INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
Int. J. Robust. Nonlinear Control 2011; **21**:1421–1433
Published online 15 April 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/rnc.1722

Column generation for a UAV assignment problem with precedence constraints

David W. Casbeer*, † and Raymond W. Holsapple

Control Science Center of Excellence, Air Force Research Laboratory, WPAFB, OH 45433, U.S.A.

SUMMARY

We apply column generation with branch-and-price optimization to a multi-target, multi-task assignment problem, with precedence constraints. Column generation transforms the nonlinear program with separable costs and constraints into a linear program. This reformulation divides the original problem into a number of smaller problems, where one of these smaller problems accounts for the coupling constraints between agents and must be known by every agent. All other divisions consider local constraints affecting only one agent; these smaller problems are known by only one corresponding agent. Because of this reformulation, the assignment problem can be solved in a distributed manner. A theorem is proven which details the central analytical result of the paper, allowing a nonlinear program to be reformulated as a linear program. Simulation results for a multi-target, single-task assignment problem, as well as a multi-target, multi-task assignment problem with precedence constraints are presented. Published 2011. This article is a US Government work and is in the public domain in the USA.

Received 27 September 2010; Revised 7 February 2011; Accepted 17 February 2011

KEY WORDS: cooperative control; vehicle routing; column generation; UAV

1. INTRODUCTION

Column generation is a means of solving linear programs [1]. In this paper, we apply column generation to the multi-target, multi-task assignment problem, with precedence constraints. Currently task assignment algorithms are calculated at a central (base) location, with assignments being fed to each agent in the team. The motivation for this work is to investigate the possibility of using column generation as a distributed means of calculating optimal assignments.

Column generation is based on the Dantzig–Wolfe decomposition (DWD) [2] and permits distribution of certain linear programs depending on the constraints. Rather than optimizing a large linear program, column generation divides and conquers by repeatedly iterating between a so-called restricted master problem with coupling constraints and a series of sub-problems, each with its own respective local constraints [1, 3, 4].

One interesting property of column generation is that, through a reformulation of the problem, relaxation gives a tighter bound than the original problem (OP) without the reformulation. Furthermore, by applying the DWD, an optimization problem with nonlinear costs and constraints becomes a linear program if separability holds [5, 6]. This last property permits us to solve the task assignment problem with nonlinear costs and precedence constraints determined by Dubins' flyable paths as a linear program.

*Correspondence to: David W. Casbeer, Control Science Center of Excellence, Air Force Research Laboratory, WPAFB, OH 45433, U.S.A.
†E-mail: david.casbeer@wpafb.af.mil

Similar works, and the references therein, include [7], which is a single-assignment problem without precedence constraints, [8], which solves the problem of routing with time windows using column generation and a branch-and-price scheme, and [9], which uses column generation to solve large traffic scheduling problems.

We begin this paper by formulating the Vehicle Routing Problem with Precedence Constraints (VRPPC) in Section 2. Then we explain column generation using a simple flow problem in Section 3. This toy problem explains column generation in an intuitive fashion, since it is easy to visualize. In Section 4 we prove Theorem 1, which permits the OP to be converted into a master problem; this conversion is requisite for one to use column generation. After the relaxation of an integrality constraint, this master problem becomes the restricted master problem mentioned above. We then apply the DWD to the VRPPC and set up the column generation problem. We conclude with some illustrative examples and remarks in Sections 5 and 6.

2. VEHICLE ROUTING PROBLEM WITH PRECEDENCE CONSTRAINTS

We address the problem of assigning multiple heterogeneous UAVs to tasks with precedence constraints. Precedence constraints indicate an ordering for tasks with no specific time for completion. Mathematically this problem is written in Equations (1)–(5), and we will refer to it as the OP. Let $\mathscr{U}$ denote a team of vehicles, where each vehicle $i \in \mathscr{U}$ is subject to some resource constraints. The team of UAVs must accomplish a number of tasks in the set $\mathscr{T}$, and let $N_c = |\mathscr{T}|$, that is, N_c is the number of pending tasks. A particular UAV's assignment is enumerated by the list $\mathbf{y}_i$, where a non-zero jth element y_{ij} indicates that agent i does task y_{ij} after task $y_{i(j-1)}$. For example, $y_i = [6\ \ 1\ \ 3]^{\mathrm{T}}$ indicates agent i will accomplish tasks 6, 1 and 3, in that order. The cost incurred for a UAV to perform an assignment is given by $c_i(\mathbf{y}_i)$. In this paper, c_i is a nonlinear cost indicating the distance traveled to perform all tasks in the assignment $\mathbf{y}_i$. We seek to minimize the total distance traveled by all UAVs, represented by Equation (1).

There exist precedence constraints between certain tasks, meaning that certain tasks must happen before other tasks. The function $t_j(\mathbf{y})$ indicates the time that task j will be completed given a set of feasible assignments for the team of agents, where $\mathbf{y} = [\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_{|\mathscr{U}|}^{\mathrm{T}}]^{\mathrm{T}}$. For any two tasks j and ℓ with a precedence constraint, the scalar $\omega_{j\ell}$ is 1, otherwise it is 0. These precedence constraints are expressed in Equation (3), which is dependent on every vehicle's assignment. In Equation (2), $d_{ij} = 1$ if agent i is assigned task j, otherwise $d_{ij} = 0$; this constraint indicates that each task is to be accomplished only once, and it also depends on every vehicle. These two constraints are coupling (or linking) constraints that consider the whole team of vehicles.

The rest of the constraints are local constraints (often called supply constraints) that depend on only one agent. Equation (4) is a (possibly) nonlinear resource constraint; r_j and R_i are used to mathematically describe local resource constraints on the vehicles, e.g. $\sum_{j \in \mathscr{T}} r_j(\mathbf{y}_i)$ is the fuel spent performing assignment $\mathbf{y}_i$ and R_i is the amount of fuel available. One of the interesting properties of DWD is that each agent needs only to know how to enumerate the constraints; it is not necessary to express the constraints as equalities or inequalities [10]. The final constraint (5) is the integrality constraint, and it is also a local constraint.

The OP is

$$\min \sum_{i \in \mathscr{U}} c_i(\mathbf{y}_i) \quad \text{subject to} \tag{1}$$

$$\sum_{i \in \mathscr{U}} d_{ij}(\mathbf{y}_i) = 1 \quad \forall j \in \mathscr{T} \tag{2}$$

$$\omega_{j\ell}(t_j(\mathbf{y}) - t_\ell(\mathbf{y})) \geqslant 0 \quad \forall j, \ell \in \mathscr{T} \tag{3}$$

$$\sum_{j \in \mathscr{T}} r_j(\mathbf{y}_i) \leqslant R_i \quad \forall i \in \mathscr{U} \tag{4}$$

$$y_{ij} \in \{0, 1, \ldots, N_c\} \quad \forall i \in \mathscr{U} \quad \forall j \in \mathscr{T} \tag{5}$$

Published 2011. This article is a US Government work and is in the public domain in the USA.

