Lagrangian relaxation and constraint generation for allocation and advanced scheduling

Yasin Gocgun* Archis Ghate†

August 22, 2011

Abstract

Diverse applications in manufacturing, logistics, health care, telecommunications, and computing require that renewable resources be dynamically scheduled to handle distinct classes of job service requests arriving randomly over slotted time. These dynamic stochastic resource scheduling problems are analytically and computationally intractable even when the number of job classes is relatively small. In this paper, we formally introduce two classes of problems called allocation and advanced scheduling, and formulate their Markov decision process (MDP) models. We establish that these MDPs are "weakly coupled" and exploit this structural property to develop an approximate dynamic programming method that uses Lagrangian relaxation and constraint generation to efficiently make good scheduling decisions. In fact, our method is presented for a general class of large-scale weakly coupled MDPs that we precisely define. Extensive computational experiments on hundreds of randomly generated test problems reveal that Lagrangian decisions outperform myopic decisions with a statistically significant margin. The relative benefit of Lagrangian decisions is much higher for advanced scheduling than for allocation scheduling.

*Sauder School of Business, University of British Columbia, Vancouver, Canada; yasin.gocgun@sauder.ubc.ca.
†Industrial and Systems Engineering, University of Washington, Seattle, USA; archis@uw.edu.

1 Introduction

In many operations management applications, multiple renewable resources need to be dynamically allocated to different types of job service requests arriving randomly over time. Time is often slotted, arriving requests are reviewed periodically, and jobs need to be serviced so as to optimize some system-performance metric. Depending on the application, job classes may be characterized by their arrival rates, resource requirements, revenue generated on completion, waiting and penalty costs, and more loosely, planner-assigned priorities. In some applications, arriving requests may either be served immediately or rejected; in other situations, arriving requests may be scheduled to future service slots in a booking horizon. The former is called *allocation* scheduling and the latter is termed *advanced* scheduling.

Both allocation and advanced scheduling are common in health care operations management. For example, in a diagnostic setting such as a CT or an MRI section, jobs correspond to patient requests for scans of different types (head, abdomen), of varying urgency (stat, routine), and from distinct patient-types (inpatient, outpatient); image scanners are the resources and time is typically slotted into fifteen or thirty-minute slots [15, 11, 16, 21]. In hospital capacity management, jobs relate to daily elective patient requests for hospital admissions, and resources correspond to equipment, hospital beds, and personnel such as residents, doctors, and nurses [9, 19, 26]. Patrick et al. [27] recently studied problems where patients of different types are scheduled to future days in a diagnostic facility. Similarly, in their extensive review of surgery scheduling practices, Erdogan and Denton [7] state that patient surgeries may be scheduled several days in advance. An advanced scheduling model of elective surgery scheduling is presented in [10].

Similar situations arise in applications beyond health care. In maintenance and repair facilities, jobs correspond to daily requests for inspecting or repairing different types of machinery whereas resources are personnel, and testing or repair equipment [28]. In a manufacturing system, jobs correspond to orders arriving daily for different products and resources include manufacturing equipment [23, 32]. In military applications such as reconnaissance and combat planning, jobs relate to requests for specific mission tasks with airplanes, ships, and crew as the resources [1]. In communication networks, jobs correspond to requests for data, voice, or video transmission and the resource is bandwidth [4, 30, 36]. In high performance computing facilities such as a supercomputer or a multi-processor machine with an operating system that allows dynamic resource reconfiguration, jobs correspond to computing tasks and resources to CPUs, memory, bandwidth, and storage space [35].

In this paper, we precisely define certain classes of allocation and advanced scheduling problems and build their Markov decision process (MDP) [5, 29, 31] models. These MDPs are intractable to exact solution methods. We therefore employ an approximation framework rooted in a key high-level structural observation that we make about these MDPs — the resource constraints are the only link that interconnects decisions made regarding different job classes. So if somehow these constraints were appropriately relaxed as a part of a computational approach, the resulting problem decomposition should help overcome the curse of dimensionality while searching for approximate solutions. This places our models within the class of *weakly coupled* MDPs and a recently developed Lagrangian relaxation technique that we review in the next section is *in principle* applicable [2, 18]. However, a linear program that needs to be solved in the key step of the Lagrangian relaxation method is intractable in our MDPs. We therefore develop a value function approximation and constraint generation technique to solve this linear program approximately. In fact, we present this approach in the more general context of *large-scale* weakly coupled MDPs that we precisely define in Section 3, and apply it to our allocation and advanced scheduling MDPs in Sections 4 and 5, respectively.

2 Background on weakly coupled MDPs

The key structural characteristic of a weakly coupled MDP is that it is composed of several independent sub-MDPs that are linked only through some linking constraints [2, 18]. In particular, consider an MDP with state-space $X = X^1 \times X^2 \ldots \times X^I$, where I is a positive integer and $X^1, X^2, \ldots, X^I$ are finite non-empty sets. We use $\mathcal{I}$ to denote the set $\{1, 2, \ldots, I\}$. For each $i \in \mathcal{I}$, let $U^i(x^i)$ be finite non-empty sets indexed by $x^i \in X^i$. Also, for each state $x = (x^1, x^2, \ldots, x^I) \in X$, let $U(x)$ equal the Cartesian product $U^1(x^1) \times U^2(x^2) \ldots \times U^I(x^I)$. Let J be a positive integer and we use $\mathcal{J}$ to denote the set $\{1, 2, \ldots, J\}$ in the sequel. Suppose $b \in \Re^J$. Let $g_i : (X^i \times U^i) \rightarrow \Re^J$ be functions defined for each $i \in \mathcal{I}$. The set of feasible actions in state $x \in X$ is given by

$$\bar{U}(x) = \Big\{(u^1, u^2, \ldots, u^I) \in U^1(x^1) \times U^2(x^2) \ldots \times U^I(x^I) : \sum_{i=1}^{I} g_i(x^i, u^i) \leq b\Big\}. \tag{1}$$

For all $x \in X$ and $u \in U(x)$, the immediate expected reward $r(x, u)$ is given by $r(x, u) = \sum_{i=1}^{I} r_i(x^i, u^i)$, where $r_i : (X^i \times U^i) \rightarrow \Re$ is called the immediate expected reward function for component $i \in \mathcal{I}$. That is, the immediate expected reward is additively separable. For any $x, x' \in X$ and $u \in U(x)$, the transition probability $p(x'; x, u)$ is multiplicatively separable. That is, $p(x'; x, u) = \prod_{i=1}^{I} p_i(x'^i; x^i, u^i)$, where $p_i(x'^i; x^i, u^i)$ is the probability that the ith component MDP makes a transition from state $x^i \in X^i$ to state $x'^i \in X^i$ on choosing action $u^i \in U^i(x^i)$ in state x^i.

For any $x \in X$, let $V(x)$ be the maximum infinite-horizon discounted expected reward accumulated starting in state x. Thus $V : X \rightarrow \Re$ is the optimal value function of this weakly coupled MDP. From the general theory of MDPs [29, 31], it is known that this optimal value function is the unique solution of Bellman's equations

$$V(x) = \max_{u \in \bar{U}(x)} \left(\sum_{i=1}^{I} r_i(x^i, u^i) + \alpha \sum_{x' \in X} \Big(\prod_{i=1}^{I} p_i(x'^i; x^i, u^i) \Big) V(x'^1, x'^2, \ldots, x'^I) \right), \quad \forall x \in X. \tag{2}$$

Feasible decisions that achieve the maximum in (2) define an optimal policy for the MDP. We use $E_{(x,u)}$ to denote an expectation with respect to the probability mass function of the random vector $\mathbf{X'}$ that represents the next state given that decision $u \in \bar{U}(x)$ was chosen in state $x \in X$. Then the above Bellman's equations can be rewritten more compactly as

$$V(x) = \max_{u \in \bar{U}(x)} \left(\sum_{i=1}^{I} r_i(x^i, u^i) + \alpha E_{(x,u)} V(\mathbf{X'}) \right), \quad \forall x \in X. \tag{3}$$

In most problems, exact solution of these equations is intractable because X and $\bar{U}(x)$ for $x \in X$ are both exponential in I.

Two approximate solution techniques have been proposed to alleviate this curse of dimensionality: the linear programming approach and the Lagrangian approach. The reader is referred to [2, 18] and references therein for more details on these. Both these methods yield approximations to the optimal value function V. More specifically, these approximations are statewise upper bounds on optimal values $V(x)$ for $x \in X$. The linear programming method provides a tighter bound but is computationally far more demanding than the Lagrangian method. In fact, when I is large, only the Lagrangian approach is computationally viable (see [2] for theoretical and empirical comparisons of the two methods). Thus our work here builds upon the Lagrangian method and we first briefly review it here.

2.1 The Lagrangian relaxation method

The Lagrangian approach essentially breaks the weakly coupled MDP into I sub-MDPs. This is achieved by relaxing the coupling constraints (1) using a vector $\lambda \in \Re_+^J$ of Lagrange multipliers and then adding the corresponding Lagrangian penalty function to the immediate expected reward. This is similar for instance to the Lagrangian relaxation method of integer programming [38], and results in the relaxed Bellman's equations

$$V^\lambda(x) = \max_{u \in U(x)} \left(\sum_{i=1}^{I} r_i(x^i, u^i) + \sum_{j=1}^{J} \lambda_j \left[b_j - \sum_{i=1}^{I} g_{ij}(x^i, u^i) \right] + \alpha E_{(x,u)} V^\lambda(\mathbf{X}') \right), \quad x \in X, \tag{4}$$

for approximate value functions V^λ. In addition, these approximate value functions are given by

$$V^\lambda(x) = \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \sum_{i=1}^{I} V_i^\lambda(x^i), \tag{5}$$

where the approximate componentwise value functions $V_i^\lambda(x^i)$, for each fixed $i \in \mathcal{I}$, are unique solutions of Bellman's equations

$$V_i^\lambda(x^i) = \max_{u^i \in U^i(x^i)} \Big(r_i(x^i, u^i) - \sum_{j=1}^{J} \lambda_j g_{ij}(x^i, u^i) + \alpha \sum_{x'^i \in X^i} p_i(x'^i; x^i, u^i) V_i^\lambda(x'^i) \Big), \quad x^i \in X^i. \tag{6}$$

We use $E_{(x^i,u^i)}$ to denote the expectation with respect to the probability mass function of the random variable $\mathbf{X}'^i$ representing the ith component of the next state given (x^i, u^i), to rewrite the componentwise Bellman's equations compactly as

$$V_i^\lambda(x^i) = \max_{u^i \in U^i(x^i)} \Big(r_i(x^i, u^i) - \sum_{j=1}^{J} \lambda_j g_{ij}(x^i, u^i) + \alpha E_{(x^i,u^i)} V_i^\lambda(\mathbf{X}'^i) \Big), \quad x^i \in X^i. \tag{7}$$

The relaxation gives a statewise bound on the optimal value function of the weakly coupled MDP; that is, for all $x \in X$, $V^\lambda(x) \geq V(x)$ for any $\lambda \geq 0$. Let $\beta(x) > 0$ be a sequence of positive real numbers indexed by states $x \in X$ such that $\sum_{x \in X} \beta(x) = 1$. The weighted values of the original MDP and its relaxation are defined as $H(\beta) = \sum_{x \in X} \beta(x) V(x)$ and $H^\lambda(\beta) = \sum_{x \in X} \beta(x) V^\lambda(x)$, respectively, and $H^\lambda(\beta) \geq H(\beta)$ for any $\lambda \geq 0$. Moreover, for each fixed $i \in \mathcal{I}$, let $\beta_i(x^i) > 0$ be real numbers indexed by state components $x^i \in X^i$ such that $\beta_i(x^i) = \sum_{\{x' : x'^i = x^i\}} \beta(x')$. Note that

$$\sum_{x^i \in X^i} \beta_i(x^i) = \sum_{x^i \in X^i} \sum_{\{x' : x'^i = x^i\}} \beta(x') = \sum_{x' \in X} \beta(x') = 1.$$

