MATHEMATICS OF OPERATIONS RESEARCH
Vol. 21, No 2, May 1996
Printed in U S.A

CONSERVATION LAWS, EXTENDED POLYMATROIDS AND MULTIARMED BANDIT PROBLEMS; A POLYHEDRAL APPROACH TO INDEXABLE SYSTEMS

DIMITRIS BERTSIMAS AND JOSÉ NIÑO-MORA

We show that if performance measures in stochastic and dynamic scheduling problems satisfy generalized conservation laws, then the feasible region of achievable performance is a polyhedron called an extended polymatroid, that generalizes the classical polymatroids introduced by Edmonds. Optimization of a linear objective over an extended polymatroid is solved by an adaptive greedy algorithm, which leads to an optimal solution having an indexability property (*indexable systems*). Under a certain condition the indices possess a stronger decomposition property (*decomposable systems*). The following problems can be analyzed using our theory: multiarmed bandit problems, branching bandits, scheduling of multiclass queues (with or without feedback), scheduling of a batch of jobs. Consequences of our results include: (1) a characterization of indexable systems as systems that satisfy generalized conservation laws, (2) a sufficient condition for indexable systems to be decomposable, (3) a new linear programming proof of the decomposability property of Gittins indices in multiarmed bandit problems, (4) an approach to sensitivity analysis of indexable systems, (5) a characterization of the indices of indexable systems as sums of dual variables, and an economic interpretation of the branching bandit indices in terms of retirement options, (6) an analysis of the indexability of undiscounted branching bandits, (7) a new algorithm to compute the indices of indexable systems (in particular Gittins indices), as fast as the fastest known algorithm, (8) a unification of Klimov's algorithm for multiclass queues and Gittins' algorithm for multiarmed bandits as special cases of the same algorithm, (9) a closed formula for the maximum reward of the multiarmed bandit problem, with a new proof of its submodularity and (10) an understanding of the invariance of the indices with respect to some parameters of the problem. Our approach provides a polyhedral treatment of several classical problems in stochastic and dynamic scheduling and is able to address variations such as: discounted versus undiscounted cost criterion, rewards versus taxes, discrete versus continuous time, and linear versus nonlinear objective functions.

1. Introduction. In the mathematical programming tradition researchers and practitioners solve optimization problems by defining decision variables and formulating constraints, thus describing the feasible space of decisions, and then applying algorithms for the solution of the underlying optimization problem. For the most part, the tradition for stochastic and dynamic scheduling problems has been, however, quite different, as it relies primarily on dynamic programming formulations. Using ingenious but often ad hoc methods, which exploit the structure of the particular problem, researchers and practitioners can sometimes derive insightful structural results that lead to efficient algorithms. In their comprehensive survey of deterministic scheduling problems Lawler, Lenstra, Rinnooy Kan and Shmoys (1989) end their paper with the following remarks: "The results in stochastic scheduling are scattered and they have been obtained through a considerable and sometimes disheartening

Received March 12, 1993; revised March 22, 1993; March 23, 1994; August 1, 1995.
AMS 1991 subject classification. Primary: 90B35; Secondary: 60K25, 90C27.
OR/MS Index 1978 subject classification. Primary: Production/Scheduling/Stochastic; Secondary: Programming/Combinatorial.
Key words. Stochastic scheduling, polymatroid, multiarmed bandit, multiclass queue.

0364-765X/96/2102/0257/$01.25

effort. In the words of Coffman, Hofri and Weiss (1989), there is great need for new mathematical techniques useful for simplifying the derivation of results."

Perhaps one of the most important successes in the area of stochastic scheduling in the last twenty years has been the solution of the *multiarmed bandit problem*, which can be described as follows:

The multiarmed bandit problem. There are K parallel projects, indexed $k = 1, \dots, K$. Project k can be in one of a finite number of states $j_k \in \mathscr{N}_k$. At each instant of discrete time $t = 0, 1, \dots$ one must work on a single project. If project k, which is in state $j_k(t)$ at time t, is engaged, then an immediate reward of $R^k_{j_k(t)}$ is earned. Rewards are additive and are discounted in time by a discount factor $0 < \beta < 1$. The state $j_k(t)$ changes to $j_k(t+1)$ according to a homogeneous Markov transition rule, with transition matrix $P^k = (p^k_{ij})_{i,j \in \mathscr{N}_k}$, while the states of the projects that have not been engaged remain unchanged. The problem is to find a *scheduling policy* that determines which project to engage at each time in order to maximize the total expected discounted reward earned over an infinite horizon.

The problem has numerous applications and a rather vast literature (see Gittins (1989) and the references therein). It was first solved by Gittins and Jones (1974), who established that to each project state one could attach an *index*, such that the optimal action at each time is to work on a project with largest current index. They also proved that these optimal indices satisfy a stronger *decomposition* property: The indices corresponding to the states of a given project only depend on characteristics of that project (states, rewards and transition probabilities), and not on those of other projects.

Since the original solution, which relied on an interchange argument, other proofs were proposed: Whittle (1980) provided a proof based on dynamic programming, which was subsequently simplified by Tsitsiklis (1986). Varaiya, Walrand and Buyukkoc (1985), and Weiss (1988) derived different proofs based on interchange arguments. Weber (1992) presented an intuitive proof. More recently, Tsitsiklis (1994) has provided a proof based on a simple inductive argument, and Ishikida and Varaiya (1994) have derived the result without making an explicit use of the interchange argument.

The multiarmed bandit problem is a special case of a *dynamic and stochastic job scheduling problem*. In this setting, a set of jobs, which are classified in a finite set $\mathscr{N}$ of classes, must be scheduled for service in the system. The goal is to optimize a function of a performance measure by means of an *admissible* scheduling policy.

DEFINITION 1 (INDEXABLE PROBLEMS). We say that a dynamic and stochastic scheduling problem is *indexable* if a policy of the following kind is optimal: to each job type i we attach an index γ_i; at each decision epoch, work on a job with largest current index.

In general the optimal indices γ_i (as functions of the model parameters) could depend on characteristics of the entire set $\mathscr{N}$ of job classes. Consider a partition of set $\mathscr{N}$ into subsets $\mathscr{N}_k$, for $k = 1, \dots K$. Job classes in $\mathscr{N}_k$ may be interpreted as corresponding to an underlying project class k. In some cases, the optimal indices of job classes in $\mathscr{N}_k$ depend only on characteristics of job classes in $\mathscr{N}_k$, and not on the entire set $\mathscr{N}$. Such a property is useful computationally, since it enables the problem to be decomposed into smaller subproblems, for which the computation of the indices can be done separately. As we have mentioned, the multiarmed bandit problem exhibits this *decomposition* property, which motivates the following definition:

DEFINITION 2 (DECOMPOSABLE PROBLEMS). An indexable problem is called *decomposable* if the indices of jobs corresponding to a project depend only on characteristics of that project.

TABLE 1
Indexable Problems and their Performance Regions

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

[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)

In addition to the multiarmed bandit problem, a variety of dynamic and stochastic scheduling problems has been solved in the last decades by indexing rules (see Table 1 for examples).

Faced with these results, one asks what is the underlying *reason* that these nontrivial problems have very efficient solutions, both in theory and in practice. 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 general way to address stochastic and dynamic scheduling problems that will lead to a deeper understanding of their structural properties?* This is the set of questions that motivates the present work.

In the last decade the following approach has been proposed to address special cases of these questions: In broad terms, researchers try to describe the feasible space of a stochastic and dynamic scheduling problem as a polyhedron; then, the stochastic and dynamic scheduling problem is translated into an optimization problem over the corresponding polyhedron, which can then be attacked by traditional mathematical programming methods. Coffman and Mitrani (1980) and Gelenbe and Mitrani (1980) first showed using *conservation laws* that the performance region of a multiclass 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 job classes have the same routing probabilities, and the same service requirements at each station of the network (see 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 region is necessarily a polymatroid. They also proved that, when the cost is linear on the performance, the optimal policy is a *static* priority rule (see Cobham (1954), and 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). Finally, Bertsimas, Paschalidis and Tsitsiklis (1994a) generalize the ideas of conservation laws to general multiclass queueing networks using potential function ideas. They find linear and nonlinear inequalities that the feasible region satisfies. Optimization over this set of constraints gives bounds on achievable performance.

Our goal in this paper is to propose a theory of conservation laws, and to establish that the strong structural properties in the optimization of a class of stochastic and dynamic scheduling problems, that include the multiarmed bandit problem and its extensions, follow from corresponding strong structural properties of the underlying polyhedra that characterize their regions of achievable performance.

By generalizing the work of Shanthikumar and Yao (1992) we show that if performance measures in stochastic and dynamic scheduling problems satisfy *generalized conservation laws*, then the feasible space of achievable performance is a polyhedron called an *extended polymatroid.* Optimization of a linear objective over an extended polymatroid is solved by an *adaptive greedy* algorithm, which leads to an optimal solution having an *indexability* property. Special cases of our theory are summarized in Table 1. Consequences of our results include:

1. A characterization of indexable systems as systems that satisfy generalized conservation laws.
2. Sufficient conditions for indexable systems to be decomposable.
3. A new, algebraic proof (based on the strong duality theory of linear programming as opposed to dynamic programming formulations) of the decomposability property of Gittins indices in multiarmed bandit problems.
4. An approach to sensitivity analysis of indexable systems, based on the well understood sensitivity analysis of linear programming.
5. A new characterization of the indices of indexable systems as sums of dual variables corresponding to a linear program over their region of achievable performance.
6. An economic interpretation of the indices in the context of branching bandits in terms of retirement options, thus generalizing the interpretation of Whittle (1980) and Weber (1992) for the indices of the classical multiarmed bandit problem.
7. A new algorithm to compute the indices of indexable systems (in particular Gittins indices), which is as fast as the fastest known algorithm (Varaiya, Walrand and Buyukkoc (1985)).
8. The realization that Klimov's algorithm for multiclass queues and Gittins's algorithm for multiarmed bandits are special cases of the same algorithm.
9. Closed formulae for the performance of the optimal policy. This also leads to an understanding of the invariance of the indices with respect to some parameters of the stochastic scheduling problem.
10. A closed formula for the maximum reward of the multiarmed bandit problem, with a new proof of its submodularity.
11. An approach to formulate and solve several classical problems in stochastic scheduling, and their variations, such as: discounted versus undiscounted cost criterion, rewards versus taxes, discrete versus continuous time, linear versus nonlinear performance objectives.

The paper is structured as follows: Section 2 introduces the main ideas of the polyhedral approach to stochastic scheduling through a simple example: a two-armed bandit problem. Section 3 develops the theory of polyhedra (extended polymatroid) that arise as the performance region of indexable scheduling problems. Section 4 presents a general polyhedral framework for formulating and solving indexable scheduling problems as linear programs with special structure. Section 5 applies that framework to the model of branching bandits, while Section 6 specializes this result to several classical problems in stochastic scheduling, including the multiarmed bandit problem. The final section contains some thoughts on the field of optimization of stochastic systems.

2. Optimal dynamic scheduling of a two-armed bandit.

This section introduces the main ideas of the mathematical programming approach to stochastic scheduling, as outlined in §1, through an example: a simple two-armed bandit problem.

Let us first describe the example problem. Although it may be regarded as a two-armed bandit problem (see e.g. Gittins (1989)), we shall introduce and analyze it as a scheduling problem in a multiclass queue, for consistency with the general framework to be presented later. In the terminology of queueing theory, the problem may be stated as the optimal dynamic scheduling of the closed queue in discrete time with three job classes and two jobs shown in Figure 1. Initially one of the jobs is in class 1 and the other in class 2. Jobs may change class after completing service, according to Markovian transition probabilities. A class-2 job may thus either remain in the same class, with probability p_{22}, or transfer to class 3, with probability $p_{23} = 1 - p_{22}$. A class-3 job behaves analogously, as shown in Figure 1. The class 1 job, however, always remains in the same class. There is a discounted reward structure associated with service completions: Each time a class-j job completes its service, a reward r_j is earned, discounted in time by a discount factor $0 < \beta < 1$. The problem is to find a scheduling policy that decides which job to serve at each time, in order to maximize the expected present value of the rewards earned over an infinite horizon.

Two natural requirements are imposed on the class of scheduling policies that may be used. First, a policy must be nonanticipative, i.e., decisions may not be based on future information on the evolution of the system. Second, a policy has to be nonidling, i.e., the server never stops working while there are jobs in the system. We shall refer to the class $\mathscr{U}$ of policies that satisfy these two conditions as the class of *admissible* policies. By defining the indicator

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

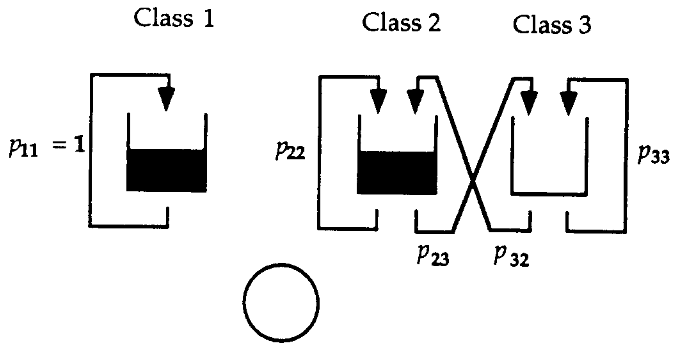

FIGURE 1. A two-armed bandit.

we can write this optimal dynamic scheduling problem as

$$\text{(SP)} \qquad Z = \max\left\{ E_u\left[\sum_{t=0}^{\infty} \sum_{j=1}^{3} r_j I_j(t) \beta^t \right] : u \in \mathcal{U} \right\}.$$

2.1. *Performance measures.* The first step of the approach involves expressing the objective to be optimized as a function of suitably defined performance measures. For the above problem, a natural performance measure is the total expected discounted number of service completions for each job class,

$$\lambda_j^u = E_u\left[\sum_{t=0}^{\infty} I_j(t) \beta^t \right], \quad \text{for } j = 1, 2, 3.$$

As scheduling policies range over the class of admissible policies, the corresponding performance vectors span the region of achievable performance

$$\Lambda = \{(\lambda_1^u, \lambda_2^u, \lambda_3^u) : u \in \mathcal{U}\}.$$

The problem of finding an optimal performance vector can now be written as the mathematical program

$$\begin{array}{ll} \text{(MP)} \qquad & Z = \max \ r_1\lambda_1 + r_2\lambda_2 + r_3\lambda_3 \\ & \quad \text{subject to} \\ & \qquad (\lambda_1, \lambda_2, \lambda_3) \in \Lambda. \end{array}$$

2.2. *Conservation laws.* We shall construct a complete polyhedral description of performance region Λ, by identifying conservation laws satisfied by the system and expressing them as linear constraints on performance vectors. First, since at each time exactly one job completes its service, it follows that the total expected discounted number of completed jobs is the same under any admissible scheduling policy. This conservation law may be written as

$$(1) \qquad \lambda_1^u + \lambda_2^u + \lambda_3^u = \frac{1}{1-\beta}, \quad \text{for } u \in \mathcal{U}.$$

We shall outline next how to construct a family of conservation laws associated with subsets of job classes. Consider, for example, the subset {2} corresponding to class-2 jobs. Let u_2 be a scheduling policy that gives priority to class-2 jobs over jobs in other classes (i.e., the job in service must be a class-2 job whenever there is one present). Under such a policy, the conservation law

$$(2) \qquad \lambda_2^{u_2} = \frac{1}{1 - \beta p_{22}} + \beta p_{32} \frac{1}{1 - \beta p_{22}} \lambda_3^{u_2}$$

holds. Equation (2) expresses the intuitive fact that the total expected discounted number of class-2 jobs completed, $\lambda_2^{u_2}$, can be decomposed into two terms: A first constant term, $1/(1 - \beta p_{22})$, that accounts for the class-2 jobs completed until the job that was initially in class 2 transfers for the first time to class 3; and a second term, that accounts for the class-2 jobs completed afterwards, using the following accounting argument: After a class 3 job completes its service at time t, the expected discounted number of successive class-2 job completions is

$$\beta^{t+1} p_{32}/(1 - \beta p_{22}).$$

It should be intuitively clear that, under other admissible policies, the right-hand side of Eq. (2) overestimates the expected discounted number of successive class-2 job completions, i.e.,

$$\lambda_2^u \le \frac{1}{1-\beta p_{22}} + \beta p_{32}\frac{1}{1-\beta p_{22}}\lambda_3^u, \quad \text{for } u \in \mathscr{U}.$$

Using conservation law (1), we may rewrite the previous inequality as

$$\lambda_1^u + \left(1 + \beta_1\frac{p_{32}}{1-\beta p_{22}}\right)\lambda_3^u \ge \frac{1}{1-\beta} - \frac{1}{1-\beta p_{22}}, \quad \text{for } u \in \mathscr{U},$$

with equality holding under any policy u that gives priority to class-2 jobs.

By applying a similar argument to policies that give priority to other subsets of job classes, we may derive corresponding conservation laws and linear constraints on performance vectors.

2.3. *Linear programming formulation.* As will be shown later, the linear constraints obtained through the procedure outlined above represent a complete polyhedral description of performance region Λ. Proceeding in this way, we obtain the following explicit linear programming formulation of problem (MP):

$$\begin{array}{llllll}
Z = \max & r_1\lambda_1 + & r_2\lambda_2 & + & r_3\lambda_3 & & \\
\text{subject to} & & \lambda_2 & + & \lambda_3 & \ge & 0 \\
 & \lambda_1 + & & & \left(1 + \beta_1\frac{p_{32}}{1-\beta p_{22}}\right)\lambda_3 & \ge & \frac{1}{1-\beta} - \frac{1}{1-\beta p_{22}} \\
 & \lambda_1 + & \left(1 + \beta_1\frac{p_{23}}{1-\beta p_{33}}\right)\lambda_2 & & & \ge & \frac{1}{1-\beta} \\
 & \lambda_1 & & & & \ge & 0 \\
 & & \lambda_2 & & & \ge & 0 \\
 & & & & \lambda_3 & \ge & 0 \\
 & \lambda_1 + & \lambda_2 & + & \lambda_3 & = & \frac{1}{1-\beta},
\end{array}$$

where each inequality constraint holds with equality under a scheduling policy that gives priority to the corresponding subset of job classes.

It will be shown in §5 that the feasible region of linear program (MP), which is called an extended contra-polymatroid, has particular structure, that will be examined in the next section.

2.4. *Optimal solution.* We will show in the next section that the extreme points of the polyhedron that is the feasible space of linear program (MP) are the performance vectors achieved under static-priority scheduling policies (i.e., the service priority of a class remains constant through time). Since the optimal value of a linear program is attained by some extreme point in its feasible region, it follows that static-priority policies are optimal for this scheduling problem. In the next section we will also see

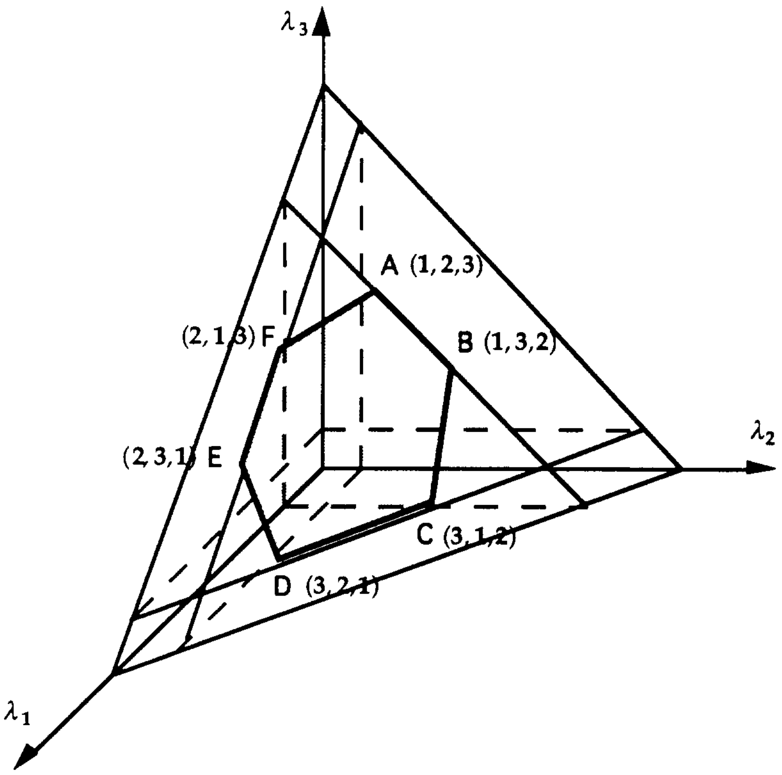

FIGURE 2. An extended contra-polymatroidal performance region in dimension three (priority policies are associated with its extreme points, with job classes ranked by increasing priority).

how to identify the optimal extreme point, and its associated optimal static-priority policy, by running a one-pass algorithm.

