Average-Case Performance of Rollout Algorithms for Knapsack Problems

Citation: Mastin, Andrew, and Patrick Jaillet. "Average-Case Performance of Rollout Algorithms for Knapsack Problems." Journal of Optimization Theory and Applications 165, no. 3 (July 26, 2014): 964–984.

As Published: http://dx.doi.org/10.1007/s10957-014-0603-x

Publisher: Springer-Verlag

Persistent URL: http://hdl.handle.net/1721.1/100430

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Average-Case Performance of Rollout Algorithms for Knapsack Problems

Andrew Mastin* Patrick Jaillet†

Communicated by Anita Schöbel

Abstract

Rollout algorithms have demonstrated excellent performance on a variety of dynamic and discrete optimization problems. Interpreted as an approximate dynamic programming algorithm, a rollout algorithm estimates the value-to-go at each decision stage by simulating future events while following a heuristic policy, referred to as the base policy. While in many cases rollout algorithms are guaranteed to perform as well as their base policies, there have been few theoretical results showing additional improvement in performance. In this paper, we perform a probabilistic analysis of the subset sum problem and 0-1 knapsack problem, giving theoretical evidence that rollout algorithms perform strictly better than their base policies. Using a stochastic model from the existing literature, we analyze two rollout methods that we refer to as the exhaustive rollout and consecutive rollout, both of which employ a simple greedy base policy. We prove that both methods yield a significant improvement in expected performance after a single iteration of the rollout algorithm, relative to the base policy.

Keywords Rollout algorithms, lookahead, knapsack problems, approximate dynamic programming

AMS Subject Classification 90C09, 90C39, 90C59

*Corresponding author. Department of Electrical Engineering and Computer Science, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139; mastin@mit.edu

†Department of Electrical Engineering and Computer Science , Laboratory for Information and Decision Systems, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA 02139; jaillet@mit.edu

1 Introduction

Rollout algorithms provide a natural and easily implemented approach for approximately solving many discrete and dynamic optimization problems. Their motivation comes from problems that can be solved using classical dynamic programming, but for which determining the value function (or value-to-go function) is computationally infeasible. The rollout technique estimates these values by simulating future events while following a simple greedy/heuristic policy, referred to as the base policy. In most cases, the rollout algorithm is ensured to perform as well as its base policy [1]. As shown by many computational studies, the performance is often much better than the base policy, and sometimes near optimal [2].

Theoretical results showing a strict improvement of rollout algorithms over base policies have been limited to average-case asymptotic bounds on the breakthrough problem [3] and a worst-case analysis of the 0-1 knapsack problem [4]. The latter work motivates a complementary study of rollout algorithms for knapsack-type problems from an average-case perspective, which we provide in this paper. Our goals are to give theoretical evidence for the utility of rollout algorithms, and to contribute to the knowledge of problem types and features that make rollout algorithms work well. We anticipate that our proof techniques may be helpful in achieving performance guarantees on similar problems.

We use a stochastic model directly from the literature that has been used to study a wide variety of greedy algorithms for the subset sum problem [5]. This model is extended in a natural manner for our analysis of the 0-1 knapsack problem [6, 7]. We analyze two rollout techniques that we refer to as the exhaustive rollout and the consecutive rollout, both of which use the same base policy. During each iteration of the exhaustive rollout, the algorithm decides which one of the available items should be added to the knapsack. The latter algorithm sequentially processes the items, and at each iteration decides if the current item should be added to the knapsack. The base policy is a simple greedy algorithm that adds items until an infeasible item is encountered.

For both techniques, we derive bounds showing that the expected performance of the rollout algorithms is strictly better than the performance obtained by only using the base policy. For the subset sum problem, this is demonstrated by measuring the gap between the total value of packed items and capacity. For the 0-1 knapsack problem, the difference between total profits of the rollout algorithm and base policy is measured. The bounds are valid after only a single iteration of the rollout algorithm and hold for additional iterations.

The organization of the paper is as follows. In the remainder of this section we review related work, and we introduce our notation in Section 2. Section 3 describes the stochastic models in detail and derives important properties of the blind greedy algorithm, which is the algorithm that we use for a base policy.

Results for the exhaustive rollout and consecutive rollout are shown in Section 4 and Section 5, respectively; these sections contain the most important proofs used in our analysis. Conclusions are given in Section 6. Omitted proofs and a section with evaluations of integrals are provided in the supplementary material.

Rollout algorithms were introduced by Tesauro and Galperin [8] as online Monte-Carlo search techniques for computer backgammon. The application to combinatorial optimization was formalized by Bertsekas et al. [1]. They gave conditions under which the rollout algorithm is guaranteed to perform as well as its base policy, namely if the algorithm is sequentially consistent or sequentially improving, and presented computational results on a two-stage maintenance and repair problem. The application of rollout algorithms to approximate stochastic dynamic programs was provided by Bertsekas and Castañon [2], where they showed extensive computational results on variations of the quiz problem. Rollout algorithms have since shown strong computational results on a variety of problems including vehicle routing (Secomandi [9]), fault detection (Tu and Pattipati [10]), and sensor scheduling (Li et al. [11]).

Beyond simple bounds derived from base policies, the only theoretical results given explicitly for rollout algorithms are average-case results for the breakthrough problem (Bertsekas [3]) and worst-case results for the 0-1 knapsack problem (Bertazzi [4]). In the breakthrough problem, the objective is to find a valid path through a directed binary tree, where some edges are blocked. If the free (non-blocked) edges occur with a given probability, independent of other edges, a rollout algorithm has a larger probability of finding a free path in comparison to a greedy algorithm [3]. Performance bounds for the 0-1 knapsack problem were recently shown by Bertazzi [4], who analyzed the rollout approach with variations of the decreasing density greedy (DDG) algorithm as a base policy. The DDG algorithm takes the best of two solutions: the one obtained by adding items in order of non-increasing profit to weight ratio, as long as they fit, and the solution resulting from adding only the item with highest profit. He demonstrated that from a worst-case perspective, running the first iteration of a rollout algorithm (specifically, what we will refer to as the exhaustive rollout algorithm) improves the approximation guarantee from $1/2$ (bound provided by the base policy) to $2/3$.

While results on approximating the subset sum problem initially focused on worst-case analysis, the performance of various greedy algorithms on random instances was analyzed from a computational perspective by Martello and Toth [12]. An early probabilistic analysis of the subset sum problem was given by d'Atri and Puech [13]. Using a discrete version of the model used in our paper, they analyzed the expected performance of greedy algorithms with and without sorting. They showed an exact probability distribution for the gap obtained by the algorithms, and gave asymptotic expressions for the probability of obtaining a non-zero

gap. These results were refined by Pferschy [14], who gave precise bounds on expected gap values for greedy algorithms.

A very extensive analysis of greedy algorithms for the subset sum problem was given by Borgwardt and Tremel [5]. They introduced the continuous model that we use in this paper and derived probability distributions of gaps for a variety of greedy algorithms. In particular, they showed performance bounds for a variety of prolongations of a greedy algorithm, where a different strategy is used on the remaining items after the greedy policy is no longer feasible. They also analyzed cases where items are ordered by size prior to use of the greedy algorithms.

In the area of probabilistic 0-1 knapsack problems, Szkatula and Libura [15] investigated the behavior of greedy algorithms, similar to the blind greedy algorithm used in our paper, for the 0-1 knapsack problem with fixed capacity. They found recurrence equations describing the weight of the knapsack after each iteration and solved the equations for the case of uniform weights. In later work [16], they studied asymptotic properties of greedy algorithms, including conditions for the knapsack to be filled almost surely as the number of items approaches infinity.

There has been some work on asymptotic properties of the decreasing density greedy algorithm (DDG) for probabilistic 0-1 knapsack problems. Diubin and Korbut [17] showed properties of the asymptotical tolerance of the algorithm, which characterizes the deviation of the solution from the optimal value. Similarly, Calvin and Leung [18] showed convergence in distribution between the value obtained by the DDG algorithm and the value of the knapsack linear relaxation. Finally, it is worth noting other probabilistic studies of knapsack problems including the work of Dean et al. [19] on adaptive polices, Lueker's [20] analysis of online algorithms, and the expected polynomial time algorithm of Beier and Vöcking [21].