Int. J. Robust. Nonlinear Control 2011; **21**:1421–1433
DOI: 10.1002/rnc

3. COLUMN GENERATION

For the purpose of explaining column generation, we shall consider its use in the solution of a constrained shortest path problem. The example presented illustrates the formulation of column generation. We will discuss how column generation can be implemented in a distributed manner later in Section 4. The constrained shortest path problem very closely follows a similar problem found in the book *Column Generation* [1]. The following sections step through the process which is summarized by the flowchart shown in Figure 1.

3.1. Constrained shortest path problem

Let $\mathcal{N}$ be a network with n vertices. The goal of this optimization problem is to find the shortest path from vertex 1 to vertex n, while observing a resource constraint along the way. For any two vertices in this network, we define $E_{ij}=1$ if and only if travel from vertex i to vertex j is permitted. We then define the set of edges of this network to be $\mathscr{E}=\{(i,j)|E_{ij}=1; i=1,\ldots,n;\ j=1,\ldots,n\}$. Associated with each edge $(i,j)\in\mathscr{E}$ are the following two quantities: c_{ij} is the cost to traverse edge (i,j), and r_{ij} is a resource consumption amount to traverse edge (i,j). The total amount of resources available is limited, as shown by Equation (10). Our goal is to compute

$$Y^* = \min \sum_{(i,j)\in\mathscr{E}} c_{ij}y_{ij} \quad \text{subject to} \tag{6}$$

$$\sum_{(1,j)\in\mathscr{E}} y_{1j} = 1 \tag{7}$$

$$\sum_{(i,j)\in\mathscr{E}} y_{ij} - \sum_{(j,i)\in\mathscr{E}} y_{ji} = 0 \quad \forall i=2,\ldots,n-1 \tag{8}$$

$$\sum_{(i,n)\in\mathscr{E}} y_{in} = 1 \tag{9}$$

$$\sum_{(i,j)\in\mathscr{E}} r_{ij}y_{ij} \leqslant R \tag{10}$$

$$y_{ij} = 0 \quad \text{or} \quad 1 \quad \forall (i,j)\in\mathscr{E} \tag{11}$$

where a possible solution of (6)–(11) is a path denoted by an $n\times n$ matrix $Y=[y_{ij}]$. A value of $y_{ij}=1$ indicates that (i,j) is an edge in the prospective path, otherwise $y_{ij}=0$. It is required that if $E_{ij}=0$, then $y_{ij}=0$. Note that this is merely a matter of notation and not one that affects the

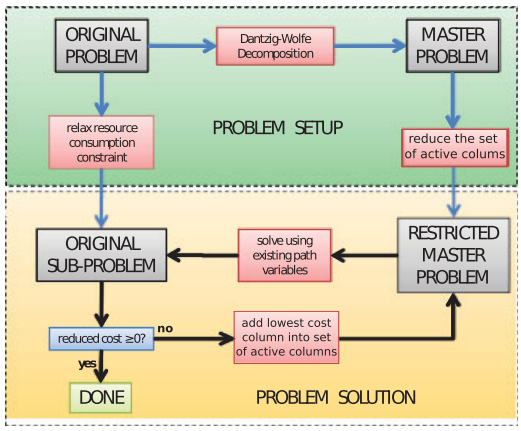

Figure 1. A high-level representation of column generation. The top problem setup block represents the steps necessary to convert the original problem into the original sub-problem and RMP. The bottom, problem solution block denotes the iterative process carried out to solve the RMP.

Published 2011. This article is a US Government work and is in the public domain in the USA.

Int. J. Robust. Nonlinear Control 2011; **21**:1421–1433
DOI: 10.1002/rnc

optimization of the problem. Hence, the problem at hand is to determine the optimal values of the y_{ij} for which $E_{ij}=1$. Here in the constrained shortest path problem, the variable y_{ij} has a different interpretation than it did in the VRPPC.

Equations (6)–(11) define the OP for the Constrained Shortest Path Problem ($\mathrm{OP_{CSPP}}$). Equation (7) ensures that one unit of flow leaves the source vertex. Equation (9) ensures that one unit of flow enters the sink vertex. The $n-2$ constraints given by Equation (8) ensures that ingoing flow equals the outgoing flow at the vertices $2,\ldots,n-1$. Equation (10) constrains the resource that is consumed by traveling between vertices. Equation (11) ensures that the state variables y_{ij} are always one of the two integers zero or one for every $i, j=1,\ldots,n$.

This OP is an integer program, which is NP-hard. It can be solved a number of ways, including an exhaustive search of all possible edge combinations. Column generation first reformulates the $\mathrm{OP_{CSPP}}$ into the master problem. For the constrained shortest path problem, the master problem searches for the optimal path from a collection of paths instead of the optimal set of edges from a collection of edges.

3.2. Master problem

In order to build the *Master Problem* for the Constrained Shortest Path Problem ($\mathrm{MP_{CSPP}}$), we first express y_{ij} as a convex combination of paths that satisfy constraints (7)–(9) and (11); let Q be the set of all paths satisfying the constraints and a path in Q is denoted by $\mathbf{q}=[q_{ij}]$, where $q_{ij}\in\{0,1\}$ is the flow on edge E_{ij} in path $\mathbf{q}$. A path is loosely defined as an ordered collection of edges in $\mathscr{E}$ that originate at vertex 1 and end at vertex n. The $\mathrm{MP_{CSPP}}$ performs the optimization over the variable $x_{\mathbf{q}}$, which is a weighting on the combination of paths $\mathbf{q}\in Q$. With these definitions, the following change of variables is used to convert the $\mathrm{OP_{CSPP}}$ into the $\mathrm{MP_{CSPP}}$:

$$y_{ij}=\sum_{\mathbf{q}\in Q} q_{ij}x_{\mathbf{q}} \quad \forall (i,j)\in\mathscr{E} \tag{12}$$

$$1=\sum_{\mathbf{q}\in Q} x_{\mathbf{q}} \tag{13}$$

$$x_{\mathbf{q}}\geqslant 0 \quad \forall \mathbf{q}\in Q \tag{14}$$

Now substitute Equation (12) into Equations (6) and (10) to get the following:

