An Anytime Algorithm for Chance Constrained Stochastic Shortest Path Problems and Its Application to Aircraft Routing

Sungkweon Hong[1], Sang Uk Lee[1], Xin Huang[1], Majid Khonji[2], Rashid Alyassi[2], Brian C. Williams[1]

***Abstract*— Aircraft routing problem is a crucial component for flight automation. Despite recent successes, challenges still remain when the environment is dynamic and uncertain. In this paper, we tackle the following two challenges. First, when the environment is uncertain, it is much safer if the route planner can guarantee a specified level of safety. Second, when the environment is dynamic, the planner needs to adapt to the changes in the environment quickly. To address these challenges, we present three contributions. First, we propose formulating the aircraft routing problem under a dynamic and uncertain environment as a chance constrained stochastic shortest path (CC-SSP) problem. Second, we introduce an anytime algorithm for the CC-SSP problem, which is effective in a dynamic environment with limited planning time. To be more specific, we present two versions of the algorithm and compare their performances. Third, we show that the algorithm can be generalized to solve a larger class of problems called chance constrained partially observable Markov decision process (CC-POMDP).**

I. Introduction

Recent advances in robotics and automation have made autonomous flight navigation possible. Flight route planning is a crucial component for autonomous flight navigation. The goal of flight route planning is to find a path (Fig. 1a) or a policy (Fig. 1b) that takes the plane from the start location to the goal location. A good route planning system would not only make the flight safer [1], but also minimize flight delays and revenue loss for airlines [2]. During route planning, we need to take the navigation environment into account. For example, we might want a route that avoids the region with convective weather, because the atmospheric status in the region might make the flight highly unstable.

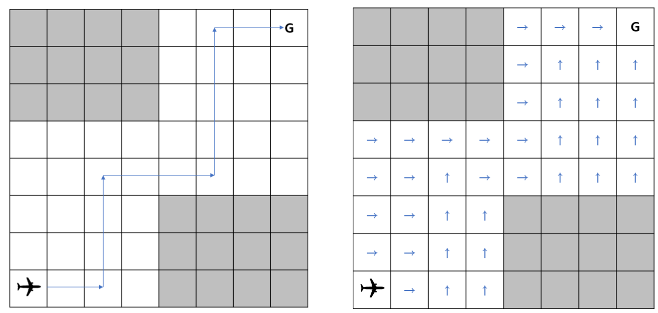
In fact, similar problems arise in many other application domains frequently. For example, a Mars rover exploring a given territory might want to avoid regions with high slip to avoid getting stuck in sand [3], [4]. Autonomous cars might want to avoid entering certain streets due to extreme traffic congestion [5], [6], or lanes occupied by other agents to maintain safety [7].

The route planning problem is nontrivial when the environment is dynamic and uncertain. For example, in autonomous flight navigation, the regions of convective weather drift frequently, and models that can anticipate how it would drift are highly inaccurate. This is because the atmospheric status of a region is affected by many factors, and the dynamics of weather forecast is highly nonlinear [8]. In fact, similar problems arise for autonomous cars trying to avoid streets with extreme traffic congestion, because it is extremely difficult to have an accurate model that anticipates traffic congestion [9].

The work was partially supported by the Boeing Corporation, under grant 6943358, and the Khalifa University, under grant reference KKJRC-2019-Trans1.

[1]Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, MA 01239, USA {sk5050,sangukbo,xhuang,williams}@csail.mit.edu

[2]Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, UAE {majid.khonji,rashid.alyassi}@ku.ac.ae



(a) Path (b) Policy

Fig. 1: An example route planning problem, in which the goal is to find a path (a) or a policy (b) to the goal location that avoids region with convective weather denoted in the shaded area.

Traditionally, the route planning problem has been formulated as a shortest path problem. Works in [10], [11] have used Dijkstra or A* algorithm to solve the problem, which are only suitable for a simple environment. In order to account for a dynamically changing environment, works in [12], [13] have developed variants of A* algorithm. However, they had not considered the stochastic behavior of the environment. In order to consider uncertainty, some works formulated the route planning problem as a stochastic shortest path (SSP) problem and computed a policy that would take the agent, such as a plane, to the goal [14], [15].

However, the previous works that have formulated the routing problem as a SSP problem had the following issues. First, they do not have a guarantee on how well the planned policy would avoid the regions with convective weather [14], [15]. In the SSP problem, a policy that never enters the undesirable regions is unrealistic and highly inefficient due to uncertainty. The previous works tried to steer the agent away from the undesirable regions by adding an extra cost term that penalizes the plan entering the region, which does not provide a guarantee on the performance. In safety-critical tasks such as aircraft routing, it might be more desirable to have a policy with a strict guarantee. Second, the previous works are not suitable in dynamic environments, in which the route planner would keep receiving updates on the changes in the environment [14], [15]. This requires the planner to compute a new policy fast enough based on the updated environment. The previous works, however, take a long time to compute an updated policy, and thus are not suitable in real time applications.

