12-6-2020

Primal-dual 2-approximation algorithm for the monotonic multiple depot heterogeneous traveling salesman problem

S. Rathinam
Texas A&M University

R. Ravi
Carnegie Mellon University

J. Bae
Michigan Technological University, bae@mtu.edu

K. Sundar
Los Alamos National Laboratory

Recommended Citation

Rathinam, S., Ravi, R., Bae, J., & Sundar, K. (2020). Primal-dual 2-approximation algorithm for the monotonic multiple depot heterogeneous traveling salesman problem. *Leibniz International Proceedings in Informatics, LIPIcs, 162*. http://doi.org/10.4230/LIPIcs.SWAT.2020.33
Retrieved from: https://digitalcommons.mtu.edu/michigantech-p/14438

Follow this and additional works at: https://digitalcommons.mtu.edu/michigantech-p

 Part of the Mechanical Engineering Commons

Primal-Dual 2-Approximation Algorithm for the Monotonic Multiple Depot Heterogeneous Traveling Salesman Problem

S. Rathinam
Texas A & M University, College Station, TX 77843, USA
srathinam@tamu.edu

R. Ravi
Carnegie Mellon University, Pittsburgh, PA 15213, USA
ravi@cmu.edu

J. Bae
Michigan Technological University, Houghton, MI 49931, USA
bae@mtu.edu

K. Sundar
Los Alamos Laboratory, NM 87545, USA
kaarthik@lanl.gov

Abstract

We study a Multiple Depot Heterogeneous Traveling Salesman Problem (MDHTSP) where the cost of the traveling between any two targets depends on the type of the vehicle. The travel costs are assumed to be symmetric, satisfy the triangle inequality, and are monotonic, *i.e.*, the travel costs between any two targets monotonically increases with the index of the vehicles. Exploiting the monotonic structure of the travel costs, we present a 2-approximation algorithm based on the primal-dual method.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Approximation Algorithm, Heterogeneous Traveling Salesman Problem, Primal-dual Method

Digital Object Identifier 10.4230/LIPIcs.SWAT.2020.33

Funding *R. Ravi*: This material is based upon work supported in part by the U. S. Office of Naval Research under award number N00014-18-1-2099.

1 Introduction

We consider a generalization of a multiple Traveling Salesman Problem (TSP) involving heterogeneous vehicles, where the cost of traveling between any two locations depends on the type of the vehicle. Given a set of targets, the initial location (depot) and metric travel costs corresponding to each vehicle, the objective is to find a tour for each vehicle such that each target is visited exactly once by some vehicle and the sum of the travel costs of all the vehicles is minimum. This problem is referred to as the Multiple Depot Heterogeneous Traveling Salesman Problem (MDHTSP) and is widely studied in the unmanned vehicle community [4, 5, 10, 11, 13–16, 20, 22].

MDHTSP is a generalization of the classic TSP and is NP-Hard. Therefore, we are interested in developing approximation algorithms for the MDHTSP. Henceforth, we assume the travel costs for each vehicle are symmetric and satisfy the triangle inequality unless otherwise mentioned. For covering all targets with multiple TSPs when all the vehicles are identical, there are several constant-factor approximation algorithms in the literature [6, 12, 17, 21]. Generally, most of these algorithms follow a three-step procedure: In the first

step, the tour requirements are relaxed to find an optimal constrained forest. In the second step, the edges in the constrained forest are doubled, or in special cases, a matching problem is solved (à la Christofides) for a subset of vertices and the corresponding edges are added to the forest to obtain an Eulerian graph for each vehicle. In the final step, the Eulerian graph for each vehicle is used to obtain an Eulerian walk and the repeated visits are shortcut to find a tour. Similar procedures are also used to find approximation algorithms for multiple Hamiltonian Path Problems in [3, 18].

No constant factor approximation algorithms are known for the MDHTSP. In [22], a $\frac{3n}{2}$-approximation algorithm is presented for the MDHTSP where n denotes the number of vehicles (not the number of nodes in the metric). For a related variant of MDHTSP where all the vehicles start from the same depot and the objective is to minimize the makespan (maximum travel cost of any vehicle), a constant-factor approximation algorithm is presented in [8]. For the same makespan objective, when the vehicles are functionally heterogeneous[1], a $(2\lceil \ln(n) \rceil + 1)$-approximation algorithm has been presented recently in [23].

Monotonic MDHTSP. We are interested in the special case of the MDHTSP in which some restriction is placed on the travel costs of the vehicles. Intuitively, we assume the vehicles are ordered and the costs are monotonic. Formally, let $D = \{d_1, d_2, \cdots, d_n\}$ denote the n depots (initial locations) of the vehicles. Let T represent the set of targets. For each $i \in \{1, \cdots, n\}$, let $V_i := T \cup \{d_i\}$ denote the set of vertices corresponding to the i^{th} vehicle, and let E_i denote the set of all the edges that join any two distinct vertices in V_i. For $i \in \{1, \cdots, n\}$, let the cost of traversing an edge $e \in E_i$ for the i^{th} vehicle be denoted by $cost^i_e$. We assume the travel costs for each vehicle satisfy the triangle inequality and are monotonic *i.e.*, for any two vehicles $i < j$, $cost^i_e \leq cost^j_e$. However, we do not assume that they are proportional (*i.e.* $cost^i_e = \rho \cdot cost^j_e$). A tour for vehicle i is given by the sequence $(d_i, u_{i1}, \cdots, u_{il_i}, d_i)$ where $u_{ij} \in T$ for $j = 1, \cdots, l_i$ and l_i denotes the number of targets visited by vehicle i. The travel cost of vehicle i is equal to $cost^i_{(d_i, u_{i1})} + \sum_{j=1}^{l_i - 1} cost^i_{(u_{ij}, u_{i(j+1)})} + cost^i_{(u_{il_i}, d_i)}$ if $l_i \geq 1$, and is equal to 0 otherwise. The objective is to find a tour for each vehicle such that each target is visited exactly once by some vehicle and the sum of travel costs of all the vehicles is minimum.