Figure 2 depicts an extended polymatroid in dimension three, representing the performance region of a 3-class bandit problem. Notice that it has $3! = 6$ extreme points, corresponding to all static-priority policies.

3. Extended polymatroids.

Extended polymatroids are polyhedra whose role in the field of stochastic scheduling is analogous to that played by classical polymatroids (see Edmonds (1970)) in combinatorial optimization. Polymatroids arise as the convex hull of feasible solutions in combinatorial optimization problems solved by a greedy algorithm, such as that of finding the minimum spanning tree in a graph. Similarly, extended polymatroids appear as the convex hull of performance vectors achievable under admissible scheduling policies (i.e., as the performance region) in stochastic scheduling problems solved by priority-index policies, including multiarmed bandit problems. Problems such as finding a minimum spanning tree or scheduling optimally a multiarmed bandit can thus be formulated as linear programs over polymatroids or extended polymatroids, respectively. These linear programs possess strong structural and algorithmic properties, which explain the optimality of greedy-like solution schemes for the problems they represent. They further provide insight on how to solve variations on the original problems, such as incorporating a nonlinear objective function, or imposing side constraints.

In the remainder of this section we establish the notation to be used, and present the formal definition of extended polymatroid. Section 3.1 develops the theory of linear programs over such polyhedra.

Let $\mathcal{N} = \{1, \dots, n\}$ be a finite set. Let x denote a real n-vector, with components x_i, for $i \in \mathcal{N}$. For $S \subset N$, let $S^c = \mathcal{N} \setminus S$, and let $|S|$ denote the cardinality of S. Let x_S be the subvector of x corresponding to components in S, i.e., $x_S = (x_i)_{i \in S}$. Let $2^{\mathcal{N}}$ denote the class of all subsets of $\mathcal{N}$. Let b: $2^{\mathcal{N}} \to \Re_+$ be a nonnegative set function that satisfies $b(\emptyset) = 0$. Let $A = (A_i^S)_{i \in \mathcal{N}, S \subseteq \mathcal{N}}$ be a matrix that satisfies

$$A_i^S > 0, \quad \text{for } i \in S \text{ and } S \subseteq \mathcal{N}.$$

Given a permutation $\pi = (\pi_1, \dots, \pi_n)$ of $\mathcal{N}$, and a vector $x = (x_1, \dots, x_n)'$ we write $x_\pi = (x_{\pi_1}, \dots, x_{\pi_n})'$, and

$$b_\pi = \left(b(\{\pi_1, \dots, \pi_n\}), \dots, b(\{\pi_{n-1}, \pi_n\}), b(\{\pi_n\})\right)'.$$

Let A_π be the upper triangular submatrix of A given by

$$A_\pi = \begin{pmatrix} A_{\pi_1}^{\{\pi_1, \dots, \pi_n\}} & \cdots & A_{\pi_{n-1}}^{\{\pi_1, \dots, \pi_n\}} & A_{\pi_n}^{\{\pi_1, \dots, \pi_n\}} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & A_{\pi_{n-1}}^{\{\pi_{n-1}, \pi_n\}} & A_{\pi_n}^{\{\pi_{n-1}, \pi_n\}} \\ 0 & \cdots & 0 & A_{\pi_n}^{\{\pi_n\}} \end{pmatrix}.$$

Let $v(\pi)$ denote the unique solution of triangular system

$$\begin{array}{lcccccl} A_{\pi_1}^{\{\pi_1, \dots, \pi_n\}} x_{\pi_1} & + \cdots + & A_{\pi_{n-1}}^{\{\pi_1, \dots, \pi_n\}} x_{\pi_{n-1}} & + & A_{\pi_n}^{\{\pi_1, \dots, \pi_n\}} x_{\pi_n} & = & b(\{\pi_1, \dots, \pi_n\}) \\ & \ddots & \vdots & & \vdots & & \vdots \\ & & A_{\pi_{n-1}}^{\{\pi_{n-1}, \pi_n\}} x_{\pi_{n-1}} & + & A_{\pi_n}^{\{\pi_{n-1}, \pi_n\}} x_{\pi_n} & = & b(\{\pi_{n-1}, \pi_n\}) \\ & & & & A_{\pi_n}^{\{\pi_n\}} x_{\pi_n} & = & b(\{\pi_n\}). \end{array} \tag{3}$$

Consider now the following polyhedra associated with matrix A and set function b:

$$\mathcal{P}_c(A, b) = \left\{x \in \Re_+^n \colon \sum_{i \in S} A_i^S x_i \geq b(S), \text{ for } S \subset \mathcal{N} \text{ and } \sum_{i \in \mathcal{N}} A_i^{\mathcal{N}} x_i = b(\mathcal{N})\right\} \quad \text{and}$$

$$\mathcal{P}(A, b) = \left\{x \in \Re_+^n \colon \sum_{i \in S} A_i^S x_i \leq b(S), \text{ for } S \subset \mathcal{N} \text{ and } \sum_{i \in \mathcal{N}} A_i^{\mathcal{N}} x_i = b(\mathcal{N})\right\}.$$

Polyhedra $\mathcal{P}_c(A, b)$ and $\mathcal{P}(A, b)$ possess, under the following consistency assumption, strong structural and algorithmic properties, that generalize those of classical polymatroids.

ASSUMPTION 1. *For every permutation π of $\mathcal{N}$, $v(\pi) \in \mathcal{P}(A, b)$.*

DEFINITION 3 (Extended Polymatroid). We say that polyhedron $\mathcal{P}(A, b)$ is an *extended polymatroid* with ground set $\mathcal{N}$, if Assumption 1 holds.

If Assumption 1 holds for polyhedron $\mathcal{P}_c(A, b)$, we say that $\mathcal{P}_c(A, b)$ is an *extended contra-polymatroid.*

REMARK. Extended polymatroids were introduced by Tsoucas (1991), who characterized the region of achievable performance in Klimov's problem (see Klimov (1974)) as a polyhedron with special structure, not previously identified in the literature. He established that the optimality of static-priority policies under linear performance objectives, first proven by Klimov (1974) using dynamic programming arguments, follows from structural properties of such polyhedra. Bhattacharya, Georgiadis and Tsoucas (1992) called this polyhedron an *extended polymatroid*, and further developed its properties.

3.1. *Linear programming over extended polymatroids.* In this section we develop structural properties of linear programs over extended polymatroids. First, we present a new duality proof that such a linear program is solved by a one-pass adaptive greedy algorithm. It is then shown that its optimal solutions are characterized by certain *allocation indices*, defined as sums of optimal dual variables. Finally, we identify a condition under which these indices exhibit a decomposition property, which simplifies their computation. The significance of these results in the field of stochastic scheduling is that they explain corresponding properties of indexable scheduling problems, as the next section will demonstrate.

Let us thus consider the problem of maximizing a given linear reward objective $\sum_{i \in \mathcal{N}} r_i x_i$ over extended contra-polymatroid $\mathscr{P}_c(A, b)$,

$$
\begin{aligned}
(\mathrm{LP}_c) \qquad Z = \max \ & \sum_{i \in \mathcal{N}} r_i x_i \\
\text{subject to} \quad & \sum_{i \in S} A_i^S x_i \geq b(S), \quad \text{for } S \subseteq \mathcal{N} \\
& \sum_{i \in \mathcal{N}} A_i^{\mathcal{N}} x_i = b(\mathcal{N}) \\
& x_i \geq 0, \quad \text{for } i \in \mathcal{N}.
\end{aligned}
$$

```
Input: $(r, A)$.
Output: $(\pi, \bar{y}, \gamma)$, where $\pi = (\pi_1, \dots, \pi_n)$ is a permutation of $\mathcal{N}$, $\bar{y} = (\bar{y}^S)_{S \subseteq \mathcal{N}}$ and $\gamma = (\gamma_1, \dots, \gamma_n)$.

Step 0. Set $S_1 = \mathcal{N}$; set $\bar{y}^{S_1} = \max\left\{ \frac{r_i}{A_i^{\mathcal{N}}} : i \in S_1 \right\}$;
    pick $\pi_1 \in \operatorname{argmax}\left\{ \frac{r_i}{A_i^{\mathcal{N}}} : i \in S_1 \right\}$;
    set $\gamma_{\pi_1} = \bar{y}^{S_1}$.

Step k. For $k = 2, \dots, n$:
    set $S_k = S_{k-1} \setminus \{\pi_{k-1}\}$; set $\bar{y}^{S_k} = \max\left\{ \frac{r_i - \sum_{j=1}^{k-1} A_i^{S_j} \bar{y}^{S_j}}{A_i^{S_k}} : i \in S_k \right\}$;
    pick $\pi_k \in \operatorname{argmax}\left\{ \frac{r_i - \sum_{j=1}^{k-1} A_i^{S_j} \bar{y}^{S_j}}{A_i^{S_k}} : i \in S_k \right\}$;
    set $\gamma_{\pi_k} = \gamma_{\pi_{k-1}} + \bar{y}^{S_k}$.

Step n. For $S \subseteq \mathcal{N}$: set
        $\bar{y}^S = 0, \quad \text{if } S \notin \{S_1, \dots, S_n\}$.
```

FIGURE 3. Adaptive greedy algorithm $\mathscr{AG}$.

Since $\mathscr{P}_c(A, b)$ is a nonempty bounded polyhedron, linear program (LP$_c$) must have a finite optimal solution. Therefore, its dual program—by strong linear programming duality—will have the same optimum value Z. We shall have a dual variable y^S for every $S \subseteq \mathcal{N}$. The dual program of (LP$_c$) is

$$(\mathrm{LD}_c) \qquad Z = \min \sum_{S \subset \mathcal{N}} b(S) y^S$$

$$\text{subject to} \quad \sum_{S: \mathcal{N} \supseteq S \ni i} A_i^S y^S \geq r_i, \quad \text{for } i \in \mathcal{N}$$

$$y^S \leq 0, \quad \text{for } S \subset \mathcal{N}$$

$$y^{\mathcal{N}} \text{ unrestricted}.$$

3.2. *Adaptive greedy algorithm.* We present next a one-pass adaptive greedy algorithm for solving linear program (LP$_c$) and its dual (LD$_c$). The input data for the algorithm consists of reward vector $r = (r_i)_{i \in \mathcal{N}}$, and an *oracle* (see Grötschel, Lovász and Schrijver (1988)) that produces the value A_i^S when called with input (i, S). Its output includes a ranking permutation π of ground set $\mathcal{N}$, an optimal dual solution $\bar{y}$, and optimal allocation indices $\{\gamma_i\}_{i \in \mathcal{N}}$ that, as will be seen, characterize optimal solutions to (LP$_c$).

As shown in Figure 3, the algorithm constructs in n steps a vector $\bar{y} = (\bar{y}^S)_{S \subseteq \mathcal{N}}$, which will later be proved to be an optimal solution to program (LD$_c$). This dual solution has at most n nonzero components, which correspond to a nested (laminar) family (see e.g. Schrijver (1986))

$$S_n \subset \cdots \subset S_2 \subset S_1 = \mathcal{N}.$$

The algorithm identifies in an adaptive greedy manner a permutation $\pi = (\pi_1, \ldots, \pi_n)$ of $\mathcal{N}$ such that, defining $S_k = \{\pi_k, \ldots, \pi_n\}$ for $k = 1, \ldots, n$, the unique solution of the triangular system

$$\begin{array}{lll}
A_{\pi_1}^{S_1} y^{S_1} & & = r_{\pi_1} \\
\vdots \qquad \ddots & & \vdots \\
A_{\pi_{n-1}}^{S_1} y^{S_1} + \cdots + A_{\pi_{n-1}}^{S_{n-1}} y^{S_{n-1}} & & = r_{\pi_{n-1}} \\
A_{\pi_n}^{S_1} y^{S_1} + \cdots + A_{\pi_n}^{S_{n-1}} y^{S_{n-1}} & + A_{\pi_n}^{S_n} y^{S_n} & = r_{\pi_n}
\end{array}$$

satisfies

$$\bar{y}^{S_k} \leq 0, \quad \text{for } k = 2, \ldots, n.$$

A corresponding primal solution is obtained by complementary slackness, as the solution $x(\pi)$ of system (3). We present next an optimality proof, based on linear programming duality.

PROPOSITION 1 (OPTIMALITY OF ADAPTIVE GREEDY ALGORITHM). *Let $(\pi, \bar{y}, \gamma)$ be an output of algorithm $\mathscr{AG}$. Then $x(\pi)$ and $\bar{y}$ are an optimal primal-dual pair for linear programs* (LP$_c$) *and* (LD$_c$).

PROOF. We shall show that $x(\pi)$ and $\bar{y}$ are primal and dual feasible solutions, respectively, and that they satisfy complementary slackness. We shall first show by induction that $\bar{y}$ is dual feasible. By definition of $\bar{y}_1^S$ in $\mathscr{AG}$, it follows that

$$r_i - A_i^{S_1}\bar{y}^{S_1} \leq 0, \quad \text{for } i \in S_1.$$

Since $S_2 \subset S_1$, we must have $\bar{y}^{S_2} \leq 0$.

Similarly, for $k = 2, \dots, n$, we have, by definition of $\bar{y}^{S_k}$,

$$r_i - \sum_{j=1}^{k} A_i^{S_j}\bar{y}^{S_j} \leq 0, \quad \text{for } i \in S_k,$$

and since $S_{k+1} \subset S_k$, it follows that $\bar{y}^{S_{k+1}} \leq 0$. Therefore $\bar{y}^{S_j} \leq 0$, for $j = 2, \dots, n$, and by definition of $\bar{y}$, it follows that $\bar{y}^S \leq 0$, for $S \subset \mathscr{N}$.

We also have, by construction,

$$\sum_{S:\mathscr{N} \supseteq S \ni \pi_k} A_{\pi_k}^S \bar{y}^S = \sum_{j=1}^{k} A_{\pi_k}^{S_j}\bar{y}^{S_j} = r_{\pi_k}, \quad \text{for } k = 1, \dots, n.$$

Therefore $\bar{y}$ is a dual feasible solution.

Now, by definition of extended contra-polymatroid, $v(\pi)$ is a primal feasible solution. Let $\bar{x} = v(\pi)$. Let us show that $\bar{x}$ and $\bar{y}$ satisfy complementary slackness. Assume $\bar{y}^S \neq 0$ for some $S \subseteq \mathscr{N}$. Then, by construction it must be the case that $S = S_k = \{\pi_k, \dots, \pi_n\}$, for some $k \in \mathscr{N}$. Now, some $\bar{x}$ is the solution of system (3), it follows that

$$\sum_{i \in S} A_i^S \bar{x}_i = \sum_{j=k}^{n} A_{\pi_j}^{\{\pi_k, \dots, \pi_n\}} \bar{x}_{\pi_j} = b(S).$$

Therefore, by strong linear programming duality, $v(\pi)$ and $\bar{y}$ are an optimal primal-dual pair, which completes the proof. □

REMARKS.

1. The result that adaptive greedy algorithm $\mathscr{AG}$ solves program (LP$_c$) was first established by Tsoucas (1991), who provided a direct optimality proof—not involving duality theory. He showed that when the underlying extended polymatroid corresponds to the performance region in Klimov's (1974) queueing model, algorithm $\mathscr{AG}$ yields the classical Klimov's algorithm for computing the optimal priority scheduling policy.

2. The running time of algorithm $\mathscr{AG}$ is polynomial in the following sense: Given its input (r, A) the algorithm performs $O(n^3)$ multiplications and $O(n^2)$ pairwise comparisons. The number of required calls to the oracle that produces the A_i^S's is $O(n^2)$. Therefore, $\mathscr{AG}$ is an *oracle-polynomial-time* algorithm (see Grötschel et al. (1988)). In most applications the oracle that returns the A_i^S's runs in polynomial time in the size of the model data, in which case the algorithm is polynomial in the usual sense.

3. Recently, Bertsimas and Teo (1994) have developed a primal-dual approximation algorithm for linear and integer programming problems of the covering type. When specialized to the case of extended polymatroids, this algorithm coincides with adaptive greedy algorithm $\mathscr{AG}$.

The optimality of adaptive greedy algorithm allows us to characterize explicitly the vertices of an extended polymatroid.

THEOREM 1 (CHARACTERIZATION OF EXTREME POINTS). *The set of extreme points of extended contra-polymatroid $\mathscr{P}_c(A, b)$ is*

$$\{v(\pi) : \pi \text{ is a permutation of } \mathscr{N}\}.$$

PROOF. First, it is easy to see that for any permutation π of $\mathscr{N}$, $v(\pi)$ is an extreme point of $\mathscr{P}_c(A, b)$. Second, we shall show that any extreme point of $\mathscr{P}_c(A, b)$ is of the form $v(\pi)$ for some permutation π. This follows from the well known result that every extreme point of a polyhedron is the unique maximizer of some linear objective, and the fact that algorithm $\mathscr{AG}$ produces an optimal primal solution of such form. □

REMARK. Edmonds (1970) proved the classical result that the *greedy* algorithm solves the linear programming problem over a polyhedron for every linear objective function if and only if the polyhedron is a polymatroid. Now, in the case that $A_i^S = 1$, for $i \in S$ and $S \subseteq \mathscr{N}$, it is easy to see that adaptive greedy algorithm $\mathscr{AG}$ reduces indeed to the classical greedy algorithm that sorts the r_i's in nonincreasing order. Since we know that algorithm $\mathscr{AG}$ solves problem (LP$_c$) optimally it follows that, in this special case, $\mathscr{P}_c(A, b)$ is indeed a polymatroid, and therefore function b is submodular. Extended polymatroids are therefore natural generalizations of polymatroids, and adaptive greedy algorithm $\mathscr{AG}$ is a natural extension of the greedy algorithm.

3.3. *Indexability.* The optimality of adaptive greedy algorithm $\mathscr{AG}$ leads naturally to the definition of certain *allocation indices*, that characterize the optimal solutions of a linear programs over an extended polymatroid. We will show later that these allocation indices correspond to the well-known Gittins indices in stochastic scheduling problems.

DEFINITION 4 (ALLOCATION INDICES). Let $\bar{y}$ be the optimal dual solution produced by adaptive greedy algorithm $\mathscr{AG}$. Let

$$\gamma_i = \sum_{S : \mathscr{N} \supseteq S \ni i} \bar{y}^S, \quad \text{for } i \in \mathscr{N}.$$

We say that $\gamma_1, \ldots, \gamma_n$ are the *allocation indices* of linear program (LP$_c$).

