas [1]

March 1994

[1]Dimitris Bertsimas, Sloan School of Management and Operations Research Center, MIT, Cambridge, MA 02139. The research of the author was partially supported by a Presidential Young Investigator Award DDM-9158118 with matching funds from Draper Laboratory.

Abstract

We survey a new approach that the author and his co-workers have developed to formulate generic stochastic and dynamic optimization problems as *mathematical programming problems.* The approach has two components: (a) it produces bounds on the performance of an optimal policy, and (b) it develops techniques to construct optimal or near-optimal policies. The central idea for developing bounds is to characterize the region of achievable performance (or performance space) in a stochastic and dynamic optimization problem, i.e., find linear or nonlinear constraints on the performance vectors that all admissible policies satisfy. With respect to this goal we review recent progress in characterizing the performance space and its implications for the following problem classes: Indexable systems (the multi-armed bandit problem and its extensions), polling systems, multiclass queueing networks and loss networks.

We propose three ideas for constructing optimal or near-optimal policies: (1) for systems for which we have an exact characterization of the performance space we outline an adaptive greedy algorithm that gives rise to indexing policies (we illustrate this technique in the context of indexable systems); (2) we use integer programming to construct policies from the underlying descriptions of the performance space (we illustrate this technique in the context of polling systems); (3) we use linear control over polyhedral regions to solve deterministic versions for this class of problems. This approach gives interesting insights for the structure of the optimal policy (we illustrate this idea in the context of multiclass queueing networks).

The unifying theme in the paper is the thesis that better formulations lead to deeper understanding and better solution methods. Overall the proposed approach for stochastic and dynamic optimization parallels efforts of the mathematical programming community in the last fifteen years to develop sharper formulations (polyhedral combinatorics and more recently nonlinear relaxations) and leads to new insights ranging from a complete characterization for indexable systems to tight lower bounds and new algorithms with provable a posteriori guarantees for their suboptimality for polling systems, multiclass queueing and loss networks.

Contents

1 Introduction

In its thirty years history the area of stochastic and dynamic optimization 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 adhoc 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 apriori 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.

In parallel, the theory of computational complexity has shed light to what we feel is the fundamental theoretical question in the area of mathematical programming: what problems are efficiently (polynomially) solvable and what problems are inherently hard. This characterization is quite important from a practical point of view as it dictates the approach for the problem (exact or heuristic). In contrast, only very recently such characterizations have been developed for stochastic optimization problems.

In view of these comments we feel that the following research program is in the heart of the field of stochastic and dynamic optimization:

1. Develop a ***unified*** approach for stochastic and dynamic optimization problems that provides both a feasible solution and a guarantee for its suboptimality (or optimality). Ideally the approach should not fundamentally change when the problem changes.

2. Classify the inherent complexity of classical stochastic and dynamic optimization problems.

Our goal in this paper is primarily to review the recent progress towards the first goal and secondarily to briefly outline the progress towards the second goal. Towards the first goal our plan is to outline the approach broadly and then apply it to 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 (The multi-armed bandit problem and its extensions such as multiclass queues, branching bandits, etc.).

2. Polling systems.

3. Multiclass queueing networks.

4. Multiclass loss networks.

There is an important distinction among these four problem classes. 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. Despite significant progress in recent years, 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 theory of computational complexity developed in computer science in the early 1970s to answer similar questions but for combinatorial optimization problems provides insights to this question (the second goal of the paper), which we briefly outline.

Classification of complexity in stochastic and dynamic optimization

The following brief discussion is informal with the goal of explaining the fundamental distinction in complexity of these problems (for formal definitions see Papadimitriou (1994)). 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 (1993) have recently shown that the multiclass queueing network problem is $\mathcal{EXPTIME}$-hard and a very natural variation of the classical bandit problem known as 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 Papadimitriou and Tsitsiklis (1993) 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 1 we summarize this discussion and classify these problems in terms of their inherent complexity. Figure 1 illustrates that

the difficulty the research community has had in developing efficient methods for classical stochastic and dynamic optimization problems is due to the inherent complexity of the problems.

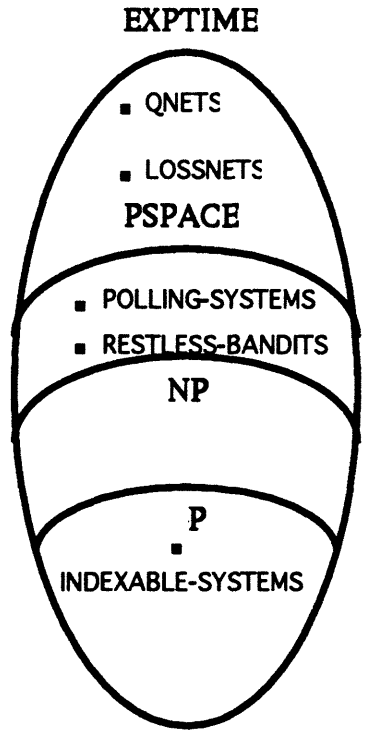

Figure 1: Classification of the complexity of classical stochastic optimization problems.

On the power of formulations in mathematical optimization

In mathematical programming our ability to solve efficiently 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 prove membership of the problem in $\mathcal{P}$. On the other hand, the general *integer optimization problem* is $\mathcal{NP}$-hard, while the general *nonlinear optimization problem* is $\mathcal{PSPACE}$-hard. Analogously to Figure 1 we classify in Figure 2 these classical mathematical programming problems in terms of their inherent complexity.

Despite the inherent hardness of integer programming (IP), 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 fifteen years much of the effort in integer programming research has been in developing sharper formulations (polyhedral combinatorics and more recently nonlinear relaxations) 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

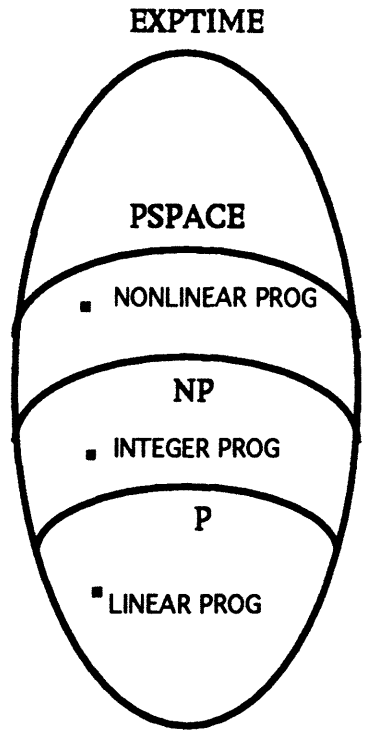

Figure 2: Classification of the complexity of classical mathematical programming problems.

relaxation is closer and closer to the IP. Moreover, one of the most exciting new research directions in mathematical programming is the field of nonlinear relaxations (see for example Lovãsz and Schrijver (1990)) for integer programming problems.

On the power of formulations in stochastic and dynamic optimization

The unifying theme in the present paper is the thesis that better formulations lead to deeper understanding and better solution methods. Motivated by the power of improved formulations for mathematical programming problems, we now like to outline in broad terms the approach to formulate stochastic and dynamic optimization problems as *mathematical programming problems* in an attempt to develop *a unified theory* of stochastic and dynamic optimization. The key idea in this development is the following: Given a stochastic and dynamic optimization 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*, i.e., find constraints on the performance vectors that all admissible policies satisfy. In this way we find a series of relaxations that are progressively closer to the exact region of achievable performance. In Figure 3 we outline the conceptual similarity of this approach to the approach used in integer programming.

As the complexity of stochastic and dynamic optimization problems increases (Figure 1), our ability to fully characterize the region of achievable performance decreases. While for indexable

systems we can obtain full characterizations of the achievable space, in the other three stochastic and dynamic optimization problems we study we only achieve partial characterizations (relaxations). In Sections 2-5 we outline the progress in characterizing the performance space for the four problem classes we are studying. Central in this development is a systematic way (*the potential function method*) to obtain constraints for a stochastic optimization problem, which is described in Section 4.

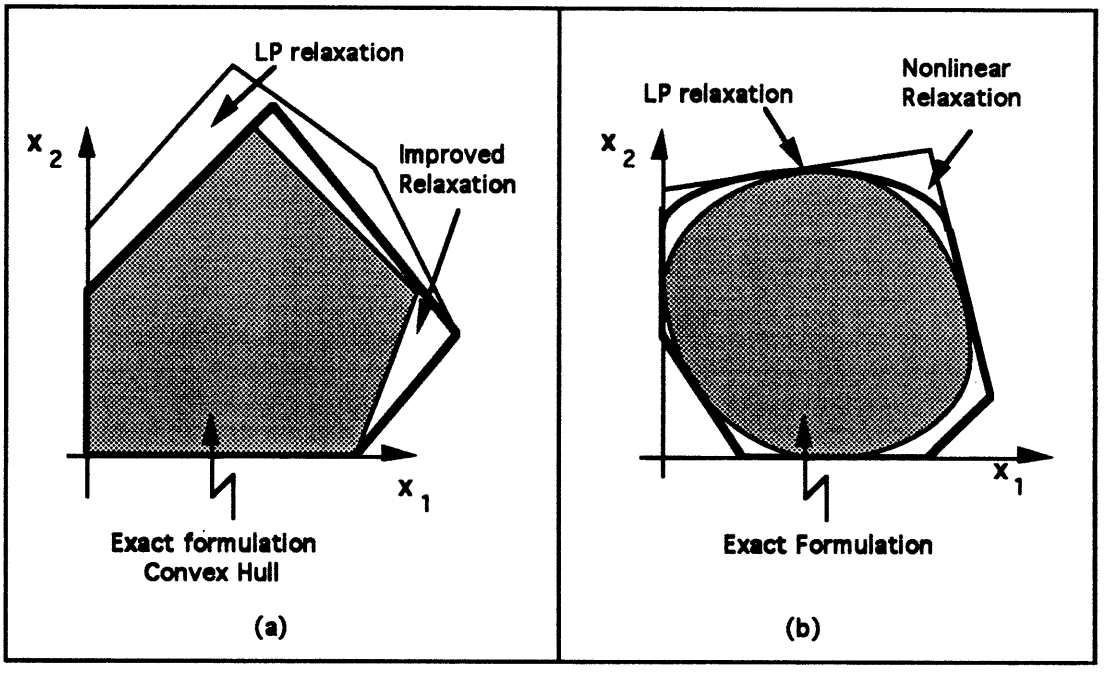

Figure 3: (a) Improved relaxations for integer programming (b) Improved relaxations for stochastic optimization problems.