Also let $\mathcal{Y}_i = \big\{(x^i, u^i) : x^i \in X^i, u^i \in U^i(x^i)\big\}$. Then for each fixed $i \in \mathcal{I}$, the values $V_i^\lambda(x^i)$ for all $x^i \in X^i$ equal the optimal values of variables $\nu_i(x^i)$ in the linear program

$$\begin{aligned} &\min \sum_{x^i \in X^i} \beta_i(x^i) \nu_i(x^i) \\ &\nu_i(x^i) - \alpha E_{(x^i,u^i)} \nu_i(\mathbf{X}'^i) + \lambda^T g_i(x^i, u^i) \geq r_i(x^i, u^i), \text{ for } (x^i, u^i) \in \mathcal{Y}_i. \end{aligned}$$

After collating the above linear programs over all i, we obtain

$$H^{\lambda}(\beta) = \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \min \sum_{i=1}^{I} \sum_{x^i \in X^i} \beta_i(x^i)\nu_i(x^i)$$

$$\nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{j=1}^{J} \lambda_j g_{ij}(x^i,u^i) \geq r_i(x^i,u^i),\ (x^i,u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I}.$$

In order to identify a $\lambda \geq 0$ that yields the tightest upper bound on $H(\beta)$, an outer minimization over λ is performed to get the Lagrangian linear program

$$(LLP)\ H^{\lambda^*}(\beta) \triangleq \min \frac{\lambda^T b}{1-\alpha} + \sum_{i=1}^{I} \sum_{x^i \in X^i} \beta_i(x^i)\nu_i(x^i)$$

$$\nu_i(x^i) - \alpha E_{(x^i,u^i)}\nu_i(\mathbf{X}'^i) + \sum_{j=1}^{J} \lambda_j g_{ij}(x^i,u^i) \geq r_i(x^i,u^i),\ (x^i,u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I},$$

$$\lambda \geq 0.$$

The numbers of variables and constraints in (LLP) are $J + \sum_{i=1}^{I} \left|X^i\right|$ and $\sum_{i=1}^{I} \sum_{x^i \in X^i} \left|U^i(x^i)\right|$, respectively. In particular, they are both only *linear* in I. Thus (LLP) can be solved efficiently in some cases as for example in bandit-like problems [2] and the scheduling problems in [12].

Solving (LLP) gives us the optimal values λ^* and $V_i^{\lambda^*}(x^i)$ of the variables in (LLP). These are then plugged into (5) to obtain the approximate value function $V^{\lambda^*}(x)$. The Lagrangian policy is defined as a policy that is myopic with respect to this value function approximation. That is, it is constructed, in theory, by plugging the value function approximation into the right-hand side of (3) and solving the resulting static optimization problem, for every $x \in X$. Performing this calculation for every $x \in X$ is however intractable owing to the size of X. Fortunately, in practice, a whole policy, that is, a decision vector $u \in \bar{U}(x)$ for every $x \in X$, is not needed *a priori* — it suffices to select a decision vector in a particular state only if and when the system reaches that state during an actual system run. Nevertheless, it is often non-trivial to retrieve even one decision vector because that involves solving a challenging deterministic combinatorial optimization problem. This is typically achieved by designing an efficient, problem-specific algorithm. For example, dynamic programming was used for this purpose in [12].

In this paper, we introduce a class of weakly coupled MDPs where the sub-MDPs are themselves computationally intractable and hence the decomposition achieved by the Lagrangian method is insufficient for efficient solution of (LLP). We therefore approximate the componentwise value functions in (LLP) using a linear combination of basis functions and use constraint generation to tackle the resulting linear program as described in the next section.

3 Large-scale weakly coupled MDPs

We now define a class of weakly coupled MDPs where, for each $i \in \mathcal{I}$, the sets X^i and $U^i(x^i)$ grow exponentially in some positive integers K_i and L_i, respectively. In the sequel, we use $\mathcal{K}_i$ to denote the set $\{1, 2, \ldots, K_i\}$. For each $i \in \mathcal{I}$, let X_k^i be finite non-empty sets for $k \in \mathcal{K}_i$, and suppose $X^i \subseteq X_1^i \times X_2^i \times \ldots \times X_{K_i}^i$. That is, state x^i is itself given by $x^i = (x_1^i; x_2^i; \ldots; x_{K_i}^i) \in X^i$, where

$x^i_k \in X^i_k$. We use $\mathcal{L}_i$ to denote the sets $\{1, 2, \ldots, L_i\}$ for $i \in \mathcal{I}$. For $i \in \mathcal{I}$ and $x^i \in X^i$, let $U^i_l(x^i)$ be finite non-empty sets for $l \in \mathcal{L}_i$, and suppose that $U^i(x^i) \subseteq U^i_1(x^i) \times U^i_2(x^i) \times \ldots \times U^i_{L_i}(x^i)$. The decision u^i is itself given by $u^i = (u^i_1; u^i_2; \ldots; u^i_{L_i}) \in U^i(x^i)$, where $u^i_l \in U^i_l(x^i)$. Our allocation and advanced scheduling MDPs in Sections 4 and 5 have this structure. For this class of weakly coupled MDPs, the numbers of variables and constraints in (LLP), although linear in I, are exponential in both K_i and L_i for each i. Thus, solving (LLP) is computationally intractable when either the K_i's or the L_i's are large. To simplify (LLP), we use a value function approximation defined by a linear combination of "basis functions." That is, we set $\nu_i(x^i) \approx \gamma_i + \sum_{k=1}^{K_i} c_{ik}\nu_{ik}(x^i_k)$, where $\nu_{ik} : X^i_k \to \Re$ are functions fixed *a priori*, and $c_{ik} \geq 0$ and γ_i are unknown coefficients. This approximates (LLP) further to

$$
\begin{aligned}
(ALLP)\ H \triangleq \min\ & \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \sum_{i=1}^{I} \gamma_i + \sum_{i=1}^{I}\sum_{k=1}^{K_i} \Big(\sum_{x^i \in X^i} \beta_i(x^i)\nu_{ik}(x^i_k) \Big) c_{ik} \\
(1-\alpha)\gamma_i + \sum_{k=1}^{K_i} \Big(\nu_{ik}(x^i_k) - \alpha E_{(x^i,u^i)}\nu_{ik}(\mathbf{X'}^i_k) \Big) c_{ik} + \sum_{j=1}^{J} \lambda_j g_{ij}(x^i, u^i) &\geq r_i(x^i, u^i),\ (x^i, u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I}, \\
\lambda &\geq 0, \\
c_{ik} &\geq 0,\ i \in \mathcal{I},\ k \in \mathcal{K}_i.
\end{aligned}
$$

The number of variables in this linear program is only $J + \sum_{i \in \mathcal{I}} (K_i + 1)$, but the size of the set $\mathcal{Y}_i$ and hence the number of constraints is still exponential in K_i and L_i for each $i \in \mathcal{I}$. We therefore use constraint generation* to tackle (ALLP). Constraint generation begins with a linear program that includes only a subset $\mathcal{C}$ of the functional constraints and all non-negativity restrictions on the variables in (ALLP). This linear program is solved to find an optimal solution. The goal then is to find a violated functional constraint from (ALLP) and add it to $\mathcal{C}$. The new linear program is then solved and the procedure continues until either no violated constraints can be found or a stopping criterion is met. We present one way to choose a violated constraint from (ALLP).

We use (ALLP)$_\mathcal{C}$ to denote a relaxation of (ALLP) that includes functional constraints from $\mathcal{C}$. Also, γ and c denote the vectors formed by variables γ_i for $i \in \mathcal{I}$, and c_{ik} for $i \in \mathcal{I}$ and $k \in \mathcal{K}_i$, respectively. Suppose that after solving (ALLP)$_\mathcal{C}$, we obtain $\gamma^\mathcal{C}, c^\mathcal{C}, \lambda^\mathcal{C}$ as its optimal solution. The violation in the (ALLP) functional constraint corresponding to $i \in I$ and $(x^i, u^i) \in \mathcal{Y}_i$ is given by

$$
\mathcal{Z}_i(x^i, u^i) \triangleq r_i(x^i, u^i) - \sum_{j=1}^{J} \lambda^\mathcal{C}_j g_{ij}(x^i, u^i) - (1-\alpha)\gamma^\mathcal{C}_i - \sum_{k=1}^{K_i} \Big(\nu_{ik}(x^i_k) - \alpha E_{(x^i,u^i)}\nu_{ik}(\mathbf{X'}^i_k) \Big) c^\mathcal{C}_{ik}.
$$

We find a constraint from (ALLP) with the largest violation and add it to $\mathcal{C}$. This "most violated constraint" problem is given by†

$$
\mathcal{Z}^* \triangleq \max_{i \in \mathcal{I}} \Big(\max_{(x^i,u^i) \in \mathcal{Y}_i} \mathcal{Z}_i(x^i, u^i) \Big). \tag{8}
$$

Proposition 3.1 below provides a bound on H at each iteration of constraint generation (see Theorem 6 in [2] for a similar result and proof).

*This is essentially equivalent to using column generation to solve the dual of (ALLP).

†As long as there exists a violated constraint, an optimal solution to this problem comes from a constraint that is not in $\mathcal{C}$. Therefore, for simplicity of notation, the maximization is formulated over *all* functional constraints in (ALLP) instead of only those that are not in $\mathcal{C}$, without loss of optimality.

Proposition 3.1. *Suppose that (ALLP)$_{\mathcal{C}}$ has an optimal solution with optimal value $H_{\mathcal{C}}$ for some constraint set $\mathcal{C}$. Then $H_{\mathcal{C}} \leq H \leq H_{\mathcal{C}} + \frac{I}{1-\alpha}\mathcal{Z}^*$.*

Proof. Since (ALLP)$_{\mathcal{C}}$ is a relaxation of (ALLP), we have $H_{\mathcal{C}} \leq H$. Let $\gamma^{\mathcal{C}}, c^{\mathcal{C}}, \lambda^{\mathcal{C}}$ be an optimal solution to (ALLP)$_{\mathcal{C}}$. Then

$$H_{\mathcal{C}} = \frac{\sum_{j=1}^{J} \lambda_j^{\mathcal{C}} b_j}{1-\alpha} + \sum_{i=1}^{I} \gamma_i^{\mathcal{C}} + \sum_{i=1}^{I} \sum_{k=1}^{K_i} \Big(\sum_{x^i \in X^i} \beta_i(x^i) \nu_{ik}(x_k^i) \Big) c_{ik}^{\mathcal{C}}.$$

The dual of (ALLP) is given by

$$\begin{aligned}
(DALLP)\ \max \sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} & r_i(x^i,u^i) z_i(x^i,u^i) \\
\sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} & g_{ij}(x^i,u^i) z_i(x^i,u^i) \leq \frac{b^j}{1-\alpha},\ j \in \mathcal{J}, \\
(1-\alpha) \sum_{(x^i,u^i)\in\mathcal{Y}_i} & z_i(x^i,u^i) = 1,\ i \in \mathcal{I}, \\
\sum_{(x^i,u^i)\in\mathcal{Y}_i} \Big(\nu_{ik}(x_k^i) - \alpha E_{(x^i,u^i)} \nu_{ik}(\mathbf{X'}^i_k) \Big) & z_i(x^i,u^i) \leq \sum_{x^i \in X^i} \beta_i(x^i) \nu_{ik}(x_k^i),\ i \in \mathcal{I},\ k \in \mathcal{K}_i, \\
& z_i(x^i,u^i) \geq 0,\ (x^i,u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I}.
\end{aligned}$$

