Chance Constrained Simultaneous Path Planning and Task Assignment for Multiple Robots with Stochastic Path Costs

Fan Yang[1] and Nilanjan Chakraborty[2]

***Abstract*— We present a novel algorithm for simultaneous task assignment and path planning on a graph (or roadmap) with stochastic edge costs. In this problem, the initially unassigned robots and tasks are located at known positions in a roadmap. We want to assign a unique task to each robot and compute a path for the robot to go to its assigned task location. Given the means and variances of travel cost of each edge, our goal is to develop algorithms that guarantee that the total path cost of the robot team is below a minimum value in any realization of the stochastic travel costs with high probability. We formulate the problem as a chance-constrained simultaneous task assignment and path planning problem (CC-STAP). We prove that the optimal solution of CC-STAP can be obtained by solving a sequence of deterministic simultaneous task assignment and path planning problem in which the travel cost is a linear combination of mean and variance of the edge cost. We show that the deterministic problem can be solved in two steps. In the first step, robots compute the shortest paths to the task locations and in the second step, the robots solve a linear assignment problem with the costs obtained in the first step. We also propose a distributed algorithm that solves CC-STAP near-optimally. We present simulation results on randomly generated networks and data to demonstrate that our algorithm is scalable with the number of robots (or tasks) and the size of the network.**

I. Introduction

Multirobot task allocation problems where robots have to move to target destinations arises in a number of applications including search and rescue, and goods or parts transfer in warehouses. In such scenarios, the robots have to navigate in environments containing both static and dynamic obstacles. There are two related problems to be solved, namely, (a) multirobot task allocation problems, wherein robots have to be assigned to a destination and (b) multirobot path planning problems wherein collision-free paths have to be planned for each robot between their origin and destination. The two problems are usually decoupled. In task allocation problems, it is commonly assumed that the cost of the paths between robot-destination pairs are known, which implicitly implies that a single collision-free path has been pre-computed between each robot-destination pair. In path planning problems the origin and destination of each robot is usually given, which implicitly implies that the task assignment for each robot has been computed.

Further, for task allocation, the path costs are usually assumed to be deterministic. However, the path costs may be stochastic, especially in *open environments*, where other (uncontrolled) mobile agents like people or other cars can occupy the space. To avoid collisions, robots may have to slow down or deviate locally from their planned paths, thus making travel costs (like time or energy consumed) stochastic. The decoupling of the task allocation and path planning problems, although conceptually convenient can lead to sub-optimal solutions. Therefore, in this paper, we consider robot task allocation problems where the robots have to simultaneously plan paths and select target destinations (or tasks) under uncertainty about the travel costs.

We assume that the robots move on a graph. The graph may represent an actual road network or a roadmap which captures the collision-free configuration space of the robots. It is assumed that robots have local collision avoidance schemes to avoid mobile obstacles. There are many algorithms like PRM [5] or RRT [7] or their optimal variants PRM* and RRT* [4] that can generate the roadmaps for any given environment. We assume that the cost on each edge of the graph is a random variable with known mean and variance. Thus the total path cost of the robot team is a random variable. Our goal is to *simultaneously compute the assignment of tasks (targets) to robots as well as paths to reach the tasks and a minimum value (say y) of the team performance objective such that we have a guarantee that the robot team performance will be less than y with high probability (say 0.95) under any realization of the random costs*. Such a solution will provide a quality guarantee (albeit probabilistic) on the solution of the simultaneous task assignment and planning (STAP) problem in the presence of uncertainty about the task execution costs.

We model the stochastic STAP problem as a chance constrained combinatorial optimization problem and call the problem chance-constrained simultaneous task assignment and planning (CC-STAP) problem. We prove that the optimal solution of CC-STAP can be obtained by solving a sequence of deterministic simultaneous task assignment and path planning problem (D-STAP) in which the travel cost on each edge is a linear combination of mean and variance of the edge cost. We show that the D-STAP problem can be solved by following steps: computing the shortest paths to the task locations and solving a linear assignment problem with the shortest path costs. The algorithms to solve CC-STAP optimally and also the algorithm to solve D-STAP optimally are the primary contributions of this work. We also present a distributed algorithm to solve CC-STAP that builds on the auction algorithm for solving linear assignment problems [1], [25].

To the best of our knowledge, there are no available

[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

algorithms with theoretical guarantees on solution quality that can solve our version of the combined task assignment and planning problem with stochastic costs. In previous work [24], we have solved a version of the CC-STAP problem, where the team objective is to minimize the maximum cost for a robot team. The chance-constrained problem is solved by solving a linear bottleneck assignment problem and a number of chance-constrained shortest path problems. In this paper, the team objective is to minimize the total costs of the robot team. In the extant literature, there are centralized algorithms such as the Hungarian algorithm [6], distributed algorithms with shared memory [1] and totally distributed algorithms [25] solving deterministic task assignment problem. In [13], the authors solve a task allocation problem under uncertainty for analyzing the sensitivity of the optimal assignment with respect to the uncertainty in payoffs. In [20], a redundant robot assignment on graphs with uncertain edge costs is studied. There has been some effort in solving different variations of the stochastic shortest path problem [2], [3], [8], [10], [11], [12], [15], [16], [18], [19], in which one robot has to plan its motion to a destination node, with random costs on the edges. In [9], the authors considered a stochastic path planning problem for one robot that has to visit a set of nodes in a predefined sequences. Our problem involves not only path planning but also the task assignment which is not predefined. In the deterministic setting combined goal assignment and collision-free trajectory planning problem has been studied in [21], [22], [17]. The distinction of these papers from our problem is that we consider stochastic costs and our planning is on a discrete structure instead of continuous space.