Even though the travel costs are monotonic, the partitioning of the targets amongst the vehicles is still non-trivial because the vehicles start their tours from different initial locations or depots. This is an important special case that naturally arises in the following practical applications: (1) If the vehicles are modelled as ground robots [19] traveling with the same speed but with different turning radius constraints and r_i denotes the minimum turning radius of vehicle i, then the vehicles can be ordered such that $r_1 \leq r_2 \cdots \leq r_n$. In this case, the travel costs (the shortest distances required to travel between targets subject to the turning radius constraints) are monotonic. (2) If the vehicles travel at different speeds and the travel cost for any vehicle between two targets is defined as the ratio of the Euclidean distance between the targets and the speed of the vehicle, then the travel costs satisfy the case of proportional costs, which are also monotonic. (3) If each vehicle has a fuel capacity and the vehicles are allowed to refuel at gas stations, then the cost of traveling between any two targets subject to the refueling constraints also increases as the fuel capacity of a vehicle decreases [9]. Here, again, the travel costs are monotonic.

[1] The travel costs for the vehicles may be the same but there are compatibility constraints between vehicles and targets.

When $n = 2$ and the travel costs are monotonic, a 2-approximation algorithm was presented for MDHTSP in [2] extending the primal-dual algorithm for the prize collecting TSP [7] to the two vehicle case. Similar to the prize-collecting TSP in which any target not visited by a vehicle must pay a penalty, for the two vehicle case, any target not visited by the first vehicle is on the tour of the second vehicle that is modeled by some form of penalty.

For the multiple vehicle case we address, it is not difficult to design a primal-dual algorithm to find a feasible solution where each target is connected to some depot. However, the key to proving a good approximation ratio is in the pruning procedure of such an algorithm so that the edges retained after the pruning procedure can be paid for by an appropriate dual solution. A closely related problem where a similar pruning procedure arises is the Prize Collecting Steiner Tree problem (PCST)[2]. In the primal-dual algorithm for the PCST [7], the greedy growth of the duals must be frozen due to the constraints represented by the penalties on the nodes, particularly when the total sum of duals associated with any subset of vertices reaches the total sum of penalties for this set. How such frozen components are handled in terms of whether they are connected to the final solution tree or not is the crucial step of the pruning procedure in [7]. This is accomplished by a labeling procedure which was also used in [2] for the two vehicle case. However, its direct generalization to the many vehicles case we address is much more involved. In this paper, we provide an alternate simpler view of the pruning procedure for the PCST and re-purpose it for our multi-vehicle extension. This is the main technical novelty in our work.

Contributions. We present a primal-dual 2-approximation algorithm for the n vehicle case of monotonic MDHTSP. Like the prior work [2], we use a primal-dual approach, but avoid the use of the prize-collecting TSP as explained above. The heart of our result is a 2-approximation for the Heterogeneous Spanning Forest (HSF) problem of finding a minimum cost collection of n trees from the depots covering all the targets among the different graphs corresponding to the vehicles[3]. The following is a summary of our key technical steps.

1. While multiple LP relaxations are possible for the MDHTSP, we present an LP relaxation and a dual that allows the primal-dual method to construct a HSF with a special nesting structure among its components (Lemma 3).
2. This structure is then used to prove that pruning appropriate edges from the output of the main loop of the primal-dual method will result in a feasible HSF (Lemma 4 and Theorem 5).
3. Finally, we show that the value of the dual can be decomposed into the sum of the dual values corresponding to each of the vehicles (Lemma 8). This allows us to decompose the proof of the bound on the cost of the edges in each tree in terms of its corresponding dual value.

Putting together the above components, we show that the cost of the HSF constructed using the proposed algorithm is at most the optimal LP relaxation cost of the MDHTSP. Short-cutting the Euler tours on the trees in this HSF provides a 2-approximation algorithm for the MDHTSP for the n-vehicle case (Theorem 7).

[2] Given a graph, a depot node, a penalty for each node and a cost of each edge in the graph, the objective of the PCST is to find a tree containing the depot such that the sum of the cost of all the edges present in the tree and the penalty of all the nodes not present in the tree is minimum.

[3] This result does not require the different costs to be metric, but only that they are monotonic across the vehicles. The metric condition is only used in converting the forests to tours.

2 LP Relaxation for MDHTSP and its Dual

We use two sets of integer variables that will be later relaxed to formulate an LP relaxation for the MDHTSP. The first set of variables denoted by x_e^i determines whether edge $e \in E_i$ is present in the tour of vehicle $i \in \{1, \cdots, n\}$. The second set of variables z_U^i is defined for $i = 1, \cdots, n-1$ and any $U \subseteq T$. Specifically, z_U^i is equal to 1 if U is the subset of all the targets visited by vehicles $i+1, \cdots, n$; otherwise, z_U^i is equal to 0. Refer to Fig. 1 for an illustration of these variables.

Note that $\sum_{U \subseteq T} z_U^i = 1 \ \forall i \in \{1, \cdots, n-1\}$, *i.e.*, every vehicle of index i up to $n-1$ picks a single subset of targets that will be covered by vehicles $i+1$ or later.

The following proposition is a simple consequence of the fact that the z-variables can be used to identify subsets of targets that need to be covered by vehicle i. For any $S \subseteq T$ and $i = 1, \cdots, n$, let $\delta_i(S) := \{(u, v) : u \in S, v \in V_i \setminus S\}$.