Since (ALLP) has an optimal solution, so does (DALLP) by strong duality. Let $z_i^*(x^i,u^i)$ be an optimal solution for (DALLP) and note that $H = \sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} r_i(x^i,u^i) z_i^*(x^i,u^i)$. Then using the constraints in (DALLP) in the above expression for $H_{\mathcal{C}}$ we get

$$\begin{aligned}
H_{\mathcal{C}} &\geq \sum_{j\in\mathcal{J}} \Big(\sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} g_{ij}(x^i,u^i) z_i^*(x^i,u^i) \Big) \lambda_j^{\mathcal{C}} + (1-\alpha) \sum_{i\in\mathcal{I}} \Big(\sum_{(x^i,u^i)\in\mathcal{Y}_i} z_i^*(x^i,u^i) \Big) \gamma_i^{\mathcal{C}} + \\
&\sum_{i=1}^{I} \sum_{k=1}^{K_i} \Big(\sum_{(x^i,u^i)\in\mathcal{Y}_i} \Big(\nu_{ik}(x_k^i) - \alpha E_{(x^i,u^i)} \nu_{ik}(\mathbf{X'}^i_k) \Big) z_i^*(x^i,u^i) \Big) c_{ik}^{\mathcal{C}} \\
&= \sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} \Big(r_i(x^i,u^i) - \mathcal{Z}_i(x^i,u^i) \Big) z_i^*(x^i,u^i), \quad \text{by definition of } \mathcal{Z}_i(x^i,u^i) \\
&= H - \sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} \mathcal{Z}_i(x^i,u^i) z_i^*(x^i,u^i), \quad \text{by definition of } H \\
&\geq H - \mathcal{Z}^* \sum_{i\in\mathcal{I}} \sum_{(x^i,u^i)\in\mathcal{Y}_i} z_i^*(x^i,u^i), \quad \text{from (8) because } z_i^*(x^i,u^i) \geq 0 \\
&= H - \frac{\mathcal{Z}^* I}{1-\alpha} \quad \text{from the equality constraints in (DALLP).}
\end{aligned}$$

□

This result leads to a natural stopping criterion for constraint generation — terminate when the absolute percentage gap between $H_{\mathcal{C}}$ and $H_{\mathcal{C}} + \frac{I}{1-\alpha}\mathcal{Z}^*$, that is, $100 \left| \frac{\frac{I}{1-\alpha}\mathcal{Z}^*}{H_{\mathcal{C}}} \right|$ drops below a tolerance. We apply this approach to allocation and advanced scheduling problems in the next two sections.

4 Application to allocation scheduling

Consider the following class of problems, which we call Dynamic Stochastic Allocation Scheduling Problems (DSALSPs).

1. (Heterogeneous job-types) The index set of job-types is denoted $\mathcal{I} = \{1, 2, \ldots, I\}$.

2. (Random arrivals) A random number $\mathbf{\Delta}_i$ of type-i jobs arrive in one time-period; $\mathbf{\Delta}_i$ takes values from the set $\{0, 1, \ldots, N_i\}$, where N_i are positive integers. The probability that n_i type-i jobs arrive in one period is $p_i(n_i)$. Arrivals across types are independent.

3. (Stochastic service-durations) The service-duration of each type-i job is a positive integer-valued random variable $\mathbf{\Gamma}_i$ independent of all else; $\mathbf{\Gamma}_i$ takes values in the set $\{1, 2, \ldots, T_i\}$ for some positive integer T_i. The probability that $\mathbf{\Gamma}_i$ equals τ_i from this set is $\theta_i(\tau_i)$. We use Θ_i to denote the cumulative distribution function of $\mathbf{\Gamma}_i$.

4. (Multiple resource constraints) $\mathcal{J} = \{1, \ldots, J\}$ denotes the set of resources and b_j (positive integer) is the total quantity of resource j available for consumption in each time-period. In order to perform each job of type i, a non-negative integer amount a_{ij} of resource j is required. We assume that for each i, there is at least one j such that $a_{ij} > 0$.

5. (Immediate decision) Each arriving job must either immediately start service or be rejected/outsourced/served through overtime. Henceforth we use the word "rejected" to refer to a generic decision of this latter type.

6. (Rewards and costs) The following rewards/costs are received/incurred:

   - (Completion reward) Reward R_i is received at the end of a time-period for completing a type-i job.
   - (Penalty cost of rejection) A penalty cost of G_i is incurred at the beginning of a period for each job of type i that is rejected.

7. (Discounted profit objective) The goal is to decide which of the arriving jobs to select for service in each period so as to maximize the total discounted expected profit over an infinite horizon with discount factor $0 < \alpha < 1$. Preemption is not allowed; that is, once a job is started, it cannot be discontinued until completion.

DSALPs are a variation of problems we introduced in [12], where, among other differences, it was assumed that service-durations were geometrically distributed and hence the state- and action-spaces were much simpler than those below because the geometric distribution is memoryless. Examples of work that is somewhat related to DSALSPs include Network Revenue Management Problems (NRMPs) [33], Dynamic Stochastic Knapsack Problems (DSKPs), dynamic fleet management problems [13, 14, 34], discrete-time queueing problems [3, 8, 20, 22, 24, 25, 36, 37], and the classic multiclass queuing models with Poisson arrivals [17]. Major differences between DSALSPs and this existing literature can be found in [10, 12]. We do not repeat those here for brevity.

4.1 A weakly coupled MDP model of DSALSPs

We now build a weakly coupled MDP model for DSALSPs. The state of this MDP is defined as $x = (x^1, x^2, \ldots, x^I)$, where $x^i = (x^i_0; x^i_1; \ldots; x^i_{T_i-1})$. In this vector, x^i_k, for $k = 0, 1, \ldots, T_i - 1$, is the number of type-$i$ jobs that have received service for k time-periods in the past. In particular,

x_0^i is the number of type-i jobs that arrived in the previous period and need to be either selected for service or rejected. Since the maximum number of type-i jobs that can arrive in one period is N_i, we have $x_0^i \leq N_i$. We therefore define the set $X_0^i = \{0, 1, \ldots, N_i\}$. Also let $M_i = \min_{j \in \mathcal{J}} \left\lfloor \frac{b_j}{a_{ij}} \right\rfloor$ be the largest number of type-i jobs that can be in service in any given period. We define sets $X_1^i = X_2^i = \ldots = X_{T_i-1}^i = \{0, 1, \ldots, M_i\}$ and let

$$X^i = \left\{ x \in X_0^i \times X_1^i \times \ldots \times X_{T_i-1}^i : \sum_{k=1}^{T_i-1} x_k^i \leq M_i \right\}.$$

Moreover, let $X = X^1 \times X^2 \times \ldots \times X^I$. Then the set of all feasible states is characterized by the resource availability as

$$\bar{X} = \left\{ x \in X : \sum_{i=1}^{I} a_{ij} \Big(\sum_{k=1}^{T_i-1} x_k^i \Big) \leq b_j, \ j \in \mathcal{J} \right\}. \tag{9}$$

The decision vector is $u = (u^1, u^2, \ldots, u^I)$, where u^i is the number of new type-i jobs we select for service from the x_0^i jobs that arrived in the previous period. Therefore $u^i \leq x_0^i$ and $x_0^i - u^i$ jobs are rejected. Let $U_1^i(x^i) = \{0, 1, \ldots, x_0^i\}$ and

$$U^i(x^i) = \Big\{ u^i \in U_1^i(x^i) : u^i + \sum_{k=1}^{T_i-1} x_k^i \leq M_i \Big\}.$$

Also let $U(x) = U^1(x^1) \times U^2(x^2) \times \ldots \times U^I(x^I)$. The set $\bar{U}(x) \subseteq U(x)$ of all decision vectors feasible in state x is defined by the resource constraints

$$\bar{U}(x) = \left\{ (u^1, u^2, \ldots, u^i) \in U(x) : \sum_{i=1}^{I} a_{ij} \Big(u^i + \sum_{k=1}^{T_i-1} x_k^i \Big) \leq b_j, \ j \in \mathcal{J} \right\}. \tag{10}$$

Let $q_i(t)$ denote the *conditional* probability that a type-i job that has received service for $0 \leq t \leq T_i - 1$ time-periods in the past will be completed at the end of the current time-period (in the case of $t = 0$, this is relevant only for jobs that we choose to serve in the current time-period). We have,

$$q_i(t) = \mathbf{P}[\mathbf{\Gamma}_i = t + 1 | \mathbf{\Gamma}_i > t] = \frac{\mathbf{P}[\mathbf{\Gamma}_i = t + 1 \cap \mathbf{\Gamma}_i > t]}{\mathbf{P}[\mathbf{\Gamma}_i > t]} = \frac{\theta_i(t+1)}{1 - \Theta_i(t)}. \tag{11}$$

Let η_0^i be the random number of type-i jobs that are completed out of the u^i new jobs started in the current period. Thus η_0^i is Binomial$(u^i, q^i(0))$. Similarly, for $k = 1, \ldots, T_i - 1$, let η_k^i be the number of type-i jobs completed in the current period out of the x_k^i ongoing jobs. For $k = 1, \ldots, T_i - 1$, η_k^i is Binomial$(x_k^i, q_i(k))$. In fact, $\eta_{T_i-1}^i = x_{T_i-1}^i$ with probability one because all jobs that have received service for $T_i - 1$ periods will be certainly completed in the next period. We define the vector $\eta^i = (\eta_0^i; \eta_1^i; \ldots; \eta_{T^i-1}^i)$. For $0 \leq n_i \leq N_i$, let

$$P_i(x^i, u^i; \eta^i, n_i) = \left[p_i(n_i) \times \binom{u^i}{\eta_0^i} \Big(q_i(0) \Big)^{\eta_0^i} \Big(1 - q_i(0) \Big)^{u^i - \eta_0^i} \times \prod_{k=1}^{T_i-2} \binom{x_k^i}{\eta_k^i} \Big(q_i(k) \Big)^{\eta_k^i} \Big(1 - q_i(k) \Big)^{x_k^i - \eta_k^i} \right].$$

This is the joint probability that, n_i new type-i jobs will arrive, and given state-vector x^i and decision-vector u^i for type-i, the vector of completed type-i jobs will equal η^i. Then with probability

$\prod_{i=1}^{I} P_i(x^i, u^i; \eta^i, n_i)$, the next state equals $x' = (x'^1, x'^2, \ldots, x'^I)$, where for $i \in \mathcal{I}$, $x'^i = (n_i; u^i - \eta_0^i; x_1^i - \eta_1^i; \ldots; x_{T_i-2}^i - \eta_{T_i-2}^i)$. In this vector, $u^i - \eta_0^i$ is the number of new type-i jobs that were started in this period but could not be completed. Therefore, they have now been processed for one time-period. Similarly, $x_k^i - \eta_k^i$, for $k = 1, 2, T_i - 2$, is the number of type-i jobs that have now been processed for $k+1$ time-periods but are not complete. The discounted expected profit accrued in this period equals $f(x, u) = \sum_{i=1}^{I} f_i(x^i, u^i)$, where $f_i(x^i, u^i) = \alpha R_i\Big(u^i q_i(0) + \sum_{k=1}^{T_i-1} x_k^i q_i(k)\Big) - G_i(x_0^i - u^i)$. Thus Bellman's equations for all $x \in X$ are now given by

$$V(x) = \max_{u \in \bar{U}(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) + \alpha E_{(x,u)} V(\mathbf{X}') \right\}, \tag{12}$$

where the shorthand $\mathbf{X}'$ denotes the I-dimensional random vector representing the next state and whose probability mass function is $\prod_{i=1}^{I} P_i(x^i, u^i; \eta^i, n_i)$. The subscript (x, u) on the expectation operator E emphasizes that this probability mass function is characterized by x and u.

This MDP is *almost* weakly coupled — it satisfies all requirements of a weakly coupled MDP *except* that the set of feasible states $\bar{X}$ in (9) does not have a Cartesian product structure as state feasibility is defined through linking resource constraints.

