2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Algorithm for Optimal Chance Constrained Knapsack Problem with Applications to Multi-robot Teaming

Fan Yang[1] and Nilanjan Chakraborty[2]

***Abstract*— Motivated by applications in multirobot team selection, in this paper, we present a novel algorithm for computing optimal solution of chance-constrained 0-1 knapsack problem. In this variation of the knapsack problem, the objective function is deterministic but the weights of the items are stochastic and therefore the knapsack constraint is stochastic. We convert the chance-constrained knapsack problem to a two-dimensional discrete optimization problem on the variance-mean plane, where each point on the plane can be identified with an assignment of items to the knapsack. By exploiting the geometry of the non-convex feasible region of the chance-constrained knapsack problem in the variance-mean plane, we present a novel deterministic technique to find an optimal solution by solving a sequence of deterministic knapsack problems (called risk-averse knapsack problem). We apply our algorithm to a multirobot team selection problem to cover a given route, where the length of the route is much larger than the length each individual robot can fly and the length that an individual robot can fly is a random variable (with known mean and variance). We present simulation results on randomly generated data to demonstrate that our approach is scalable with both the number of robots and increasing uncertainty of the distance an individual robot can travel.**

I. Introduction

The knapsack problem is a fundamental problem in combinatorial optimization that has multiple applications in task allocation and team formation in multi-robot systems, especially when we consider limited battery life of the robots. In this paper, we consider multirobot team formation problems, where robots with limited battery life (and therefore a constraint on the distance they can travel) have to cover together a given distance in the course of their task execution. The distance the robots have to cover is assumed to be much larger than the distance an individual robot can cover. Furthermore, the distance a robot can cover is a stochastic variable since it may depend on variables that are unknown at the team formation time. Such situations arise quite naturally in many applications including point-to-point material transfer, patrolling, and persistent surveillance.

For example, consider a perimeter patrolling application (see Figure 1) by a team of quadrotors, in which the robots with limited battery life have to move along a route of given length (possibly multiple times). The total distance that has to be covered is much larger than the distance an individual robot can fly. Furthermore, the distance an individual robot can travel is uncertain because it depends on uncertain environmental variables (like wind speed for unmanned aerial vehicles). Thus, the distance a robot can travel can vary from one run of the robot to the next depending on the environmental conditions. There is operating cost for each robot. The total cost of covering the route is a sum of individual costs of robots. Our goal is to select a team of robots of minimum size or minimum total operating cost from a group of heterogeneous robots that covers the route with high probability (specified *a priori*), irrespective of the realization of the random distance that a robot can travel. Such a solution has the advantage that the selected team would be able to perform the patrolling task despite high variability in the environmental conditions.

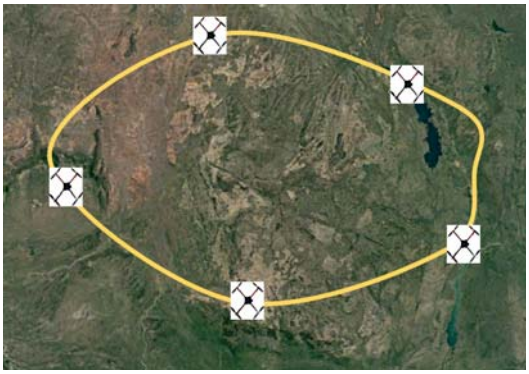

Fig. 1. Figure shows a patrolling route for perimeter surveillance with a team of quadrotors. The goal is to compute the number of robots in the team so that the operating cost of patrolling (or number of robots in the team) is minimized.

The deterministic version of our problem where the travel distances are known constants can be formulated as a 0-1 knapsack problem. The classical 0-1 knapsack problem can be stated as follows: *Given a set of items, each with a weight and a value, determine the items to include in a knapsack so that the total weight is less than or equal to the weight carrying capacity of the knapsack and the total value is as large as possible [7].* To see the correspondence of our problem to the knapsack problem, note that the total length to be traveled by the robots corresponds to the capacity of the knapsack and the weights of the items correspond to the distance an individual robot can fly. At first glance, the deterministic version of our problem seems to be different from the classical knapsack problem since we are minimizing the total cost. However, the knapsack problem we are considering is equivalent to the classical 0-1 maximization knapsack problem. In fact, we can think

This work was supported in part by AFOSR award FA9550-15-1-0442. [1]Fan Yang and [2]Nilanjan Chakraborty are with the Mechanical Engineering Department at Stony Brook University. Email: fan.yang.3@stonybrook.edu, nilanjan.chakraborty@stonybrook.edu

of the minimization problem as the following maximization problem: find a subset of robots such that the total cost of unchosen robots is maximized and the total travel length of the robots is less than the total length of all robots in the given set minus the length of the given route. Therefore the algorithms that solve classical 0-1 knapsack problem can also be used to solve the deterministic version of our problem. There are many methods to solve the knapsack problem such as dynamic programming [21], branch and bound method [12] and other methods that combine both dynamic programming and branch and bound [16], [10], [11]. Although solving knapsack problem is NP-hard, there is a fully polynomial time approximation scheme [21].

