

Resource Allocation With Non-Deterministic Demands and Profits

Nan Hu*, Diego Pizzocaro[†], Matthew P. Johnson[‡], Thomas La Porta*, Alun D. Preece[†]

*Department of Computer Science and Engineering, The Pennsylvania State University, US

[‡]Networked and Embedded Systems Laboratory, University of California at Los Angeles, US

[†]School of Computer Science and Informatics, Cardiff University, UK

Abstract—Support for intelligent and autonomous resource management is one key factor to the success of modern sensor network systems. The limited resources, such as exhaustible battery life, moderate processing ability and finite bandwidth, restrict the system’s ability to serve multiple users simultaneously. It always happens that only a subset of tasks is selected with the goal of maximizing total profit. Besides, because of uncertain factors like unreliable wireless medium or variable quality of sensor outputs, it is not practical to assume that both demands and profits of tasks are deterministic and known a priori, both of which may be stochastic following certain distributions.

In this paper, we model this resource allocation challenge as a stochastic knapsack problem. We study a specific case in which both demands and profits follow normal distributions, which are then extended to Poisson and Binomial variables. A couple of tunable parameters are introduced to configure two probabilities: one limits the capacity overflow rate with which the combined demand is allowed to exceed the available supply, and the other sets the minimum chance at which expected profit is required to be achieved. We define *relative values* for random variables in given conditions, and utilize them to search for the best resource allocation solutions. We propose heuristics with different optimality/efficiency tradeoffs, and find that our algorithms run relatively fast and provide results considerably close to the optimum.

I. INTRODUCTION

Resource allocation is a fundamental and critical problem studied in different forms in many communities. Consider a surveillance sensor network system, in which resources, like still and video cameras, batteries and transmission bandwidth, are associated with nodes and links, and shared among multiple tasks from different users. The productivity of a task may depend on receiving exclusive access to certain resources (i.e., a tracking task may benefit from a specific directional camera while videos are authorized to be transmitted via a reserved high-quality channel). In many cases, since resources are often limited, the system cannot serve every user simultaneously. A natural objective for such a problem is to decide which tasks should be admitted under given conditions so that the greatest profit may be achieved, consistent with the capacity constraints.

If the demands and profits of tasks are fixed and known exactly, this problem can be formulated as the extensively studied knapsack problem [1] [2], where

each type of resource represents one feature of the knapsack (e.g., length, volume, etc.), and each productive task requesting resources is like an item of certain value with specific characteristics to be placed into the knapsack. In the *single-dimensional* cases, where only one type of resource is considered, this problem can be solved by a Fully Polynomial-Time Approximation Scheme (FPTAS) based on Dynamic Programming (DP) [3]; and in the *multi-dimensional* cases, where multiple resources are taken into account, the problem is solvable by a Polynomial-Time Approximation Scheme (PTAS) [4] and various heuristics and approximation algorithms.

In practice, however, this deterministic model is not suitable for many realistic applications. It is often hard to predict *how much* resource is actually needed or *how much* profit a successfully completed task (i.e., the task that is allocated required resources) will return.

Consider the surveillance system again. More cameras may be needed when many big trucks block the sight of the targets. Also, if variable rate coding is used, the network capacity consumed by the sensors will be hard to predict exactly. Likewise, the pictures or videos delivered may be noisy or obscured, leading to difficulty of predicting the profit achieved precisely.

Therefore, a more conservative assumption assumes that the demands and profits are random variables, changing over time depending on different conditions, and only their distributions are known. Still, the best decision is sought to yield the greatest profit.

Reasonably replacing random variables with constant values and solving the transformed problem is feasible, but such a solution may not only lead to low utilization of resources, but also fail to guarantee the minimum probability of achieving the suggested profit.

For instance, given two tasks, whose demands and profits all independently follow the uniform distribution $U(0, 100)$, the method we mentioned above fixes each demand or profit as 75 (i.e., 75th percentile of the random variable), leading to a new problem in which we need at least 150 units of the resource to admit both tasks to get 150 units of profit. However, the probability that the combined demand actually requires less than 150 units is quite high (i.e., 87.5%), and there is a low chance of 12.5% for obtaining at least 150 units of profit.

This variant of the knapsack problem is often posed as the stochastic knapsack problem (SKP) [5]. In particular, if only the demands are considered as stochastic, it then becomes a chance-constrained problem, in which an overflow threshold is set to limit the maximum probability that the total demands may violate the capacity constraints. Both sampling-based approximation methods [6] and equivalent transformation algorithms [7] have been proposed for this variant. Instead, if only the profits are stochastic, the solution then depends on a target value [8] or an acceptable probability [9]. Rather than maximizing the sum of fixed profits, the objective now is to optimize the chance of achieving a certain profit or to maximize the lower bound of profit which is guaranteed to be obtained with high confidence.

In this paper we focus on the resource allocation scenarios with both stochastic resource demands and achievable profits. After formally defining this variant of SKP, we solve a special case with normal distributed demands and profits. Unlike the method proposed in [7] which converts all random variables into constants by extending the concept of *effective bandwidth* [10], we define *relative values* for random variables in given conditions in order to search for the most efficient way of utilizing the resources without making the running time unreasonable.

To our best knowledge, there is little work on this type of resource allocation problem in which both demands and profits are non-deterministic. We develop heuristics for the case with normal distributed variables, apply them to the Poisson and Binomial variables as extensions, and conduct comprehensive numerical experiments to evaluate their performance. The results show that, in both single- and multi-dimensional cases, our algorithms not only match the optimal solutions quite closely, but also run efficiently.

Our main contributions include:

- Introduction of a novel method based on the *relative values* of random variables to efficiently solve the stochastic resource allocation problem;
- Development of algorithms for both single- and multi-dimensional cases with normal distributed variables, which closely matches the optimal solutions in practice with fast running speed;
- Extended algorithms to be adaptable for Poisson and Binomial scenarios.

The rest of this paper is organized as follows. Section II presents related work. Section III formally defines the stochastic resource allocation problem, and outlines its mathematical formulation. Section IV describes the algorithms for both single- and multi-dimensional cases. Section V presents the results of our numerical experiments. Finally, Section VI concludes the paper.