To endow the state-space with a Cartesian structure, we extend it to X by including (artificial) states in $X \setminus \bar{X}$. In each $x^i \in X^i$, we allow the (artificial) action $\mu^i(x^i) = -\sum_{k=1}^{T_i-1} x_k^i$. This particular choice for an artificial action may seem *ad hoc* at first, but it allows us to conveniently extend the set $\bar{U}(x)$ for $x \in \bar{X}$ to the set $\bar{A}(x)$ for $x \in X$ in (13) below. We first define the set $A^i(x^i) = U^i(x^i) \cup \{\mu^i(x^i)\}$, and for states $x \in X$, we let $A(x) = A^1(x^1) \times A^2(x^2) \times \ldots \times A^I(x^I)$. We then define the set of feasible decisions in state $x \in X$ as

$$\bar{A}(x) = \left\{ (u^1, u^2, \ldots, u^I) \in A(x) : \sum_{i=1}^{I} a_{ij}\Big(u^i + \sum_{k=1}^{T_i-1} x_k^i\Big) \le b_j,\ j \in \mathcal{J} \right\}. \tag{13}$$

Now note that $\mu^i(x^i) + \sum_{k=1}^{T^i-1} x_k^i = 0$. This implies that (13) does not put any unnecessary restrictions on components of u that are not artificial. After choosing a decision $u \in A(x)$ whose ith component is the artificial decision $\mu^i(x^i)$ in a state with ith component $x^i \in X^i$, we (artificially) set the ith component of the next state to $x'^i = (n_i; 0; 0; \ldots; 0)$ with probability $p_i(n_i)$. This particular choice of state is based purely on its simplicity. Finally, we set $f_i(x^i, \mu^i(x^i))$, for all $x^i \in X^i$ and $i \in \mathcal{I}$, to $-L$, where L is a sufficiently large positive number.

Let $\Phi(x)$ be the maximum infinite-horizon discounted expected reward obtained if the current state in the transformed MDP is $x \in X$. Then Bellman's equations for the transformed MDP are

$$\Phi(x) = \max_{u \in \bar{A}(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) + \alpha E_{(x,u)} \Phi(\mathbf{X}') \right\},\ x \in X, \tag{14}$$

where the ith component of the random vector $\mathbf{X}'$ depends on whether u^i is artificial or not. The expectation is with respect to the probability mass function of this random vector, and it is characterized by (x, u). It is easy to show that the addition of artificial states and actions does not alter the optimal values of, and optimal decisions in, the original states (we provided a proof for a different MDP model in [10]). In particular, $V(x) = \Phi(x)$ for every $x \in \bar{X}$. Since this new MDP is weakly coupled, we decompose it using Lagrangian relaxation below.

4.2 Lagrangian relaxation of the allocation scheduling MDP

By applying the Lagrangian relaxation method to the weakly coupled MDP above, we obtain the relaxed Bellman's equations

$$\Phi^{\lambda}(x) = \max_{u \in A(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) + \sum_{j=1}^{J} \lambda_j \Big(b_j - \sum_{i=1}^{I} a_{ij}(u^i + \sum_{k=1}^{T_i - 1} x_k^i) \Big) + \alpha E_{(x,u)} \Phi(\mathbf{X}') \right\} \quad x \in X.$$

The Lagrangian value function is given by $\Phi^{\lambda}(x) = \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \sum_{i=1}^{I} \Phi_i^{\lambda}(x^i)$, where, for $x^i \in X^i$,

$$\Phi_i^{\lambda}(x^i) = \max_{u^i \in A^i(x^i)} \left\{ f_i(x^i, u^i) - \sum_{j=1}^{J} \lambda_j a_{ij}(u^i + \sum_{k=1}^{T_i - 1} x_k^i) + \alpha E_{(x^i, u^i)} \Phi_i^{\lambda}(\mathbf{X'}^i) \right\}.$$

In the above equation, E denotes the expectation with respect to the probability mass function of the ith component $\mathbf{X'}^i$ of the random vector $\mathbf{X'}$. The subscript (x^i, u^i) emphasizes that this probability mass function is characterized by the state x^i and decision u^i. Using the notation $\mathcal{Y}_i = \big\{(x^i, u^i) : x^i \in X^i, u^i \in A^i(x^i)\big\}$ for $i \in \mathcal{I}$, the Lagrangian linear program is now given by

$$(LLP1)\ H^{\lambda^*}(\beta) = \min \ \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \sum_{i=1}^{I} \sum_{x^i \in X^i} \beta_i(x^i) \nu_i(x^i)$$

$$\nu_i(x^i) - \alpha E_{(x^i,u^i)} \nu_i(\mathbf{X'}^i) + \sum_{j=1}^{J} \lambda_j a_{ij}(u^i + \sum_{k=1}^{T_i - 1} x_k^i) \geq f_i(x^i, u^i),\ (x^i, u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I},$$

$$\lambda \geq 0.$$

Notice that even though the numbers of variables and constraints in (LLP1) are both linear in I, they are exponential in T^i for each i. Thus (LLP1) is intractable when T^i's are large. We therefore apply the value function approximation and constraint generation method developed in Section 3.

But first, we show that dropping the functional constraints that correspond to artificial actions from (LLP1) does not alter its optimal solution. For each $i \in \mathcal{I}$, let $\bar{\mathcal{Y}}_i \subset \mathcal{Y}_i$ be the subset given by $\{(x^i, u^i) \in \mathcal{Y}_i : u^i \neq \mu^i(x^i)\}$. That is, $\bar{\mathcal{Y}}_i$ does not include artificial actions and hence is equivalently defined as $\bar{\mathcal{Y}}_i = \{(x^i, u^i) : x^i \in X^i,\ u^i \in U^i(x^i)\}$. Now consider a relaxation of (LLP1), which only includes functional constraints over sets $\bar{\mathcal{Y}}_i$ for $i \in \mathcal{I}$. It is given by

$$(LLP1R)\ \min \ \frac{\sum_{j=1}^{J} \lambda_j b_j}{1-\alpha} + \sum_{i=1}^{I} \sum_{x^i \in X^i} \beta_i(x^i) \nu_i(x^i)$$

$$\nu_i(x^i) - \alpha E_{(x^i,u^i)} \nu_i(\mathbf{X'}^i) + \sum_{j=1}^{J} \lambda_j a_{ij}(u^i + \sum_{k=1}^{T_i - 1} x_k^i) \geq f_i(x^i, u^i),\ (x^i, u^i) \in \bar{\mathcal{Y}}_i,\ i \in \mathcal{I},$$

$$\lambda \geq 0.$$

Let $\nu_i^*(x^i)$ for $x^i \in X^i$ and $i \in \mathcal{I}$ be an optimal solution of (LLP1R). Then this solution is also optimal to (LLP1). To establish this, we only need to show that this solution satisfies the functional constraints in (LLP1) that correspond to $\bigcup_{i \in \mathcal{I}} \big(\mathcal{Y}_i \backslash \bar{\mathcal{Y}}_i\big)$. The left hand side of such a functional constraint

corresponding to $x^i \in X^i$ and artificial action $\mu^i(x^i)$ simply equals $\nu_i^*(x^i) - \alpha E_{(x^i,\mu^i(x^i))}\nu_i^*(\mathbf{X}'^i)$ because $\mu^i(x^i)+\sum_{k=1}^{T_i-1} x_k^i = 0$ by definition of $\mu^i(x^i)$. Let $\nu^* = \min_{i\in\mathcal{I}}\{\min_{x^i\in X^i}\big(\nu_i^*(x^i)-\alpha E_{(x^i,\mu^i(x^i))}\nu_i^*(\mathbf{X}'^i)\big)\}$ be the smallest among all such left hand side values. Then our claim holds because $f_i(x^i,\mu^i(x^i))$ is sufficiently small, and in particular we assume it to be smaller than ν^*, for all $i\in\mathcal{I}$ and $x^i\in X^i$. Thus it suffices to solve (LLP1R).

4.3 Affine approximation and constraint generation

We employ affine functions in our componentwise value function approximations. That is, for each $i\in\mathcal{I}$ and $x^i\in X^i$, we set $\nu_i(x^i)\approx\gamma_i+\sum_{k=0}^{T_i-1}c_{ik}x_k^i$, where γ_i and $c_{ik}\geq 0$ are unknown coefficients. Here, c_{ik} can be interpreted as the marginal value of having an incomplete type-i job that has received service for k time-periods in the past, in the system. This simplifies (LLP1R) to

$$(ALLP1)\ H = \min\ \frac{\sum_{j=1}^{J}\lambda_j b_j}{1-\alpha}+\sum_{i=1}^{I}\gamma_i+\sum_{i=1}^{I}\sum_{k=0}^{T_i-1}\Big(\sum_{x^i\in X^i}\beta_i(x^i)x_k^i\Big)c_{ik}$$

$$(1-\alpha)\gamma_i+\sum_{k=0}^{T_i-1}\Big(x_k^i-\alpha E_{(x^i,u^i)}\mathbf{X}'^i_k\Big)c_{ik}+\sum_{j=1}^{J}\lambda_j a_{ij}(u^i+\sum_{k=1}^{T_i-1}x_k^i)\geq f_i(x^i,u^i),\ (x^i,u^i)\in\bar{\mathcal{Y}}_i,\ i\in\mathcal{I},$$

$$\lambda\geq 0,$$

$$c_{ik}\geq 0,\ i\in\mathcal{I},\ k=0,1,\ldots,T^i-1.$$

In our numerical results below, the constraint generation procedure for (ALLP1) starts with an initial set of constraints $\mathcal{C}_0$ that includes constraints corresponding to $x_k^i=0$ for $k=0,1,\ldots,T_i-1$ and $u^i=0$ for all $i\in\mathcal{I}$. It is easy to see that $\gamma_i=0$ for all $i\in\mathcal{I}$, $c_{ik}=0$ for $k=0,1,\ldots,T_i-1$ and all $i\in\mathcal{I}$, and $\lambda_j=0$ for all $j\in\mathcal{J}$ is an optimal solution to this initial problem. Now suppose that, during the constraint generation procedure, after solving $(\text{ALLP1})_{\mathcal{C}}$ with some set of constraints $\mathcal{C}$, we obtain $\gamma^{\mathcal{C}}, c^{\mathcal{C}}, \lambda^{\mathcal{C}}$ as its optimal solution. The violation in the functional constraint corresponding to any $(x^i,u^i)\in\bar{\mathcal{Y}}_i$ for any $i\in\mathcal{I}$ is given by

$$\mathcal{Z}_i(x^i,u^i)\triangleq f_i(x^i,u^i)-(1-\alpha)\gamma_i^{\mathcal{C}}-\sum_{k=0}^{T_i-1}\Big(x_k^i-\alpha E_{(x^i,u^i)}\mathbf{X}'^i_k\Big)c_{ik}^{\mathcal{C}}-\sum_{j=1}^{J}\lambda_j^{\mathcal{C}}a_{ij}(u^i+\sum_{k=1}^{T_i-1}x_k^i).$$

To simplify this, recall that $\mathbf{X}'^i_0=n_i$ with probability $p_i(n_i)$ for $0\leq n_i\leq N_i$. Moreover, $\mathbf{X}'^i_1 = u^i-\eta_0^i$, where η_0^i is Binomial$(u^i,q_i(0))$. Similarly, for $k=1,2,\ldots,T_i-2$, $\mathbf{X}'^i_{k+1}=x_k^i-\eta_k^i$, where η_k^i is Binomial$(x_k^i,q_i(k))$. Therefore, $\sum_{k=0}^{T^i-1}c_{ik}^{\mathcal{C}}E_{(x^i,u^i)}\mathbf{X}'^i_k = c_{i0}^{\mathcal{C}}E\boldsymbol{\Delta}_i+c_{i1}^{\mathcal{C}}u^i(1-q_i(0))+\sum_{k=1}^{T_i-2}c_{ik+1}^{\mathcal{C}}x_k^i(1-q_i(k))$. Also recall that $f_i(x^i,u^i)=\alpha R_i\Big(u^iq_i(0)+\sum_{k=1}^{T_i-1}x_k^iq_i(k)\Big)-G_i(x_0^i-u^i)$. Using this, after

algebraic simplification, $\mathcal{Z}_i(x^i, u^i)$ can be expressed as $\mathcal{Z}_i(x^i, u^i) = \sum_{k=0}^{T_i-1} \delta_{ik}^{\mathcal{C}} x_k^i + \epsilon_i^{\mathcal{C}} u^i + \varepsilon_i^{\mathcal{C}}$, where

$$\begin{aligned}
\delta_{i0}^{\mathcal{C}} &\triangleq -G_i - c_{i0}^{\mathcal{C}}, \\
\delta_{ik}^{\mathcal{C}} &\triangleq \alpha q_i(k) R_i - c_{ik}^{\mathcal{C}} + \alpha c_{ik+1}^{\mathcal{C}}(1 - q_i(k)) - \sum_{j=1}^{J} \lambda_j a_{ij}, \text{ for } k = 1, 2, \ldots, T_i - 2, \\
\delta_{iT_i-1}^{\mathcal{C}} &\triangleq \alpha q_i(k) R_i - c_{iT_i-1}^{\mathcal{C}} - \sum_{j=1}^{J} \lambda_j a_{ij}, \\
\epsilon_i^{\mathcal{C}} &\triangleq \alpha R_i q_i(0) + G_i + \alpha c_{i1}^{\mathcal{C}}(1 - q_i(0)) - \sum_{j=1}^{J} \lambda_j a_{ij}, \text{ and} \\
\varepsilon_i^{\mathcal{C}} &\triangleq -(1-\alpha)\gamma_i^{\mathcal{C}} + \alpha c_{i0}^{\mathcal{C}} E \boldsymbol{\Delta}_i.
\end{aligned}$$