In this paper, we propose an anytime algorithm to address the route planning problem under dynamic and uncertain environments with a performance guarantee. We provide three main contributions. First, we formulate the routing problem as a chance constrained stochastic shortest path (CC-SSP) problem. Compared to the previously introduced SSP problem formulation, the CC-SSP problem formulation guarantees the planned policy to avoid entering the undesirable regions with a certain performance level as a probability that can be specified by the user. For instance, we can specify that the planned policy has to avoid entering the undesirable regions with at least 95% probability in the CC-SSP formulation. Second, we propose an anytime algorithm to find an optimal deterministic policy for the CC-SSP problem. An anytime algorithm has two properties: i) it outputs an initial feasible policy very fast and ii) it computes an optimal policy given enough time by iteratively improving the initial policy. The anytime property is highly desirable when the environment is dynamic – When the route planner needs to compute for a new policy due to a changed environment, an anytime algorithm first outputs an initial policy fast so that the agent can start following. Then, as time permits, the policy would keep getting better until it converges to the optimal. Third, we show that the proposed algorithm can be applied to a more general set of problems called chance constrained partially observable Markov decision process (CC-POMDP) [16]–[18] with only small modifications. This would make our algorithm useful for problems other than the route planning problem.

We would like to further emphasize the third contribution. In fact, to the author's knowledge, no existing algorithms for CC-POMDP achieved the anytime property. A work in [16] introduced an algorithm called RAO*, which computes for a solution for CC-POMDP problem fast, but without the guarantee on the optimality of the solution. In [17], the CC-POMDP problem is formulated as a mixed integer linear programming (MILP) problem and solved through MILP solvers. Though the MILP-based solution for CC-POMDP problems is optimal, it takes a long time to compute and is not applicable in real time systems. Several works have solved for a relaxed problem that allows the computed policy to be stochastic [19]–[21]. As a result, the stochastic policy would steer the plane based on some probability distribution over possible actions. Such random behavior is not desirable in safety-critical scenarios. This motivates us to generate a deterministic policy, which is considered to be a more difficult problem, as pointed out in [17], [22].

This paper is organized as follows. In Section II, we provide some backgrounds. The problem formulation of the route planning problem as a CC-SSP problem and its anytime algorithm is presented in Section III. In Section IV, we present our experimental results. The paper is concluded in Section V.

II. Backgrounds

A. Stochastic Shortest Path Problem

A SSP is a tuple $\mathcal{P} = \langle \mathcal{S}, \mathcal{A}, T, C, s_0, G, H \rangle$ [23] in which $\mathcal{S}$ is a set of discrete states; $\mathcal{A}$ is a set of discrete actions; $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition function, where $T(s, a, s') = Pr(s'|s, a)$ is the probability of being in state s' after executing action a in state s; $C : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the cost function, where $C(s, a)$ is the cost of executing action a in state s; s_0 is the initial state; $G \subset \mathcal{S}$ is a set of goal states; H is a planning horizon.

A policy π is defined as a mapping from state and time step to an action, i.e., $\pi(s, k) = a$. Then, the problem of the SSP is to find an optimal policy π^* that minimizes the expected cost up to a fixed horizon H, i.e.,

$$\pi^* = \operatorname*{argmin}_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{H-1} \overline{C}(s_k, a_k) \middle| s_0, \pi \right], \tag{1}$$

where Π is a set of all policies and $\overline{C}(s, a)$ is an augmented cost function which is defined as follows:

$$\overline{C}(s_k, a) = \begin{cases} C(s_k, a), & \text{if} \quad k < H - 1, \\ C(s_k, a) + h_V(s_{k+1}), & \text{if} \quad k = H - 1, \end{cases}$$

where $h_V(\cdot)$ is an admissible heuristic function. Note that $h_V(\cdot)$ can be set as 0 when it is not available.

B. AND/OR Heuristic Forward Search

A SSP can be represented as a tree [24][1] that can be solved by an AND/OR heuristic forward search methods such as AO* [25], [26]. The root node of the tree is the pair $(s_0, 0)$ where s_0 is the initial state and 0 is the initial time step. Starting from the root, the tree interleaves OR node and AND node. Each OR node is represented as a pair (s, k) that can choose from different actions. On the other hand, each AND node is represented as a triplet (s, a, k) that expands all possible transitions by applying the action a from s. The terminal nodes are the OR-nodes with a goal state or at the planning horizon, i.e., (s, k) such that $s \in G$ or $k = H$.