REMARKS. 1. If permutation π is produced by algorithm $\mathscr{AG}$, then

$$\gamma_{\pi_i} = \bar{y}^{\{\pi_1, \ldots, \pi_n\}} + \cdots + \bar{y}^{\{\pi_i, \ldots, \pi_n\}}, \quad \text{for } i \in \mathscr{N}. \tag{4}$$

2. Notice that the allocation indices of linear program (LP$_c$) depend only on (r, A) (not on the right-hand side b).
3. Notice that in order for the allocation indices of linear program (LP$_c$) to be well defined, the optimal dual solution $\bar{y}$ computed by algorithm $\mathscr{AG}$ must be uniquely determined by its input (r, A).

Consistency of the definition of allocation indices. We address next the issue of whether the allocation indices $\gamma_1, \ldots, \gamma_n$ of linear program (LP$_c$) are indeed uniquely determined by the input (r, A) of algorithm $\mathscr{AG}$. This question arises because ties may occur in some of the maximizations performed by the algorithm. In the presence of ties, the permutation π produced by the algorithm is not uniquely determined by its input (r, A): it clearly depends on the way ties are broken. We shall establish next that the optimal dual solution produced by the algorithm remains indeed invariant under different tie-breaking rules, and therefore our definition of allocation indices of linear program (LP_c) is consistent. We shall also characterize the structure of the permu produced by algorithm $\mathscr{AG}$.

In order to prove that the dual solution produced by algorithm $\mathscr{AG}$ is uniquely determined by its input, we introduce next algorithm $\mathscr{AG}'$, which is simply an unambigous version of the former—its output is uniquely determined by its input. Algorithm $\mathscr{AG}'$, shown in Figure 4, produces a partition $P = \{P_1, \ldots, P_m\}$ of the ground set, in addition to a dual vector $\hat{y}$. Each subset in that partition groups together elements of the ground set that would attain the same maximum in the maximizations performed by algorithm $\mathscr{AG}$, thus eliminating the ambiguity due to different tie-breaking rules.

The next result, which is easily seen to hold by induction (see Niño-Mora (1995)), shows the relation between the outputs of both algorithms. It establishes that the dual solution returned by adaptive greedy $\mathscr{AG}$ is invariant under tie-breaking rules. It also characterizes the structure of the permutations that can be returned by algorithm $\mathscr{AG}$.

PROPOSITION 2. *Let $(\pi, \bar{y}, \gamma)$ and $(P, \hat{y})$ be outputs of algorithms $\mathscr{AG}$ and $\mathscr{AG}'$, respectively, corresponding to the same input (r, A). Then*
(a) $\hat{y} = \bar{y}$.
(b) *Permutation π satisfies*

$$(5) \qquad P_k = \{\pi_{|P_1 \cup \quad \cup P_{k-1}|+1}, \ldots, \pi_{|P_1 \cup \ldots \cup P_k|}\}, \quad \textit{for } k = 1, \ldots, m.$$

```
Input: $(r, A)$.
Output: $(P, \hat{y})$, where $P = \{P, \ldots, P_m\}$ is a partition of $\mathscr{N}$ and $\hat{y} = (\hat{y}^S)_{S \subseteq \mathscr{N}}$ Step 1.
Set $k := 1$; set $U_1 = \mathscr{N}$;
    set $\hat{y}^{U_1} = \max\left\{\frac{r_i}{A_i^{\mathscr{N}}} : i \in U_1\right\}$ and $P_1 = \operatorname{argmax}\left\{\frac{r_i}{A_i^{\mathscr{N}}} : i \in U_1\right\}$.
Step k. While $U_k \neq P_k$ do:
    begin
    Set $k := k + 1$; set $U_k = U_{k-1} \setminus P_{k-1}$;
    set $\hat{y}^{U_k} = \max\left\{\frac{r_i - \sum_{j=1}^{k-1} A_i^{U_j} \hat{y}^{U_j}}{A_i^{U_k}} : i \in U_k\right\}$ and
    $P_k = \operatorname{argmax}\left\{\frac{r_i - \sum_{j=1}^{k-1} A_i^{U_j} \hat{y}^{U_j}}{A_i^{U_k}} : i \in U_k\right\}$.
    end {while}
Step m. Set $m = k$;
    for $S \subseteq \mathscr{N}$: set
        $\hat{y}^S = 0$,    if $S \notin \{U_1, \ldots, U_m\}$.
```

FIGURE 4. Algorithm $\mathscr{AG}'$: unambiguous version of adaptive greedy algorithm $\mathscr{AG}$.

REMARK. Proposition 2(b) characterizes the structure of the ranking permutations π that can be produced by adaptive greedy algorithm $\mathscr{AG}$. Elements $\pi \in P_k$ in identity (5) would tie in the maximizations performed by algorithm $\mathscr{AG}$ for determining permutation π and can, therefore, be rearranged in any order, maintaining the optimality of primal solution $v(\pi)$.

Optimality conditions. We present next several equivalent optimality conditions for a linear program over an extended polymatroid, including one based on ranking its indices. Let $\Re_- = \{x \in \Re : x \leq 0\}$. Let $\gamma_1, \dots, \gamma_n$ be the allocation indices of program (LP$_c$). Let T be the following $n \times n$ upper triangular matrix:

$$T = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

PROPOSITION 3 (EQUIVALENT OPTIMALITY CONDITIONS). *The following statements are equivalent*:

(a) *Permutation π satisfies* (5);

(b) *permutation π is produced by algorithm $\mathscr{AG}$*;

(c) $r_\pi A_\pi^{-1} \in \Re \times \Re_-^{n-1}$, *and then the allocation indices are given by* $\gamma_\pi = r_\pi A_\pi^{-1} T$;

(d) $\gamma_{\pi_n} \leq \gamma_{\pi_{n-1}} \leq \cdots \leq \gamma_{\pi_1}$.

PROOF. (a) $\Rightarrow$ (b): This is the result in Proposition 2(b).

(b) $\Rightarrow$ (c): It is clear, by construction in $\mathscr{AG}$, that

$$\left(\bar{y}^{\{\pi_1, \dots, \pi_n\}}, \dots, \bar{y}^{\{\pi_{n-1}, \pi_n\}}, \bar{y}^{\{\pi_n\}}\right) = r_\pi A_\pi^{-1}. \tag{6}$$

Now, in the proof of Proposition 1 we showed that

$$\bar{y}^S \leq 0, \quad \text{for } S \subset \mathscr{N},$$

which together with (6) implies $r_\pi A_\pi^{-1} \in \Re \times \Re_-^{n-1}$. Furthermore, by (4) we have

$$\gamma_\pi = \left(\bar{y}^{\{\pi_1, \dots, \pi_n\}}, \dots, \bar{y}^{\{\pi_{n-1}, \pi_n\}}, \bar{y}^{\{\pi_n\}}\right) T,$$

and by (6) it follows that $\gamma_\pi = r_\pi A_\pi^{-1} T$.

(c) $\Rightarrow$ (d): By (c) we have

$$(\gamma_{\pi_1}, \gamma_{\pi_2} - \gamma_{\pi_1}, \dots, \gamma_{\pi_n} - \gamma_{\pi_{n-1}}) T^{-1} = r_\pi A_\pi^{-1} \in \Re \times \Re_-^{n-1},$$

whence the result follows.

(d) $\Rightarrow$ (a): By construction of $\hat{y}$ in algorithm $\mathscr{AG}'$, the fact that $\bar{y} = \hat{y}$ and the definition of allocation indices, it follows that

$$\gamma_i = \bar{y}^{U_1} + \cdots + \bar{y}^{U_k}, \quad \text{for } i \in U_k \text{ and } k = 1, \dots, m.$$

where the U_ks are as constructed in algorithm $\mathscr{AG}'$. Also, it is easy to see that $\bar{y}^{U_j} < 0$, for $j \geq 2$. These two facts imply that π must satisfy (5), which completes the proof. □

Combining the result that algorithm $\mathscr{AG}$ solves linear program (LP$_c$) optimally with the equivalent conditions in Proposition 3, we obtain the sufficient optimality conditions presented next.

THEOREM 2 (SUFFICIENT OPTIMALITY CONDITIONS AND INDEXABILITY. *Assume that any of the conditions* (a)–(d) *of Proposition* 3 *holds. Then* $v(\pi)$ *solves linear program* (LP_c) *optimally.*

The following results follow from our previous analysis:

1. **Indexability:** Optimality condition (d) of Proposition 3 shows that any permutation that sorts the allocation indices of linear program (LP_c) provides a corresponding optimal solution. Condition (d) therefore shows that this class of linear programs has an *indexability* property.

2. **Sensitivity analysis:** Optimality condition (c) of Proposition 3 is specially well suited for performing sensitivity analysis. Consider the following question: given a permutation π of $\mathcal{N}$, for what vectors r and matrices A does $v(\pi)$ solve problem (LP_c) optimally? We know that $v(\pi)$ is optimal for r and A that satisfy the condition

$$r_\pi A_\pi^{-1} \in \Re \times \Re_-^{n-1}.$$

We may also ask: For which permutations π is it guaranteed that $v(\pi)$ is optimal? By Proposition 3(d), we know that $v(\pi)$ is optimal for permutations π that satisfy

$$\gamma_{\pi_n} \le \gamma_{\pi_{n-1}} \le \cdots \le \gamma_{\pi_1},$$

thus providing an $O(n \log n)$ optimality test for π. Glazebrook (1994) has recently applied these polyhedral results to provide a range of index-based suboptimality bounds for general policies in a variety of stochastic scheduling problems (see also Glazebrook (1987)).

3. **Closed formulae for allocation indices:** Proposition 3(c) provides a closed formula for the vector of allocation indices. It shows that the indices are piecewise linear functions of the reward vector.

4. **Optimal objective value:** The optimal objective value, Z, is given by:

$$\begin{aligned} Z &= r_\pi x_\pi \\ &= r_\pi A_\pi^{-1} b_\pi \\ &= \gamma_\pi T^{-1} b_\pi \\ &= (\gamma_{\pi_1}, \gamma_{\pi_2}, \dots, \gamma_{\pi_n}) \begin{pmatrix} b(\{\pi_1, \dots, \pi_n\}) - b(\{\pi_2, \dots, \pi_n\}) \\ \vdots \\ b(\{\pi_{n-1}, \pi_n\}) - b(\{\pi_n\}) \\ b(\{\pi_n\}) \end{pmatrix}. \end{aligned} \tag{7}$$

3.4. *Index decomposition.* We show in this section that the allocation indices of linear program (LP_c) possess, under a certain assumption, a strong *decomposition* property. In that case, the indices can be computed by solving a number of smaller subproblems, thus reducing the amount of computations required. We will show later that this property explains an analogous decomposition property of the optimal priority-indices in some stochastic scheduling problems, including multiarmed bandits.

In this setting the ground set $\mathcal{N}$ is partitioned into subsets $\mathcal{N}_1, \dots, \mathcal{N}_K$. Let A^k denote the submatrix of A corresponding to subsets of $\mathcal{N}_k$, i.e., $A^k = (A_i^S)_{i \in \mathcal{N}_k, S \in \mathcal{F}_k}$. Let $r = (r_i)_{i \in \mathcal{N}}$, and $r^k = (r_i)_{i \in \mathcal{N}_k}$, for $k = 1, \dots, K$. Let γ be the vector of allocation indices of linear program (LP$_c$), and let γ^k be the index vector produced by adaptive greedy algorithm $\mathscr{AG}$ when fed with input (r^k, A^k).

The required decomposition assumption on the parameters of matrix A is:

ASSUMPTION 2.

$$A_i^S = A_i^{S \cap \mathscr{N}_k}, \quad \textit{for } i \in S \cap \mathscr{N}_k \quad \textit{and } S \subseteq \mathscr{N}.$$

We show next that, under Assumption 2, vector of allocation indices γ extends vectors $\gamma^1, \dots, \gamma^K$ over their respective ranges.

THEOREM 3 (INDEX DECOMPOSITION). *If Assumption 2 holds, then*

$$\gamma_i = \gamma_i^k, \quad \textit{for } i \in \mathscr{N}_k \quad \textit{and } k = 1, \dots, K.$$

PROOF. Let us define $b^k\colon 2^{\mathscr{N}_k} \to \Re_+$ by $b^k(S) = b(S)$, for $S \subseteq \mathscr{N}_k$. Since $\mathscr{P}_c(A, b)$ is an extended contra-polymatroid, it is easily seen that $\mathscr{P}_c(A^k, b^k)$ is also an extended contra-polymatroid—with ground set $\mathscr{N}_k$. For $k = 1, \dots, K$, let us write $x^k = (x_i^k)_{i \in \mathscr{N}_k}$, and let (LP$_k$) be the linear program

$$(\mathrm{LP}_k) \qquad \max\left\{ \sum_{i \in \mathscr{N}_k} r_i x_i^k \colon x^k \in \mathscr{P}_c(A^k, b^k) \right\}.$$

By definition, the allocation indices of linear program (LP$_k$) are obtained by running algorithm $\mathscr{AG}$ with input (r^k, A^k), and are therefore given by vector γ^k. Let us define

$$g_i = \gamma_i^k, \quad \text{for } i \in \mathscr{N}_k \quad \text{and } k = 1, \dots, K. \tag{8}$$

Let us renumber, for simplicity, the elements of $\mathscr{N}$, so that

$$g_n \le g_{n-1} \le \cdots \le g_1. \tag{9}$$

Let $\pi = (1, \dots, n)$. Permutation π of $\mathscr{N}$ induces permutations π^k of $\mathscr{N}_k$, for $k = 1, \dots, K$, that satisfy

$$\gamma^k_{\pi^k_{|\mathscr{N}_k|}} \le \cdots \le \gamma^k_{\pi^k_1}.$$

Hence, by Proposition 3 it follows that

$$\gamma^k_{\pi^k} = r^k_{\pi^k}\left(A^k_{\pi^k}\right)^{-1} T_k, \quad \text{for } k = 1, \dots, K$$

or, equivalently,

(10)

$$\left(\gamma^1_{\pi_1}, \gamma^2_{\pi_2}, \dots, \gamma^k_{\pi^k}\right) \begin{pmatrix} T_1^{-1} A^1_{\pi^1} & 0 & \cdots & 0 \\ 0 & T_2^{-1} A^2_{\pi^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_k^{-1} A^k_{\pi^k} \end{pmatrix} = \left(r^1_{\pi^1}, r^2_{\pi^2}, \dots, r^k_{\pi^k}\right),$$

where T_k is an $|\mathscr{N}_k| \times |\mathscr{N}_k|$ matrix with the same structure as matrix T, for $k = 1, \dots, K$.

On the other hand, we have

$$T^{-1}A_\pi = \begin{pmatrix} 1 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & -1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & 0 & 1 & -1 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix} A_\pi$$

$$= \begin{pmatrix} A_1^{\{1,\dots,n\}} & \cdots & A_{n-1}^{\{1,\dots,n\}} - A_{n-1}^{\{1,\dots,n-1\}} & A_n^{\{1,\dots,n\}} - A_n^{\{1,\dots,n-1\}} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & A_{n-1}^{\{n-1,n\}} & A_n^{\{n-1,n\}} - A_n^{\{n\}} \\ 0 & \cdots & 0 & A_n^{\{n\}} \end{pmatrix}.$$

Now, notice that if $i \in \mathscr{N}_k$, $j \in \mathscr{N} \setminus \mathscr{N}_k$ and $i > j$ then, by Assumption 2 it follows that

$$(11) \qquad A_i^{\{j,\dots,n\}} = A_i^{\{j,\dots,n\} \cap \mathscr{N}_k} = A_i^{\{j+1,\dots,n\} \cap \mathscr{N}_k} = A_i^{\{j+1,\dots,n\}}.$$

Hence, by (8) and (11) it follows that system (10) can be written equivalently as

$$(12) \qquad g_\pi T^{-1} A_\pi = r_\pi.$$

Now, (9) and (12) imply that

$$r_\pi A_\pi^{-1} g_\pi T^{-1} = (g_1, g_2 - g_1, \dots, g_n - g_{n-1}) \in \Re \times \Re_-^{n-1},$$

and by Proposition 3 it follows that the allocation indices of linear program (LP$_c$) satisfy $\gamma_\pi = r_\pi A_\pi^{-1} T$. Therefore, by (12),

$$g_i = \gamma_i, \quad \text{for } i \in \mathscr{N},$$

which completes the proof. □

A useful consequence of Theorems 2 and 3 is the following:

COROLLARY 1. *Under the assumptions of Theorem* 3, *an optimal solution of linear program* (LP$_c$) *can be computed by running algorithm* $\mathscr{AG}$ *with inputs* (r^k, A^k), *for* $k = 1, \dots, K$.

REMARK. It is important to emphasize that the index decomposition property is much stronger that the indexability property. We will see later in the context of stochastic scheduling that the classical multiarmed bandit problem exhibits the index decomposition property. This condition is not satisfied in general, however, by the optimal priority-indices in Klimov's problem (see Klimov (1974)).

We have focused our discussion in properties of linear programs over extended contra-polymatroids. The properties of linear programs over extended polymatroids are analogous. In Table 2 we show how to solve linear program (LP)—that minimizes a cost function hx over extended polymatroid $\mathscr{P}(A, b)$—by running algorithm $\mathscr{AG}$ with input (h, A), this obtaining corresponding allocation indices that characterize the optimal solution.

TABLE 2
Linear Programming Over Extended Polymatroids

Problem	*Allocation Indices*	*Optimal Solution*
$(\mathrm{LP})\min_{x \in \mathscr{P}(A,b)} hx$	$(h, A) \overset{\mathscr{AG}}{\mapsto} \gamma$	$\gamma_{\pi_n} \leq \cdots \leq \gamma_{\pi_1}$ $\bar{\mathrm{x}}_{\pi} = A_{\pi}^{-1} b_{\pi}$
$(\mathrm{LP}_c)\max_{x \in \mathscr{P}_c(A,b)} rx$	$(r, A) \overset{\mathscr{AG}}{\mapsto} \gamma$	$\gamma_{\pi_n} \leq \cdots \leq \gamma_{\pi_1}$ $\bar{x}_{\pi} = A_{\pi}^{-1} b_{\pi}$

4. A polyhedral approach to indexable scheduling problems.

In this section we develop a framework for formulating and solving stochastic scheduling problems solved by priority-index rules.

Section 4.1 introduces a concept of conservation laws that generalizes other definitions used in the literature. We show that the performance region corresponding to a scheduling problem that satisfies these laws is necessarily a polyhedron with special structure: an extended polymatroid. The vertices of the performance region are shown to be achieved by static-priority scheduling policies.

Section 4.2 shows how to formulate the problem of optimizing a linear performance objective, in a system that satisfies generalized conservation laws, as a linear program over an extended polymatroid, and how to obtain an optimal scheduling policy from the solution the formulation. We establish the optimality of priority-index rules and we present as assumption under which the optimal indices satisfy a decomposition property, which simplifies their computation.

4.1. *Generalized conservation laws.*

The wide variety of stochastic scheduling problems solved optimally by priority-index rules leads us naturally to consider the question: What physical properties of the system account for that indexability property? We provide in this section an answer to this question, by introducing and applying a general concept of conservation laws. We thus show that the performance region corresponding to a performance measure that satisfies such laws is necessarily an extended polymatroid, whose vertices are achievable under static-priority scheduling policies. As will be shown in the next section, the indexability property of linear programs over extended polymatroids translates into the optimality of priority-index policies in the scheduling problems that satisfy those laws.

Consider a general dynamic and stochastic multiclass queueing system. There are n job classes, which we label $i \in \mathscr{N} = \{1, \dots, n\}$. Jobs have to be scheduled for service in the system under an admissible scheduling policy. Let $\mathscr{U}$ denote the class of admissible scheduling policies. Let x_i^u be a performance measure of class-i jobs under scheduling policy u. We assume that x_i^u is an expectation. Let x^u denote the corresponding performance vector. Let x^{π} be the performance vector corresponding to a *static-priority* policy (i.e., the service priority of a job depends only on its class and does not change over time) that assigns priorities to the job classes according to permutation $\pi = (\pi_1, \dots, \pi_n)$ of $\mathscr{N}$, where class-π_1 has the highest priority.