2 Notation

Before we describe the model and algorithms, we summarize our notation. Since we must keep track of ordering in our analysis, we use sequences in place of sets and slightly abuse notation to perform set operations on sequences. These operations will mainly involve index sequences, and our index sequences will always contain unique elements. Sequences will be denoted by bold letters. If we wish for $\boldsymbol{S}$ to be the increasing sequence of integers ranging from 2 to 5, we write $\boldsymbol{S} = \langle 2, 3, 4, 5\rangle$. We then have $2 \in \boldsymbol{S}$, while $1 \notin \boldsymbol{S}$. We also say that $\langle 2, 5\rangle \subseteq \boldsymbol{S}$ and $\boldsymbol{S} \setminus \langle 3\rangle = \langle 2, 4, 5\rangle$. The concatenation of sequence $\boldsymbol{S}$ with sequence $\boldsymbol{R}$ is denoted by $\boldsymbol{S} : \boldsymbol{R}$. If $\boldsymbol{R} = \langle 1, 7\rangle$, then $\boldsymbol{S} : \boldsymbol{R} = \langle 2, 3, 4, 5, 1, 7\rangle$. A sequence is indexed by an index sequence if the

index sequence is shown in the subscript. Thus, $\boldsymbol{a_S}$ indicates the sequence $\langle a_2, a_3, a_4, a_5\rangle$. For a sequence to satisfy equality with another sequence, equality must be satisfied element by element, according to the order of the sequence. We use the notation $\boldsymbol{S^i}$ to denote the sequence $\boldsymbol{S}$ with item i moved to the front of the sequence: $\boldsymbol{S^3} = \langle 3, 2, 4, 5\rangle$.

The notation $\mathbb{P}(\cdot)$ indicates probability, and $\mathbb{E}[\cdot]$ indicates expectation. We define $\overline{\mathbb{E}}[\cdot] := 1 - \mathbb{E}[\cdot]$. For random variables, we will use capital letters to denote the random variable (or sequence) and lowercase letters to denote specific instances of the random variable (or sequence). The probability density function for a random variable X is denoted by $f_X(x)$. For random variables X and Y, we use $f_{X|Y}(x|y)$ to denote the conditional density of X, given the event $Y = y$. When multiple variables are involved, all variables on the left side of the vertical bar are conditioned on all variables on the right side of vertical bar. The expression $f_{X,Y|Z,W}(x, y|z, w)$ should be interpreted as $f_{(X,Y)|(Z,W)}((x, y)|(z, w))$ and not $f_{X,(Y|Z),W}(x, (y|z), w)$, for example. Events are denoted by the calligraphic font, such as $\mathcal{A}$, and the disjunction of two events is shown by the symbol $\vee$. We often write conditional probabilities of the form $\mathbb{P}(\cdot|X = x, Y = y, \mathcal{A})$ as $\mathbb{P}(\cdot|x, y, \mathcal{A})$. The notation $\mathcal{U}[a, b]$ indicates the density of a uniform random variable on interval $[a, b]$. The indicator function is denoted by $\mathbb{1}(\cdot)$, and the positive part of an expression is denoted by $(\cdot)_+$. Finally, we use the symbol $\leftarrow$ for assignment, and $O(\cdot)$ asymptotic growth. A summary of symbols that we use throughout the text is given as follows.

2.1 List of Symbols

$\leftarrow$	assignment
$\vee$	disjunction
$(\cdot)_+$	positive part
A	weight of the last packed item, $A := W_{K-1}$
B	knapsack capacity
$\mathcal{C}_j$	event that item j is the critical item
$\overline{\mathcal{C}_1}$	event that the first item is not critical
$\overline{\mathcal{C}_{1n}}$	event that neither the first nor the last item are critical
$\mathcal{D}_j$	event that item j is the drop critical item
$\mathcal{E}$	event that $g + w_{K-1} < 1$
G	gap given by the Blind-Greedy algorithm

$H(\cdot)$	harmonic number
$\mathbb{1}(\cdot)$	indicator function
$\boldsymbol{I}$	index sequence, $\boldsymbol{I} := \langle 1, 2, \ldots, n \rangle$
K	critical item index (first item that is infeasible for BLIND-GREEDY to pack)
L_1	drop critical item index (first infeasible item for BLIND-GREEDY when first item skipped)
$L_j,\ j \geq 2$	insertion critical item index (first infeasible item for BLIND-GREEDY when j inserted first)
M	number of remaining items, $M := n - K$
n	number of items
$O(\cdot)$	asymptotic growth
P_i	profit of item i (for the 0-1 knapsack problem)
$\boldsymbol{P_S}$	sequence of item profits, indexed by sequence $\boldsymbol{S}$
Q	profit of last packed item, $Q := P_{K-1}$
$\mathcal{U}[x, y]$:	uniform distribution with support $[x, y]$
V_1	drop gap (gap when the first item is skipped)
$V_j,\ j \geq 2$	insertion gap (gap when item j is inserted first)
V_j^u	upper bound on V_j
V_*	minimum gap; gap obtained after the first iteration of the rollout algorithm
V_*^u	upper bound on V_*
W_i	weight of item i
$\boldsymbol{W_S}$	sequence of item weights, indexed by sequence $\boldsymbol{S}$
$Z_j,\ j \geq 2$	insertion gain (gain when item j is inserted first)
Z_j^l	lower bound on Z_j
Z_*	maximum gain; gain obtained after the first iteration of the rollout algorithm
Z_*^l	lower bound on Z_*

3 Stochastic Model and Blind Greedy Algorithm

In the 0-1 knapsack problem, we are given a sequence of items $\boldsymbol{I} = \langle 1, 2, \ldots, n \rangle$, where each item $i \in \boldsymbol{I}$ has a weight $w_i \in \mathbb{R}_+$ and profit $p_i \in \mathbb{R}_+$. Given a knapsack with capacity $b \in \mathbb{R}_+$, the goal is to select a subset of items with maximum total profit such that the total weight does not exceed the capacity. This is given

by the following integer linear program.

$$\max \sum_{i=1}^{n} p_i x_i \qquad \text{s.t. } \sum_{i=1}^{n} w_i x_i \leq b, \quad x_i \in \{0,1\}, \quad i = 1, \ldots, n. \tag{1}$$

The subset sum problem refers to the 0-1 knapsack problem, where $p_i = w_i$ for all $i \in \boldsymbol{I}$.

We use the stochastic subset sum model given in [5] and a variation of this model for the 0-1 knapsack problem. In their subset sum model, for a specified number of items n, item weights W_i and the capacity B are drawn independently from the following distributions:

$$W_i \sim \mathcal{U}[0,1], \quad i = 1, \ldots, n, \qquad B \sim \mathcal{U}[0,n]. \tag{2}$$

Our stochastic knapsack model simply assigns item profits that are independently and uniformly distributed,

$$P_i \sim \mathcal{U}[0,1], \quad i = 1, \ldots, n. \tag{3}$$

These values are also independent with respect to the weights and capacity.

For evaluating performance, we only consider cases where $\sum_{i=1}^{n} W_i > B$. In all other cases, any algorithm that tries adding all items is optimal. Since it is difficult to understand the stochastic nature of optimal solutions, we use $\mathbb{E}[B - \sum_{i \in \boldsymbol{S}} W_i | \sum_{i=1}^{n} W_i > B]$ as a performance metric for the subset sum problem, where $\boldsymbol{S}$ is the sequence of items selected by the algorithm of interest. This is the same metric used in [5], where they note with a simple symmetry argument that for all values of n,

$$\mathbb{P}\left(\sum_{i=1}^{n} W_i > B\right) = \frac{1}{2}. \tag{4}$$

For the 0-1 knapsack problem, we directly measure the difference between the rollout algorithm profit and the profit given by the base policy, which we refer to as the gain of the rollout algorithm.

For both the subset sum problem and the 0-1 knapsack problem, we use the Blind-Greedy algorithm, shown in Algorithm 1, as a base policy. The algorithm simply adds items (without sorting) until it encounters an item that exceeds the remaining capacity, then stops. Throughout the paper, we will sometimes refer to Blind-Greedy simply as the greedy algorithm.

