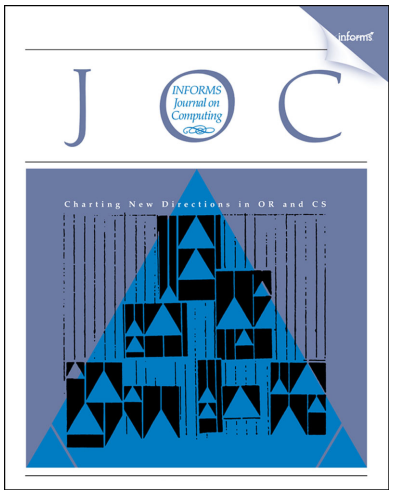


INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Fast Approximation Methods for Online Scheduling of Outpatient Procedure Centers

Bjorn P. Berg, Brian T. Denton

To cite this article:
Bjorn P. Berg, Brian T. Denton (2017) Fast Approximation Methods for Online Scheduling of Outpatient Procedure Centers. INFORMS Journal on Computing 29(4):631-644. https://doi.org/10.1287/ijoc.2017.0750

http://pubsonline.informs.org/journal/ijoc/

INFORMS JOURNAL ON COMPUTING
Vol. 29, No. 4, Fall 2017, pp. 631–644
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

Fast Approximation Methods for Online Scheduling of Outpatient Procedure Centers

Bjorn P. Berg,[a] **Brian T. Denton**[b]

[a] Mayo Clinic, Rochester, Minnesota 55905; [b] Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109

Contact: berg.bjorn@mayo.edu (BPB); btdenton@umich.edu (BTD)

Received: January 27, 2015
Revised: March 3, 2016; October 13, 2016; December 20, 2016
Accepted: December 29, 2016
Published Online: August 8, 2017

https://doi.org/10.1287/ijoc.2017.0750

Abstract. This paper presents a new model for online decision making. Motivated by the healthcare delivery application of dynamically allocating patients to procedure rooms in outpatient procedure centers, the online stochastic extensible bin-packing problem is described. The objective is to minimize the combined costs of opening procedure rooms and utilizing overtime to complete a day's procedures. The dynamic patient-allocation decisions are made in an uncertain environment where the number of patients scheduled and the procedure durations are not known in advance. The resulting optimization model's tractability focuses the paper's attention on approximation methods and a special case that is amenable to decomposition-based solution methods. Theoretical performance guarantees are presented for list-based approximation methods as well as an approximation that is common in practice, where procedure rooms are reserved for patient groups in advance. Numerical results based on a real outpatient procedure center demonstrate the favorable results of the list-based approximations based on their average and worst case performances, as well as their computational requirements. Further, the numerical experiments show that the policy of reserving procedure rooms for patient groups in advance can perform poorly. These results are contrary to common practice and favor alternative, and still easy-to-implement, policies.

History: Accepted by Allen Holder, Area Editor for Application in Biology, Medicine, and Health Care.
Funding: This material is also based in part on work supported by the National Science Foundation [Grant CMMI 1258323].
Supplemental Material: The online supplement is available at https://doi.org/10.1287/ijoc.2017.0750.

Keywords: healthcare: outpatient procedure centers • scheduling: approximations • stochastic programming

1. Introduction

This article considers an online decision process that commonly arises in healthcare settings: the sequential assignment of procedures to rooms. The immediate motivation is based on outpatient procedure centers (OPCs), where patients are allocated to procedure rooms dynamically. In this setting there are two types of patients: routine and urgent add-ons. Routine patients are allocated to procedure rooms in advance, whereas the urgent add-on patients need to be allocated at a later time (e.g., the morning of the planned procedures). We relate this problem context to that of the well-known bin-packing problem with several generalizations.

The bin-packing problem is a well-studied problem, and variants arise in many applications, including operating room management (Van Houdenhoven et al. 2007), steel slab design (Dawande et al. 2004), and Internet advertising (Adler et al. 2002). Healthcare delivery applications prompt us to relax several assumptions in the classic bin-packing problem. The first is the assumption that the number of items to be allocated is known a priori. In this generalization, referred to as the *dynamic*, or *online, bin-packing problem* (Coffman et al. 1983), items are sequentially allocated to bins following the decision of how many bins to open. The second assumption we relax is to allow the size of the items to be stochastic to reflect possible uncertainty in service times. Finally, we relax the assumption that the size of the bins is fixed and allow the size of the bins to be extended, i.e., that there is an option to increase the size of the bins at some (overtime) cost. Together, the relaxation of these assumptions lead to the formulation of a new model we refer to as the *dynamic extensible bin-packing* (DEBP) problem.

In the problems we consider, items are allocated sequentially to bins. Items to be allocated may arise one at a time or in batches. The size of the batches (and thus the total number of items) at each stage may not be known with certainty. The size of individual items also may not be known with certainty. This problem can be formulated as a multistage stochastic integer program. In the first stage, a decision is made about how many bins to open, subject to a fixed cost per bin and an upper limit on the total number of available bins.

In each subsequent stage, allocation decisions must be made for the assignment of the current batch of items to bins. The size of each batch is not observed prior to each stage; only the probability distribution for the size of each batch is known in advance. Finally, in the last stage, the sizes of the items are observed and overtime costs are incurred if the total number of items in a bin exceeds the bin size.

The most general version of this problem is intractable. Therefore, we focus on (a) a special case for which decomposition methods can achieve exact or near optimal solutions and (b) approximations, for the general case, for which the worst case performance ratio can be bounded above. The special case is a three-stage (two-batch) case of the problem. This is motivated by a problem commonly faced by OPC managers in which a batch of *routine* patients is scheduled in advance and a second batch of *add-on* patients arrives on short notice. In this problem, the size of the first batch is known with certainty and the size of the second batch is only observed following the allocation decisions of the first batch. Using nonanticipativity constraints, we reformulate this as a two-stage stochastic integer program in which all the binary decisions appear in the first stage. This reformulation allows us to apply decomposition-based solution methods, which we compare for a collection of realistic test instances of the problem based on a gastroenterology and hepatology clinic at Mayo Clinic in Rochester, Minnesota. In addition to this special case, we also test approximations for the more difficult problem in which there are more than two batches.

The remainder of this article is organized as follows: In Section 2 we discuss previous work related to extensible bin-packing, online bin-packing, and related research from healthcare settings. In Section 3, we introduce a multistage stochastic integer programming formulation for the DEBP problem and identify special cases that are important for healthcare delivery. In Section 4 we provide bounds on the optimal objective function value for the stochastic integer program and a worst case performance measure bound for a fast approximation method. In Section 5 we present numerical results in the context of a case study of a real OPC. Key insights are summarized in Section 6 along with future research directions.

2. Literature Review

Bin-packing problems have been thoroughly examined in various contexts. Because of the combinatorial challenges associated with this NP-hard problem, much of the focus has been on the development of bounds (Martello and Toth 1990a, b; Fekete and Schepers 2001) and approximation algorithms (Fernandez de La Vega and Lueker 1981, Karmarkar and Karp 1982). The literature reviewed in this section focuses on variants of the bin-packing problem that are closely related to the problem considered in this article. However, the interested reader is referred to a survey by Coffman et al. (1996) for further discussion.

The online bin-packing problem received significant attention during the early study of bin-packing problems. In the online bin-packing problem, items are allocated to bins one at a time without knowledge of future items. Well-known approximation algorithms include *First Fit* (the next item is allocated to the first bin with sufficient remaining capacity), *Next Fit* (the next item is allocated to the same bin as the previous item, or a new bin is opened if capacity is insufficient), *Revised First Fit* (items are classified based on their size and *First Fit* is applied), and *Best Fit* (the next item is allocated to the bin with the least, yet still sufficient, capacity). Performance bounds have been derived for each of these approximations (Ullman 1971, Johnson 1974, Yao 1980). While improved bounds continued to be developed in the past decade, Seiden (2002) notes that attention to the online bin-packing problem has diminished.

Online bin-packing problems with stochastic item sizes have also been studied. For example, Bentley et al. (1984) analyzed common deterministic online bin-packing algorithms, such as First Fit and Best Fit, for stochastic online bin-packing. The authors analyzed the expected performance of the algorithms for certain distributions of item size. Hoffmann (1982) demonstrated that online algorithms that are based on the distribution of the item sizes can be adjusted through a parameter to obtain a desired expected performance ratio as the number of items increases. Later studies focused on uncertainty in the item arrival process, where queues of items develop as a result of limited resource availability. Stochastic online bin-packing processes were initially motivated by problems in bandwidth packing (Coffman and Stolyar 2001), where traditional First Fit and Best Fit algorithms were analyzed in the stochastic process setting. Gamarnik and Squillante (2005) further analyzed the stability and performance of such algorithms focusing on the development of matrix-analytic solution methods. Shah and Tsitsiklis (2008) extended the work by Coffman and Stolyar (2001) and Gamarnik and Squillante (2005), focusing on the average queue size relative to an arrival process parameter.