II. RELATED WORK

Different variants of SKP have been widely discussed during the past decades. Some works like [5], [11], [12] have addressed the univariate dynamic version of SKP (D-SKP), in which both the size and value of each item may follow some probabilistic distributions, but items are placed in the knapsack sequentially, and the act of making an accept/reject decision instantiates the values of random variables, which do not change thereafter. The solution is returned as a particular order of insertions. [5] proposes a linear time on-line algorithm for which the expected difference between the optimum and the approximate total value is $O(\log^{3/2} n)$. Dean *et al.* [11] consider both nonadaptive policies and adaptive policies, and devise a polynomial-time adaptive policy that approximates the optimal adaptive policy to within a factor of $3 + \varepsilon$ for any constant $\varepsilon > 0$. Goel *et al.* [12] then improve this approximation factor to $\frac{8}{3} + \varepsilon$.

Dean *et al.* also study the multi-dimensional version of D-SKP as a special case of stochastic packing problem [13], and İlhan *et al.* [8] formulate it as a DP solvable problem for discrete random rewards and present a heuristic that mixes adaptive and static policies to overcome the “curse of dimensionality”. Johnson *et al.* [14] propose algorithms to solve a version of the multiple knapsack problem in which tasks arrive over time and may have different durations. In their settings, tradeoffs between the profit of a given resource assignment and the total expected profit are evaluated under different situations. Again, our problem formulation differs from these in that the set of tasks are fixed and known, and the demands and profits are not decided even after being selected, which is defined as a static stochastic knapsack problem (S-SKP).

Most prior works on S-SKP focus only on either random sizes or random rewards. The work of Steinberg *et al.* [15] presents a DP-based approximate algorithm for cases with deterministic sizes and normal distributed rewards. In the same setting, [9] combines DP with a search procedure, [16] offers an alternative hybrid DP/branch-and-bound algorithm, and [17] introduces solutions for cases with normal and more general distributions. Chen *et al.* [7] extends the concept of effective bandwidth to solve the multi-dimensional version of another kind of S-SKP, in which item sizes follow some known distributions but profit variables are constants.

III. FORMULATION

In this section, we formally define the stochastic resource allocation problem and formulate a model for the special case with normal distributed demands and profits.

A. Problem Definition

We are given a set R of r resources (i.e., cameras, bandwidth) and a set T of t tasks (i.e., detection,

tracking). Each resource R_i has an integral capacity c_i . Each task T_j requests A_{ij} of resource R_i and is able to produce a profit V_j if all its demands are satisfied.

In a general knapsack problem, both A_{ij} and V_j are deterministic and fixed. In this paper, however, they are random variables following some known distributions, and all these random variables of demands and profits are assumed to be independent with each other.

We use a binary decision variable x_j to indicate whether T_j is accepted or not. $x_j = 1$ means that T_j is admitted and all its demands are satisfied; otherwise, $x_j = 0$. To decide which tasks should be chosen, the frequency that the admitted set may violate the capacity constraints must be lower than a preset threshold; also, with a given level of confidence, the lower bound of achievable profit should be maximized.

For any single resource R_i , the probability that the total demand of the chosen tasks does not exceed available capacity c_i should be no less than some preset threshold. We define p_d^i as such a capacity overflow threshold denoting the lower bound of this acceptable probability on R_i . Then this set of constraints on each resource can be formulated as follows:

$$\Pr\left(\sum_{j=1}^t A_{ij}x_j \leq c_i\right) \geq p_d^i \quad \forall i \quad (1)$$

In addition, the total profit may vary, and the chance of achieving the greatest result is always too trivial. Thus our goal is to search for the solution which is capable to contribute the highest level of profit with confidence. For example, if the probability of achieving at least a specific level of profit is acceptably high, we call this level meaningful and define the threshold of the acceptable chance as p_v . Given a task set S and p_v , we can calculate the greatest meaningful profit for this set and define it as $\lambda_{p_v}^S$. Then our objective is to search for the greatest achievable $\lambda_{p_v}^S$ among all potential solutions.

Suppose that S_1 and S_2 are two task sets, and p_v is set as 75%. S_1 either returns 50 units or 200 units of profit with probability of 75% and 25%, respectively. S_2 always generates 100 units. The greatest meaningful profit for S_1 ($\lambda_{p_v}^{S_1}$) is 50 because the chance for S_1 to achieve 200 or more is lower than the threshold p_v (i.e. 200 is not meaningful). Since $\lambda_{p_v}^{S_2}$ is 100 and greater than $\lambda_{p_v}^{S_1}$, although S_1 may produce a greater profit twice as much as S_2 , we still prefer S_2 if only one of them can be admitted.

We can describe the relationship above as follows:

$$\Pr\left(\sum_{j=1}^t V_jx_j \geq \lambda_{p_v}^S\right) \geq p_v \quad \forall S \quad (2)$$

Note that (2) is equivalent to:

$$\Pr\left(\sum_{j=1}^t V_jx_j < \lambda_{p_v}^S\right) < 1 - p_v \quad (3)$$

and we define $\hat{p}_v = 1 - p_v$. Then combining (1) and (3), we are able to formulate this problem as follows:

$$\begin{aligned} \max \quad & \lambda_{p_v}^S \\ \text{s.t.} \quad & \Pr\left(\sum_{j=1}^t A_{ij}x_j \leq c_i\right) \geq p_d^i \quad \forall i \\ & \Pr\left(\sum_{j=1}^t V_jx_j < \lambda_{p_v}^S\right) < \hat{p}_v \\ \text{where} \quad & S = \{T_j | x_j = 1\} \end{aligned}$$

B. A Model for Normal Distributed Variables

For the case of the 0/1 knapsack problem with normal distributed demands and profits, define $A_{ij} \sim N(m_{ij}, v_{ij}^2)$ and $V_j \sim N(\mu_j, \sigma_j^2)$, where all m_{ij} , μ_j , v_{ij}^2 , and σ_j^2 are integers. It is easy to prove that any linear combination of A_{ij} 's (i.e., $\sum_j A_{ij}x_j$) also follows a normal distribution with mean of $\sum_j m_{ij}x_j$ and variance of $\sum_j v_{ij}^2x_j$.