II. Problem Formulation

Given a graph $G = (V, E)$, a set of N_r heterogeneous robots r_i, a set of N_t tasks t_i with known initial positions, the travelling cost ${}^i c_{uv}$ incurred when the robot r_i goes through any edge e_{uv}. It is assumed that ${}^i c_{uv}$ is random variable with known mean ${}^i\mu_{uv}$ and variance ${}^i\sigma^2_{uv}$. The goal is to compute the path for each robot to a unique target such that the total path cost of robot team is less than a value y with probability at least p. Our objective is to minimize the value y.

$$\begin{aligned}
&\min \quad y \\
&\text{s.t. } \mathbb{P}\left(\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i c_{uv}\ {}^i x_{uv} \le y\right) \ge p \\
&\sum_{v\in\mathcal{N}(u)} {}^i x_{uv} - \sum_{v\in\mathcal{N}(u)} {}^i x_{vu} = \begin{cases} 1, & \text{if } u = s_{r_i}, \\ -z_{ij}, & \text{if } u = t_j, \\ 0, & \text{otherwise} \end{cases} \quad \forall i \\
&{}^i x_{vu} \in \{0,1\}, \quad \forall u,v \in V,\ i = 1,\ldots,N_r. \\
&\sum_{i=1}^{N_r} z_{ij} = 1,\ \forall j; \quad \sum_{j=1}^{N_t} z_{ij} = 1,\ \forall i; \quad z_{ij} \in \{0,1\}\ \forall i,j.
\end{aligned} \tag{1}$$

The solution of this problem includes two parts: assignment indicated by decision variable $\{z_{ij}\}, \forall i, j$ and path by $\{{}^i x_{uv}\}, \forall u, v \in V, \forall i$. In particular, $z_{ij} = 1$ when r_i is assigned to t_j and ${}^i x_{uv} = 1$ when edge e_{uv} is an edge of path for r_i. The probabilistic guarantee on the performance of robot team is ensured by the chance constraint which guarantees that the total travel cost for all robots, $\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i c_{uv}\ {}^i x_{uv}$, is at most the cost value y in any realization of the random travel cost with probability at least p. The second set of constraints is the path constraints for every robot. In particular, for any robot r_i the difference in the number of the edges along the path leaving and entering one node, say u, is equal to 1 when u is the source node s_{r_i}, equal to $-z_{ij}$ when u is a node for any task, say t_j, and equal to 0 otherwise. More precisely for case when $u = t_j$, the difference is equal to -1 when t_j is the task assigned to r_i and is 0 when t_j is not assigned to r_i (u is an intermediate point on the path or a point not on the path). The constraints for z_{ij} ensure that each robot performs only one task and each task is assigned to one robot.

The chance constraint can be written as inequality (2) based on different assumptions on the probability distribution of the travel cost. If the travel costs are independent Gaussian random variables, i.e., ${}^i c_{uv} \sim \mathcal{N}({}^i\mu_{uv},\ {}^i\sigma^2_{uv})$, the total path cost for the robot team is thus a Gaussian random variable with mean and variance equal to the sum of means and variances of the edge along the paths. By standardized total travel cost, the following equivalent constraint can be obtained:

$$\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i\mu_{uv}\ {}^i x_{uv} + C\sqrt{\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i\sigma^2_{uv}\ {}^i x_{uv}} \le y \tag{2}$$

where $c = \Phi^{-1}(p)$ and $\Phi^{-1}(\cdot)$ denotes the inverse cumulative distribution function of $\mathcal{N}(0, 1)$.

Note that the independent and distribution assumption are for clarification and brevity of the mathematical exposition. It can be generalized to more realistic scenarios where only means and variances of travel costs are available while the distribution information is absent. In particular, by Chebyshevs inequality solving problem with $C = \sqrt{\frac{p}{1-p}}$ will obtain a feasible solution that satisfies the chance constraint. Further with appropriate graph transformation, our method could be extended to the case where the travel costs are dependent. An graph transformation example was provided in [14].

For limited space, we use $\mathcal{F}$ to denote the feasible space of all constraints for ${}^i x_{uv}$ and z_{ij} except chance constraint in (1). Let y equal to the left-hand side of (2). The formulation in (1) can thus be equivalently written as

$$\begin{aligned}
&\min \quad \sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i\mu_{uv}\ {}^i x_{uv} + C\sqrt{\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i\sigma^2_{uv}\ {}^i x_{uv}} \\
&\text{s.t.} \quad {}^i x_{vu}, z_{ij} \in \mathcal{F}, \forall i, u, v
\end{aligned} \tag{3}$$

The problem above is a non-liner integer program which is difficult to solve in general.

