Approximation Algorithms for Distance Constrained Vehicle Routing Problems

Viswanath Nagarajan[*] R. Ravi[†]

Tepper School of Business, Carnegie Mellon University, Pittsburgh PA 15213.

Email: {viswa,ravi}@cmu.edu

October 17, 2008

Abstract

We study the *distance constrained vehicle routing* problem (DVRP) [1, 2]: given a set of vertices in a metric space, a specified depot, and a distance bound D, find a minimum cardinality set of tours originating at the depot that covers all vertices, such that each tour has length at most D. This problem is NP-complete, even when the underlying metric is induced by a weighted star.

We design efficient approximation algorithms for distance constrained vehicle routing, and also study the integrality gap of a natural set-covering based LP relaxation for the problem. We present a 2-approximation algorithm for the problem on tree metrics, and an $O(\log D)$ approximation algorithm on general metrics. This algorithm can also be used to obtain a continuous trade-off between the violation of the distance bound and the number of tours in the approximate solution. Our methods can incorporate additional capacity bounds on the vehicles with a slight loss in the performance ratio. We extend our analysis to show a constant factor integrality gap for DVRP on tree metrics, and an $O(\log D)$ integrality gap in general. The unrooted version of the problem (where tours need not originate at the same depot) is known to have a constant factor approximation

[*]Supported by NSF ITR grant CCR-0122581 (The ALADDIN project) and CCF-0728841

[†]Supported in part by NSF grants CCF-0430751, CCF-0728841 and ITR grant CCR-0122581 (The ALADDIN project)

guarantee [3]; we obtain a constant integrality gap for this case.

Keywords: Vehicle Routing, Traveling Salesman Problem, Approximation Algorithms, Linear Programming relaxations.

1 Introduction

1.1 Motivation

At the core of logistics operations facing modern firms is the problem of routing materials to and from manufacturing or consolidating depots at minimum cost [4, 5]. The most common constraints on such problems involve the capacity of the vehicles and deadlines on delivery/pickup of materials. Typical cost objectives are the total mileage of all the routes or more coarsely, the number of vehicles deployed to satisfy the demands. In this paper, we study the problem with distance constraints on each route where the objective is to minimize the number of vehicles, called the *Distance Constrained Vehicle Routing Problem (DVRP)* in the literature [4, 1, 2].

A bound on the distance traveled by any vehicle arises commonly, e.g., in scheduling daily routes for courier carriers or milkruns from manufacturing facilities. The distance bound translates to a quality of service guarantee for all customers to be served on the day they are scheduled. Minimizing the number of vehicles over a period of typical demands also allows for better fleet and driver planning and management. However, these problems generalize the classical TSP and are NP-complete.

In this paper, we obtain approximation algorithms for distance constrained vehicle routing problems. We use the well studied notion of approximation guarantees [6, 7] to measure the performance of heuristics. An approximation algorithm for a minimization problem is said to achieve an approximation ratio α (which may be a function of the input instance), if on every instance, the cost of the solution obtained by the algorithm is at most α times the cost of an optimal solution. Such an algorithm is also referred to as an α-approximation algorithm.

1.2 Problem formulation

We model demand locations as vertices in a finite metric space (V, d), with $|V| = n$. The distance function $d : V \times V \to \mathbb{N}$ is symmetric and satisfies the triangle inequality. Throughout this paper we assume that all distances are integral: this can be ensured by a suitable scaling. The input to the *distance constrained vehicle routing problem* (DVRP) is specified by a metric space (V, d), a depot $r \in V$, and a distance constraint D. The objective is to find a *minimum cardinality* set of tours originating from r (corresponding to routes for vehicles), that covers all the vertices in V. Each tour is required to have length at most D (the distance constraint). Tours originating from r are referred to as *r-tours.* The maximum distance of any vertex from the depot is denoted by Δ. We assume that $\Delta \leq \frac{D}{2}$, as otherwise there is no feasible solution.