$$Y^*=\min \sum_{\mathbf{q}\in Q}\left(\sum_{(i,j)\in\mathscr{E}} c_{ij}q_{ij}\right)x_{\mathbf{q}} \quad \text{subject to} \tag{15}$$

$$\sum_{\mathbf{q}\in Q}\left(\sum_{(i,j)\in,\mathscr{E}} r_{ij}q_{ij}\right)x_{\mathbf{q}}\leqslant R \tag{16}$$

$$y_{ij}=0 \quad \text{or} \quad 1 \quad \forall (i,j)\in\mathscr{E} \tag{17}$$

$$y_{ij}=\sum_{\mathbf{q}\in Q} q_{ij}x_{\mathbf{q}} \quad \forall (i,j)\in\mathscr{E} \tag{18}$$

$$\sum_{\mathbf{q}\in Q} x_{\mathbf{q}}=1 \tag{19}$$

$$x_{\mathbf{q}}\geqslant 0 \quad \forall \mathbf{q}\in Q \tag{20}$$

Equations (15)–(20) are the $\mathrm{MP_{CSPP}}$, the solution of which can be used to determine the solution of the $\mathrm{OP_{CSPP}}$. In the following we relax Equation (17), which causes the link between y_{ij} and $\mathbf{q}$ to become unnecessary, thus allowing both Equations (17) and (18) to be removed. The $\mathrm{MP_{CSPP}}$ formulation described above is the DWD mentioned previously. A more thorough and general description of this process is described by Desaulniers *et al*. [1].

DOI: 10.1002/rnc

3.3. Restricted master problem

Column generation does not seek to solve the $\mathrm{OP}_{\mathrm{CSPP}}$ or the $\mathrm{MP}_{\mathrm{CSPP}}$, because of the large number of variables (or columns) associated with the $\mathrm{MP}_{\mathrm{CSPP}}$. Instead, column generation solves the *Restricted Master Problem* for the Constrained Shortest Path Problem ($\mathrm{RMP}_{\mathrm{CSPP}}$), which works with a relevant subset of the 'columns' from the $\mathrm{MP}_{\mathrm{CSPP}}$. Here, the columns (or variables) refer to the paths $\mathbf{q}$ and their associated variables $\sum c_{ij}q_{ij}$ and $\sum r_{ij}q_{ij}$, which constitute the columns of the cost and constraint matrices in the $\mathrm{RMP}_{\mathrm{CSPP}}$. This relevant subset of columns is denoted by $\check{Q} \subset Q$. Using $\check{Q}$, along with the above-mentioned relaxation, we express the $\mathrm{RMP}_{\mathrm{CSPP}}$ as[§]:

$$Y^* = \min \sum_{\mathbf{q}\in\check{Q}} \left(\sum_{(i,j)\in\mathscr{E}} c_{ij}q_{ij} \right) x_{\mathbf{q}} \quad \text{subject to} \tag{21}$$

$$\sum_{\mathbf{q}\in\check{Q}} \left(\sum_{(i,j)\in\mathscr{E}} r_{ij}q_{ij} \right) x_{\mathbf{q}} \leqslant R \tag{22}$$

$$\sum_{\mathbf{q}\in\check{Q}} x_{\mathbf{q}} = 1 \tag{23}$$

$$x_{\mathbf{q}} \geqslant 0 \quad \forall \mathbf{q} \in \check{Q} \tag{24}$$

The solution of the $\mathrm{RMP}_{\mathrm{CSPP}}$ will require us to associate dual variables with the constraints given by Equations (22) and (23). Denote these dual variables as γ_1 and γ_2, respectively. The dual variables are used to search for a reduced cost variable (path). This is accomplished by solving a sub-problem to the $\mathrm{OP}_{\mathrm{CSPP}}$. Consider the following problem:

$$C^* = \min \sum_{(i,j)\in\mathscr{E}} (c_{ij} - \gamma_1 r_{ij}) y_{ij} - \gamma_2 \quad \text{subject to} \tag{25}$$

$$\sum_{(1,j)\in\mathscr{E}} y_{1j} = 1 \tag{26}$$

$$\sum_{(i,j)\in\mathscr{E}} y_{ij} - \sum_{(j,i)\in\mathscr{E}} y_{ji} = 0 \quad \forall i = 2, \ldots, n-1 \tag{27}$$

$$\sum_{(i,n)\in\mathscr{E}} y_{in} = 1 \tag{28}$$

$$y_{ij} = 0 \quad \text{or} \quad 1 \quad \forall (i,j) \in \mathscr{E} \tag{29}$$

Equations (25)–(29) are commonly referred to as the *Orig inal Sub-Problem* for the Constrained Shortest Path Problem ($\mathrm{OSP}_{\mathrm{CSPP}}$) or the pricing problem. Column generation iterates between the $\mathrm{RMP}_{\mathrm{CSPP}}$ and $\mathrm{OSP}_{\mathrm{CSPP}}$. The $\mathrm{RMP}_{\mathrm{CSPP}}$ finds the dual variables associated with the current iteration's optimal solution. Using these dual variables, the $\mathrm{OSP}_{\mathrm{CSPP}}$ determines the path $\mathbf{q}^*_{\mathrm{OSP}} \in Q$ that yields the minimum reduced cost. If the minimum reduced cost C^* is nonnegative, then there is no variable that can be added to the $\mathrm{RMP}_{\mathrm{CSPP}}$ that will improve the solution and the current solution to the relaxed $\mathrm{MP}_{\mathrm{CSPP}}$ is optimal, which when rounded generally yields the optimal solution to the $\mathrm{OP}_{\mathrm{CSPP}}$ [3, 11]. However, if $C^* < 0$ then $\mathbf{q}^*_{\mathrm{OSP}}$ is appended to the set $\check{Q}$ for use in the next iteration of the $\mathrm{RMP}_{\mathrm{CSPP}}$. The process is repeated until $C^* \geqslant 0$ and is illustrated by the flowchart shown in Figure 1. If the solution to the RMP is integer, then the optimal solution to the $\mathrm{OP}_{\mathrm{CSPP}}$ is guaranteed; otherwise, a branch-and-price scheme will need to be used to determine the $\mathrm{OP}_{\mathrm{CSPP}}$'s optimal solution [10, 12, 13].

[§] In the following, we will not explicitly mention this relaxation step.