Techniques to find optimal or near-optimal policies for stochastic and dynamic optimization problems

We propose three ideas for constructing optimal or near-optimal policies:

1. For systems for which we have an exact characterization of the performance space we outline an adaptive greedy algorithm that gives rise to indexing policies. We illustrate this technique in the context of indexable systems in Section 2. This leads to significant advantages, as one can use structural and algorithmic results from a mature field (mathematical programming) to attack stochastic optimization problems, and offers new insights ranging from new algorithms to solve stochastic and dynamic optimization problems to a deeper understanding of their structure and properties. In Section 2 we illustrate that for indexable systems we can exactly characterize the region of achievable performance as a polyhedron of special structure, which is interestingly related with the theory of polymatroids of Edmonds (1970). More importantly, we can use well understood linear programming techniques to unify a very large set of results

regarding indexable systems.

2. We use integer programming to construct policies from the underlying descriptions of the performance space. We illustrate this technique in the context of polling systems in Section 3.

3. We use linear control over polyhedral regions to solve deterministic versions for this class of problems. This approach gives interesting insights for the structure of the optimal policy (we illustrate this idea in the context of multiclass queueing networks in Section 6). This approach is based on the thesis that the *essential difficulty* in stochastic and dynamic optimization is primarily the dynamic and combinatorial character of the problem and only secondarily the stochastic character of the problem. Putting the randomness back in the problem, we simulate proposed policies and thus estimate their performance. The degree of suboptimality is then found by comparing these simulation results to the bounds found from the relaxations.

The paper is structured as follows. In Section 2, 3, 4 and 5 we outline the progress to describe the performance space for indexable systems, polling systems, multiclass queeuing networks and loss networks respectively. In Section 6 we outline the linear control approach over a polyhderal space as an approach to obtain qualitative insight for the structure of the optimal policy. The final section includes concluding remarks and a list of open problems in the field.

2 A Polyhedral Approach to Indexable Systems

Perhaps one of the most important successes in the area of stochastic optimization in the last twenty years is the solution of the celebrated *multi-armed bandit problem*, a generic version of which in discrete time can be described as follows:

The multi-armed 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 $P^k = (p^k_{ij})_{i,j \in E_k}$, 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 (according to a policy u among a set of policies $\mathcal{U}$) in order to maximize expected total discounted reward over an infinite horizon:

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

The problem has numerous applications and a rather vast literature (see Gittins (1989) and the references therein). It was originally solved by Gittins and Jones (1974), 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 (1980) provided a proof based on dynamic programming, subsequently simplified by Tsitsiklis (1986). Varaiya, Walrand and Buyukkoc (1985) and Weiss (1988) provided different proofs based on interchange arguments. Weber (1992) and Tsitsiklis (1993) outlined intuitive proofs.

The multi-armed bandit problem is a special case of a dynamic and 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.

Definition 1 (Indexable Systems) We say that a dynamic and 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 multi-armed 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 multi-armed bandit problem, a variety of dynamic and stochastic scheduling 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 *deep reason* that these nontrivial problems have very efficient solutions both theoretically as well as practically. In particular, *what is the class of stochastic and dynamic scheduling problems that are indexable? Under what conditions are indexable systems decomposable?* But most importantly, *is there a unified way to address stochastic and dynamic scheduling problems that will lead to a deeper understanding of their strong structural properties?*

In the last decade researchers have been using the approach outlined in the Introduction (describing the feasible space of a stochastic and dynamic scheduling problem as a mathematical programming problem) in order to give answers to some of these questions. Coffman and Mitrani (1980) and Gelenbe and Mitrani (1980) first showed using *conservation laws* 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 (1988a), (1988b) 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 also Ross and Yao (1989)). Shanthikumar and Yao (1992) generalized the theory further by observing that if a system satisfies *strong 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 (1961)). Tsoucas (1991) derived the region of achievable performance in the problem of scheduling a multiclass nonpreemptive $M/G/1$ queue with Bernoulli feedback, introduced by Klimov (1974). All these results have been further generalized by Bertsimas and Niño-Mora (1993) who propose *a theory* of conservation laws to establish that the very strong structural properties in the optimization of a class of stochastic and dynamic systems that include the multi-armed bandit problem and its extensions follow from the corresponding strong structural properties of the underlying polyhedra that characterize the regions of achievable performance.

In this section we review the work of Bertsimas and Niño-Mora (1993). In Section 2.1 we define a class of polyhedra called extended polymatroids which describe the region of achievable performance in a large collection of stochastic scheduling problems. In Section 2.2 we consider optimization problems over extended polymatroids. In Section 2.3 we define generalized conservation laws. In Sections 2.4 we consider examples of natural problems that can be analyzed using the theory developed, while in Section 2.5 we outline applications of these characterizations.

2.1 Extended Polymatroids

Let us first establish the notation we will use. Let $E = \{1, \ldots, n\}$. Let x denote a real n-vector, with components x_i, for $i \in E$. For $S \subseteq N$, let $S^c = E \setminus S$, and let $|S|$ denote the cardinality of S. Let 2^E denote the class of all subsets of E. Let $b: 2^E \rightarrow \Re_+$ be set functions that satisfy $b(\emptyset) = 0$. Let $A = (A_i^S)_{i \in E,\ S \subseteq E}$ be a matrix that satisfies

$$A_i^S > 0, \quad \text{for} \quad i \in S \qquad \text{and} \qquad A_i^S = 0, \quad \text{for} \quad i \in S^c, \qquad \text{for all } S \subseteq E, \tag{1}$$

Let $\pi = (\pi_1, \ldots, \pi_n)$ be a permutation of E. For clarity of presentation, it is convenient to intro-

duce the following additional notation. For an n-vector $x = (x_1, \ldots, x_n)^T$ let $x_\pi = (x_{\pi_1}, \ldots, x_{\pi_n})^T$. Let us write

$$b_\pi = (b(\{\pi_1\}), b(\{\pi_1, \pi_2\}), \ldots, b(\{\pi_1, \ldots, \pi_n\}))^T.$$

Let A_π denote the following lower triangular submatrix of A:

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

Let $v(\pi)$ be the unique solution of the linear system

$$\sum_{i=1}^{j} A_{\pi_i}^{\{\pi_1,\ldots,\pi_j\}} x_{\pi_i} = b(\{\pi_1, \ldots, \pi_j\}), \quad j = 1, \ldots, n \tag{2}$$

or, in matrix notation:

$$x_\pi = A_\pi^{-1} b_\pi. \tag{3}$$

Consider the following polyhedra with A, b:

$$\mathcal{P}_c(A,b) = \{\, x \in \Re_+^n : \sum_{i \in S} A_i^S x_i \geq b(S), \quad \text{for } S \subseteq E \,\}, \tag{4}$$

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

$$\mathcal{P}(A,b) = \{\, x \in \Re_+^n : \sum_{i \in S} A_i^S x_i \leq b(S), \quad \text{for } S \subseteq E \,\}, \tag{6}$$

$$\mathcal{B}(A,b) = \{\, x \in \Re_+^n : \sum_{i \in S} A_i^S x_i \geq b(S), \quad \text{for } S \subset E \quad \text{and} \quad \sum_{i \in E} A_i^E x_i = b(E) \}. \tag{7}$$

The following definition is based on Bhattacharya et. al. (1991):

Definition 3 (Extended Polymatroid) We say that polyhedron $\mathcal{P}(A,b)$ is an *extended polymatroid* with ground set E, if the following condition holds:

(i) For every permutation π of E, $v(\pi) \in \mathcal{P}(A,b)$.

In this case we say that $\mathcal{B}(A,b)$ is the corresponding *base polytope*. If condition (i) holds for $\mathcal{P}_c(A,b)$, we say that polyhedron $\mathcal{P}_c(A,b)$ is an *extended contra-polymatroid* with base polytope $\mathcal{B}_c(A,b)$.

If $A_i^S = 1$ for all $i \in S$, $v(\pi) \in \mathcal{P}(A,b)$ 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 (1970).

The next theorem characterizes the extreme points of the polyhedra $\mathcal{B}_c(A,b)$ and $\mathcal{B}(A,b)$.