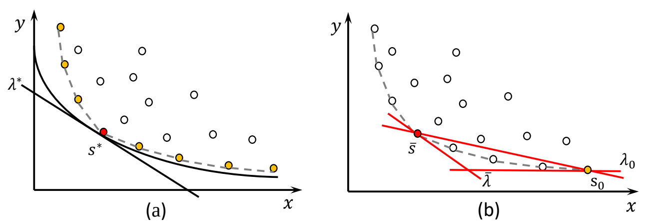

Fig. 1. All feasible solutions of (3) and (4) are a set of points on variance-mean plane. (a) The optimal objective function in (3) is the level curve through a point with smallest vertical intercept. The optimal solution of (3) is an extreme point which can be obtained by solving (4) with proper risk-averse parameter, e.g., λ^*. (b) Our method finds the search region that allows us obtain the optimal solution of (3) with smaller number of risk-averse problems (4) solved than enumerating all extreme points on variance-mean plane.

III. Geometric analysis

Based on our previous work [23], we consider the solution in (1) and (3) as a point on a two-dimensional space called variance-mean plane as Fig. 1. Recall that a feasible solution is essentially a set of the paths. Each solution can be treated as a point with horizontal and vertical coordinates equal to the sum of the variances and means of random cost of edge along the paths respectively, i.e. $\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|}\ {}^i\sigma_{uv}^2\ {}^ix_{uv}$ and $\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|}\ {}^i\mu_{uv}\ {}^ix_{uv}$. The level curve of the objective function in (3) is a parabola (Fig. 1 (a)). The objective value of a solution is equal to the vertical intercept of the level curve through the associated point of the solution. *Therefore the optimal solution of the chance-constrained problem* (3) *is the point with the smallest vertical intercept of the level curve through it.* The optimal point can be computed by solving a risk-averse problem given below.

$$\begin{aligned} \min \quad & \sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} \left({}^i\mu_{uv} + \lambda\ {}^i\sigma_{uv}^2\right)\ {}^ix_{uv} \\ \text{s.t.} \quad & {}^ix_{vu}, z_{ij} \in \mathcal{F}, \forall i, u, v \end{aligned} \tag{4}$$

This problem is a deterministic version of our problem in (1) in which the edge costs are linear combination of means and variances i.e., ${}^i\mu_{uv} + \lambda\ {}^i\sigma_{uv}^2$. λ is the risk-averse parameter (also known as the Arrow-Pratt index of absolute risk aversion in economics). The higher the value of λ the more risk averse the robot. We need to simultaneously compute the assignment of robots to tasks and plan the path for robots to reach the tasks. The objective is to minimize the total path cost of the robot team. The level curve of the objective function in (4) is a straight line with the slope equal to $-\lambda$ as shown in Fig. 1. Since constraints in (4) and (3) are same, all feasible solutions have same coordinates on variance-mean plane. From a geometrical point of view, solving the optimal solution is equivalent to find the straight line through a feasible point with the lowest vertical intercept. Hence there is no feasible point below the straight line. Therefore a key observation is that *the optimal solution of* (4) *is an extreme point of the feasible point set* (An extreme point of a set of points by definition is the point that there exists a straight line through it such that all other points are on one side of the line.). The following lemma demonstrates the relationship between risk-averse problem (4) and chance-constrained problem (3).

Lemma 1: The optimal solution of problem (3) is the optimal solution of (4) with proper value of λ.

Proof: As shown in Fig. 1 (a), the optimal solution of (3) is the point passed by the parabola level curve with the lowest vertical intercept. Therefore there is no feasible point below the level curve. There always exists a straight line through the optimal point such that there is also no feasible point below it, such as the tangent of the parabola at this point. Let λ equal to the negation of the slope. The optimal solution of (4) is the optimal solution of (3). ■

Let λ^* denote the risk-averse parameter such that the optimal solution of (4) with $\lambda = \lambda^*$ is optimal to (3). We present a method in Alg. 1 that finds the upper bound of λ^*.

IV. Solution Approach

Although, we do not show explicitly, Lemma 1 implies that the optimal solution of CC-STAP is an extreme point of the feasible solution set. Thus, one way to compute the optimal CC-STAP solution is to enumerate all extreme points on variance-mean plane (this is shown as yellow dots in Fig. 1 (a)). However the number of extreme points in the worst case for this problem could be large.

We propose a novel two-step algorithm to solve CC-STAP: (1) We first find a search region (shown in Fig. 1 (b)) for the extreme points within which the optimal solution is guaranteed to lie, by computing an upper bound for λ^*. (2) We then enumerate extreme points in this search region by solving a sequence of risk-averse problems (4) with methodically generated λ. As results shown in section VI, the number of risk-averse problem solved by our method is less than enumerating all extreme points.