Blind-Greedy may seem inferior to a greedy algorithm that first sorts the items by weight or profit to

Algorithm 1 BLIND-GREEDY

Input: Item sequence $\boldsymbol{I} = \langle 1, \ldots, n \rangle$ with profit sequence $\boldsymbol{p_I}$ and weight sequence $\boldsymbol{w_I}$, capacity b.
Output: Feasible solution sequence $\boldsymbol{S}$, value U.

1. Initialize solution sequence $\boldsymbol{S} \leftarrow \langle\rangle$, remaining capacity $\overline{b} \leftarrow b$, current item $i \leftarrow 1$, and value $U \leftarrow 0$.
2. Iteration i: if the weight of item i exceeds remaining capacity ($w_i > \overline{b}$), then stop. Otherwise,
   (a) Add item i to solution sequence $\boldsymbol{S} \leftarrow \boldsymbol{S} : \langle i \rangle$.
   (b) Update remaining capacity $\overline{b} \leftarrow \overline{b} - w_i$ and value $U \leftarrow U + p_i$.
   (c) If the final item has been reached ($i = n$), stop. Otherwise, move to the next item $i \leftarrow i + 1$ and goto 2.

weight ratio, and then adds them in non-decreasing value. Surprisingly, for the subset sum problem, it was shown in [5] that the algorithm that adds items in order of non-decreasing weight (referred to as GREEDY 1S) performs equivalently to BLIND-GREEDY. Of course, we cannot say the same about the 0-1 knapsack problem. A greedy algorithm that adds items in decreasing profit to weight ratio is likely to perform much better. Applying our analysis to a sorted greedy algorithm requires work beyond the scope of this paper.

In analyzing BLIND-GREEDY, we refer to the index of the first item that is infeasible as the critical item. Let K be the random variable for the index of the critical item, where $K = 0$ indicates that there is no critical item (meaning $\sum_{i=1}^{n} W_i \leq B$). Equivalently, assuming $\sum_{i=1}^{n} W_i > B$, the critical item index satisfies

$$\sum_{i=1}^{K-1} W_i \leq B < \sum_{i=1}^{K} W_i. \tag{5}$$

We will refer to items with indices $i < K$ as packed items. We then define the gap of BLIND-GREEDY as

$$G := B - \sum_{i=1}^{K-1} W_i, \tag{6}$$

for $K > 0$. The gap is relevant to both the subset sum problem and the 0-1 knapsack problem. We use it as a performance measure only on the subset sum problem, however, since the capacity B is a natural upper bound on the optimal value, and the gap is thus an upper bound on the difference between a given solution value and the optimal value. There is no analogous upper bound on optimal value for the 0-1 knapsack problem, so for this problem we define the gain of the rollout algorithm as

$$Z := \sum_{i \in \boldsymbol{R}} P_i - \sum_{i=1}^{K-1} P_i, \tag{7}$$

where $\boldsymbol{R}$ is the sequence of items selected by the rollout algorithm. A central result of [5] is the following, which does not depend on the number of items n.

Theorem 3.1 (Borgwardt and Tremel [5])**.** Independently of the critical item index $K > 0$, the probability distribution of the gap obtained by Blind-Greedy satisfies

$$\mathbb{P}\left(G \leq g \left| \sum_{i=1}^{n} W_i > B \right.\right) = 2g - g^2, \quad 0 \leq g \leq 1, \qquad \mathbb{E}\left[G \left| \sum_{i=1}^{n} W_i > B \right.\right] = \frac{1}{3}. \tag{8}$$

Many studies measure performance using an approximation ratio (bounding the ratio of the value obtained by some algorithm to the optimal value) [6, 4]. While this metric is generally not tractable under the stochastic model, we can observe a simple lower bound on the ratio of expectations of the value given by Blind-Greedy to the optimal value, for the subset sum problem. A natural upper bound on the optimal solution is B, and the solution value given by Blind-Greedy is equal to $B - G$. Thus, by Theorem 3.1 and linearity of expectations, the ratio of expected values is at least $\frac{\mathbb{E}[B-G]}{\mathbb{E}[B]} = 1 - \frac{2}{3n}$. For $n \geq 2$, this value is at least $\frac{2}{3}$, which is the best worst-case approximation ratio derived in [4]. A similar comparison for the 0-1 knapsack problem is not possible because there is no simple bound on the expected optimal solution value.

We describe some important properties of the Blind-Greedy solution that will be used in later sections and that provide a proof of Theorem 3.1. For the proofs in this section, as well as other sections, it is helpful to visualize the Blind-Greedy solution sequence on the non-negative real line, as shown in Figure 1.

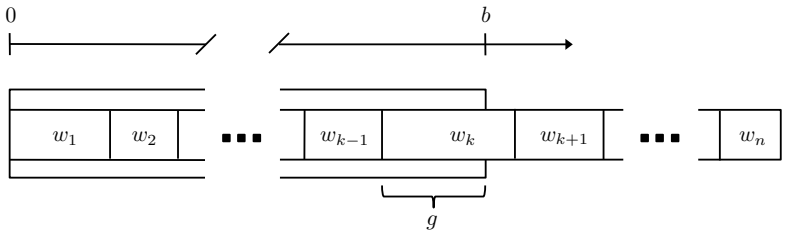


Figure 1: Sequence given by Blind-Greedy on the non-negative real line, where $G = g$, $B = b$, and $\boldsymbol{W_S} = \boldsymbol{w_s}$. Each item $\ell = 1, \ldots, n$ occupies the interval $\left[\sum_{i=1}^{\ell-1} w_i, \sum_{i=1}^{\ell} w_i\right[$ and the knapsack is given on the interval $[0, b]$. The gap is the difference between the capacity and the total weight of the packed items.

Previous work on the stochastic model has demonstrated that the critical item index is uniformly distributed on $\{1, 2, \ldots, n\}$ for cases of interest (i.e., $\sum_{i=1}^{n} W_i > B$) [5]. In addition to this property, we show that the probability that a given item is critical is independent of weights of all other items. In other sections

we follow the convention of associating the index k with the random variable K. The index ℓ is used in this section to make the proofs clearer.

Lemma 3.1. For each item $\ell = 1, \ldots, n$, for all subsequences of items $\boldsymbol{S} \subseteq \boldsymbol{I} \setminus \langle \ell \rangle$ and all weights $\boldsymbol{w_S}$, the probability that item ℓ is critical is

$$\mathbb{P}(K = \ell | \boldsymbol{W_S} = \boldsymbol{w_S}) = \frac{1}{2n}. \tag{9}$$

Proof. Assume that we are given the weights of all items $\boldsymbol{W_I} = \boldsymbol{w_I}$. We can divide the interval $[0, n]$ into $n + 1$ segments as a function of item weights as shown in Figure 1, so that the ℓth segment occupies the interval $\left[\sum_{i=1}^{\ell-1} w_i, \sum_{i=1}^{\ell} w_i\right[$ for $\ell = 1, \ldots, n$ and the last segment is on $[\sum_{i=1}^{n} w_i, n]$. The probability that item ℓ is critical is the probability that B intersects the ℓth segment. Since B is distributed uniformly over the interval $[0, n]$, we have

$$\mathbb{P}(K = \ell | \boldsymbol{W_I} = \boldsymbol{w_I}) = \frac{w_\ell}{n}, \tag{10}$$

showing that this event only depends on w_ℓ. Integrating over the uniform density of w_ℓ gives the result. □

An important property of this stochastic model, which is key for the rest of our development, is that conditioning on the critical item index only changes the weight distribution of the critical item; all other item weights remain independently distributed on $\mathcal{U}[0, 1]$.