Theorem 1 (Characterization of Extreme Points) *The set of extreme points of* $\mathcal{B}_c(A,b)$ *is*

$$\{\, v(\pi) : \pi \text{ is a permutation of } E \,\}.$$

As an example, let us 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 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

$$A_i^{\{i\}} x_i^u \geq b(\{i\}),$$

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 give 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 and it is indeed an extended contra-polymatroid base, which is drawn in Figure 4.

The polytope has $3! = 6$ extreme points corresponding to the 6 possible permutations. Extreme points in an extended polymatroid correspond to the performance vectors of complete priority policies. For example, extreme point A in Figure 4 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:

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

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

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

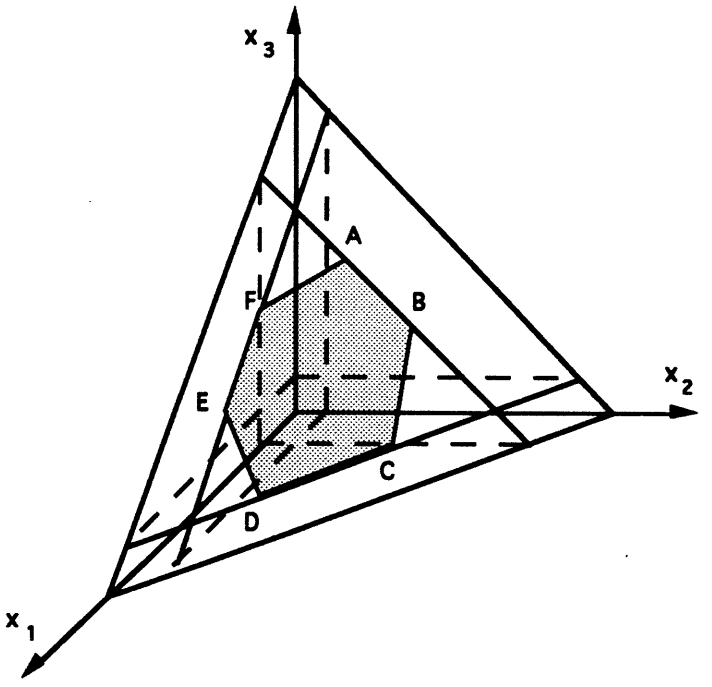

Figure 4: An extended polymatroid in dimension 3.

2.2 Optimization over Extended Polymatroids

Let us consider the following linear program over an extended contra-polymatroid base:

$$(LP_c) \qquad \max\left\{ \sum_{i \in E} R_i x_i : x \in \mathcal{B}_c(A,b) \right\}. \tag{8}$$

The theory over extended polymatroids is completely analogous. Given that the extreme points of the polytope $\mathcal{B}_c(A,b)$ correspond to priorities (permutations), the problem reduces to finding which priority is optimal. Tsoucas (1991) and Bertsimas and Niño-Mora (1993) presented the following *adaptive greedy* algorithm:

Algorithm $\mathcal{AG}$

Input: (R,A).

Output: $(\gamma, \pi, \overline{y}, \nu, \mathcal{S})$, where $\pi = (\pi_1, \ldots, \pi_n)$ is a permutation of E, $\overline{y} = (\overline{y}^S)_{S \subseteq E}$, $\nu = (\nu_1, \ldots, \nu_n)$, 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 $\nu_n = \max\{ \frac{R_i}{A_i^E} : i \in E \}$;

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

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 $\nu_{n-k} = \max\{ \frac{R_i - \sum_{j=0}^{k-1} A_i^{S_{n-j}} \nu_{n-j}}{A_i^{S_{n-k}}} : i \in S_{n-k} \}$;

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

Table 1: Linear programming over extended polymatroids

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

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

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 (1993) show that outputs γ and $\overline{y}$ of algorithm $\mathcal{AG}$ are uniquely determined by the input (R, A). Moreover, $\overline{y}$ 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 4 (Generalized Gittins Indices) Let $\overline{y}$ 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) (a) *Linear program (LP_c) is optimized by $v(\pi)$, where π is any permutation of E such that*

$$\gamma_{\pi_1} \le \cdots \le \gamma_{\pi_n}. \tag{9}$$

Bertsimas and Niño-Mora (1993) also provide closed form expressions for the generalized Gittins indices which can be used for sensitivity analysis. Optimization over the base of an extended polymatroid is analogous to the extended contra-polymatroid case (see Table 1).

In the case that matrix 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 A^k be the submatrix of matrix A corresponding to subsets $S \subseteq E_k$. Let $R^k = (R_i)_{i\in E_k}$. Assume that the following condition holds:

$$A_i^S = A_i^{S\cap E_k} = (A^k)_i^{S\cap E_k}, \quad \text{for } i \in S \cap E_k \quad \text{and } S \subseteq E. \tag{10}$$

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

Theorem 3 (Index Decomposition) *Under condition (10), the generalized Gittins indices corresponding to* (R, A) *and* (R^k, A^k) *satisfy*

$$\gamma_i = \gamma_i^k, \quad \textit{for } i \in E_k \quad \textit{and } k = 1, \ldots, K. \tag{11}$$

Theorem 3 implies that the fundamental reason for decomposition to hold is (10). A useful consequence is the following:

Corollary 1 *Under the assumptions of Theorem 3, an optimal solution of problem* (LP_c) *can be computed by solving* K *subproblems: Apply algorithm* $\mathcal{AG}$ *with inputs* (R^k, A^k), *for* $k = 1, \ldots, K$.

2.3 Generalized Conservation Laws

A natural question is how one can show that the performance space of a stochastic optimization problem is indeed an extended polymatroid. More importantly, what physical properties of the problem imply that the performance space is an extended polymatroid? In this section we identify certain properties of the problem that imply that the performance space is an extended polymatroid.

Consider a generic dynamic and 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 x^π 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 5 (Generalized Conservation Laws) The performance vector x is said to satisfy *generalized conservation laws* if there exist a set function $b(\cdot) : 2^E \to \Re_+$ and a matrix $A = (A_i^S)_{i\in E, S\subseteq E}$ satisfying (1) such that:

(a)

$$b(S) = \sum_{i\in S} A_i^S x_i^\pi \quad \text{for all } \pi : \{\pi_1, \ldots, \pi_{|S|}\} = S \quad \text{and} \quad S \subseteq E; \tag{12}$$

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

$$\sum_{i\in S} A_i^S x_i^u \geq b(S) \quad \text{for all } S \subset E \qquad \text{and} \qquad \sum_{i\in E} A_i^E x_i^u = b(E), \tag{13}$$

or respectively,

$$\sum_{i\in S} A_i^S x_i^u \leq b(S) \quad \text{for all } S \subset E \qquad \text{and} \qquad \sum_{i\in E} A_i^E x_i^u = b(E). \tag{14}$$

The following theorem formalizes the intuition developed in the example of Figure 4 and establishes that if a system satisfies conservation laws (a physical property of the system, which is relatively easy to check) then the performance space is an extended polymatroid:

Theorem 4 *Assume that performance vector x satisfies generalized conservation laws* (12) *and* (13) *(respectively* (12) *and* (14)). *Then*
(a) The vertices of $\mathcal{B}_c(A,b)$ (respectively $\mathcal{B}(A,b)$) are the performance vectors of the static priority rules, and $x^\pi = v(\pi)$ for every permutation π of E.
(b) The extended contra-polymatroid base $\mathcal{B}_c(A,b)$ (respectively the extended polymatroid base $\mathcal{B}(A,b)$) is the performance space.

Suppose that we want to find an admissible policy $u \in U$ that maximizes a linear reward function on the performance $\sum_{i\in E} R_i x_i^u$. This optimal control problem can be expressed as

$$(LP_U) \qquad \max\left\{ \sum_{i\in E} R_i x_i^u : u \in U \right\}. \tag{15}$$

By Theorem 4 this optimal control problem can be transformed into the following linear programming problem:

$$(LP_c) \qquad \max\left\{ \sum_{i\in E} R_i x_i : x \in \mathcal{B}_c(A,b) \right\}. \tag{16}$$

The strong structural properties of extended polymatroids lead to strong structural properties in the control problem.

Let $\gamma_1, \ldots, \gamma_n$ be the generalized Gittins indices of linear program (LP_c). As a direct consequence of the results of Section 2.2 it follows that the control problem (LP_U) is solved by an index policy, with indices given by $\gamma_1, \ldots, \gamma_n$.

Theorem 5 (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.*

2.4 Branching Bandit Processes

Consider the following *branching bandit process* introduced by Weiss (1988), who observed that it can model a large number of stochastic and dynamic optimization 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 $N_i = (N_{ij})_{j\in E}$. 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:

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

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

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

$$I_j(t) = \begin{cases} 1 & \text{if a type } j \text{ job is being engaged at time } t; \\ 0 & \text{otherwise,} \end{cases} \tag{17}$$

the performance measure is

$$x_j^u(\alpha) = \mathrm{E}_u \left[\int_0^\infty e^{-\alpha t} I_j(t)\, dt \right]. \tag{18}$$

Performance Space: The performance vector for branching bandits $x^u(\alpha)$ satisfies generalized conservation laws

(a) $\sum_{i \in S} A^S_{i,\alpha} x^u_i(\alpha) \geq b_\alpha(S)$, for $S \subset E$, with equality if policy u gives complete priority to S^c-jobs.
(b) $\sum_{i \in E} A^E_{i,\alpha} x^u_i(\alpha) = b_\alpha(E)$.