Let μ_k and σ_k^2 denote the sum of the mean and variance of the edges along the path computed from risk-averse problem (4) with λ_k respectively, i.e., $\mu_k = \sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|}\ {}^i\mu_{uv}\ \ {}^ix_{uv}^*$, $\sigma_k^2 = \sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|}\ {}^i\sigma_{uv}^2\ \ {}^ix_{uv}^*$. Further let $\sigma_k = \sqrt{\sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|}\ {}^i\sigma_{uv}^2\ \ {}^ix_{uv}^*}$. Therefore the optimal solution of (4) is a point with coordinate (σ_k^2, μ_k) and the optimal objective function value for (4) is $\mu_k + \lambda_k\sigma_k^2$. The objective function value for (3) at this point is $\mu_k + C\sigma_k$.

We will provide two observations from our previous work [23] that solve chance-constrained linear assignment problem in which each solution is also treated as a point and the level curve of objective is also parabola though the distribution of feasible points are different because of different constraints. The observations are still valid in this paper. Let $\lambda_k < \lambda_{k+1}$ be two different risk-averse parameters, then:

1) $\mu_k + \lambda_k\sigma_k^2 < \mu_{k+1} + \lambda_{k+1}\sigma_{k+1}^2$
2) ${\sigma_k}^2 \geq \sigma_{k+1}^2$

Based on the above, there is a key lemma used to design the first step of our algorithm.

Lemma 2: Let $\bar{\lambda}$ be a risk-averse parameter, such that the optimal objective function value of (4) with $\bar{\lambda}$ is equal to the objective value of (3) at the same solution, i.e., $\bar{\mu} + C\bar{\sigma} = \bar{\mu} + \bar{\lambda}\bar{\sigma}^2$. This objective value gives an upper bound for the optimal objective value of (3). The optimal risk-averse parameter λ^* must lie in the interval $[0, \bar{\lambda}]$.

Proof: Let μ and σ^2 are obtained from the optimal solution of (4) with $\lambda > \bar{\lambda}$. We need to prove that the objective values of (3) at the solution obtained from the optimal solution of (4) with λ are greater than the objective value obtained at $\bar{\lambda}$, i.e., $\bar{\mu} + C\bar{\sigma} < \mu + C\sigma$ for any $\lambda > \bar{\lambda}$.

There are two different cases:

- $\lambda\sigma \leq C$. Obviously $\mu + C\sigma \geq \mu + \lambda\sigma^2$. And because $\lambda > \bar{\lambda}$, from observation 1 we know that $\mu + \lambda\sigma^2 > \bar{\mu} + \bar{\lambda}\bar{\sigma}^2 = \bar{\mu} + C\bar{\sigma}$. Therefore $\bar{\mu} + C\bar{\sigma} < \mu + C\sigma$.
- $\lambda\sigma > C$. Let $\lambda' = C/\sigma$. Thus, $\lambda' < \lambda$. Since $\lambda > \bar{\lambda}$, from observation 2 we know $\sigma < \bar{\sigma}$. Thus, $\frac{C}{\sigma} > \frac{C}{\bar{\sigma}}$, which implies $\lambda' > \bar{\lambda}$. Therefore, $\mu + C\sigma = \mu + \lambda'\sigma^2 \geq \mu' + \lambda'\sigma'^2 > \bar{\mu} + \bar{\lambda}\bar{\sigma}^2 = \bar{\mu} + C\bar{\sigma}$.

Thus, $\bar{\mu} + C\bar{\sigma}$ is the upper bound of optimal objective value for chance-constrained problem (3). Since the optimal risk-averse parameter, λ^* is a positive number, λ^* must be in the range $[0, \bar{\lambda}]$. ■

Once $\bar{\lambda}$ is computed, we can obtain a triangular search region shown in Fig. 1 (b). We will prove that $\bar{\lambda}$ always exists and can be found in finite number of steps in Lemma 3.

V. ALGORITHM

We will present our distributed algorithm comprised of three components: the first step of our algorithm that finds $\bar{\lambda}$ and a search region in Alg. 1, the distributed algorithm solving risk-averse problem (4) in Alg. 2 used as a subroutine for two-step algorithm, and the second step of our algorithm that enumerates the extreme points in the search region.

Initial Knowledge of the Robots: We assume that each robot, r_i, only knows the pre-specified probability p (hence C), and graph structure G with the means and variances of its own cost for travelling any edges e_{uv}, i.e., ${}^i\mu_{uv}$ and ${}^i\sigma^2_{uv}$.

A. Searching for the bound

Alg. 1 is the procedures for any robot r_i that compute $\bar{\lambda}$ in the first step of our algorithm. At any iteration k (line 1 or line 6), r_i solves risk-averse problem (4) with λ_k (initially $\lambda_0 = 0$). Let α_i denote the index of the task assigned to r_i. We obtain the assigned task t_{α_i}, the path to the assigned task ${}^i\Pi_k = \{{}^ix^*_{uv}\}$ and the variance of path cost, i.e., $\sigma^2_k = \sum_{i=1}^{N_r}\sum_{u,v=1}^{|V|} {}^i\sigma^2_{uv}\ {}^ix^*_{uv}$. Thus r_i can compute $\sigma_k = \sqrt{\sigma^2_k}$ (line 2 or line 7). Then r_i determines whether λ_k is the upper bound $\bar{\lambda}$ by the condition $\lambda_k\sigma_k = C$ (line 3, from lemma 2). If no, the risk-averse parameter is updated as $\lambda_{k+1} = \frac{C}{\sigma_k}$ (line 4). Then r_i moves on to the next iteration (line 4-7). The procedure repeats until the condition is satisfied. Now λ_k is equal to upper bound $\bar{\lambda}$. Finally r_i return the path to the assigned task ${}^i\bar{\Pi}$ obtained from risk-averse parameter with $\bar{\lambda}$.