Contributions: In this paper, we present a novel algorithm that solves 0-1 knapsack problem with chance constraint. In [4], the authors consider a stochastic knapsack problem similar to our setting and provide a polynomial time approximation scheme (PTAS) by using a parametric linear program. However, we present a more efficient method, where we solve the chance constrained problem by repeatedly solving a deterministic knapsack problem called the risk-averse knapsack problem. By analyzing the feasible region of both chance constrained and risk averse knapsack problems on variance-mean plane, we prove that there exists a risk-averse knapsack problem such that the optimal solution of chance constrained knapsack problem is also the optimal solution of risk-averse knapsack problem. We use this insight to develop an iterative algorithm where we solve the chance constrained problem by repeatedly solving a sequence of risk-averse knapsack problem. The key aspect of our algorithm is that we maintain a probabilistic guarantee irrespective of the realization of the random variables (the lengths the robots could move). We present simulation results on randomly generated data which show that our algorithm works efficiently.

II. Related Work

Chance constrained optimization problems are a class of stochastic optimization problem [19], [3]. They are usually hard to solve (except for some special cases like linear optimization [19], minimum spanning tree [6]). Chance-constrained shortest path problems have been studied in [15] and the algorithm has been extended to a class of chance constrained optimization problems, where the objective function is quasi-convex [14]. In our previous work [22], we presented an algorithm for solving the chance-constrained linear assignment problem. In [22], we demonstrate the connection between chance constrained problem and risk-averse linear assignment problem and show that the optimal solution is obtained by using a one-dimensional search on risk-averse parameter. Within the robotics literature on task allocation or team formation, problems with stochastic constraints have been studied in [17], [18], [13].

There are different stochastic variations of the classical 0-1 knapsack problem that have been studied in the extant literature. In [1], [5], [20], the authors have studied the stochastic knapsack problem with deterministic weights and random costs whereas in our problem we have deterministic costs and random weights. In [2], the authors compute a solution policy that optimize the expected total values. Optimizing expected values provide no performance guarantees on a particular realization of the random variables. We want to develop methods that ensures the constraints are satisfied with a high probability irrespective of the realization of the random weights. An algorithm is designed to obtain good solutions to the chance-constrained problem in [8], by running a sequence of robust problems. The algorithm provides an optimal solution when the costs are identical or the uncertain weights present all the same characteristic. In this paper, our method computes the optimal solution in more general situation. In [4], the authors consider a stochastic knapsack problem similar to our setting and provide a polynomial time approximation scheme (PTAS) by using a parametric linear programming reformulation. Our solution to the chance-constrained problem is based on a geometric interpretation of the problem on variance-mean plane. Our method finds the optimal solution of chance-constrained problem by solving a sequence of a deterministic knapsack problems called the risk-averse knapsack problems.

III. Chance Constrained Knapsack Problem

Let L be the length of the closed curve (or a route) that a team of robots have to cover. We have a collection of heterogeneous robots that have different battery life and they can fly for different lengths. Let ℓ_i be the distance that robot i can fly. Each robot has a different operating and maintenance cost denoted by c_i. The variable ℓ_i is assumed to be a Gaussian random variable with mean μ_i and variance σ_i^2, i.e., $\ell_i \sim \mathcal{N}(\mu_i, \sigma_i^2), i = 1, ..., n$. Our goal is to find a set of robots from the collection of n robots that can cover the total length L with probability p (where $0 \leq p \leq 1$) while minimizing the total cost. Let f_i be an integer variable that takes the value 1 if robot i is part of the team and 0 otherwise. The integer program formulation of our problem is as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i f_i \\ \text{s.t.} \quad & \mathbb{P}\left(\sum_{i=1}^{n} \ell_i f_i \geq L\right) \geq p \\ & f_i \in \{0,1\}, \quad \forall i = 1, \ldots, n \end{aligned} \tag{1}$$

The formulation above is a stochastic variation of the minimization version of 0-1 knapsack problem, which we call the chance constrained knapsack problem (CC-KAP). In the deterministic knapsack problem, the probabilistic constraint in (1) constraint is replaced by the deterministic constraint $\sum_{i=1}^{n} \ell_i f_i \geq L$. The minimization version of the knapsack (which we are generalizing with the stochastic constraint) is equivalent to the maximization version [21]. Intuitively, minimizing the cost of robots in the team is equivalent to maximizing the cost of robots that are not chosen in the team. Note that if we set the costs to be identical (say 1), we find the robot team with minimum number of robots such that the route is covered by the team with high probability, despite

the uncertainty in the length that each robot can fly. The formulation in (1) cannot be solved easily and we reformulate it by rewriting the chance constraints in an equivalent form.