DEFINITION 5 (GENERALIZED CONSERVATION LAWS). Performance vector x^u is said to satisfy *generalized conservation laws* if there exist a set function $b\colon 2^{\mathscr{N}} \to \mathscr{R}_+$ and a matrix $A = (A_i^S)_{i \in \mathscr{N}, S \subseteq \mathscr{N}}$ that satisfies $A_i^S > 0$, for $S \subseteq \mathscr{N}$, such that:
(a)

$$b(S) = \sum_{i \in S} A_i^S x_i^{\pi}, \quad \text{for all } \pi\colon \{\pi_1, \dots, \pi_{|S^c|}\} = S^c \quad \text{and } S \subseteq \mathscr{N}. \tag{13}$$

(b) For every admissible policy $u \in \mathscr{U}$,

$$(14) \qquad \sum_{i \in S} A_i^S x_i^u \geq b(S), \quad \text{for all } S \subset \mathscr{N} \quad \text{and} \quad \sum_{i \in \mathscr{N}} A_i^{\mathscr{N}} x_i^u = b(\mathscr{N}),$$

or respectively,

$$(15) \qquad \sum_{i \in S} A_i^S x_i^u \leq b(S), \quad \text{for all } S \subset \mathscr{N} \quad \text{and} \quad \sum_{i \in \mathscr{N}} A_i^{\mathscr{N}} x_i^u = b(\mathscr{N}).$$

REMARKS. 1. In words, a performance vector is said to satisfy generalized conservation laws if there exist weights A_i^S such that the total weighted performance over all job classes is invariant under any admissible policy, and the minimum (or maximal) weighted performance over job classes in any subset $S \subset \mathscr{N}$ is achieved by any static-priority policy that gives priority to all other classes (in S^c) over classes in S.

2. Shanthikumar and Yao (1992) formalized a definition of *strong conservation laws* for performance measures in general multiclass queues, that implies a polymatroidal structure in the performance space. These laws correspond to the special case that all weights are $A_i^S = 1$ in Definition 5.

The connection between generalized conservation laws and extended polymatroids is given by the following theorem:

THEOREM 4 (PERFORMANCE REGION CHARACTERIZATION). *Assume that performance vector x^u satisfies generalized conservation laws* (13) *and* (14) (*resp.* (13) *and* (15)). *Then*

(a) *The performance vectors corresponding to static-priority policies are the vertices of $\mathscr{P}_c(A, b)$ (resp. $\mathscr{P}(A, b)$), and $x^\pi = v(\pi)$, for every permutation π of $\mathscr{N}$.*

(b) *The performance region is the extended contra-polymatroid $\mathscr{P}_c(A, b)$ (resp. the extended polymatroid $\mathscr{P}(A, b)$).*

PROOF. We shall prove the theorem in the case that x^u satisfies generalized conservation laws (3) and (14). The other case ((13) and (15)) is analogous.

(a) By (13) it follows that $x^\pi = v(\pi)$. By Theorem 1 the result follows.

(b) Let $\mathscr{P} = \{x^u \colon u \in \mathscr{U}\}$ be the performance region. Let $\mathscr{P}_c^v(A, b)$ be the set of extreme points of $\mathscr{P}_c(A, b)$. By (14) it follows that $\mathscr{P} \subseteq \mathscr{P}_c(A, b)$. By (a), $\mathscr{P}_c^v(A, b) \subseteq X$. Hence, since $\mathscr{P}$ is a convex set ($\mathscr{U}$ contains randomized policies) we have

$$\mathscr{P}_c(A, b) = \operatorname{conv}(\mathscr{P}_c(A, b)) \subseteq X,$$

where conv$(\cdot)$ denotes the convex hull operator. Hence $\mathscr{P} = \mathscr{P}_c(A, b)$, which completes the proof. □

Consider the problem of designing an admissible policy that achieves a given performance vector x. It easily follows from Theorem 4 and Carathéodory theorem (see e.g. Bazaraa and Shetty (1979)) that any given performance vector x can be achieved by a randomization of at most n static-priority rules.

4.2. *Optimization over systems that satisfy conservation laws*: *Indexability.* Let x^u be a performance vector for a multiclass queueing system that satisfies generalized conservation laws (13) and (14). Suppose that we want to find an admissible policy u that maximizes a linear reward function $\sum_{i \in \mathscr{N}} r_i x_i^u$. This optimal scheduling control problem can be expressed as

$$(\mathrm{LP}_{\mathscr{U}}) \qquad \max\left\{ \sum_{i \in \mathscr{N}} r_i x_i^u : u \in \mathscr{U} \right\}.$$

By Theorem 4, problem $(\mathrm{LP}_{\mathscr{U}})$ can be translated into the linear program

$$(\mathrm{LP}_c) \qquad \max\left\{ \sum_{i \in \mathscr{N}} r_i x_i : x \in \mathscr{P}_c(A, b) \right\}.$$

The strong structural properties of extended polymatroids lead to strong structural properties in the control problem. Suppose that to each job class-i we attach an index, γ_i. A policy that selects for service at each decision epoch a job of currently largest index will be referred to as a *priority-index* policy. Let $\gamma_1, \ldots, \gamma_n$ be the allocation indices of linear program (LP_c), obtained by running adaptive-greedy algorithm with input (r, A), as described in §3. As a direct consequence of the results of §3.1 we obtain that control problem $(\mathrm{LP}_{\mathscr{U}})$ is solved by an index policy, with optimal priority indices given by $\gamma_1, \ldots, \gamma_n$.

THEOREM 5 (INDEXABILITY UNDER GENERALIZED CONSERVATION LAWS). (a) *Let $v(\pi)$ be an optimal solution of linear program* (LP_c); *then the static-priority policy that assigns priorities to job classes according to permutation π (class π_1 has highest priority) solves problem* $(\mathrm{LP}_{\mathscr{U}})$ *optimally*;

(b) *A policy that selects at each decision epoch a job of currently largest allocation index solves problem* $(\mathrm{LP}_{\mathscr{U}})$ *optimally.*

4.3. *Index decomposition.* A stronger *index decomposition* property holds under certain conditions. Assume $\mathscr{N}_1, \ldots, \mathscr{N}_K$ is a given partition of the set of job classes $\mathscr{N}$. Job classes in $\mathscr{N}_k$ may be interpreted as corresponding to a class-k *project.* Assume also that submatrix $A^k = (A_i^S)_{i \in \mathscr{N}_k, S \subseteq \mathscr{N}_k}$ of A (as a function on the parameters of the system) depends only on characteristics of job classes in $\mathscr{N}_k$ (i.e., of project class-k). If Assumption 2 holds, i.e.,

$$A_i^S = A_i^{S \cap \mathscr{N}_k}, \quad \text{for } i \in S \cap \mathscr{N}_k \quad \text{and } S \subseteq \mathscr{N},$$

then the index decomposition theorem for extended polymatroids (Theorem 3) applies, and therefore

$$\gamma_i = \gamma_i^k, \quad \text{for } i \in \mathscr{N}_k,$$

where the $\gamma_i^k(r^k, A^k)$'s, for $i \in \mathscr{N}_k$, are the indices obtained by algorithm $\mathscr{AG}$ with input (r^k, A^k), and $r^k = (r_i)_{i \in \mathscr{N}_k}$. Combining this result with Theorem 5 we obtain:

THEOREM 6 (INDEX DECOMPOSITION FOR MULTICLASS QUEUEING SYSTEMS). *The allocation indices corresponding to job classes in $\mathscr{N}_k$ only depend on characteristics of project class-k.*

The previous theorem identifies a sufficient condition for the indices of an indexable system to have a strong *decomposition* property. Therefore, systems that in addition to generalized conservation laws further satisfy Assumption 2 are *decomposable.* For such systems the solution of optimal scheduling control problem $(\mathrm{LP}_{\mathscr{U}})$ can be obtained by solving K smaller independent subproblems using algorithm $\mathscr{AG}$.

An example of a decomposable system is the multiclass-$M/G/1$ queue with performance measure

(16)
$$x_i^u = \frac{N_i^u}{\mu_i}, \qquad i \in \mathcal{N},$$

where N_i^u is the time average number of class-i jobs (with mean processing time $1/\mu_i$) in the system under scheduling policy u. Given a holding cost c_i per unit time for each job class-i, the goal is to assign the jobs to the server according to a nonanticipative, nonidling and nonpreemptive scheduling policy in order to minimize time average holding cost:

$$\min_u \sum_{i \in \mathcal{N}} c_i N_i^u = \sum_{i \in \mathcal{N}} (c_i \mu_i) x_i^u.$$

Gelenbe and Mitrani (1980) first showed that performance measure x^u given by (16) satisfies the following conservation laws:

(17)
$$\sum_{i \in S} x_i^u \geq b(S), \quad \text{for } S \subset \mathcal{N},$$
$$\sum_{i \in \mathcal{N}} x_i^u = b(\mathcal{N}),$$

with equality in (17) if policy u gives priority to jobs with classes in S. Assumption 2 clearly holds in this case, since all $A_i^S = 1$, for $i \in S$ and $S \subseteq \mathcal{N}$. This is the reason that the optimal index for each job class depends only on characteristics of that job class: the allocation indices are given by $\gamma_i = c_i \mu_i$, thus explaining the optimality of the $c\mu$ rule from a linear programming perspective.

Another well known example of decomposable system is the multiarmed bandit problem. We will see later that in this case the allocation indices reduce indeed to the original Gittins indices. Furthermore, Theorem 6 explains the fact that Gittins indices correspondıng to the states of project k only depend on characteristics of that project.

Let us consider briefly the problem of optimizing a nonlinear cost function on the performance vector. Analogously as what we did in the linear rewards case, the optimal scheduling control problem in the case of a nonlinear reward function can be translated into a nonlinear program whose feasible region is an extended polymatroid. See Bhattacharya et al. (1991) for a discussion of algorithmic methods for solving separable convex, min-max, lexicographic and semi-separable convex optimization problems over an extended polymatroid.

5. Branching bandit problems.

In this section we apply the framework developed in §4 to formulate and solve in a unifying way a variety of stochastic scheduling problems whose optimal policies are known to be of priority-index type. These include the classical multiarmed bandit problem (see e.g. Gittins (1989)), Klimov's problem (see Klimov (1974)), and the minimum weighted flow time problem in deterministic scheduling (see Smith (1956)). These and other indexable scheduling problems correspond to special cases of the branching bandit problem (see Meilijson and Weiss (1977), and Weiss (1988)).

The section introduces the branching bandit model, defines associated optimal scheduling problems, and shows how to cast them in the framework presented in §4. Suitable performance measures are defined, and it is established that they satisfy generalized conservation laws. The scheduling problems are thus formulated as linear programs over extended polymatroids, and they are shown to be solved by a priority-index policy, with the optimal indices being computed by a one-pass adaptive-greedy algorithm.

A branching bandit is a versatile model of a multiclass single-server queue. The systems that can be modeled as branching bandits include multiclass queues in discrete or continuous time, and with or without arrivals, as well as multiarmed bandits. This model was first introduced by Meilijson and Weiss (1977).

In a branching bandit model, a single server must be allocated over time to jobs demanding its service attention. Jobs are classified in a finite number of job classes $i \in \mathcal{N} = \{1, \ldots, n\}$. We associate with job class i a random service time v_i and random arrivals $(N_{ij})_{j \in \mathcal{N}}$. When a class-$i$ job completes its service, it is replaced by new jobs N_{ij} of class-j, for $j \in \mathcal{N}$. Given the job class-i, the service time and the descendants $(v_i, (N_{ij})_{j \in \mathcal{N}})$ are random variables with an arbitrary joint distribution, independent of all other jobs. Jobs are to be selected for service under an *admissible* policy u, which must be nonanticipative (decisions may be based on past and present information on the evolution of the system, but not on future information, such as the service time of the next job to be serviced), nonidling (the server is busy as long as there are jobs in the system) and nonpreemptive (the service of a job, once started, must proceed without interruption until its completion). Let us denote $\mathcal{U}$ the class of admissible policies. The decision epochs are $t = 0$ and the instants at which a job is completed and there is some other job present.

We next introduce some notation and concepts that will be useful for analyzing the sample-path of a branching bandit process. Let $S \subseteq \mathcal{N}$ be a subset of job classes. We shall refer to jobs whose classes are in S as S-jobs. We are interested in studying the busy period of a branching bandit process. This busy period may be represented as a tree. We say that S-job i_1 is an S-descendant of job i_0, if i_1 belongs in the subtree of the busy period that is rooted at i_0. Given a job i in a busy period, the union of intervals where S-descendants of i are being processed is called an (i, S)-period. Notice that under a policy that gives complete priority to S-jobs, these intervals will be consecutive. Let T_i^S be the duration (possibly infinite) of an (i, S)-period. We define $T_{L(0)}^S$ as the time needed for first clearing the system of S-jobs, under a policy that gives priority to S-jobs. The distributions of T_i^S and of $T_{L(0)}^S$ are independent of the admissible policy used, as long as it gives priority to S-jobs. Notice that an $(i, \varnothing)$-period is distributed as the service time v_i. It will be convenient to introduce the following additional notation:

v_{ik} = service time corresponding to the kth selection of a class-i job; notice that the distribution of $v_{i,k}$ is independent of k (v_i).

τ_{ik} = time at which the kth selection of a class-i job occurs;

ν_i = number of times a class-i job is selected (can be infinite);

$\{T_{ik}^S\}_{k \geq 1}$ = duration of the (i, S)-period that starts with the kth selection of a class-i job for the kth time. Notice that the distribution of $T_{i,k}^S$ is independent of k (T_i^S).

$I_i(t) = 1$, if a class-i job is in service at time t; 0, otherwise.

$L_i(t)$ = number of class-i jobs in the system at time t. We denote $L(t) = (L_i(t))_{i \in \mathcal{N}}$.

$T_{L(0)}^S$ = time until the system is first cleared of S-jobs (can be infinite) under a policy that gives priority to S-jobs; notice that $T_{L(0)}^{\mathcal{N}}$ is the length of the busy period.

The busy period of a branching bandit process has a simple structure under priority policies. This fact, which will be needed later for proving that certain performance measures satisfy conservation laws, is made precise and shown next. Let $S \subseteq \mathcal{N}$.

PROPOSITION 4. *Under an admissible policy that gives priority to S^c-jobs, the busy period $[0, T_{L(0)}^{\mathcal{N}})$ can be partitioned as follows*:

(18) $$\left[0, T_{L(0)}^{\mathcal{N}}\right) = \left[0, T_{L(0)}^{S^c}\right) \bigcup_{i \in S} \bigcup_{k=1}^{\nu_i} \left[\tau_{ik}, \tau_{ik} + T_{ik}^{S^c}\right).$$

PROOF OUTLINE. Identity (18) simply expresses the intuitive fact that under a policy that gives complete priority to S^c-jobs, the busy period is partitioned into: (1) an initial interval, in which the system is first cleared of S^c-jobs; and (2) a sequence of consecutive intervals, each of which starts with the service of an S-job, and lasts until the system is first cleared of its descendant S^c-jobs. □

5.1. *Discounted branching bandits*

5.1.1. *The discounted reward-tax problem.* Consider the following linear discounted reward-tax structure on a branching bandit process: an instantaneous reward r_i is received at the completion epoch of a class-i job. In addition, a holding tax αh_i is incurred continuously while a class-i job is in the system. Rewards and taxes are discounted in time with a discount factor $\alpha > 0$.

The discounted reward-tax problem consists in finding an admissible scheduling policy that maximizes the total expected discounted value of rewards earned minus taxes incurred. This problem was first introduced and shown to be solved by a priority-index policy by Weiss (1988).

Let us define the objective to be maximized as $Z_u^{(r,h)}(\alpha)$, where

$Z_u^{(r,h)}(\alpha)$-expected total discounted value of rewards received minus taxes incurred under scheduling policy u.

The problem can now be written as

$$\max_{u \in \mathscr{U}} Z_u^{(r,h)}(\alpha).$$

We shall show how to formulate and solve this problem in the framework developed in §4.

5.1.2. *Performance measures.* We introduce next two families of performance measures for branching bandits, $\{\lambda^u(\alpha)\}_{\alpha > 0}$ and $\{L_j^{*u}(\alpha)\}_{\alpha > 0}$, that are appropriate for modeling the linear discounted reward-tax structure described above. For a given $\alpha > 0$, we define performance measure $\lambda_j^u(\alpha)$ of class-j jobs under policy u to be total expected discounted number of class-j job service completions, i.e.,

$$\lambda_j^u(\alpha) = E_u\left[\sum_{k=1}^{\nu_j} e^{-\alpha(\tau_{jk}+v_{jk})}\right] \tag{19}$$

$$= E[e^{-\alpha v_j}]E_u\left[\sum_{k=1}^{\nu_j} e^{-\alpha\tau_{jk}}\right], \quad \text{for } j \in \mathscr{N},$$

and we shall write $\lambda^u(\alpha) = (\lambda_j^u(\alpha))_{j \in \mathscr{N}}$.

Let us define $L_j^{*u}(\alpha)$ as the total expected discounted number of class-j jobs in the system under policy $u \in \mathscr{U}$, i.e.,

$$L_j^{*u}(\alpha) = E_u\left[\int_0^{T_{L(0)}^{\mathscr{N}}} L_j(t) e^{-\alpha t}\, dt\right], \quad \text{for } j \in \mathscr{N},$$

and let $L^{*u}(\alpha) = (L_j^{*u}(\alpha))_{j \in \mathscr{N}}$.

We shall show next that the objective to be maximized, $Z_u^{(r,h)}(\alpha)$, can be expressed as a linear function of performance vector $\lambda^u(\alpha)$.