In the *extensible* bin-packing problem, the size of the bins can be extended, if necessary, at a cost proportional to the size of the bin extension. This problem was initially considered by Dell'Olmo et al. (1998), where the longest processing time approximation was applied and a worst case performance bound was developed for special cases. Later, Coffman and Lueker (2006) analyzed approximation algorithms for extensible bin-packing and provided a fully polynomial time-asymptotic approximation scheme motivated by previous bin-packing problem approximation

schemes. In the online extensible bin-packing problem developed by Ye and Zhang (2004), the authors considered special cases, where bins are assumed to be of unequal size, as well as the two- and three-bin cases and analyze algorithms and performance bounds. Closely related to the problem considered in this article is the work by Speranza and Tuza (1999) on online algorithms for machine scheduling with extendable work time. Based on a fixed (and known) number of machines, the authors provide worst case analysis for online algorithms.

Previous work has applied bin-packing models to healthcare settings. Dexter and Traub (2002) used a simulation model to evaluate heuristics for dynamically allocating surgeries to operating rooms for the purpose of maximizing operating room utilization. The *Earliest Start Time* heuristic allocates the next surgery to the operating room with the earliest available start time. The *Latest Start Time* heuristic allocates the next surgery to the operating room with the latest available start time, such that the expected surgery duration is less than the available time remaining without using overtime. While the Earliest Start Time heuristic maximized operating room utilization, the Latest Start Time heuristic provided a more balanced workload across operating rooms.

Hans et al. (2008) considered the robust surgery-loading problem, where surgeries are allocated to operating rooms to maximize utilization while limiting the use of overtime. The problem was analyzed as an extensible bin-packing problem with unequal bin sizes and item size uncertainty. The authors compare the performance of heuristics, including *First Fit* and *Longest Processing Time*, and conclude that improved operating room management would significantly increase operating room capacity without increasing overtime. The authors highlight the impact of collecting and using historical surgery durations in mitigating the effects of surgery duration variability.

Denton et al. (2010) developed a two-stage stochastic integer program for allocating surgeries to operating rooms. Similar to the problem studied in this article, the duration of surgeries are unknown; however, the problem formulation is for a static (not dynamic) allocation process in which the total number of items to be allocated to bins is known a priori. Bounds and valid inequalities were developed to improve the solution time for the two-stage stochastic integer program using the L-shaped method. A comparison of solution quality was presented based on the optimal solution to the stochastic integer program, the *Longest Processing Time* heuristic, and the solution to a robust optimization model. The authors show that instances of their model corresponding to practical surgery scheduling problems can be solved in reasonable time frames, but the heuristic and the robust formulation obtain solutions that perform very well with much shorter computation time. Further, the heuristic has the benefit of being easy to implement, and the robust formulation has the ability to limit worst case performance.

This article contributes to the literature in the following ways. First, as opposed to the above-cited literature, we consider a new model, where the decisions are made sequentially (online), item sizes are stochastic, the bins are extensible, and the number of bins to open is a decision variable. The bin-packing literature rarely includes extensible bins, much less stochastic extensible bin-packing. Second, we include uncertainty in the number of items to be allocated. While previous online bin-packing studies often assume no prior information about item size or number of items, in practice it is often the case that historical data can be used for probability distributions for both item sizes and number of items to be allocated. Third, we provide an analysis of the model that yields insight into the optimal solution as well as special cases that commonly arise in healthcare delivery. Further, we analyze fast approximations that are applicable to large instances of the problem and we provide lower and upper bounds on the performance ratio of these approximations. This approach differs from previous work by focusing on fast approximations that require little information and therefore are easier to implement in practice. Finally, we present a case study in which the model is populated using data from a real OPC, to evaluate its practical and managerial importance and implications. This analysis provides further credence for the use of the proposed approximations in a context where access to advanced solvers is often limited and algorithms that are intuitive, fast, perform well, and are easy to implement are preferred to exact solution methods.

3. Model Formulation

The traditional bin-packing problem requires two primary decisions: first, deciding the appropriate number of bins needed to accommodate a packing, or allocation, of a set of items, and second, deciding on the allocation of items to the bins so as to minimize the number of required bins. An extension of this, in which bins are extensible, can be stated as

$$\min_{x,y}\left\{\sum_{j=1}^{m} x_j + c^v \sum_{j=1}^{m} \max\left\{0, \sum_{i=1}^{n} d_i y_{ij} - S x_j\right\} \;\middle|\; \sum_{j=1}^{m} y_{ij} = 1, \quad \forall\, i;\ x_j, y_{ij} \in \{0,1\}\ \forall\, j\right\}, \quad (1)$$

where $x \in \mathbb{B}^m$ and $y \in \mathbb{B}^{m\times n}$ are vectors of binary decision variables for which of m bins ($j = 1, \ldots, m$) to open and how to allocate n items ($i = 1, \ldots, n$) to bins, respectively; S represents the capacity of the bins; and $c^v \in \mathbb{R}^1_+$ is the per unit time cost of extending a bin. In this model, the number of items to be allocated as well as the size of each item d_i, are known in advance.

3.1. Dynamic Extensible Bin-Packing

In this section we develop the DEBP problem as a multistage stochastic integer program. We follow the conventional notation and presentation and refer the reader to surveys on stochastic integer programming for further detail (Louveaux and Schultz 2003, Klein Haneveld and van der Vlerk 1999, Birge and Louveaux 1997). In the DEBP problem, the number of items to be allocated is realized in discrete stages. At each stage, a random number of items is observed. The batch size, $b^t(\sigma^t)$ in stage $t = 1, \dots, T$, is a discrete and finite random variable with probabilities $p^t_{\sigma^t}$ for each of the outcomes indexed by $\sigma^t = \varnothing, 0, \dots, b^t_u$, where $\varnothing$ represents the termination of the arrival process and b^t_u is an upper limit on the batch size in stage t. The set of $b^t(\sigma^t)$ items that are to be allocated in stage t is represented by $I^t(\sigma^t) = \{0, \dots, \sigma^t\}$ and indexed by $i = 1, \dots, |I^t(\sigma^t)|$. The size of each item i within a batch is a random variable $d_i(\omega^t)$, where $\omega^t = 1, \dots, K$ is an index of a finite number of random scenarios for the collective item sizes at stage t. Note that the index for the batch size scenarios differs from the index for the item size scenarios, as each identifies a different scenario that may be observed. The DEBP problem can be written as

$$\min_x \left\{ \sum_{j=1}^{m} c^f x_j + \mathcal{Q}^1(x) \,\middle|\, x_j \in \{0,1\},\ \forall j \right\}, \qquad \text{(DEBP)}$$

where $x \in \mathbb{B}^m$ is a vector of binary decisions, $x_1, \dots, x_m$, that determines which of m bins to open at stage 0, $c^f \in \mathbb{R}^1_+$ is the cost of opening a bin, and $\mathcal{Q}^1(x)$ represents the expected *cost-to-go*, referred to as the *stage 1 recourse function*. This stage 1 recourse function $\mathcal{Q}^1(x)$ can be written as

$$\min_{y^1} \ E_{\omega^1,\sigma^1}\left[(p^2_\varnothing) c^v \sum_{j=1}^{m} \max\left\{0, \sum_{i \in I^1(\sigma^1)} d_i(\omega^1) y^1_{ij}(\sigma^1) - S x_j \right\}\right] + (1 - p^2_\varnothing)\mathcal{Q}^2(y^1) \qquad (2a)$$

$$\text{s.t. } y^1_{ij}(\sigma^1) \le x_j, \quad \forall j, \forall i \in I^1(\sigma^1), \qquad (2b)$$

$$\sum_{j=1}^{m} y^1_{ij}(\sigma^1) = 1, \quad \forall i \in I^1(\sigma^1), \qquad (2c)$$

$$y^1_{ij}(\sigma^1) \in \{0,1\}, \quad \forall j, i, \sigma^1, \qquad (2d)$$

where $y^1 \in \mathbb{B}^{m \times b^1_u}$ is a vector of item allocation decisions in the first stage. We assume that $c^f \le S c^v$ to avoid the trivial case where it is optimal to open a single bin. This assumption is reasonable for any practical problem. The expectation of extending the bins is taken over all scenarios.

The value of the recourse function in (2a) is the sum of the expected cost of extending the bins in stage 1 if no more items are to be allocated and the expected costs-to-go if further items are to be allocated. Constraints (2b) and (2c) require that the first-stage items in scenario σ^1 be allocated to an open bin.