Thus the maximum constraint violation problem is given by $\mathcal{Z}^* \triangleq \max_{i \in \mathcal{I}} \mathcal{Z}_i^*$, where $\mathcal{Z}_i^*$ is the optimal value of the linear integer program

$$\begin{aligned}
\mathcal{Z}_i^* \triangleq \max \ & \sum_{k=0}^{T_i-1} \delta_{ik}^{\mathcal{C}} x_k^i + \epsilon_i^{\mathcal{C}} u^i + \varepsilon_i^{\mathcal{C}} \qquad (15) \\
& x_0^i \leq N_i, \\
& u^i \leq x_0^i, \\
u^i + & \sum_{k=1}^{T_i-1} x_k^i \leq M_i, \\
& u^i \geq 0, \text{ and integer}, \\
& x_k^i \geq 0, \text{ and integer, for } k = 0, 1, \ldots, T_i - 1.
\end{aligned}$$

4.4 Retrieving decisions from the approximate value function

We denote the values of variables obtained after terminating the constraint generation procedure as λ_j^* for $j \in \mathcal{J}$, γ_i^* for $i \in \mathcal{I}$, and c_{ik}^* for $k = 0, 1, \ldots, T_i - 1$ and $i \in \mathcal{I}$. Now suppose for some state $x \in \bar{X}$, we wish to compute a decision vector that is myopic with respect to the approximation

$$V(x') = \Phi(x') \approx \Phi^{\lambda^*}(x') = \frac{\sum_{j=1}^{J} \lambda_j^* b_j}{1-\alpha} + \sum_{i=1}^{I} \Phi_i^{\lambda^*}(x'^i) \approx \frac{\sum_{j=1}^{J} \lambda_j^* b_j}{1-\alpha} + \sum_{i=1}^{I} \left(\gamma_i^* + \sum_{k=0}^{T_i-1} c_{ik}^* x'^i \right)$$

in the right hand side of Bellman's equation (12). The decision retrieval problem then simplifies to

$$\frac{\alpha \sum_{j=1}^{J} \lambda_j^* b_j}{1-\alpha} + \alpha \sum_{i=1}^{I} \gamma_i^* + \max_{u \in \bar{U}(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) + \alpha \sum_{i=1}^{I} \sum_{k=0}^{T_i-1} c_{ik}^* E_{(x^i, u^i)} \mathbf{X}_k'^i \right\}.$$

We ignore the terms outside the maximization in the discussion below. Then substituting appropriate expressions for $f_i(x^i, u^i)$ and $E_{(x^i,u^i)}$, we get

$$\max_{u\in\bar{U}(x)} \left\{ \sum_{i=1}^{I} \Big(\alpha R_i \Big(u^i q_i(0) + \sum_{k=1}^{T_i-1} x_k^i q_i(k) \Big) - G_i(x_0^i - u^i) \Big) + \alpha \sum_{i=1}^{I} \Big[c_{i0}^* E\boldsymbol{\Delta}_i + c_{i1}^* u^i (1 - q_i(0)) + \sum_{k=1}^{T_i-2} c_{ik+1}^* x_k^i (1 - q_i(k)) \Big] \right\}.$$

Finally, ignoring the terms in the objective function that do not depend on u and using the definition of $\bar{U}(x)$ in (10), we further rewrite this problem as

$$\begin{aligned} \max \ & \sum_{i=1}^{I} \Big[\alpha R_i q_i(0) + G_i + \alpha c_{i1}^* (1 - q_i(0)) \Big] u^i \\ & u^i \le x_0^i, \ i \in \mathcal{I}, \\ \sum_{i=1}^{I} a_{ij} u^i \le b_j - & \sum_{i=1}^{I} a_{ij} \sum_{k=1}^{T_i-1} x_k^i, \ j \in \mathcal{J}, \\ & u^i \ge 0, \ i \in \mathcal{I}, \\ & u^i \text{ integer}, \ i \in \mathcal{I}. \end{aligned}$$

This is a linear integer program that needs to be solved in order to recover decisions on the fly.

4.5 Numerical experiments

All numerical experiments in this paper were performed on a Dell Optiplex 960 desktop computer running Windows XP on an Intel Core Duo CPU with 3.16GHz of processing speed and 3.21GB of RAM. Computer programs were written in MATLAB, which called IBM CPLEX for solving linear and integer programs. We used a test bed of randomly generated problems to compare the performance of our Lagrangian policy with a myopic policy. We set the number of job types to $I = 5$. The maximum number of type-i arrivals in one period, denoted N_i, were set to DU$[1, 5]$. Once N_i was sampled, the probability mass function $p_i(\cdot)$ was obtained by normalizing $N_i + 1$ uniform $(0, 1)$ random variables. The number of resources were set to $J = 1, 2$. The resource availabilities and consumption values were constructed using a slight modification of a standard method for generating multidimensional knapsack constraints [6]. In particular, the unit resource consumption values a_{ij} were set to $DU[1, 100]$. The resource availability b_j was then set to $I \times \rho \times \sum_{i=1}^{I} a_{ij}$, where ρ is called a tightness ratio. Tightness ratio values were set to $\rho = 0.25, 0.5$. The maximum service durations T_i were set to 1+DU$[1, 10]$; the 1 was added to prevent the scenario where $T_i = 1$ because the deterministic unit-duration case was studied in our earlier work in [12]. Once a T_i was sampled, the service-duration probabilities $\theta_i(\tau_i)$, for $\tau_i \in \{1, 2, \ldots, T_i\}$, were obtained by normalizing T_i uniform $(0, 1)$ random variables. The job-completion rewards R_i were set to DU$[1, 1000]$ and the rejection costs G_i were set to DU$[1, 100]$. Following Adelman and Mersereau [2], the discount factor was set to 0.85. Thus, our problems sets are characterized by parameters J, ρ. For each combination of parameter values, we solved 50 randomly generated problem instances.

Recall from Section 4.3 that our initial set of constraints includes constraints corresponding to $x_k^i = 0$ for $k = 0, 1, \ldots, T_i - 1$ and $u^i = 0$ for all $i \in \mathcal{I}$. We terminated the constraint generation

Problem set	J	ρ	CPU seconds	Lagrangian	Myopic	% improvement
1	1	0.25	3	4271	4187	2.00
2	1	0.5	4	8341	8201	1.70
3	2	0.25	3	3474	3453	0.61
4	2	0.5	4	7324	7230	1.30
Average	-	-	3.5	5852	5768	1.46

Table 1: Results for 4 allocation scheduling problem sets. For each problem set, the profit estimates reported are averages over 50 randomly generated problem instances. Thus the table summarizes results for 200 allocation scheduling problems.

procedure when the absolute percentage gap $\frac{I}{1-\alpha}\left|\frac{\mathcal{Z}^*}{H_{\mathcal{C}}}\right|$ dropped below 0.5%. For each problem instance, the infinite-horizon discounted expected profit accrued by the Lagrangian or the myopic policy was estimated by averaging the total discounted profit accumulated over 200 independent simulations of 50 periods. Each of the 200 simulations was started in the "empty" state $x_k^i = 0$ for $k = 0, 1, \ldots, T_i$. As suggested in [2], running a simulation for 50 periods sufficiently well-approximates the infinite-horizon profits because $0.85^{50} \approx 10^{-4}$ and hence the tail profits beyond the 50th period are relatively insignificant. These profit estimates, averaged over the 50 independent problem instances for each problem set are reported in Table 1. The table also lists the percentage improvement in profit obtained by the Lagrangian policy over the myopic policy for each problem set. The average CPU time needed to complete constraint generation is also reported in seconds for each problem set. The table shows that the Lagrangian policy generates higher profit than the myopic policy in all four problem sets. The last row of the table shows averages over all 200 problem instances. Note that the average improvement achieved by the Lagrangian policy over the myopic policy in these 200 instances equals $100 \times (5852 - 5768)/5768$, which is 1.46%. A paired t-test for the difference between Lagrangian and myopic profit estimates over the 200 instances was statistically significant at the 0.05 level with p-value of the order 10^{-5} and returned $[46.97, 122.08]$ as the 95% confidence interval for the true mean of the difference.

5 Application to advanced scheduling

We now consider the following class of problems that we term Dynamic Stochastic Advanced Scheduling Problems (DSADSPs).

1. (Heterogeneous job types) The index set of job types is denoted $\mathcal{I} = \{1, 2, \ldots, I\}$.
2. (Random arrivals) A random number $\mathbf{\Delta}_i$ of type-i jobs arrive in one time-period; $\mathbf{\Delta}_i$ takes values from the set $\{0, 1, \ldots, N_i\}$, where N_i are positive integers. The probability that n_i type-i jobs arrive in one period is $p_i(n_i)$. Arrivals across types are independent.
3. (Multiple resource constraints) $\mathcal{J} = \{1, \ldots, J\}$ denotes the set of resources and b_j (positive integer) is the total quantity of resource $j \in \mathcal{J}$ available for consumption in each time-period. In order to perform each job of type i, a non-negative integer amount a_{ij} of resource j is required. We assume that for each job of type i, there is at least one resource j such that $a_{ij} > 0$.
4. (Immediate decision) Each arriving job must be scheduled to any one of the next T periods — the booking horizon. The planner can also reject/outsource/serve jobs using overtime. Henceforth we use the word "reject" to refer to a generic decision of this latter type.

5. (Costs) The following costs are incurred.
   - (Delay cost) Type-i jobs are associated with a deadline $0 \leq D_i \leq T$. The cost of scheduling a type-i job on day $1 \leq t \leq T$ in the booking horizon is given by the non-negative function $C_i(t; D_i)$. We assume that $C_i(t; D_i)$ is non-decreasing in t.
   - (Penalty cost of rejection) The planner incurs a penalty cost of G_i per type-i job that is rejected.
6. (Total discounted cost objective) The goal is to schedule and/or reject arriving jobs so as to minimize the total discounted expected cost over an infinite horizon with discount factor $0 < \alpha < 1$.

This problem is similar to [27], who studied patient scheduling problems in a diagnostic setting. The key differences are that they only had one capacity constraint and all patients "consumed" equal, and in fact, unit, capacity. In this sense, our problem statement is a generalization of [27]. Consequently, the state and action spaces in their MDP model are simpler in structure and smaller in size than ours below. We note that similar to [27], the notion of service-duration is not explicit in our problem statement. It can, however, be incorporated implicitly through resource constraints. For example, in advanced scheduling of elective surgeries, hospitals reserve blocks of fixed durations, say one hour or two hours, depending on the type of surgery in an operating room. So in an eight-hour day shift, eight one-hour surgeries or four two hour surgeries or four one-hour surgeries and two two-hour surgeries, etc. can be performed. This kind of service-durations can be incorporated into our advanced scheduling framework through a resource constraint where time itself is the resource (see [10] for an application of advanced scheduling to elective surgeries).