In the pure rewards case, i.e. when $h = 0$,

$$(20) \qquad Z_u^{(r,0)}(\alpha) = E_u\left[\sum_{i \in \mathcal{N}} \sum_{k=1}^{\nu_i} r_i e^{-\alpha(\tau_{ik}+v_{ik})}\right]$$
$$= \sum_{i \in \mathcal{N}} r_i \lambda_i^u(\alpha).$$

We show now how to reduce the general reward-tax problem to the pure rewards case, using an accounting argument introduced by Bell (1971). The total expected discounted value of holding taxes is the same whether they are charged continuously in time, or according to the following charging scheme: At the arrival epoch of a class-i job, charge the system with an instantaneous *entrance change* of h_i, equal to the total discounted holding cost that would be incurred if the job remained within the system forever; at the departure epoch of the job (if it ever departs), credit the system with an instantaneous *departure refund* of h_i, thus refunding that portion of the entrance cost corresponding to residence beyond the departure epoch. We can thus write

$$(21) \qquad Z_u^{(r,h)}(\alpha) = \sum_{i \in \mathcal{N}} r_i \lambda_i^u(\alpha) - \alpha \sum_{j \in \mathcal{N}} h_j L^{*u}(\alpha)$$
$$= E_u[\text{Rewars}] - E_u[\text{Charges at } t = 0]$$
$$+ (E_u[\text{Departure refunds}] - E_u[\text{Entrance charges}])$$
$$= Z_u^{(r,0)}(\alpha) - \sum_{i \in \mathcal{N}} h_i L_i(0) + Z_u^{(r_\alpha,0)}(\alpha)$$
$$= Z_u^{(r+r_\alpha,0)}(\alpha) - \sum_{i \in \mathcal{N}} h_i L_i(0)$$
$$= \sum_{i \in \mathcal{N}} (r_i + r_{i,\alpha}) \lambda_i^u(\alpha) - \sum_{i \in \mathcal{N}} h_i L_i(0),$$

where the components of vector $r_\alpha = (r_{i,\alpha})_{i \in \mathcal{N}}$ are given by

$$(22) \qquad r_{i,\alpha} = h_i - \sum_{j \in \mathcal{N}} \frac{E[N_{ij} e^{-\alpha v_i}]}{E[e^{-\alpha v_j}]} h_j, \quad \text{for } j \in \mathcal{N}.$$

Notice that by letting $r = 0$, $h_j = 1$ and $h_i = 0$ for $i \neq j$ in (21) we obtain a linear relation between performance vectors $L^{*u}(\alpha)$ and $\lambda^u(\alpha)$:

$$(23) \qquad \alpha L_j^{*u}(\alpha) = L_j(0) - \lambda_j^u(\alpha) + \sum_{i \in \mathcal{N}} \frac{E[N_{ij} e^{-\alpha v_i}]}{E[e^{-\alpha v_j}]} \lambda_i^u(\alpha), \quad \text{for } j \in \mathcal{N}.$$

5.1.3. *Generalized conservation laws.* We show in this section that the performance measure for branching bandits $\lambda^u(\alpha)$ satisfies generalized conservation laws. Let us define, for $S \subseteq \mathcal{N}$,

$$A_{i,\alpha}^{S} = \frac{\alpha E\left[\int_0^{T_i^{S^c}} e^{-\alpha t}\,dt\right]}{E[e^{-\alpha v_i}]}, \quad \text{for } i \in S, \text{ and} \tag{24}$$

$$b_\alpha(S) = \alpha E\left[\int_0^{T_{L(0)}^{\mathcal{N}}} e^{-\alpha t}\,dt\right] - \alpha E\left[\int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt\right]. \tag{25}$$

The conservation laws we present next represent physical work conservation relations in a branching bandit process. In particular, the total expected discounted amount of work performed by the server during the busy period is

$$E\left[\int_0^{T_{L(0)}^{\mathcal{N}}} e^{-\alpha t}\,dt\right],$$

under any admissible scheduling policy $u \in \mathcal{U}$.

Notice that coefficient $A_{i,\alpha}^{S}$ may be interpreted as the total expected discounted amount of work performed by the server during and (i, S^c)-period under a policy that gives priority to S^c-jobs (and hence under which that (i, S^c)-period is a single interval). The sum

$$\sum_{i \in S} A_{i,\alpha}^{S} \lambda_i^{u}(\alpha)$$

is thus an overestimate of the total expected discounted work performed after the server first starts servicing s-jobs. This estimate is exact under a policy that gives priority to S^c-jobs. In this case, the expected discounted work done until the system is first cleared of S^c-jobs is

$$E\left[\int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt\right].$$

The remaining work is, in that case,

$$E\left[\int_0^{T_{L(0)}^{\mathcal{N}}} e^{-\alpha t}\,dt\right] - E\left[\int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt\right],$$

which is precisely $b_\alpha(S)$. This intuitive interpretation is made precise in the proof of the next result.

THEOREM 7 (GENERALIZED CONSERVATION LAWS FOR DISCOUNTED BRANCHING BANDITS. *The performance vector for branching bandits $\lambda^u(\alpha)$ satisfies the following generalized conservation laws*:

(a) $\sum_{i \in S} A_{i,\alpha}^{S} \lambda_i^{u}(\alpha) \ge b_\alpha(S)$, *for $S \subset \mathcal{N}$, with equality if policy u gives complete priority to S^c-jobs.*

(b) $\sum_{i \in \mathcal{N}} A_{i,\alpha}^{\mathcal{N}} \lambda_i^{u}(\alpha) = b_\alpha(\mathcal{N})$.

PROOF. Let $S \subseteq \mathcal{N}$. Let us assume that jobs are selected under an admissible policy u. Let us define two random vectors, $(r_i)_{i \in \mathcal{N}}$ and $(d_i^S)_{i \in S}$, as functions of the sample path of the generated branching bandit process as follows:

$$(26)\qquad r_i = \int_0^\infty I_i(t) e^{-\alpha t}\,dt = \sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+\upsilon_{ik}} e^{-\alpha t}\,dt$$

$$= \sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{\upsilon_{ik}} e^{-\alpha t}\,dt, \quad \text{and}$$

$$d_i^S = \sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{T_{ik}^{S^c}} e^{-\alpha t}\,dt, \qquad i \in S.$$

Now, we have

$$(27)\qquad E_u[r_i] = E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{\upsilon_{ik}} e^{-\alpha t}\,dt\right]$$

$$= E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{\upsilon_{ik}} e^{-\alpha t}\,dt \mid \nu_i\right]\Bigg]$$

$$= E_u\left[\sum_{k=1}^{\nu_i} E[e^{-\alpha\tau_{ik}} \mid \nu_i] E\left[\int_0^{\upsilon_{ik}} e^{-\alpha t}\,dt\right]\right]$$

$$(28)\qquad = E\left[\int_0^{\upsilon_i} e^{-\alpha t}\,dt\right] E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}}\right]$$

$$= \frac{E\left[\int_0^{\upsilon_i} e^{-\alpha t}\,dt\right]}{E[e^{-\alpha\upsilon_i}]} \lambda_i^u(\alpha).$$

Notice that equality (27) holds because, since u is nonanticipative, τ_{ik} and υ_{ik} are independent random variables. Furthermore,

$$(29)\quad E_u[d_i^S] = E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{T_{ik}^{S^c}} e^{-\alpha t}\,dt\right] = E_u\left[E\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}} \int_0^{T_{ik}^{S^c}} e^{-\alpha t}\,dt \mid \nu_i\right]\right]$$

$$= E_u\left[\sum_{k=1}^{\nu_i} E\left[\int_0^{T_i^{S^c}} e^{-\alpha t}\,dt\right] E[e^{-\alpha\tau_{ik}} \mid \nu_i]\right]$$

$$(30)\qquad = E\left[\int_0^{T_i^{S^c}} e^{-\alpha t}\,dt\right] E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}}\right]$$

$$= A_{i,\alpha}^S E\left[\int_0^{\upsilon_i} e^{-\alpha t}\,dt\right] E_u\left[\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}}\right].$$

Notice that equality (29) holds because, since u is nonanticipative, τ_{ik} and $T_{ik}^{S^c}$ are independent. Hence, by (28) and (30)

$$E_u[d_i^S] = A_{i,\alpha}^S \lambda_i^u(\alpha), \quad \text{for } i \in S,$$

and we thus obtain

$$(31) \qquad E_u\left[\sum_{i\in S} d_i^S\right] = \sum_{i\in S} A_{i,\alpha}^S \lambda_i^u(\alpha).$$

We first show that if policy $u = \pi$ gives complete priority ty S^c-jobs then generalized conservation law (a) holds with equality. Applying Proposition 4, we obtain

$$(32) \qquad \int_0^{T_{L(0)}^{\mathcal{N}}} e^{-\alpha t}\,dt = \int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt + \sum_{i\in S}\sum_{k=1}^{\nu_i}\int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\,dt$$
$$= \int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt + \sum_{i\in S}\sum_{k=1}^{\nu_i} e^{-\alpha\tau_{ik}}\int_0^{T_{ik}^{S^c}} e^{-\alpha t}\,dt$$
$$= \int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt + \sum_{i\in S} d_i^S.$$

Hence, taking expectations and using equation (31) we obtain

$$E\left[\int_0^{T_{L(0)}^{\mathcal{N}}} e^{-\alpha t}\,dt\right] = E\left[\int_0^{T_{L(0)}^{S^c}} e^{-\alpha t}\,dt\right] + \sum_{i\in S} A_{i,\alpha}^S \lambda_i^\pi(\alpha)$$

or equivalently, by (25),

$$\sum_{i\in S} A_{i,\alpha}^S \lambda_i^\pi(\alpha) = b_\alpha(S),$$

which proves that generalized conservation law (a) holds with equality. Notice that by letting $S = \varnothing$ we obtain the conservation law in part (b).

We next show that generalized conservation law (a) is satisfied in the inequality case. We will use a sample path interchange argument.

Let the jobs be selected under an admissible policy u. For a given sample path of the branching bandit process let us consider the sum

$$d^S = \sum_{i\in S} d_i^S = \sum_{i\in S}\sum_{k=1}^{\nu_i}\int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\,dt.$$

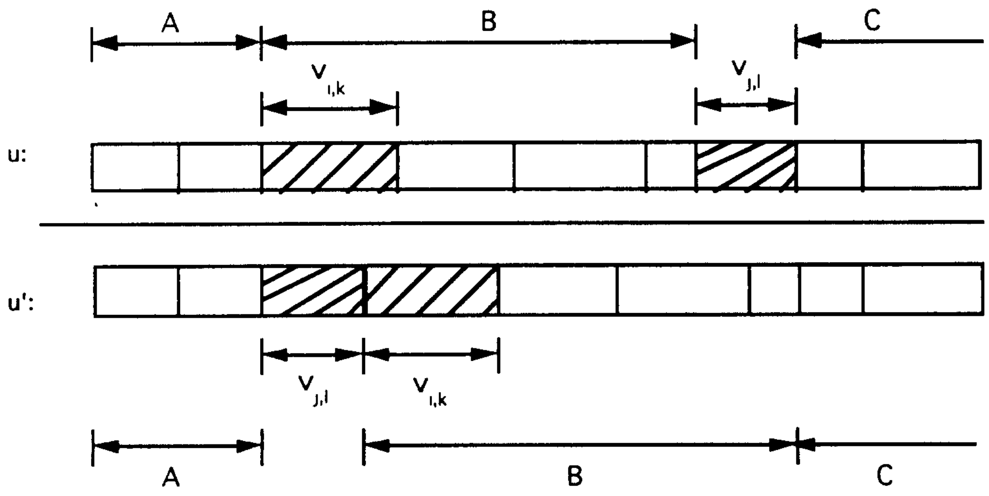

FIGURE 5. Interchange argument.

Suppose that at time τ_{i^*,k^*} a class-i^* job, with $i^* \in S$ is selected for the k^*th time. Suppose that at that time a class-j^* job, with $j^* \in S^c$ is also available, but it is selected later, at time τ_{j^*,l^*}. Let us consider the effect of selecting instead that class-j^* job in S^c at time τ_{i^*,k^*}, and selecting immediately afterwards the class-i^* job. Let us call the corresponding policy u'. Let A, B and C be the segments shown in Figure 5. The sum d^S can be decomposed as

$$d^S = \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in A}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\, dt + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in B}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\, dt + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in C}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\, dt. \tag{33}$$

Let d'^S be the sum corresponding to policy u'. It is clear from Figure 5, (33) and that fact that the function $e^{-\alpha t}$ is decreasing in t that

$$\begin{aligned} d'^S = {} & \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in A}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\, dt \\ & + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in B}^{\nu_i} \int_{\tau_{ik}-v_{j^*,l^*}}^{\tau_{ik}+v_{j^*,l^*}+T_{ik}^{S^c}} e^{-\alpha t} dt \\ & + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in C}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} e^{-\alpha t}\, dt \\ < {} & d^S. \end{aligned} \tag{34}$$

It follows that for this sample path a policy that gives complete priority to S^c-jobs minimizes the sum d^S. Hence this result holds taking expectations, and by equation (31) conservation law (a) follows, which completes the proof of the theorem. □

Since performance measure $\lambda^u(\alpha)$ satisfies generalized conservation laws, the results of §4 apply. Direct application of Theorem 4 yields the next result.

COROLLARY 2. *The performance region for branching bandits corresponding to performance vector $\lambda^u(\alpha)$ is the extended contra-polymatroid $\mathcal{P}_c(A_\alpha, b_\alpha)$; furthermore, the vertices of $\mathcal{P}_c(A_\alpha, b_\alpha)$ are the performance vectors corresponding to static-priority policies.*

5.1.4. *Optimal solution.* From equation (21) it is clear how to apply the results of §4 to solve the reward-tax problem: run adaptive-greedy algorithm $\mathcal{AG}$ with input $(r + r_\alpha, A_\alpha)$. Let $\gamma_1(\alpha), \ldots, \gamma_n(\alpha)$ be the allocation indices so obtained. Then we have, by Theorem 5.

THEOREM 8 (INDEXABILITY: DISCOUNTED BRANCHING BANDITS). *An optimal scheduling policy is to serve at each decision epoch a job with largest current allocation index $\gamma_i(\alpha)$.*

The previous theorem characterizes the structure of the optimal policy. We will see later that we can also compute the performance of the optimal policy, as we will present in Proposition 6 closed formulae for matrix A_α and set function b_α.

5.1.5. *Economic interpretation of allocation indices in discounted branching bandits.* The original definition of dynamic allocation indices in multiarmed bandit problems (see e.g. Gittins (1989)) characterized them as optimal reward rates with respect to a family of stopping times. Our definition of allocation indices, on the other hand, involves sums of optimal dual variables in a certain linear program. In this section we clarify the relation between these two definitions, and show that they are, indeed, equivalent.

Given a discounted branching bandit problem, as described above, we shall define a modified problem by adding an additional job class, which we label 0, with infinite service time (i.e., $v_0 = \infty$). A reward of r_0, continuously discounted in time, is received for each unit of time that a class-0 job is in service. Notice that the option to serve a class-0 job may be interpreted as an option to retire from the game modeled by the original problem for a *pension* of r_0, discounted in time. Notice also that the modified problem is still a branching bandit problem.

Let us now assume that at time $t = 0$ there are only two jobs present, a class-0 and a class-i job, with $i \in \mathcal{N}$. We may then consider the following question: What is the smallest value of the pension r_0 that makes the option of retirement (serving the class-0 job) preferable to the option of continuation (serving the class-i job)? Let us call this *break-even value* $r_0^*(i)$. Let $\gamma_1, \ldots, \gamma_n$ be the allocation indices corresponding to the original branching bandit problem.

PROPOSITION 5. $\gamma_i = r_0^*(i)$.

PROOF. Let $\gamma_0^0, \gamma_1^0, \ldots, \gamma_n^0$ be the allocation indices for the modified problem. Let us partition the corresponding modified state space as $\hat{\mathcal{N}} = \{0\} \cup \mathcal{N}$. It is easily seen that Assumption 2 holds for the modified problem. Hence Theorem 3 applies, and the problem is decomposable. Consequently,

$$\gamma_0^0 = r_0 \quad \text{and} \quad \gamma_j^0 = \gamma_j, \qquad j \in \mathcal{N}.$$

Now, since by Theorem 8 it is optimal to serve a job with largest current allocation index, it follows that the break-even reward r_0 which makes the options of continuation and of retirement (with reward r_0) equally attractive is $r_0 = \gamma_0^0$. But, by definition, $r_0^*(i)$ is such a breakpoint. Therefore $r_0^*(i) = \gamma_0^0$, which completes the proof. □

REMARK. Whittle (1980), (1982) introduced the idea of a retirement option in his analysis of the multiarmed bandit problem, and provided an interpretation of the Gittins indices as break-even values. Weber (1992) also made use of this characterization of the Gittins indices in his intuitive proof. Here we extend this interpretation to the more general case of branching bandits. From this characterization it follows that the allocation indices coincide with the well known Gittins indices in the classical multiarmed bandit problem.

5.1.6. *Parameter computation.* The results of the previous sections are structural, but do not lead to explicit computations of matrix A_α and set function b_α appearing in the generalized conservation laws for the branching bandit problem. Our goal in this section is to compute from the model data matrix A_α and set function b_α. Combined with the previous results these computations make possible to evaluate the performance of specific policies.

As generic data for the branching bandit model, we assume that the joint distribution of $(v_i, (N_{ij})_{j \in \mathcal{N}})$ is given by the transform

$$\Phi_i(\alpha, z_1, \ldots, z_n) = E\left[e^{-\alpha v_i} z_1^{N_{i1}} \cdots z_n^{N_{in}}\right]. \tag{35}$$

Notice that

$$E\left[N_{ij}\, e^{-\alpha v_i}\right] = \frac{\partial}{\partial z_j} \Phi_i(\alpha, 1).$$

In addition, we are given the Laplace transform of the marginal distribution of service time v_i,

$$\Psi_i(\alpha) = E[e^{-\alpha v_i}].$$

Finally the vector $L(0) = (L_1(0), \ldots, L_n(0))$ of jobs present at the start is assumed to be known.

As we saw in the previous section the duration of an (i, S)-period, T_i^S, plays a crucial role. We will compute its Laplace transform function,

$$\Psi_i^S(\alpha) = E\left[e^{-\alpha T_i^S}\right].$$

For this reason we decompose the duration of an (i, S)-period as a sum of independent random variables as follows:

(36) $$T_i^S \overset{\mathrm{d}}{=} v_i + \sum_{j \in S} \sum_{k=1}^{N_{ij}} T_{j,k}^S,$$

where $v_i, \{T_{j,k}^S\}_{k \geq 1}$ are independent. Therefore,

(37) $$\begin{aligned} \Psi_i^S(\alpha) &= E\left[e^{-\alpha v_i} E\left[e^{-\alpha \sum_{j \in S} \sum_{k=1}^{N_{ij}} T_{j,k}^S} | v_i\right]\right] \\ &= E\left[e^{-\alpha v_i} \prod_{j \in S} E\left[e^{-\alpha T_j^S}\right]^{N_{ij}}\right] \\ &= \Phi_i\left(\alpha, \Psi_S^S(\alpha), 1_{S^c}\right), \quad \text{for } i \in \mathcal{N}, \end{aligned}$$

where $\Psi_S^S(\alpha) = (\Psi_i^S(\alpha))_{i \in S}$. Given S, fixed point system (37) provides a way to compute the values of $\Psi_i^S(\alpha)$, for $i \in \mathcal{N}$. We can now prove the following result:

PROPOSITION 6 (COMPUTATION OF A_α AND b_α). *For a branching bandit process, matrix A_α and set function b_α satisfy the following relations*:

(38) $$A_{i,\alpha}^S = \frac{1 - \Psi_i^S(\alpha)}{\Psi_i(\alpha)}, \quad \textit{for } i \in S \textit{ and } S \subseteq \mathcal{N};$$

(39) $$b_\alpha(S) = \prod_{j \in S^c} \left[\Psi_j^{S^c}(\alpha)\right]^{L_j(0)} - \prod_{j \in \mathcal{N}} \left[\Psi_j^{\mathcal{N}}(\alpha)\right]^{L_j(0)}, \quad \textit{for } S \subseteq \mathcal{N}.$$

PROOF. Relation (38) follows directly from the definition of $A_{i,\alpha}^S$. Furthermore,

(40) $$T_{L(0)}^S \overset{\mathrm{d}}{=} \sum_{i \in S} \sum_{k=1}^{L_i(0)} T_{ik}^S.$$

Hence,

(41)
$$E\left[\int_0^{T^S_{L(0)}} e^{-\alpha t}\,dt\right] = \frac{1}{\alpha} - \frac{1}{\alpha}E\left[e^{-\alpha\sum_{i\in S}\sum_{k=1}^{L_i(0)} T^S_{ik}}\right]$$
$$= \frac{1}{\alpha} - \frac{1}{\alpha}\prod_{i\in S}\left[\Psi_i^S(\alpha)\right]^{L_i(0)}.$$

Therefore, from (25), (39) follows. □

REMARKS. 1. Notice that $A^{\mathcal{N}}_{i,\alpha} = (1 - \Psi_i(\alpha)/\Psi_i(\alpha)$, for $i \in \mathcal{N}$, and

$$b_\alpha(\mathcal{N}) = 1 - \prod_{j\in\mathcal{N}}\left[\Psi_j^{\mathcal{N}}(\alpha)\right]^{L_j(0)}, \quad \text{for } S \subseteq \mathcal{N}.$$

2. From Proposition 6 we can compute matrix A_α and set function b_α provided we can solve system (37). As an example, we illustrate the form of the equations in the special case, in which the class-j jobs that arrive during the time that we work on class-i job form a Poisson process with rate λ_{ij}, i.e.,

$$\Phi_i(\alpha, z_1, \dots, z_n) = E\left[e^{-v_i(\alpha + \sum_{j\in\mathcal{N}}\lambda_{ij}(1-z_j))}\right] = \Psi_i\left(\alpha + \sum_{j\in\mathcal{N}}\lambda_{ij}(1 - z_j)\right).$$

In this case, (37) yields

(42)
$$\Psi_i^S(\alpha) = \Psi_i\left(\alpha + \sum_{j\in S}\lambda_{ij}\left(1 - \Psi_j^S(\alpha)\right)\right), \quad \text{for } i \in \mathcal{N}.$$

As a result, an algorithm to compute $\Psi_i^S(\alpha)$ is as follows:

(1) Find a fixed point for the system of nonlinear equations (42) in terms of $\Psi_i^S(\alpha)$. Although in general (42) might not have a closed form solution, in special cases (v_i exponential) a closed form solution can be obtained.