The recourse function for stage t, $\mathcal{Q}^t(y^{t-1})$, is written as

$$\min_{y^t} \ E_{\omega^t,\sigma^t}\left[(p^{t+1}_\varnothing) c^v \sum_{j=1}^{m} \max\left\{0, \sum_{i \in \{I^k(\sigma^k) \mid k \le t\}} d_i(\omega^t) y^t_{ij}(\sigma^t) - S x_j \right\}\right] + (1 - p^{t+1}_\varnothing)\mathcal{Q}^{t+1}(y^t), \qquad (3a)$$

$$\text{s.t. } y^t_{ij}(\sigma^t) \le x_j, \quad \forall j, i \in I^t(\sigma^t), \qquad (3b)$$

$$\sum_{j=1}^{m} y^t_{ij}(\sigma^t) = 1, \quad \forall i \in I^t(\sigma^t), \qquad (3c)$$

$$y^t_{ij}(\sigma^t) = y^{t-1}_{ij}(\sigma^{t-1}), \quad \forall j, \forall i \in I^{t-1}(\sigma^{t-1}), \qquad (3d)$$

$$y^t_{ij}(\sigma^t) \in \{0,1\}, \quad \forall j, i, \sigma^t, \qquad (3e)$$

where $y^t \in \mathbb{B}^{m \times b^t_u}$ is the vector of item-allocation decisions at stage t and $y^{t-1} \in \mathbb{B}^{m \times b^{t-1}_u}$ is the vector of item-allocation decisions made in the previous period, $t-1$.

In the DEBP problem, allocation decisions from previous stages are fixed, which is enforced by linking constraints (3d) from stage $t-1$ to stage t. Finally, the recourse function for stage T, $\mathcal{Q}^T(y^{T-1})$, can be written as

$$\min_{y^T} \ E_{\omega^T,\sigma^T}\left[c^v \sum_{j=1}^{m} \max\left\{0, \sum_{i \in \{I^k(\sigma^k) \mid k \le T\}} d_i(\omega^T) y^T_{ij}(\sigma^T) - S x_j \right\}\right] \qquad (4a)$$

$$\text{s.t. } y^T_{ij}(\sigma^T) \le x_j, \quad \forall j, i \in I^T(\sigma^T), \qquad (4b)$$

$$\sum_{j=1}^{m} y^T_{ij}(\sigma^T) = 1, \quad \forall i \in I^T(\sigma^T), \qquad (4c)$$

$$y^T_{ij}(\sigma^T) = y^{T-1}_{ij}(\sigma^{T-1}), \quad \forall j, \forall i \in \{I^k(\sigma^k) \mid k = T-1\}, \qquad (4d)$$

$$y^T_{ij}(\sigma^T) \in \{0,1\}, \quad \forall j, i, \sigma^T. \qquad (4e)$$

There are several notable differences between (DEBP) and the extensible bin-packing problem in (1). For example, the objective in (DEBP) is to minimize the total cost of opening bins and the expected cost of extending their capacities. Whereas (1) is an offline problem, where all item information is known before allocation decisions are made, in (DEBP), items arrive sequentially in batches of an uncertain number of items, with uncertain sizes.

3.2. Special Case of an OPC

We emphasize a special case of the above formulation that is relevant to many OPCs and hospital operating rooms. In this context, patients (items) are allocated to procedure rooms (bins) in two phases (batches). The first batch of patients is scheduled in advance (often at least 48–72 hours in advance) based on requests for a particular day of service that have arisen weeks or months in advance. After this schedule for the first set of patients is fixed, it is difficult to change because

of preparations that occur shortly before the day of surgery. The second batch is comprised of urgent add-on patients whose needs arise during the 48–72 hours before, and up to, the beginning of the day of service. Thus, while the size of the second batch is unknown prior to the day being scheduled, the urgent requests are known at the beginning of the day they are to be scheduled. The challenge for decision makers, and where this model contributes, is that the number of procedure rooms to make available must be decided on prior to the arrival of a random number of urgent add-on patients; moreover, the schedule for the first batch of patients must be set without knowing how many urgent add-on patients will arise or how long the procedures will take. Thus, in this three-stage problem ($T = 3$), patients are allocated to procedure rooms in two batches. The first batch size is of size n. The size of the second batch, however, is uncertain. With respect to formulation (DEBP), this special case can be interpreted as the scenario resulting from $p^1_n = 1$ and $p^3_\varnothing = 1$, where the first-stage batch is of size n with certainty and the arrival process terminates following the arrival of the second batch.

The two-batch problem is a three-stage stochastic integer program (referred to as 3SIP). In the first stage, decisions are made about which bins to open and how to allocate the first batch of items. In the second stage, the second batch of items is allocated to bins. Finally, in the third stage, bin extension decisions are made following the observation of all the item sizes. While our formulation allows for a theoretically unlimited accommodation of urgent requests using overtime, the use of overtime is balanced with additional procedure rooms based on parameter estimates, as discussed in Section 5. In other words, overtime is used to extend the nominal procedure room time if the procedure rooms procedure durations extend beyond this nominal amount of time. In the online supplement we show how this special case of (DEBP) can be reformulated as a two-stage stochastic program with all the binary decisions in the first stage (2SIP).

4. Model Properties and Fast Approximations

In this section we present lower and upper bounds on the optimal solution for the DEBP problem. We then analyze worst case performance guarantees for approximations that are easy to implement in practice. The analysis of approximations is motivated by the fact that an instance of DEBP (in which $p^1_{b^1_u} = p^2_\varnothing = 1$ and the arrival process terminates after b^1_u items arrive in the first stage with certainty) corresponds to the extensible bin-packing (EBP) problem, which is known to be NP-hard, and thus DEBP is NP-hard as well (Denton et al. 2010). Proofs of the results in this section can be found in the online supplement.

We begin by presenting lower and upper bounds on the optimal value for DEBP, z^*. We first provide the following definition for the set of items in all batches. We let $\mathbb{I} = \bigcup_{t=1}^{T} I^t(\sigma^t)$ be the union of all item batches, and the expected value of the total size of all items across all scenarios is denoted by

$$\alpha = E_{\sigma,\omega}\left[\sum_{i\in\mathbb{I}} d_i(\omega^t)\right].$$

Proposition 1. *If the random item sizes, d_i, have a finite support $d_i \in [d_i^L, d_i^U]$ on the item size distribution, then the following are bounds on the optimal value of DEBP*:

$$c^f\left\lceil \frac{\alpha}{S(1+c^f/(Sc^v))}\right\rceil \le z^* \le \frac{2c^f\sum_{i\in\mathbb{I}} d_i^U}{S(1+c^f/(Sc^v))}.$$

Note that the lower and upper bounds require no further assumptions regarding the distribution of d_i beyond defining α and d_i^U. That is, the lower bound is based on the expected value of the total size of all items across all scenarios, and the upper bound is based on the maximum possible value of d_i in its finite support. In general, these bounds will be tighter in cases where the difference in α and $\sum_{i\in\mathbb{I}} d_i^U$ is small. These bounds can be used to fix decision variables prior to the solution process and to prune nodes in the branch-and-bound tree. Further, as we will discuss in more detail in Section 4.1, the bounds can be extended to evaluate the performance of approximations.

4.1. Performance of Approximation Methods

We begin by presenting some characteristics that all approximations for DEBP have in common. Next, we present results for a particular approximation that is commonly implemented in practice at OPCs, which we refer to as the *reserved bin approximation*. We then present results for an approximation called *List* that allocates items online such that each item is allocated to the least loaded bin.

Definition 1. The performance ratio (PR) of an approximation for a DEBP problem instance $\mathcal{I}$ is the ratio of the approximate objective value, $A(\mathcal{I})$, to the optimal objective value, $\text{Opt}(\mathcal{I})$.

Thus, bounds on a performance ratio PR^A for approximation A can be stated as

$$\min_{\mathcal{I}}\left\{\frac{A(\mathcal{I})}{\text{Opt}(\mathcal{I})}\right\} \le \text{PR}^A \le \max_{\mathcal{I}}\left\{\frac{A(\mathcal{I})}{\text{Opt}(\mathcal{I})}\right\}.$$

We begin by presenting the following lower bound on PR^A for any approximation, A, which is an extension of Theorem 1 in Speranza and Tuza (1999).

Proposition 2. *No approximation method for* (DEBP) *can have a worst case performance ratio smaller than* $1 + Sc^v/(6c^f)$.

Next, we consider the asymptotic performance of any *rational approximation*, which we define as any approximation where all nominal bin space is used prior to allocating an item to a bin by using extensible space exclusively. In this context the following proposition holds.

Proposition 3. *If there is a finite upper bound on the number of bins in* (DEBP), *then as* $\alpha \to \infty$, $\mathrm{PR}^{A} \to 1$, *for any rational approximation.*

4.1.1. Performance of the Reserved Bin Approximation. In the reserved bin approximation, denoted by A^r, bins are reserved in advance for batches of items. This is a common practice in environments such as OPCs because of the simplicity it affords managers in planning for an uncertain number of future procedures. However, preassigning items to bins is a restriction that may be suboptimal. An upper bound on the performance ratio for A^r can be obtained by considering the reserved bin problem as multiple separate extensible bin-packing problems. In the following theorem, let α^t denote the expected value of the sum of item sizes in the stage t batch.

Theorem 1. *As* $\alpha \to \infty$, *an upper bound on the performance ratio for the bin reservation approximation* PR^{A^r} *is* $1 + Sc^v/c^f$.

While the expected total item size will never approach infinity in the context of this application, Theorem 1 helps understand how these algorithms behave as the system becomes large. As is supported by the numerical results in Section 5, we see that the performance ratios of certain algorithms do tend toward 1 as the system becomes large.