5.1 A weakly coupled MDP model of DSADSPs

The state of our MDP model is defined as $x = (x^1, x^2, \ldots, x^I)$, where each x^i is a $T+1$ dimensional vector written as $x^i = (x_0^i; x_1^i; \ldots; x_T^i)$. In this vector, x_0^i is the number of type-i jobs that arrived in the previous period, and x_k^i, for $k = 1, \ldots, T$, is the number of type-i jobs that are currently scheduled on the kth day in the booking horizon. As in the allocation scheduling model, since the maximum number of type-i jobs that can arrive in one period is N_i, we have $x_0^i \leq N_i$. We therefore define the set $X_0^i = \{0, 1, \ldots, N_i\}$. Also let $M_i = \min_{j \in \mathcal{J}} \left\lfloor \frac{b_j}{a_{ij}} \right\rfloor$ be the largest number of type-i jobs that could be scheduled in any given period. We define sets $X_1^i = X_2^i = \ldots = X_T^i = \{0, 1, \ldots, M_i\}$ and let $X^i = X_0^i \times X_1^i \times \ldots \times X_T^i$. Finally, let $X = X^1 \times X^2 \times \ldots \times X^I$. The set $\bar{X}$ of all feasible states is then given by

$$\bar{X} = \left\{ x \in X : \sum_{i=1}^{I} a_{ij} x_t^i \leq b_j,\ j \in \mathcal{J},\ t = 1, 2, \ldots, T \right\}. \tag{16}$$

That is, state x is feasible if and only if the number of jobs of different types that are currently scheduled for period t in the booking horizon must respect the resource constraint for each resource j. The decision vector is $u = (u^1, u^2, \ldots, u^I)$, where each u^i is a T-dimensional vector written as $u^i = (u_1^i; \ldots; u_T^i)$. Here, u_t^i, for $t = 1, \ldots, T$, is the number of type-i jobs (among the ones that arrived in the previous period) that we choose to schedule on the tth day in the booking horizon. For $i \in \mathcal{I}$, $x^i \in X^i$ and $t = 1, 2, \ldots, T$, let $U_t^i(x^i) = \{0, 1, \ldots, x_0^i\}$ and

$$U^i(x^i) = \{u^i \in U_1^i(x^i) \times \ldots \times U_T^i(x^i) : \sum_{t=1}^{T} u_t^i \leq x_0^i, \text{ and } x_t^i + u_t^i \leq M_i, \text{ for } t = 1, 2, \ldots, T\}.$$

Thus $x_0^i - \sum_{t=1}^{T} u_t^i$ is the number of type-i jobs that are rejected. Also, for $x \in X$, let $U(x) = U^1(x^1) \times U^2(x^2) \times \ldots \times U^I(x^I)$. The set $\bar{U}(x) \subseteq U(x)$ of all decision vectors feasible in state $x \in \bar{X}$ is defined as

$$\bar{U}(x) = \left\{ (u^1, u^2, \ldots, u^I) \in U(x) : \sum_{i=1}^{I} a_{ij}(u_t^i + x_t^i) \leq b_j,\ j \in \mathcal{J},\ t = 1, 2, \ldots, T \right\}.$$

On selecting decision $u \in \bar{U}(x)$ in state x, the ith component of the next state equals $x'^i = (n_i; x_2^i + u_2^i; \ldots; x_T^i + u_T^i; 0)$ with probability $p_i(n_i)$, for $0 \leq n_i \leq N_i$. Note here that x_t^i, for $t = 2, 3, \ldots, T$, is the number of type-i jobs previously scheduled on the $t-1$st day in the (new) booking horizon. Adding u_t^i, for $t = 2, 3, \ldots, T$, to this yields the total number of type-i jobs scheduled on these days. The last component in x'^i is zero because this is the $T+1$st day in the (old) booking horizon and hence no jobs can be scheduled there. The next state then is $x' = (x'^1, x'^2, \ldots, x'^I)$ with probability $\prod_{i=1}^{I} p_i(n_i)$. The immediate cost incurred in this period is given by $f(x, u) = \sum_{i=1}^{I} f_i(x^i, u^i)$, where

$$f_i(x^i, u^i) = G_i\Big(x_0^i - \sum_{t=1}^{T} u_t^i\Big) + \sum_{t=1}^{T} u_t^i C_i(t; D_i),\ \forall x^i \in X^i,\ u^i \in U^i(x^i). \tag{17}$$

Let $V(x)$ be the minimum infinite-horizon discounted expected cost incurred starting in state $x \in \bar{X}$. Then, Bellman's equations for our MDP model are given by

$$V(x) = \min_{u \in \bar{U}(x)} \left\{ f(x, u) + \alpha \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \ldots \sum_{n_I=0}^{N_I} \left(\prod_{i=1}^{I} p_i(n_i) \right) V(x'^1, \ldots, x'^I) \right\},\ x \in \bar{X}. \tag{18}$$

Similar to the allocation scheduling MDP in Section 4.1, the above MDP is *almost* weakly coupled — it satisfies all requirements of a weakly coupled MDP *except* that the set of feasible states $\bar{X}$ in (16) does not have a Cartesian product structure as state feasibility is defined through linking resource constraints. We first convert it into an equivalent weakly coupled MDP as follows.

We extend the feasible state-space $\bar{X}$ of our MDP to X by including (artificial) states $X \setminus \bar{X}$. In each $x^i \in X^i$, we allow the (artificial) action $\mu^i(x^i)$ whose components $\mu^i(x^i)_t$, for $t = 1, \ldots, T$, are given by $(\mu^i(x^i)_1; \mu^i(x^i)_2; \ldots; \mu^i(x^i)_T) = (-x_1^i; -x_2^i; \ldots; -x_T^i)$. This particular choice of artificial actions allows us to conveniently extend the set $\bar{U}(x)$ for $x \in \bar{X}$ to the set $\bar{A}(x)$ for $x \in X$ in (19) below. We first define the set $A^i(x^i) = U^i(x^i) \bigcup \mu^i(x^i)$, and for states $x \in X$, we let $A(x) = A^1(x^1) \times A^2(x^2) \times \ldots \times A^I(x^I)$. We then define the set of decisions that are feasible in state $x \in X$ as

$$\bar{A}(x) = \left\{ (u^1, u^2, \ldots, u^I) \in A(x) : \sum_{i=1}^{I} a_{ij}(u_t^i + x_t^i) \leq b_j,\ j \in \mathcal{J},\ t = 1, 2, \ldots, T \right\}. \tag{19}$$

Now note that owing to the way artificial actions are defined, $\mu_t^i(x^i) + x_t^i = 0$ for $t = 1, 2, \ldots, T$. This implies that (19) does not put any unnecessary restrictions on components of u that are not artificial. After choosing a decision $u \in A(x)$ whose ith component is the artificial decision $\mu^i(x^i)$ in a state with ith component $x^i \in X^i$, we (artificially) set the ith component of the next state

to $x'^i = (n_i; 0; \ldots; 0)$ with probability $\prod_{i=1}^{I} p_i(n_i)$. This particular choice of state was based on its simplicity. Finally, we set the ith component of the immediate cost incurred on choosing the artificial action component $\mu^i(x^i)$ in state component x^i to a sufficiently large positive number B.

Let $\Phi(x)$ be the minimum infinite-horizon discounted expected cost incurred when starting the transformed MDP in state $x \in X$. Bellman's equations for the transformed MDP are given by

$$\Phi(x) = \min_{u \in \bar{A}(x)} \left\{ f(x,u) + \alpha \sum_{n_1=0}^{N_1} \sum_{n_2=0}^{N_2} \cdots \sum_{n_I=0}^{N_I} \left(\prod_{i=1}^{I} p_i(n_i) \right) \Phi(x'^1, \ldots, x'^I) \right\}, \ x \in X. \tag{20}$$

It is easy to show that the addition of artificial states and actions does not alter the optimal values of, and optimal decisions in, the original states (see [10] for a proof). In particular, $V(x) = \Phi(x)$ for every $x \in \bar{X}$. Since this new MDP is weakly coupled, we decompose it using Lagrangian relaxation below.

5.2 Lagrangian relaxation of the advanced scheduling MDP

Let $\lambda_t = (\lambda_{t1}, \lambda_{t2}, \ldots, \lambda_{tJ}) \geq 0$ be the Lagrange multipliers associated with the J resources on days $1 \leq t \leq T$ in the booking horizon. Also let λ represent the vector $(\lambda_1; \lambda_2; \ldots; \lambda_T)$. The Lagrangian relaxation of Bellman's equations (20) is given by

$$\Phi^\lambda(x) = \min_{u \in A(x)} \left\{ \sum_{i=1}^{I} f_i(x^i, u^i) - \sum_{t=1}^{T} \sum_{j=1}^{J} \lambda_{tj} \Big(b_j - \sum_{i=1}^{I} a_{ij}(u_t^i + x_t^i) \Big) + \alpha E_{(x,u)} \Phi^\lambda(\mathbf{X}') \right\}, \ \forall x \in X. \tag{21}$$

The Lagrangian value function $\Phi^\lambda(x)$ can be expressed as

$$\Phi^\lambda(x) = -\frac{1}{1-\alpha} \sum_{t=1}^{T} \sum_{j=1}^{J} \lambda_{tj} b_j + \sum_{i=1}^{I} \Phi_i^\lambda(x^i), \ \forall x \in X, \tag{22}$$

where

$$\Phi_i^\lambda(x^i) = \min_{u^i \in A^i(x^i)} \left\{ f_i(x^i, u^i) + \sum_{t=1}^{T} \sum_{j=1}^{J} \lambda_{tj} a_{ij}(u_t^i + x_t^i) + \alpha E_{(x^i,u^i)} \Phi_i^\lambda(\mathbf{X}'^i) \right\}, \ \forall x^i \in X^i. \tag{23}$$

Let $\mathcal{Y}_i = \{(x^i, u^i) : x^i \in X^i, u^i \in A^i(x^i)\}$ for $i \in \mathcal{I}$. The Lagrangian linear program is then given by

$$\begin{aligned} (LLP2) \ H^{\lambda^*}(\beta) = \max \ & \frac{-\sum_{t=1}^{T} \sum_{j=1}^{J} \lambda_{tj} b_j}{1-\alpha} + \sum_{i=1}^{I} \sum_{x^i \in X^i} \beta_i(x^i) \nu_i(x^i) \\ & \nu_i(x^i) - \alpha E_{(x^i,u^i)} \nu_i(\mathbf{X}'^i) - \sum_{t=1}^{T} \sum_{j=1}^{J} \lambda_{tj} a_{ij}(u_t^i + x_t^i) \leq f_i(x^i, u^i), \ (x^i, u^i) \in \mathcal{Y}_i, \ i \in \mathcal{I}, \\ & \lambda \geq 0. \end{aligned}$$

This linear program has $TJ + \sum_{i=1}^{I} (N_i + 1)(M_i + 1)^T$ variables and at most $TJ + \sum_{i=1}^{I} \Big(1 + \Big((N_i + 1)(M_i + 1)^T \binom{N_i+T}{T} \Big) \Big)$ constraints. Thus the numbers of variables and constraints are linear in I

but unfortunately still exponential in T. Thus (LLP2) is intractable for realistic values of booking horizons T. In Section 5.3, we therefore apply the value function approximation and constraint generation method developed in Section 3.

5.3 Affine approximation and constraint generation

We simplify (LLP2) using the affine approximation $\nu_i(x^i) \approx \gamma_i + \sum_{t=0}^{T} c_{it} x_t^i$, where γ_i and c_{it} are unknown coefficients. The coefficients c_{it} can be interpreted as the marginal cost of having a type-i job scheduled in period t in the booking horizon. This simplifies (LLP2) to the approximate Lagrangian linear program

$$
\begin{aligned}
(ALLP2)\ H = \max\ & \frac{-\sum_{t=1}^{T}\sum_{j=1}^{J} \lambda_{tj} b_j}{1-\alpha} + \sum_{i=1}^{I} \gamma_i + \sum_{i=1}^{I}\sum_{t=0}^{T} \Big(\sum_{x^i \in X^i} \beta_i(x^i) x_t^i \Big) c_{it} \\
& (1-\alpha)\gamma_i + \sum_{t=0}^{T} \Big(x_t^i - \alpha E_{(x^i,u^i)} \mathbf{X'}_t^i \Big) c_{it} - \sum_{t=1}^{T}\sum_{j=1}^{J} \lambda_{tj} a_{ij} (u_t^i + x_t^i) \le f_i(x^i,u^i),\ (x^i,u^i) \in \mathcal{Y}_i,\ i \in \mathcal{I}, \\
& \lambda \ge 0, \\
& c_{it} \ge 0,\ i \in \mathcal{I},\ t = 0,1,\ldots,T.
\end{aligned}
$$

Unlike (ALLP1), it is not easy to identify a set of initial constraints $\mathcal{C}_0$ in (ALLP2) that guarantees existence of an optimal solution to (ALLP2)$_{\mathcal{C}_0}$. Empirically, we found after running some initial experiments, that starting with certain peculiar constraints that correspond to state-action pairs which include artificial actions, helps in this regard (see our numerical experiments in Section 5.5). Therefore, unlike in (ALLP1), we did not entirely drop artificial actions from (ALLP2).