We consider the natural set covering formulation of distance constrained vehicle routing. This is a special case of the set partitioning model for vehicle routing with time windows, studied in [8]. There is a binary variable x_τ for every r-tour τ of length at most D. The constraints require that every vertex be covered by at least one such r-tour. The LP relaxation is obtained by dropping the integrality on the variables and is as follows.

$$\begin{array}{lll} & \min \sum_\tau x_\tau & \\ & s.t. & \\ (\mathcal{LP}) & \sum_{\tau : v \in \tau} x_\tau \geq 1 & \forall v \in V \setminus r \\ & x_\tau \geq 0 & \forall \tau : r\text{-tour of length at most } D \end{array}$$

Although this LP has an exponential number of variables, it can be solved approximately in polynomial time. The dual to this linear program has a variable for every vertex and an exponential number of constraints; the separation routine for these dual constraints is the *orienteering problem*: given profits on vertices, root r, and length bound D, find an r-tour of length at most D that collects the maximum profit. There is a polynomial time 3-approximation algorithm [9] for orienteering. This implies that we can obtain a 3-approximation to $\mathcal{LP}$ in polynomial time, using the ellipsoid algorithm on its dual (see eg., Carr and Vempala [10], Section 2.1 for a rigorous argument).

The *unrooted DVRP* [3] is defined as follows: given a metric space (V, d) and a distance constraint D, find a minimum cardinality set of paths (each of length at most D) that covers all the vertices. Note that in this version, the vehicle routes are paths that are allowed to start and end at any two vertices. The unrooted version can be reduced to DVRP by adding a root vertex that is located at some large distance $L \gg \mathit{diameter}(V)$ from all vertices in V, and setting the distance constraint to $D + 2L$.

1.3 Our Results

First we study the DVRP on metrics induced by an underlying edge-weighted tree (Section 2). In this case, we obtain a 2-approximation algorithm. This algorithm can be implemented in a single depth first search of the tree and runs in linear time. We note that distance constrained vehicle routing is NP-complete even in the special case when the tree is a star, via a reduction from 3-Partition [11]. For DVRP on general metrics, we obtain an $O(\log D)$-approximation algorithm (Section 3), which has a running time of $O(n^2 \log n \cdot \log D)$. Our algorithm can also be adapted to give for any $\epsilon > 0$, an $(O(\log \frac{1}{\epsilon}), 1+\epsilon)$ *bi-criteria approximation*: If the vehicles are allowed to violate the distance constraint by a small factor ϵ, then we obtain a solution using at most $O(\log \frac{1}{\epsilon})$ times the optimal number of vehicles that do not violate the distance constraint. As shown in Jothi and Raghavachari [12] the tour-partitioning algorithm of Li et al. [2] gives an $(O(\frac{1}{\epsilon}), 1+\epsilon)$ bi-criteria approximation for DVRP. We improve upon the approximation ratio on the number of vehicles significantly.

Next we consider the set-covering based LP-relaxation for DVRP ($\mathcal{LP}$). We show constant factor integrality gaps for two special cases: DVRP on tree metrics has integrality gap at most 20 (Section 4.1), and unrooted DVRP has integrality gap at most 17 (Section 4.2). We note that a 3-approximation algorithm for unrooted DVRP is known due to Arkin et al. [3], however this algorithm is not LP-based. Combining our algorithm for DVRP on general metrics with the integrality gap for unrooted DVRP, we also obtain an $O(\min\{\log D, \log n\})$ bound on the integrality gap of general DVRP. Our LP-relaxation $\mathcal{LP}$ is a special case of the set-partitioning model for vehicle routing with time windows [8] (i.e. when all time windows are identical and have width equal to the distance bound). Although it has been observed computationally in

Desrochers et al. [8] that this LP-relaxation provides excellent bounds on the optimal value, there are no theoretical upper bounds on the integrality gap. Our results provide worst case bounds on the integrality gap, for the special case of distance constrained vehicle routing.

Finally, we close with some observations on closely related problems in Section 5. These extensions involve adding capacity constraints to DVRP and the variant where tours are replaced by paths originating at the depot.

Remark: It is immediate to see that the integrality gap of $\mathcal{LP}$ is at most $O(\log n)$, by means of randomized rounding applied to the optimal fractional solution (as in set cover, c.f. [7]). In fact this observation holds for much more general classes of problems such as vehicle routing with time-windows; however this argument is non-constructive (since it assumes that a near-optimal solution to the linear relaxation is available). Our results on the integrality gap for DVRP improve upon this naïve bound and are also constructive.

1.4 Related Work

Vehicle routing problems (VRPs) are surveyed in [4, 5]. Practical applications of DVRP can be found in Assad [4] and Laporte et al. [1]. Exact approaches for the objective of minimizing total distance were studied in Laporte et al. [1]. They gave two algorithms using an integer programming formulation: one based on Gomory cuts and the other using branch-and-bound.