As a feature of normal distributions, (1) is equivalent to:

$$\sum_{j=1}^t m_{ij}x_j + \sqrt{\sum_{j=1}^t v_{ij}^2x_j} \cdot Z_{p_d^i} \leq c_i \quad \forall i \quad (4)$$

where $Z_{p_d^i}$ is the inverse CDF $\phi^{-1}(\cdot)$ of standard normal distribution at the point of p_d^i , satisfying $\phi(Z_{p_d^i}) = p_d^i$.

Similarly, $\sum_j V_jx_j \sim N(\sum_j \mu_jx_j, \sum_j \sigma_j^2x_j)$. Since the total profit is a continuous random variable, we can change the equation (3) to

$$\Pr\left(\sum_{j=1}^t V_jx_j \leq \lambda_{p_v}^S\right) = \hat{p}_v$$

which is equivalent to the following form:

$$\sum_{j=1}^t \mu_jx_j + \sqrt{\sum_{j=1}^t \sigma_j^2x_j} \cdot Z_{\hat{p}_v} = \lambda_{p_v}^S \quad (5)$$

where $Z_{\hat{p}_v} = \phi^{-1}(\hat{p}_v)$. It is reasonable to assume that p_v is more than 50%, resulting in $Z_{\hat{p}_v} < 0$, which means that we always require the probability to achieve meaningful profits to be more than 50%.

(5) finds a way of representing our objective for this special case, which can be reformulated as follows:

$$\max \quad \sum_{j=1}^t \mu_jx_j + \sqrt{\sum_{j=1}^t \sigma_j^2x_j} \cdot Z_{\hat{p}_v} \quad (6)$$

while the constraint (4) holds.

IV. ALGORITHMS

In this section, we discuss different algorithms for both scenarios with single or multiple types (dimensions) of resources. For the single-dimensional case, an optimal solution is provided along with two faster heuristics. For the multi-dimensional case, a polynomial time heuristic is proposed.

A. Single-dimensional Case

Note that there is only one type of resource in the single-dimensional case. In this subsection we use C and P_D to denote the corresponding capacity c_1 and overflow threshold p_d^1 defined in the previous section.

1) Optimal Solution (ALG-1)

[17] provides an optimal solution for the unbounded S-SKP (i.e., the number of instances of each task is unlimited), in which only profits are normal distributed. Since our problem focuses on the 0/1 S-SKP in which the demands are also normal variables, we design the optimal solution ALG-1 as follows.

For any given value of $\sum_j \sigma_j^2 x_j$, the objective function (6) is equivalent to maximizing the corresponding $\sum_j \mu_j x_j$. Let $F_k(m, v^2, \sigma^2)$ denote the greatest $\sum_j \mu_j x_j$ among all potential eligible selections, each of which consists of the first k tasks and satisfies the following conditions:

- The sum of demands follows $N(m, v^2)$;
- The variance of the total profit is exactly σ^2 .

Furthermore, if any parameter of $F_k()$ is negative, define $F_k() = -\infty$, which means that we can never find such a solution.

Given a capacity C , we can simply calculate both upper and lower bounds for k , m , v^2 , and σ^2 . k is the index of tasks ranging from 1 to t . $m \in [0, C]$ because $m + v \cdot Z_{P_D} \leq C$ and $Z_{P_D} > 0$. In addition, there would be no more than N^* tasks being admitted, where

$$N^* = \left\lceil \frac{C}{\min_j \{m_j\} + \min_j \{v_j\} \cdot Z_{P_D}} \right\rceil$$

Thus, $v^2 \leq N^* \cdot \max_j \{v_j^2\}$ and $\sigma^2 \leq N^* \cdot \max_j \{\sigma_j^2\}$.

Initialize the value of every $F_k()$ as $-\infty$, and let $F_0(0, 0, 0)$ be 0, which indicates the empty set. Then use DP to calculate every $F_k(I)$ by the following recurrence equation:

$$F_k(I) = \max\{F_{k-1}(I), F_{k-1}(I - I_k) + \mu_k\}$$

where I denotes a 3-tuple input (m, v^2, σ^2) , and I_k is defined as (m_k, v_k^2, σ_k^2) derived from task T_k . Our objective is to maximize the meaningful profit, which could be found by

$$\begin{aligned} \max \quad & F_k(m, v^2, \sigma^2) + \sigma \cdot Z_{p_v} \\ \text{where} \quad & m + v \cdot Z_{P_D} \leq C \end{aligned}$$

ALG-1 tries every combination of all parameters so that it is able to find the optimal $\lambda_{p_v}^S$ and the corresponding set S of admitted tasks. The time complexity of ALG-1 is $O(t \cdot C \cdot \sigma_{max} \cdot v_{max})$, in which σ_{max} and v_{max} are $N^* \cdot \max_j \{\sigma_j^2\}$ and $N^* \cdot \max_j \{v_j^2\}$, respectively.

2) Heuristic Solution (ALG-2)

When C , σ_{max} and v_{max} are bounded by a polynomial in the size of t , ALG-1 is a polynomial time optimal algorithm for our problem setting. However, when they

are not, we prefer a heuristic to quickly search for an alternative suboptimal result when processing the large-scale problem instances.

It is well known that Steinberg's heuristic [15] solves the bounded knapsack problem (i.e., each task could have multiple instances but the number of instances is limited) with deterministic demands and normal profits.

Here for this more complicated case where demands are also normal distributed, we first let ALG-2 perform a transformation on the demands, and then apply Steinberg's algorithm to search for a lower bound of the heuristic solution.

Since the left side of (4) satisfies

$$\begin{aligned} & \sum_j m_j x_j + \sqrt{\sum_j v_j^2 x_j} \cdot Z_{P_D} \\ & \leq \sum_j m_j x_j + \sum_j v_j x_j \cdot Z_{P_D} \\ & = \sum_j (m_j + v_j \cdot Z_{P_D}) \cdot x_j \\ & = \sum_j A_j^* \cdot x_j \end{aligned}$$

where we replace the demand of task T_j , say A_j , by a transformed A_j^* with a fixed value of $m_j + v_j \cdot Z_{P_D}$, we can solve the problem (6) with the following new capacity constraint (7) by applying Steinberg's work:

$$\sum_j A_j^* x_j \leq C \quad (7)$$

The solution to the problem above is a vector $X = (x_1 \cdots x_t)$, which also meets the constraint (4). That is, ALG-2 provides a lower bound of solutions to the question defined by (4) and (6).