(2) From Proposition 6 compute (A_α, b_α) in terms of $\Psi_i^{S^c}(\alpha)$.

5.2. *Undiscounted branching bandits.* In this section we apply the framework developed in §4 to the branching bandit problem with a linear undiscounted cost criterion. This problem was first introduced and solved by Meilijson and Weiss (1977) using dynamic programming ideas.

We shall assume in what follows that matrix $E[N] = (E[N_{ij}])_{i,j\in\mathcal{N}}$ satisfies the following condition:

ASSUMPTION 3. *Matrix* $E[N]$ *has spectral radius smaller than* 1.

Bertsimas, Paschalidis and Tsitsiklis (1994b) proved that under Assumption 3 the branching bandit process is *stable*, in the sense that the first and second moments of its busy period are finite.

5.2.1. *The undiscounted tax problem.* In the undiscounted tax problem, a holding tax h_i per unit time is incurred continuously while a class-i job is in the system. We do not consider undiscounted rewards (i.e. a reward r_i is earned on completion of a class-i job) since the total expected reward earned is the same under all policies. Let us define the objective to be maximized, Z_u^h, as

$$Z_u^h = -(\text{total expected tax incurred under policy } u).$$

The tax problem can now be written as the optimal control problem

$$\max_{u \in \mathcal{U}} Z_u^h. \tag{43}$$

5.2.2. *Performance measures.* We introduce next two performance measures, $C^u = (C_j^u)_{j \in \mathcal{N}}$ and $W^u = (W_j^u)_{j \in \mathcal{N}}$, that are appropriate for modeling a linear undiscounted delay cost structure. We assume in what follows that all the expectations that appear are finite. Later we will show necessary and sufficient conditions for this assumption to hold. Using the indicator

$$I_j(t) = \begin{cases} 1, & \text{if a class-}j\text{ job is in service at time } t; \\ 0, & \text{otherwise,} \end{cases}$$

we introduced earlier, let us define performance measure C_j^u as the total expected completion time of class-j jobs under policy u, i.e.,

$$C_j^u = E_u\left[\sum_{k=1}^{\nu_j} (\tau_{jk} + v_{jk})\right]. \tag{44}$$

Let us define another performance measure, W_j^u, as the total expected system time of class-j jobs under policy u, i.e.,

$$W_j^u = E_u\left[\int_0^{T_{L(0)}^{\mathcal{N}}} L_j(t)\, dt\right]. \tag{45}$$

We show now how to express objective Z_u^h as a linear function of performance vectors W^u or C^u. First, it is clear from the definition of the undiscounted tax problem that

$$Z_u^h = -\sum_{j \in \mathcal{N}} h_j W_j^u. \tag{46}$$

As for the relation with performance vector C^u, it is obtained by taking the limit as $\alpha \searrow 0$ on Eq. (23), which yields

$$L_j^{*u}(0) = -\frac{d}{d\alpha}\lambda_j^u(0) + \sum_{i \in \mathcal{N}} E[N_{ij}]\frac{d}{d\alpha}\lambda_i^u(0) + g_j, \quad \text{for } j \in \mathcal{N}, \text{ where} \tag{47}$$

$$g_j = \sum_{i \in \mathcal{N}} E[\nu_i]\big(E[v_i]E[N_{ij}] - E[v_i N_{ij}]\big).$$

Since it can be seen that

$$\frac{d}{d\alpha}\lambda_j^u(0) = -C_j^u,$$

we obtain the following linear relation between performance vectors W^u and C^u:

$$W_j^u = C_j^u - \sum_{i \in \mathcal{N}} E[N_{in}]C_i^u + g_j, \quad \text{for } j \in \mathcal{N}. \tag{48}$$

We can also express objective Z_u^h as a linear function of performance vector C^u. By (48),

$$Z_u^h = -\sum_{j \in \mathcal{N}} h_j W_j^u \tag{49}$$

$$= -\sum_{i \in \mathcal{N}} \left(h_i - \sum_{j \in \mathcal{N}} E[N_{ij}] h_j \right) C_i^u - \sum_{i \in \mathcal{N}} h_i g_i .$$

5.2.3. *Conservation laws for the completion times.* We show next that performance measures C^u satisfies generalized conservation laws. Let us define

$$A_i^S = E\left[T_i^{S^c}\right], \quad \text{for } i \in S, \text{ and} \tag{50}$$

$$b(S) = \frac{1}{2} E\left[\left(T_{L(0)}^{\mathcal{N}}\right)^2\right] - \frac{1}{2} E\left[\left(T_{L(0)}^{S^c}\right)^2\right] + \sum_{i \in S} b_i(S), \quad \text{where} \tag{51}$$

$$b_i(S) = E[\nu_i]\left(E[v_i] E\left[T_i^{S^c}\right] - \frac{1}{2} E\left[\left(T_i^{S^c}\right)^2\right]\right), \quad \text{for } i \in S.$$

THEOREM 9 (GENERALIZED CONSERVATION LAWS FOR THE COMPLETION TIMES). *The performance vector for branching bandits C^u satisfies the following generalized conservation laws*:

(a) $\sum_{i \in S} A_i^S C_i^u \leq b(S)$, *for $S \subset \mathcal{N}$, with equality if policy u gives complete priority to S^c-jobs.*

(b) $\sum_{i \in \mathcal{N}} A_i^{\mathcal{N}} C_i^u = b(\mathcal{N})$.

PROOF. Let $S \subseteq \mathcal{N}$. Let us assume that jobs are selected under an admissible policy u. This generates a branching bandit process. Let us define two random vectors, $(r_i)_{i \in \mathcal{N}}$ and $(d_i^S)_{i \in S}$, as functions of the sample path as follows:

$$r_i = \int_0^\infty I_i(t) t \, dt = \sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik} + v_{ik}} t \, dt \tag{52}$$

$$= \sum_{k=1}^{\nu_i} \left(v_{ik} \tau_{ik} + \frac{v_{ik}^2}{2} \right), \qquad i \in \mathcal{N}, \quad \text{and}$$

$$d_i^S = \sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik} + T_{ik}^{S^c}} t \, dt, \qquad i \in S.$$

Now, we have

$$E_u[r_i] = E_u\left[\sum_{k=1}^{\nu_i} E\left[\left(v_{ik} \tau_{ik} + \frac{v_{ik}^2}{2} \right) \middle| \nu_i \right] \right] \tag{53}$$

$$= E_u\left[\sum_{k=1}^{\nu_i} \left(E[v_i] E[\tau_{ik} | \nu_i] + \frac{E[v_i^2]}{2} \right) \right]$$

$$= E[v_i] E_u\left[\sum_{k=1}^{\nu_i} \tau_{ik} \right] + \frac{E[\nu_i] E[v_i^2]}{2}. \tag{54}$$

Note that equality (53) holds because, since u is nonanticipative, τ_{ik} and υ_{ik} are independent random variables. Furthermore,

$$(55)\qquad E_u[d_i^S] = E_u\left[\sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\,dt\right] = E_u\left[E\left[\sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\,dt \mid \nu_i\right]\right]$$

$$= E_u\left[\sum_{k=1}^{\nu_i} E\left[\left(\tau_{ik}T_{ik}^{S^c} + \frac{(T_{ik}^{S^c})^2}{2}\right)\mid \nu_i\right]\right]$$

$$= E[T_i^{S^c}]E_u\left[\sum_{k=1}^{\nu_i} \tau_{ik}\right] + \frac{E[\nu_i]E\left[(T_i^{S^c})^2\right]}{2}.$$

Note that equality (55) holds because, since policy u is nonanticipative, τ_{ik} and $T_{ik}^{S^c}$ are independent random variables. Hence, by (54) and (55),

$$C_i^u - E[\nu_i]E[\upsilon_i] = \frac{E_u[r_i] - \frac{1}{2}E[\nu_i]E[\upsilon_i^2]}{E[\upsilon_i]}$$

$$= \frac{E_u[d_i^S] - \frac{1}{2}E[\nu_i]E\left[(T_i^{S^c})^2\right]}{E\left[T_i^{S^c}\right]}, \qquad i \in S$$

and therefore we obtain

$$(56)\qquad \sum_{i\in S} A_i^S C_i^u = E_u\left[\sum_{i\in S} d_i^S\right] + \sum_{i\in S} b_i(S).$$

We will first show that if policy $u = \pi$ gives complete priority to S^c jobs then generalized conservation law (a) holds with equality. Applying Proposition 4 we obtain:

$$(57)\qquad \int_0^{T_{L(0)}^{\mathcal{N}}} t\,dt = \int_0^{T_{L(0)}^{S^c}} t\,dt + \sum_{i\in S}\sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\,dt$$

$$= \frac{\left(T_{L(0)}^{S^c}\right)^2}{2} + \sum_{i\in S} d_i^S.$$

Hence, taking expectations and using Eq. (56) and the definition of $b(S)$ we obtain

$$\sum_{i\in S} A_i^S C_i^\pi = b(S),$$

which proves that generalized conservation law (a) holds with equality. Notice that by letting $S = \varnothing$ part (b) follows.

We next show that generalized conservation law (a) is satisfied in the inequality case. We will use a sample path interchange argument. Let the jobs be selected under an admissible policy u. For a given sample path of the branching bandit process let us consider the sum

$$d^S = \sum_{i \in S} d_i^S = \sum_{i \in S} \sum_{k=1}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt.$$

Suppose that at time τ_{i^*,k^*} a class-i^* job, with $i^* \in S$ is selected for the k^*th time. Suppose that at that time a class-j^* job, with $j^* \in S^c$ is also available, but it is selected later, at time τ_{j^*,l^*}. Let us consider the effect of selecting instead that class-j^* job in S^c at time τ_{i^*,k^*}, and selecting immediately afterwards the class-i^* job. Let us call the corresponding policy u'. Let A, B and C be the segments shown in Figure 5. The sum d^S can be decomposed as

$$(58) \qquad d^S = \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in A}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in B}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in C}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt.$$

Let d'^S be the sum corresponding to policy u'. It is clear from Figure 5, (58) and the fact that the function t is increasing in t that

$$\begin{aligned} d'^S &= \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in A}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt \\ &\quad + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in B}^{\nu_i} \int_{\tau_{ik}+v_{j^*,l^*}}^{\tau_{ik}+v_{j^*,l^*}+T_{ik}^{S^c}} t\, dt \\ &\quad + \sum_{i \in S} \sum_{k=1:\, \tau_{ik} \in C}^{\nu_i} \int_{\tau_{ik}}^{\tau_{ik}+T_{ik}^{S^c}} t\, dt \\ &> d^S. \end{aligned}$$

It follows that for this sample path a policy that gives complete priority to S^c-jobs maximizes the sum d^S. Hence this result holds taking expectations, and by Eq. (56) conservation law (a) follows, which completes the proof of the theorem. □

COROLLARY 3. *The performance region for branching bandits corresponding to performance vector C^u is the extended polymatroid $\mathscr{P}(A, b)$; furthermore, the vertices of $\mathscr{P}(A, b)$ are the performance vectors corresponding to static-priority rules.*

5.2.4. *Conservation laws for the number in system.* We show next that the performance measure for branching bandits W^u defined by (45) satisfies generalized conservation laws. Let us define, for $S \subseteq \mathscr{N}$,

$$(59) \qquad \bar{A}_i^S = E\left[T_i^S\right], \quad \text{for } i \in S, \text{ and}$$

$$(60) \qquad \bar{b}(S) = b(\mathscr{N}) - b(S^c) + \sum_{j \in S} g_j E\left[T_j^S\right],$$

where g_j is given by (47).

THEOREM 10 (GENERALIZED CONSERVATION LAWS FOR THE NUMBER IN SYSTEM). *The performance vector for branching bandits W^u satisfies the following generalized conservation laws:*

(a) $\sum_{i \in S} \bar{A}_i^S W_i^u \geq \bar{b}(S)$, *for $S \subset \mathcal{N}$, with equality if policy u gives complete priority to S-jobs.*

(b) $\sum_{i \in \mathcal{N}} \bar{A}_i^{\mathcal{N}} W_i^u = \bar{b}(\mathcal{N})$.

PROOF. By applying Eq. (48) for relating C^u with W^u, we obtain

$$(61) \qquad \sum_{j \in S} \bar{A}_i^S W_i^u = C^{u\prime}(I - E[N]) \begin{pmatrix} E[T_S^S] \\ 0 \end{pmatrix} - g_S' E[T_S^S]$$

$$= C^{u\prime} \begin{pmatrix} (I_S - E[N_{SS}]) E[T_S^S] \\ -E[N_{S^cS}] E[T_S^S] \end{pmatrix} + g_S' E[T_S^S]$$

$$= C^{u\prime} \begin{pmatrix} E[v_S] \\ E[v_{S^c}] - E[T_{S^c}^S] \end{pmatrix} + g_S' E[T_S^S]$$

$$(62) \qquad = b(\mathcal{N}) - \sum_{i \in S^c} A_i^{S^c} C_i^u + \sum_{j \in S} g_j E[T_j^S],$$

where (61) follows from the results in §5.2.6 below. Now, by the conservation laws satisfied by C^u (see Theorem 9) the result follows. □

COROLLARY 4. *The performance region for branching bandits corresponding to performance vector W^u is the extended contra-polymatroid $\mathcal{P}_c(\bar{A}, \bar{b})$; furthermore, the vertices of $\mathcal{P}_c(\bar{A}, \bar{b})$ are the performance vectors corresponding to static-priority rules.*

5.2.5. *Optimal solution.* From Eqs. (46) and (49) and the conservation laws satisfied by C^u and W^u we obtain two different algorithms for solving tax problem (43): the first one corresponds to running adaptive greedy algorithm $\mathcal{AG}$ with input $(-h, \bar{A})$, and it is a bottom-up algorithm (i.e., priorities are computed from lowest to highest); the second one corresponds to running algorithm $\mathcal{AG}$ with input $(\hat{r}, A)$, where

$$(63) \qquad \hat{r}_i = h_i - \sum_{j \in \mathcal{N}} E[N_{ij}] h_j,$$

and it is a top-down algorithm (i.e., priorities are computed from highest to lowest).

5.2.6. *Parameter computation.* In this section we show how to compute matrix A and set function b from the branching model data. Recall that

$$A_i^S = E\left[T_i^{S^c}\right], \quad \text{for } i \in S, \text{ and}$$

$$b(S) = \frac{1}{2} E\left[\left(T_{L(0)}^{\mathcal{N}}\right)^2\right] - \frac{1}{2} E\left[\left(T_{L(0)}^{S^c}\right)^2\right] + \sum_{i \in S} E[\nu_i] \left(E[v_i] E\left[T_i^{S^c}\right] - \frac{1}{2} E\left[\left(T_i^{S^c}\right)^2\right] \right).$$

From Eq. (36) we obtain, taking expectations:

$$(64) \qquad E\left[T_i^S\right] = E[v_i] + \sum_{j \in S} E[N_{ij}] E\left[T_j^S\right], \quad \text{for } i \in \mathcal{N}.$$

Solving this linear system we obtain $E[T_i^S]$. Note that the computation of A_i^S is much easier in the undiscounted case compared with the discounted case, where we had to solve a system of nonlinear equations. Also, applying the conditional variance formula to (36) we obtain:

$$\operatorname{Var}[T_i^S] = \operatorname{Var}[v_i] + \left(E[T_j^S]\right)'_{j\in S} \operatorname{Cov}\left[(N_{ij})_{j\in S}\right]\left(E[T_j^S]\right)_{j\in S}$$
$$+ \sum_{j\in S} E[N_{ij}]\operatorname{Var}[T_j^S], \qquad i \in \mathcal{N}.$$

Solving this linear system we obtain $\operatorname{Var}[T_i^S]$ and thus $E[(T_i^S)^2]$. Moreover, the expected number of class j jobs serviced during the busy period, $E[\nu_j]$, for $j \in \mathcal{N}$, can be obtained by solving the linear system

$$E[\nu_j] = L_j(0) + \sum_{i\in\mathcal{N}} E[N_{ij}]E[\nu_i], \quad \text{for } j \in \mathcal{N}.$$

Furthermore, from Eq. (40) we obtain

$$E\left[T_{L(0)}^S\right] = \sum_{i\in S} L_i(0)E\left[T_i^S\right], \quad \text{and}$$
$$\operatorname{Var}\left[T_{L(0)}^S\right] = \sum_{i\in S} L_i(0)\operatorname{Var}\left[T_i^S\right].$$

For computing $\bar{b}(S)$ the quantity $E[v_i N_{ij}]$ is needed. It is easy to see that

$$E\left[v_i N_{ij}\right] = \frac{\partial}{\partial\alpha\,\partial z_j}\Phi_i(1, 1_{\mathcal{N}}).$$

5.3. *Summary.* Table 3 summarizes the problems we considered, the performance measures used, the conservation laws, the corresponding performance region, as well as the input to algorithm $\mathcal{AG}$.

6. Special cases. In this section we present new linear programming formulations for several classical stochastic scheduling problems, by specializing the results of §5 for branching bandit problems. We show how to model each problem as a branching bandit problem, then characterize explicitly its performance region, and finally derive an optimal solution algorithm. The problems we address are: the multiarmed bandit problem, the problem of optimal dynamic scheduling of a multiclass queue with feedback, and the problem of scheduling a fixed batch of jobs.