4.1.2. List-Based Algorithms. Algorithm 1 describes *a-List*, an approximation for the deterministic formulation (1) in Section 3, that allocates each item sequentially to the least utilized bin and iteratively increases the number of bins to determine how many bins, m, to select. Note that in the case of a tie in the minimum load between bins, the bin with the lower index is chosen. Speranza and Tuza (1999) present a worst case performance analysis of online list-based algorithms for a problem similar to DEBP. In this section we extend the results in Speranza and Tuza (1999) by relaxing the assumption of a fixed and predetermined number of open bins and by differentiating the cost of opening a bin c^f and the cost of overtime c^v.

Algorithm 1 (*a-List* approximation)

Data: Set of items, $i = 1 \ldots n$; set of bins, $j = 1 \ldots m$; size of items, d_i; and costs c^v and c^f. L_j denotes the current load (total size of items) allocated to bin j.

Result: Number of bins and assignment of items to bins.

for $j = 1$ *to* m **do**
  $Cost = 0$ and $L_j = 0 \; \forall \, j$
  for $i = 1$ *to* n **do**
    $\hat{\jmath}$ = index of the component with the minimum value in $\{L_1, \ldots, L_j\}$
    $L_{\hat{\jmath}} = L_{\hat{\jmath}} + d_i$
  end
  for $k = 1$ *to* j **do**
    $Cost = Cost + c^v(\max(L_k - S, 0))$
  end
  $TotalCost_j = Cost + c^f j$
end
Return $\min_j\{TotalCost_j\}$
Return $\arg\min_j\{TotalCost_j\}$.

The algorithm starts with the trivial case of one bin and calculates the resulting cost. In subsequent iterations where there are more bins (up to a maximum of m), the next item is allocated to the bin with the least load currently assigned to it. After all items have been assigned, the fixed and overtime costs are calculated. The resulting solution is returned based on the number of bins that yields the smallest sum of fixed and overtime costs.

We let $\mathrm{Opt}(\mathcal{I})$ and $\mathrm{LS}(\mathcal{I})$ denote the optimal and resulting *a-List* values, respectively. We present two lemmas that can be used to derive bounds on the performance ratio for Algorithm 1. The first lemma is adapted from Lemma 1 in Speranza and Tuza (1999) to account for varying cost parameters (c^v and c^f). Here, q represents a known upper bound for all of the item sizes. Lemma 1 provides a bound on the performance of *a-List* when the optimal packing does not use any overtime. Here the overtime costs are reduced from one item's maximum size per bin to one-fourth of that cost by analyzing the bins that have overtime and those that have idle time in the resulting packing (see the proof in the online supplement for details).

Lemma 1. *Given a fixed number of bins* m, *if* $\mathrm{Opt}(\mathcal{I}) = mc^f$ *and* $d_i \le q$ *for all* i, *then* $\mathrm{LS}(\mathcal{I}) \le mc^f + mc^v q/4$.

The second lemma provides the means to extend results of Speranza and Tuza (1999) for the performance ratio for the special case of a fixed number of open bins to the more general algorithm *a-List* that varies the number of open bins.

Lemma 2. *Suppose there exists a function* $\hat{\mathcal{Q}}^1(\,)$, *such that* $\mathcal{Q}^1(x) \le \hat{\mathcal{Q}}^1(x) \le \beta\mathcal{Q}^1(x)$ *for all* x, *where* $\beta \ge 1$. *Let* $x^* = \arg\min\{\mathcal{Q}^1(x)\}$ *and* $\hat{x} = \arg\min\{\hat{\mathcal{Q}}^1(x)\}$. *Then* $\mathcal{Q}^1(x^*) \le \hat{\mathcal{Q}}^1(\hat{x}) \le \beta\mathcal{Q}^1(x^*)$.

Lemmas 1 and 2 can be used to derive the following bounds on the worst case performance ratio for the *a-List* approximation.

Theorem 2. *Given that* $d_i \le S \; \forall \, i$, *then* $1 + Sc^v/(6c^f) \le \mathrm{PR}^{a\text{-}List} \le 1 + Sc^v/(4c^f)$.

The difference in the performance ratio lower and upper bounds is $Sc^v/(12c^f)$. Note that this gap is small when the ratio Sc^v/c^f is small. When $c^f = Sc^v$, the worst case performance ratio is $\frac{5}{4}$, which is consistent with the special case analyzed in Speranza and Tuza (1999). In general, the following corollary characterizes the difference between the bounds relative to the cost ratio $Sc^v/(12c^f)$.

Corollary 1. *The error in* $\text{PR}^{a\text{-List}}$ *increases linearly with respect to* Sc^v/c^f.

This result is intuitive since as the overtime cost increases the problem approaches that of the bin-packing problem and the error grows as the problem becomes more difficult. For the numerical examples presented in Section 5, the error in Corollary 1 ranges between $\frac{1}{6}$ and $\frac{2}{3}$.

Next we describe a related algorithm for formulation (DEBP), in which the number of items and the item sizes may be uncertain. We assume the number of items is not known a priori and only the mean for the item's size is known. We refer to the algorithm as $a(\mu)$-*List*, to indicate items are allocated to the least utilized bin on the basis of mean size. The pseudocode for $a(\mu)$-*List* is described in Algorithm 2. The number of items are indexed by scenario index σ, where the number of items is denoted as $n(\sigma)$ and the size of the items by index ω, where the size of each item i is denoted as $d_i(\omega)$. Again, note that in the case of a tie in the minimum load between bins, the bin with the lower index is chosen. Theorem 3 presents lower and upper bounds for $a(\mu)$-*List*, for the case of finite support defined by parameter $\theta \in [1, \infty)$.

Algorithm 2 ($a(\mu)$-*List* approximation)

Data: Set of items, $i = 1 \ldots n$; set of bins, $j = 1 \ldots m$; size of items, $d_i(\omega)$; mean item sizes according to an item's type, $\bar{d}_i$; and costs c^v and c^f. L_j denotes the current load (total size of items) allocated to bin j. The number of items and item sizes are indexed by $\sigma = 1, \ldots, T$ and $\omega = 1, \ldots, K$, with probabilities $p(\sigma)$ and $p(\omega)$, respectively.

Result: Number of bins to open.

for $j = 1$ *to* m **do**
  for $\sigma = 1$ *to* T **do**
    for $\omega = 1$ *to* K **do**
      $L_j = 0$ and $C_j = \varnothing \; \forall \, j$
      for $i = 1$ *to* $n(\sigma)$ **do**
        $\hat{\jmath}$ = index of the component with the minimum value in $\{L_1, \ldots, L_j\}$
        $L_{\hat{\jmath}} = L_{\hat{\jmath}} + \bar{d}_i$
        Add i to C_j
      end
      $Cost = 0$
      for $k = 1$ *to* j **do**
        $Cost = Cost + c^v\left(\max\left(\sum_{i \in C_j} d_i(\omega), 0\right)\right)$
      end
      $TotalCost_j(\sigma, \omega) = Cost + c^f j$
    end
    $TotalCost_j(\sigma) = \frac{1}{K}\sum_{\omega} p(\omega) TotalCost_j(\sigma, \omega)$
  end
  $TotalCost_j = \frac{1}{T}\sum_{\sigma} p(\sigma) TotalCost_j(\sigma)$
end
Return $\min_j \{TotalCost_j\}$
Return $\arg\min_j \{TotalCost_j\}$.

As expected from Theorem 2, $\text{PR}^{a(\mu)\text{-List}} \to 1 + Sc^v/(4c^f)$ as $\theta \to 1$. Theorem 3 establishes the worst case performance of $a(\mu)$-*List*.

Theorem 3.

$$1 + \frac{Sc^v}{6c^f} \le \text{PR}^{a(\mu)\text{-List}} \le 1 + \frac{\theta Sc^v}{4c^f} + (\theta - 1)\frac{Sc^v}{c^f}$$

Algorithm 2 is similar to Algorithm 1 in that the next item is allocated to the bin with the least load currently assigned to it; however, Algorithm 2 uses the items' expected sizes ($\bar{d}_i$) in its allocation decision. To evaluate the expected costs of the bin and item assignments resulting from using the items' expected sizes, Algorithm 2 iterates over a finite number of sampled scenarios ($\sigma = 1, \ldots, T$ and $\omega = 1, \ldots, K$). The algorithm evaluates the expected costs for using one to m bins and returns the number of bins that results in the lowest expected costs.

In practice, the $a(\mu)$-*List* approximation would be applied in determining the appropriate number of procedure rooms to open and staff in advance of a scheduled day of procedures. Subsequently, as procedure requests are made and patients are scheduled (both in advance and as urgent add-ons) the *List*-based scheme provides an easy-to-implement means of assigning patients to particular procedure rooms. In Section 5, we provide estimates of the average case performance for typical instances arising in the context of OPCs.

5. Numerical Results

In this section we present numerical results based on practical instances of the model motivated by an OPC at Mayo Clinic in Rochester, Minnesota. In this context, the decisions are how many procedure rooms to plan for, how routine procedures should be allocated to procedure rooms, and how to allocate urgent procedures that arise on short notice. Neither the number of urgent procedures nor the duration for each procedure is known in advance.