Lemma 3.2. For any critical item $K > 0$ and any subsequence of items $\boldsymbol{S} \subseteq \boldsymbol{I} \setminus \langle K \rangle$, the weights $\boldsymbol{W_S}$ are independently distributed on $\mathcal{U}[0, 1]$, and W_K independently follows the distribution

$$f_{W_K}(w_K) = 2w_K, \quad 0 \leq w_K \leq 1. \tag{11}$$

Proof. For any item $\ell = 1, \ldots, n$, consider the subsequence of items $\boldsymbol{S} = \boldsymbol{I} \setminus \langle \ell \rangle$. Using Bayes' Theorem, the conditional joint density for $\boldsymbol{W_S}$ is given by

$$\begin{aligned} f_{\boldsymbol{W_S}, W_\ell | K}(\boldsymbol{w_S}, w_\ell | \ell) &= \frac{\mathbb{P}(K = \ell | \boldsymbol{W_S} = \boldsymbol{w_S}, W_\ell = w_\ell)}{\mathbb{P}(K = \ell)} f_{\boldsymbol{W_S}}(\boldsymbol{w_S}) f_{W_\ell}(w_\ell) \\ &= \frac{w_\ell / n}{1/(2n)} f_{\boldsymbol{W_S}}(\boldsymbol{w_S}) \\ &= 2w_\ell f_{\boldsymbol{W_S}}(\boldsymbol{w_S}), \quad 0 \leq w_\ell \leq 1, \end{aligned} \tag{12}$$

where we have used the results of Lemma 3.1. This holds for the $K = \ell$ and $\ell = 1, \ldots, n$, so we replace the index ℓ with K in the expression. □

We can now analyze the gap obtained by BLIND-GREEDY for $K > 0$. This gives the following lemma and a proof of Theorem 3.1.

Lemma 3.3. Independent of the critical item index $K > 0$, the conditional distribution of the gap obtained by BLIND-GREEDY satisfies

$$f_{G|W_K}(g|w_K) = \mathcal{U}[0, w_K]. \tag{13}$$

Proof. For any $\ell = 1, \ldots, n$ and any $\boldsymbol{W_I} = \boldsymbol{w_I}$, the posterior distribution of B, given the event $K = \ell$, satisfies

$$f_{B|\boldsymbol{W_I},K}(b|\boldsymbol{w_I}, \ell) = \mathcal{U}\left[\sum_{i=1}^{\ell-1} w_i, \sum_{i=1}^{\ell} w_i\right], \tag{14}$$

since we have a uniform random variable B that is conditionally contained in a given interval. Now, using the definition of G in (6),

$$f_{G|W_\ell,K}(g|w_\ell, \ell) = \mathcal{U}[0, w_\ell]. \tag{15}$$

□

Proof of Theorem 3.1. Using Lemma 3.3 and the distribution for W_K from Lemma 3.2, we have for $K > 0$,

$$f_G(g) = \int_0^1 f_{G|W_K}(g|w_K) f_{W_K}(w_K) \mathrm{d}w_K = \int_g^1 \frac{1}{w_K} 2w_K \mathrm{d}w_K = 2 - 2g, \tag{16}$$

where we have used that $G \leq W_K$ with probability one. This serves as a simpler proof of the theorem from [5]; their proof is likely more conducive to their analysis. □

Finally, we need a modified version of Lemma 3.2, which will be used in the subsequent sections.

Lemma 3.4. Given any critical item $K > 0$, gap $G = g$, and any subsequence of items $\boldsymbol{S} \subseteq \boldsymbol{I} \setminus \langle K \rangle$, the weights $\boldsymbol{W_S}$ are independently distributed on $\mathcal{U}[0, 1]$, and W_K is independently distributed on $\mathcal{U}[g, 1]$.

Proof. Fix $K = \ell$ for any $\ell > 0$. The statement of the lemma is equivalent to the expression

$$f_{\boldsymbol{W_S},W_\ell|G,K}(\boldsymbol{w_S}, w_\ell|g, \ell) = \frac{1}{1-g} f_{\boldsymbol{W_S}}(\boldsymbol{w_S}), \quad g \leq w_\ell \leq 1. \tag{17}$$

Note that

$$f_{G|\boldsymbol{W_S},W_\ell,K}(g|\boldsymbol{w_S}, w_\ell, \ell) = \mathcal{U}[0, w_\ell], \tag{18}$$

which can be shown by the same argument for Lemma 3.3. Then,

$$\begin{aligned} f_{\boldsymbol{W_S},W_\ell|G,K}(\boldsymbol{w_S}, w_\ell|g,\ell) &= \frac{f_{G|\boldsymbol{W_S},W_\ell,K}(g|\boldsymbol{w_S}, w_\ell, \ell) f_{\boldsymbol{W_S},W_\ell|K}(\boldsymbol{w_S}, w_\ell|\ell)}{f_{G|K}(g|\ell)} \\ &= \frac{f_{G|\boldsymbol{W_S},W_\ell,K}(g|\boldsymbol{w_S}, w_\ell, \ell) f_{\boldsymbol{W_S}}(\boldsymbol{w_S}) f_{W_\ell|K}(w_\ell|\ell)}{f_{G|K}(g|\ell)} \\ &= \frac{1}{w_\ell}\frac{2w_\ell}{2-2g} f_{\boldsymbol{W_S}}(\boldsymbol{w_S}), \quad g \le w_\ell \le 1, \end{aligned} \tag{19}$$

where we have used Lemma 3.2, (18), and Theorem 3.1. □

4 Exhaustive Rollout

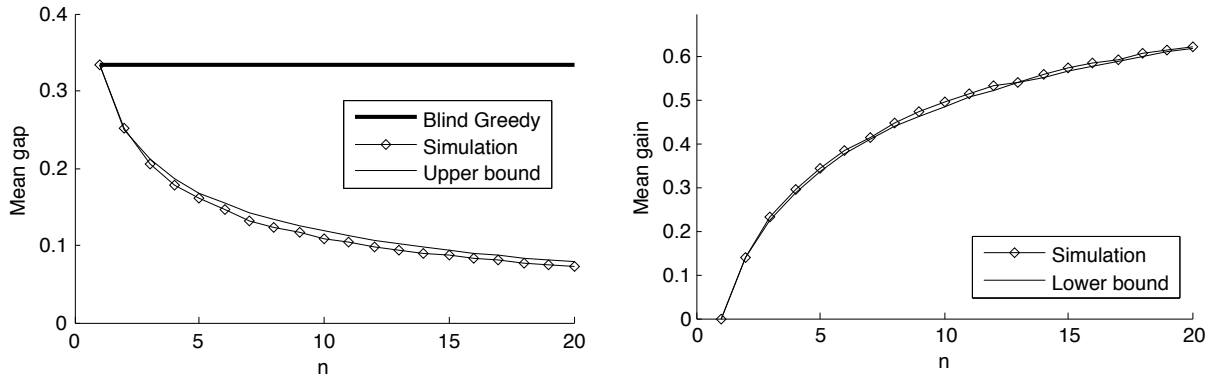
The EXHAUSTIVE-ROLLOUT algorithm is shown in Algorithm 2. It takes as input a sequence of items $\boldsymbol{I}$ and capacity b. At each iteration, indexed by t, the algorithm considers all items in the available sequence $\overline{\boldsymbol{I}}$. It calculates the value obtained by moving each item to the front of the sequence and applying the BLIND-GREEDY algorithm. The algorithm then adds the item with the highest estimated value (if it exists) to the solution. We implicitly assume a consistent tie-breaking method, such as giving preference to the item with the lowest index. The next iteration then proceeds with the remaining sequence of items.

Algorithm 2 EXHAUSTIVE-ROLLOUT

Input: Item sequence $\boldsymbol{I} = \langle 1, \ldots, n \rangle$ with profit sequence $\boldsymbol{p_I}$ and weight sequence $\boldsymbol{w_I}$, capacity b.
Output: Feasible solution sequence $\boldsymbol{S}$, value U.