▶ **Proposition 1.** *Any feasible solution to the MDHTSP satisfies the following constraints:*

$$\sum_{e \in \delta_i(S)} x_e^i \geq 2 \sum_{U : S \subseteq U \subseteq T} (z_U^{i-1} - z_U^i) \quad \forall\ S \subseteq T,\ |S| \geq 1,\ i \in \{1, \cdots, n\};$$

$$z_T^0 = 1; \quad z_U^0 = 0 \ \ \forall U \neq T; \quad z_U^n = 0 \ \ \forall U \subseteq T.$$

Proof. Both $\sum_{U : S \subseteq U \subseteq T} z_U^{i-1}$ and $\sum_{U : S \subseteq U \subseteq T} z_U^i$ can either be 0 or 1. The constraint is trivially satisfied except for the case when $\sum_{U : S \subseteq U \subseteq T} z_U^{i-1} = 1$ and $\sum_{U : S \subseteq U \subseteq T} z_U^i = 0$. But $\sum_{U : S \subseteq U \subseteq T} z_U^{i-1} = 1$ can occur only if all the targets in S are visited by vehicles in $\{i, \cdots, n\}$. Also, $\sum_{U : S \subseteq U \subseteq T} z_U^i = 0$ can occur only if there is at least one target in S visited by a vehicle in $\{1, \cdots, i\}$. Therefore, if $\sum_{U : S \subseteq U \subseteq T} z_U^{i-1} = 1$ and $\sum_{U : S \subseteq U \subseteq T} z_U^i = 0$, there is at least one target in S that must be visited by vehicle i. This is implied by the constraint as it reduces to $\sum_{e \in \delta_i(S)} x_e^i \geq 2$ which is true. ◀

The LP relaxation of the MDHTSP that we work with contains only the constraints from the above proposition.

$$Cost_{lp} = \min \sum_{i=1}^{n} \sum_{e \in E_i} cost_e^i \ x_e^i \tag{1}$$

$$\sum_{e \in \delta_i(S)} x_e^i \geq 2 \sum_{U : S \subseteq U \subseteq T} (z_U^{i-1} - z_U^i) \quad \forall\ S \subseteq T,\ i = 1, \cdots, n, \tag{2}$$

$$z_T^0 = 1; \quad z_U^0 = 0 \quad \forall U \neq T; \quad z_U^n = 0 \quad \forall U \subseteq T \tag{3}$$

$$x_e^i \geq 0 \quad \forall\ e \in E_i, i = 1, \cdots, n, \tag{4}$$

$$z_U^i \geq 0 \quad \forall\ U \subseteq T, i = 1, \cdots, n-1. \tag{5}$$

The dual of the above linear program is as follows:

$$\max\ 2 \sum_{S \subseteq T} Y_1(S) \tag{6}$$

$$\sum_{S : e \in \delta_i(S)} Y_i(S) \leq cost_e^i \quad \forall\ e \in E_i,\ \forall i = 1, \cdots, n, \tag{7}$$

$$\sum_{S : S \subseteq U} Y_i(S) \leq \sum_{S : S \subseteq U} Y_{i+1}(S) \quad \forall\ U \subseteq T,\ \forall i = 1, \cdots, n-1, \tag{8}$$

$$Y_i(S) \geq 0 \quad \forall\ S \subseteq T,\ \forall\ i = 1, \cdots, n. \tag{9}$$

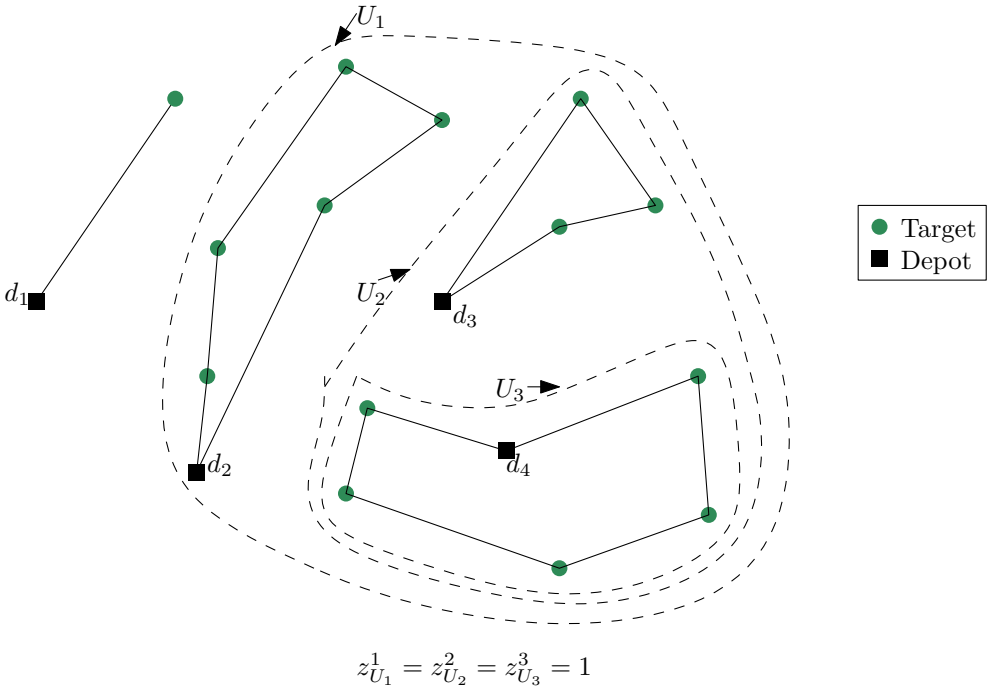

Figure 1 An illustration of a feasible solution involving four vehicles. In this example, $U_1 \supseteq U_2 \supseteq U_3$.

Other than the usual packing constraints in (7), the more interesting constraints (8) restrict the dual value of the subsets of any set U in a cheaper cost level to be at most that accumulated in any higher cost level.

3 Primal-Dual Algorithm

We define some terms before presenting the main steps of the primal-dual algorithm. The algorithm maintains a forest of edges defined over the targets and the respective depot for each vehicle separately, by using the known primal-dual moat-growing procedure separately and simultaneously in each of the graphs (V_i, E_i) [1, 7].