Lemma 1: The CC-KAP problem in (1) with a given probability p is equivalent to the following formulation

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i f_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} \mu_i f_i - C\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i} \geq L \\ & f_i \in \{0,1\}, \quad \forall i = 1,\ldots,n \end{aligned} \tag{2}$$

where $C = \Phi^{-1}(p)$ with Φ representing the cumulative distribution function (cdf) of a standard normal distribution with zero mean and unit variance.

Proof: In the first constraint of formulation (1), $\sum_{i=1}^{n} \ell_i f_i$ is a random variable that is distributed normally with mean $\sum_{i=1}^{n} \mu_i f_i$ and variance $\sum_{i=1}^{n} \sigma_i^2 f_i$. Standardizing this normally distributed random variable, we have

$$\mathbb{P}\left(\sum_{i=1}^{n} \ell_i f_i \geq L\right) \geq p$$

$$\implies \mathbb{P}\left(\frac{\sum_{i=1}^{n} \ell_i f_i - \sum_{i=1}^{n} \mu_i f_i}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}} \geq \frac{L - \sum_{i=1}^{n} \mu_i f_i}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}}\right) \geq p$$

Now the left hand side of the inequality in the bracket is a random variable with standard normal distribution. Therefore

$$1 - \Phi\left(\frac{L - \sum_{i=1}^{n} \mu_i f_i}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}}\right) \geq p$$

Since $\Phi(-x) = 1 - \Phi(x)$. Therefore

$$\Phi\left(\frac{\sum_{i=1}^{n} \mu_i f_i - L}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}}\right) \geq p$$

$$\implies \left(\frac{\sum_{i=1}^{n} \mu_i f_i - L}{\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}}\right) \geq \Phi^{-1}(p)$$

$$\sum_{i=1}^{n} \mu_i f_i - \Phi^{-1}(p)\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i} \geq L$$

Let $C = \Phi^{-1}(p)$. Thus, we obtain the first constraint in Equation (2),

$$\sum_{i=1}^{n} \mu_i f_i - C\sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i} \geq L.$$

■

If we relax f_i, the problem in (2) is a second order cone program with integrality gap $\Omega(\sqrt{n})$ [4]. In [4], the authors converted the problem in Equation (2) to a parametric linear program and presented an algorithm that for $\epsilon > 0$ gives a $1 - 3\epsilon$ approximate solution with running time $O\left(\frac{1}{\epsilon^2} n^{\frac{1}{\epsilon}}\right)$. We present an alternate parametric formulation, where different choices of the parameter leads to different knapsack problems. In the discussion below we will refer to both (1) and (2) as chance constrained knapsack problem (CC-KAP), which is a chance constrained integer optimization problem and is hard to solve in general. In this paper, instead of solving CC-KAP directly, we show that the solution to CC-KAP can be obtained by solving a number of deterministic knapsack problems (given below), which we call risk-averse knapsack problem (RA-KAP)

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i f_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} \mu_i f_i - \lambda \sum_{i=1}^{n} \sigma_i^2 f_i \geq L' \\ & f_i \in \{0,1\}, \quad \forall i = 1,\ldots,n \end{aligned} \tag{3}$$

Here λ is the risk-averse parameter that performs a weighted combination of the mean and variance of the travel lengths of each robot. The parameter L' is the constraint for the total length in RA-KAP. Consider the situation when $\lambda = 0$ and $L' = L$, the problem becomes the classical 0-1 knapsack problem. Our solution strategy is to iteratively search over λ and L' to find the appropriate λ and L' such that the solution to the RA-KAP gives a solution to the CC-KAP. We justify this in the next section.

IV. Geometric Interpretation

In this section, we present a geometric interpretation of the CC-KAP on the variance-mean plane in which the horizontal axis is the variance and the vertical axis is the mean (see Figure 2). The CC-KAP is an integer optimization problem in which any solution is a vector of binary decision variables f_i. Given any particular solution $s = [f_1, ..., f_n]$, we can identify this solution with a point on the variance-mean plane. The y-coordinate of this point is the sum of means for all travel distance of robots chosen in the solution, $\sum_i^n \mu_i f_i$, and the x-coordinate is the sum of variances, $\sum_i^n \sigma_i^2 f_i$. The coordinate of this point related to solution s is denoted by $(\sigma^2(s),\ \mu(s))$. Thus the space of all possible robot teams can be identified with points in the variance-mean plane (however, we do not construct this explicitly because the number of such points will be exponential in the number of robots). Moreover, we can find the feasible region of solution for the CC-KAP based on the chance constraint in Formulation 2. As shown in Figure 2, the feasible region for CC-KAP is the space above the parabola in the first quadrant on variance-mean plane. Since the constraint in the RA-KAP is a linear inequality of σ^2 and μ, the feasible region for RA-KAP is the space above the line whose slope is equal to risk-averse parameter λ and y-intercept is equal to the length of the route L'. Based on this geometric viewpoint, we present the following lemmas:

Lemma 2: The optimal solution of RA-KAP that satisfies the chance constraint provides an upper bound of optimal solution of CC-KAP.