1. Initialize solution sequence $\boldsymbol{S} \leftarrow \langle\rangle$, remaining item sequence $\overline{\boldsymbol{I}} \leftarrow \boldsymbol{I}$, remaining capacity $\overline{b} \leftarrow b$, current iteration $t \leftarrow 1$, and value $U \leftarrow 0$.
2. Iteration t:
   (a) For each item in remaning sequence $i \in \overline{\boldsymbol{I}}$,
      i. Let $\overline{\boldsymbol{I}}^{\boldsymbol{i}}$ denote the sequence $\overline{\boldsymbol{I}}$ with i moved to the first position.
      ii. Estimate the value of the sequence: let U_i be the value given by BLIND-GREEDY on the item sequence $\overline{\boldsymbol{I}}^{\boldsymbol{i}}$ with capacity $\overline{b}$.
   (b) If $\max_i U_i > 0$, then
      i. Determine the item with the maximum estimated value: $i^* \leftarrow \operatorname{argmax}_i U_i$.
      ii. Add item i^* to solution sequence $\boldsymbol{S} \leftarrow \boldsymbol{S} : \langle i^* \rangle$ and remove item i^* from remaining item sequence $\overline{\boldsymbol{I}} \leftarrow \overline{\boldsymbol{I}} \setminus \langle i^* \rangle$.
      iii. Update remaining capacity $\overline{b} \leftarrow \overline{b} - w_i$ and value $U \leftarrow U + p_i$.
   (c) If n iterations have occurred ($t = n$), stop. Otherwise, set iteration $t \leftarrow t + 1$ and goto 2.

We only consider the first iteration, which tries using BLIND-GREEDY after moving each item to the



(a) (b)

Figure 2: Performance bounds and simulated values for (a) expected gap $\mathbb{E}[V_*(n)|\cdot]$ and (b) expected gain $\mathbb{E}[Z_*(n)|\cdot]$ after running a single iteration of EXHAUSTIVE-ROLLOUT on the subset sum problem and 0-1 knapsack problem, respectively. For each n, the mean values are shown for 10^5 simulations.

front of the sequence, and takes the best of these solutions. This gives an upper bound for the subset sum gap and a lower bound on the 0-1 knapsack problem gain following from additional iterations. The technical condition for these bounding properties to hold is that the base policy/algorithm is sequentially consistent, as defined in [1]. It is easy to verify that BLIND-GREEDY satisfies this property. For the subset sum problem, let $V_*(n)$ denote the gap obtained after a single iteration of EXHAUSTIVE-ROLLOUT on the stochastic model with n items. We have the following bounds.

Theorem 4.1. For the subset sum problem, the gap $V_*(n)$, after running a single iteration of EXHAUSTIVE-ROLLOUT, satisfies

$$\mathbb{E}\left[V_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] \quad \leq \quad \frac{1}{n(2+n)} + \frac{1}{n}\sum_{m=0}^{n-2}\frac{9+2m}{3(3+m)(4+m)}. \tag{20}$$

Corollary 4.1.

$$\mathbb{E}\left[V_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] \quad \leq \quad \frac{1}{n(2+n)} + \frac{1}{n}\log\left[\left(\frac{3+2n}{5}\right)\left(\frac{7}{5+2n}\right)^{1/3}\right]. \tag{21}$$

Theorem 4.2.

$$\lim_{n\to\infty} \mathbb{E}\left[V_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] = 0, \qquad \mathbb{E}\left[V_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] = O\left(\frac{\log n}{n}\right). \tag{22}$$

A plot of the bounds and simulated results is shown in Figure 2(a). For the 0-1 knapsack problem, let $Z_*(n)$ denote the gain given by a single iteration of Exhaustive-Rollout. The expected gain is bounded by the two following theorems, where $H(n)$ is the nth harmonic number, $H(n) := \sum_{\ell=1}^{n} \frac{1}{\ell}$.

Theorem 4.3. For the 0-1 knapsack problem, the gain $Z_*(n)$, after running a single iteration of Exhaustive-Rollout, satisfies

$$\begin{aligned}
&\mathbb{E}\left[Z_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] \geq 1 + \frac{2}{n(n+1)} - \frac{2H(n)}{n^2} \\
&+\frac{1}{n}\sum_{m=0}^{n-2}\left(\sum_{j=1}^{m+1} T(j,m) + \big((186 + 472m + 448m^2 + 203m^3 + 45m^4 + 4m^5)\right. \\
&-(244 + 454m + 334m^2 + 124m^3 + 24m^4 + 2m^5)H(m+1) \\
&\left.-(48 + 88m + 60m^2 + 18m^3 + 2m^4)(H(m+1))^2\big)\,\frac{1}{(m+1)(m+2)^3(m+3)^2}\right),
\end{aligned} \tag{23}$$

where

$$\begin{aligned}
T(j,m) \quad := \quad & \frac{2\left(-4 + j - 4m + jm - m^2 - \left(j + (2+m)^2\right)H(j)\right)}{j(-3+j-m)(-2+j-m)(1+m)(2+m)} \\
& + \frac{2(j + (2+m)^2)H(3+m)}{j(-3+j-m)(-2+j-m)(1+m)(2+m)}.
\end{aligned} \tag{24}$$

Theorem 4.4.

$$\lim_{n\to\infty} \mathbb{E}\left[Z_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] = 1, \qquad 1 - \mathbb{E}\left[Z_*(n)\left|\sum_{i=1}^{n} W_i > B\right.\right] = O\left(\frac{\log^2 n}{n}\right). \tag{25}$$

The expected gain approaches unit value at a rate slightly slower than the convergence rate for the subset sum problem. This is likely a result of the fact that in the subset sum problem, the algorithm is searching

for an item with one criteria: a weight approximately equal to the gap. For the 0-1 knapsack problem, however, the algorithm must find an item satisfying two criteria: a weight smaller than the gap and a profit approximately equal to one. The gain is plotted with simulated values in Figure 2(b). While the bound in Theorem 4.3 does not admit a simple integral bound, omitting the nested summation term $\sum_{j=1}^{m} T(j,m)$ gives a looser but valid bound. We show the proof of Theorem 4.1 in the remainder of this section. All remaining results are given in the supplementary material.

Exhaustive Rollout: Subset Sum Problem Analysis

The proof method for Theorem 4.1 is to visually analyze the solution sequence given by Blind-Greedy on the non-negative real line, as shown in Figure 1. We consider the effect of individually moving each item to the front of the sequence, which will cause the other items to shift to the right. Our strategy is to perform this analysis while conditioning on three parameters: the greedy gap G, the critical item K, and the weight of the last packed item W_{K-1}. We then find the minimum gap given by trying all items and integrate over conditioned variables to obtain the final bound.

To analyze solutions obtained by using Blind-Greedy after moving a given item to the front of the sequence, we introduce two definitions. The jth insertion critical item L_j, defined for $j \geq 2$, is the first item that is infeasible to pack by Blind-Greedy when item j is moved to the front of the sequence. Equivalently, L_j satisfies

$$\begin{cases} W_j + \sum_{i=1}^{L_j-1} W_i \ \mathbb{1}(i \neq j) \leq B < W_j + \sum_{i=1}^{L_j} W_i \ \mathbb{1}(i \neq j), & \text{if } W_j \leq B, \\ L_j = j, & \text{if } W_j > B. \end{cases} \tag{26}$$

We then define the corresponding jth insertion gap V_j, which is the gap given by the greedy algorithm when item j is moved to the front of the sequence:

$$V_j := B - \mathbb{1}(W_j \leq B) \left(W_j + \sum_{i=1}^{L_j-1} W_i \ \mathbb{1}(i \neq j) \right), \quad j \geq 2. \tag{27}$$

In the following three lemmas, we bound the probability distribution of the insertion gap for packed items ($j \leq K-1$), the critical item ($j = K$), and the remaining items ($j \geq K+1$), while assuming that $K > 1$. Lemma 4.4 then handles the case where $K = 1$. Thereafter, we bound the minimum of these gaps and the greedy gap G, and finally integrate over the conditioned variables to obtain the bound on the expected

minimum gap. The key analysis is illustrated in the proof of Lemma 4.2; the related proofs of Lemma 4.3 and Lemma 4.4 are given in the supplementary material. The event $\mathcal{C}_j$ indicates that item j is critical, and $\overline{\mathcal{C}_1}$ indicates the event that the first item is not critical. Recall that $\boldsymbol{P_I} = \boldsymbol{W_I}$ for the subset sum problem.

Lemma 4.1. For $K > 1$ and $j = 2, \ldots, K-1$, the jth insertion gap satisfies

$$V_j = G \tag{28}$$

with probability one.