Therefore, the performance space for branching bandits corresponding to the performance vector $x^u(\alpha)$ is the extended contra-polymatroid base $\mathcal{B}_e(A_\alpha, b_\alpha)$. The matrix A_α and set function $b_\alpha(\cdot)$ are calculated as follows:

$$A^S_{i,\alpha} = \frac{1 - \Psi_i^{S^c}(\alpha)}{1 - \Psi_i(\alpha)}, \quad i \in S, \quad S \subseteq E; \tag{19}$$

$$b_\alpha(S) = \frac{1}{\alpha}\prod_{j\in S^c}[\Psi_j^{S^c}(\alpha)]^{m_j} - \frac{1}{\alpha}\prod_{j\in E}[\Psi_j^{E}(\alpha)]^{m_j}, \quad S\subseteq E \tag{20}$$

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

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

Objective function:

$$\max_u V_{u,\alpha}^{(R,C)}(m) = \sum_{i\in E}\hat{R}_{i,\alpha}x_i^u(\alpha) - \sum_{i\in E} m_i C_i = \sum_{i\in E}\Big\{R_i + C_i - \sum_{j\in E}\mathrm{E}[N_{ij}]C_j\Big\}\frac{\alpha\Psi_i(\alpha)}{1-\Psi_i(\alpha)}x_i^u(\alpha) - \sum_{i\in E} m_i C_i \tag{21}$$

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 $E[N]$ 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 $E[N] < I$ (the matrix $I - E[N]$ is positive definite).

Performance space:

I. θ^u satisfies generalized 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 A and set function $b(\cdot)$ are calculated as follows:

$$A_i^S = \frac{\mathrm{E}[T_i^{S^c}]}{\mathrm{E}[v_i]}, \quad i\in S, \tag{22}$$

and

$$b(S) = \frac{1}{2}\mathrm{E}\big[(T_m^E)^2\big] - \frac{1}{2}\mathrm{E}\big[(T_m^{S^c})^2\big] + \sum_{i\in S}\frac{\mathrm{E}[\nu_i]\,\mathrm{E}[v_i^2]}{2}\left(\frac{\mathrm{E}[T_i^{S^c}]}{\mathrm{E}[v_i]} - \frac{\mathrm{E}[(T_i^{S^c})^2]}{\mathrm{E}[v_i^2]}\right), \tag{23}$$

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:

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

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

Problem	Performance measure	Performance space	Algorithm
$\max_{u\in\mathcal{U}} V_{u,\alpha}^{(R,C)}$	$x_j^u(\alpha) = \int_0^\infty \mathrm{E}_u[I_j(t)]e^{-\alpha t}\,dt$	$\mathcal{B}_c(A_\alpha, b_\alpha)^a$ A_α, $b_\alpha(\cdot)$: see (19), (20)	$(\hat{R}_\alpha, A_\alpha) \overset{\mathcal{AG}}{\longmapsto} \gamma(\alpha)$ $\hat{R}_\alpha$: see (21)
$\min_{u\in\mathcal{U}} V_u^{(0,C)}$ (finite busy period case)	$\theta_j^u = \int_0^\infty \mathrm{E}_u[I_j(t)]t\,dt]$	$\mathcal{B}(A, b)^b$ A, $b(\cdot)$: see (22), (23)	$(\hat{C}, A) \overset{\mathcal{AG}}{\longmapsto} \gamma$ $\hat{C}$: see (26)
	$W_j^u = \int_0^\infty \mathrm{E}_u[Q_j(t)]\,dt]$	$\mathcal{B}_c(A', b')$ A', $b'(\cdot)$: see (24), (25)	$(C, A') \overset{\mathcal{AG}}{\longmapsto} \gamma$

Table 2: Modelling Branching Bandit Problems.

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

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

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 for branching bandits W^u satisfies the following generalized conservation laws:

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

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

$$A_i'^S = \mathrm{E}[T_i^S], \quad i \in S, \tag{24}$$

$$b'(S) = b(E) - b(S^c) + 2\sum_{i\in S} h_i \mathrm{E}[T_j^S], \tag{25}$$

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

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

and $\mathrm{Diag}(x)$ denotes a diagonal matrix with diagonal entries corresponding to vector x.

Objective function:

$$\min_u V_u^{(0,C)} = \sum_{i\in E} \hat{C}_i\theta_i^u + 2\sum_{i\in E} C_i h_i = \sum_{i\in E} \frac{1}{\mathrm{E}[v_i]}\left(C_i - \sum_{j\in E} \mathrm{E}[N_{ij}]C_j\right)\theta_i^u + 2\sum_{i\in E} C_i h_i \tag{26}$$

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 $\mathcal{AG}$.

2.5 Applications

How in general do we show that the performance space is an extended polymatroid? From Theorem 4 it is sufficient to check that generalized conservation laws hold. In addition, if the problem can be

System	Criterion	Indexability	Performance Space
Batch of jobs	LC[a]	Smith (1956): D[b] Rothkopf (1966b)	Queyranne (1993): D, P[c] Bertsimas & Niño-Mora (1993): P
	DC[d]	Rothkopf (1966a): D Gittins & Jones (1974)	Bertsimas & Niño-Mora (1993): P
Batch of jobs with out-tree prec. constraints	LC	Horn (1972): D Meilijson & Weiss (1977)	Bertsimas & Niño-Mora (1993): EP[e]
	DC	Glazebrook (1976)	Bertsimas & Niño-Mora (1993): EP
Multiclass $M/G/1$	LC	Cox & Smith (1961)	Coffman & Mitrani (1980): P Gelenbe & Mitrani (1980): P
	DC	Harrison (1975a, 1975b)	Bertsimas & Niño-Mora (1993): EP
Multiclass [f] $M/G/c$	LC	Federgruen & Groenevelt (1988b) Shanthikumar & Yao (1992)	Federgruen & Groenevelt (1988b): P Shanthikhumar & Yao (1992): P
Multiclass $G/M/c$	LC	Federgruen & Groenevelt (1988a) Shanthikumar & Yao (1992)	Federgruen & Groenevelt (1988a): P Shanthikumar & Yao (1992): P
Multiclass Jackson network [g]	LC	Ross & Yao (1989)	Ross & Yao (1989): P
Multiclass $M/G/1$ with feedback	LC	Klimov (1974)	Tsoucas (1991): EP
	DC	Tcha & Pliska (1977)	Bertsimas & Niño-Mora (1993): EP
Multi-armed bandits	DC	Gittins & Jones (1974)	Bertsimas & Niño-Mora (1993): EP
Branching bandits	LC	Meilijson & Weiss (1977)	Bertsimas & Niño-Mora (1993): EP
	DC	Weiss (1988)	Bertsimas & Niño-Mora (1993): 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)

modeled as a branching bandit problem (with or without discounts), have taxes and/or rewards, the performance space is an extended polymatroid. 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 and dynamic 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 & Niño-Mora (1993).

What are the implications of characterizing the performance space as an extended polymatroid?

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 (1985)).

2. An understanding of whether or not the indices decompose. For example, in the classical multi-armed bandits problem Theorem 3 applies and therefore, the indices decompose, while

in the general branching bandits formulation they do not.

3. A unified and practical approach to sensitivity analysis of indexable systems, based on the well understood sensitivity analysis of linear programming (see Bertsimas and Niño-Mora (1993)).

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 (1980) and Weber (1992) for the indices of the classical multi-armed bandit problem.

5. The realization that the algorithm of Klimov for multiclass queues and the algorithm of Gittins for multi-armed 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 (1992) that the objective function value of the classical multi-armed bandit problem is submodular) and b) to show that the indices for some stochastic and dynamic optimization 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 and dynamic optimization and is able to address in a unified way their variations such as: discounted versus undiscounted cost criterion, rewards versus taxes, preemption versus non-preemption, discrete versus continuous time, work conserving versus idling policies, linear versus nonlinear objective functions.

3 Optimization of Polling Systems

Polling systems, in which a single server in a multiclass queueing system serves several classes of customers incurring switch-over 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 almost exclusively 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 (1990) and Takagi (1986), (1988).

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 $x_i^{(2)}$ respectively. 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 switch-over 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 $\rho = \sum_{i=1}^{N} \rho_i$. It is well known (see for example Takagi (1986)) that the system is stable if and only if $\rho < 1$. Note that this condition does not depend on the switch-over times.

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 non-preemptive, non-anticipative and stable policies. Within $\mathcal{U}$ we further distinguish between static ($\mathcal{U}_{static}$) and dynamic ($\mathcal{U}_{dynamic}$) policies. Static policies at each decision epoch 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. For example the policy under which the server visits the stations in a predetermined order according to a routing table is a static policy. 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 switch-over 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 (1988) and Reiman and Wein (1994) address the optimization problem over dynamic policies for polling systems with two stations. Boxma et. al. (1990) propose heuristic polling table strategies. In this section we review the work in Bertsimas and Xu (1993) 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.

3.1 Lower bounds on achievable performance

We develop in this section lower bounds on the weighted mean waiting time for polling systems non-preemptive, non-anticipative and stable policies. We call these policies admissible. We first focus on the class of static policies $\mathcal{U}_{static}$, in which 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.