A policy of a SSP is a sub-tree with $(s_0, 0)$ as the root node, in which each OR-node activates one of its children and each AND-node activates all of its children, recursively. The value of a policy can be computed by backward induction from the terminal nodes. The value of the terminal nodes are equal to heuristic function value $h_V(s)$, i.e., $V(s, k) = h_V(s)$ if $s \in G$ or $k = H$. Then the value of an AND-node that has children as OR-nodes can be computed by a weighted sum of children values as follows:

$$V(s, a, k) = \sum_{s' \in \mathcal{S}} Pr(s'|s, a) \cdot V(s', k + 1). \tag{2}$$

[1]In general, a SSP can be represented as a graph if two nodes from different ancestors coincide. In this paper, we assume that this does not occur for simplicity. Our method can also be applied to the graph representation with simple modifications, as we intend to show in future work.

The value of an OR-node is set as equal to its child AND-node, based on the choice of the action. Using the backward induction in this way, the value of the $V(s_0, 0)$ can be computed as the value of the policy.

To find an optimal policy, the AO* algorithm starts from the root node and alternates between two main steps: expansion and backup. In the expansion step, it expands one of the tip nodes in the current best policy. In the backup step, it updates values for the expanded node and its ancestors with the backward induction. For each backup step, different actions are evaluated, and the best actions with minimal expected cost are marked for the OR-nodes. The new best policy is then updated by following the marked actions from the root node. Finally, the algorithm terminates when all the tip nodes in the current best policy are terminal nodes. This method guarantees optimality, given that the heuristic function is admissible, similar to the A* algorithm [26].

C. Chance Constraint

Chance constrained approach has been widely studied in various domains including operations research, motion planning, control, to name a few [3], [27], [28]. One of the major advantages of the chance constrained approach over the others, such as the penalty-based method [14], is that the user can define the level of safety guarantees and thus systematically balance the efficiency and the risk by specifying the probability of success of the plan.

Within the SSP framework, the useful notion of chance constraint was introduced in [16], [29]. Given a set of constrained states $\overline{\mathcal{S}} \subset \mathcal{S}$, the execution risk er is defined as the risk of violating a constraint when following a policy π up to horizon H from the initial state s_0. It is defined formally as follows:

$$er(s_0|\pi) = 1 - Pr\left(\bigwedge_{i=0}^{H} Sa_i = 1 \middle| s_0, \pi \right), \tag{3}$$

where Sa_i is a Bernoulli random variable with value 1, when an agent has not violated constraints at time i. Then we can enforce the chance constraint with the following constraint:

$$er(s_0|\pi) \leq \Delta, \tag{4}$$

where Δ is a user-specific parameter for allowable risk.

III. Dynamic Routing Under Convective Weather Conditions

Commercial flight operation is based on a flight plan, which designates procedures and air routes the flight should follow to accomplish its flight task [30]. However, there are several situations where an aircraft has to deviate from the flight plan. One of the most common such situations is when there is a convective weather cell around the planned route of the flight.

Due to the uncertainty in weather forecast, air traffic controllers tend to give conservative alternative path guidance for safety, which usually results in loss of efficiency. To balance safety and efficiency, there have been several research efforts to systematically generate alternatives given weather forecast [14], [31]. Although those efforts enable air traffic controllers and/or flight crews to find more efficient alternatives while maintaining safety, the methods are not able to specify the level of safety, which is crucial in air traffic management [32].

In this section, we provide an alternative model for the aircraft routing problem under probabilistic weather forecast which specifies the level of safety as a chance constraint.

A. Problem Description

The problem of aircraft routing under convective weather conditions is explained with the exemplary scenario shown in Fig. 2, in which an aircraft plans to travel from its current position to the goal waypoint. There are predefined waypoints that aircraft can navigate which are shown with triangle markers in Fig. 2. In addition, there is a weather cell shown with a red polygon in the figure which drifts by time stochastically.

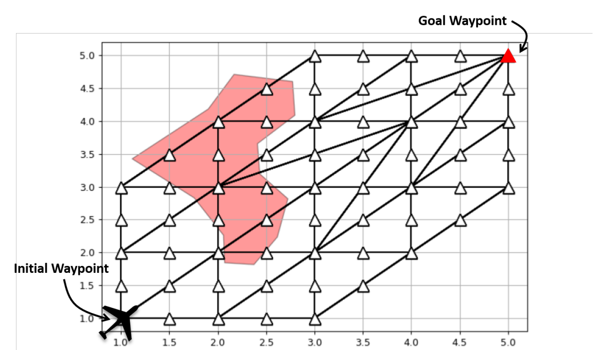


Fig. 2: An exemplary aircraft routing scenario under convective weather condition. The waypoints are denoted as triangle markers, and a stochastic weather cell is dentoed in red. The initial and goal waypoints are marked respectively.

The described aircraft routing problem can be modeled as CC-SSP as follows.