Li et al. [2] studied DVRP under the objective functions of total distance and number of vehicles. They showed that the optimal solutions under both objectives are closely related, and any approximation guarantee for one objective implies a guarantee with an additional loss of factor 2, for the other objective. They also studied a tour-partitioning heuristic for this problem, which was shown to achieve a worst case performance guarantee of D.

Although there is a large body of work on both exact and heuristic approaches for VRPs, there is relatively less work on proving performance guarantees of heuristics. There has been some interesting work on approximating the related *orienteering* problem (defined earlier). Improving on work by Blum et al. [13] and Bansal et al. [9], Chekuri et al. [14] presented a $2 + \epsilon$ approximation algorithm (for any constant $\epsilon > 0$) for the orienteering problem. Using this as a greedy subroutine within a set-covering framework, it is straightforward to design an $O(\log n)$ approximation for DVRP.

The distance constrained VRP was also studied by Bazgan et al. [15], where the authors gave a constant-factor *differential* approximation algorithm. However, bounds in the differential measure do not imply any bounds in the standard (multiplicative) approximation measure, which we consider in this paper.

Related to the tree metric version we study, Labbe et al. [16] and Karuno et al. [17] discuss some practical situations where tree shaped networks are encountered in VRPs. In the case of capacitated VRP (only capacity constraints), Labbe et al. [16] gave a 2-approximation algorithm on trees when demands are unsplittable. When demands are splittable, Hagamochi and Katoh [18], and Asano et al. [19] gave improved approximation algorithms; the currently best known guarantee is ≈ 1.35. Many other vehicle routing problems on trees have been studied in [20, 21, 17].

The vehicle routing problem with time windows [8, 22] is a generalization of DVRP, where each demand location can only be served in a particular time window. The set covering formulation for DVRP that we study is a special case of a more general formulation for VRP with time windows. Desrochers et al. [8] considered a natural set partitioning formulation for VRP with time windows, which was solved optimally in a branch-and-bound algorithm. The LP relaxation of this integer program was solved using column generation, and the authors noted that this LP relaxation was generally an excellent lower bound. We are not aware of any worst-case bounds on the integrality gap of this LP, even in special cases of VRP with time windows.

2 DVRP on tree metrics

In this section, we consider the special case of DVRP when the metric space is induced by a weighted tree $T = (V, d)$. Even in the special case of a star, the problem remains NP-complete (by a reduction from 3-Partition). Here we present a 2-approximation for DVRP on trees. The main ingredient in this is proving a lower bound on the optimal number of vehicles, which is based on forming clusters of vertices that can not be covered by a single r-tour (Lemma 2).

For ease of description, we assume (without loss of generality) that the tree T is binary, and rooted at the depot r. This can be ensured by splitting high degree vertices, and adding zero-length edges. Algorithm *minTVR* for DVRP on trees is as follows.

1. Initialize $T' = T$.
2. While $(T' \neq \{r\})$ do
    (a) Pick a deepest vertex $v \in T'$ s.t. the subtree T'_v below v *can not* be covered by just one r-tour, of length at most D. If no such v exists, add an r-tour covering T', and END.
    (b) Let w_1 and w_2 be the two children of v. For $i = 1, 2$, set W_i to be the minimum length r-tour traversing the subtree below w_i.
    (c) Add r-tours W_1 and W_2 to the solution.
    (d) $T' = T' \setminus T'_v$.

Note that the minimum length r-tour covering all the vertices of a subtree is just an Euler tour of the subtree (including the path from r), traversing each edge twice. Thus the condition in step 2a can be checked efficiently.

Theorem 1 *Algorithm minTVR obtains a 2-approximation to DVRP on trees.*

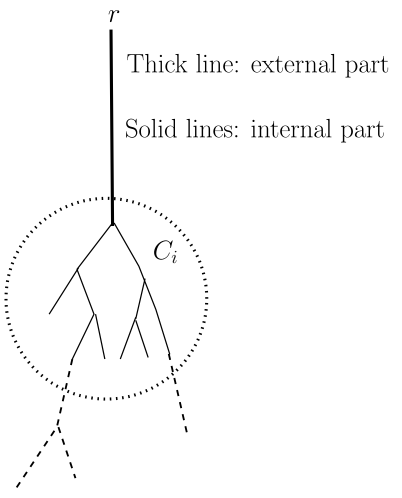
Proof: It is not hard to see that algorithm *minTVR* can be implemented in a single depth-first search of the tree; so the time complexity is linear in the input size. From the choice of vertex v in step 2a, each r-tour added in step 2c (corresponding to the children of v), has length at most D. So algorithm *minTVR* indeed produces a feasible solution.