Let $F_i(t)$ be the forest obtained in (V_i, E_i) for the vehicle $i \in \{1, \cdots, n\}$ at the end of iteration t of the main loop. For any two distinct vehicles $i, j \in \{1, \cdots, n\}, i < j$ and any iteration t, consider (connected) components C_i in forest $F_i(t)$ and C_j in forest $F_j(t)$. Because of the monotonicity of costs across the indices, typically the edges between targets in the lower level i will become packed and hence chosen in the forest before those in the higher cost level j. C_i is considered as an ancestor of C_j, or C_j is a descendant of C_i if $C_i \supseteq C_j$. Note that C_j cannot be a descendant of C_i if it contains a depot. Note carefully that we have the subset inclusion only among the targets in the components in different levels but not among the set of tight edges connecting them.

A component can either be active, inactive or frozen. A component is **active** at the start of an iteration if its dual variable will be allowed to increase during the iteration without violating any of the constraints in the dual problem. A component is **inactive** if it either contains a depot or if any of its ancestor contains a depot. A component is **frozen** if it stopped growing due to the constraint in (8), *i.e.*, each descendant of this component is not active as it contains targets connected to some higher level depots. If a component is inactive or frozen at the start of an iteration, its dual value will not change during the iteration.

Initialization. Initially, each forest consists of singleton components which is either a depot or a target. The singleton components with only targets are active; any component with a depot is inactive. The dual variables corresponding to all the active components are initialized to zero.

Main loop. In each iteration of the primal-dual algorithm, the dual variables of all active components in all the forests (in all graphs (V_i, E_i)) are increased simultaneously as much as possible by the same amount until at least one of the constraints in the dual problem becomes tight. If multiple constraints in both (7) and (8) become tight simultaneously during iteration t, only one constraint either in (7) or (8) is chosen and processed based on the following procedure.

- If constraints in (7) become tight, then a tight constraint corresponding to the vehicle **with the least vehicle number** (say i) is chosen. The edge corresponding to the chosen constraint is added to the forest corresponding to vehicle i merging two components at its ends (say C_1 and C_2). If both C_1 and C_2 do not contain a depot, then the merged component is active. If one of the components contains a depot (say $d_i \in C_2$), then the merged component and all its descendants (specifically the descendants of C_1) become inactive.
- If a constraint in (8) becomes tight (*i.e.*, it risks being violated if the current dual variables all continue to grow) for some vehicle i, then the corresponding component is deactivated and becomes frozen.

The main loop of the algorithm stops when each component in all the forests is either inactive or frozen. We will detail some properties of the components before describing the pruning step of the algorithm in Section 3.1.

Remarks.

1. If a component C_1 merges with a component C_2 that contains a depot, the merged component and its descendants are deactivated and will never become active again in the main loop.
2. It is straightforward to compute the maximum possible increase ($\triangle_1$) in the dual variables that do not violate the constraints in (7) using standard techniques involving internal variables for each target in polynomial time. Since we do not grow dual variables more than this amount, it is straightforward to verify that the dual solution we construct obeys the constraints in (7). Specifically, we can use internal variables $p_i(u)$ defined for each target u and vehicle i in the following way: All these internal variables are first set to zero during the initialization. Suppose, at the start of an iteration, C_1 and C_2 are two components corresponding to vehicle i, and $u \in C_1$ and $v \in C_2$. Assume at least one of these components is active. Let $active(C)$ denote if a component C is active or not. For $j = 1, 2$, $active(C_j) = 1$ if C_j is active and is equal to 0 otherwise. Then, the maximum amount by which the dual variable corresponding to C_1 or C_2 can be increased before violating the constraint corresponding to edge $e = (u, v)$ is given by $\frac{Cost_e^i - p_i(u) - p_i(v)}{active(C_1) + active(C_2)}$. This amount can be computed for all such candidate edges and the least of these amounts is equal to $\triangle_1$. During an iteration, if the primal-dual algorithm decides to increase the dual variable of each active component by $\triangle$, then $p_i(u)$ for each target u and vehicle i will be set to $p_i(u) + \triangle$ if the component containing u is active; otherwise $p_i(u)$ doesn't change.
3. Similarly, we can compute the maximum possible increase ($\triangle_2$) in the dual variables that do not violate the constraints in (8), using internal variables defined for each component again in polynomial time. Specifically, we use two internal variables $\bar{Y}_i(C)$ and $Bound_i(C)$

for each component $C \in F_i(t)$, $i = 1, \cdots, n-1$ defined[4] as follows: $\bar{Y}_i(C) := \sum_{S:S\subseteq C} Y_i(C)$ and $Bound_i(C) := \sum_{S:S\subseteq C} Y_{i+1}(C)$. All the internal variables are first set to zero during the initialization ($t = 0$). For any $i = 1, \cdots, n-1$, suppose at the end of iteration t, an active component $C \in F_i(t)$ has m_C active descendants in $F_{i+1}(t)$. Then, during iteration $t+1$, the constraint corresponding to C in (8) can become tight only if $m_C = 0$. In the case $m_C = 0$, the maximum amount by which the dual variable of C can be increased without violating its constraint is given by $Bound_i(C) - \bar{Y}_i(C)$. This amount can be computed for each of components in $\{C : C \text{ is active and } m_C = 0, C \in F_i(t+1), i = 1, \cdots, n-1\}$ and the least of these amounts is equal to $\triangle_2$. In addition, if the primal-dual algorithm decides to increase the dual variable of each active component by $\triangle$ during iteration $t+1$, before any merger occurs, for all $i = 1, \cdots, n-1$, for all active $C \in F_i(t)$, $\bar{Y}_i(C) \leftarrow \bar{Y}_i(C) + \triangle$ and $Bound_i(C) \leftarrow Bound_i(C) + m_C\triangle$; in addition, if two components $C_1, C_2 \in F_i(t)$ merge, $\bar{Y}_i(C_1 \bigcup C_2) \leftarrow \bar{Y}_i(C_1) + \bar{Y}_i(C_2)$ and $Bound_i(C_1 \bigcup C_2) \leftarrow Bound_i(C_1) + Bound_i(C_2)$.