1) State Space $\mathcal{S}$: A state is defined as a tuple, including aircraft position and heading, and position of a convective weather cell. The positions of an aircraft are limited to waypoints defined in the airspace of interest.
2) Action Space $\mathcal{A}$: Available actions for a state includes every movements to adjacent waypoints that are reachable from the current aircraft position and heading.
3) State Transition Function T: State transitions consist of transition for an aircraft and a convective weather cell. Since the magnitude of navigational error is negligible in comparison with the magnitude of stochasticity in the weather forecast, we assume the state transition of an aircraft is deterministic. The transition of the weather cell depends on the prediction and the duration. We discretize the weather prediction with cardinal directions to provide discrete state transitions.
4) Cost Function C: The cost function is proportional to the length of the flight segment given an action applied in a certain state.

5) Set of Constrained States $\overline{\mathcal{S}}$: A set of constrained states consists of the states where an aircraft position is within a weather cell.

The initial state s_0, goal state G, horizon H, heuristic cost function h_V and the risk bound parameter Δ are problem specific. Hence, they will be described in the experimental results section.

Finally, we want to find a policy for the aircraft from its current waypoint to the goal waypoint which minimizes cost function

$$f(\pi) = \mathbb{E}\left[\sum_{k=0}^{H-1} \overline{C}(s_k, a_k)\middle| s_0, \pi\right] \tag{5}$$

while maintaining the risk of entering the convective weather cell less than or equal to Δ by satisfying the constraint

$$g(\pi) = er(s_0|\pi) - \Delta \leq 0. \tag{6}$$

B. Anytime Algorithms

In this section, we provide two anytime algorithms for a CC-SSP.

1) k-best Enumeration: A proposed k-best enumeration method finds an optimal deterministic chance constrained policy with two successive stages. In the first stage, the method uses AO* to find an optimal solution $\overline{\pi}^*$ with respect to the modified cost function which is defined as follows:

$$\overline{f}(\pi) = f(\pi) + \lambda \cdot g(\pi), \tag{7}$$

where λ is a sufficiently large positive constant. By optimizing Eq. (7), we can find a conservative but feasible solution.

In the second stage, the method takes as input the first solution $\overline{\pi}^*$ then finds k-best solutions in terms of the modified cost function in Eq. (7). Suppose the k-th solution found during the second stage is the first solution which violates the chance constraint in Eq. (6). Then, it is guaranteed that the current incumbent solution is an optimal chance constrained solution with respect to the original problem.

The proposed algorithm has its importance especially in risk-sensitive online planning applications such as aircraft routing. In the first stage of the algorithm, it finds a conservative but safe policy first to enable an agent to safely execute a mission. On the other hand, in the second stage, the method keeps updating the solution that enables an agent to navigate more efficiently given extra planning time.

The algorithm is outlined in Algorithm 1. The inputs are the initial solution $\overline{\pi}^*$, the lower (LB) and upper bounds (UB) of the original cost values, where they are initialized as the modified cost function value of the initial solution $\overline{f}(\overline{\pi}^*)$ and the original cost function value of the initial solution $f(\overline{\pi}^*)$, respectively. The algorithm returns immediately in the following two cases. First, if the initial solution is infeasible ($g(\overline{\pi}^*) > 0$), the problem is infeasible. Second, if $LB = UB$, $\overline{\pi}^*$ becomes a true optimal solution. Otherwise, the algorithm iteratively finds the next best solution until the termination condition $LB \geq UB$ is satisfied. Note that $\overline{f}_k$ and f_k are the modified and the original cost function values of the k-th best policy, respectively. Hence the first time $LB \geq UB$ is satisfied is the time when we found the first solution which violates the chance constraint in Eq. (6).

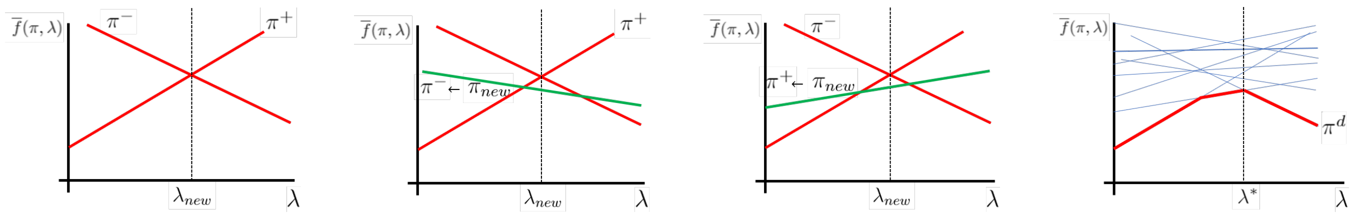
Algorithm 1: Anytime Algorithm for CC-SSP