DOI: 10.1002/rnc

4. COLUMN GENERATION APPLIED TO THE OP

In the previous section, column generation was thoroughly explained for the simpler constrained shortest path problem. We now develop the column generation formulation of the more challenging VRPPC. The process is essentially the same as before, as long as care is taken to account for a subtle, yet significant difference in the two problems. This difference is captured by the inequalities described by (4) and (10). In the constrained shortest path problem, the resource constraints are described by the linear inequalities (10); however, in the VRPPC the resource constraint inequalities (4) are nonlinear. When the constraint sets described by (10) are considered as Euclidean sets, they will necessarily be convex polytopes. The theory of extreme points of convex polytopes is well-studied, and the change of variables described by (12)–(14) is a trivial substitution. However, when the constraint sets described by (4) are considered as Euclidean sets, they need not be convex sets. In fact, the resulting sets need not even be polytopes. Very little analysis exists in the literature on extreme points of general Euclidean sets. In order to proceed with the development of the column generation formulation, it will be necessary to make a substitution that is analogous to the one described by (12)–(14). We prove the following theorem to show that such a substitution is possible for general Euclidean sets. Theorem 1 is the central analytical result of this paper and permits the conversion of the OP into the Master Problem.

4.1. Extreme points of convex and non-convex sets

Let $\mathscr{S}$ be a subset of the metric space $\mathscr{X}$. The relative interior, closure and boundary of $\mathscr{S}$ are denoted by $\mathscr{S}^o, \overline{\mathscr{S}}$ and $\partial\mathscr{S}$, respectively; if $\overline{\mathscr{S}}=\mathscr{X}$, then $\mathscr{S}$ is dense in $\mathscr{X}$.

Definition 1
Let $\mathscr{C}$ be a subset of $\mathbb{R}^n$. A point $c\in\mathscr{C}$ is an *extreme point* of $\mathscr{C}$ if and only if there do not exist two distinct points c_1 and c_2 in $\mathscr{C}$ such that $c=\alpha c_1+(1-\alpha)c_2$ for some $\alpha, 0<\alpha<1$. In other words, if c does not lie in any open line segment joining two distinct points of $\mathscr{C}$, then c is an extreme point of $\mathscr{C}$. Let ext($\mathscr{C}$) denote the set of all extreme points of $\mathscr{C}$.

Definition 2
A set $\mathscr{K}\subset\mathbb{R}^n$ is a *convex polytope* if and only if $\mathscr{K}$ is compact, convex and the cardinality of ext($\mathscr{K}$) is finite. A subset of $\mathbb{R}^n$ is a *non-convex polytope* if and only if it is compact, connected, not convex and a finite union of convex polytopes. A subset of $\mathbb{R}^n$ that is not a convex polytope nor a non-convex polytope is a *non-polytope*.

Definition 3
Let $a_1, a_2, \ldots, a_n$ be real numbers not all equal to zero. The set of all vectors $\mathbf{x}=[x_1\ \ x_2\ \ \cdots\ \ x_n]^{\mathrm{T}}$ in $\mathbb{R}^n$ such that $a_1x_1+a_2x_2+\cdots+a_nx_n=b$ for some constant $b\in\mathbb{R}$ is a *hyperplane* in $\mathbb{R}^n$.

Definition 4
Let $\mathscr{D}$ be a compact subset of $\mathbb{R}^n$, and let $\mathfrak{L}_n(\mathscr{D})$ denote the Lebesgue measure of $\mathscr{D}$ in $\mathbb{R}^n$. A set $\mathscr{M}\subset\partial\mathscr{D}$ is an *affine boundary section* of $\mathscr{D}$ if and only if all of the following conditions hold: (a) $\mathscr{M}$ is nonempty, connected and closed; (b) there exists exactly one hyperplane $\mathscr{H}$ in $\mathbb{R}^n$ such that $\mathfrak{L}_{n-1}(\mathscr{M}\cap\mathscr{H})>0$; (c) $\mathscr{M}$ is maximal in the sense that there does not exist a connected set $\mathscr{N}\subset\partial\mathscr{D}$ such that $\mathscr{M}\subsetneq\mathscr{N}$ and $\mathfrak{L}_{n-1}(\mathscr{N}\cap\mathscr{H})>\mathfrak{L}_{n-1}(\mathscr{M}\cap\mathscr{H})$; (d) $\mathscr{M}\cap\mathscr{H}^c=\emptyset$. A set $\mathscr{E}\subset\partial\mathscr{D}$ is a *non-affine boundary section* of $\mathscr{D}$ if and only if all of the following conditions hold: (a) $\mathscr{E}$ is nonempty, connected and closed; (b) $\mathscr{E}^o\cap\mathscr{M}=\emptyset$ for every set $\mathscr{M}$ that is an affine boundary section of $\mathscr{D}$; (c) $\mathscr{E}$ is maximal in the sense that there does not exist a connected set $\mathscr{F}\subset\partial\mathscr{D}$ such that $\mathscr{E}\subsetneq\mathscr{F}$ and $\mathscr{F}^o\cap\mathscr{M}=\emptyset$ for every set $\mathscr{M}$ that is an affine boundary section of $\mathscr{D}$.

Definition 5
Let $\mathscr{R}$ be a subset of $\mathbb{R}^n$. $\mathscr{R}$ is *simply composed* if and only if each of the following conditions hold: (a) $\mathscr{R}$ is the disjoint union of finitely many nonempty, compact connected sets, denoted $\mathscr{R}_i$

DOI: 10.1002/rnc

for each $i \in \{1, 2, \ldots, K\}$ for some integer $K \geqslant 1$; (b) for each $i \in \{1, 2, \ldots, K\}$, $\partial \mathscr{R}_i$ is the union of finitely many affine boundary sections and non-affine boundary sections.

Definition 6
A metric space is *separable* if and only if there exists a sequence of elements of the space such that every nonempty open subset of the space contains at least one element of the sequence, i.e. if the space contains a countable dense subset.

Theorem 1
Let $\mathscr{P}$ be any simply composed subset of $\mathbb{R}^n$, and let $\mathscr{Q} = \text{ext}(\mathscr{P})$. There exists a countable subset of $\mathscr{Q}$, denoted Q, such that for any point $p \in \mathscr{P} \setminus \mathscr{Q}$, there exists a real, nonnegative coefficient c_q^p for each $q \in Q$ satisfying the following conditions:

$$p = \sum_{q \in Q} c_q^p q \tag{30}$$

$$1 = \sum_{q \in Q} c_q^p \tag{31}$$

Proof
This argument is broken into four cases, two cases for $\mathscr{P}$ being convex and two cases for $\mathscr{P}$ being non-convex. In each of the cases, $\mathscr{P}$ can be either a polytope or a non-polytope.

Case 1: $\mathscr{P}$ is a convex polytope.