Proof. This follows trivially since the term $\sum_{i=1}^{K-1} W_i$ in (5) does not depend on the order of summation. □

Lemma 4.2. For $K > 1$ and $j = K+1, \ldots, n$, the jth insertion gap satisfies $V_j \leq V_j^u$ with probability one, where V_j^u is a deterministic function of (G, W_{K-1}, W_j), and conditioning only on (G, W_{K-1}) gives

$$\begin{aligned} \mathbb{P}(V_j^u > v | g, w_{K-1}, \overline{\mathcal{C}_1}) &= (g-v)_+ + (w_{K-1}-v)_+ - (g + w_{K-1} - v - 1)_+ + (1 - g - w_{K-1})_+ \\ &=: \mathbb{P}(V^u > v | g, w_{K-1}, \overline{\mathcal{C}_1}). \end{aligned} \tag{29}$$

Proof. Fix $K = k$ for $k > 1$. To simplify notation, we make the event $\overline{\mathcal{C}_1}$ implicit throughout the proof. Define the random variable V_j^u so that

$$V_j^u := \begin{cases} V_j, & \text{if } L_j = k \vee L_j = k-1, \\ 1, & \text{if } L_j \leq k-2 \vee L_j = j. \end{cases}$$

While V_j may, in general, depend on $(G, W_j, W_1, \ldots, W_{k-1})$, the variable V_j^u is chosen so that it only depends on (G, W_{k-1}, W_j). In cases where V_j does only depend on (G, W_{k-1}, W_j), we have $V_j^u = V_j$. When V_j depends on more than these three variables, V_j^u assumes a worst-case bound of unit value.

We begin by analyzing the case where $L_j = k \vee L_j = k-1$, so that the insertion gap V_j is equal to V_j^u. For $G = g$ and $\boldsymbol{W_I} = \boldsymbol{w_I}$, a diagram illustrating the insertion gap as determined by g, w_{k-1}, and w_j is shown in Figure 3. We will follow the convention of using lowercase letters for random variables shown in figures and when referring these variables. The knapsack is shown at the top of the figure with items packed sequentially from left to right. The plot at the bottom shows the insertion gap V_j that occurs when item j is inserted at the front of the sequence, causing the remaining packed items to slide to the right. The plot is best understood by visualizing the effect of varying sizes of w_j. If w_j is very small, the items slide to the

right and reduce the gap by the amount w_j. Clearly, if $w_j = g$, then $v_j = 0$, as indicated by the function. As soon as w_j is slightly larger than g, it is infeasible to pack item $k-1$ and the gap jumps. Thus for the instance shown, the jth insertion gap is a deterministic function of (g, w_{k-1}, w_j).

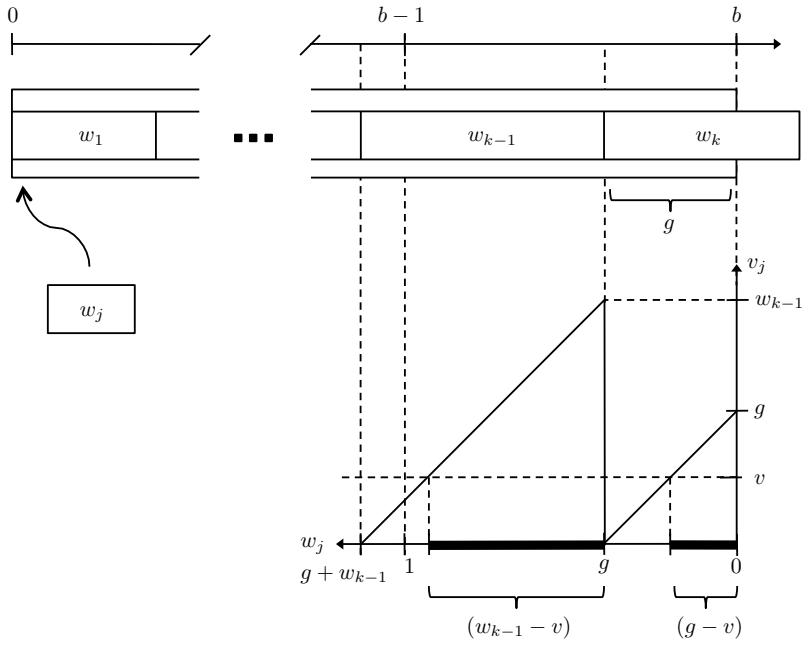


Figure 3: Insertion gap v_j as a function of w_j, parameterized by (w_{k-1}, g). The function starts at g and decreases at unit rate, except at $w = g$, where the function jumps to value w_{k-1}. The probability of the event $V_j > v$ conditioned only on w_{k-1} and g is given by the total length of the bold regions, assuming that $v < g$ and $g + w_{k-1} - v \leq 1$. Based on the sizes of g and w_{k-1} shown, only the events $L_j = k$ and $L_j = k-1$ are possible.

Considering the instance in the figure, if we only condition on g and w_{k-1} and allow W_j to be random, then V_j becomes a random variable, whose only source of uncertainty is W_j. Since by Lemma 3.4 W_j has distribution $\mathcal{U}[0,1]$, the probability of the event $V_j > v$ is given by the length of the bold regions on the w_j axis.

We now explicitly describe the length of the bold regions for all cases of w_{k-1} and g; this will include the case $L_j = k-2 \ \vee \ L_j = j$ (not possible for the instance in the figure), so the length of the bold regions will define V_j^u. Starting with the instance shown, we have $\mathbb{P}(V_j^u > v | g, w_{k-1}) = (g - v) + (w_{k-1} - v)$, as given by the lengths of the two bold regions, corresponding to the events $L_j = k$ and $L_j = k-1$, respectively. This requires that $v \leq g$ and $v \leq w_{k-1}$, so the expression becomes $\mathbb{P}(V_j^u > v | g, w_{k-1}) = (g - v)_+ + (w_{k-1} - v)_+$. We must account for the case where $g + w_{k-1} - v > 1$, requiring that we subtract length $(g + w_{k-1} - v - 1)$, so we revise the expression to $\mathbb{P}(V_j^u > v | g, w_{k-1}) = (g - v)_+ + (w_{k-1} - v)_+ - (g + w_{k-1} - v - 1)_+$. Finally,

for the case of $g + w_{k-1} < 1$, we must take care of the region where $w_i \in [g + w_{k-1}, 1]$. It is at this point that the event $L_j \leq k-2$ or $L_j = j$ becomes possible and the distinction between V_j^u and V_j is made. Here, we have by definition $V_j^u = 1$, which trivially satisfies $V_j \leq V_j^u$, so for any $0 \leq v < 1$ this region contributes $(1 - g - w_{k-1})$ to $\mathbb{P}(V_j^u > v|g, w_{k-1})$. This is handled by adding the term $(1 - g - w_{k-1})_+$ to the expression. We finally arrive at

$$\mathbb{P}(V_j^u > v|g, w_{k-1}) \quad = \quad (g - v)_+ + (w_{k-1} - v)_+ - (g + w_{k-1} - v - 1)_+ + (1 - g - w_{k-1})_+. \tag{30}$$

This holds true for any fixed k, as long as $k > 1$, so we may replace w_{k-1} with w_{K-1} and make the event $\overline{\mathcal{C}_1}$ explicit to obtain the statement of the lemma. □

Lemma 4.3. For $K > 1$, the Kth insertion gap satisfies $V_K \leq V_K^u$ with probability one, where V_K^u is a deterministic function of (G, W_{K-1}, W_K), and conditioning only on (G, W_{K-1}) gives

$$\begin{aligned}\mathbb{P}(V_K^u > v|g, w_{K-1}, \overline{\mathcal{C}_1}) \quad &= \quad \left(\frac{1}{1-g}\right)((w_{K-1} - v)_+ - (g + w_{K-1} - v - 1)_+ + (1 - g - w_{K-1})_+) \\ &=: \quad \mathbb{P}(\widetilde{V}^u > v|g, w_{K-1}, \overline{\mathcal{C}_1}).\end{aligned} \tag{31}$$

Proof. Supplementary material.

Lemma 4.4. For $K = 1$ and $j = 2, \ldots, n$, the jth insertion gap is a deterministic function of (W_j, G), and conditioning only on G gives $\mathbb{P}(V_j > v|g, \mathcal{C}_1) = (1 - v)\mathbb{1}(v < g)$.