Input: $\overline{\pi}^*, LB, UB$
if $g(\overline{\pi}^*) > 0$ **then**
  return infeasible
else if $LB = UB$ **then**
  $\pi^* \leftarrow \overline{\pi}^*$
  return π^*
else
  $k = 1, \Psi = \{\overline{\pi}^*\}$ and $\Gamma = \{\}$
  while *True* **do**
    if $LB \geq UB$ **then**
      $\pi^* \leftarrow \overline{\pi}^*$
      return π^*
    else
      $k \leftarrow k + 1$
      $(\pi_k, \overline{f}_k, f_k, g_k), \Psi, \Gamma \leftarrow$ Next-Best-Policy(k, Ψ, Γ)
      $LB \leftarrow \overline{f}_k$
      if $g_k \leq 0$ *and* $f_k < UB$ **then**
        $UB \leftarrow f_k$ and $\overline{\pi}^* \leftarrow \pi_k$

Algorithm 2: Next-Best-Policy

Input: index k, best policies set Ψ, candidates set Γ
for $(s, k) \in \pi_{k-1}$ **do**
  $\bar{\mathcal{A}} = \{\}$
  for $\pi_i \in \Psi$ **do**
    if $\pi_i^{root}((s,k)) = \pi_{k-1}^{root}((s,k))$ **then**
      $\bar{\mathcal{A}} \leftarrow \bar{\mathcal{A}} \cup \{\pi_i((s,k))\}$
  $\mathcal{A}'((s,k)) \leftarrow \mathcal{A}((s,k)) \backslash \bar{\mathcal{A}}$
  Run AO* with $\mathcal{A}'((s,k))$ to compute $\pi^{\text{spur}}((s,k))$
  Join $\pi_{k-1}^{\text{root}}((s,k))$ and $\pi^{\text{spur}}((s,k))$ to compute π_{new}, $\overline{f}_{\text{new}}$, f_{new} and g_{new}
  $\Gamma \leftarrow \Gamma \cup \{(\pi_{\text{new}}, \overline{f}_{\text{new}}, f_{\text{new}}, g_{\text{new}})\}$
$(\pi_k, \overline{f}_k, f_k, g_k) \leftarrow$ pop(Γ)
$\Psi \leftarrow \Psi \cup \{\pi_k\}$
return $(\pi_k, \overline{f}_k, f_k, g_k), \Psi, \Gamma$

The crucial part of the second stage is finding the k-th best solution given a set of $(k-1)$-best solutions. We extend the Yen's algorithm [33], which is summarized in the Algorithm 2. Let Ψ is a set of $(k-1)$-best policies and Γ is a set of candidate policies that we found so far. Note that Γ is a priority queue sorted by the modified cost function value $\overline{f}(\pi)$. The algorithm starts with the last best policy $\pi_{k-1} \in \Psi$, and updates Γ by including the deviated policies from π_{k-1}. To find each deviation, it selects an OR node (s,k) in π_{k-1} and collects all the actions that have already been selected at (s,k) from the policies in Ψ with the same root policy, i.e., $\pi_i^{\text{root}}((s,k)) = \pi_{k-1}^{\text{root}}((s,k))$. Note



(a) The process starts with two initial solutions π^- and π^+.

(b) Update π^- if π_{new} is feasible.

(c) Update π^+ if π_{new} is infeasible.

(d) The process is terminated if λ_{new} is unchanged.

Fig. 3: Graphical explanation of the process of solving Lagrangian dual problem.

that $\pi_i^{\text{root}}((s,k)) = \pi_i \backslash \pi_i^{\text{spur}}((s,k))$ where $\pi_i^{\text{spur}}((s,k))$ is a sub-policy of π_i starting with (s,k). Then it runs AO* by excluding the actions previously taken from any policy with the same root policy to find a deviated sub-policy $\pi^{\text{spur}}((s,k))$ starting at node (s,k). Then a new candidate can be generated by joining $\pi_{k-1}^{\text{root}}((s,k))$ and $\pi^{\text{spur}}((s,k))$. After generating all the candidates from π_{k-1}, the next k-th best policy can be obtained by selecting the best policy from the candidates set with respect to the modified cost function value.

2) Lagrangian Dual Based Enumeration: Although k-best enumeration method enables us to find an optimal solution in an anytime fashion, there are several limitations. First, the initial solution tends to overly conservative with a large constant λ. Second, the algorithm has to search all of the feasible solutions in the second stage before the termination condition is satisfied. To overcome these limitations, we propose to modify the k-best enumeration method based on a Lagrangian dual method.

In the first stage, instead of taking λ as a large constant as in Eq. (7), we consider λ as a variable, which is in fact a *Lagrangian variable* λ, i.e.,

$$\overline{f}(\pi,\lambda) = f(\pi) + \lambda \cdot g(\pi). \tag{8}$$

Then we solve the following:

$$\lambda^* = \arg\max_{\lambda \geq 0} \Big[\min_{\pi \in \Pi} \overline{f}(\pi,\lambda) \Big]. \tag{9}$$

Ep. (9) is known as Lagrangian dual problem and we can solve for an associated policy π^d which is feasible unless the problem is infeasible [34].