$G_k(w)$ is represented by a pair (μ, σ^2) , which indicates that when choosing tasks from T_1 to T_k , if the total transformed demand $\sum_j A_j^*$ does not exceed w , a suboptimal solution can be found, whose total profit follows $N(\mu, \sigma^2)$.

Define $|G_k(w)| = \mu + \sigma \cdot Z_{p_v}$. Initiate $G_0(0)$ as $(0, 0)$, then every $G_k(w)$ can be calculated recursively by

$$G_k(w) = \begin{cases} G_{k-1}(w) & \text{if } w < A_k^* \\ G_{k-1}(w) & \text{else if } V_1 > V_2 \\ G_{k-1}(w^*) + (\mu_k, \sigma_k^2) & \text{otherwise} \end{cases}$$

where

$$\begin{aligned} w^* &= \lfloor w - A_k^* \rfloor \\ V_1 &= |G_{k-1}(w)| \\ V_2 &= |G_{k-1}(w^*) + (\mu_k, \sigma_k^2)| \end{aligned}$$

The value of $|G_t(C)|$ is returned as the maximized meaningful profit $\lambda_{p_v}^{S'}$ of ALG-2, in which the suggested task set S' may not be the same as the optimal solution S calculated by ALG-1. However, the time complexity of ALG-2 is significantly reduced to $O(t \cdot C)$.

3) Improved Heuristic Solution (ALG-3)

Another heuristic, ALG-3, is shown below, which is capable of finding better solutions compared to ALG-2 with a relatively quicker speed than ALG-1.

Suppose that we have already admitted some tasks selected from T_1 to T_k , whose combined demand follows $N(m, v^2)$, and now try to make a decision on T_{k+1} . If we add T_{k+1} to the admitted set, the distribution of total demand should become $N(m + m_{k+1}, v^2 + v_{k+1}^2)$. According to the constraint (4), by adding T_{k+1} as a new task, we request \tilde{A}_{k+1} more units of resource, where

$$\tilde{A}_{k+1} = m_{k+1} + \left(\sqrt{v^2 + v_{k+1}^2} - v \right) \cdot Z_{P_D}$$

Note that \tilde{A}_{k+1} drops into the interval

$$[m_{k+1}, [m_{k+1} + v_{k+1} \cdot Z_{P_D}]]$$

which is decided only by the distribution characteristics of the demand of T_{k+1} . We call this interval, \mathcal{I}_{k+1} , the *relative demand interval* of task T_{k+1} .

Instead of considering T_k 's demand as a constant A_k^* as ALG-2 does, without violating the capacity constraint, our heuristic checks every possible value of \tilde{A}_k in \mathcal{I}_k to search for a better solution. Since $A_k^* \in \mathcal{I}_k$, ALG-3 performs at least as well as ALG-2. Actually, in most cases, ALG-3 successfully achieves much better results, which is shown by our experiments in Section V.

Algorithm 3 Improved Heuristic Solution

```

1: initiate each  $H_k(w) = \mathcal{O}$ 
2:  $OPT_v \leftarrow 0$ ;  $OPT_k \leftarrow 0$ ;  $OPT_w \leftarrow 0$ 
3: for all  $k$  and  $w$  do
4:    $H_k(w) \leftarrow H_{k-1}(w)$ ;  $P_a \leftarrow |P_k(w)|$ 
5:   for all  $\tilde{A}_k \in \mathcal{I}_k$  do
6:     if  $w - \tilde{A}_k \geq 0$  then
7:        $\tilde{w} \leftarrow w - \tilde{A}_k$ 
8:       if  $|D_{k-1}(\tilde{w}) + (m_k, v_k^2)| \leq w$  then
9:          $P_b \leftarrow |P_{k-1}(\tilde{w}) + (\mu_k, \sigma_k^2)|$ 
10:        if  $P_a \leq P_b$  then
11:           $D_k(w) \leftarrow D_{k-1}(\tilde{w}) + (m_k, v_k^2)$ 
12:           $P_k(w) \leftarrow P_{k-1}(\tilde{w}) + (\mu_k, \sigma_k^2)$ 
13:           $S_k(w) \leftarrow S_{k-1}(\tilde{w}) \cup \{T_k\}$ 
14:           $H_k(w) \leftarrow (S_k(w), D_k(w), P_k(w))$ 
15:           $P_a \leftarrow |P_k(w)|$ 
16:        end if
17:      end if
18:    end if
19:  end for
20:  if  $OPT_v \leq P_a$  then
21:     $OPT_v \leftarrow P_a$ ;  $OPT_k \leftarrow k$ ;  $OPT_w \leftarrow w$ 
22:  end if
23: end for
24: return  $OPT_v$  and  $S_{OPT_k}(OPT_w)$ 

```

Let w denote the resource capacity. $S_k(w)$ is a decision vector denoting the best admitted set when only

choosing tasks from T_1 to T_k . The demand and profit of $S_k(w)$ are represented by $D_k(w)$ and $P_k(w)$ in the forms of (m, v^2) and (μ, σ^2) . OPT_v is the maximized meaningful profit which can be achieved by $S_k(w)$ when k and w are OPT_k and OPT_w , respectively. In addition, we define 3-tuple $H_k(w)$ as $(S_k(w), D_k(w), P_k(w))$ and

$$\begin{aligned} |D_k(w)| &= m + v \cdot Z_{P_D} \\ |P_k(w)| &= \mu + \sigma \cdot Z_{\tilde{p}_v} \end{aligned}$$

Unlike the process of getting $G_k(w)$ in ALG-2, to calculate each $H_k(w)$, ALG-3 needs to check through corresponding \mathcal{I}_k , whose size is bounded by $v_k \cdot Z_{P_D}$. Thus the time complexity is changed to $O(t \cdot C \cdot \max\{v_j\})$, which is worse than ALG-2 but still significantly reduced compared to ALG-1.

B. Multi-dimensional Cases

ALG-1, ALG-2 and ALG-3 are all DP-based heuristics, which are not suitable for the multi-dimensional cases due to the limitation known as the ‘‘curse of dimensionality’’.

A polynomial-time heuristic was proposed for solving the deterministic multi-dimensional knapsack problem [18], in which the preference of each task is sorted in terms of *effective gradient* (i.e., profit per unit of aggregate necessary resource). The multi-dimensional SSKP, however, is more complicated.