Suppose that after solving (ALLP2)$_{\mathcal{C}}$ for some set of constraints $\mathcal{C}$, we obtain $\gamma^{\mathcal{C}}, c^{\mathcal{C}}, \lambda^{\mathcal{C}}$ as its optimal solution. Similar to (ALLP1), we define $\bar{\mathcal{Y}}_i \triangleq \{(x^i,u^i) : x^i \in X^i, u^i \in U^i(x^i)\}$ as the subset of $\mathcal{Y}_i$ that excludes artificial actions. The violation in the functional constraint corresponding to any $(x^i,u^i) \in \bar{\mathcal{Y}}_i$ for any $i \in \mathcal{I}$ is given by

$$
\mathcal{Z}_i^1(x^i,u^i) \triangleq (1-\alpha)\gamma_i^{\mathcal{C}} + \sum_{t=0}^{T} \Big(x_t^i - \alpha E_{(x^i,u^i)} \mathbf{X'}_t^i \Big) c_{it}^{\mathcal{C}} - \sum_{t=1}^{T}\sum_{j=1}^{J} \lambda_{tj}^{\mathcal{C}} a_{ij}(u_t^i + x_t^i) - f_i(x^i,u^i).
$$

To simplify this, recall that $\mathbf{X'}_0^i = n_i$ with probability $p_i(n_i)$ for $0 \le n_i \le N_i$, and $\mathbf{X'}_t^i = x_{t+1}^i + u_{t+1}^i$ for $t = 1,2,\ldots,T-1$ with probability one. Finally, $\mathbf{X'}_T^i = 0$ with probability one. Consequently,

$$
\sum_{t=0}^{T} E_{(x^i,u^i)} \mathbf{X'}_t^i c_{it}^{\mathcal{C}} = c_{i0}^{\mathcal{C}} E\boldsymbol{\Delta}_i + \sum_{t=1}^{T-1} c_{it}^{\mathcal{C}} (x_{t+1}^i + u_{t+1}^i).
$$

Recalling that $f_i(x^i,u^i) = G_i(x_0^i - \sum_{t=1}^{T} u_t^i) + \sum_{t=1}^{T} u_t^i C_i(t, D_i)$, we get

$$
\begin{aligned}
\mathcal{Z}_i^1(x^i,u^i) = (1-\alpha)\gamma_i^{\mathcal{C}} + (x_0^i - \alpha E\boldsymbol{\Delta}_i) c_{i0}^{\mathcal{C}} + \sum_{t=1}^{T-1} \Big(x_t^i - \alpha(x_{t+1}^i + u_{t+1}^i) \Big) c_{it}^{\mathcal{C}} + x_T^i c_{iT}^{\mathcal{C}} \\
- \sum_{t=1}^{T}\sum_{j=1}^{J} \lambda_{tj}^{\mathcal{C}} a_{ij}(u_t^i + x_t^i) - G_i(x_0^i - \sum_{t=1}^{T} u_t^i) - \sum_{t=1}^{T} u_t^i C_i(t, D_i).
\end{aligned}
$$

After algebraic simplification, the above is seen to be of the form $\mathcal{Z}_i^1(x^i,u^i) = \sum_{t=0}^{T} \delta_{it}^{\mathcal{C}} x_t^i + \sum_{t=1}^{T} \epsilon_{it}^{\mathcal{C}} u_t^i + \varepsilon_i^{\mathcal{C}}$, where

$$
\begin{aligned}
\delta_{i0}^{\mathcal{C}} &\triangleq c_{i0}^{\mathcal{C}} - G_i, \\
\delta_{i1}^{\mathcal{C}} &\triangleq c_{i1}^{\mathcal{C}} - \sum_{j=1}^{J} \lambda_{1j}^{\mathcal{C}} a_{ij}, \\
\delta_{it}^{\mathcal{C}} &\triangleq c_{it}^{\mathcal{C}} - \alpha c_{it-1}^{\mathcal{C}} - \sum_{j=1}^{J} \lambda_{tj}^{\mathcal{C}} a_{ij}, \text{ for } t = 2,3,\ldots,T \\
\epsilon_{i1}^{\mathcal{C}} &\triangleq -\sum_{j=1}^{J} \lambda_{1j} a_{ij} + G_i - C_i(1,D_i), \\
\epsilon_{it}^{\mathcal{C}} &\triangleq -\alpha c_{it-1}^{\mathcal{C}} - \sum_{j=1}^{J} \lambda_{1j} a_{ij} + G_i - C_i(1,D_i), \text{ for } t = 2,3,\ldots,T, \text{ and} \\
\varepsilon_i^{\mathcal{C}} &\triangleq (1-\alpha)\gamma_i^{\mathcal{C}} - \alpha c_{i0}^{\mathcal{C}} E\boldsymbol{\Delta}_i.
\end{aligned}
$$

Let $\mathcal{Z}_i^{1*}$ be the maximum violation among all constraints in the set $\bar{\mathcal{Y}}_i$. This is given by the optimal value of the linear integer program

$$
\begin{aligned}
\mathcal{Z}_i^{1*} = \max \ & \sum_{t=0}^{T} \delta_{it}^{\mathcal{C}} x_t^i + \sum_{t=1}^{T} \epsilon_{it}^{\mathcal{C}} u_t^i + \varepsilon_i^{\mathcal{C}} \qquad (24) \\
& x_0^i \le N_i, \\
& \sum_{t=1}^{T} u_t^i \le x_0^i, \\
& u_t^i + x_t^i \le M_i, \ t = 1,2,\ldots,T, \\
& u_t^i \ge 0, \text{ and integer, for } t = 1,2,\ldots,T, \\
& x_t^i \ge 0, \text{ and integer, for } t = 0,1,\ldots,T.
\end{aligned}
$$

Now consider any state-action pair $(x^i, \mu^i(x^i)) \in \mathcal{Y}_i \setminus \bar{\mathcal{Y}}_i$ for any $i \in \mathcal{I}$. The violation in the corresponding constraint is given by $\mathcal{Z}_i^2(x^i,\mu^i(x^i)) \triangleq (1-\alpha)\gamma_i^{\mathcal{C}} - \alpha E\boldsymbol{\Delta}_i c_{i0}^{\mathcal{C}} + \sum_{t=1}^{T} c_{it}^{\mathcal{C}} x_t^i - B$, which follows by algebra identical to that for $\mathcal{Z}_i^1(x^i,u^i)$ after noting that $\mu^i(x^i)_t + x_t^i = 0$ for $t = 1,2,\ldots,T$ and that $f_i(x^i,\mu^i(x^i)) = B$. Let $\mathcal{Z}_i^{2*}$ be the maximum violation among all constraints in the set $\mathcal{Y}_i \setminus \bar{\mathcal{Y}}_i$. It is easy to observe from the definition of $\mathcal{Z}_i^2(x^i,\mu^i(x^i))$ that this maximum violation occurs when $x_0^i = N_i$ and $x_t^i = M_i$ for $t = 1,2,\ldots,T$. Thus

$$
\mathcal{Z}_i^{2*} = (1-\alpha)\gamma_i^{\mathcal{C}} - \alpha E\boldsymbol{\Delta}_i c_{i0}^{\mathcal{C}} + N_i c_{i0}^{\mathcal{C}} + M_i \sum_{t=1}^{T} c_{it}^{\mathcal{C}} - B.
$$

Now we set $\mathcal{Z}_i^* = \max(\mathcal{Z}_i^{1*}, \mathcal{Z}_i^{2*})$ and the maximum constraint violation problem reduces to $\mathcal{Z}^* \triangleq \max_{i \in \mathcal{I}} \mathcal{Z}_i^*$. Also note that once the state $x_0^i = N_i$, $x_t^i = M_i$ for $t = 1,2,\ldots,T$ for each $i \in \mathcal{I}$ and the corresponding artificial actions have been included in the set of constraints $\mathcal{C}$ and an optimal solution to $(\text{ALLP2})_{\mathcal{C}}$ has been found, all artificial actions can be removed from consideration in

subsequent iterations of constraint generation. This is because all constraints corresponding to state-action pairs that include artificial actions will be satisfied in these iterations. Consequently, the constraint violation problem in these iterations simplifies to $\max_{i \in \mathcal{I}} \mathcal{Z}_i^{1*}$.

5.4 Retrieving decisions from the approximate value function

Let λ_{tj}^* for $t = 1, 2, \ldots, T$ and $j \in \mathcal{J}$, γ_i^* for $i \in \mathcal{I}$, and c_{it}^* for $t = 0, 1, \ldots, T$ and $i \in \mathcal{I}$, denote the values of variables in (ALLP2) after terminating constraint generation. Now suppose that for some state $x \in \bar{X}$, we wish to compute a decision vector that is myopic with respect to the approximation

$$V(x') = \Phi(x') \approx \Phi^{\lambda^*}(x') = -\frac{\sum_{t=1}^{T}\sum_{j=1}^{J}\lambda_{tj}^* b^j}{1-\alpha} + \sum_{i=1}^{I}\Phi_i^{\lambda^*}(x'^i) \approx -\frac{\sum_{t=1}^{T}\sum_{j=1}^{J}\lambda_{tj}^* b^j}{1-\alpha} + \sum_{i=1}^{I}(\gamma_*^i + \sum_{t=0}^{T} c_{it}^* x_t'^i).$$

After substituting this in the right hand side of Bellman's equation (20) and then following steps similar to Section 4.4, the decision retrieval problem simplifies to the linear integer program

$$\begin{aligned}
&\min \sum_{i=1}^{I}\Big[\Big(C_i(1;D_i) - G_i\Big)u_1^i + \sum_{t=2}^{T}\Big(C_i(t;D_i) - G_i + \alpha c_{it-1}^*\Big)u_t^i\Big] \\
&\sum_{t=1}^{T} u_t^i \le x_0^i,\ i \in \mathcal{I}, \\
&\sum_{i=1}^{I} a_{ij}u_t^i \le b^j - \sum_{i=1}^{I} a_{ij}x_t^i,\ j \in \mathcal{J},\ t = 1, 2, \ldots, T, \\
&u_t^i \ge 0 \text{ and integer},\ i \in \mathcal{I},\ t = 1, 2, \ldots, T.
\end{aligned}$$

5.5 Numerical experiments

Similar to allocation scheduling, we used a test bed of randomly generated problems to compare the performance of our Lagrangian policy with a myopic policy. We set the number of job types to $I = 5, 10, 15$. The maximum number of type-i arrivals in one period, denoted N_i, were set to DU$[1, 20]$. Once N_i was sampled, the probability mass function $p_i(\cdot)$ was obtained by normalizing $N_i + 1$ uniform $(0, 1)$ random variables. The number of resources were set to $J = 1, 2$. The resource availabilities and consumption values were constructed exactly as for allocation scheduling. Tightness ratio values were set to $\rho = 0.25, 0.5$ for $I = 5, 10$ and $\rho = 0.25$ for $I = 15$[‡]. The booking horizon was set to $T = 10$ and the deadline for type-i jobs was set to DU$[1, T]$. Delay costs for all $i \in \mathcal{I}$ were calculated using the expression

$$C_i(t; D_i) = F_i \times \max(t - D_i, 0),\ t = 1, \ldots, T$$

from [27]. The delay cost coefficient F_i was set to DU$[1, 100]$. The rejection cost G_i per job of type i was set to a random multiple r_i of the cost of scheduling that job on the last day of the booking horizon. In particular, we set $G_i = r_i \times C_i(T; D_i)$. We set r_i to a uniform $(0, 2)$ random variable. Similar to allocation scheduling, following Adelman and Mersereau [2], the discount factor

[‡]We empirically observed that larger values of ρ than those considered here generated excessive amounts of resources in relation to our job arrival streams. This essentially removed bottlenecks from many of the problem instances leading to the existence of zero-cost policies and rendering the problem uninteresting.