Given π^d, the Algorithm 1 remains the same except that $\overline{\pi}^*$, LB and UB are initialized as π^d, $\overline{f}(\pi^d, \lambda^*)$ and $f(\pi^d)$, respectively. Also $\overline{f}_k$ is now $\overline{f}(\pi_k, \lambda^*)$, where π_k is k-th best solution.

Although solving a Lagrangian dual problem in Eq. (9) is not as simple as optimizing Eq. (7), there are a wide variety of methods available. In particular, since Eq. (8) is a Lagrangian function with a single Lagrangian variable, Eq. (9) can be efficiently solved with bisection-like method as proposed in [35], which is illustrated in Fig. 3. As shown in Fig. 3a, we start with two initial solutions π^+ and π^-, which can be found by minimizing Eq. (8) with $\lambda = 0$ and $\lambda = M$, respectively, where M is a large constant. Note that, if π^- is infeasible, then the problem is infeasible. Given π^+ and π^- and their corresponding function and constraint values f^+, g^+, f^- and g^-, we update λ as the intersection of two solutions, i.e., $(f^- - f^+)/(g^+ - g^-)$. Then we solve for π by minimizing Eq. (8) with updated λ. New π can be either feasible or infeasible, which result in updating π^- (Fig. 3b) or π^+ (Fig. 3c), respectively. By iterating this process, we can finally find the λ^* and corresponding feasible solution π^d (Fig. 3d).

IV. Experimental Results

In this section, we present the experimental results to validate the proposed method for the aircraft routing problem under convective weather conditions. In addition, we show the potential application of the proposed method to a more general class of problems by applying it to a CC-POMDP problem.

A. Aircraft Routing Under Convective Weather Condition

The configuration of the airspace and the weather condition are shown in Fig. 2, which also shows the initial state of the problem. The goal is a set of states where the aircraft is located at the goal waypoint. In addition, the euclidean distance was used as an admissible cost heuristic. The convective weather cell drifts stochastically, with the distribution shown in Table I.

dx/dt	dy/dt	Prob.
-0.2	0	0.8
-0.2	-0.2	0.1
0	-0.2	0.1

TABLE I: Probability distribution of weather cell drift.

The proposed Lagrangian dual based enumeration method is compared with the penalty based method which is used in [14]. Instead of having chance constraint, the penalty based method adds a penalty term to the cost function if a state violates a constraint. Note that this is mathematically equivalent to finding an optimal solution based on Eq. (7) with some fixed λ. For each method, 500 number of simulations have been conducted, where the planning and execution are interleaved with planning horizon 6. For every simulations, the chance constraint probability Δ was set as 0.2.

Table II summarizes the experimental results. The second column shows the λ values for the penalty based method. The third and the fourth columns show the percentage of the cases with constraint violations and the average accumulated cost until the goal is reached, respectively. As shown in the table, the penalty based method could obtain better cost than the proposed method in cases where $\lambda = 0.8, 1.2$. In those cases, however, the method weighted cost term much more than the risk term, which results in higher failure rates compared to the desired chance constraint probability 0.2. With $\lambda = 1.5, 1.8$, the penalty based method failed less than 20% of the time, but the costs were higher than the proposed method. This result shows that the proposed method could outperform penalty based method in both safety and efficiency. Moreover, the λ value has to be heuristically selected for the penalty based method, which takes extra effort, but it is not an issue for our method.

	λ	Failure Rate	Avg. Cost
Penalty Method	0.8	88.0%	5.82
	1.2	46.0%	6.02
	1.5	18.0%	6.20
	1.8	12.0%	6.21
Lagrangian Method		10.0%	6.13

TABLE II: Summary of experimental results of aircraft routing under convective weather condition.

In addition to the overall experimental results, Fig. 4 presents an ablation study on the anytime solution histories between the k-best and Lagrangian dual based enumeration methods, visualized in blue circles and red squares, respectively. As shown in the figure, the convergence rate of the Lagrangian dual based enumeration method was much faster than the k-best enumeration, which demonstrates the advantage of the proposed method empirically.

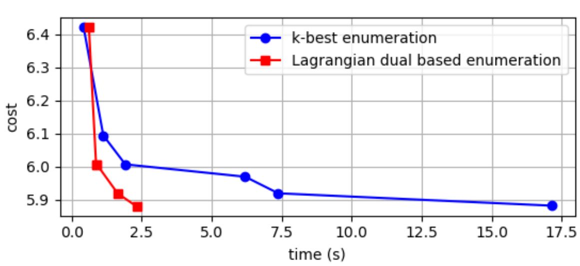


Fig. 4: k-best vs. Lagrangian dual based enumeration, in terms of cost history evolved over time.