TABLE 3
Branching Bandit Problems: Formulation and Solution

Problem	Performance measure	Performance region	Indices
$\max_{u\in\mathcal{U}} Z_u^{(r,h)}(\alpha)$	$\lambda_j^u(\alpha)$	$\mathrm{P}_c(A_\alpha, b_\alpha)$	$(r_\alpha, A_\alpha) \overset{\mathcal{AG}}{\mapsto} \gamma(\alpha)$
		A_α, b_α: see (38), (39)	r_α: see (22)
$\max_{u\in\mathcal{U}} Z_u^h$	C_j^u	$\mathcal{P}(A, b)$	$(\hat{r}, A) \overset{\mathcal{AG}}{\mapsto} \gamma$
		A, b: see (50), (51)	$\hat{r}$: see (63)
	W_j^u	$\mathcal{P}_c(\bar{A}, \bar{b})$	$(-h, \bar{A}) \overset{\mathcal{AG}}{\mapsto} \gamma$
		$\bar{A}, \bar{b}$: see (59), (60)	

6.1. *The multiarmed bandit problem*

6.1.1. *Problem definition.* The multiarmed bandit problem can be described as follows: There are K parallel projects, indexed $k = 1, \dots, K$. Project k can be in one of a finite number of states $j_k \in \mathcal{N}_k$. At each instant of discrete time $t = 0, 1, \dots$ one must work on exactly one project. If one works on project k in state $j_k(t)$ at time t, then an immediate reward of $r^k_{j_k(t)}$ is earned. Rewards are additive and are 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 $P^k = (p^k_{ij})_{i,j \in \mathcal{N}_k}$, while the states of the projects one has not engaged remain unchanged. The problem is to find an admissible scheduling policy $u \in \mathcal{U}$ (the class of admissible policies $\mathcal{U}$ consists of all nonanticipative and nonidling policies) that selects at each time a project to engage in order to maximize the total expected discounted reward earned over an infinite horizon,

$$Z = \max_{u \in \mathcal{U}} E_u \left[\sum_{t=0}^{\infty} \beta^t r^{k(t)}_{j_k(t)} \right].$$

6.1.2. *Modeling as a branching bandit problem.* We shall model the problem as a discounted branching bandit problem in order to apply the results of §5.1. We thus identify project states with job classes, in such a way that working on a project in state i corresponds to serving a class-i job. Using the notation of §5.1, we define the set of job classes to be $\mathcal{N} = \cup_{k=1}^{K} \mathcal{N}_k$. All jobs have unit processing times, so that $v_i \equiv 1$, for $i \in \mathcal{N}$. The continuous-time discount factor α is related to β by $e^{-\alpha} = \beta$. Let us define matrix $P = (p_{ij})_{i,j \in \mathcal{N}}$ by

$$p_{ij} = \begin{cases} p^k_{ij}, & \text{if } i, j \in \mathcal{N}_k, \text{ for some } k = 1, \dots, K; \\ 0, & \text{otherwise,} \end{cases}$$

and vector $r = (r_i)_{i \in \mathcal{N}}$ by $r_i = r^i_k$, if $i \in \mathcal{N}_k$, for $k = 1, \dots, K$. The vector of class-j descendants $(N_{ij})_{j \in \mathcal{N}}$ of a class-i job is a multinomial random variable, as a class-i job has only one descendent, which is of class-j with probability p_{ij}. The reward earned upon completion of a class-i job is r_i.

Let us also define the discrete-time indicator

$$I_j(t) = \begin{cases} 1, & \text{if a project in state } j \text{ is engaged at time } t; \\ 0, & \text{otherwise.} \end{cases}$$

The performance measure $\lambda^u(\alpha) = (\lambda^u_j(\alpha))_{i \in \mathcal{N}}$ for discounted branching bandits given by (19) simplifies in this case into

$$\lambda^u_j(\alpha) = \beta E_u \left[\sum_{t=0}^{\infty} I_j(t) \beta^t \right], \quad \text{for } j \in \mathcal{N}.$$

We may interpret $\lambda_j(\alpha)$ as the total expected discounted time spent working on a project in state j under policy u.

In terms of these performance measures, the multiarmed bandit problem can be written as

$$\beta Z = \max_{u \in \mathcal{U}} \sum_{i \in \mathcal{N}} r_i \lambda^u_i(\alpha).$$

6.1.3. *Performance region.* By Theorem 7, we know that performance measure $\lambda^u(\alpha)$ satisfies generalized conservation laws, and that the performance region it spans is the extended contra-polymatroid $\mathscr{P}_c(A_\alpha, b_\alpha)$, where A_α and b_α are defined in (24) and (25), respectively. We shall show next how to compute A_α and b_α explicitly from the data of the multiarmed bandit model.

Parameter computation. For $S \subseteq \mathscr{N}$, let vector $t_S^S = (t_i^S)_S$ be defined as the solution of the linear system

$$t_i^S = 1 + \beta \sum_{j \in S} p_{ij} t_j^S, \quad \text{for } i \in S. \tag{65}$$

Let us also define

$$L_j(0) = \begin{cases} 1, & \text{if there is a project in state } j \text{ at time } t = 0; \\ 0, & \text{otherwise.} \end{cases}$$

Proposition 7. *Matrix A_α and set function b_α are given by the following expressions:*
(a)

$$A_{i,\alpha}^S = \frac{1-\beta}{\beta}\left(1 + \beta \sum_{j \in S^c} p_{ij} t_j^{S^c}\right), \quad \textit{for } i \in S;$$

(b)

$$b_\alpha(S) = \prod_{j \in S^c} \left(1 - (1-\beta) t_j^{S^c}\right)^{L_j(0)}, \quad \textit{for } S \subseteq \mathscr{N}$$

Proof. We have, by Eq. (35),

$$\begin{aligned} \Phi_i(\alpha, z_1, \dots, z_n) &= E\left[e^{-\alpha v_i} z_1^{N_{i1}} \dots z_n^{N_{in}}\right] \\ &= e^{-\alpha} \sum_{j \in \mathscr{N}} p_{ij} z_j \\ &= \beta \sum_{j \in \mathscr{N}_k} p_{ij} z_j, \quad \text{for } i \in \mathscr{N}_k \end{aligned} \tag{66}$$

and, by (37),

$$\begin{aligned} \Psi_i^S(\alpha) &= \Phi_i\left(\alpha, \left(\Psi_j^S(\alpha)\right)_{j \in S}, 1_{S^c}\right) \\ &= \beta\left(\sum_{j \in S} p_{ij} \Psi_j^S(\alpha) + \sum_{j \in S^c} p_{ij}\right) \\ &= \beta\left(1 - \sum_{j \in S} p_{ij}\left(1 - \Psi_j^S(\alpha)\right)\right), \quad \text{for } i \in \mathscr{N}. \end{aligned} \tag{67}$$

Now, it follows from (67) and (65) that

$$t_i^S = \frac{1 - \Psi_i^S(\alpha)}{1 - \Psi_i(\alpha)}, \quad \text{for } i \in S,$$

and part (a) follows by (67) and Proposition 6. Moreover, since $\Psi_j^{\mathcal{N}}(\alpha) = 0$, Proposition 6 yields

$$(68) \qquad b_\alpha(S) = \prod_{j \in S^c} \left[\Psi_j^{S^c}(\alpha)\right]^{L_j(0)}$$

$$= \prod_{j \in S^c} \left(1 - (1 - \beta) t_j^{S^c}\right)^{L_j(0)},$$

which proves (b). □

Proposition 7 together with Theorem 7 and Corollary 2 yield directly the following result.

PROPOSITION 8 (PERFORMANCE REGION FOR MULTIARMED BANDITS). *The performance region spanned by performance vectors $\lambda^u(\alpha)$ in the multiarmed bandit problem is the extended contra-polymatroid defined by*

$$(69) \quad \sum_{i \in S} \left(1 + \beta \sum_{j \in S^c} p_{ij} t_j^{S^c}\right) \lambda_i \geq \frac{\beta}{1 - \beta} \prod_{j \in S^c} \left(1 - (1 - \beta) t_j^{S^c}\right)^{L_j(0)}, \quad \textit{for } S \subset \mathcal{N},$$

$$\sum_{i \in \mathcal{N}} \lambda_i = \frac{\beta}{1 - \beta},$$

$$\lambda_i \geq 0, \quad \textit{for } i \in \mathcal{N}.$$

Furthermore, the performance vector $\lambda^u(\alpha)$ corresponding to a scheduling policy that gives complete priority to projects with states in S^c achieves equality in (69).

6.1.4. *Optimal solution.* Gittins and Jones (1974) first showed that the optimal policy for the multiarmed bandit problem is a priority-index policy. The optimal priority indices can be computed by running adaptive greedy algorithm $\mathcal{AG}$ with input (r, A_α).

Index decomposition. Gittins and Jones (1974) further showed that the optimal indices associated with states of a project only depend on characteristics of that project (states, rewards and transition probabilities).

THEOREM 11 (GITTINS AND JONES 1974). *For each project k there exist indices $\{\gamma_i^k\}_{i \in \mathcal{N}_k}$, depending only on characteristics of that project, such that an optimal policy is to work at each time on a project with largest current index.*

This classical result follows in our framework as a consequence of Theorem 3 on the decomposition of optimal indices. In particular, the structure of matrix $P = (p_{ij})$ implies that

$$A_{j,\alpha}^S = A_{j,\alpha}^{S \cap \mathcal{N}_k}, \quad \text{for } j \in S \cap \mathcal{N}_k,$$

so that Assumption 2 holds.

By the results of §5.1.5 we know that the allocation indices for this bandit problem are precisely the dynamic allocation indices introduced by Gittins and Jones (1974) (also called Gittins indices). Furthermore, by definition of allocation indices, we obtain a characterization of Gittins indices as sums of dual variables, a purely algebraic characterization. By Theorem 6, the Gittins indices can be computed by solving K subproblems, applying adaptive greedy algorithm $\mathcal{AG}$, presented in §3, to subproblem k, which has $|\mathcal{N}_k|$ job classes, for $k = 1, \ldots, K$.

The index-computing algorithm proposed by Varaiya, Walrand and Buyukkoc (1985) has the same time complexity as adaptive greedy algorithm $\mathscr{AG}$. In fact, both algorithms are closely related, as we will see next. Let t_i^S be as given by (65). Let r_i^S be given by

$$r_i^S = r_i + \beta \sum_{j \in S} p_{ij} r_j^S, \quad \text{for } i \in S.$$

The algorithm of Varaiya, Walrand and Buyukkoc can be stated as follows:

Algorithm $\mathscr{VWB}$:

Step 0. Pick $\pi_1 \in \operatorname{argmax}\{r_i^{\{i\}}/t_i^{\{i\}}\colon i \in \mathscr{N}\}$; let $g_{\pi_1} = \max\{r_i^{\{i\}}/t_i^{\{i\}}\colon i \in \mathscr{N}\}$; set $J_1 = \{\pi_1\}$.

Step k. For $j = 2, \dots, n$:

$$\text{pick } \pi_j \in \operatorname{argmax}\left\{\frac{r_i^{J_{j-1} \cup \{i\}}}{t_i^{J_{j-1} \cup \{i\}}} : i \in \mathscr{N} \setminus J_{j-1}\right\}; \text{ set } g_{\pi_{j-1}} = \left\{\frac{r_i^{J_{j-1} \cup \{i\}}}{t_i^{J_{j-1} \cup \{i\}}} : i \in \mathscr{N} \setminus J_{j-1}\right\};$$

$$\text{set } J_j = J_{j-1} \cup \{\pi_j\}.$$

Varaiya et al. (1985) proved that $g_1, \dots, g_n$, as given by algorithm $\mathscr{VWB}$, are the Gittins indices of the multiarmed bandit problem. Let $(\pi, \bar{y}, \gamma)$ be an output of algorithm $\mathscr{AG}$. The following relation between algorithms $\mathscr{AG}$ and $\mathscr{VWB}$ can be easily seen to hold by induction:

PROPOSITION 9. *The following relations hold: For $j = 1, \dots, n-1$,*

$$\frac{r_i^{\{\pi_1, \dots, \pi_j\} \cup \{i\}}}{t_i^{\{\pi_1, \dots, \pi_j\} \cup \{i\}}} - \frac{r_i - \sum_{l=1}^{j} A_{i,\alpha}^{\{\pi_l, \dots, \pi_n\}} \bar{y}^{\{\pi_l \pi_l, \dots, \pi_n\}}}{A_{i,\alpha}^{\{\pi_{j+1}, \dots, \pi_n\}}}$$

$$\equiv \frac{r_{\pi_j}^{\{\pi_1, \dots, \pi_j\}}}{t_{\pi_j}^{\{\pi_1, \dots, \pi_j\}}}, \quad \textit{for } i \in \{\pi_{j+1}, \dots, \pi_n\}, \textit{ and}$$

$$\frac{r_i^{\{i\}}}{t_i^{\{i\}}} - \frac{r_i}{A_{i,\alpha}^{\mathscr{N}}} \equiv 0, \quad \textit{for } i \in \mathscr{N},$$

and therefore, algorithms $\mathscr{AG}$ and $\mathscr{VWB}$ are equivalent.

Algorithm $\mathscr{AG}$ thus provides a new *off-line* top-down method (i.e., priorities are computed from highest to lowest) for computing Gittins indices. As shown above, it has the same computational complexity as the algorithm of Varaiya et al. (the faster off-line algorithm for computing Gittins indices known). The algorithms presented by Chen and Katehakis (1986) and by Katehakis and Veinott (1987) are *on-line* methods (they compute the Gittins index on a given state).

6.1.5. *A closed formula for the optimal value function*; *submodularity.* In this section we present a closed formula for the optimal value of a multiarmed bandit problem. We apply that formula to provide a new proof that the optimal value of the problem, seen as a function of the subset of projects involved, is a submodular set function.

Given a multiarmed bandit problem with K projects, let us consider subproblem k, the one-armed bandit problem corresponding to project k, for $k = 1, \dots, K$. Let

$$\gamma_{i_k}^k, \quad \text{for } i_k \in \mathscr{N}_k$$

be the Gittins indices corresponding to subproblem (project) k. Let b^k be the set function corresponding to subproblem k as given by Eq. (68), i.e.,

$$b^k(S_k) = \prod_{j \in \mathscr{N}_k \setminus S_k} \left(1 - (1-\beta) t_j^{\mathscr{N}_k \setminus S_k}\right)^{L_j(0)}, \quad \text{for } S_k \subseteq \mathscr{N}_k.$$

PROPOSITION 10. *For each k, l:*

$$b(S_k \cup S_l) = b^k(S_k) b^l(S_l), \quad \textit{for } S_k \subseteq \mathscr{N}_k \textit{ and } S_l \subseteq \mathscr{N}_l.$$

PROOF OUTLINE. Using the fact that $t_j^S = t_j^{S \cap \mathscr{N}_k}$, if $j \in \mathscr{N}_k$ the result follows trivially. □

Now, in order to solve the maximum reward problem, we run algorithm $\mathscr{AG}$ with input (r, A_α). Let π be a permutation of $\mathscr{N}$ produced by the algorithm. If $\{\gamma_i\}_{i \in \mathscr{N}}$ are the corresponding Gittins indices, π must satisfy $\gamma_{\pi_n} \leq \cdots \leq \gamma_{\pi_1}$. Permutation π of $\mathscr{N}$ induces permutations π^k of $\mathscr{N}_k$, for $k = 1, \dots, K$.

If $\{\gamma_i^k\}_{i \in \mathscr{N}_k}$ are the Gittins indices corresponding to subproblem k, π^k must satisfy $\gamma_{\pi^k_{|\mathscr{N}_k|}} \leq \cdots \leq \gamma_{\pi^k_1}$.

Let us define independent random variables η, $\eta_k \in \mathscr{N}_k$, for $k = 1, \dots, K$, by

$$P\{\eta \in \{\pi_1, \dots, \pi_i\}\} = b(\{\pi_1, \dots, \pi_i\}), \quad \text{for } i = 1, \dots, n, \text{ and}$$

$$P\{\eta_k \in \{\pi_1^k, \dots, \pi_i^k\}\} = b^k(\{\pi_1^k, \dots, \pi_i^k\}), \quad \text{for } i = 1, \dots, |\mathscr{N}_k|.$$

Given a subset of projects $H \subseteq \{1, \dots, K\}$, let $Z(H)$ denote the optimal reward in the multiarmed bandit problem obtained when only projects in subset H are available. We have the following result:

THEOREM 12 (OPTIMAL REWARD OF THE MULTIARMED BANDIT PROBLEM). *The optimal reward $Z(H)$ can be expressed as*

$$Z(H) = E\left[\max_{k \in H} \gamma_{\eta_k}^k\right]. \tag{70}$$

PROOF. Since projects can be aggregated, it is enough to prove the theorem for the case of two projects, i.e., $K = 2$. First, notice that by Eq. (70), the expression for the optimal objective value of a linear program over an extended contra-polymatroid, and the characterization of the performance region of the multiarmed bandit as an extended polymatroid, we obtain:

$$Z(\{1,2\}) = (\gamma_{\pi_1}, \gamma_{\pi_2}, \dots, \gamma_{\pi_n}) \begin{pmatrix} b(\{\pi_1, \dots, \pi_n\}) - b(\{\pi_2, \dots, \pi_n\}) \\ \vdots \\ b(\{\pi_{n-1}, \pi_n\}) - b(\{\pi_n\}) \\ b(\{\pi_n\}) \end{pmatrix}$$

$$= E[\gamma_\eta].$$

Now, let $H = \{1, 2\}$. In order to prove that Eq. (70) holds, it is enough to show that the following two random variables have the same distribution:

$$\max\left(\gamma_{\eta_1}^1, \gamma_{\eta_2}^2\right) \overset{\mathrm{d}}{=} \gamma_\eta. \tag{71}$$

We have, for a given γ_{i^*}:

$$(72)\quad P\left\{\max\left(\gamma^1_{\eta_1}, \gamma^2_{\eta_2}\right) \le \gamma_{i^*}\right\} = P\left\{\gamma^1_{\eta_1} \le \gamma_{i^*}\right\}P\left\{\gamma^2_{\eta_2} \le \gamma_{i^*}\right\}$$
$$= b^1\left(\left\{i_1 \in \mathcal{N}_1 : \gamma^1_{i_1} \le \gamma_{i^*}\right\}\right)b^2\left(\left\{i_2 \in \mathcal{N}_2 : \gamma^2_{i_2} \le \gamma_{i^*}\right\}\right)$$
$$= b(\{i \in \mathcal{N} : \gamma_i \le \gamma_{i^*}\})$$
$$= P\{\gamma_\eta \le \gamma_{i^*}\}.$$

Hence, the equality in distribution (71) holds, and the result follows. □

COROLLARY 5 (SUBMODULARITY OF THE OPTIMAL REWARD FUNCTION). *The optimal reward function $Z(H)$ of the multiarmed bandit problem is a submodular set function.*

PROOF. Since the function $H \mapsto \max_{k \in H} d_k$ is submodular, for any given vector of d_ks the result follows directly from formula (70). □

The fact that the optimal reward function of the multiarmed bandit problem is submodular was first shown by Weber (1992), who proved it from first principles. Tsitsiklis (1986) provided an early result in this direction.

6.2. *Scheduling control of a multiclass queue with Bernoulli feedback*

6.2.1. *System description.* A multiclass $M/G/1$ queue with Bernoulli feedback can be described as follows: A single server provides service to jobs, which are classified in a finite number n of classes. External arrivals of class-i jobs follow a Poisson process of rate λ_i, for $i \in \mathcal{N} = \{1, \dots, n\}$. Service times for class-i jobs are independent and identically distributed as a random variable v_i with distribution function G_i. When the service of a class-i job is completed, the job may either join the queue of class-j jobs, with probability p_{ij} (thus becoming a class-j job) or, with probability $1 - \sum_{j \in \mathcal{N}} p_{ij}$, leave the system.

The server must select the jobs for service according to an admissible scheduling policy $u \in \mathcal{U}$, which must be nonidling, nonpreemptive and nonanticipative; the decision epochs are $t = 0$ (if there is initially some job present), the times at which a job arrives to find the system empty, and the times at which a job completes service.

Klimov (1974) solved, by direct methods, the associated optimal control problem with a time-average holding cost criterion. Harrison (1975a) solved, using dynamic programming, the optimal control problem with a discounted reward-cost criterion, in the special case that there is no job feedback. Tcha and Pliska (1977) extended Harrison's results to the case with feedback.

6.2.2. *Modeling as a branching bandit.* The first busy period of a multiclass queue with Bernoulli feedback, as described above, can easily be modeled as a branching bandit. The set $\mathcal{N}$ of job classes, and the service times v_i of class-j jobs are as in the corresponding queueing system. The class-j descendants N_{ij} of a class-i job correspond to the number of external class-j job arrivals, along with the internal job transfers to class-j, occurring during the service of a class $- i$ job.