Lemma 3: The Alg. 1 finds $\bar{\lambda}$ and terminates in finite number of iterations.

Algorithm 1 The distributed algorithm for robot r_i searching for the upper bound $\bar{\lambda}$

1: Let $k = 0$, solve risk-averse problem with $\lambda_k = 0$.
2: Compute σ_k from the output.
3: **while** $\lambda_k\sigma_k \neq C$ **do**
4:     Update the risk-averse parameter by $\lambda_{k+1} = \frac{C}{\sigma_k}$.
5:     $k = k + 1$.
6:     Solve risk-averse problem with λ_k
7:     Compute σ_k from the output.
8: **return** Path to assigned task ${}^i\bar{\Pi}$.

Proof: In the first iteration $\lambda_0 = 0$ and obviously $\lambda_0\sigma_0 < C$ and $\lambda_1 > \lambda_0$. In any iteration k before the termination, $\lambda_k\sigma_k \neq C$. In fact $\lambda_k\sigma_k < C$ because $\lambda_k\sigma_{k-1} = C$ by updating rule and $\sigma_k < \sigma_{k-1}$ by observation 2. Thus, we have a strictly increasing sequence of values of λ, namely, $\lambda_0 < \lambda_1 < \lambda_2 < \cdots$. Before termination, in each iteration a new extreme point is found by solving risk-averse problem (4) with λ_k. The algorithm terminates when the obtained extreme point is same with the previous iteration. There is some value of λ such that the risk-averse problem with it produces the solution with minimum variance of path cost. In the worst case, when λ_k is greater than that value, the solution of risk-averse problem in the next iteration would remain the same. Therefore $\lambda_{k+1}\sigma_{k+1} = \lambda_{k+1}\sigma_k = C$. The condition is satisfied. Since the number of iterations must be less than or equal to the number of extreme points of the solution set which is finite, the number of iteration should be finite. ■

Note that if there are multiple risk-averse parameters satisfying the condition $\lambda\sigma = C$, our algorithm finds the smallest one. The reason is that for all λ in the obtained interval $[0, \bar{\lambda})$, it is always true that $\mu + C\sigma > \mu + \lambda\sigma^2$.

B. Risk-averse problem

The risk-averse problem in (4) is a deterministic STAP. The edge costs are linear combination of means and variance, i.e., ${}^i\mu_{uv} + \lambda\ {}^i\sigma^2_{uv}$. The goal is to assign each robot a unique task and compute the path for robot to reach the assigned task. The objective is to minimize the total path cost of the robot team. The following lemma shows the idea of our algorithm.

Lemma 4: The risk-averse problem in (4) can be solved optimally by solving a (minimization) linear assignment problem with assignment cost equal to the shortest path cost to the task under graph with deterministic edge cost ${}^i\mu_{uv} + \lambda\ {}^i\sigma^2_{uv}$.

Proof: Our first claim is that in the optimal solution each path should be the shortest path from robot to task. Because if any path is not the shortest path, we can always obtain a better solution using the shortest path for the same robot and task. Therefore we can use the shortest path costs for all robot-task pairs as the assignment costs. Further the optimal assignment of robots to tasks should provide the minimum total path cost and satisfies the assignment constraints for z_{ij} in $\mathcal{F}$, which are the constraints for linear assignment problem. Therefore the optimal assignment can be computed by solving such linear assignment problem. ■

Therefore we can have a distributed algorithm without shared memory for risk-averse problem by modifying the auction algorithm in [25]. In any iteration k of Alg. 1, given edge cost ${}^i\mu_{uv} + \lambda_k \ {}^i\sigma^2_{uv}$, each robot, say r_i computes the shortest path cost to all tasks denoted by $\{\ell_{ij}\}_{j=1}^{N_t}$ by Dijkstra. $-\ell_{ij}$ is used as the assignment cost in the auction algorithm (because of minimization problem). r_i should also compute variance for the path in the original problem (1), i.e., $v_{ij} = \sum_{u,v=1}^{|V|} {}^i\sigma^2_{uv} \ {}^ix^*_{uv}$ where $\{{}^ix^*_{uv}\}$ is the shortest path obtained from Dijkstra algorithm. The input of our auction algorithm for each robot r_i is therefore $\{\ell_{ij}, v_{ij}\}_{j=1}^{N_t}$. The local variable and message communicated among all robots is $\{p_{ij}, b_{ij}, \beta_{ij}\}_{j=1}^{N_t}$ where p_{ij} is the price that r_i has to pay in order to be assigned to t_j, b_{ij} is the local knowledge of index of the highest bidder for a task t_j. β_{ij} is the additional message used in our auction iteration, which indicates the variance of the path to task t_j in current auction iteration.