Proof: Since the optimal solution of RA-KAP satisfies the chance constraint, the corresponding point on variance-mean plane must be in the intersection of feasible regions of

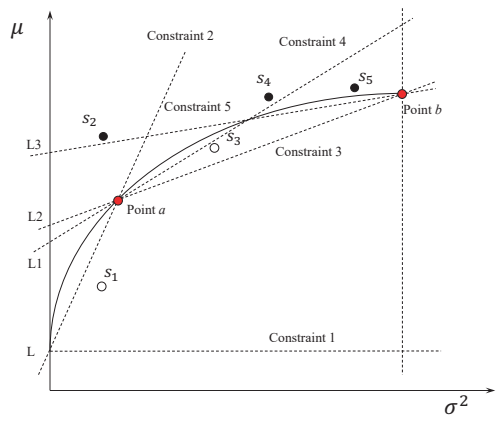

Fig. 2. The geometric interpretation of chance-constrained knapsack problem. Any solution is related to a point on variance-mean plane. The feasible region for CC-KAP is the space above the parabola while the feasible region for RA-KAP with given λ and L is the space above the line whose slope is equal to λ and y-intercept is equal to L.

CC-KAP and RA-KAP. If the optimal solution of CC-KAP is in this intersection, then the optimal objective function value of CC-KAP and RA-KAP are same. If the optimal solution of CC-KAP is not in the intersection, it implies that the optimal objective value of RA-KAP is greater than the optimal objective value of CC-KAP. Therefore the optimal objective function value of RA-KAP must be greater than or equal to the optimal objective function value of CC-KAP. ■

Lemma 3: If the constraint of RA-KAP is tangent to the parabola of chance constraint, the optimal solution of this particular RA-KAP must be a feasible solution of CC-KAP.

Proof: When the linear constraint of RA-KAP is tangent to the parabola, the feasible region of RA-KAP is the subset of feasible region of CC-KAP. Therefore points related to all feasible solutions including the optimal solution of RA-KAP are in the feasible region of CC-KAP. ■

Lemma 4: There exists a RA-KAP such that the optimal solution of CC-KAP is also the optimal solution of the RA-KAP.

Proof: As it is stated in the proof of Lemma 3, the feasible region of RA-KAP in which the constraint is tangent to the parabola is a subset of feasible region of CC-KAP. There always exists a tangent to the parabola (i.e., a pair $(\bar{\lambda}, \bar{L}')$), such that the half-space defined by this pair contains the point corresponding to the optimal solution of CC-KAP (say $s*$). Thus s^* is also the optimal solution of RA-KAP. In fact the lemma also works for the RA-KAP in which the constraint is not tangent to the parabola. For example, if we move the previous tangent by rotating and translating, the optimal solution of RA-KAP may still be the optimal solution of CC-KAP.

■

Lemma 4 implies that the key to solving the CC-KAP is to solve the RA-KAP with particular slope $\bar{\lambda}$ and y-intercept $\bar{L}'$ whose optimal solution satisfies the chance constraint. However Lemma 2 tells that solving the RA-KAP for a particular choice of λ and L' does not mean we have obtained the optimal solution, probably upper bound instead. Therefore we need to design a method to iterate over different values of λ and L' methodically so as to cover the whole area of the feasible region of CC-KAP. The optimal solution would be the solution with the smallest total cost among those upper bounds. Our algorithm provides a systematic approach to perform this search.

V. Algorithm

In this section, we present our algorithm to solve the CC-KAP. The input for our Algorithm 1 is the length of the route, L, the confidence-level that the robots should finish the tour of the route, p, the mean (μ_i) and variance (σ_i^2) of travel distance for each robot. The algorithm outputs the robot team that can complete the route following task with probability p and minimum total cost.

Algorithm 1 Algorithm to solve CC-KAP

Input: $L,\ p,\ \mu_i,\ \sigma_i^2\ \forall i = 1, \ldots, n.$
Output: $v^*,\ s^* = \{f_1, \ldots, f_n\}$.