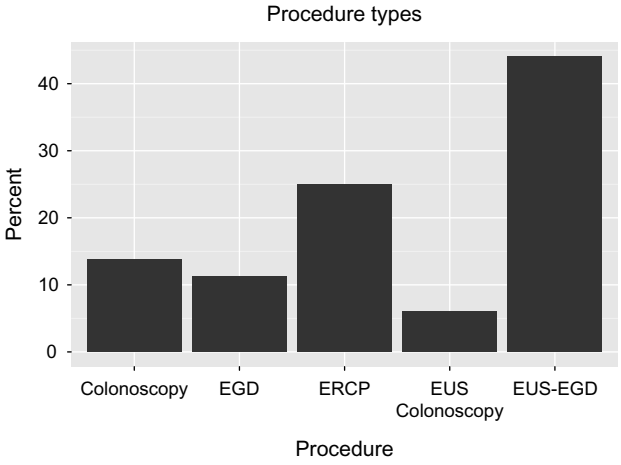
The remainder of this section is organized as follows. We begin by describing the parameter estimates used for generating problem instances for experiments. Three exact solution methods of the formulation discussed in Section 3.2 are compared based on their

computational performance for a range of problem instances. Results from these methods are used to evaluate performance of the $a(\mu)$-*List* approximation. Finally, some additional approximations are evaluated.

5.1. Design of Test Instances

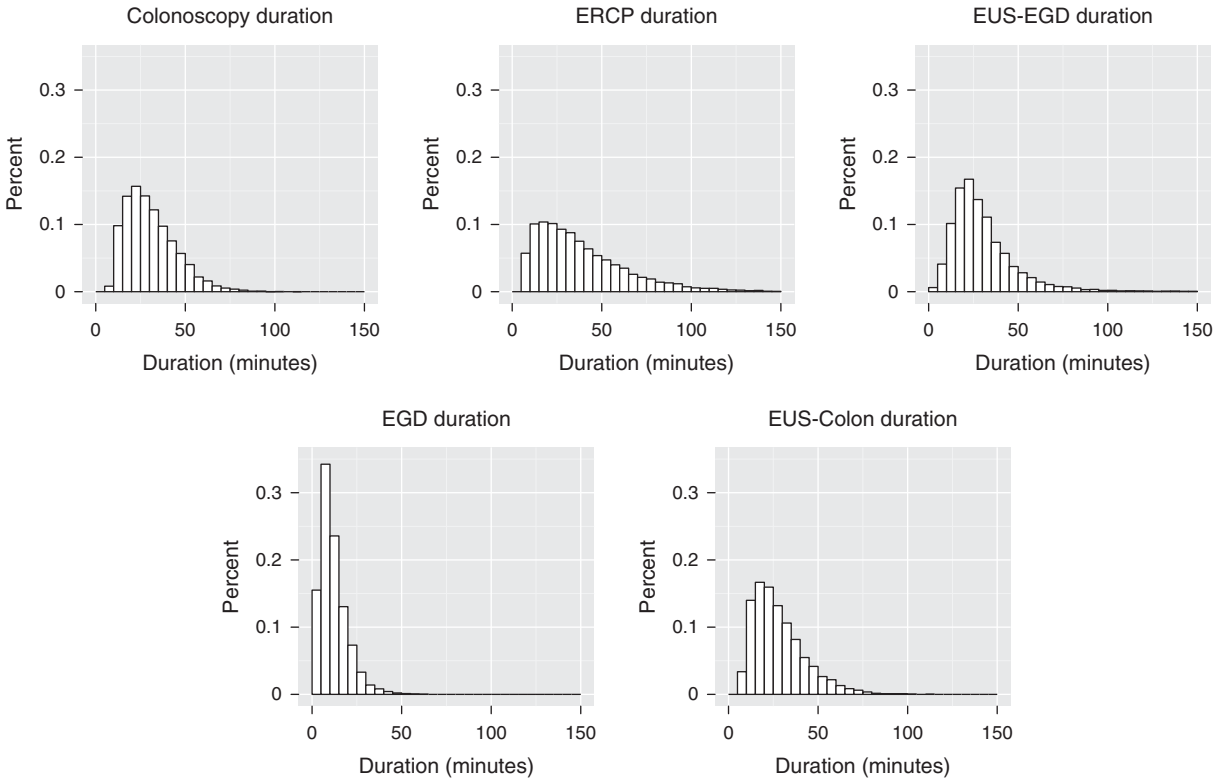
Test instances were generated based on data from the Division of Gastroenterology and Hepatology at Mayo Clinic. The number of routine procedures n was varied between $n = 10$ and $n = 30$. These values are representative of the demand volumes and day-to-day variation within practices of this type. The maximum number of urgent procedures b_u^2 was varied between $b_u^2 = 0$ and $b_u^2 = 10$ with each scenario's probability $p_{\sigma^2}^2$ being uniformly distributed. Procedure duration distributions and case mix were sampled from historical procedure data for the uncertain procedure duration parameters $d_i(\omega)$. Figure 1 presents the procedure type mix for five procedures: colonoscopy, esophagogastroduodenoscopy, endoscopic retrograde cholangiopancreatography, endoscopic ultrasound colonoscopy, and endoscopic ultrasound–esophagogastroduodenoscopy. This procedure mix is based on historical trends for the observed practice. Figure 2 presents the procedure duration density for the five procedure types. The ratio of variable overtime costs c^v to the fixed cost of opening a procedure room c^f was varied as the cost of opening a procedure room being equivalent to one (high), two (medium), and four (low) hours of overtime. While overtime cost estimates vary by practice, using

Figure 1. Procedure Duration Data for Five Types of Procedures for Model Parameters



Notes. EGD = Esophagogastroduodenoscopy; ERCP = endoscopic retrograde cholangiopancreatography; EUS = endoscopic ultrasounds; EUS-EGD = Endoscopic ultrasound–esophagogastroduodenoscopy.

Figure 2. Historical Data for Procedure Durations Based on Data from the Division of Gastroenterology and Hepatology at Mayo Clinic



Notes. EGD = Esophagogastroduodenoscopy; ERCP = endoscopic retrograde cholangiopancreatography; EUS = endoscopic ultrasound; EUS-EGD = Endoscopic ultrasound–esophagogastroduodenoscopy.

values of one, two, and four hours of overtime as equivalent to the cost of opening a new room provided for the evaluation of a range of realistic estimates. For the OPC in consideration, the high overtime cost is most relevant, but we include lower estimates to establish the relevance of the algorithms to other environments, such as hospital settings where absorbing overtime may be easier because of a larger staff and flexibility in shift assignments. The length of day S was set to 480 minutes. A total of 27 test instances resulted from varying the number of routine procedures, urgent procedures, and overtime costs. Data for the test instances are available in the online supplement.

All experiments for exact methods used CPLEX V12.2 on a Dell Linux server with 2 Quad-Core Intel Xeon E5420 2.5 GHz CPUs and 16 GB shared RAM.

5.2. Exact Solution Methods

The stochastic integer program based on the formulation 2SIP in Section 3.2 is difficult to solve because of binary decision variables in the first and second stages. Although the problem is naturally a three-stage decision process, we reformulate it as a 2SIP with all binary decisions in the first stage. The complete model formulation can be found in the online supplement. Three solution methods were implemented: (a) traditional branch-and-bound implemented on the extensive formulation, (b) the traditional L-shaped method (Van Slyke and Wets 1969), and (c) the integer L-shaped method (Laporte and Louveaux 1993). In the traditional L-shaped method, the recourse function is outer-linearized using optimality cuts based on the second stage dual solutions. In this method, the master problem is repeatedly solved with an optimality cut being added at each iteration. In the integer L-shaped method the recourse function is outer-linearized incrementally at integer feasible nodes in the branch-and-bound tree.

Each problem instance was evaluated using 10 random seeds with 1,000 random scenarios. A maximum of 15,000 CPU seconds was allowed for each instance. The computational results for the solution methods are presented in an aggregated summary in Table 1. For each of the three methods, the average and maximum optimality gaps across the 10 random seeds and 27 problem instances are reported. For problem instances that did not solve to an optimality tolerance of 1%, the optimality gap is calculated as a percentage of the lower bound at termination.

The results in Table 1 demonstrate that the DEBP is challenging. Branch-and-bound applied to the extensive formulation solved more problem instances to optimality than the L-shaped or integer L-shaped methods (66.67% versus 48.15% and 44.44%, respectively). However, the L-shaped and integer L-shaped methods were both competitive for harder instances, solving more instances to within 10% of optimality (88.89% and 88.89% versus 74.07%, respectively). Further, the L-shaped method provided the lowest average optimality gap (3.21%), and the integer L-shaped method provided the best worst case performance optimality gap (29.38%).