A *heavy cluster* is defined to be a set of vertices $C \subseteq V$ such that the subgraph $T[C]$ induced by C on tree T is connected, and the vertices in C can *not* all be covered by a single r-tour of length at most D. Note that all the subtrees T'_v seen in step 2a of algorithm *minTVR* are heavy clusters in tree T. Suppose, in its entire execution, the algorithm finds k heavy clusters $C_1, \cdots C_k$ (these vertex sets will be disjoint). Then algorithm *minTVR* produces a solution using at most $2k+1$ r-tours. The key lemma is the following, which shows that the optimal solution requires at least $k+1$ vehicles, and thus proves Theorem 1.

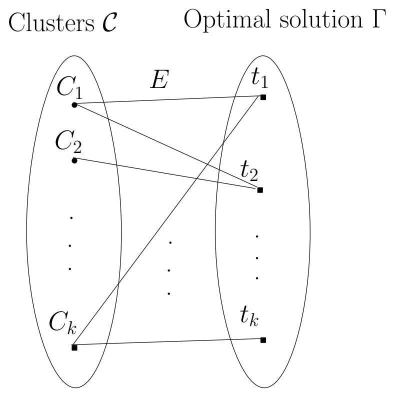
Lemma 2 *If there are k disjoint heavy clusters $C_1, \cdots C_k \subseteq V$ in the tree T, the minimum number of r-tours (of length at most D) required to cover $\bigcup_{i=1}^k C_i$ is more than k.*

Proof: The proof of this lemma is by induction on k. For $k = 1$, the lemma is trivially true. Suppose $k > 1$, and assume that the lemma holds for all values up to $k-1$. Suppose, for contradiction, that the minimum number of r-tours required to cover all these clusters, $OPT \leq k$. Note that OPT can not be smaller than k: taking any $k-1$ of these k clusters, we would get a contradiction to the induction hypothesis with $k-1$ clusters. So we may assume $OPT = k$. In the rest of the proof, fix an optimal solution consisting of r-tours $t_1, \cdots, t_k$.

From the definition of a heavy cluster, each C_i forms a connected subtree in T. It will be convenient to think of the lengths associated with C_i in the following parts (see Figure 1a): the path from r to the highest vertex in C_i (*external part*); and the induced subgraph $T[C_i]$ (*internal part*). The length of the external part of a cluster C_i is denoted $d(r, C_i)$. We now define a bipartite graph $H = (\Gamma, \mathcal{C}, E)$ where $\Gamma = \{t_1, \cdots, t_k\}$ is the set of r-tours in the optimal solution, and $\mathcal{C} = \{C_1, \cdots, C_k\}$ is the set of the k heavy clusters (see Figure 1b). There is an edge $(t_j, C_i) \in E$ iff r-tour t_j visits some vertex of cluster C_i.



(a) Lengths associated with heavy cluster



(b) The graph H

Figure 1: DVRP on trees

We claim that H must have a perfect matching between $\mathcal{C}$ and Γ. Suppose not - then by Hall's Theorem, we get a set $S \subseteq \mathcal{C}$ such that S has fewer than $|S|$ neighbors in Γ. Note that $S \neq \mathcal{C}$, as $\mathcal{C}$ has $OPT = |\mathcal{C}|$ neighbors. This implies that the clusters in S are visited completely by fewer than $|S|$ r-tours, which contradicts the induction hypothesis with the set of heavy clusters S (as $|S| < k$). Thus H has a perfect matching $\pi : \mathcal{C} \rightarrow \Gamma$.

Let $l_1, l_2, \cdots, l_k$ denote the lengths of the r-tours in Γ; clearly each $l_i \leq D$. We assign a *capacity* to each edge $e \in T$: $cap_e = 2\sum_{j=1}^{k} I_e(t_j)$, where $I_e(t_j) = 1$ iff edge e is traversed in r-tour t_j, and 0 otherwise. Note that if an edge is traversed in an r-tour, it is traversed at least 2 times; so each edge in T has capacity at least 2 (as each vertex is visited). Now, the total weighted capacity over all edges is exactly $\sum_{e\in T} d_e \cdot (2\sum_{j=1}^{k} I_e(t_j)) \leq \sum_{i=1}^{k} l_i \leq kD$.