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

$$\min_{u \in \mathcal{U}_{static}} E[W] = \frac{1}{\lambda} \sum_{i=1}^{N} c_i \lambda_i E[W_i]. \tag{27}$$

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

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

$$E[W] \geq \max \left\{ \frac{1}{\lambda} \left[\sum_{i=1}^{N} \frac{\lambda_i x_i^{(2)}}{2} \right] \left[\sum_{i=1}^{N} \frac{c_i \lambda_i}{(1-\sigma_{i-1})(1-\sigma_i)} \right], \; z_{static} \right\}, \tag{28}$$

where $\sigma_i = \sum_{j=1}^{i} \rho_j$ *and* z_{static} *is the solution of the following convex programming problem:*

$$z_{static} = \min \frac{1}{2\lambda} \sum_{i=1}^{N} \frac{c_i \lambda_i^2 x_i^{(2)}}{1-\rho_i} + \frac{1}{2\lambda} \sum_{i=1}^{N} \frac{c_i \lambda_i (1-\rho_i)}{(\sum_{j=1}^{N} m_{ji})} \tag{29}$$

subject to

$$\sum_{j=1}^{N} m_{ij} - \sum_{k=1}^{N} m_{ki} = 0, \qquad i = 1, 2, \ldots, N \tag{30}$$

$$\sum_{i,j=1}^{N} d_{ij} m_{ij} \leq 1 - \rho \tag{31}$$

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

$$m_{ii} = 0 \qquad \forall \; i,$$

where m_{ij} $(i, j = 1, 2, \ldots, N)$ *have the interpretation of the average number of visits per unit time by the server from station* i *to station* 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 (30) represents conservation of flow, while (31) is the stability condition. These constraints are still valid even under dynamic policies. The fact that the objective function in (29) represents a lower bound for the optimal solution value, only holds for static policies. Notice that while the feasible space of (29) 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 bounds of the previous theorem, the flow conservation constraints (30) are relaxed to obtain a closed form formula for the lower bounds.

Theorem 7 *a) For a polling system, the weighted mean delay for all static and admissible policies is bounded from below by:*

$$E[W] \geq z_{closed} = \max\left\{\frac{1}{\lambda}\left[\sum_{i=1}^{N}\frac{\lambda_i x_i^{(2)}}{2}\right]\left[\sum_{i=1}^{N}\frac{c_i\lambda_i}{(1-\sigma_{i-1})(1-\sigma_i)}\right],\right.$$
$$\left.\frac{1}{2\lambda}\sum_{i=1}^{N}\frac{c_i\lambda_i^2 x_i^{(2)}}{1-\rho_i}+\frac{(\sum_{i=1}^{N}\sqrt{c_i\lambda_i(1-\rho_i)d_i^*})^2}{2\lambda(1-\rho)}\right\}, \tag{32}$$

where $\sigma_i = \sum_{j=1}^{i}\rho_j$ *and* $d_i^* = d_{j(i),i} = \min_j\{d_{ji}\}$.

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

$$E[W] \geq z_{closed}^{hom} = \frac{\lambda x^2}{2(1-\rho)}+\frac{(\sum_{i=1}^{N}\sqrt{\lambda_i(1-\rho_i)d_i^*})^2}{2\lambda(1-\rho)}. \tag{33}$$

3.2 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 $e_{ij} = m_{ij}/\sum_{k,l} m_{kl}$ be the ratio of switch-overs from station i to station j over all switch-overs in the system. $E = [e_{ij}]$ is the corresponding switch-over 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 switch-overs 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 (1988) 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 $p_{ij} = e_{ij}/\sum_{k=1}^{N} e_{ik}$. $P = [p_{ij}]$ is the corresponding switch-over 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. (1990) 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 switch-overs from station i to station j in an optimal routing table. $H = [h_{ij}]$ is the switch-over 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 switch-overs in the periodic sequence. Moreover, $e_{ij} \sum_{k,l} h_{kl}$ is the desired number of switch-overs 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 switch-overs h_{ij} from station i to station j in the optimal routing table and the desired number of switch-overs determined by $e_{ij} \sum_{k,l} h_{kl}$. i.e.,

$$\min_{h}\{\max_{i,j}\{|h_{ij} - e_{ij} \sum_{k,l} h_{kl}|\}\}. \tag{34}$$

In addition, the flow conservation at each station requires that

$$\sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, \qquad i = 1, 2, \ldots, N, \tag{35}$$

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

$$\sum_{i \in G,\ j \in \bar{G}} h_{ij} \geq 1, \ \ \forall\, G \subset I,\ G \neq \phi. \tag{36}$$

In summary, the problem becomes

$$\begin{array}{lll} (P_{Eulerian}) & \min_h\{\max_{i,j}\{|h_{ij} - e_{ij} \sum_{k,l} h_{kl}|\}\} & \\ \text{subject to}: & \sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, & i = 1, 2, \ldots, N \\ & \sum_{i \in G,\ j \in \bar{G}} h_{ij} \geq 1, & \forall\, G \subset I,\ G \neq \phi \\ & h_{ij} \geq 0,\ \textit{integer}, & i, j = 1, 2, \ldots, N \end{array} \tag{37}$$

Equation (37) 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 (36). But if equation (36) is relaxed, $h_{ij} = 0,\ \forall\, i, j$ will be a feasible solution and will minimize the objective function in (37). In order to exclude this infeasible solution to (37),

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:

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

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

$$\begin{array}{lll} (P_{approx}) & \min\ y & \\ \text{subject to} & y - h_{ij} + e_{ij}\sum_{k,l} h_{kl} \ge 0, & i,j = 1,2,\ldots,N \\ & z + h_{ij} - e_{ij}\sum_{k,l} h_{kl} \ge 0, & i,j = 1,2,\ldots,N \\ & \sum_{j=1}^{N} h_{ij} - \sum_{k=1}^{N} h_{ki} = 0, & i = 1,2,\ldots,N \\ & N \le \sum_{i,j} h_{ij} \le L_{max} & \\ & h_{ij} \ge 0,\ integer, & i,j = 1,2,\ldots,N \end{array} \tag{38}$$

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.

3.3 Performance of proposed policies

Based on extensive simulation results Bertsimas and Xu (1993) 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.

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 class policies (Hofri and Ross (1988)).

4. Based on the intuition from the proof of the static lower bound and the numerical results, Bertsimas and Xu (1993) 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.

4 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. The fundamental optimization problem that arises in open networks is 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. The fundamental optimization problem that arises in a multiclass closed network is 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 (1992) 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 $\vec{n}(t) = (n_1(t), \ldots, n_R(t))$ 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 $\vec{n}(t)$ 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 $x \in X'$ 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* or, simply, the *achievable region*.

4.1 Sequencing of Multiclass Open Networks: Approximate Polyhedral Characterization

The traffic equations for our network model take the form

$$\lambda_r = \lambda_{0r} + \sum_{r'=1}^{R} \lambda_{r'} p_{r'r}, \qquad r = 1, \ldots, R. \tag{39}$$

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

$$R^S(t) = \sum_{r \in S} f_S(r) n_r(t), \tag{40}$$

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

$$\mu_r \left[\sum_{r' \in S} p_{rr'}(f(r) - f(r')) + \sum_{r' \notin S} p_{rr'} f(r) \right] \tag{41}$$

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 (1992) 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 and dynamic optimization problems.

Theorem 8 *For any set S of classes, for any choice of the* f-parameters *satisfying the restriction (41), and for any policy satisfying Assumption A, the following inequality holds:*

$$\sum_{r \in S} \lambda_r f(r) x_r \geq \frac{N'(S)}{D'(S)} \tag{42}$$

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 $\vec{n}(t)$ is right-continuous.

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

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

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 (41). 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 (41). 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 (41) and the relation (valid in steady-state) $E[1\{B_r(\tau_k)\}] = \lambda_r/\mu_r$, we finally obtain (42). □

Remarks : In order to apply Theorem 8, we must choose some *f-parameters* that satisfy (41). 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 (41). For a set S of classes, (41) 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 (41), in order to explicitly determine the *f-parameters*, it suffices to select nonnegative constants f_i, for each station i with $C_i \cap S$ non-empty. 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 $C_i \cap S$ non-empty. 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

$$f_S(r) = \frac{1}{\mu_r} + \sum_{r' \in S} p_{rr'} f_S(r'), \qquad r \in S. \tag{43}$$