6.2.3. *Discounted reward-cost objective.* Let us consider the following reward-cost structure: a continuous holding cost αh_i is incurred per unit time that a class class-i job stays in the system. Furthermore, an instantaneous reward r_i is earned at the service completion epoch of a class-i job. All costs and rewards are continuously discounted in time by a discount factor $\alpha > 0$.

The optimal scheduling problem is to find an admissible policy for assigning the server to the jobs demanding its service, in order to maximize the total expected discounted value of the rewards earned minus the holding costs incurred over an infinite horizon.

6.2.4. *Performing region and optimal solution.* For a given $\alpha > 0$, we define performance measure $\lambda_j^u(\alpha)$ of class-j jobs under policy u to be total expected discounted number of class-j job service completions, i.e.,

$$\lambda_j^u(\alpha) = E_u\left[\sum_{k=1}^{\nu_j} e^{-\alpha(\tau_{jk}+v_{jk})}\right], \quad \text{for } j \in \mathscr{N},$$

exactly as we did in our treatment of the discounted branching bandit problem.

The optimal value of the problem $Z_u^{(r,h)}(\alpha)$, can be written in terms of performance measure $\lambda^u(\alpha)$ as

$$Z_u^{(r,h)}(\alpha) = \sum_{i \in \mathscr{N}} (r_i + r_{i,\alpha})\lambda_i^u(\alpha) - \sum_{i \in \mathscr{N}} h_i L_i(0),$$

by (21), where vector $r_\alpha = (r_{i,\alpha})_{i \in \mathscr{N}}$ is given by (22).

By Theorem 7, we know that performance measure $\lambda^u(\alpha)$ satisfies generalized conservation laws, and that the performance region it spans is the extended contra-polymatroid $\mathscr{P}_c(A_\alpha, b_\alpha)$, where A_α and b_α are defined by (24) and (25), respectively. We shall show next how to compute A_α and b_α from the data of the multiarmed bandit model.

Parameter computation. In the notation introduced in §5.1, we have that transform $\Phi_i(\cdot)$ is given by

$$(73) \qquad \Phi_i(\alpha, z_1, \ldots, z_n) = E\left[e^{-\alpha v_i} z_1^{N_{i1}} \ldots z_n^{N_{in}}\right]$$
$$= E\left[\left(1 - \sum_{j \in \mathscr{N}} p_{ij}(1 - z_j)\right) e^{-v_i(\alpha + \sum_{j \in \mathscr{N}} \lambda_j(1 - z_j))}\right]$$
$$= \left(1 - \sum_{j \in \mathscr{N}} p_{ij}(1 - z_j)\right)\Psi_i\left(\alpha + \sum_{j \in \mathscr{N}} \lambda_j(1 - z_j)\right).$$

Therefore, by (37) and (73) we obtain that the values of $\Psi_i^S(\alpha)$, for $i \in \mathscr{N}$, satisfy the system of equations

$$\Psi_i^S(\alpha) = \left(1 - \sum_{j \in S} p_{ij}\left(1 - \Psi_j^S(\alpha)\right)\right)\Psi_i\left(\alpha + \sum_{j \in S} \lambda_j\left(1 - \Psi_j^S(\alpha)\right)\right), \quad \text{for } i \in \mathscr{N}.$$

Proposition 6, yields closed formulae for computing matrix A_α and set function b_α in terms of the $\Psi_i^S(\alpha)$s and the $\Psi_i(\alpha)$s. By the results in §5.1, the performance region spanned by performance vector $\lambda^u(\alpha)$ is the extended contra-polymatroid $\mathscr{P}_c(A_\alpha, b_\alpha)$, and it shown there how to apply adaptive greedy algorithm $\mathscr{AG}$ in order to compute the priority indices corresponding to the optimal scheduling policy.

6.2.5. *Undiscounted cost objective.* Klimov (1974) first considered the problem of optimal control of a single-server multiclass queue with Bernoulli feedback, with the criterion of minimizing the time-average holding cost. He established that the optimal nonidling, nonpreemptive and nonanticipative scheduling policy is a static-priority policy, and presented an algorithm for computing the optimal priorities. Tsoucas (1991) formulted Klimov's problem as a linear program over an extended polymatroid, using as performance measures

$$Q_i^u = \text{time-average length of queue } i \text{ under policy } u.$$

Adaptive greedy algorithm $\mathscr{AG}$ applied to this problem yields Klimov's original algorithm. A disadvantage is that Klimov's is a bottom-up algorithm: priorities are computed from lowest to highest. Also, Tsoucas does not obtain closed formulae for the right hand sides of the extended polymatroid inequalities, so it is not possible to evaluate the performance of an optimal policy. Our approach yields a top-down algorithm, gives explicit formulae for all the parameters of the extended polymatroid, and explains the somewhat surprising property that the optimal priorities do not depend on the arrival rates. The key observation for reducing the problem to a branching bandit problem is that an optimal policy under the time-average holding cost criterion must minimize the total expected holding cost in a busy period (see, e.g., Nain, Tsoucas and Walrand (1989)).

6.2.6. *Performance region.* By the results in §5.2 on the undiscounted branching bandit problem, we know that the performance regions spanned by the vector of expected system times, W^u (resp. expected completion times C^u) is the extended contra-polymatroid (resp. extended polymatroid) $\mathscr{P}_c(\overline{A}, \overline{b})$ (resp. $\mathscr{P}(A, b)$), where parameters $\overline{A}$, $\overline{b}$, A and b are as defined in §5.2. We show next how to compute such parameters from the model data.

It is easily seen that $E[N_{ij}] = p_{ij} + E[v_i]\lambda_j$. Therefore, by (64),

$$E\left[T_i^{S^c}\right] = E[v_i] + \sum_{j \in S^c} \left(p_{ij} + E[v_i]\lambda_j\right) E\left[T_j^{S^c}\right], \quad \text{for } i \in \mathscr{N},$$

which is vector notation is written as

$$E\left[T_{S^c}^{S^c}\right] = E[v_{S^c}] + (P_{S^cS^c} + E[v_{S^c}]\lambda'_{S^c}) E\left[T_{S^c}^{S^c}\right], \quad \text{i.e.,}$$

$$E\left[T_{S^c}^{S^c}\right] = (I_{S^c} - P_{S^cS^c} - E[v_{S^c}]\lambda'_{S^c})^{-1} E[v_{S^c}], \quad \text{and}$$

$$E\left[T_S^{S^c}\right] = E[v_S] + (P_{s,S^c} + E[v_S]\lambda'_{S^c}) E\left[t_{S^c}^{S^c}\right].$$

Let us define

$$K_{S^c} = \frac{\det(I_{S^c} - P_{S^cS^c})}{\det(I_{S^c} - P_{S^cS^c} - E[v_{S^c}]\lambda'_{S^c})}.$$

The following algebraic invariance relations can be shown to hold:

$$E\left[T_{S^c}^{S^c}\right] = K_{S^c}(I_{S^c} - P_{S^cS^c})^{-1} E[v_{S^c}], \quad \text{and}$$

$$E\left[T_S^{S^c}\right] = K_{S^c}\left(E[v_S] + P'_{S,S^c}(I_{S^c} - P_{S^cS^c})^{-1} E[v_{S^c}]\right).$$

Therefore, by definition of A_i^S in (50), we have that $A_i^S = E[T_i^{S^c}]$, for $i \in S$, while $b(S)$ is given by (51). Now, we may define $\hat{A}_i^S = A_i^S / K_{S^c}$, and $\hat{b}(S) = b(S)/K_{S^c}$, thus eliminating the dependence on the arrival of matrix $\hat{A}$. As for the objective function, by (49), and the fact that $E[N_{ij}] = p_{ij} + \lambda_j E[v_i]$ we obtain:

$$-Z_u^h = \sum_{i \in \mathcal{N}} \Big(h_i - \sum_{j \in \mathcal{N}} p_{ij} h_j \Big) C_i^u - b(\mathcal{N}) \sum_{j \in \mathcal{N}} h_j \lambda_j E[v_j] - \sum_{i \in \mathcal{N}} h_i g_i,$$

where h and b are as given in §5.2. Therefore the control problem can be solved by running adaptive greedy algorithm $\mathcal{AG}$ with input $(\hat{r}, \hat{A})$, where

$$\hat{r}_i = h_i - \sum_{j \in \mathcal{N}} p_{ij} h_j, \quad \text{for } i \in \mathcal{N},$$

and since $(\hat{r}, \hat{A})$ does not depend on the arrival rates neither does the optimal policy. Notice that in contrast to Klimov's algorithm, with this method priorities are computed from highest to lowest. This top-down algorithm was proposed by Lai and Ying (1988) and by Nain, Tsoucas and Walrand (1989), who proved its optimality using interchange arguments. Bhattacharya, Georgiadis and Tsoucas (1991) provided a direct optimality proof.

Notice that by modeling the busy period of Klimov's problem as a branching bandit's tax problem, using performance measure W^u, we obtain exactly Klimov's algorithm.

Moreover, in the case that the arriving jobs are divided into K projects, where a class-k project consists of jobs with classes in a finite set $\mathcal{N}_k$, jobs in $\mathcal{N}_k$ can only make transitions within $\mathcal{N}_k$, and $\mathcal{N}$ is partitioned as $\mathcal{N} = \cup_{k=1}^K \mathcal{N}_k$, then it is easy to see that the index decomposition theorem (Theorem 6) applies, and therefore we can decompose the problem into K smaller subproblems.

6.3. *Optimal scheduling without job arrivals; deterministic scheduling.* There is a batch of n jobs to be processed by a single server. Job i has a service requirement distributed as the random variable v_i, with Laplace transform Ψ_i. It is clear how to model this job scheduling process as a branching bandit process in which jobs have no descendants. Let us consider first the discounted case: For $\alpha > 0$ it is clear by definition of $A_{i,\alpha}^S$, in (24), that $A_{i,\alpha}^S \equiv \alpha$, for $i \in S$. Therefore the performance region of the vectors $\lambda^u(\alpha)$ studied in §5.1 is a polymatroid. Consider the discounted reward-tax problem discussed in §5.1, in which a instantaneous reward r_i is received at the completion of job i, and a holding tax αh_i is incurred for each unit of time that job i is in the system. Rewards and taxes are discounted in time with discount factor α. By (21) it follows that the allocation index for job i, in the problem of maximizing rewards minus taxes, is

$$\gamma_i(\alpha) = (r_i + h_i) \frac{\alpha \Psi_i(\alpha)}{1 - \Psi_i(\alpha)}.$$

Let us consider now the problem of minimizing the total weighted expected completion time of the jobs, where a holding cost h_i is incurred per unit of time that a class i job is in the system. By definition of $\overline{A}_i^S$ in (59), $\overline{A}_i^S \equiv E[v_i]$, for $i \in S$. Hence the performance region of the performance vectors $E[v_i] W_i^u$, where W_i^u is the expected system time of job i is the base of a polymatroid. Thus by Eq. (49) it follows that the allocation index for job i in the undiscounted tax problem is $\gamma_i = h_i / E[v_i]$. We thus obtain that for every nonanticipative, nonpreemptive and nonidling scheduling policy u,

$$(74) \qquad \sum_{i \in S} E[v_i] W_i^u \geq \frac{1}{2}\left(\left(\sum_{i \in S} E[v_i]\right)^2 + \sum_{i \in S} E[v_i]^2\right), \quad \text{for } S \subset \mathcal{N}, \text{ and}$$

$$\sum_{i \in \mathcal{N}} E[v_i] W_i^u = \frac{1}{2}\left(\left(\sum_{i \in \mathcal{N}} E[v_i]\right)^2 + \sum_{i \in \mathcal{N}} E[v_i]^2\right),$$

with equality in (74) if policy u gives priority to S-jobs. Queyranne (1993) characterized this performance region in the case that the processing times v_i are deterministic.

In the case that there are precedence constraints among the jobs that form out-forests, i.e., each job can have at most one immediate predecessor, the problem can also be modeled as a branching bandit problem. The formulations developed in this section apply therefore to it (see also Horn (1972) and Glazebrook (1976)).

7. Conclusions.

We presented a polyhedral treatment of several classical problems in stochastic and dynamic scheduling using polyhedral methods that leads, we believe, to a deeper understanding of their structural and algorithmic properties. Indeed, we hope that our results will be of interest to applied probabilists, as they provide new interpretations, proofs, algorithms, insights and connections to important problems in stochastic scheduling, as well as to discrete optimizers, since they reveal a new interesting structure (extended polymatroids), which has a genuinely applied origin.

Acknowledgements. A preliminary version of this paper appeared in the Proceedings of the Third Conference in Integer Programming and Combinatorial Optimization, Erice, Italy, 1993. The research of the first author was partially supported by a Presidential Young Investigator Award DDM-9158118 with matching funds from Draper Laboratory. The research of the second author was partially supported by a Ministry of Education & Science of Spain (MEC)/Fulbright Doctoral Fellowship.

References

Bazaraa, M. S. and C. M. Shetty (1979). *Nonlinear Programming: Theory and Algorithms*. Wiley, New York.

Bell, C. (1971). Characterization and computation of optimal policies for operating an $M/G/1$ queue with removable server. *Oper. Res.* **19** 208–218.

Bertsimas, D., I. Paschalidis and J. Tsitsiklis (1994a). Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance. *Ann. Appl. Probab.* **4** 43–75.

____ and C. Teo (1994). From valid inequalities to heuristics: A unified view of primal-dual approximation algorithms in covering problems. Working paper, Operations Research Center, MIT.

Bhattacharya, P. P., 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*, 9–11, 1991, Monterey, California.

____, ____ and ____ (1992). Extended polymatroids: Properties and optimization. In *Proceedings of Second International Conference on Integer Programming and Combinatorial Optimization*, Mathematical Programming Society, Carnegie Mellon University, 298–315.

Chen, Y. R. and M. N. Katehakis (1986). Linear programming for finite state multiarmed bandit problems. *Math. Oper. Res.* **11** 180–183.

Cobham, A (1954). Priority assignment in waiting line problems. *Oper. Res.* **2** 70–76.

Coffman, E. G., Jr., M. Hofri and G. Weiss (1989). Scheduling stochastic jobs with a two point distribution on two parallel machines. *Probab. Engrg. Inform. Sci.* **3** 89–116.

____ and I. Mitrani (1980). A characterization of waiting time performance realizable by single server queues. *Oper. Res.* **28** 810–821.

Cox, D. R. and W. L. Smith (1961). *Queues.* Methuen, London.

Edmonds, J. (1970). Submodular functions, matroids and certain polyhedra. In *Proceedings of Calgary International Conference on Combinatorial Structures and Their Applications*, R. Guy, H. Hanani, N. Sauer and J. Schnheim, eds., Gordon and Breach, New York, 69–87.

Federgruen, A. and H. Groenevelt (1988a). Characterization and optimization of achievable performance in general queueing systems. *Oper. Res.* **36** 733–741.

____ and ____ (1988b). $M/G/c$ queueing systems with multiple customer classes: Characterization and control of achievable performance under nonpreemptive priority rules. *Management Sci.* **34** 1121–1138.

Gelenbe, E. and I. Mitrani (1980). *Analysis and Synthesis of Computer Systems.* Academic Press, London.

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

____ (1989). *Multiarmed Bandit Allocation Indices.* Wiley, Chichester.

Glazebrook, K. D. (1976). Stochastic scheduling with order constraints. *Internat. J. Systems Sci.* **7** 657–666.

____ (1987). Sensitivity analysis for stochastic scheduling problems. *Math. Oper. Res.* **12** 205–223.

____ (1994). Reflections on a new approach to Gittins indexation. Working paper, Dept. of Mathematics and Statistics, Newcastle University, UK.

Grötschel, M., L. Lovász and A. Schrijver (1988) *Geometric Algorithms and Combinatorial Optimization,* Springer-Verlag, Berlin.

Harrison, J. M. (1975a). A priority queue with discounted linear costs. *Oper. Res.* **23** 260–269.

____ (1975b). Dynamic scheduling of a multiclass queue: Discount optimality. *Oper. Res.* **23** 270–282.

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

Ishikida, T. and P. Varaiya (1994). Multiarmed bandit problem revisited. *J. Optim. Theory Appl.* **83** 113–154.

Katehakis, M. N. and A. F. Veinott (1987). The multiarmed bandit problem: Decomposition and computation, *Math. Oper. Res.* **12** 262–268.

Klimov, G. P. (1974). Time sharing service systems I. *Theory Probab. Appl.* **19** 532–551.

Lai, T. L. and Z. Ying (1988). Open bandit processes and optimal scheduling of queueing networks. *Adv. Appl. Probab.* **20** 447–472.

Lawler, E. L., J. K Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys (1989). Sequencing and scheduling; algorithms and complexity. Report BS-R8909, Centre for Mathematics and Computer Science, Amsterdam.

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

Nain, P., P. Tsoucas and J. Walrand (1989). Interchange arguments in stochastic scheduling. *J. Appl. Probab.* **27** 815–826.

Niño-Mora, J. (1995). *Optimal Resource Allocation in a Dynamic and Stochastic Environment: A Mathematical Programming Approach.* PhD Dissertation, Sloan School of Management, MIT.

Queyranne, M. (1993). Structure of a simple scheduling polyhedron. *Math. Programming* **58** 263–285.

Ross, K. W. and D. D. Yao (1989). Optimal dynamic scheduling in Jackson networks. *IEEE Trans. Automat. Control* **34** 47–53.

Rothkopf, M. H. (1966a). Scheduling independent tasks on parallel processors. *Management Sci.* **12** 437–447.

____ (1966b). Scheduling with random service times. *Management Sci.* **12** 707–713.

Schrijver, A. (1986). *Theory of Linear and Integer Programming.* Wiley, Chichester.

Shanthikumar, J. G. and D. D. Yao (1992). Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Oper. Res.* **40** S293–299.

Shapley, L. S (1971). Cores of convex games. *Internat. J. Game Theory* **1** 11–26.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Res. Logist. Quart.* **3** 59–66.

Stidham, S., Jr. (1972). $L = \lambda W$: A discounted analog and a new proof. *Oper. Res.* **20** 1115–1126.

Tcha, D. W. and S. R. Pliska (1977). Optimal control of single-server queueing networks and multiclass $M/G/1$ queues with feedback. *Oper. Res.* **25** 248–258.

Tsitsiklis, J. N. (1986). A lemma on the multiarmed bandit problem. *IEEE Trans. Automat. Control* **31** 576–577.

____ (1994). A short proof of the Gittins index theorem. *Ann. Appl. Probab.* **4** 194–199.

Tsoucas, P. (1991). The region of achievable performance in a model of Klimov. Research Report RC16543, IBM T. J. Watson Research Center, Yorktown Heights, NY.

Varaiya, P. P., J. C. Walrand and C. Buyukkoc (1985). Extensions of the multiarmed bandit problem: The discounted case. *IEEE Trans. Automat. Control* **30** 426–439.

Weber, R. (1992). On the Gittins index for multiarmed bandits. *Ann. Appl. Probab.* **2** 1024–1033.

Weiss, G. (1988). Branching bandit processes. *Probab. Engrg. Inform. Sic.* **2** 269–278.

Whittle, P. (1980). Multiarmed bandits and the Gittins index. *J. Roy. Statist. Soc. Ser. B* **42** 143–149.

—— (1982). *Optimization Over Time: Dynamic Programming and Stochastic Control, Vols. I, II.* Wiley, Chichester.

D. Bertsimas: Sloan School of Management, Massachusetts Institute of Technology, 50 Memorial Drive, Cambridge, Massachusetts 02142-1347; e-mail: dbertsim@aris.mit.edu

J. Niño-Mora: Operations Research Center, Massachusetts Institute of Technology, 50 Memorial Drive, Cambridge, Massachusetts 02142-1347; e-mail: jnimora@mit.edu