Proof. Supplementary material.

Recall that $V_*(n)$ is the gap obtained after the first iteration of the rollout algorithm on an instance n items, which we refer to as the minimum gap,

$$V_*(n) := \min(G, V_2, \ldots, V_n). \tag{32}$$

We will make the dependence on n implicit in what follows, so that $V_* = V_*(n)$. We may now prove the final result.

Proof of Theorem 4.1. For $K = k > 1$, we have $V_* \leq V_*^u$ with probability one, where

$$V_*^u \quad := \quad \min(G, V_k^u, V_{k+1}^u, \ldots, V_n^u). \tag{33}$$

This follows from Lemmas 4.1 - 4.3, as $V_j = G$ for $j \leq k-1$. From the analysis in Lemmas 4.2 and 4.3, for each $j \geq k$, V_j^u is a deterministic function of (G, W_{k-1}, W_k, W_j). Furthermore, from Lemma 3.4, the item weights W_j for $j \geq k+1$ are independently distributed on $\mathcal{U}[0,1]$, and W_k is independently distributed on $\mathcal{U}[g,1]$. Thus, conditioning only on G and W_{k-1} makes V_j^u independent for $j \geq k$, and by the definition of the minimum function,

$$\begin{aligned}\mathbb{P}(V_*^u > v|g, w_{k-1}, k, \overline{\mathcal{C}_1}) &= \mathbb{P}(G > v|g, w_{k-1}, \overline{\mathcal{C}_1})\mathbb{P}(V_k^u > v|g, w_{k-1}, \overline{\mathcal{C}_1}) \prod_{j=k+1}^{n} \mathbb{P}(V_j^u > v|g, w_{k-1}, \overline{\mathcal{C}_1}) \\ &= \mathbb{P}(G > v|g, w_{k-1}, \overline{\mathcal{C}_1})\mathbb{P}(\widetilde{V}^u > v|g, w_{k-1}, \overline{\mathcal{C}_1}) \left(\mathbb{P}(V^u > v|g, w_{k-1}, \overline{\mathcal{C}_1})\right)^{(n-k)}.\end{aligned} \tag{34}$$

Marginalizing over W_{k-1} and G using Lemma 3.4 and Theorem 3.1,

$$\mathbb{P}(V_*^u > v|k, \overline{\mathcal{C}_1}) = \int_0^1 \int_0^1 \mathbb{P}(V_*^u > v|g, w_{k-1}, k, \overline{\mathcal{C}_1}) f_{W_{k-1}}(w_{k-1}) f_G(g) \mathrm{d}w_{k-1} \mathrm{d}g. \tag{35}$$

We refer to $\mathbb{P}(V_*^u > v|k, \overline{\mathcal{C}_1})$ as $\mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1})$ via the substitution $M := n - K$ to simplify expressions. As shown in the supplementary material, evaluation of the integral gives

$$\mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1}) = \begin{cases} \mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1})_{\leq \frac{1}{2}}, & v \leq \frac{1}{2} \\ \mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1})_{> \frac{1}{2}}, & v > \frac{1}{2}, \end{cases} \tag{36}$$

where

$$\begin{aligned}\mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1})_{\leq \frac{1}{2}} &= \frac{1}{3(3+m)} \big(2m(1-2v)^m + m(1-v)^m + 9(1-v)^{3+m} \\ &\quad -12m(1-2v)^m v - 3m(1-v)^m v + 24m(1-2v)^m v^2 \\ &\quad +3m(1-v)^m v^2 - 16m(1-2v)^m v^3 - m(1-v)^m v^3\big), \end{aligned} \tag{37}$$

$$\mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1})_{> \frac{1}{2}} = \frac{1}{3}(1-v)^{3+m} + \frac{2(1-v)^{3+m}}{3+m}. \tag{38}$$

Calculating the expected value gives a surprisingly simple expression

$$\mathbb{E}[V_*^u|m, \overline{\mathcal{C}_1}] = \int_0^1 \mathbb{P}(V_*^u > v|m, \overline{\mathcal{C}_1}) \mathrm{d}v = \frac{9+2m}{3(3+m)(4+m)}. \tag{39}$$

We now consider the case $\mathcal{C}_1$, where the first item is critical. By Lemma 4.4, each V_j for $j \geq 2$ is a

deterministic function of G and W_j. All W_j for $j \geq 2$ are independent by Lemma 3.2, so

$$\mathbb{P}(V_* > v|g, \mathcal{C}_1) = \prod_{j=2}^{n} \mathbb{P}(V_j > v|g, \mathcal{C}_1) = (1-v)^{(n-1)} \mathbb{1}(v < g). \tag{40}$$

Integrating over G by Theorem 3.1, we have

$$\mathbb{P}(V_* > v|\mathcal{C}_1) = \int_0^1 \mathbb{P}(V_* > v|g, \mathcal{C}_1) f_G(g) \mathrm{d}g = (1-v)^{(n-1)}(1-2v+v^2), \tag{41}$$

which can be used to calculate the expected value. Finally, accounting for all cases of K using total expectation and Lemma 3.1,

$$\mathbb{E}[V_*] \quad \leq \quad \frac{1}{n}\mathbb{E}[V_*|\mathcal{C}_1] + \frac{1}{n}\sum_{m=0}^{n-2} \mathbb{E}[V_u^*|\overline{\mathcal{C}_1}, m] = \frac{1}{n(2+n)} + \frac{1}{n}\sum_{m=0}^{n-2} \frac{9+2m}{3(3+m)(4+m)}. \tag{42}$$

Throughout all of the analysis in this section, we have implicitly assumed that $\sum_{i=1}^{n} W_i > B$. Making this condition explicit gives the desired bound. □

5 Consecutive Rollout

The Consecutive-Rollout algorithm is shown in Algorithm 3. The algorithm takes as input a sequence of items $\boldsymbol{I}$ and capacity b, and makes calls to Blind-Greedy as a subroutine. At iteration i, the algorithm calculates the value (U_+) of adding item i to the solution and using Blind-Greedy on the remaining items, and the value (U_-) of not adding the item to the solution and using Blind-Greedy thereafter. The item is then added to the solution only if the former valuation (U_+) is larger. Again, we assume a consistent tie breaking method.

We again only focus on the result of the first iteration of the algorithm; bounds from the first iteration are valid for future iterations. A single iteration of Consecutive-Rollout effectively takes the best of two solutions, the one obtained by Blind-Greedy and the solution obtained from using Blind-Greedy after removing the first item. Let $V_*(n)$ denote the gap obtained by a single iteration of the rollout algorithm for the subset sum problem with n items under the stochastic model. The results in this section hold for $n \geq 3$ and are tight for $n = 3$, as the proofs only depend on the first three items; employing this number of items strikes a balance between proof tractability and performance tightness.

Theorem 5.1. For the subset sum problem with $n \geq 3$, the gap $V_*(n)$, obtained by running a single iteration

Algorithm 3 CONSECUTIVE-ROLLOUT

Input: Item sequence $\boldsymbol{I} = \langle 1, \ldots, n \rangle$ with profit sequence $\boldsymbol{p_I}$ and weight sequence $\boldsymbol{w_I}$, capacity b.
Output: Feasible solution sequence $\boldsymbol{S}$, value U.