Moreover this choice of the *f-parameters* causes the denominator of (42) 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 (1993). We use again a function of the form

$$R(t) = \sum_{r=1}^{R} f(r) n_r(t) \tag{44}$$

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

$$I_{rr'} = E[1\{B_r(\tau_k)\} n_{r'}(\tau_k)] \tag{45}$$

and

$$N_{ir'} = E[1\{B_{0i}(\tau_k)\} n_{r'}(\tau_k)], \tag{46}$$

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:*

$$2\mu_r I_{rr} - 2\sum_{r'=1}^{R} \mu_{r'} p_{r'r} I_{r'r} - 2\lambda_{0r}\lambda_r x_r = \lambda_{0r} + \lambda_r(1-p_{rr}) + \sum_{r'\neq r} \lambda_{r'} p_{r'r} \qquad r = 1,\ldots,R \tag{47}$$

and

$$\mu_r I_{rr'} + \mu_{r'} I_{r'r} - \sum_{w=1}^{R} \mu_w p_{wr} I_{wr'} - \sum_{w=1}^{R} \mu_w p_{wr'} I_{wr} - \lambda_{0r}\lambda_{r'} x_{r'} - \lambda_{0r'}\lambda_r x_r = -\lambda_r p_{rr'} - \lambda_{r'} p_{r'r} \qquad \forall r, r' \text{ such that } r > r'. \tag{48}$$

$$\sum_{r\in C_i} I_{rr'} + N_{ir'} = \lambda_{r'} x_{r'} \tag{49}$$

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

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

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

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

$$\begin{aligned} 2\sum_{r=1}^{R}\mu_r\left[\sum_{r'=1}^{R} p_{rr'}(f(r)-f(r')) + p_{r0}f(r)\right] E[1\{B_r(\tau_k)\}R(\tau_k)] - 2\sum_{r=1}^{R}\lambda_{0r}f(r)E[R(\tau_k)] \\ = \sum_{r=1}^{R}\lambda_{0r}f^2(r) + \sum_{r=1}^{R}\lambda_r\left[\sum_{r'=1}^{R} p_{rr'}(f(r)-f(r'))^2 + p_{r0}f^2(r)\right] \end{aligned} \tag{50}$$

Moreover, it is seen from (44) and (45) that

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

Let us define the vector $f = (f(1),\ldots,f(R))$. We note that both sides of (50) are quadratic functions of f. In particular, (50) can be written in the form

$$f^T Q f = f^T Q_0 f, \tag{51}$$

for some symmetric matrices Q, Q_0. Since (51) is valid for all choices of f, we must have $Q = Q_0$. It only remains to carry out the algebra needed in order to determine the entries of the matrices Q and Q_0. From (50), equality of the rth diagonal entries of Q and Q_0 yields (47), and equality of the off-diagonal terms yields (48). 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 (49). □

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

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

as long as the indicated expectations exist. By rewriting the previous equality in terms of the instantaneous transition rate matrix for the Markov chain $\vec{n}(t)$, and by introducing some new variables, we obtain certain relations between the variables. In particular, (47) can be derived by considering test functions of the form $g(\vec{n}(t)) = n_r^2(t)$, while (48) can be derived by considering the test functions of the form $g(\vec{n}(t)) = n_r(t)n_{r'}(t)$. 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.

4.2 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 (1994) obtain exactly the polyhedron (extended polymatroid) outlined in the previous section. Surprisingly, 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 for at least two reasons: a) It makes it very easy to solve indexable systems with side constraints, using linear programming techniques (see Bertsimas et. al. (1994)), b) It has been conjectured in mathematical programming theory 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 somewhat surprising that a purely stochastic method (potential function method) proves a rather nonobvious result in mathematical programming. This is an example of the very positive synergies that can result from the interaction of stochastic and deterministic methods.

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 non-preemptive policy satisfies the following formula that describes the evolution of the system:

$$N_i(\tau_{k+1}) = N_i(\tau_k) + \sum_{j=0}^{R} \chi_j(\tau_k)(N_{ji} - \delta_{ij}), \tag{52}$$

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

$$\sum_{j=0}^{R} \rho_j^+ E[N_{ji}] = \rho_i^+, \qquad i = 1, \ldots, R, \tag{53}$$

which is obtained by taking expectations of both sides of (52) with respect to the stationary distribution. Moreover,

$$\sum_{i=0}^{R} \rho_i^+ = 1, \tag{54}$$

which follows from the definition of ρ_i^+.

By considering the potential function

$$R(t) = \sum_{i \in E} f_i N_i(t) \tag{55}$$

and applying the evolution equation (52) 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. (1994)).

More interestingly, consider the auxiliary variables

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

Let 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 (n^+, 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. (1994) 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.

4.3 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 4.1. 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 (45) and (46), 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 4.1 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 (49), 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.

4.4 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

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

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 4.1, where there are no routing decisions to be made. We apply the nonparametric method by deriving an expression for $E[R^3(\tau_{k+1}) \mid \vec{n}(\tau_k)]$ and then collecting terms; alternatively, we can use test functions $g(\vec{n}(t)) = n_r(t)n_j(t)n_k(t)$. (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,

$$H_{rjk} = E[1\{B_r(\tau_k)\}n_j(\tau_k)n_k(\tau_k)] \tag{56}$$

and

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

The recursion for $E[R^3(\tau_{k+1}) \mid \vec{n}(\tau_k)]$ 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 $E[R^i(\tau_{k+1}) \mid \vec{n}(\tau_k)]$ 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 $E[R^i(\tau_{k+1}) \mid \vec{n}(\tau_k)]$, 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 (1992) have used the symbolic manipulation program Maple to write down the recursion for $E[R^i(\tau_{k+1}) \mid \vec{n}(\tau_k)]$, and generate equality constraints by collecting the coefficients of each monomial in the f–parameters.

Nonlinear Interactions

In several examples, we found that although the method provides relatively tight bounds, it does not exactly characterize the achievable region and there is a gap between the lower bound and the performance of an optimal policy. We believe that the reason is that the achievable region is not always a polyhedron. We will therefore discuss how to extend our methods so as to allow the possibility of nonlinear characterizations.

Let $\vec{Y}$ be a vector of random variables and let Q be a symmetric positive semidefinite matrix. Clearly,

$$E[(\vec{Y} - E[\vec{Y}])^T Q(\vec{Y} - E[\vec{Y}])] \geq 0,$$

which implies that

$$E[\vec{Y}^T Q\vec{Y}] \geq E[\vec{Y}^T]QE[\vec{Y}]. \tag{57}$$

Notice that (57) holds for every symmetric semidefinite matrix Q. By selecting particular values for matrices Q, one obtains a family of inequalities. For example, consider the model of Section 4.1 and fix some r. Let $\vec{Y}$ be the vector with components $(1\{B_r(\tau_k)\}n_j(\tau_k))$, $j = 1, \ldots, R$ and use the

identity $1\{B_r(\tau_k)\} = (1\{B_r(\tau_k)\})^2$, to obtain the quadratic inequalities

$$\sum_{i,j} H_{rij} Q_{ij} \geq \sum_{i,j} Q_{ij} I_{ri} I_{rj}, \ r = 1, \ldots, R, \tag{58}$$

where I_{ri} and H_{rij} have been defined in (45) and (56).

Any choice of Q leads to a new set of quadratic inequalities. We will actually impose the constraints of the form (58) for all choices of Q. Bertsimas et. al. (1992) show how to solve the problem with the nonlinear inequalities as a semidefinite-programming problem.

Higher order nonlinear constraints can be also obtained by using inequalities such as

$$E[1\{B_r(\tau_k)\} n_j^h(\tau_k)] \geq E[n_j(\tau_k)\}]^h, \ h = 1, 2, \ldots$$

which again follow from Jensen's inequality. We thus obtain a sequence of progressively more complicated convex sets that approximate the achievable region.

4.5 Performance of the bounds

In this section we summarize the principal insights form the extensive computational results reported in Bertsimas et. al. (1992) 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 (1992).

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 and dynamic optimization problem as follows:

1. Given a stochastic and dynamic optimization problem, first write the equations that describe the evolution of the system (see for example (52)).
2. Define a potential function of the type (55) 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.

5 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 (1991)).

We will describe the problem in terms of a telephone network, which is represented by a complete directed graph $G = (V, A)$ with $N = |V|$ nodes and $|A| = N(N-1)$ 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 obtain an approximate characterization of the achievable space using the potential function method and thus obtain an upper bound on the expected reward. We report results obtained in Bertsimas and Cryssikou (1994). Kelly (1993) has developed an approach based on nonlinear optimization to obtain bounds. We first present approximate and exact characterizations for the case of a single link and an approximate characterization for the case of a full network.

5.1 Single Link

In order to illustrate the methodology we present the simpler problem of one link and two classes of arriving calls. 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 $E[n_i]$ is the expected value of the number of calls of type i. Let τ_k be the time at which the k-th event (arrival or service completion) occurs. Let $\vec{n}(\tau_k)$ denote the state of the system after the kth transition. We will be using the notation $1\{\cdot\}$ to denote the indicator function. Finally, $A_i(\tau_k)$ ($\bar{A}_i(\tau_k)$) denotes the decision $(0,1)$ of whether a call of type i is accepted (rejected) at time τ_k . By applying the potential function nonparametric method to the potential function

$$R(t) = \sum_{i=1}^{2} f_i n_i(t)$$

we obtain

Theorem 11 *The performance space for the single link loss system is contained in the following polyhedron:*

$$\lambda_i \, E[\, 1\{A_i(\tau_k)\}\,] - \mu_i \, E[\, 1\{n_i(\tau_k) \geq 1\}\,] \;=\; 0, \quad i = 1,2$$

$$\lambda_i \, E[\, n_i(\tau_k)\, 1\{A_i(\tau_k)\}\,] - \mu_i \, E[\, n_i(\tau_k)\,] + \mu_i \, E[\, 1\{n_i(\tau_k) \geq 1\}\,] = 0, \quad i = 1,2$$

$$\lambda_1 E[n_2(\tau_k) 1\{A_1(\tau_k)\}] + \lambda_2 E[n_1(\tau_k) 1\{A_2(\tau_k)\}] = \mu_1 E[n_2(\tau_k) 1\{n_1(\tau_k) \geq 1\}] + \mu_2 E[n_1(\tau_k) 1\{n_2(\tau_k) \geq 1\}].$$

$$E[n_1(\tau_k)] + E[n_2(\tau_k)] \leq C$$

$$E[(n_1(\tau_k) + n_2(\tau_k)) 1\{n_i(\tau_k) \geq 1\}] \leq C E[1\{n_i(\tau_k) \geq 1\}], \quad i = 1,2$$

$$E[(n_1(\tau_k) + n_2(\tau_k)) 1\{n_i(\tau_k) \geq 1\}] \leq E[(n_1(\tau_k) + n_2(\tau_k))], \quad i = 1,2$$

$$0 \leq E[1\{n_i(\tau_k) \geq 1\}] \leq 1, \; E[n_i(\tau_k)] \geq E[1\{n_i(\tau_k) \geq 1\}] \geq 0, \quad i = 1,2.$$

Maximizing the linear function $w_1 E[n_1(\tau_k)] + w_2 E[n_2(\tau_k)]$ provides a bound on the optimal reward.