Given different conditions, one demand variable A_{ij} may not contribute the same to the left side of (4), while the same profit variable V_j could increase the level of $\lambda_{p_v}^S$ differently. Thus we redefine the concept of effective gradient for this type of problem and calculate it in a modified way by using the *relative value* of effective gradient, which is similar to the thinking of relative demand interval mentioned in ALG-3.

Given a binary decision vector X in the form of (x_1, x_2, \dots, x_t) , we define some notations as follows:

$$\begin{aligned} m_X &: (\Sigma_j m_{1j} x_j, \Sigma_j m_{2j} x_j \cdots \Sigma_j m_{rj} x_j) \\ v_X &: (\sqrt{\Sigma_j v_{1j}^2 x_j}, \sqrt{\Sigma_j v_{2j}^2 x_j} \cdots \sqrt{\Sigma_j v_{rj}^2 x_j}) \\ \mu_X &: \Sigma_j \mu_j x_j \\ \sigma_X^2 &: \Sigma_j \sigma_j^2 x_j \\ Z_{p_d} &: \text{diag}(Z_{p_d^1}, Z_{p_d^2} \cdots Z_{p_d^r}) \\ R_{s_X} &: m_X + v_X \cdot Z_{p_d} \\ V_X &: \mu_X + \sigma_X \cdot Z_{\tilde{p}_v} \\ O_X &: (\max\{(R_{s_X})_i - c_i, 0\})_{1 \times r} \\ \bar{O}_X &: (\max\{c_i - (R_{s_X})_i, 0\})_{1 \times r} \end{aligned}$$

where the vector R_{s_X} (i.e., Resultant Size) represents the current value of left side of (4) for each resource, and O_X and \bar{O}_X indicate the level of demand overflow and resource availability, respectively.

Suppose that we have a given decision vector X and would like to remove an admitted task T_k . Denote

that the new decision vector is X' , then the removed profit, ΔV_X^{k-} , would be $V_X - V_{X'}$. Similarly, the vector of removed demand, represented as ΔA_X^{k-} , would be $Rs_X - Rs_{X'}$. Then we can calculate the effective gradient of deleting T_k , say G_X^{k-} , by the following equation:

$$G_X^{k-} = \frac{\Delta V_X^{k-}}{\Delta A_X^{k-} \cdot O_X}$$

where the operation \cdot indicates the dot product of vectors ΔA_X^{k-} and O_X .

We also define notations as follows when adding T_k to an admitted set, in which the decision vector change to X' from X .

$$\begin{aligned} \Delta V_X^{k+} &= V_{X'} - V_X \\ \Delta A_X^{k+} &= Rs_{X'} - Rs_X \\ G_X^{k+} &= \frac{\Delta V_X^{k+}}{\Delta A_X^{k+} \cdot \bar{O}_X} \end{aligned}$$

Algorithm 4 Heuristic for Multidimensional Case

```

1: initiate  $X \leftarrow (1 \dots 1)$ ; update  $Rs_X$ 
2: while any  $(Rs_X)_i > c_i$  do
3:   for all  $j$  do
4:     calculate  $G_X^{j-}$ 
5:   end for
6:    $k \leftarrow \operatorname{argmin}_j \{G_X^{j-}\}$  where  $x_j = 1$ 
7:    $x_k \leftarrow 0$ 
8:   update  $Rs_X$ 
9: end while
10:  $k \leftarrow 0$ 
11: while all  $(Rs_X)_i < c_i$  do
12:   for all  $j$  do
13:     if  $x_j = 0$  and adding  $T_j$  won't violate (4) then
14:       calculate  $G_X^{j+}$ 
15:     else
16:        $G_X^{j+} = -1$ 
17:     end if
18:   end for
19:    $k \leftarrow \operatorname{argmax}_j \{G_X^{j+}\}$  where  $G_X^{j+} \geq 0$ 
20:   if  $k \neq 0$  then
21:      $x_k \leftarrow 1$ 
22:     update  $Rs_X$ 
23:   else
24:     break
25:   end if
26: end while
27: return  $V_X$  and  $X$ 

```

The procedure of ALG-4 follows two steps. First of all, let X be $(1 \dots 1)$, indicating that all tasks are temporarily selected. Initiate $Rs_X = (\sum_j A_{1j} \dots \sum_j A_{rj})$, and define a feasible zone $\mathcal{F} = \{(d_1 \dots d_r) | d_i \in [0, c_i]\}$. Before Rs_X falls into \mathcal{F} , we repeatedly remove one task from the admitted set in the increasing order of

effective gradient, which is the ratio of the profit shifted and projected length of demands on the direction from Rs_X to \mathcal{F} . After Rs_X enters \mathcal{F} , we try to add some dropped tasks back in the decreasing order of effective gradient, as long as there are still enough resources left.

ALG-4 has a time complexity of $O(t^2 \cdot r)$. We compare the results of ALG-4 with that of the enumeration method in the experimental section, showing good performance of ALG-4 on both results and running time.

V. NUMERICAL EXPERIMENTS

In this section, we explain our numerical experiments and present the performance of our algorithms running under different settings.

A. Experiment Setup

Recall that $U(a, b)$ denotes a uniform distribution on interval $[a, b]$. In our experiments, all parameters are integers and randomly drawn. Every demand A_{ij} of task T_j on a resource R_i independently follows a normal distribution $N(m_{ij}, v_{ij}^2)$, and every admitted T_j independently generates a profit $V_j \sim N(\mu_j, \sigma_j^2)$, where each parameter independently follows $U(1, 25)$.

In each series of experiments, we vary the values of t , r , p_v and p_d to check how they affect the results and computational effectiveness. 100 problem instances are generated for each particular combination. Each time we randomly assign an integer value to the capacity of resource R_i , which is greater than $\max_j \{m_{ij}\} + \max_j \{v_{ij}\} \cdot Z_{p_d^i}$ (C_{min}^i) to guarantee that at least one task can be admitted. In addition, a capacity greater than the value of $\sum_j m_{ij} + \sqrt{\sum_j v_{ij}^2} \cdot Z_{p_d^i}$, say C_{max}^i , is enough to meet the combined demands of all tasks. Therefore, we draw c_i from the interval $[C_{min}^i, C_{max}^i]$.