Observations on the Main Loop. We review a few facts that follow from the running of the main loop. Consider any $i \in \{1, \cdots, n\}$ and a vertex $u \in T$. Let $C_i(t, u)$ denote the component containing u in the forest corresponding to vehicle i at the start of iteration t of the main loop. Let $active_i(t, u)$ be an indicator denoting if $C_i(t, u)$ is active or not. That is, $active_i(t, u) = 1$ if $C_i(t, u)$ is active and $active_i(t, u) = 0$ otherwise.

▶ **Lemma 2.** *The main loop of the primal-dual algorithm terminates in* $2n(|T|+1)$ *iterations.*

Proof. At the start of the main loop, the number of components in all the forests is $n(|T|+1)$ and the number of active components in all the forests is $n|T|$. During each iteration of the main loop, the sum of the number of components and the number of active components in all the forests decreases by at least 1. Therefore, the main loop will require at most $2n|T| + n$ iterations. ◀

Note that components in smaller index vehicles typically merge before their corresponding analogues in the larger indices since the distances are shorter in the smaller index metric. Hence, in addition to the laminar structure on the target nodes among these components for a fixed vehicle, there is an inclusion relation among these components when viewing them across the vehicle indices (after we ignore the depots that are present only in their corresponding graphs). The following lemma shows that at any time t' in the main loop of the algorithm, for $j > i$, the set of connected components for vehicle index j are contained in those for vehicle index i. Let the main loop of the primal-dual procedure terminate after t_f iterations. Also, let $\triangle_t$ denote the amount by which the dual value of each active component is increased during iteration t.

▶ **Lemma 3.** *In any iteration* $t' = 1, \cdots, t_f$, *for any vehicles* $i, j \in \{1, \cdots, n\}, i < j$ *and any target* $u \in T$, *the following relations hold true:* $active_i(t', u) \geq active_j(t', u)$ *and* $C_i(t', u) \supseteq C_j(t', u)$ *if* $d_j \notin C_j(t', u)$.

Proof. We prove this lemma by induction on t'. For $t' = 1$, the lemma is trivially satisfied. Assume the lemma is true for iterations $t' = 1, \cdots, t$ for some $t < t_f$.

Suppose that in iteration t, the constraint in (7) becomes tight for some edge (u, v) corresponding to vehicle i . There are two cases.

[4] In the special case when C only consists of d_i, we define $Bound_i(C) = 0$.

- *The merged component does not contain d_i*: In this case, the merged component will be active at the end of the iteration and will be an ancestor to the descendants of $C_i(t,u)$ and $C_i(t,v)$. In addition, if $i \geq 2$, $C_{i-1}(t,u)$ must be the same component as $C_{i-1}(t,v)$. If this is not true, for edge $e=(u,v)$,

$$cost_e^{i-1} - \sum_{t'=1}^{t-1}(active_{i-1}(t',u) + active_{i-1}(t',v))\triangle_{t'} \leq cost_e^{i} - \sum_{t'=1}^{t-1}(active_i(t',u) + active_i(t',v))\triangle_{t'}. \quad (10)$$

  Therefore, during iteration t, the algorithm would have added (u,v) to the forest corresponding to vehicle $i-1$ due to our rule in processing the vehicle with the least index in the main loop, which is a contradiction. One can now verify that for any target u, $active_i(t+1,u) \geq active_j(t+1,u)$ and $C_i(t+1,u) \supseteq C_j(t+1,u)$ if $d_j \notin C_j(t+1,u)$.
- *Merged component contains d_i*: If $d_i \in C_i(t,u)$, then the merged component and all the descendants of the merged component also become inactive. Again, one can verify the lemma is true for iteration $t+1$.

Suppose a constraint in (8) becomes tight for some component C. If C is frozen, then C cannot have any descendants that are active. Again, it is easy to check that the lemma is true for iteration $t+1$. ◀

▶ **Lemma 4.** *Consider a frozen component C corresponding to vehicle $i \in \{1, \cdots, n-1\}$ at the start of iteration t. Consider any target $u \in C$. Then, either $C_{i+1}(t,u)$ is frozen and $C_{i+1}(t,u) \subseteq C$ or $C_{i+1}(t,u)$ contains d_{i+1} and $C_{i+1}(t,u) \setminus \{d_{i+1}\} \subseteq C$.*

Proof. If $d_{i+1} \notin C_{i+1}(t,u)$, using Lemma 3, $C_{i+1}(t,u) \subseteq C_i(t,u) = C$; also, since $C_{i+1}(t,u)$ is not active and cannot[5] have an ancestor that is connected to a depot, $C_{i+1}(t,u)$ is frozen. If $d_{i+1} \in C_{i+1}(t,u)$, then any target $v \in C_{i+1}(t,u) \setminus \{d_{i+1}\}$ must also belong to C. If this is not the case, there is an edge joining targets w and v in component $C_{i+1}(t,u)$ such that $w \in C$ and $v \notin C$. From Lemma 3, using a similar argument as in (10), this is not possible. ◀

3.1 Pruning

For $i = 1, \cdots, n$, the pruning phase of the primal-dual procedure selects a subgraph $\bar{F}_i$ of $F_i(t_f)$ such that each target is connected to exactly one of the depots. We need a few definitions before describing the pruning procedure. The degree of a subgraph C of forest F is defined as $|\{(u,v) : u \in C, v \notin C, (u,v) \in F\}|$. A subgraph C of $F_i(t_f)$ is referred as a **frozen subgraph** if C is a component that is frozen during some iteration $t \leq t_f$. A subgraph C of $F_i(t_f)$ is referred to as a **pendent-frozen subgraph** if C is a frozen subgraph and its degree is equal to 1 (Ref to Fig. 2). A **maximal pendent-frozen subgraph** is a pendent-frozen subgraph $C \in F_i(t)$ such that there is no other pendent-frozen subgraph $C' \in F_i(t)$ with $C' \supset C$. Given vehicle i, iteration t of the main loop and a component C spanning a subset of targets, let $F_i^C(t)$ be a subgraph of $F_i(t)$ induced by d_i and the targets in C. It follows from this definition that if C was frozen during iteration t of the main loop for vehicle i, then for all $j = i, i+1, \cdots, n$, $F_j^C(t) = F_j^C(t_f)$.