The previous characterization used only $O(1)$ variables. We can obtain an exact formulation in this case by considering $O(C)$ variables

$$E[1\{n_1(\tau_k) = k, n_2(\tau_k) = l, A_i(\tau_k)\}].$$

This characterization is equivalent to modeling the problem as a Markov decision process. Clearly this approach leads to an exact characterization, but its extension to the network case is problematic because of the very large number of variables. Kelly (1993) has also presented an exact method based on a closed form nonlinear formula to solve the single link problem.

5.2 Network

Let $n_{ij}^r(t)$ be the number of calls of type (i,j) routed through path r at time t under the stationary distribution. We consider *Markovian* policies, in the sense that a decision is only based on the current state. It is then clear that under a Markovian policy, the loss network evolves as a continuous-time Markov chain. Also $1\{A_{ij}^r\}(t)$ denotes the $(0,1)$ decision of whether an incoming call (i,j) is routed through route $r \in S(i,j)$ at time t. Let $E[n_{ij}^r] = E[n_{ij}^r(t)]$, $E[1\{A_{ij}^r\}(t)] = E[1\{A_{ij}^r\}]$ where the expectation is taken under the stationary distribution. By introducing the potential function $R(t) = \sum_{(i,j),r} f_{ij}^r n_{ij}^r(t)$ Bertsimas and Cryssikou (1994) prove the following.

Theorem 12 *The optimal expected reward is bounded above by the value attained by the following linear problem*

$$\textit{maximize} \quad \sum_{i,j} \sum_{r \in S(i,j)} w_{ij} E[n_{ij}^r]$$

subject to

$$\lambda_{ij} E[1\{A_{ij}^r\}] - \mu_{ij} E[1\{n_{ij}^r \geq 1\}] = 0 \qquad \forall\, r \in S(i,j) \;\;,\;\; \forall\,(i,j)$$

$$\lambda_{ij} E[n_{ij}^r\, 1\{A_{ij}^r\}] - \mu_{ij} E[n_{ij}^r] + \mu_{ij} E[1\{n_{ij}^r \geq 1\}] = 0 \qquad \forall\, r \in S(i,j) \;\;,\;\; \forall\,(i,j)$$

$$\lambda_{ij} E[n_{kl}^q\, 1\{A_{ij}^r\}] + \lambda_{kl} E[n_{ij}^r\, 1\{A_{kl}^q\}] = \mu_{ij} E[n_{kl}^q\, 1\{n_{ij}^r \geq 1\}] +$$

$$\mu_{kl} E[n_{ij}^r\, 1\{n_{kl}^q \geq 1\}] \;\; \forall (i,j),(k,l),\; r \in S(i,j),\; q \in S(k,l)$$

$$\sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r] \leq C_{ij} \qquad \forall (i,j)$$

$$\sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r 1\{n_{st}^q \geq 1\}] \leq C_{ij} E[1\{n_{st}^q \geq 1\}] \qquad \forall (i,j),(s,t),r$$

$$\sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r 1\{n_{st}^q \geq 1\}] \leq \sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r] \qquad \forall (i,j),(s,t),r$$

$$\sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r 1\{A_{st}^q\}] \leq C_{ij} E[1\{A_{st}^q\}] \qquad \forall (i,j),(s,t),r,$$

$$\sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r 1\{A_{st}^q\}] \leq \sum_{r \in S(k,l):(i,j) \in r} E[n_{kl}^r] \qquad \forall (i,j),(s,t),r,$$

$$E[n_{ij}^r] \geq E[1\{n_{ij}^r \geq 1\}],$$

$$E[1\{n_{ij}^r \geq 1\}] \leq 1.$$

$$E[1\{A_{st}^q\}] \leq 1,$$

all variables are nonnegative.

Compared with the variables we introduced, Kelly (1993) considers as variables:

$$x_{ij} = n_{ij}^{(i,j)}, \; y_{ij} = \sum_{r \neq (i,j)} n_{ij}^r,$$

The linear program of Theorem 12 has $O(N^2R^2)$ variables and constraints, where N is the number of demands (i,j) and R is the number of alternative paths. In contrast Kelly's approach uses $O(N)$ variables and $O(A)$ number of constraints where A is the number of links.

Based on very limited computational experience, the bounds of Theorem 12 are stronger compared with the ones derived in Kelly (1993), (based on nonlinear optimization) for sparser asymmetric networks, while the bounds in Kelly (1993) are better for complete symmetric networks. An attractive alternative seems to combine both bounds, as they seem to capture somewhat different behavior.

By introducing variables of the type

$$E[1\{n_{ij}^r = k_1\}, 1\{n_{pq}^s = k_2\}, 1\{A_{st}^f\}],$$

we can obtain tighter bounds at the expense of higher computational needs. By continuing in this manner, we can recapture the exact formulation of the problem as a Markov decision process. In this way the method produces a series of relaxations which are progressively closer to the exact formulation.

6 An optimal control approach to dynamic optimization

In this section we introduce a method of constructing near optimal policies based on an optimal control approach. We will introduce the method by addressing the following multiclass open network with two types (not classes) of customers, depicted in Figure 5.

Type 1 customers visit stations 1 and 2, in that order, before exiting the network and type 2 customers visit only station 1 before exiting the network. We define *class 1,2* customers to be type 1 customers at stations 1,2, respectively, and *class 3* customers to be type 2 customers at station 1. Let λ_1 and λ_2 be the (deterministic) arrival rates for customers of class 1 and 2, respectively. Let μ_1, μ_2 and μ_3 be the (deterministic) service rates of classes 1, 2 and 3 respectively. Let us define the traffic intensities: $\rho_1 = \frac{\lambda_1}{\mu_1}$, $\rho_2 = \frac{\lambda_1}{\mu_2}$, $\rho_3 = \frac{\lambda_2}{\mu_3}$. In order to ensure that at least one stable policy exists, we assume that $\rho_1 + \rho_3 < 1$ and $\rho_2 < 1$. Let c_i be the holding cost of class i, $i = 1, 2, 3$.

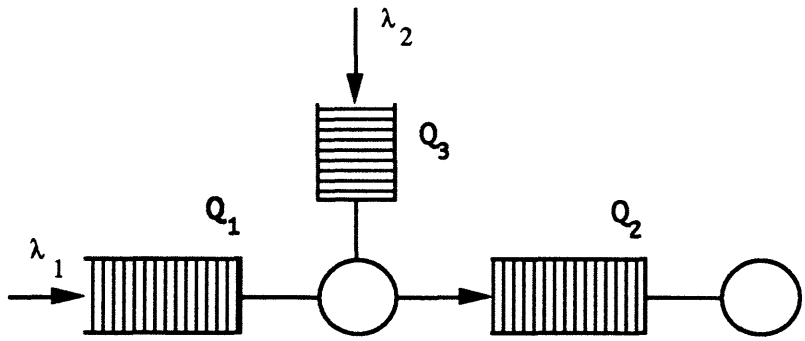

Figure 5: A three class open queueing network.

Let us notice that we have assumed that there is no randomness in the system. Let x_i be the initial number of customers for class i. Let $x_i^u(t)$ be the number of customers for class i at time t if we follow a policy u. Note that for this problem, a policy amounts to a rule according to which the first server can decide which customer class, if any, to serve. We are interested in a scheduling policy that minimizes the total holding cost

$$\min_u \int_0^\infty \sum_{i=1}^3 c_i x_i^u(t) dt.$$

Notice that although the integral extends to infinity, under a stable policy there is a finite time at which the network empties. In order to formulate this problem we define $u_i(t)$ be the effort that class i receives at time t, which is a number in $[0, 1]$. Then the problem can be formulated as follows:

$$\min \int_0^\infty (c_1 x_1(t) + c_2 x_2(t) + c_3 x_3(t)) dt. \tag{59}$$

subject to

$$\dot{x}_1(t) = \lambda_1 - \mu_1 u_1(t)$$

$$\dot{x}_2(t) = \mu_1 u_1(t) - \mu_2 u_3(t)$$

$$\dot{x}_3(t) = \lambda_2 - \mu_3 u_3(t)$$

$$u_1(t) + u_3(t) \leq 1$$

$$u_2(t) \leq 1$$

$$x_i(t), u_i(t) \geq 0.$$

The approach we outline in this section is based on the thesis that the *essential difficulty* in stochastic and dynamic optimization is primarily the dynamic and combinatorial character of the problem and only secondarily the stochastic character of the problem. Therefore, a fluid approximation of the system is expected to capture the qualitative character of an optimal policy. In general,

given an open multiclass network, the general problem can be formulated as follows:

$$(CLP)\ \min \int_0^\infty \sum_i c_i x_i(t) dt$$

subject to

$$\dot{x}(t) = Au(t) + b$$

$$Du(t) \leq e$$

$$u(t) \geq 0$$

$$x(t) \geq 0,$$

where the matrix A and vector b are easily found from the given parameters, the matrix D depends on the topology of the network and the vectors $x(t)$ and $u(t)$ are the queue lengths and the controls at time t. The above optimization problem is known in the literature as *continuous linear programming* (see for example Anderson (1987)).