1: Set $k = 1$, $\lambda_k = 0$, $L_k = L$.
2: Solve RA-KAP with λ_k and L_k. $\mathbb{S} \leftarrow s_k, \mathbb{V} \leftarrow v_k$.
3: **while** the optimal solution of RA-KAP s_k does not satisfy the chance constraint **do**
4: let $k = k + 1$, $\lambda_k = C/\sigma_{k-1}$ where $C = \Phi^{-1}(p)$.
5: Solve RA-KAP with λ_k and L_k. $\mathbb{S} \leftarrow s_k \cup \mathbb{S}$, $\mathbb{V} \leftarrow v_k \cup \mathbb{V}$.
6: **end while**
7: Compute point a and point b on the parabola.
8: Compute $\hat{\lambda}_{ab}, \hat{L}_{ab}$ by connecting point a and point b. $\Lambda \leftarrow \hat{\lambda}_{ab}$, $\mathbb{L} \leftarrow \hat{L}_{ab}$.
9: **while** $\Lambda \neq \emptyset$ **do**
10: **for** $i = 1$ to $|\Lambda|$ **do**
11: Solve RA-KAP with $\hat{\lambda}_i$ and $\hat{L}_i$.
12: **if** $\hat{s}_i$ does not satisfy the chance constraint **then**
13: Compute $\hat{\lambda}_{i,1}, \hat{\lambda}_{i,2}, \hat{L}_{i,1}, \hat{L}_{i,2}$.
14: $\Lambda' \leftarrow \hat{\lambda}_{i,1} \cup \hat{\lambda}_{i,2} \cup \Lambda'$ and $\mathbb{L}' \leftarrow \hat{L}_{i,1} \cup \hat{L}_{i,2} \cup \mathbb{L}'$.
15: **else**
16: $\mathbb{S} \leftarrow \hat{s}_i \cup \mathbb{S},\ \mathbb{V} \leftarrow \hat{v}_i \cup \mathbb{V}$
17: **end if**
18: **end for**
19: $\Lambda \leftarrow \Lambda'$, $\mathbb{L} \leftarrow \mathbb{L}'$ and $\Lambda', \mathbb{L}' \leftarrow \emptyset$.
20: **end while**
21: $v^* = \max_{v_i \in \mathbb{V}} v_i,\ s^* = \max_{v_i \in \mathbb{V}} s_i$

In Algorithm 1, first, we compute a feasible solution of CC-KAP which gives a upper bound of the optimal solution by Lemma 2 (lines 1 to 6). We begin with solving RA-KAP for λ equal to 0. If the optimal solution of RA-KAP does not satisfy the chance constraint (e.g., point s_1 in Figure 2), we update λ using $\lambda = C/\sigma$ where $C = \Phi^{-1}(p)$ and $\sigma = \sqrt{\sum_{i=1}^{n} \sigma_i^2 f_i}$. Then, we solve RA-KAP with the updated λ and store the corresponding optimal solution and objective value in set $\mathbb{S}$ and $\mathbb{V}$ respectively. This procedure continues until the optimal solution of RA-KAP satisfies the chance

constraint (e.g., point s_2 in Figure 2). Once we find a feasible solution to CC-KAP, it implies that we have explored the region that is the intersection of feasible region of CC-KAP and the feasible region of RA-KAP which gives the first feasible solution of CC-KAP. In practice, this region is a large portion of feasible region of CC-KAP. We will discuss this further in the next section.

Now we need to explore the rest of the feasible region of CC-KAP. In line 7, we compute the intersection of parabola of chance constraint and the line from the constraint of the last RA-KAP, denoted by point a shown in Figure 2, whose coordinates in variance-mean plane is $(\frac{C^2}{\lambda^2},\ L+\frac{C^2}{L})$. Then the algorithm computes the point with the largest x-coordinate on the parabola. The coordinate of this point is $(\sum_{i=1}^{n}\sigma_i^2,\ L+C\sqrt{\sum_{i=1}^{n}\sigma_i^2})$, denoted by point b. Note that all solutions will be to the left of the vertical line passing through point b. Thus, the region to be explored is bounded by a triangle formed by constraint line passing through point a, the line through both point a and point b (which are constraint 4 and constraint 3, respectively, in Figure 2), and the vertical line through point b. In line 8, we compute the equation of line that goes through point a and point b. Let $\hat{\lambda}_{ab}$ equal to its slope, stored in set of risk-averse parameter set Λ and the length of route $\hat{L}_{ab}$ equal to its y-intercept, stored in set $\mathbb{L}$.