The pruning procedure is implemented in n iterations. Let $i := 1$ at the start of the pruning procedure and let $G_1 = F_1(t_f)$.

[5] If $C_{i+1}(t,u)$ has an ancestor that is connected to a depot, then C also has an ancestor that is connected to a depot which makes C inactive and this is not possible.

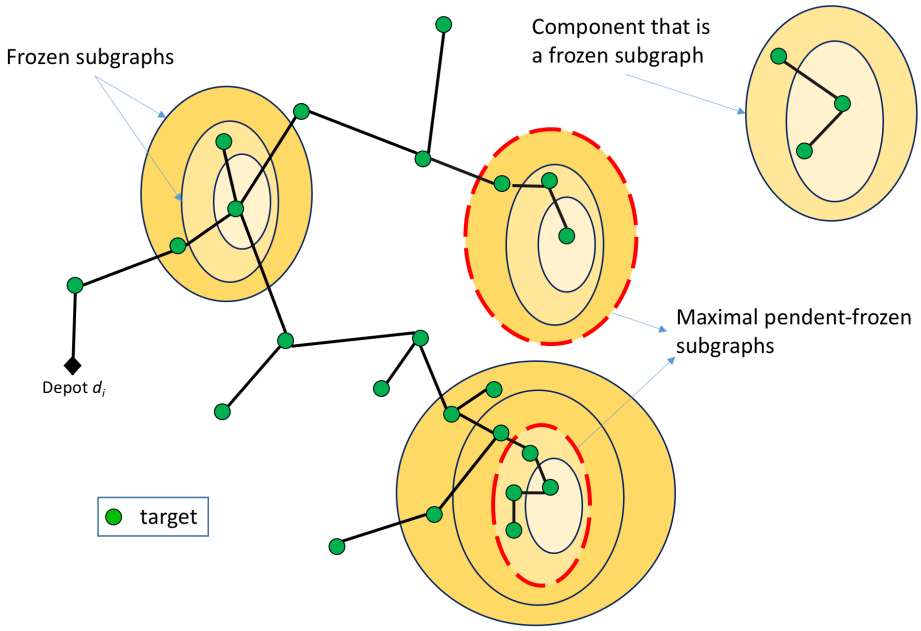

Figure 2 An illustration of forest G_i corresponding to vehicle i. Each shaded region corresponds to a frozen subgraph.

1. Remove all the frozen subgraphs that are components (not containing d_i) from G_i. Furthermore, in the tree containing d_i in G_i, remove all the maximal pendent-frozen subgraphs. The resultant pruned tree is $\bar{F}_i$ (Refer to Figs. 2, 3).
2. If $i = n$, stop. Else, let G_{i+1} be the union of all the subgraphs $F^C_{i+1}(t_C)$ obtained for every frozen subgraph C frozen at time t_C and discarded from G_i in the previous step. Set $i = i + 1$ and go to step 1.

Intuitively, frozen components contain targets connected to depots in higher index graphs and hence the pruning step discards them for processing in an appropriate (later) iteration. Similarly, maximal pendent-frozen subgraphs must be pruned so as to ensure that no inactive component in this index contributes degree one in the standard degree-based inductive argument for the 2-approximation ratio in the primal-dual method [7].

3.2 Feasibility

We are ready to prove the following main theorem.

▶ **Theorem 5.** *Each target in T is connected to exactly one of the depots in $\{\bar{F}_1, \cdots, \bar{F}_k\}$, i.e, $\{\bar{F}_1, \cdots, \bar{F}_k\}$ is a feasible Heterogeneous Spanning Forest (HSF).*

Proof. We will prove the Theorem by induction on the stages of the pruning procedure.

▶ **Lemma 6.** *Consider any iteration k of the pruning procedure. Let the subset of targets connected to d_i in $\bar{F}_i$ be denoted as T_i. Then, each target in $\bigcup_{i=1}^{k} T_i$ is connected to exactly one of the depots in $\{\bar{F}_1, \cdots, \bar{F}_k\}$ and the remaining targets are contained in the frozen subgraphs discarded in G_k.*

Proof. Clearly the lemma is true for $k = 1$. Assume that the lemma is true for $k = k' - 1$. We will now show that the lemma is true for $k = k'$ also. Applying Lemma 4 to each discarded frozen subgraph in $G_{k'-1}$, all the targets in $T \setminus \bigcup_{i=1}^{k'-1} T_i$ are either present in the

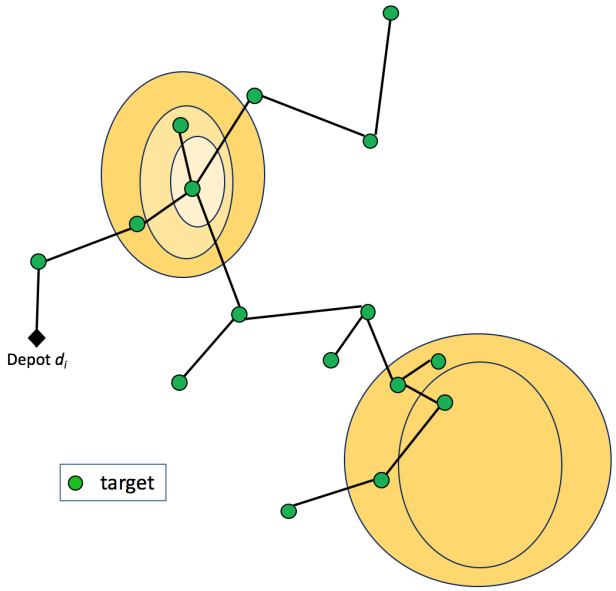

Figure 3 Output forest $\bar{F}_i$ after removing the maximal pendent-frozen subgraphs and frozen subgraphs that are components from G_i.

frozen components of $G_{k'}$ or connected to $d_{k'}$ using only targets from $T \setminus \bigcup_{i=1}^{k'-1} T_i$. By the definition of the pruning step on $G_{k'}$, the targets that are pruned away do not affect the connectivity from $d_{k'}$ to the targets retained in $T_{k'}$. Hence the induction step is proved. ◀

By the above lemma, when $k = n$, we see that the targets T_i covered in the various iterations form a partition of T. ◀

4 Approximation Guarantee

▶ **Theorem 7.** *The approximation ratio of the primal-dual algorithm for MDHTSP is 2.*

We show that the dual value of the LP relaxation can be equivalently written as the sum of the dual values corresponding to each of the vehicles (Lemma 8). This will allow us to bound the cost of the edges in each forest with respect to its dual value (Lemma 9). The approximation ratio will readily follow from these results.