Problem set	I	J	ρ	CPU seconds	Lagrangian	Myopic	% improvement
1	5	1	0.25	10	15748	18400	14.41
2	5	1	0.5	9	9895	11767	15.91
3	5	2	0.25	9	16120	17126	5.87
4	5	2	0.5	9	12945	13920	7.00
5	10	1	0.25	30	9737	12812	24.00
6	10	1	0.5	32	1008	903	-11.63
7	10	2	0.25	29	14978	18853	20.55
8	10	2	0.5	31	1976	2118	6.70
9	15	1	0.25	60	4758	6661	28.57
10	15	2	0.25	66	4625	8558	45.96
Average	-	-	-	28.5	9179	11112	17.39

Table 2: Results for 10 advanced scheduling problem sets. For each problem set, the cost estimates reported are averages over 50 randomly generated problem instances. Thus the table summarizes results for 500 advanced scheduling problems.

was set to 0.85. In summary, our problem sets are characterized by parameters I, J, ρ. For each combination of parameter values, we solved 50 randomly generated problem instances.

The initial set of constraints in our constraint generation algorithm was constructed using state vectors $x^i = (N_i; s; s; \ldots; s)$ with $s \in \{0, 1, \ldots, M_i\}$ for all $i \in \mathcal{I}$, and the real and artificial action vectors $u^i = (0; 0; \ldots; 0)$ and $u^i = (-s; -s; \ldots; -s)$ for all $i \in \mathcal{I}$, respectively. We terminated the constraint generation procedure when the absolute percentage gap $\frac{I}{1-\alpha}\left|\frac{\mathcal{Z}^*}{H_C}\right|$ dropped below 0.5%. For each problem instance, the infinite-horizon discounted expected cost incurred by the Lagrangian or the myopic policy was estimated by averaging the total discounted cost incurred over 200 independent simulations of 50 periods. Each of the 200 simulations was started in the "empty" state $x_t^i = 0$ for $t = 0, 1, \ldots, T$. As suggested in [2], running a simulation for 50 periods sufficiently well-approximates the infinite-horizon costs because $0.85^{50} \approx 10^{-4}$ and hence the tail costs beyond the 50th period are relatively insignificant. These cost estimates, averaged over the 50 independent problem instances for each problem set are reported in Table 2. The table also lists the percentage improvement in cost obtained by the Lagrangian policy over the myopic policy for each problem set. The average CPU time needed to complete constraint generation is also reported in seconds for each problem set. The table shows that the Lagrangian policy incurs lower cost than the myopic policy in nine of the ten problem sets. The last row of the table shows averages over all 500 problem instances. Note that the average improvement achieved by the Lagrangian policy over the myopic policy in these 500 instances equals $100 \times (11112 - 9179)/11112$, which is 17.39%. A paired t-test for the difference between myopic and Lagrangian cost estimates over the 500 instances was statistically significant at the 0.05 level with p-value of the order 10^{-19} and returned $[1518.4, 2347.1]$ as the 95% confidence interval for the true mean of the difference.

6 Conclusions and future work

We proposed a hybrid Lagrangian relaxation and column generation approach for approximate solution of large-scale weakly coupled MDPs. This method was applied to two classes of dynamic stochastic scheduling problems. The performance of resulting Lagrangian decisions was compared with that of myopic decisions on a test bed of randomly generated problems using simulations.

The relative benefit of Lagrangian decisions was higher in advanced scheduling problems than in allocation scheduling problems. Several additional features could be incorporated into our problem statements for allocation and advanced scheduling. It would be interesting to investigate whether the resulting MDP models continue to be weakly coupled so that our hybrid Lagrangian relaxation and column generation approach could be applied.

One such variation is where jobs leave the queue, at a penalty cost to the planner, rather than waiting too long. For instance, if an inpatient scan request is not fulfilled within a certain amount of time, the inpatient may have to be served using an emergency department scanner, or special mobile scanning equipment. A related modification involves non-linear holding costs that depend on how long a job has waited in queue.

Our advanced scheduling problem statement required that arriving service requests be "handled" immediately. An alternative approach worth considering in the future is where these decisions can be postponed, in effect placing some of the arriving requests on a waiting list. Intuitively, this gives the planner additional flexibility hopefully improving service quality and other system performance metrics. It may also be possible to incorporate cancelations into our advanced scheduling problem.

The problems discussed in this paper assumed stationary data. Even though ubiquitous in infinite-horizon decision models in Operations Research [29, 31], this assumption may not be realistic. For instance, in the health care setting, when each time-period corresponds to a day, arrival probabilities may vary from one day to another, but are likely to have a period of five or seven (depending on whether service is provided on weekdays only or on weekends also). Similarly, if the hospital provides diagnostic imaging services from 7:00 AM to 5:00 PM, patient arrival probabilities will vary over this 10-hour period, but are likely to follow the same trend throughout the week; that is, the period is one day. Such patterns were observed in the data collected in our recent work in [11]. It would be interesting to model allocation and advanced scheduling problems with such periodic data.

More generally, our hope is that researchers would find new applications of large-scale weakly coupled MDPs and our hybrid Lagrangian relaxation and constraint generation method would help find their approximate solutions.

Acknowledgments

This research was funded in part by the Department of Radiology at the University of Washington Medical Center. The authors also appreciate financial support from the National Science Foundation through grant CCF-0830380. The authors thank the IBM Academic Initiative for making their CPLEX Optimization Studio available at no charge.

References

[1] D Aberdeen, S Thiebaux, and L Zhang. Decision-theoretic military operations planning. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling*, Whistler, British Columbia, Canada, 2004.

[2] D Adelman and A J Mersereau. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research*, 56(3):712–727, 2008.

[3] M M Ali and X Song. A performance analysis of a discrete-time priority queueing system with correlated arrivals. *Performance Evaluation*, 57(3):307–339, 2004.

[4] E Altman. Applications of markov decision processes in communication networks. In E Feinberg and A Shwartz, editors, *Handbook of Markov Decision Processes: Methods and Applications*, chapter 16, pages 489–536. Springer, 2002.

[5] D P Bertsekas. *Dynamic Programming and Optimal Control*, volume 1 and 2. Athena Scientific, Nashua, NH, USA, 2007.

[6] P C Chu and J E Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(3):63–86, 1998.

[7] S A Erdogan and B T Denton. Surgery planning and scheduling: A literature review. *Working Paper*, 2009.

[8] P Gao, S Wittevrongel, and H Bruneel. Discrete-time multiserver queues with geometric service times. *Computers & Operations Research*, 31(1):81–99, 2004.

[9] L Garg, S McClean, B Meenan, and P Millard. A non-homogeneous discrete time markov model for admission scheduling and resource planning in a cost or capacity constrained healthcare system. *Health Care Management Science*, 13(2):155–169, 2010.

[10] Y Gocgun. *Approximate dynamic programming for dynamic stochastic resource allocation with applications to healthcare*. PhD thesis, University of Washington, Seattle, WA, USA, 2010.

[11] Y Gocgun, B Bresnahan, A Ghate, and M Gunn. A Markov decision process approach to multi-category patient scheduling, forthcoming in Artificial Intelligence in Medicine, 2011.

[12] Y Gocgun and A Ghate. A Lagrangian approach to dynamic resource allocation. In B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yucesan, editors, *Proceedings of the Winter Simulation Conference*, pages 3330–3338, Baltimore, 2010.

[13] G A Godfrey and W B Powell. An adaptive dynamic programming algorithm for dynamic fleet management, i: Single period travel times. *Transportation Science*, 36(1):21–39, 2002.

[14] G A Godfrey and W B Powell. An adaptive dynamic programming algorithm for dynamic fleet management, ii: Multiperiod travel times. *Transportation Science*, 36(1):40–54, 2002.

[15] L V Green, S Savin, and B Wang. Managing patient service in a diagnostic medical facility. *Operations Research*, 54(1):11–25, 2006.

[16] D Gupta and L Wang. Revenue management for a primary-care clinic in the presence of patient choice. *Operations Research*, 56(3):576–592, 2008.

[17] M J Harrison. Dynamic scheduling of a multiclass queue: Discount optimality. *Operations Research*, 23(2):270–282, 1975.

[18] J Hawkins. *A Lagrangian decomposition approach to weakly coupled dynamic optimization problems and its applications*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2003.

[19] B Jahn, E Theurl, U Siebert, and K P Pfeiffer. Tutorial in medical decision modeling incorporating waiting lines and queues using discrete event simulation. *Value in Health*, 13(4):501–506, 2010.

[20] A Khamisy and M Sidi. Discrete-time priority queues with two-state markov modulated arrivals. *Stochastic Models*, 8(2):337–357, 1992.

[21] R. Kolisch and S. Sickinger. Providing radiology health care services to stochastic demand of different customer classes. *OR Spectrum*, 30:375–395, 2008.

[22] K Laevens and H Bruneel. Discrete-time multiserver queues with priorities. *Performance Evaluation*, 33(4):249–275, 1998.

[23] J Li, D E Blumenfeld, N Huang, and J M Alden. Throughput analysis of production systems: recent advances and future topics. *International Journal of Production Research*, 47(14):3823–3851, 2009.

[24] P B Luh, X Miao, S Chang, and D A Castanon. Stochastic task selection and renewable resource allocation. *IEE Transactions on Automatic Control*, 34(3):335–338, 1989.

[25] S Ndreca and B Scoppola. Discrete time GI/Geom/1 queueing system with priority. *European Journal of Operational Research*, 189(3):1403–1408, 2008.

[26] L G Nadal Nunes, S V de Carvalho, and R de Cássia Meneses Rodrigues. Markov decision process applied to the control of hospital elective admissions. *Artificial Intelligence in Medicine*, 47(2):159 – 171, 2009.

[27] J Patrick, M L Puterman, and M Queyranne. Dynamic multi-priority patient scheduling for a diagnostic resource. *Operations Research*, 56(6):1507–1525, 2008.

[28] A Pavitsos and E G Kyriakidis. Markov decision models for the optimal maintenance of a production unit with an upstream buffer. *Computers & Operations Research*, 36(6):1993 – 2006, 2009.

[29] M Puterman. *Markov Decision Processes*. John Wiley and Sons, New Jersey, 1994.

[30] K W Ross and D Tsang. The stochastic knapsack problem. *IEEE Transactions on Communications*, 37(7):740–747, 1989.

[31] S M Ross. *Introduction to stochastic dynamic programming*. Academic Press, New York, 1983.

[32] T Tolio. *Design of Flexible Production Systems – Methodologies and Tools*. Springer, Berlin, Germany, 2009.

[33] H Topaloglu. Using lagrangian relaxation to compute capacity-dependent bid prices in network revenue management. *Operations Research*, 57(3):637–649, 2009.

[34] H Topaloglu and W B Powell. A distributed decision-making structure for dynamic resource allocation using nonlinear functional approximations. *Operations Research*, 53(2):281–297, 2005.

[35] D Vengerov. A reinforcement learning approach to dynamic resource allocation. *Engineering Applications of Artificial Intelligence*, 20(3):383–390, 2007.

[36] J Walraevens, B Steyaert, and H Bruneel. Performance analysis of a single-server ATM queue with a priority scheduling. *Computers & Operations Research*, 30(12):1807–1829, 2003.

[37] J Walraevens, B Steyaert, and H Bruneel. Analysis of a discrete-time preemptive resume priority buffer. *European Journal of Operational Research*, 186(1):182–201, 2008.

[38] L Wolsey. *Integer Programming*. Wiley-Interscience, New York, NY, USA, 1998.