In lines 9 to 20, we explore the triangle described in the previous paragraph (formed by constraint 3, constraint 4, and the vertical line through point b). Note that the feasible region of the RA-KAP with $\hat{\lambda}_{ab}$ and $\hat{L}_{ab}$ includes this triangle. In line 11, we solve the RA-KAP with $\hat{\lambda}_{ab}$ and $\hat{L}_{ab}$. If the solution s_{ab} satisfy the chance constraint, in other words the corresponding point of $\hat{s}_{ab}$ is within the enclosed region, the whole area is explored. Therefore, we store the objective value in $\mathbb{V}$ and store the solution in $\mathbb{S}$ (lines 12 to line 17). Otherwise the solution is in the bow-shaped region between parabola and line. We have to reduce this bow-shaped region by inserting two line segments so that the solution will not dominate over the solution in unexplored feasible region. One endpoint of those line segments are point a and point b respectively. Another endpoint is shared by both, which is the intersection of parabola and the line going through point a. The slope of this line segment is equal to ϵ plus the slope of line connecting point a and point related to solution s_{ab}, i.e., $\hat{\lambda}_{1,1}=\epsilon+\frac{\hat{\mu}_{ab}-\mu_a}{\hat{\sigma}^2_{ab}-\sigma^2_a}$ where $(\sigma_a^2,\ \mu_a)$ is the point a and $(\hat{\sigma}^2_{ab},\ ,\hat{\mu}_{ab})$ is the point related to the solution $\hat{s}_{ab}$. Note that this point should be under the line segment otherwise the solution of RA-KAP would turn out to be same as s_{ab}. Also ϵ should be small enough so that the intersection point is within the parabola from point a to point b. After computing the endpoint, we simply compute the slope of line segment going through point b and therefore the y-intercepts of extension of both line segments can be computed, denoted by $\hat{L}_{1,1}, \hat{L}_{1,2}$ respectively (shown in line 13). Store all computed slopes and y-intercepts in set Λ' and set $\hat{L}'$ respectively (shown in line 14). Update $\hat{\Lambda}$ and $\mathbb{L}$ (shown in line 19). Then solve RA-KAP with all $\hat{\lambda}_i \in \hat{\Lambda}$ and $\hat{L}_i \in \mathbb{L}$, and further determine if the solutions satisfy the chance constraint. If yes, store the solution and objective in $\mathbb{S}$ and $\mathbb{V}$ respectively. Otherwise update $\hat{\Lambda}$ and $\hat{L}$ and solve RA-KAP with these updates. Repeat procedure until $\hat{\Lambda}=\emptyset$ (shown in line 9). Now the optimal solutions of RA-KAPs related to all line segments between point a and point b satisfy the chance constraint. It means that the whole area of the feasible region of CC-KAP is explored. Finally, in line 21, the algorithm finds the optimal objective function value of CC-KAP by choosing the smallest value stored in $\mathbb{V}$ and the optimal solution is the corresponding solution in $\mathbb{S}$.

VI. Simulation Results

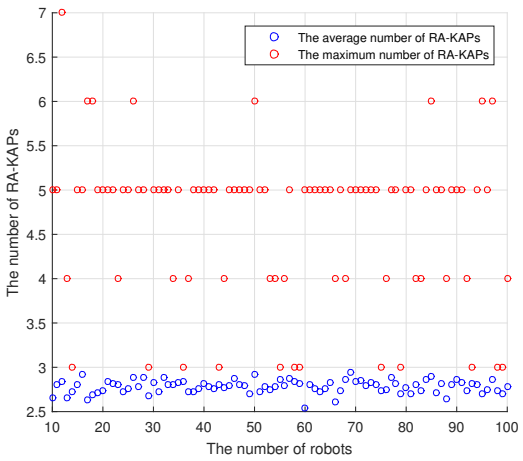

Fig. 3. For a given number of robots, the blue dots show the average number of call to RA-KAPs for solving a CC-KAP. The red dots show the maximum number of calls to RA-KAPs. Each dot is generated using 100 simulations with randomly generated mean and variance for travel distance of each robot. The length of the route is $10,000$ meters and the number of robots vary from 10 to 100.

In this section, we present simulation results to characterize the performance of our algorithm. The goal of the simulation studies is to understand the scalability of our algorithm as a function of the number of robots and the uncertainty in knowledge about the distance the robots can travel (i.e., variance). Our algorithm solves the chance constrained knapsack problem by solving a number of deterministic risk-averse knapsack problems. Thus, the efficiency of our algorithm depends on the number of deterministic knapsack problems solved. To understand the effects of parameters such as the number of robots and the variance of travel distance of robots, we generated different scenarios based on randomly generated parameter values. We will first present simulation results in which the mean and variances of travel distance of robots are randomly generated and the number of robots is varied methodically. Then we will present the simulation results in which the variances of travel distance are varied. Our simulations were performed on Lenovo Y50 laptop with Intel 2.60 GHz processor and 16.0 GB RAM.

In the first simulation, we test the effect of number of robots on the number of RA-KAPs to be solved which in

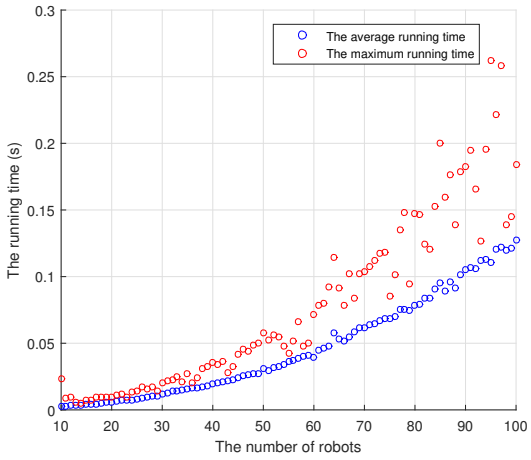

Fig. 4. For a given number of robots, the blue dots show average running time (in seconds) for solving CC-KAP. The red dots show the maximum running time. Each dot is generated using 100 simulations with randomly generated mean and variance for travel distance of each robot. The length of the route is $10,000$ meters and the number of robots vary from 10 to 100.