The single auction iteration of our algorithm is shown in Alg. 2. For each task t_j, robot r_i determines the robot in the neighborhood with the highest price, denoted by $r_d \in \mathcal{N}_i$, based on the message obtained from $\mathcal{N}_i$ (line 3). If there are multiple such robots, robot with greatest b_{hj} ($h \in \mathcal{N}_i$) is identified as r_d. Next r_i updates the local variables by copying the message from r_d (line 4). After updating local variables for all tasks, r_i determines whether the updated price of the current assigned task $p_{i\alpha_i}$ increases or the price does not change but the highest bidder of the task $b_{i\alpha_i}$ changes (line 5). If yes, r_i should re-compute the assigned task t_{α_i} with the highest net value, i.e., $-\ell_{ij} - p_{ij}$ based on the updated prices (line 6). Let q_i and w_i denote the highest and second highest net value for r_i. Then the price of the updated assigned task t_{α_i} is increased by $\gamma_i = q_i - w_i + \epsilon$ where ϵ is to prevent the cycle in the auction (line 7). Now in its neighborhood, r_i provides the highest price for the assigned task t_{α_i}. Therefore the highest bidder of the assigned tasks $b_{i\alpha_i}$ and the task variance $\beta_{i\alpha_i}$ are updated to i and $v_{i\alpha_i}$ respectively (line 8). Then r_i sends updated local variables to its neighbors (line 9). The auction iteration continues until the local prices does not change for Δ iterations where $\Delta \leq n-1$ is the maximum diameter of the network. The local knowledge of the task prices and task variances $\{p_{ij}, \beta_{ij}\}_{j=1}^{N_t}$ is now equal to the global information. Each robot r_i outputs the its assigned task t_{α_i}, the path to the assigned task denoted by ${}^i\Pi_k$ and the variance of total path cost, $\sigma_k^2 = \sum_{j=1}^{N_t} \beta_{ij}$.

C. Enumerating extreme points within the search region

In this subsection, we provide an outline of the second step of our algorithm. For this step, each robot needs to know both the mean and the variance of the total path cost obtained for a given λ, i.e., (σ_k^2, μ_k). The additional variable required for the second step is thus the mean of total path cost. Therefore in the auction iteration, a variable indicating the mean of the path to task should be used (similar to β_{ij}).

Algorithm 2 Auction iteration for robot r_i

1: Extract message $\{p_{hj}, b_{hj}, \beta_{hj}\}_{j=1}^{N_t}$ from neighbors, i.e., $\forall h \in \mathcal{N}_i$.
2: **for** each task t_j **do**
3:     Find the neighbor r_d with the highest price for t_j, i.e., $d = \arg\max_{h\in\{i,\ \mathcal{N}_i\}} p_{hj}$.
4:     Update local variables $\{p_{ij}, b_{ij}, \beta_{ij}\}$ by $p_{ij} = p_{dj}$, $b_{ij} = b_{dj}$ and $\beta_{ij} = \beta_{dj}$.
5: **if** $p_{i\alpha_i}$ increases or unchanged but $b_{i\alpha_i} \neq i$ **then**
6:     Update the assigned task t_{α_i} by $\alpha_i = \arg\max_{1\leq j\leq N_t} -\ell_{ij} - p_{ij}$.
7:     Increase the price for the assigned task $p_{i\alpha_i}$ by γ_i.
8:     Update $b_{i\alpha_i} = i$ and $\beta_{i\alpha_i} = v_{i\alpha_i}$.
9: Send local variables $\{p_{ij}, b_{ij}, \beta_{ij}\}_{j=1}^{N_t}$ to neighbors.
10: **return** α_i, ${}^i\Pi_k$ and $\sigma_k^2 = \sum_{j=1}^{N_t} \beta_{ij}$.

Let $s_k = (\sigma_k^2, \mu_k)$ denote the solution obtained in Alg. 1 for λ_k. After obtaining $\bar{\lambda}$, each robot r_i has a set of obtained solution $\{s_0, s_1, ..., \bar{s}\}$. Then r_i compute a set of point pairs, as $\{(s_0, s_1), (s_1, s_2), ...\}$. For each pair of solutions, say (s_a, s_b), each robot r_i computes the slope of the line connecting both points and let a risk-averse parameter λ_c equal to the negation, i.e., $\lambda_c = -\frac{\mu_a - \mu_b}{\sigma_a^2 - \sigma_b^2}$. Then r_i solves risk-averse problem with λ_c by auction algorithm. If the output solution, say $s_c = (\sigma_c^2, \mu_c)$ is different from s_a and s_b, then s_c is a new extreme point and r_i stores two pairs of points (s_a, s_c) and (s_c, s_b) that are used for the next iteration. When all pairs of points in the current iteration are processed, r_i moves on to the next iteration and computes λ by same procedure to each pair of points generated from last iteration. The procedure continues until no new extreme point is obtained. The optimal solution is the one with smallest objective value of problem (3).