This is exactly the well-studied case of polytopic convex sets. The set of extreme points Q and coefficients c_q^p sought can be shown to exist in numerous texts, e.g., Luenberger [14] or Rockafellar [15]. For this case Q is the set of vertices of $\mathscr{P}$. The limitations of Case 1 motivate this more general theorem.

Case 2: $\mathscr{P}$ is a non-convex polytope.

Let $\text{conv}(\mathscr{P})$ denote the convex hull of $\mathscr{P}$, and let $Q = \text{ext}(\text{conv}(\mathscr{P}))$, i.e. the set of vertices of $\text{conv}(\mathscr{P})$. Since $\text{conv}(\mathscr{P})$ is a convex polytope by construction, it must be that Q has finite cardinality; i.e. Q is countable. Apply Case 1 to $\text{conv}(\mathscr{P})$, and note that $\mathscr{P} \subset \text{conv}(\mathscr{P})$ and $Q \subset \mathscr{Q}$ by construction. Then, Case 1 guarantees the existence of the coefficients c_q^p that satisfy Equations (30) and (31).

Case 3: $\mathscr{P}$ is a convex non-polytope.

Since $\mathscr{P}$ is a simply composed non-polytope, there exist finitely many non-affine boundary sections of $\mathscr{P}$, denoted $\mathscr{B}_i$ where $i \in \{1, 2, \ldots, m\}$ and $m \geqslant 1$. $\mathscr{P}$ is also convex, so there must exist uncountably many points in each boundary section $\mathscr{B}_i$ that are extreme points of $\mathscr{P}$. Now define

$$\mathscr{B} = \bigcup_{i=1}^{m} \mathscr{B}_i \tag{32}$$

Let $Q^b = \mathscr{Q} \cap \mathscr{B}$, and note that Q^b is uncountable by construction. Now let $\mathscr{A} = \overline{\partial \mathscr{P} \setminus \mathscr{B}}$ and $Q^a = \mathscr{Q} \cap \mathscr{A}$; i.e. $\mathscr{A}$ is the union of all the affine boundary sections of $\mathscr{P}$. Since $\mathscr{P}$ is simply composed, Q^a is a finite subset of $\mathscr{Q}$. Note that $Q^a \cup Q^b = \mathscr{Q}$ since all of the extreme points of $\mathscr{P}$ will necessarily be on the boundary of $\mathscr{P}$. The goal is to construct a countable, dense subset of Q^b, denoted Q^b_{cd}. Clearly, Q^b is a closed and bounded subset of $\mathbb{R}^n$, and hence compact. But every compact metric space is separable; see Kelley [16]. Thus, Q^b is separable as a metric space with the topology induced by $\mathbb{R}^n$. This implies the existence of the countable, dense set $Q^b_{cd} \subset Q^b$. Now define

$$Q = Q^a \cup Q^b_{cd} \tag{33}$$

By construction Q is countable. Furthermore and most importantly, $\overline{Q} = \mathscr{Q}$. Now choose any point $p \in \mathscr{P} \setminus \mathscr{Q}$. Either $p \in \partial \mathscr{P}$ or $p \in \mathscr{P}^o$. If $p \in \partial \mathscr{P}$, then it must be that $p \in \mathscr{A}$, because if p were in $\mathscr{B}$ then p would be an extreme point of $\mathscr{P}$, which is a contradiction to p being in $\mathscr{P} \setminus \mathscr{Q}$. Since $p \in \mathscr{A}$ and p is not an extreme point of $\mathscr{P}$, then it must be the case that p lies in the relative

DOI: 10.1002/rnc

interior of an affine boundary section of $\mathscr{P}$, which is itself a convex polytope in $\mathbb{R}^{n-1}$. Then, Case 1 implies the existence of the coefficients satisfying Equations (30) and (31).

Hence, let us assume that $p \in \mathscr{P}^o$. Since Q is dense in $\mathscr{Q}$ it follows that there exists a finite subset of Q, denoted $\widetilde{Q}^p$, such that p is contained in the relative interior of the convex polytope formed using the elements of $\widetilde{Q}^p$ as vertices. We will denote this convex polytope as $\widetilde{\mathscr{P}}$ and denote a vertex in $\widetilde{Q}^p$ by $\tilde{q}^p$. Now let

$$\widehat{Q}^p = Q \setminus \widetilde{Q}^p \tag{34}$$

By construction $\widetilde{\mathscr{P}}$ is a convex polytope, the $\tilde{q}^p$ are extreme points of $\widetilde{\mathscr{P}}$, and $p \in \widetilde{\mathscr{P}}$. Thus, we can apply Case 1 to $\widetilde{\mathscr{P}}$, which implies that there exist real, nonnegative coefficients $c^p_{\tilde{q}^p}$ satisfying the following equations:

$$p = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} \tilde{q}^p \tag{35}$$

$$1 = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} \tag{36}$$

The sets $\widetilde{Q}^p$ and $\widehat{Q}^p$ depend on the point p that was chosen. In other words, no matter which finite collection of extreme points are used to construct $\widetilde{\mathscr{P}}$, there always exists a point $p^* \in \mathscr{P}^o$ such that p^* is not in $\widetilde{\mathscr{P}}$. This is not what the statement of the theorem requires. A single countable set of extreme points of $\mathscr{P}$ must be found such that for any point $p \in \mathscr{P}^o$, there exist real, nonnegative coefficients satisfying Equations (30) and (31). The coefficients will necessarily depend on p, but the collection of extreme points must not. The countable set of extreme points required has already been constructed; it is Q. Q is fixed once the set $\mathscr{P}$ is chosen. For every $p \in \mathscr{P}^o$ and for every $\hat{q}^p \in \widehat{Q}^p$, let $c^p_{\hat{q}^p} = 0$. Then one can easily verify that

$$\sum_{\hat{q}^p \in \widehat{Q}^p} c^p_{\hat{q}^p} = \sum_{\hat{q}^p \in \widehat{Q}^p} c^p_{\hat{q}^p} \hat{q}^p = 0 \tag{37}$$

Now using Equations (34), (35) and (37),

$$p = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} \tilde{q}^p = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} \tilde{q}^p + 0 = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} \tilde{q}^p + \sum_{\hat{q}^p \in \widehat{Q}^p} c^p_{\hat{q}^p} \hat{q}^p = \sum_{q \in Q} c^p_q q \tag{38}$$

Similarly, using Equations (34), (36) and (37),

$$1 = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} + 0 = \sum_{\tilde{q}^p \in \widetilde{Q}^p} c^p_{\tilde{q}^p} + \sum_{\hat{q}^p \in \widehat{Q}^p} c^p_{\hat{q}^p} = \sum_{q \in Q} c^p_q \tag{39}$$

Since all the coefficients are nonnegative, Equations (38) and (39) are exactly what needed to be shown.