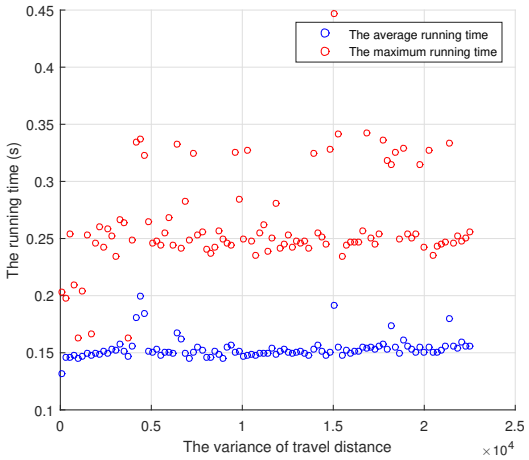

Fig. 6. The blue dots show the average and the red dots the maximum running times (second) for solving CC-KAP as the variance of the travel distance is changed from 100 to 22500. Each dot is obtained from 100 simulations with randomly generated mean travel distance of each robot. The number of robots is 100 and the length of the route is $50,000$ meters.

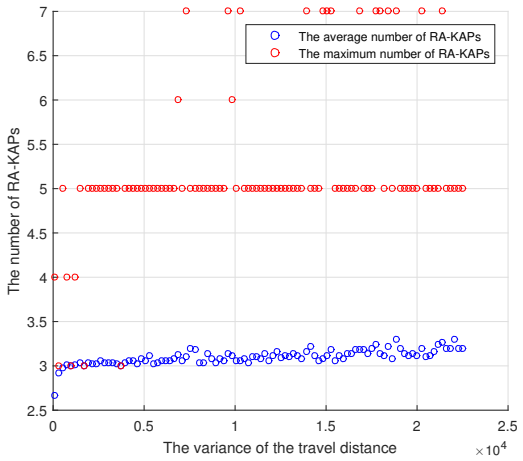

Fig. 5. The blue dots show the average and the red dots show the maximum number of RA-KAPs called for solving CC-KAP, as the variance of the travel distance is changed from 100 to 22500. Each dot is obtained from 100 simulations with randomly generated mean travel distance of each robot. The number of robots is 100 and the length of the route is $50,000$ meters.

turn determines the algorithm performance. We use dynamic programming to solve the RA-KAP optimally in pseudo-polynomial time, $\mathcal{O}(n^2P)$, where n is the number of robots and P is the largest cost among all robots [21]. As mentioned in Section III, the travel distance of each robot is Gaussian random variable. The means and variances of these random variables are generated independently from a uniform distribution $\mu_i \sim \mathcal{U}(1000, 3000)$ and $\sigma_i^2 \sim \mathcal{U}(10000, 12500)$. The total length that the robots have to traverse as a team, L, is $10,000$ meters. The probability that the robots cover the route is 0.99. In other words, the total distance the chosen robots can cover is greater than $10,000$ with probability 0.99, despite the actual realization of their travel length in any given scenario. Moreover the operation and maintenance cost for each robot is drawn uniformly from 50 to 150, i.e., $c_i \sim \mathcal{U}(50, 150)$. We set $\epsilon = 1 \times 10^{-7}$. We count the number of RA-KAPs for solving CC-KAP when number of robots is equal to $10, 11, \ldots, 100$. For each case with a given number of robots, we generate the means and variances randomly for 100 times.

Figure 3 and Figure 4 show the performance of our algorithm with the different number of robots. In Figure 3, the blue dots represent the average number of RA-KAPs to solve CC-KAP with given number of robots while the red dots represent the maximum number of calls to RA-KAPs. Each dot is generated with the results from 100 simulations. As it is shown, the average numbers of RA-KAP solved is almost constant (between 2.5 to 3) irrespective of the number of robots. In Figure 4, the maximum numbers of RA-KAPs solved are between 3 and 7, which is a small value for practical applications. The results reveal that the number of robots does not have a significant influence on the speed of our algorithm and the number of calls to RA-KAPs is nearly a constant. Another way to measure the speed of our algorithm is the actual running time. As shown in Figure 4, the maximum running time is less than 0.3 seconds and both the average and maximum numbers increase polynomially with respect to the number of robots. This is because the algorithm used to solve deterministic RA-KAP is a $\mathcal{O}(n^2P)$ algorithm while the number of calls RA-KAPs is nearly a constant.

In the second simulation, we study the effect of the uncertainty of travel distance of the robots on the performance of our algorithm by counting the number of calls to RA-KAPs and the actual running time for our algorithm solving CC-KAP with variance equal to $100, 324, 548, \ldots, 22500$. For each value of variance, we generate mean of travel distance of robots randomly, by sampling from the uniform

distribution, $\mathcal{U}(1000, 3000)$. For each value of variance, we denote 100 different scenarios. The number of robots is 100 and the length of the route is $50,000$ meters for all scenarios in this simulation. The other parameters such as the cost, c_i, the probability in the chance constraint, p, and ϵ are same as that in the first simulation. Figure 5 shows that the average number of calls (blue dots) to RA-KAPs is practically constant as the variance of travel distance increases. The maximum numbers of calls (red dots) range from 3 to 7. Figure 6 shows that the average running time is about 0.15 seconds and the maximum running times are less than 0.45 seconds. This is because the increased variance does not increase the computational burden for the algorithm solving the deterministic RA-KAP and the number of calls is constant. Figure 5 and Figure 6 implies that our algorithm scales well when the uncertainty of travel distances of robots are high.