B. CC-POMDP Benchmark

Although the proposed algorithm was applied to the CC-SSP problem, the algorithm can be applied to a more general class of problems that can be represented as AND/OR tree, such as chance constrained partially observable Markov decision process (CC-POMDP) [16], [17], [24]. To demonstrate the application of the proposed algorithm for CC-POMDP, we used *Paint Problem* which is a publicly available benchmark problem in `pomdp.org`. To modify the *Paint Problem* as chance constrained problem, we define the constraints as a set of states where an agent ships a part with fault or rejects a part without fault. For the demonstration purpose, we compared the proposed method with MILP based method which is one of the state-of-the-arts proposed in [17]. Note, the MILP was solved using CPLEX 12.9

Table III summarizes the evaluation results. The first two columns show the planning horizon and the chance constraint probability of each case. For each case, we show the results for both the proposed algorithm and a MILP method. For the baseline method, optimal value and computation time in seconds are reported. The computation time includes both time for expanding the tree and the CPLEX solving time, and the time-out was set to 7200 seconds.

H	Δ	Dual based enumeration method				MILP	
		$< 5\%$		$< 1\%$		val.	t_{MILP} (s)
		val. (opt gap)	t/t_{MILP}	val. (opt gap)	t/t_{MILP}		
9	0.1	1.08 (2.42%)	0.65	-	-	1.10	11.34
	0.2	1.25 (4.62%)	0.46	1.30 (0.79%)	0.51	1.31	17.65
10	0.1	1.14 (4.59%)	0.35	1.18 (0.65%)	0.37	1.19	170.25
	0.2	1.42 (0.40%)	0.30	1.42 (0.40%)	0.30	1.42	181.55
11	0.1	1.24 (3.52%)	0.57	1.28 (0.79%)	0.58	1.29	1042.94
	0.2	1.50 (2.01%)	0.48	1.52 (0.38%)	0.48	1.53	1219.72

TABLE III: Evaluation results of our proposed method compared to a MILP-based method.

Similarly, for the proposed algorithm, the value of the solution and computation time are presented. To show the anytime history, two different solutions are summarized, which obtain 5%, and 1% optimality gaps, respectively. The optimality gap is indicated in the parenthesis next to the value of the solution, and computation time is shown as a proportion to the MILP computation time, to demonstrate the speeding up of our algorithm. The time-out was set as the baseline computation time, and a "-" sign in a cell indicates a time-out.

As shown in the table, the proposed algorithm could obtain a near-optimal solution, with a less than 5% ($< 5\%$) optimality gap, much faster than the MILP-based method in all problem settings. The proposed algorithm could improve the solutions with a less than 1% ($< 1\%$) optimality gap with only marginal additional time for most of the cases. The results show that even the MILP based method is exact and optimal, the proposed method can benefit by finding anytime solutions especially with time limited planning domains.

V. Conclusion

In this paper, we proposed formulating the aircraft routing problem under a dynamic and uncertain environment as a CC-SSP problem. The CC-SSP formulation is useful in risk-sensitive uncertain environments, because it can guarantee a specified level of safety. Next, we introduced an anytime algorithm for the CC-SSP formulation, which is effective in a dynamic environment with limited planning time. Finally, we generalized the algorithm to a larger class of problem called CC-POMDP.

Future efforts could focus on explaining how fast the initial solution converges in the anytime algorithm. Although our experimental results show the fast convergence rate of the algorithm empirically, we expect a theoretical analysis on the convergence rate can be useful in many applications.

REFERENCES

[1] L. F. Vismari and J. B. C. Junior, "A safety assessment methodology applied to cns/atm-based air traffic control system," *Reliability Engineering & System Safety*, vol. 96, no. 7, pp. 727–738, 2011.

[2] X. Diao and C.-H. Chen, "A sequence model for air traffic flow management rerouting problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 110, pp. 15–30, 2018.

[3] S. U. Lee, R. Gonzalez, and K. Iagnemma, "Robust sampling-based motion planning for autonomous tracked vehicles in deformable high slip terrain," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2569–2574.

[4] S. U. Lee and K. Iagnemma, "Robust motion planning methodology for autonomous tracked vehicles in rough environment using online slip estimation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3589–3594.

[5] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA urban challenge: autonomous vehicles in city traffic*. springer, 2009, vol. 56.

[6] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann *et al.*, "Stanley: The robot that won the darpa grand challenge," *Journal of field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[7] X. Huang, A. Jasour, M. Deyo, A. Hofmann, and B. C. Williams, "Hybrid risk-aware conditional planning with applications in autonomous vehicles," in *Decision and Control (CDC), 2018 IEEE International Conference on*. IEEE, 2018.