Table 1. Summary of the Performance of Each Solution Method Aggregated over All the Problem Instances

Method	% optimal (< 1%) (%)	% optimal (< 10%) (%)	Average gap (%)	Max gap (%)
Extensive form	66.67	74.07	20.80	288.05
L-shaped	48.15	88.89	3.21	37.46
Integer L-shaped	44.44	88.89	4.10	29.38

The value of the stochastic solution (VSS) was also analyzed. The VSS is defined as the relative improvement over the expected value of using the mean value problem solution. The best known solution at termination was used in the VSS calculation when the solution methods did not converge to an optimal solution. Thus, the VSS analyzed is a lower bound. In general, the VSS is sensitive to the overtime cost estimates. As overtime cost estimates increase and the ratio c^f/c^v decreases, the VSS increases to as high as 16.39%. However, the VSS for smaller instances is as low as 0%.

5.3. Evaluation of Approximation Methods

In this section we present results for the reserved bin approximation, and the *a-List* and $a(\mu)$*-List* approximations. Finally, we present an evaluation of other related approximations. We note that while computational results were presented for the exact solution methods, we do not present a similar analysis of the approximation methods, as the computational time was minimal. Instances of the *a-List* approximation, for example, were all solved in less than one second. We do, however, emphasize the computational advantage of the approximation methods over the exact solution methods.

Figure 3 presents the reserved bin-approximation performance ratios for varying cost values of overtime as the maximum number of (uniformly distributed) urgent patients increases (the number of routine patients is fixed at $n = 10$). The general trend illustrated in Figure 3 is a decreasing performance ratio as the expected procedure duration sum increases. The results in Figure 3 suggest that dedicating a procedure room for urgent add-on procedures performs better as the expected demand increases, especially at high overtime cost settings. However, when the number of possible add-on patients is small, reserving a procedure room for these patients can result in poor performance. Figure 3 highlights the potential poor performance of the reserved bin approximation in practice. While this approximation is easy to implement, the need for better performing approximations is clear.

Figure 3. Performance Ratio for Multiple Overtime Cost Estimates and a Range of Maximum (Uniformly Distributed) Urgent Patients

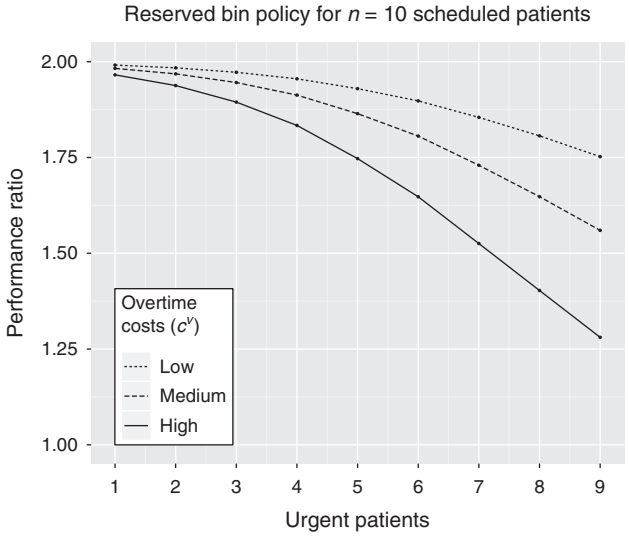
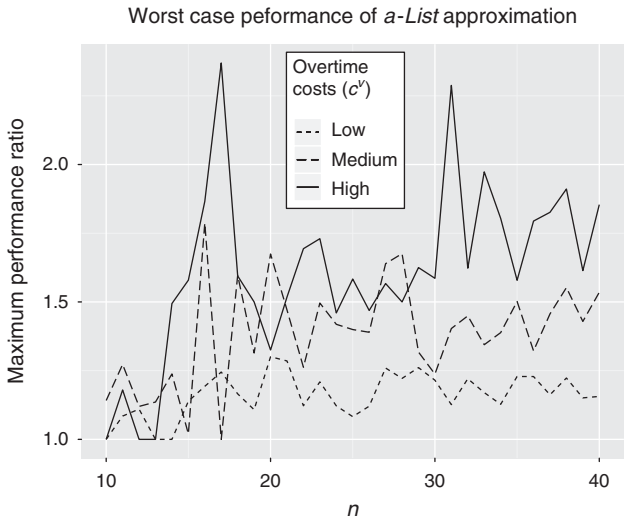
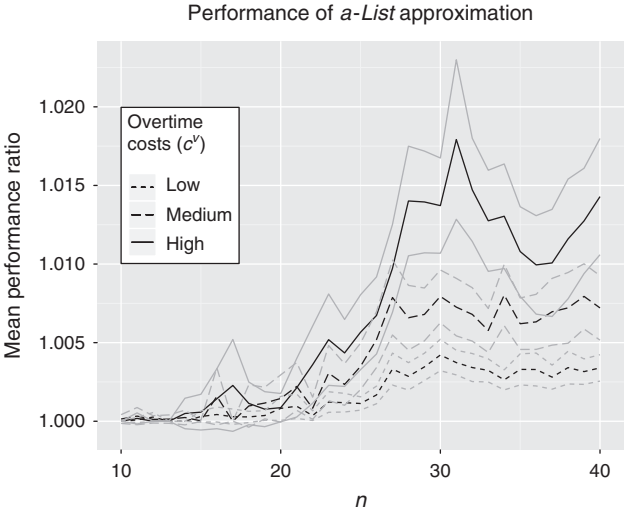


Figure 5. Maximum Performance Ratios Across 1,000 Randomly Generated Problem Instances for Low, Medium, and High Overtime Cost Estimates



The *a-List* approximation was evaluated by comparing the values of the resulting *a-List* solutions to the values of the optimal solutions to the optimization problem as formulated in (1). The performance ratio was calculated for 1,000 randomly generated problem instances ranging in size from 10 to 40. The optimal solution for each problem instance was obtained by solving the resulting extensible bin-packing problem defined in (1). The average performance ratio for the *a-List* approximation, along with a 95% confidence interval, is presented in Figure 4 for the low, medium, and high estimates of overtime costs c^v. The results show the *a-List* approximation performs very well with average performance ratios less than 1.025, and much lower than the theoretical worst case performance ratio of Theorem 2. The worst case performance from the numerical experiments is illustrated in Figure 5; the maximum performance ratio was 2.16, achieved when $c^v S/c^f = 8$. For lower $c^v S/c^f$, the worst case was typically less than 1.5.

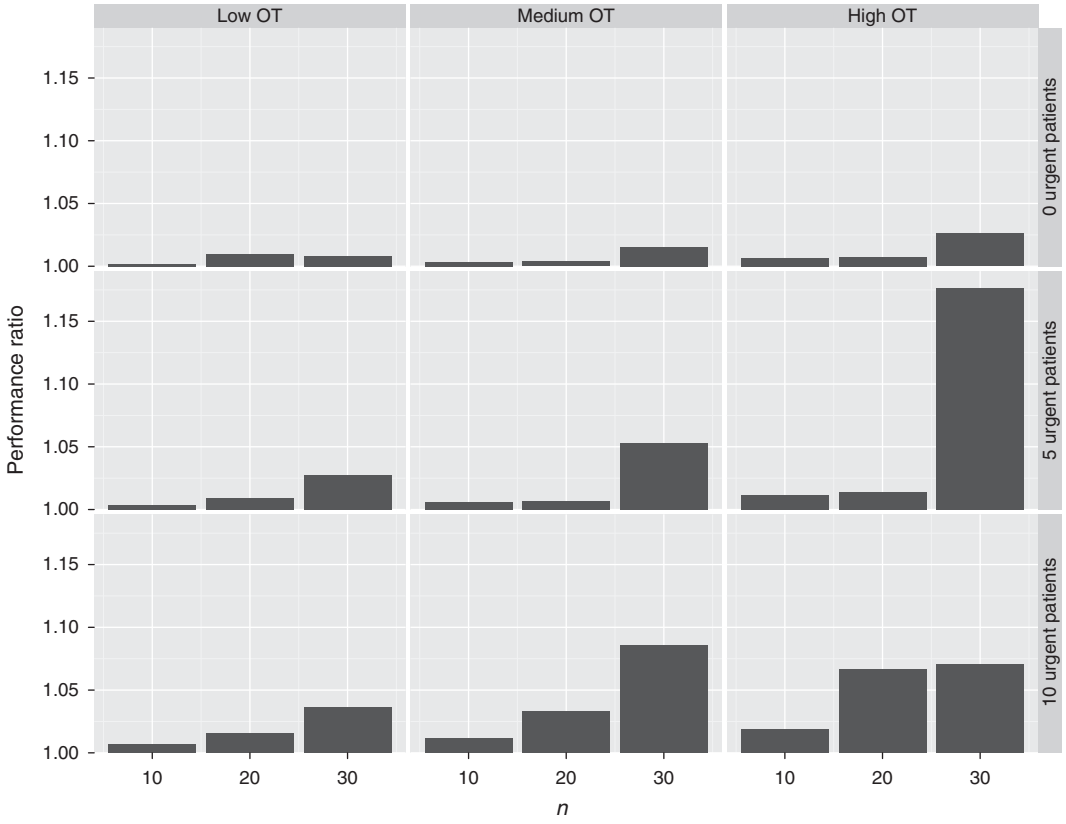
Figure 4. Average Performance Ratio Presented with 95% Confidence (in Grey) Intervals for Low, Medium, and High Overtime Cost Estimates



The performance ratio of the $a(\mu)$-*List* approximation was computed for the instances described in Section 5.2. The $a(\mu)$-*List* approximation results are summarized in Figure 6. They are based on the best solution obtained within 15,000 CPU seconds for the results in Table 1. The average performance ratio across all instances was less than 18%. In general, the performance ratio tends to increase as the size of the problem, urgent patient demand uncertainty, and overtime costs increase.