VII. Conclusion

In this paper, we presented an algorithm for computing optimal solutions for chance-constrained knapsack problems with deterministic objective and stochastic knapsack constraint. The key idea in our approach is to convert CC-KAP to a deterministic discrete optimization problem on the variance-mean plane, where each point on the plane can be identified with an assignment of items to the knapsack. By exploiting the geometry of the non-convex feasible set of the CC-KAP in the variance-mean plane, we showed that CC-KAP can be solved optimally by solving a sequence of deterministic risk-averse knapsack problems. We presented an algorithm to systematically generate the sequences of RA-KAPs to solve the CC-KAP.

Our problem formulation is motivated by team selection problems in multirobot systems, where robots have limited battery life, under uncertain knowledge about their durability in the mission. We presented simulation results on randomly generated data to show empirically that the number of RA-KAPs required to solve the CC-KAP is a small constant (less than 7), irrespective of the number of robots or the uncertainty in the knowledge about the distance that a robot can travel. Thus, our algorithm scales well with the number of robots from which we can choose the team and also with the variance of the battery life, i.e., the distance that a robot can travel.

Although, empirically, our algorithm seems to be computationally efficient, we do not have any theoretical guarantees about the computational complexity of our algorithm. In future work, we will work on analyzing the computational complexity of our algorithm. We also intend to extend our method to solve the stochastic versions of the multiple knapsack problem and the generalized assignment problem. These problems arise in task allocation problems for multiple robots that have limited battery life [9].

References

[1] 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)*, 40(2):161–173, 1993.

[2] B. C. Dean, M. X. Goemans, and J. Vondrk. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Mathematics of Operations Research*, 33(4):945–964, 2008.

[3] E. Delage and S. Mannor. Percentile optimization for markov decision processes with parameter uncertainty. *Operations Research*, 58(1):203–213, 2010.

[4] V. Goyal and R. Ravi. A ptas for the chance-constrained knapsack problem with random item sizes. *Operations Research Letters*, 38(3):161 – 164, 2010.

[5] M. I. Henig. Risk criteria in a stochastic knapsack problem. *Operations Research*, 38(5):820–825, 1990.

[6] H. Ishii, S. Shiode, T. Nishida, and Y. Namasuya. Stochastic spanning tree problem. *Discrete Applied Mathematics*, 3(4):263 – 273, 1981.

[7] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2011.

[8] O. Klopfenstein and D. Nace. A robust approach to the chance-constrained knapsack problem. *Operations Research Letters*, 36(5):628 – 632, 2008.

[9] L. Luo, N. Chakraborty, and K. P. Sycara. Distributed algorithm design for multi-robot generalized task assignment problem. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, pages 4765–4771, 2013.

[10] S. Martello, D. Pisinger, and P. Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Manage. Sci.*, 45(3):414–424, Mar. 1999.

[11] S. Martello and P. Toth. A mixture of dynamic programming and branch-and-bound for the subset-sum problem. *Management Science*, 30(6):765–771, 1984.

[12] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[13] C. Nam and D. A. Shell. Analyzing the sensitivity of the optimal assignment in probabilistic multi-robot task allocation. *IEEE Robotics and Automation Letters*, 2(1):193–200, Jan 2017.

[14] E. Nikolova. *Approximation Algorithms for Reliable Stochastic Combinatorial Optimization*, pages 338–351. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[15] E. Nikolova, J. A. Kelner, M. Brand, and M. Mitzenmacher. *Stochastic Shortest Paths Via Quasi-convex Maximization*, pages 552–563. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[16] V. Poirriez, N. Yanev, and R. Andonov. A hybrid algorithm for the unbounded knapsack problem. *Discrete Optimization*, 6(1):110 – 124, 2009.

[17] S. S. Ponda, L. B. Johnson, and J. P. How. Distributed chance-constrained task allocation for autonomous multi-agent teams. In *American Control Conference (ACC)*, June 2012.

[18] S. S. Ponda, L. B. Johnson, and J. P. How. Risk allocation strategies for distributed chance-constrained task allocation. In *American Control Conference (ACC)*, June 2013.

[19] A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2014.

[20] M. Sniedovich. Preference order stochastic knapsack problems: Methodological issues. *The Journal of the Operational Research Society*, 31(11):1025–1032, 1980.

[21] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[22] F. Yang and N. Chakraborty. Algorithm for optimal chance constrained linear assignment. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 801–808, May 2017.