Optimization of fluid models as approximations of stochastic and dynamic optimization problems have been introduced in Bellman (1957). Anderson (1978) introduced this particular model for dynamic scheduling problems while Chen and Yao (1989) and Atkins and Chen (1993) propose myopic strategies. Regarding algorithms for the general problem it is fair to say that there are no efficient algorithms for the problem. Perhaps the most efficient one is due to Pullan (1993), who also contains a nice survey of the area.

We next review some recent work of Avram and Bertsimas (1994) to solve instances of (LCP). We start with the system in Figure 5. Applying Pontryagin's maximum principle, Avram and Bertsimas (1994) show that the following policy is optimal for problem (59).

Theorem 13 *There are the following three cases:*

1. *If* $c_1\mu_1 < c_3\mu_3$, *give absolute priority to class 3* ($u_3(t) = 1$).
2. *If* $c_3\mu_3 < c_1\mu_1 < c_3\mu_3 + c_2\mu_2$ *give priority to class 3 if*

$$x_2 > x_1 \frac{\mu_1 - \lambda_1}{\mu_2 - \mu_1} \frac{c_1\mu_1 - c_3\mu_3}{c_2\mu_2};$$

   give priority to class 1 otherwise (see Figure 6).

3. *If* $c_3\mu_3 + c_2\mu_2 < c_1\mu_1$, *give absolute priority to class 1.*

The above policy which is consistent with the heavy traffic policy in Harrison and Wein (1989) generalizes the well known $c\mu$ rule and establishes formally the optimality of thresholds policies. What insights do we obtain about the structure of optimal policies in the stochastic case, for example when arrivals are Poisson and services are exponential? We conjecture that cases 1 and 3 (absolute priority rules) will be identical in the stochastic case; however, in case 2 we conjecture that there

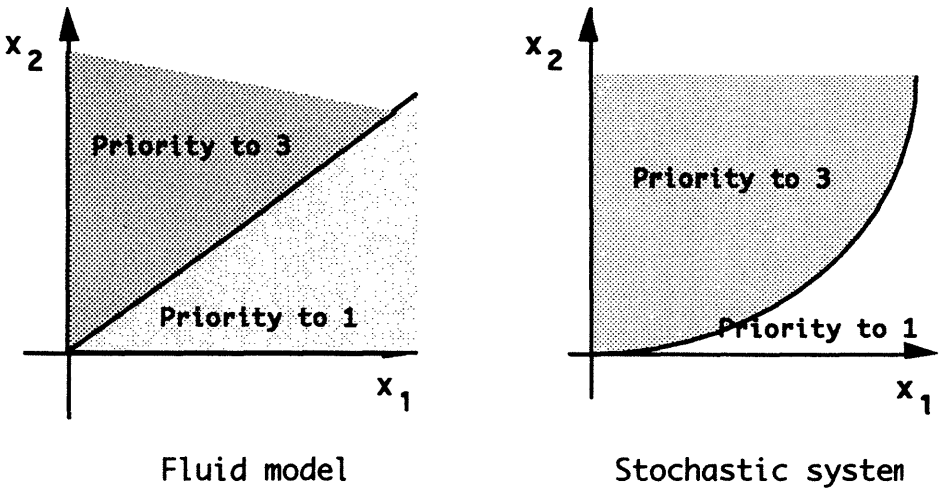

Figure 6: Threshold optimal policies for the Example of Figure 5.

is a threshold curve which is nonlinear (see Figure 6). In other words, we believe that the optimal policy for the fluid model gives significant qualitative insights for the optimal policy.

Other evidence of the close connection between the stochastic and deterministic models include the Klimov model (see Section 2), where the control approach reproduces Klimov's indexing policy. Moreover, Avram and Bertsimas (1994) include further examples.

In general, for an open multiclass queueing network with n classes, the control approach gives rise to a partition of the space of $(x_1, \ldots, x_n)$ into polygonal regions, in which an absolute priority policy is optimal. The determination of these partitions is combinatorially explosive, but conceptually simple. While, to analytically characterize the partition becomes very difficult very fast, a numerical approach would be highly desirable.

7 Concluding Remarks and Open Problems

We presented what we believe is a powerful and unified approach for stochastic and dynamic optimization problems based on mathematical programming. To summarize the method consists of the following steps:

1. We first define appropriate decision variables.

2. Using the potential function method or using particular properties of the 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 polling systems) using integer programming.

5. By approximating the stochastic and dynamic system by a deterministic but still dynamic model, we reduce the problem to a linear control problem, which we then solve by linear programming methods. Simulating the policies obtained gives feasible policies to the stochastic systems. Comparing the performance of these policies to the bounds obtained, we obtain a guarantee of the suboptimallity of the proposed policies.

We finally want to emphasize that the proposed approach gives insights to the complexity of stochastic optimization problems. We have seen that for indexable systems, we can obtain an exact formulation by the first order relaxation obtained from the potential function method. The order of the relaxation provides information regarding the complexity of the problem, i.e., the higher the order of the relaxation needed to exactly characterize the region of achievable performance, the more difficult the problem is. Notice that this view of complexity is different, but we believe insightful, than the traditional one (Figure 1).

Regarding open questions we feel that the following list will give important insights to the field of stochastic and dynamic optimization:

1. Regarding the method of characterizing the performance space, it would be desirable to find systematic techniques to propose heuristic policies from relaxations (an example of this type is the heuristic in Section 3 for polling systems).

2. It would be desirable to 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 (1994)).

3. The potential function method is a systematic way to describe a series of relaxations for the region of achievable space in stochastic and dynamic optimization problems. Investigating the power of higher order relaxations and experimenting with non-polynomial (in particular piece wise linear) potential functions, will enhance the power of the method.

4. It would be desirable to find an efficient algorithm for problem (LCP) for a general multiclass queueing network problem of Section 6.

In closing, although the proposed approach illustrates how mathematical programming techniques can be used in applied probability, we have also seen an example of a reverse application: the potential function method provides a nontrivial reformulation of an extended polymatroid using quadratic number of variables and constraints. As a result, we hope that these results will be of interest to applied probabilists, as they provide new interpretations, proofs, algorithms, insights and

connections to important problems in stochastic optimization, as well as to discrete optimizers, since they reveal a new and important area of application.

Acknowledgments

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 and Michael Ricard and my colleagues Florin Avram of Northeastern University, John Tsitsiklis and Larry Wein of MIT for significantly contributing to my understanding of the area of stochastic and dynamic optimization.

References

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

[2] F. Avram, D. Bertsimas (1994), "A linear control approach to optimization of multiclass queueing networks", in preparation.

[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, (1994), "Bounds for loss networks", in preparation.

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

[7] D. Bertsimas, I. Paschalidis and J. Tsitsiklis, (1992), "Optimization of multiclass queueing networks: polyhedral and nonlinear characterizations of achievable performance", to appear in *Annals Applied Probability.*

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

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

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

[11] P. P. Bhattacharya, L. Georgiadis and P. Tsoucas, (1991), "Problems of adaptive optimization in multiclass $M/GI/1$ queues with Bernoulli feedback". Paper presented in part at the ORSA/TIMS *Conference on Applied Probability in the Engineering, Information and Natural Sciences*, January 9-11, 1991, Monterey, California.

[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] A. Cobham, (1954), "Priority assignment in waiting line problems", *Operations Research*, **2**, 70-76.

[16] E. G. Coffman, M. Hofri and G. Weiss, (1989), "Scheduling stochastic jobs with a two point distribution on two parallel machines", *Probability in the Engineering and Informational Sciences*, **3**, 89-116.

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

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

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

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

[21] 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.

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

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

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

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

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

[27] 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.

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

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

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

[31] 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.

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

[33] F. Kelly, (1992), "Bounds on the performance of dynamic routing schemes for highly connected networks", to appear in *Mathematics of Operations Research.*

[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", *Queueing systems and applications.*

[38] L. Lovãsz and A. Schrijver, (1990), "Cones of matrices and setfunctions and 0-1 optimization", *SIAM Jour. Opt.*, 166-190.

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

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

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

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

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

[44] M. Pullan, (1993), "An algorithm for a class of continous linear programs", *SIAM J. Control and Optimization*, 1558-1577.

[45] M. Reiman and L. Wein (1994), "Dynamic scheduling of a two-class queue with setups", in preparation.

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

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

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

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

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

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

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

[53] 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.

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

[55] J. N. Tsitsiklis, (1993), "A short proof of the Gittins index theorem", *Annals of Applied Probability*, to appear.

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

[57] 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.

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

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

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

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