[8] M. Wolfson, B. Forman, K. Calden, W. Dupree, R. Johnson Jr, R. Boldi, C. Wilson, P. Bieringer, E. Mann, and J. Morgan, "Tactical 0-2 hour convective weather forecasts for faa," in *11th Conference on Aviation, Range and Aerospace Meteorology, Hyannis, MA*. Citeseer, 2004.

[9] X. Kong, Z. Xu, G. Shen, J. Wang, Q. Yang, and B. Zhang, "Urban traffic congestion estimation and prediction based on floating car trajectory data," *Future Generation Computer Systems*, vol. 61, pp. 97–107, 2016.

[10] T. Chen, G. Zhang, X. Hu, and J. Xiao, "Unmanned aerial vehicle route planning method based on a star algorithm," in *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 2018, pp. 1510–1514.

[11] A. V. Sadovsky, "Application of the shortest-path problem to routing terminal airspace air traffic," *Journal of Aerospace Information Systems*, vol. 11, no. 3, pp. 118–130, 2014.

[12] W. Hao, R. Wang, A. Krolicki, and Y. Han, "Cell a* for navigation of unmanned aerial vehicles in partially-known environments," *arXiv preprint arXiv:2009.07404*, 2020.

[13] D. S. Altner, "Resource-enabled pathfinding with mandatory waypoints and turn constraints," 2018.

[14] E. Balaban, I. Roychoudhury, L. Spirkovska, S. Sankararaman, C. S. Kulkarni, and T. Arnon, "Dynamic routing of aircraft in the presence of adverse weather using a pomdp framework," in *17th aiaa aviation technology, integration, and operations conference*, 2017, p. 3429.

[15] S. Kim, M. E. Lewis, and C. C. White, "Optimal vehicle routing with real-time traffic information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, 2005.

[16] P. Santana, S. Thiébaux, and B. Williams, "RAO*: an algorithm for chance constrained POMDPs," in *Proc. AAAI Conference on Artificial Intelligence*, 2016.

[17] M. Khonji, A. Jasour, and B. Williams, "Approximability of constant-horizon constrained POMDP," *International Joint Conference on Artificial Intelligence*, pp. 5583–5590, 2019.

[18] X. Huang, S. Hong, A. Hofmann, and B. C. Williams, "Online risk-bounded motion planning for autonomous vehicles in dynamic environments," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, no. 1, 2019, pp. 214–222.

[19] F. Trevizan, S. Thiébaux, P. Santana, and B. Williams, "Heuristic search in dual space for constrained stochastic shortest path problems," in *Twenty-Sixth International Conference on Automated Planning and Scheduling*, 2016.

[20] ——, "I-dual: solving constrained SSPs via heuristic search in the dual space," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 4954–4958.

[21] F. Geißer, G. Povéda, F. Trevizan, M. Bondouy, F. Teichteil-Königsbuch, and S. Thiébaux, "Optimal and heuristic approaches for constrained flight planning under weather uncertainty," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 30, 2020, pp. 384–393.

[22] E. A. Feinberg, "Constrained discounted markov decision processes and hamiltonian cycles," *Mathematics of Operations Research*, vol. 25, no. 1, pp. 130–140, 2000.

[23] D. P. Bertsekas and J. N. Tsitsiklis, "An analysis of stochastic shortest path problems," *Mathematics of Operations Research*, vol. 16, no. 3, pp. 580–595, 1991.

[24] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[25] N. J. Nilsson, *Principles of artificial intelligence*. Morgan Kaufmann, 2014.

[26] E. A. Hansen and S. Zilberstein, "LAO: A heuristic search algorithm that finds solutions with loops," *Artificial Intelligence*, vol. 129, no. 1-2, pp. 35–62, 2001.

[27] A. Charnes and W. W. Cooper, "Chance-constrained programming," *Management science*, vol. 6, no. 1, pp. 73–79, 1959.

[28] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[29] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. Citeseer, 2008, pp. 3427–3432.

[30] I. P.-A. Doc, "4444, atm/501, procedures for air navigation services-air traffic management, annex 6-operation of aircraft, canada, montreal, quebec: Icao, 2010."

[31] D. Isaacson, S. Harrison, K. Sheth, and C. Gong, "Atd-3 dynamic routes for arrivals in weather (draw) operational concept, v2. 0," 2019.

[32] A. C. Busch, *Methodology for Establishing A Target Level of Safety*. Federal Aviation Administraton Technical Center, 1985.

[33] J. Y. Yen, "Finding the k shortest loopless paths in a network," *management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[34] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[35] B. L. Fox and D. M. Landi, "Searching for the multiplier in one-constraint optimization problems," *Operations Research*, vol. 18, no. 2, pp. 253–262, 1970.