Case 4: $\mathscr{P}$ is a non-convex non-polytope.

This case is merely an application of Cases 1–3. First form conv($\mathscr{P}$) and note that conv($\mathscr{P}$) will be either a convex polytope or a convex non-polytope. Hence, we may apply either Case 1 or Case 3 to conv($\mathscr{P}$). This process will yield a set Q and real, nonnegative coefficients satisfying Equations (30) and (31) for every point $p \in \text{conv}(\mathscr{P}) \setminus \mathscr{Q}$. But now one can appeal to the argument used in Case 2, where we note that any point $p \in \mathscr{P}$ is also in conv($\mathscr{P}$), which implies that any point $p \in \mathscr{P} \setminus \mathscr{Q}$ is also in conv($\mathscr{P}$)$\setminus \mathscr{Q}$. These three cases guarantee the existence of the set Q and real, nonnegative coefficients satisfying Equations (30) and (31). □

4.2. Column generation formulation of the VRPPC

As previously mentioned, each agent has its own collection of local constraints. Let P_i denote the subset of $\mathbb{R}^n$ that is defined by agent i's local constraints; P_i is defined as follows:

$$P_i = \left\{ \mathbf{y}_i : \sum_{j \in \mathscr{T}} r_j(\mathbf{y}_i) \leqslant R_i, 0 \leqslant y_{ij} \leqslant N_c \right\} \tag{40}$$

DOI: 10.1002/rnc

Now for any point $\mathbf{y}_i \in P_i$, either $\mathbf{y}_i$ is an extreme point of P_i or it is not. If it is the case that $\mathbf{y}_i$ is an extreme point of P_i, then there exists a trivial convex combination of extreme points of P_i that equals $\mathbf{y}_i$, namely

$$\mathbf{y}_i = 1 \cdot \mathbf{y}_i + \sum_{\substack{\mathbf{q}_i \in Q_i \\ \mathbf{q}_i \neq \mathbf{y}_i}} 0 \cdot \mathbf{q}_i \tag{41}$$

where Q_i is any countable collection of extreme points of P_i. For the purpose of this paper, the concern is actually for the case in which $\mathbf{y}_i$ is not an extreme point of P_i. For this case, first note that P_i is a compact subset of $\mathbb{R}^n$. Furthermore, assume that P_i is simply composed. This is not a significantly intrusive assumption. Essentially, this assumption prevents the consideration of 'wild' sets, e.g. the Cantor set or any set exhibiting fractal behavior. Then Theorem 1 may be directly applied, which guarantees that there exists a countable collection of extreme points of P_i, denoted Q_i, and real, nonnegative coefficients, denoted $x_{\mathbf{q}_i}^{\mathbf{y}_i}$, that satisfy the following equations:

$$\mathbf{y}_i = \sum_{\mathbf{q}_i \in Q_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} \mathbf{q}_i \tag{42}$$

$$1 = \sum_{\mathbf{q}_i \in Q_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} \tag{43}$$

Applying Equations (42) and (43) to the original problem in Equations (1)–(5) yields the MP for the VRPPC:

$$\min \sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in Q_i} c_{\mathbf{q}_i}^{\mathbf{y}_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} \quad \text{subject to} \tag{44}$$

$$\sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in Q_i} [\mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j x_{\mathbf{q}_i}^{\mathbf{y}_i} = 1 \quad \forall j \in \mathscr{T} \tag{45}$$

$$\omega_{j\ell} \sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in \mathscr{Q}_i} ([\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j - [\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_\ell) x_{\mathbf{q}_i}^{\mathbf{y}_i} \geqslant 0 \quad \forall j, \ell \in \mathscr{T} \tag{46}$$

$$\sum_{\mathbf{q}_i \in Q_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} = 1, \quad x_{\mathbf{q}_i}^{\mathbf{y}_i} \geqslant 0 \quad \forall i \in \mathscr{U} \tag{47}$$

where $c_{\mathbf{y}_i}^{\mathbf{q}_i} = c_i(\mathbf{q}_i)$, $[\mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j = d_{ij}(\mathbf{q}_i)$, $[\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j = t_j(\mathbf{q}_i)$, and each extreme point, $\mathbf{q}_i$, denotes a locally feasible assignment for agent i. Assuming Equation (2) holds, then any task j can only be permitted once during an assignment; $[\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j$ is the time of this assignment,

$$t_j(\mathbf{y}) = t_j^i(\mathbf{y}_i) = t_j\left(\sum_{\mathbf{q}_i \in Q} x_{\mathbf{q}_i}^{\mathbf{y}_i} \mathbf{q}_i\right) = \sum_{\mathbf{q}_i \in Q} x_{\mathbf{q}_i}^{\mathbf{y}_i} t_j(\mathbf{q}_i) = \sum_{\mathbf{q}_i \in Q} [\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j x_{\mathbf{q}_i}^{\mathbf{y}_i}$$

where $t_j^i(\mathbf{y}_i)$ is the time agent i performs task j and $[\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j = 0$ if $[\mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j = 0$. The coefficients $c_{\mathbf{q}_i}^{\mathbf{y}_i}$, $[\mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j$ are found in a similar manner. Practically, there is no quandary in defining $[\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j$ under the constraint of Equation (2); this is because agents can locally ignore any candidate assignments that violate Equation (2).

Because of the large number of variables, the RMP regards a subset of the columns, i.e. $\check{Q}_i \subset Q_i$. This allows the RMP to be written as follows:

$$\min \sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in \check{Q}_i} c_{\mathbf{q}_i}^{\mathbf{y}_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} \quad \text{subject to} \tag{48}$$

$$\sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in \check{Q}_i} [\mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j x_{\mathbf{q}_i}^{\mathbf{y}_i} = 1 \quad \forall j \in \mathscr{T} \tag{49}$$

$$\omega_{j\ell} \sum_{i \in \mathscr{U}} \sum_{\mathbf{q}_i \in \check{Q}_i} ([\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j - [\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_\ell) x_{\mathbf{q}_i}^{\mathbf{y}_i} \geqslant 0 \quad \forall j, \ell \in \mathscr{T} \tag{50}$$

$$\sum_{\mathbf{q}_i \in \check{Q}_i} x_{\mathbf{q}_i}^{\mathbf{y}_i} = 1, \quad x_{\mathbf{q}_i}^{\mathbf{y}_i} \geqslant 0 \quad \forall i \in \mathscr{U} \tag{51}$$