Consider any vehicle $i \in \{1, \cdots, n-1\}$. Let $C_{i1}, \cdots, C_{im_i}$ denote the discarded, frozen subgraphs of G_i (as defined in the pruning procedure) for vehicle i. Also, let the subset of targets in all these discarded components be U_i. To simplify the ensuing derivation (with a slight abuse of notation), we also refer to C as a subset of targets present in component C.

▶ **Lemma 8.**

$$\sum_{S \subseteq T} Y_1(S) = \sum_{S \subseteq T, S \not\subseteq U_1} Y_1(S) + \sum_{j=2}^{n-1} \sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S) + \sum_{S \subseteq U_{n-1}} Y_n(S).$$

Proof. Note that

$$\sum_{S \subseteq T} Y_1(S) = \sum_{S \subseteq T, S \not\subseteq U_1} Y_1(S) + \sum_{S \subseteq U_1} Y_1(S). \tag{11}$$

For any $j = 2, \cdots, n-1$,

$$\sum_{S \subseteq U_{j-1}} Y_{j-1}(S) = \sum_{k=1}^{m_{j-1}} \sum_{S \subseteq C_{(j-1)k}} Y_{j-1}(S).$$

Each $C_{(j-1)k}$ is a frozen component. Therefore, its corresponding constraint in (8) is tight.

$$\Rightarrow \sum_{S \subseteq U_{j-1}} Y_{j-1}(S) = \sum_{k=1}^{m_{j-1}} \sum_{S \subseteq C_{(j-1)k}} Y_j(S) = \sum_{S \subseteq U_{j-1}} Y_j(S)$$
$$= \sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S) + \sum_{S \subseteq U_j} Y_j(S).$$

Applying the above relation recursively in equation (11), the lemma follows. ◀

For any $j = 1, \cdots, n$, let $Cost(\bar{F}_j) = \sum_{e \in \bar{F}_j} cost_e^j$.

▶ **Lemma 9.** *For any $j = 2, \cdots, n-1$,*

$$Cost(\bar{F}_j) \leq 2 \sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S).$$

Proof.

$$Cost(\bar{F}_j) = \sum_{e \in \bar{F}_j} cost_e^j = \sum_{e \in \bar{F}_j} \sum_{S: e \in \delta_j(S)} Y_j(S) = \sum_{S \subseteq T} Y_j(S) |\delta_j(S) \bigcap \bar{F}_j|.$$

Note that from Lemma 3, any $S \subseteq T$ that loaded an edge $e \in \bar{F}_j$ must be a subset of U_{j-1}. In addition, $|\delta_j(S) \bigcap \bar{F}_j| = 0$ for any $S \subseteq U_j$, since U_j was discarded in the pruning. Therefore, we get

$$Cost(\bar{F}_j) = \sum_{S \subseteq T} Y_j(S) |\delta_j(S) \bigcap \bar{F}_j| = \sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S) |\delta_j(S) \bigcap \bar{F}_j|.$$

Therefore, the lemma reduces to proving that

$$\sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S) |\delta_j(S) \bigcap \bar{F}_j| \leq 2 \sum_{S \subseteq U_{j-1}, S \not\subseteq U_j} Y_j(S). \tag{12}$$

The above result can be proved by induction on the main loop, using the usual degree argument for such primal-dual algorithms [7]. At the start of any iteration t and vehicle j, let A denote the set of active components defined as follows: $A := \{C : C \text{ is active}, C \subseteq U_{j-1}, C \not\subseteq U_j\}$. Similarly, let I denote the set of inactive or frozen components defined as follows $I := \{C : C \text{ is inactive or frozen}, C \subseteq U_{j-1}, C \not\subseteq U_j\}$. Form a graph H with components in $A \cup I$ as vertices and $e \in \bar{F}_j \cap \delta_j(C)$ for $C \in A \cup I$ as edges.

Let $deg(u)$ represent the degree of a vertex u in graph H. Let the dual variable of each active component during the iteration increase by $\triangle_t$. Due to this dual increase, the right hand side of the inequality (12) will increase by $2\triangle_t |A|$, whereas the left hand side of the inequality (12) will increase by $\triangle_t \sum_{u \in A} deg(u)$. Therefore, the lemma is proved if we can show that $\sum_{u \in A} deg(u) \leq 2|A|$.

Note that H is a tree that spans all the the components in $A \cup I$. Therefore, $deg(u) \geq 1$ for any component u in $A \cup I$. There is exactly one inactive component in I, and this inactive component contains d_j[6]. In addition, for any vertex u that represents a frozen component,

[6] A component in a forest corresponding to vehicle j can also be inactive if its ancestor is connected to a depot. But from Lemma 3, such a component never becomes active again and never gets connected to d_j.