Every solution suggests an admitted task set S , the corresponding $\lambda_{p_v}^S$ of which, for a given p_v , indicates that S has a probability p_v to achieve a profit greater than $\lambda_{p_v}^S$. For the same single-dimensional problem, let the solutions of ALG-1, ALG-2 and ALG-3 be S_1 , S_2 and S_3 , respectively. We separately compare $\lambda_{p_v}^{S_3}$ to $\lambda_{p_v}^{S_1}$ (the optimum) and $\lambda_{p_v}^{S_2}$ (a lower bound of the suboptimal solution), to see how well our heuristic works. Similarly, for any multi-dimensional case, the results of ALG-4 ($\lambda_{p_v}^{S_4}$) and the enumeration method (eNUM) ($\lambda_{p_v}^{S_e}$) are compared as the latter is always maximized.

The computational effectiveness is also tested, by which we show that although ALG-1 and eNUM can find the best solutions, ALG-3 and ALG-4 are potentially better heuristics because of their ability to return very good suboptimal results in most cases while accelerating running speed by thousands of times.

B. Single-dimensional Cases

In the experiments for the single-dimensional cases, the values of p_v (the probability to achieve the greatest meaningful profits), p_d (the capacity overflow threshold)

and t (the number of tasks) are changed individually, to compare the results and running time of ALG-1, ALG-2 and ALG-3. Theoretically, ALG-1 spends a longer time to compute the optimal results, ALG-2 returns suboptimal results with the fastest speed, and ALG-3 is able to find better solutions than ALG-2 and returns within a shorter time than ALG-1.

1) $t = 15$, $p_d = 85\%$ and varied p_v :

Here t and p_d are fixed, while p_v is chosen from 55% and 95%. 500 problem instances are simulated in this series of experiments, and the results are illustrated in Fig. 1, where histograms display the PDF's of the ratio of profits achieved by ALG-2/ALG-3 ($\lambda_{p_v}^{S_2}/\lambda_{p_v}^{S_3}$) versus the actual optimum $\lambda_{p_v}^{S_1}$ calculated by ALG-1.

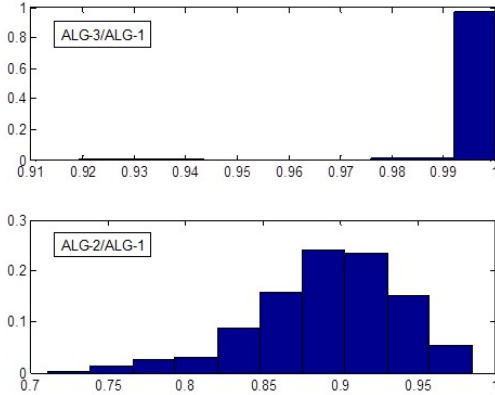


Fig. 1: PDF of $\lambda_{p_v}^S$ ratio with varied p_v (single resource)

The top figure presents the PDF of the ratio between $\lambda_{p_v}^{S_3}$ and $\lambda_{p_v}^{S_1}$. It shows that ALG-3 achieves the same results as ALG-1 about 95% of the time. The bottom figure compares ALG-2 with ALG-1, in which the former fails to achieve more than 90% of the corresponding $\lambda_{p_v}^{S_1}$ more than 50% of the time. As a result, we can see that ALG-3 closely matches the optimal solutions and achieves better profits than ALG-2.

2) $t = 15$, $p_v = 85\%$ and varied p_d :

In this set of experiments, p_d is changed from 55% to 95% with constant t and p_v , while another 500 problems are tested.

Likewise, the PDF's of $\lambda_{p_v}^{S_3}/\lambda_{p_v}^{S_1}$ and $\lambda_{p_v}^{S_2}/\lambda_{p_v}^{S_1}$ are shown in Fig. 2. The histograms show that ALG-3 works as well as ALG-1 about 92% of the time and ALG-2 achieves less than 90% of $\lambda_{p_v}^{S_1}$ almost 40% of the time. Again, ALG-3 outperforms ALG-2.

Fig. 3 shows the ratio between $\lambda_{p_v}^{S_3}$ and $\lambda_{p_v}^{S_2}$ based on given p_d 's, in which we can see an ascending trend of superiority of ALG-3 over ALG-2 when p_d is increasing. That is, the closer p_d approaches 1, which indicates that the capacity overflow constraint is being tightened, the better ALG-3 performs compared to ALG-2.

3) $p_d = 85\%$, $p_v = 85\%$ and varied t :

In this set of experiments of 500 instances, t is drawn from 5 to 25, while p_d and p_v are both fixed at 85%.

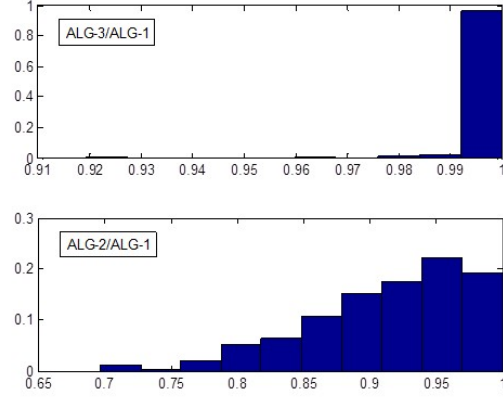


Fig. 2: PDF of $\lambda_{p_v}^S$ ratio with varied p_d (single resource)

The PDF's in Fig. 4 illustrate that, compared to ALG-2, ALG-3 always works well in searching for the optimized solutions no matter how t changes.