1. Initialize solution sequence $\boldsymbol{S} \leftarrow \langle\rangle$, remaining item sequence $\overline{\boldsymbol{I}} \leftarrow \boldsymbol{I}$, remaining capacity $\overline{b} \leftarrow b$, current item $i \leftarrow 1$, and value $U \leftarrow 0$.
2. Iteration i:
   (a) Estimate the value of adding item i: let U_+ be the value given by BLIND-GREEDY on the item sequence $\overline{\boldsymbol{I}}$ with capacity $\overline{b}$.
   (b) Estimate the value of skipping item i: let U_- be the value given by BLIND-GREEDY on the item sequence $\overline{\boldsymbol{I}} \setminus \langle i \rangle$ with capacity $\overline{b}$.
   (c) If the estimated value of adding item i is greater ($U_+ > U_-$),
      i. Add item i to solution sequence $\boldsymbol{S} \leftarrow \boldsymbol{S} : \langle i \rangle$.
      ii. Update remaining capacity $\overline{b} \leftarrow \overline{b} - w_i$ and value $U \leftarrow U + p_i$.
   (d) Remove item i from remaining item sequence $\overline{\boldsymbol{I}} \leftarrow \overline{\boldsymbol{I}} \setminus \langle i \rangle$.
   (e) If the final item has been reached ($i = n$), stop. Otherwise, move to next item $i \leftarrow i + 1$ and goto 2.

of CONSECUTIVE-ROLLOUT, satisfies

$$\mathbb{E}\left[V_*(n) \middle| \sum_{i=1}^{n} W_i > B \right] \leq \frac{3 + 13n}{60n} \leq \frac{7}{30} \approx 0.233. \tag{43}$$

As expected, there is not a strong dependence on n for this algorithm. The bounds are shown with simulated performance in Figure 4(a). A similar result holds for the 0-1 knapsack problem.

Theorem 5.2. For the 0-1 knapsack problem with $n \geq 3$, the gain $Z_*(n)$, obtained by running a single iteration of CONSECUTIVE-ROLLOUT, satisfies

$$\mathbb{E}\left[Z_*(n) \middle| \sum_{i=1}^{n} W_i > B \right] \geq \frac{-26 + 59n}{288n} \geq \frac{151}{864} \approx 0.175. \tag{44}$$

The bound is plotted with simulated values in Figure 4(b). The proofs of Theorem 5.1 and Theorem 5.2 are given in the supplementary material.

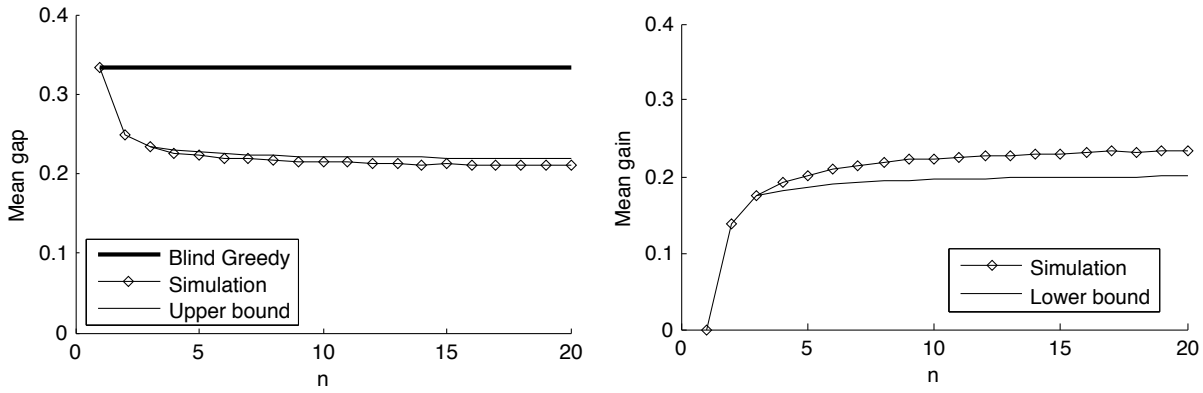


Figure 4: Performance bounds and simulated values for the expected gap $\mathbb{E}[V_*(n)|\cdot]$ and expected gain $\mathbb{E}[Z_*(n)|\cdot]$ after running a single iteration of the CONSECUTIVE-ROLLOUT algorithm on the subset sum problem and 0-1 knapsack problem, respectively. For each n, the mean values are shown for 10^5 simulations.

6 Conclusions

We have shown strong performance bounds for both the exhaustive rollout and consecutive rollout techniques on the subset sum problem and 0-1 knapsack problem. These results hold after only a single iteration and provide bounds for additional iterations. Simulation results indicate that these bounds are very close in comparison with realized performance of a single iteration. Our results also indicate the asymptotic behavior (asymptotic with respect to the total number of items) of the expected performance of both rollout techniques for the two problems.

An interesting direction in future work is to consider a second iteration of the rollout algorithm. The worst-case analysis of rollout algorithms for the 0-1 knapsack problem in [4] shows that running one iteration results in a notable improvement, but it is not possible to guarantee additional improvement with more iterations for the given base policy. This behavior is generally not observed in practice [2], and it may be possible to prove performance guarantees from additional iterations in the average-case scenario. A related topic is to still consider only the first iteration of the rollout algorithm, but with a larger lookahead length (e.g., trying all pairs of items for the exhaustive rollout, rather than just each item individually). Finally, it is desirable to have theoretical results for more complex problems. Studying problems with multidimensional state space is appealing, since these are the types of problems where, in practice, rollout techniques are often used and perform well. In this direction, it would be useful to consider problems such as the bin packing problem, the multiple knapsack problem, and the multidimensional knapsack problem.

Acknowledgements. The authors would like to thank the two referees for the helpful comments. Research supported by NSF grant 1029603 and ONR grant N00014-12-1-0033; the first author is supported in part by a NSF graduate research fellowship.

References

1. Bertsekas, D.P., Tsitsiklis, J., Wu, C.: Rollout algorithms for combinatorial optimization. Journal of Heuristics 3, 245–262 (1997)

2. Bertsekas, D.P., Castanon, D.A.: Rollout algorithms for stochastic scheduling problems. Journal of Heuristics 5, 89–108 (1999)

3. Bertsekas, D.P.: Dynamic Programming and Optimal Control. Athena Scientific, 3rd edn. (2007)

4. Bertazzi, L.: Minimum and worst-case performance ratios of rollout algorithms. Journal of Optimization Theory and Applications 152, 378–393 (2012)

5. Borgwardt, K., Tremel, B.: The average quality of greedy-algorithms for the subset-sum-maximization problem. Mathematical Methods of Operations Research 35, 113–149 (1991)

6. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack problems. Springer (2004)

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

8. Tesauro, G., Galperin, G.R.: On-line policy improvement using monte-carlo search. Advances in Neural Information Processing Systems pp. 1068–1074 (1997)

9. Secomandi, N.: A rollout policy for the vehicle routing problem with stochastic demands. Oper. Res. 49, 796–802 (2001)

10. Tu, F., Pattipati, K.: Rollout strategies for sequential fault diagnosis. AUTOTESTCON Proceedings, pp. 269–295. IEEE (2002)

11. Li, Y., Krakow, L.W., Chong, E.K.P., Groom, K.N.: Approximate stochastic dynamic programming for sensor scheduling to track multiple targets. Digit. Signal Process. 19, 978–989 (2009)

12. Martello, S., Toth, P.: Worst-case analysis of greedy algorithms for the subset-sum problem. Mathematical programming 28, 198–205 (1984)

13. D'Atri, G., Puech, C.: Probabilistic analysis of the subset-sum problem. Discrete Applied Mathematics 4, 329–334 (1982)

14. Pferschy, U.: Stochastic analysis of greedy algorithms for the subset sum problem. Central European Journal of Operations Research 7, 53–70 (1999)

15. Szkatula, K., Libura, M.: Probabilistic analysis of simple algorithms for binary knapsack problem. Control and Cybernetics 12, 147–157 (1983)

16. Szkatula, K., Libura, M.: On probabilistic properties of greedy-like algorithms for the binary knapsack problem. Proceedings of Advanced School on Stochastics in Combinatorial Optimization pp. 233–254 (1987)

17. Diubin, G., Korbut, A.: The average behaviour of greedy algorithms for the knapsack problem: general distributions. Mathematical Methods of Operations Research 57, 449–479 (2003)

18. Calvin, J.M., Leung, J.Y.T.: Average-case analysis of a greedy algorithm for the 0/1 knapsack problem. Operations Research Letters 31, 202–210 (2003)

19. Dean, B.C., Goemans, M.X., Vondrdk, J.: Approximating the stochastic knapsack problem: The benefit of adaptivity. Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on, pp. 208–217. IEEE (2004)

20. Lueker, G.S.: Average-case analysis of off-line and on-line knapsack problems. Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms, pp. 179–188. Society for Industrial and Applied Mathematics (1995)

21. Beier, R., Vöcking, B.: Random knapsack in expected polynomial time. Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, pp. 232–241. ACM (2003)