Note that both Alg. 1 and the second step of our algorithm discussed here includes two components: updating λ_k and solving risk-averse problem. Since each robot can update λ_k based on its own knowledge of (σ_k^2, μ_k), our overall two-step algorithm that solve CC-STAP in (1) is a distributed algorithm.

VI. Simulation Results

The computational cost for each robot is $K(T_1 + I \cdot T_2)$ where K is the number of risk-averse problems (4) solved, $T_1 = O(|E| + |V|\log|V|)$ is the computational cost for Dijkstra algorithm, $I = O(\Delta N_r^2 \lceil \frac{\max_{i,j} \ell_{ij} - \min_{ij} \ell_{ij}}{\epsilon} \rceil)$ is the number of auction iterations [25] and T_2 is the computational cost for single auction iteration. The value for K depends on the number of extreme points on variance-mean plane. However, this number is problem parameter dependent and it is hard to give *a priori* bounds. In this section, we study the values for K and KI with different number of robots and the size of graph. We show through extensive simulations that: (a) our algorithm is scalable to the number of robots (tasks) and the size of the maps. (b) Our algorithm is more efficient than enumerating all extreme points to obtain the optimal solution. (c) Our distributed algorithm is efficient and the solution is nearly optimal.

The simulations were done on computer with Intel i7 2.60GHZ CPU and 16G RAM. We assume the number of robots is same with the number of tasks. The desired probability p in chance constraint is 99% and ϵ of auction iteration in Alg. 2 is 10. We create different instances with randomly generated means and variance for edge cost, i.e. ${}^{i}\mu_{uv}, {}^{i}\sigma_{uv}^2$. The means are generated from a continuous uniform distributions $\mathcal{U}(20, 100)$ and variances are generated from continuous uniform distribution with different magnitude of the range so that the edge with higher mean also tends to have a higher variance.

We compare three algorithms (shown in plot (a) of Fig. 2 and 3) : (1) Distributed algorithm presented in this paper that implements the first step of our method. It produces the approximate solution. The results are represented by red line. (2) Centralized algorithm that implements our two-step method. It solves CC-STAP optimally. The results are represented by blue line. (3) Method that enumerates all extreme points. The solution is optimal. The results are shown by the black line. All methods solve CC-STAP by solving a sequence of risk-averse problems. We evaluate the performance by computing the number of risk-averse problems solved, namely K the number of iterations. We further evaluate the performance of our distributed algorithm by computing the number of auction iterations taken by each robot for solving CC-STAP, i.e., KI.

A. Scability to the number of robots

We study the scability of our algorithm to the number of robot varying from 20 to 100 with 20 increment. Robots are working on a graph with 500 nodes and 8470 edges. The results are provided in Fig. 2. For a certain number of robots (x coordinate) in plot (a), we compute the average number of iterations taken by three methods from 100 instances with randomly generated ${}^{i}\mu_{uv}, {}^{i}\sigma_{uv}^2$. The black line has the highest value and growth rate. The value for our two-step algorithm increases slowly while the value for our distributed algorithm is nearly constant (less than 4). The relative difference of distributed algorithm is less than 3% for all numbers of robots (could reduce to order of 1×10^{-4} with smaller ϵ at the cost of higher number of auction iterations). In plot (b), we further present average (red) and maximum (blue) number of auction iterations taken by each robot over 100 instances. The average number grows linearly.

It implies that the search region (see Fig. 1 (b)) obtained from Alg. 1 is small and excludes a large percentage of extreme points. Our algorithms solve less number of risk-averse problems than enumerating all extreme points and is scalable to the number of robots. Further, our distributed algorithm produces a good approximate solution with the smallest number of iterations.

B. Scability to the size of the map

We also study the scability of our algorithms to the number of nodes in graph varying from 500 to 2500 with

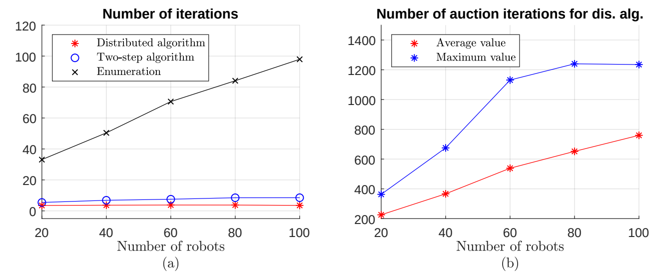

Fig. 2. (a) The average number of risk-averse problems solved and (b) the total number of auction iterations for each robot solving CC-STAP with different number of robots

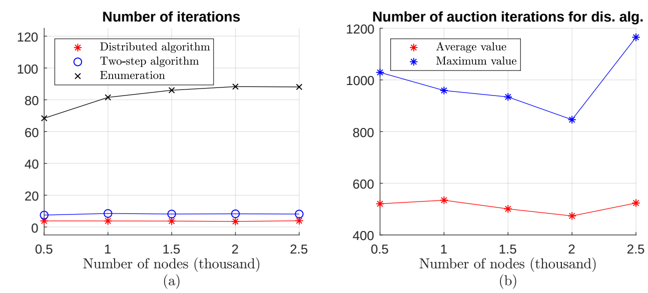

Fig. 3. (a) The average number of risk-averse problems solved and (b) the total number of auction iterations for each robot solving CC-STAP with different number of nodes in graph

500 increment. The number of edges increases from about 8000 to 27000 accordingly. The number of robots is 60. The results are presented in Fig. 3. For certain number of nodes in plot (a), we compute the average number over 100 instances under different graphs having same number of nodes and randomly generated ${}^{i}\mu_{uv}, {}^{i}\sigma_{uv}^2$. The value for our distributed algorithm and two-step algorithm are nearly constant while the number for black line grows slowly. The relative difference for the distributed algorithm is less than 2%. In plot (b) we present both average (red) and maximum (blue) number of the auction iterations for each robot.