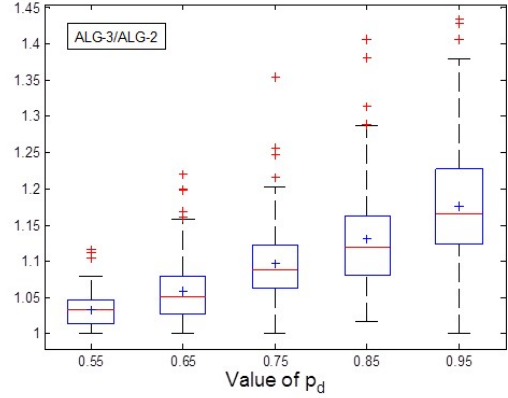


Fig. 3: $\lambda_{p_v}^{S_3}/\lambda_{p_v}^{S_2}$ with varied p_d (single resource)

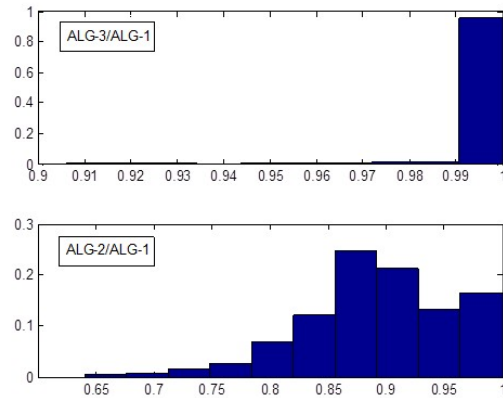


Fig. 4: PDF of $\lambda_{p_v}^S$ ratio with varied t (single resource)

Fig. 5 includes some boxplots, indicating that the average percentage of outperformance of ALG-3 over ALG-2 is stable around 10%, while the variance of this increased performance slightly decreases with the increase of t .

4) Computational Effectiveness:

The running time comparisons of all three series of experiments are shown in Fig. 6. For the same problem instance, the ratio of running time between ALG-1 (t_1) and ALG-3 (t_3) is represented in the form of $\log(t_1/t_3)$. We can see that although ALG-1 returns optimized results, ALG-3 may be a better choice because it is often able to optimize solutions with an average speed up of a factor of 10^3 .

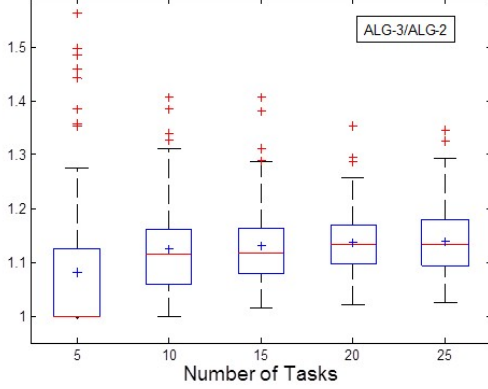


Fig. 5: $\lambda_{p_v}^{S_3} / \lambda_{p_v}^{S_2}$ with varied t (single resource)

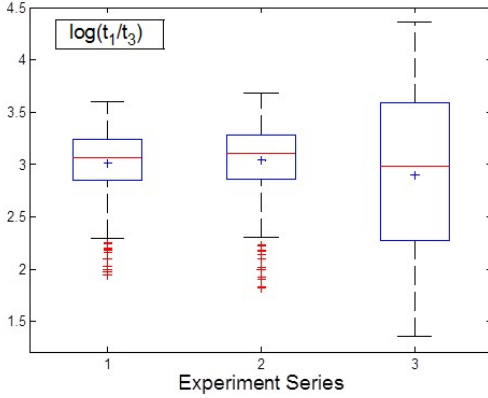


Fig. 6: Ratio of running time (single resource)

C. Multi-dimensional Cases

For the multi-dimensional cases, all four parameters, t , r , p_d and p_v , are individually changed. ALG-4 is a linear approximate heuristic searching for suboptimal solutions and compared with eNUM. Note that eNUM is not scalable but always finds the best solutions, thus we use it as a basis to show how well ALG-4 performs.

Fig. 7 illustrates the results of three series of experiments with different combinations of parameters. We can see in each histogram that the returns of ALG-4 ($\lambda_{p_v}^{S_4}$) mostly range over 95% of the corresponding optimum ($\lambda_{p_v}^{S_e}$) regardless of the combinations of parameters.

The corresponding running time of eNUM (t_e) and ALG-4 (t_4) for the problem instances shown in Fig. 7 are shown in Fig. 8. The first set of results ranges widely

from about -1 to 4 because when the number of tasks is small, eNUM can run much faster than ALG-4 but slows down quickly with the increase of t . For the other two experiment, a nearly 10^3 times speedup is achieved by ALG-4, which shows its computational effectiveness for our problem setting in the multi-dimensional cases.

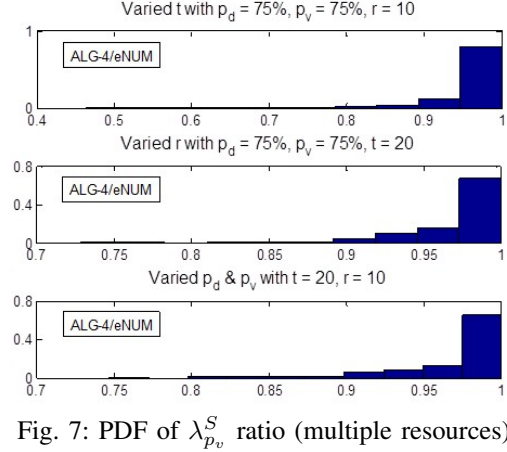


Fig. 7: PDF of $\lambda_{p_v}^S$ ratio (multiple resources)

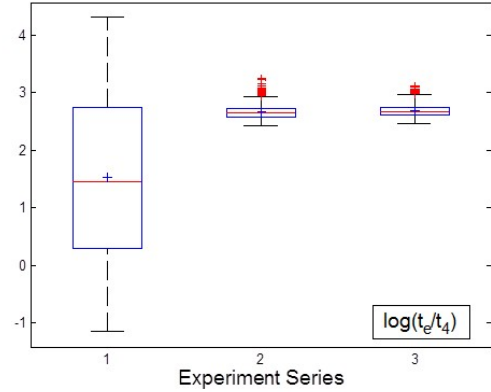


Fig. 8: Ratio of running time (multiple resources)

D. Poisson and Binomial Distributed Cases

Previous experiments in this section only focus on those S-SKP problem instances with normal distributed demands and profits. Here we consider the Poisson and Binomial distributed variables as extensions. Since both of these two distributions have corresponding normal approximation methods, we try to normalize every random variable first, and then apply our heuristics (i.e., ALG-3 and ALG-4) to the modified problems to check if they are still capable of performing well in both single- and multi-dimensional cases. Again, the optimal solutions returned by eNUM are set as the basis to be compared with our algorithms.