DOI: 10.1002/rnc

Notice, the optimal solution to the RMP is not necessarily the same as that of the MP because the optimal assignments for the MP might not be contained in the sets $\check{Q}_i$ for each $i \in \mathscr{U}$. The latter is overcome through successive iterations between the RMP and the subproblem, which for the ith agent is given by

$$\max_{\mathbf{q}_i \in Q_i} c_{\mathbf{q}_i}^{\mathbf{y}_i} - \pi \mathbf{d}_{\mathbf{q}_i}^{\mathbf{y}_i} - \lambda \mathbf{a}_{\mathbf{q}_i}^{\mathbf{y}_i} - \mu_i$$

where π, λ and μ are the Lagrangian dual variables of appropriate dimension and $\mathbf{a}_i^q$ is a vector with elements $\mathbf{a}_{\mathbf{q}_i}^{\mathbf{y}_i} = [[\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_j - [\mathbf{t}_{\mathbf{q}_i}^{\mathbf{y}_i}]_\ell]$ for $\{j, \ell | \omega_{j\ell} = 1\}$.

The assignment, $\mathbf{q}_i^*$, is the solution to the ith subproblem and has an associated column that improves the RMP. The assignment is introduced into the RMP as $\check{Q}_i \leftarrow \check{Q}_i \cup \{\mathbf{q}_i^*\}$. The column generated in the subproblem refers to the variables $c_{\mathbf{q}_i^*}^{\mathbf{y}_i}$, $\mathbf{d}_{\mathbf{q}_i^*}^{\mathbf{y}_i}$, and $\mathbf{a}_{\mathbf{q}_i^*}^{\mathbf{y}_i}$, which is a column in the cost and constraint matrix that formulates the RMP. Column generation operates by iterating between the RMP and the subproblem until every subproblem solution is greater than 0 for all $i \in \mathscr{U}$, at which point the solution to the RMP (and the relaxed MP) is the current RMP solution. In a distributed setting, the subproblems are solved locally on each agent, and the columns are broadcast to all teammates. Consequently, agents redundantly solve the RMP with identical sets of active columns.

It has been frequently noted that the relaxed MP will often have a solution that when rounded gives the optimal integer solution [3, 11]. However, we found that with the precedence constraints the solution to the relaxed MP was not close enough for rounding to yield a feasible integer solution. Therefore, a subsequent branch-and-price technique was necessary to find the optimal solution [17].

5. SIMULATION

5.1. No precedence constraints

We now present a simulation solving the OP in Equations (1)–(5) where the precedence constraints in Equation (3) are irrelevant. This is equivalent to having $\omega_{j\ell} = 0$ for all j and ℓ. There are five UAVs and ten targets placed randomly in the area of responsibility. The locations of the UAVs and targets can be seen in Figure 2 where the trajectories, found by solving the OP, for each UAV are plotted. Every scenario we encountered with five agents and ten targets resulted in an exact integer solution; no rounding was necessary.

In Figure 3, we plot the values of the cost function as the distributed column generation algorithm iterates. The blue line is the sum of every UAV's cost. All other lines are the cost for an individual UAV. The algorithm iterates 27 times before finishing, i.e. the reduced-cost calculated by each UAV is positive. The plot begins at the fourth iteration because the column generation algorithm was initialized using the big-M method, and the first four steps contain large cost, irrelevant feasible plans [4]. After approximately five iterations, enough flyable paths have been introduced such that a true feasible plan can be found among the active columns. Once a true feasible plan is active, column generation guarantees feasible solutions that improve monotonically.

Notice after 20 iterations the costs do not change. During the subproblem phase of iterations 20 through 27, at least one agent is able to find a flyable path that reduces the global cost, where a 'flyable path' is a locally feasible path. Remember, each UAV's subproblem ignores coupling constraints. After the 20th iteration, all locally feasible paths, found during the subproblem, are in conflict with other agents in the team. Thus, when added to the RMP, these conflicting paths do not change the RMP's current solution. After iteration 27, there are no locally feasible paths with negative reduced cost, and the optimization terminates.

5.2. Precedence constraints

In this scenario, we have four heterogeneous UAVs that are tasked to search a road and destroy two targets. To destroy a target, the team of UAVs must first confirm the target, attack the target

DOI: 10.1002/rnc

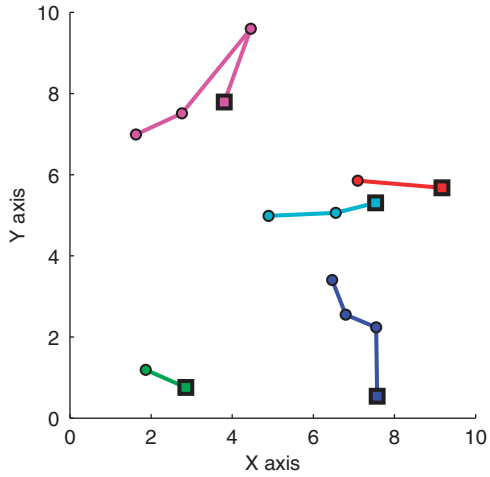

Figure 2. The trajectories, for each UAV, found using a distributed column generation algorithm.

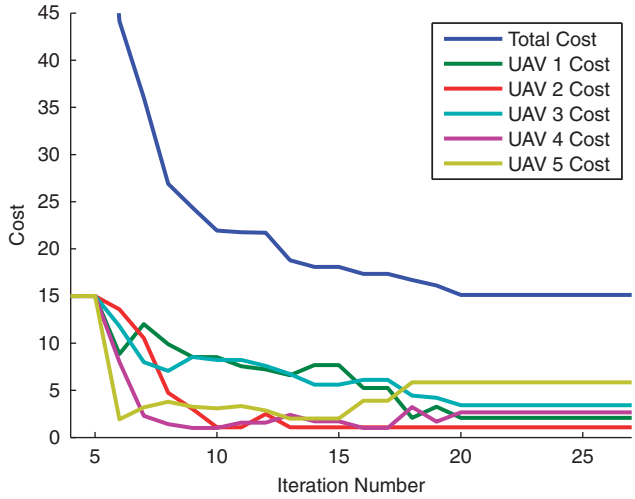

Figure 3. Convergence of the cost function. The top line is the sum of each UAV's cost. All other lines are the cost for a particular UAV's trajectory.

and verify its destruction. After destroying the two targets, the team of UAVs is free to search the road. The scenario is depicted in Figure 4. The circles indicate the two targets. The four squares are the UAVs. UAVs 1 and 4 can classify, verify, search the road and have one munition onboard able to destroy the target. UAV 2 is a flying bomb that can only destroy the target (classification and verification must be done by another agent). UAV 3 is a sensor craft that can classify, verify and search the road.