The reserved bin approximation was applied to a range of test instances where the number of scheduled patients (10, 20, 30), the number of possible urgent patients (0–30), and the overtime estimates (low, medium, high) were compared. Within each set of reserved bins (scheduled and urgent), *a-List* was applied. The number of bins for each patient group was identified by increasing the number of open bins until no overtime was used for the reserved bin and *a-List* approximations. The number of bins for each patient group that minimized the expected total costs was retained. Figure 7 presents the performance of applying *a-List* within the reserved bin approximation compared to applying *a-List* in a shared-bin context (i.e., not reserved for each patient group). Having demonstrated the strong performance of *a-List*, we use its solution value for comparison, as opposed to the

Figure 6. *a*(*μ*)-*List* Performance Ratio for Multiple Instances, Based on the Results from the Stochastic Program



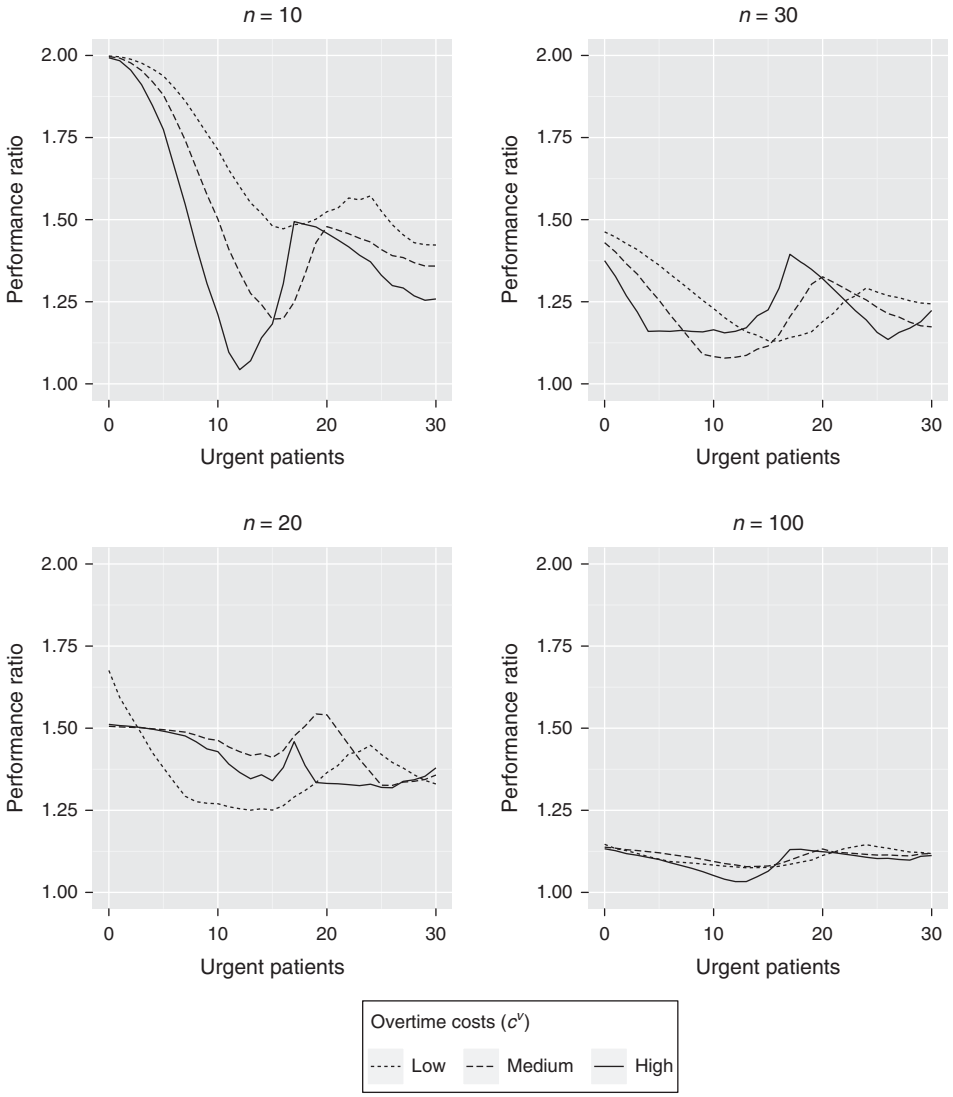
optimal solution to the stochastic program, to evaluate larger problem instances. The performance ratios in Figure 7 decrease as the number of urgent patients, and scheduled patients, increases. Sudden peaks are observed where the reserved bin approximation opens an additional bin and *a-List* is able to maintain more efficient use of share bins at the same demand level. Similar to Figure 3, the reserved bin approximation tends to perform better when the overtime cost is high. Further, consistent with Theorem 1, the performance of the reserved bin approximation improves as the total expected procedure duration becomes large.

5.4. Other Approximations

In addition to the reserved bin, *a-List*, and *a*(*μ*)-*List* approximations for which we developed theoretical worst case guarantees, we also tested some other allocation approximations motivated by related problems in the literature. Dexter and Traub (2002) evaluated two approximations for dynamically allocating elective surgeries to operating rooms, with the objective of maximizing operating room utilization. Their *Earliest Start Time* (EST) and *Latest Start Time* (LST) approximations allocate the next requested surgery to the procedure room with the earliest and latest start time available, respectively. The LST approximation requires that there be sufficient capacity remaining for the surgery to be allocated to the room with the latest start time available. In our implementation, the scheduled procedures are sorted by increasing expected duration and the approximations are then applied. In sorting the procedures prior to their allocations, these algorithms use additional information, which we previously assumed was unavailable in the general DEBP model. A variant of the EST approximation, *Earliest Start Time by Variance* (ESTV), was also evaluated. The ESTV approximation operates in the same manner, except the scheduled procedures are sorted by increasing variance instead of expected procedure duration. Finally, the number of procedure rooms to open was decided through an enumeration process where the number of open procedure rooms was fixed and the approximations were applied. The number of open procedure rooms was then incremented and the approximations were again applied. This process continued until there was a solution for each approximation where no overtime was used, and the lowest cost solution was selected.

Using the test instances described in Section 5.2, the performance results for the three approximations are summarized in Table 2. We note that the optimal solutions for the $n = 10$ instances were identified by each of the approximations. In the instances

Figure 7. Performance Ratio for the Reserved Bin Approximation, Where *a-List* Is Applied Within the Reserved Bins for Multiple Instances and Is Based on Comparing the Results from Applying *a-List*, Where Bins Are Shared Across Patient Groups



Note. The horizontal axis represents the maximum number of uniformly distributed urgent patients.

where an optimal solution was not identified with the solution methods, the lower bound at termination was used in the approximation's optimality gap calculation. Thus, the summary of optimality gaps reported in Table 2 represent an upper bound on their performance.

Table 2. Approximations' Performance Is Summarized and Aggregated over All Experiment Instances

Approximation	Within 1% of optimal (%)	Average gap (%)	Max gap (%)
LST	66.67	6.94	36.55
EST	83.33	1.61	7.75
ESTV	88.89	1.58	8.09

Contrary to previous research, the LST approximations did not perform as well as the EST approximation. This is likely because of the semidynamic nature of the problem where in allocating scheduled procedures, the LST approximation will fill a single procedure room before it considers utilizing another. The ESTV approximation provided the optimal solution for 88.89% of the instances that were solved to optimality. This is in contrast to the LST and EST approximations, which produced optimal solutions for 66.67% and 83.33% of the instances, respectively. While all three approximations provided average optimality gaps less than 10%, EST and ESTV resulted in lower averages (1.61% and 1.58% versus 6.94%, respectively). Similarly, the two EST approximations both limited worst

case performance to less than 10%, whereas the LST approximation produced a worst case performance of 36.55% from optimal. Both the EST and the ESTV approximations performed well, with small optimality gaps for larger problems with average optimality gaps between 0.69% and 6.99% for the $n = 30$ problem instances.

Although not presented in table format, the performance of the three additional heuristics followed the general performance trend of the proposed approximations in that when uncertainty about the number of urgent add-ons was high and when overtime costs were high, the heuristics had higher optimality gaps. For example, the average optimality gaps for the LST heuristic when $n = 20$ and there are 0 urgent add-ons were 0.0%, 6.76%, and 26.83% for low, medium, and high overtime cost values, respectively. However, when up to five urgent add-ons are possible, the average optimality gaps are 0.0%, 12.28%, and 24.04% for low, medium, and high overtime cost values, respectively. These general trends further support the insight that finding good solutions becomes more important when uncertainty increases as well as overtime costs.