$deg(u) \geq 2$ due to the pruning procedure that discards maximal pendant-frozen subgraphs. Therefore,

$$\begin{aligned}\sum_{u\in A} deg(v) &= \sum_{u\in A\cup I} deg(u) - \sum_{u\in I} deg(u)\\ &= \sum_{u\in A\cup I} deg(u) - \sum_{u\in\{C:C\in I, d_j\notin C\}} deg(u) - \sum_{u\in\{C:C\in I, d_j\in C\}} deg(u)\\ &\leq 2(|A|+|I|-1) - 2(|I|-1) - 1\\ &< 2|A|.\end{aligned}$$

◀

Similarly, one can also show that $Cost(\bar{F}_1) \leq 2\sum_{S\subseteq T, S\not\subseteq U_1} Y_1(S)$ and $Cost(\bar{F}_n) \leq 2\sum_{S\subseteq U_{n-1}} Y_n(S)$. Hence, using Lemma 8, we get

$$\begin{aligned}\sum_{i=1}^{n} Cost(\bar{F}_i) \leq& 2\sum_{S\subseteq T, S\not\subseteq U_1} Y_1(S) + 2\sum_{j=1}^{n-2}\sum_{S\subseteq U_j, S\not\subseteq U_{j+1}} Y_{j+1}(S) + 2\sum_{S\subseteq U_{n-1}} Y_n(S)\\ =& 2\sum_{S\subseteq T} Y_1(S) \leq Cost_{lp}.\end{aligned}$$

Therefore, the cost of the constructed HSF will be at most the optimal cost of the MDHTSP. Doubling the edges in the constructed HSF and short cutting the repeated visits to the targets in an Euler walk suitably leads to a 2-approximation algorithm for the MDHTSP.

References

1. Ajit Agrawal, Philip Klein, and Ramamoorthi Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM journal on Computing*, 24(3):440–456, 1995.
2. Jung Yun Bae and Sivakumar Rathinam. A primal-dual approximation algorithm for a two depot heterogeneous traveling salesman problem. *Optimization Letters*, 10, 2015. doi:10.1007/s11590-015-0924-1.
3. Jungyun Bae and Sivakumar Rathinam. Approximation algorithms for multiple terminal, hamiltonian path problems. *Optimization Letters*, 6(1):69–85, 2012. doi:10.1007/s11590-010-0252-4.
4. Bruno N. Coelho, Vitor N. Coelho, Igor M. Coelho, Luiz S. Ochi, Roozbeh Haghnazar K., Demetrius Zuidema, Milton S.F. Lima, and Adilson R. da Costa. A multi-objective green uav routing problem. *Comput. Oper. Res.*, 88(C):306–315, December 2017. doi:10.1016/j.cor.2017.04.011.
5. R. Doshi, S. Yadlapalli, S. Rathinam, and S. Darbha. Approximation algorithms and heuristics for a 2-depot, heterogeneous hamiltonian path problem. *International Journal of Robust and Nonlinear Control*, 21(12):1434–1451, 2011. doi:10.1002/rnc.1701.
6. A.M. Frieze. An extension of christofides heuristic to the k-person travelling salesman problem. *Discrete Applied Mathematics*, 6(1):79–83, 1983. doi:10.1016/0166-218X(83)90102-6.
7. Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, April 1995.
8. I. Gortz, M. Molinaro, V. Nagarajan, and R. Ravi. *Capacitated Vehicle Routing with Non-uniform Speeds*, volume 6655 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2011.
9. Samir Khuller, Azarakhsh Malekian, and Julián Mestre. To fill or not to fill: The gas station problem. *ACM Trans. Algorithms*, 7(3):36:1–36:16, July 2011. doi:10.1145/1978782.1978791.

10 David Levy, Kaarthik Sundar, and Sivakumar Rathinam. Heuristics for routing heterogeneous unmanned vehicles with fuel constraints. *Mathematical Problems in Engineering*, 2014, 2014.

11 Parikshit Maini and P.B. Sujit. On cooperation between a fuel constrained uav and a refueling ugv for large scale mapping applications. In *2015 International Conference on Unmanned Aircraft Systems*, 2015. doi:10.1109/ICUAS.2015.7152432.

12 W. Malik, S. Rathinam, and S. Darbha. An approximation algorithm for a symmetric generalized multiple depot, multiple travelling salesman problem. *Operations Research Letters*, 35:747–753, 2007.

13 P. Oberlin, S. Rathinam, and S. Darbha. Today's traveling salesman problem. *IEEE Robotics and Automation Magazine*, 17(4):70–77, December 2010.

14 Alena Otto, Niels A. H. Agatz, James F. Campbell, Bruce L. Golden, and Erwin Pesch. Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks*, 72(4):411–458, 2018. doi:10.1002/net.21818.

15 Stefan Poikonen, Xingyin Wang, and Bruce Golden. The vehicle routing problem with drones: Extended models and connections. *Networks*, 70(1):34–43, 2017. doi:10.1002/net.21746.

16 Amritha Prasad, Shreyas Sundaram, and Han-Lim Choi. Min-max tours for task allocation to heterogeneous agents. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1706–1711, 2018. doi:10.1109/CDC.2018.8619118.

17 S. Rathinam, R. Sengupta, and S. Darbha. A resource allocation algorithm for multivehicle systems with nonholonomic constraints. *IEEE Transactions Automation Science and Engineering*, 4:98–104, 2007. doi:10.1109/TASE.2006.872110.

18 Sivakumar Rathinam and Raja Sengupta. 3/2-approximation algorithm for two variants of a 2-depot hamiltonian path problem. *Operations Research Letters*, 38(1):63–68, 2010. doi:10.1016/j.orl.2009.10.001.

19 J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific J. Math.*, 145(2):367–393, 1990.

20 Kaarthik Sundar and Sivakumar Rathinam. An exact algorithm for a heterogeneous, multiple depot, multiple traveling salesman problem. In *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*, pages 366–371. IEEE, 2015.

21 Zhou Xu and Brian Rodrigues. An extension of the christofides heuristic for the generalized multiple depot multiple traveling salesmen problem. *European Journal of Operational Research*, 257(3):735–745, 2017. doi:10.1016/j.ejor.2016.08.054.

22 S. Yadlapalli, S. Rathinam, and S. Darbha. 3-approximation algorithm for a two depot, heterogeneous traveling salesman problem. *Optimization Letters*, pages 1–12, 2010.

23 Miao Yu, Viswanath Nagarajan, and Siqian Shen. An approximation algorithm for vehicle routing with compatibility constraints. *Operations Research Letters*, 46(6):579–584, 2018. doi:10.1016/j.orl.2018.10.002.