The cost for a given UAV assignment is calculated using the Dubins' path to the first objective and then the subsequent Dubins' path to the following objective and so forth until the end of the assignment. Each UAV has a minimum turn radius of 20 m. For a path search, the distance is given by the distance to whichever end of the path can be reached first plus the length of the path. A branch-and-price scheme was needed to find the optimal integer solution. The optimal solution assigned UAV 1 to perform classify, attack, verify on target 1, followed by a search on paths 1 and 2. UAV 4 completed the remaining tasks by classifying, attacking and verifying target 2 and searching path 2. UAVs 2 and 3 were not given an assignment.

DOI: 10.1002/rnc

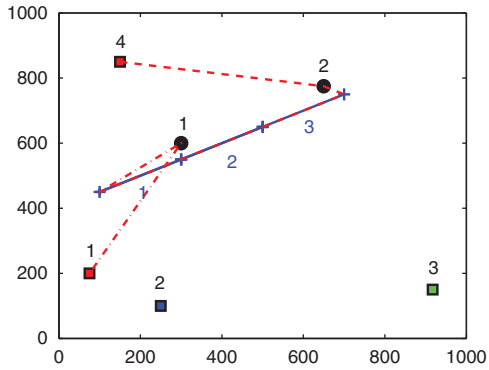

Figure 4. Heterogeneous agents (squares) tasked to destroy 2 targets (circles) and then search the path. Here the axis have units of meters. The optimal solution to the OP in Equations (1)–(5) is depicted by the dashed lines. UAV 1 is assigned to classify attack, and verify target 1, followed by a search on paths 1 and 2. UAV 4 classifies, attacks and verifies target 2 and then searches path 2. UAVs 2 and 3 were not assigned.

The solution took 114 iterations to complete over four branches. Different branching variables could yield less iterations by cutting away more solutions. Branch 1 consisted of having UAV 3 do nothing or something, and it was found that UAV 3 should do nothing. Branch 2 found that UAV 2 does not do the attack on Target 2. Branch 3 found that UAV 1 will do the classify on Target 1. Branch 4 found that UAV 4 will attack Target 2 and yielded the optimal integer solution.

6. CONCLUSION

We applied column generation for routing autonomous agents in a task assignment problem with precedence constraints. To achieve this, we proved a critical analytical result which facilitated the process of reformulating a nonlinear program as a linear program. The motivation for this work was to assess the possibility of column generation as a distributed means for creating vehicle assignments.

We found that due to the precedence constraints, rounding the final relaxed solution was not sufficient to yield the optimal solution, let alone a feasible solution. A branch-and-price scheme was needed to solve the problem. In a distributed setting, this would require the agents to agree on a feasible solution before commencing each branch. Deciding how to branch is less of an issue if the agents are in a fully connected network. However, realistically, the communication topology would not be fully connected and branching would need to be coordinated. In this situation, further work is needed to determine convergence properties of the optimization routine if each agent's local RMP is different from a neighbor's RMP.

Due to the complexity of the problem, it seems unlikely that column generation alone would be practical for distributed task assignment. Perhaps it could be used in conjunction with other methods or a hierarchical design to allow sub-groups to solve smaller-sized problems. Ofttimes approximation methods are used for practical situations due to the need for speed. One possible choice for a distributed approximation is [18], but further work would be needed to include precedence constraints (see [19]).

REFERENCES

1. Desaulniers G, Desrosiers J, Solomon M (eds). *Column Generation. GERAD 25th Anniversary Series*. Springer: New York, NY, 2005.
2. Dantzig G, Wolfe P. Decomposition principle for linear programs. *Operations Research* 1960; **8**(1):101–111.
3. Wolsey L. *Integer Programming*. Wiley: New York, 1998.
4. Luenberger DG, Ye Y. *Linear and Nonlinear Programming*. Springer: New York, 2008.

DOI: 10.1002/rnc

5. Lubbecke ME, Desrosiers J. Selected topics in column generation. *Operations Research* 2005; **53**(6):1007–1023. DOI: 10.1287/opre.1050.0234.
6. Vanderbeck F, Savelsbergh MW. A generic view of dantzig-wolfe decomposition in mixed integer programming. *Operations Research Letters* 2006; **34**(3):296–306. DOI: 10.1016/j.orl.2005.05.009.
7. Karaman S, Inalhan G. Large-scale task/target assignment for UAV fleets using a distributed branch and price optimization scheme. *Proceedings of the 17th IFAC World Congress*, Seoul, Republic of Korea, 2008.
8. Kallehauge B, Larsen J, Madsen OB, Solomon MM. Vehicle routing problem with time windows. In *Column Generation*, Desaulniers G, Desrosiers J, Solomon MM (eds). Springer Science—Business Media Inc.: New York, NY, 2005; 67–98.
9. Rios J, Ross K. Massively parallel Dantzig-Wolfe decomposition applied to traffic flow scheduling. *AIAA Guidance, Navigation and Control Conference*, Chicago, IL, 2009.
10. Barnhart C, Johnson EL, Nemhauser GL, Savelsbergh MWP, Vance PH. Branch-and-price: column generation for solving huge integer programs. *Operations Research* 1998; **46**(3):316–329.
11. Marcotte O. The cutting stock problem and integer rounding. *Mathematical Programming* 1985; **33**(1):82–92.
12. Desrosiers J, Soumis F, Desrochers M. Routing with time windows by column generation. *Networks* 1984; **14**(4):545–565.
13. Vanderbeck F, Savelsbergh MW. An exact algorithm for IP column generation. *Operations Research Letters* 1996; **19**(4):151–159.
14. Luenberger DG. *Linear and Nonlinear Programming* (2nd edn). Addison-Wesley Publishing Company: Reading, MA, 1984.
15. Rockafellar RT. *Convex Analysis. Princeton Mathematical Series*. Princeton University Press: Princeton, NJ, 1970.
16. Kelley JL. *General Topology. Graduate Texts in Mathematics*. Springer: New York, NY, 1975.
17. Desrosiers J, Lubbecke M. A primer in column generation. In *Column Generation*, Desaulniers G, Desrosiers J, Solomon MM (eds). Springer Science—Business Media Inc.: New York, NY, 2005; 1–32.
18. Choi HL, Brunet L, How J. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics* 2009; **25**(4):912–926. DOI: 10.1109/TRO.2009.2022423.
19. Mercker T, Casbeer DW, Millet PT, Akella MR. An extension of consensus-based auction algorithms for decentralized, time-constrained task assignment. *American Control Conference*, Baltimore, MD, 2010.