According to the Central Limit Theorem, the limit form of a Poisson distribution $\mathcal{P}(\lambda)$ is a normal distribution in the form of $N(\lambda, \lambda)$, and when λ is sufficiently large (i.e., $\lambda \geq 20$), the error from approximation would

be relatively small. Similarly, a Binomial distribution $\mathcal{B}(n, p)$ also can be normalized as $N(np, np(1-p))$, which is usually appropriate if the constraints $np \geq 10$ and $n(1-p) \geq 10$ hold.

In our experiments, each series consists of 500 problem instances, and all parameters, λ , n and p , are independently drawn from $U(1, 25)$, $U(1, 25)$ and $U(0, 1)$, respectively. Although this setting may lead to some larger error caused by the normal approximation of a single random variable, the overall performance shown in Fig. 9 is still good.

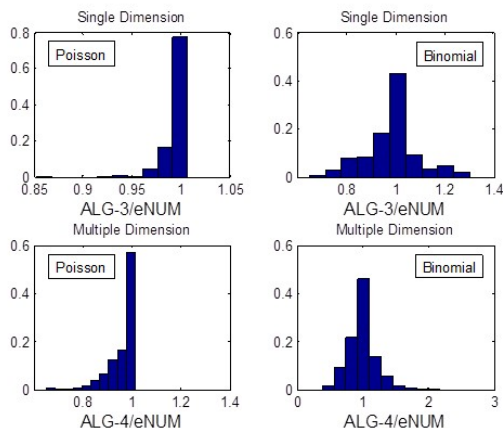


Fig. 9: PDF of $\lambda_{p_v}^S$ ratio for normal approximation

When there is only one type of resource, most solutions suggested by the approximation methods are able to achieve more than 90% of the optimal profits calculated by eNUM. For the Poisson distribution, the results are very close to the optimal. The ratio between heuristic and optimal results ranges wider when multiple resources are present, but is still distributed tightly around the area of 100%. Note that due to the error of the normal approximation, the approximation method may return a solution that violates the capacity constraints and produces greater profit than the optimal, as shown in the two histograms on the right.

VI. CONCLUSION

In this paper, we study the resource allocation problem with stochastic demands and profits. We model this problem as a variant of the stochastic knapsack problem and solve a specific case with a normal distribution. Via numerical experiments, we found that, with relatively faster speed, our heuristics always provide satisfactory solutions compared with the optimal results, and keep working well as the number of tasks increases, the capacity constraints become tighter/looser, and the distributions of demands and profits are extended to Poisson and Binomial.

In the future work, utilizing the concept of *relative values* of random variables, we plan to find a more general solution which is feasible for more extensions of stochastic knapsack problems with other types of

distributions. In addition, we also plan to study different ways in which to provide feedback regarding the bottleneck of resources. It would be, for example, useful to highlight not only which tasks should be selected to yield the greatest profit, but also which resources are the bottleneck that prevent some specific tasks being selected.

REFERENCES

- [1] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer, 2004.
- [2] S. Martello and P. Toth, *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.
- [3] H. Kellerer and U. Pferschy, "A new fully polynomial time approximation scheme for the knapsack problem," *Journal of Combinatorial Optimization*, vol. 3, pp. 59–71, 1999.
- [4] A. K. Chandra, D. S. Hirschberg, and C. K. Wong, "Approximate algorithms for some generalized knapsack problems," *Theor. Comput. Sci.*, vol. 3, no. 3, pp. 293–304, 1976.
- [5] A. Marchetti-Spaccamela and C. Vercellis, "Stochastic on-line knapsack problems," *Math. Program.*, vol. 68, no. 1, pp. 73–104, Jan. 1995.
- [6] B. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programming: Theory and applications," *Journal of Optimization Theory and Applications*, vol. 142, pp. 399–416, 2009.
- [7] F. Chen, T. La Porta, and M. Srivastava, "Resource allocation with stochastic demands," in *Distributed Computing in Sensor Systems (DCOSS), 2012 IEEE 8th International Conference on*, may 2012, pp. 257–264.
- [8] T. İlhan, S. M. R. Iravani, and M. S. Daskin, "The adaptive knapsack problem with stochastic rewards," *Operations Research*, vol. 59, no. 1, pp. 242–248, Jan. 2011.
- [9] M. I. Henig, "Risk criteria in a stochastic knapsack problem," *Operations Research*, vol. 38, no. 5, pp. 820–825, Sep. 1990.
- [10] F. Kelly, S. Zachary, and I. Ziedins, "Notes on effective bandwidth," *Stochastic networks: theory and applications*, vol. 4, pp. 141–168, 1996.
- [11] B. C. Dean, M. X. Goemans, and J. Vondrák, "Approximating the stochastic knapsack problem: The benefit of adaptivity," *Mathematics of Operations Research*, vol. 33, no. 4, pp. 945–964, 2008.
- [12] A. Bhalgat, A. Goel, and S. Khanna, "Improved approximation results for stochastic knapsack problems," in *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '11. SIAM, 2011, pp. 1647–1665.
- [13] B. C. Dean, M. X. Goemans, and J. Vondrák, "Adaptivity and approximation for stochastic packing problems," in *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2005, pp. 395–404.
- [14] M. Johnson, H. Rowaihy, D. Pizzocaro, A. Bar-Noy, S. Chalmers, T. La Porta, and A. Preece, "Sensor-mission assignment in constrained environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 11, pp. 1692–1705, Nov. 2010.
- [15] E. Steinberg and M. S. Parks, "A preference order dynamic program for a knapsack problem with stochastic rewards," *The Journal of the Operational Research Society*, vol. 30, no. 2, pp. pp. 141–147, 1979.
- [16] R. L. Carraway, R. L. Schmidt, and L. R. Weatherford, "An algorithm for maximizing target achievement in the stochastic knapsack problem with normal returns," *Naval Research Logistics (NRL)*, vol. 40, no. 2, pp. 161–173, 1993.
- [17] D. P. Morton and R. K. Wood, "On a stochastic knapsack problem and generalizations," *Advances in computational and stochastic optimization, logic programming, and heuristic search*, pp. 149–168, 1998.
- [18] Y. Toyoda, "A simplified algorithm for obtaining approximate solutions to zero-one programming problems," *Management Science*, vol. 21, no. 12, pp. pp. 1417–1427, 1975.