6. Conclusions

In this article we presented a new model, referred to as DEBP. Special cases of the model that are especially relevant to healthcare delivery applications were discussed. Lower and upper bounds were developed to provide insights into the performance of approximations as well as improve the computational efficiency of the described solution methods. Well-known approximations motivated by related problems in the literature were also described and evaluated.

The results presented demonstrate that the DEBP is a computationally challenging problem with some practical instances not being solved to optimality within 15,000 CPU seconds using decomposition-based solution methods. The *a-List* and $a(\mu)$*-List* approximations were found to perform very well on average across a range of problem sizes and cost parameter estimates. While performance ratios for the *a-List* and $a(\mu)$*-List* approximations were higher for larger test instances with high overtime cost estimates, the average performance ratios were less than 1.02 and 1.2, respectively. In extreme cases where the overtime cost was high, the *a-List* approximation had a worst case performance ratio greater than 2.

As a practical matter, we draw attention to the fact that standard methods are often inadequate, as the online version of the problem corresponds to a multistage stochastic integer program that is unsolvable in many practical cases. Moreover, in those cases where it can be solved, the computation time is very high. This presents a challenge because the problem must be solved on a daily basis (Monday–Friday) for most surgery practices, since the mix of routine (first-stage) surgeries changes from day to day. The approximations proposed here have the added advantage of being intuitive and easier to understand for the practitioner in addition to being shown to perform well. Further, most clinical settings do not have access to, or resources to use, advanced solvers necessary to compute exact solutions. We believe the ease of understanding of the algorithms and the speed of and accuracy of them over existing methods makes the practical implications of this article important.

The performance of these approximations has implications on the use of probability distributions based on historical data for scheduling. For example, whereas the optimal solution to the stochastic programming formulation relies on using historical data to populate the model, the approximation methods require only the number to be scheduled and expected procedure durations. In this scheduling context, the value of using rich historical data appears to be low. We found that the reserved bin approximation, which is common in practice, can perform quite poorly. In contrast, other easy-to-implement approximations were shown, both theoretically and numerically, to perform well across a wide range of test instances. We derived tight bounds on the performance ratio for the case of deterministic item sizes and extended these bounds to the stochastic case.

The VSS was shown to be sensitive to the estimate of overtime costs as well as the size of the problem. In larger problem instances, the VSS was as high as 16% for problems where the overtime estimate was high and as low as 0% for problems where the overtime estimate was low. In many smaller problem instances, however, the solution to the semideterministic problem was optimal resulting in a VSS of 0%.

The performance of the approximations from the literature was generally robust with the optimal solution being identified for many instances, and small average and maximum optimality gaps otherwise. The EST by variance provided the optimal solution most often as well as the lowest average optimality gap. Rules of thumb can easily be gleaned from these approximations in a manner that is easy for managers to implement. For example, delaying the scheduling process may afford the collection of information (e.g., procedure mix) that could be used in sorting the procedures and constructing a better schedule.

The DEBP model presented does not include all the important aspects that may arise in OPC scheduling in practice. For example, certain procedures may require specific equipment or staff and need to be scheduled in particular procedure rooms. The constraints necessary for these settings can easily be added to the model formulation, and approximation methods could be adapted and analyzed. In addition, the current modeling framework assumes a linear cost of bin extensions, i.e., overtime. This may or may not be relevant to

a particular application context. Incorporating a convex function-based extension cost structure or constraining the amount by which bins can be extended is an important and open direction for future work. The setting of procedure start times was not addressed in the DEBP. However, any sequence reordering and appointment scheduling can be done without affecting the optimal solutions to DEBP. Further, while attendance behavior such as no-shows is not a concern in the particular context motivating this work, the model can easily be adapted for settings where nonattendance needs to be incorporated by assigning a procedures duration $d_i(\omega^t)$ a value of 0 with a certain probability. Finally, as we have shown, the approximation methods can yield poor solutions in rare instances; however, it is reasonable to assume those working most closely with the scheduling system will recognize these situations and adjust accordingly. Although our model does not address all conditions that may arise in practice, it provides a foundation for future study of the diverse range of operating conditions.

Acknowledgments

The authors are grateful for the comments and suggestions of three anonymous reviewers and the associate editor. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Adler M, Gibbons PB, Matias Y (2002) Scheduling space-sharing for Internet advertising. *J. Scheduling* 5(2):103–119.

Bentley JL, Johnson DS, Leighton FT, McGeoch CC, McGeoch LA (1984) Some unexpected expected behavior results for bin packing. DeMillo R, ed. *Proc. Sixteenth Annual ACM Sympos. Theory Comput.* (ACM, New York), 279–288.

Birge JR, Louveaux F (1997) *Introduction to Stochastic Programming* (Springer, New York).

Coffman EG, Lueker GS (2006) Approximation algorithms for extensible bin packing. *J. Scheduling* 9(1):63–69.

Coffman EG, Stolyar AL (2001) Bandwidth packing. *Algorithmica* 29(1):70–88.

Coffman EG Jr, Garey MR, Johnson DS (1983) Dynamic bin packing. *SIAM J. Comput.* 12(2):227–258.

Coffman EG, Garey MR, Johnson DS (1996) Approximation algorithms for bin packing: A survey. *Approximation Algorithms for NP-Hard Problems* (PWS Publishing Co., Boston), 46–93.

Dawande M, Kalagnanam J, Lee HS, Reddy C, Siegel S, Trumbo M (2004) The slab-design problem in the steel industry. *Interfaces* 34(3):215–225.

Dell'Olmo P, Kellerer H, Speranza MG, Tuza Z (1998) A 13/12 approximation algorithm for bin packing with extendable bins. *Inform. Processing Lett.* 65(5):229–233.

Denton BT, Miller AJ, Balasubramanian HJ, Huschka TR (2010) Optimal allocation of surgery blocks to operating rooms under uncertainty. *Oper. Res.* 58(4-part-1):802–816.

Dexter F, Traub RD (2002) How to schedule elective surgical cases into specific operating rooms to maximize the efficiency of use of operating room time. *Anesthesia Analgesia* 94(4):933–942.

Fekete SP, Schepers J (2001) New classes of fast lower bounds for bin packing problems. *Math. Programming* 91(1):11–31.

Fernandez de La Vega W, Lueker GS (1981) Bin packing can be solved within $1 + \varepsilon$ in linear time. *Combinatorica* 1(4):349–355.

Gamarnik D, Squillante MS (2005) Analysis of stochastic online bin packing processes. *Stochastic Models* 21(2–3):401–425.

Hans E, Wullink G, Van Houdenhoven M, Kazemier G (2008) Robust surgery loading. *Eur. J. Oper. Res.* 185(3):1038–1050.

Hoffmann U (1982) A class of simple stochastic online bin packing algorithms. *Comput.* 29(3):227–239.

Johnson DS (1974) Fast algorithms for bin packing. *J. Comput. System Sci.* 8(3):272–314.

Karmarkar N, Karp RM (1982) An efficient approximation scheme for the one-dimensional bin-packing problem. *Foundations Comput. Sci.*, 1982. *SFCS'08. 23rd Annual Sympos.* (IEEE, New York), 312–320.

Klein Haneveld WK, van der Vlerk MH (1999) Stochastic integer programming: General models and algorithms. *Ann. Oper. Res.* 85:39–57.

Laporte G, Louveaux FV (1993) The integer L-shaped method for stochastic integer programs with complete recourse. *Oper. Res. Lett.* 13(3):133–142.

Louveaux FV, Schultz R (2003) Stochastic integer programming. *Handbooks Oper. Res. Management Sci.* 10:213–266.

Martello S, Toth P (1990a) *Knapsack Problems: Algorithms and Computer Implementations* (John Wiley & Sons, Inc., Hoboken, NJ).

Martello S, Toth P (1990b) Lower bounds and reduction procedures for the bin packing problem. *Discrete Appl. Math.* 28(1):59–70.

Seiden SS (2002) On the online bin packing problem. *J. ACM* 49(5):640–671.

Shah D, Tsitsiklis JN (2008) Bin packing with queues. *J. Appl. Probab.* 45(4):922–939.

Speranza MG, Tuza Z (1999) On-line approximation algorithms for scheduling tasks on identical machines with extendable working time. *Ann. Oper. Res.* 86:491–506.

Ullman JD (1971) The performance of a memory allocation algorithm. Technical Report 100, Princeton University, Princeton, NJ.

Van Houdenhoven M, Van Oostrum JM, Hans EW, Wullink G, Kazemier G (2007) Improving operating room efficiency by applying bin-packing and portfolio techniques to surgical case scheduling. *Anesthesia Analgesia* 105(3):707–714.

Van Slyke RM, Wets R (1969) L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.* 17(4):638–663.

Yao ACC (1980) New algorithms for bin packing. *J. ACM* 27(2): 207–227.

Ye D, Zhang G (2004) On-line extensible bin packing with unequal bin sizes. Solis-Oba R, Jansen K, eds. *Approximation and Online Algorithms* (Springer, Berlin), 235–247.