The results show that the number of nodes does not have a great influence on the number of iterations and auction iterations. It influences more on the complexity of Dijkstra solving shortest path problems than the efficiency of our way of generating risk-averse problems.

VII. Summary

We presented a novel algorithm for solving chance constrained simultaneous task assignment and path planning on a graph (or roadmap) with stochastic edge costs. We proved that CC-STAP can be solved optimally by solving a sequence of deterministic STAP. We proved that the deterministic STAP can be solved optimally by a linear assignment problem with cost equal to the shortest path to the task location. We also present a distributed algorithm to solve CC-STAP based on the auction algorithm. The simulation results show that both our distributed algorithm and centralized two-step algorithm is scalable with the number of robots and the size of the graph.

REFERENCES

[1] D. P. Bertsekas. The auction algorithm: A distributed relaxation method for the assignment problem. *Annals of Operations Research*, 14:105–123, 1988.

[2] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Vol. II*. Athena Scientific, 3rd edition, 2007.

[3] J. Boyan, M. Mitzenmacher, and M. Mitzenmacher. Improved results for route planning in stochastic transportation. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 895–902, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[4] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[5] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, Aug 1996.

[6] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97, March 1955.

[7] S. M. Lavalle, J. J. Kuffner, and Jr. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and Computational Robotics: New Directions*, pages 293–308, 2000.

[8] S. Lim, H. Balakrishnan, D. Gifford, S. Madden, and D. Rus. Stochastic motion planning and applications to traffic. *The International Journal of Robotics Research*, 30(6):699–712, 2011.

[9] S. Lim and D. Rus. Stochastic motion planning with path constraints and application to optimal agent, resource, and route planning. In *2012 IEEE International Conference on Robotics and Automation*, pages 4814–4821, May 2012.

[10] S. Lim, C. Sommer, E. Nikolova, and D. Rus. Practical route planning under delay uncertainty: Stochastic shortest path queries. In *Robotics: Science and Systems*, pages 249–256, United States, 1 2013. MIT Press Journals.

[11] E. D. Miller-Hooks and H. S. Mahmassani. Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science*, 34(2):198–215, 2000.

[12] J. Mote, I. Murthy, and D. L. Olson. A parametric approach to solving bicriterion shortest path problems. *European Journal of Operational Research*, 53(1):81–92, 1991.

[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. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 338–351, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[15] E. Nikolova, J. A. Kelner, M. Brand, and M. Mitzenmacher. Stochastic shortest paths via quasi-convex maximization. In *Proceedings of the 14th Conference on Annual European Symposium - Volume 14*, ESA'06, pages 552–563, London, UK, UK, 2006. Springer-Verlag.

[16] S. Pallottino and M. G. Scutellà. Shortest path algorithms in transportation models: Classical and innovative aspects. In *Equilibrium and Advanced Transportation Modelling*, pages 245–281, Boston, MA, 1998. Springer US.

[17] D. Panagou, M. Turpin, and V. Kumar. Decentralized goal assignment and trajectory generation in multi-robot networks: A multiple lyapunov functions approach. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6757–6762, May 2014.

[18] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84(1):127–150, 1991.

[19] G. H. Polychronopoulos and J. N. Tsitsiklis. Stochastic shortest path problems with recourse. *NETWORKS*, 27:133–143, 1996.

[20] A. Prorok. Redundant robot assignment on graphs with uncertain edge costs. In *Distributed Autonomous Robotic Systems*, pages 313–327. Springer International Publishing, 2019.

[21] M. Turpin, N. Michael, and V. Kumar. Trajectory planning and assignment in multirobot systems. In E. Frazzoli, T. Lozano-Perez, N. Roy, and D. Rus, editors, *Algorithmic Foundations of Robotics X*, pages 175–190, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[22] M. Turpin, N. Michael, and V. Kumar. Capt: Concurrent assignment and planning of trajectories for multiple robots. *The International Journal of Robotics Research*, 33(1):98–112, 2014.

[23] 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.

[24] F. Yang and N. Chakraborty. Multirobot simultaneous path planning and task assignment on graphs with stochastic costs (extended abstract). In *International Symposium on Multi-Robot and Multi-Agent Systems*, New Brunswick, New Jersey, August 2019.

[25] M. M. Zavlanos, L. Spesivtsev, and G. J. Pappas. A distributed auction algorithm for the assignment problem. In *Proc. 47th IEEE Conf. Decision and Control*, pages 1212–1217, 2008.