We will now *charge* each edge an amount at most its capacity, and show that the total weighted charge over all edges is larger than kD, which would be a contradiction. Corresponding to every cluster C_i, charge each edge in its external part (the path from r to C_i) two units against the capacity on that edge attributed to r-tour $\pi(C_i)$; note that tour $\pi(C_i)$ visits C_i and hence traverses all the edges from r to C_i. Since π is a perfect matching, no edge has a charge more than its capacity. The total weighted charge after this step is exactly $2\sum_{i=1}^{k} d(r, C_i)$. Now we will further assign a charge of 2 units to each edge in the internal part of every cluster $C_1, \cdots, C_k$.

Consider any edge e on the internal part of some cluster C_i. Let m denote the number of clusters that appear *below* e in tree T (this does not include C_i). If $m = 0$, this edge has never been charged so far, and thus has at least 2 units of residual capacity. If $0 < m \leq k-1$, then applying induction on the set of m clusters below e, there are at least $m+1$ r-tours that traverse e. So e has a capacity of at least $2m+2$. But we have charged e exactly $2m$ units so far, 2 units corresponding to each cluster below it. So again we have at least 2 units of residual capacity, and we can charge this edge an extra 2 units. The total weighted charge over all edges can now be written as follows:

$$\sum_{i=1}^{k} \left[2d(r, C_i) + 2 \cdot d(\text{internal part of } C_i)\right]$$

The i-th term above corresponds to an r-tour covering C_i. Since each C_i is a heavy cluster, this is more than D. So the total weighted charge is more than $kD \geq$ the total capacity, which is a contradiction. Thus $OPT > k$, and the lemma is proved.■

3 DVRP on general metrics

In this section, we study DVRP on general metrics, and present an approximation algorithm achieving a guarantee of $O(\log \frac{D}{D-2\Delta+2}) \leq O(\log D)$. The approximation guarantee of the tour-partitioning heuristic of Li et al. [2] was $\frac{D}{D-2\Delta+2}$. As mentioned, using orienteering as a subproblem in a greedy algorithm, one can obtain an $O(\log n)$ approximation algorithm for this problem (even though there is a constant factor approximation algorithm for orienteering, the greedy framework only gives an $O(\log n)$ guarantee). However, due to the large running time of the algorithm for orienteering [9], this approach also has a large running time. On the other hand, the algorithm that we present here has a much smaller running time of $O(n^2 \cdot \log n \cdot \log D)$. We note that the approximation guarantees from our algorithm and the orienteering-based algorithm are incomparable in general.

Our algorithm uses as a subroutine, the *unrooted* DVRP (Section 1.2), and the 3-approximation algorithm for this problem from Arkin et al. [3]. The basic idea of the algorithm for DVRP is the following: if an r-tour visits some vertices a "large" distance from the root, it resembles an unrooted path (with smaller length) when restricted to just those vertices. So we partition the vertices of the graph into $O(\log D)$ parts, according to their distance from the root, and solve the unrooted DVRP (with appropriate distance bounds) in each part. Algorithm *minVR* for DVRP on general metrics is described below. Below, the slack between the distance constraint and the farthest vertex from the depot is denoted $\delta = \frac{D}{2} - \Delta + 1$; note that $\delta \geq 1$.

1. Define vertex sets $V_0, V_1, \cdots, V_t$ as follows (where $t = \lceil \log_2(D/2\delta) \rceil$):

$$V_j = \begin{cases} \{v : \frac{D}{2} - \delta < d(r,v) \leq \Delta\} & \text{if } j = 0 \\ \{v : \frac{D}{2} - 2^j \cdot \delta < d(r,v) \leq \frac{D}{2} - 2^{j-1} \cdot \delta\} & \text{if } 1 \leq j \leq t-1 \\ \{v : 0 < d(r,v) \leq \frac{D}{2} - 2^{t-1} \cdot \delta\} & \text{if } j = t \end{cases}$$

2. For $j = 0, \cdots, t$ do:

(a) Run the algorithm for unrooted DVRP [3], for the vertex set V_j, with distance constraint $2^j \cdot \delta - 1$. Let Π_j denote the set of paths obtained.

(b) For every path in Π_j, append both its end points with edges from the depot r, to obtain the r-tours $\{r \cdot \pi \cdot r \mid \pi \in \Pi_j\}$.

Theorem 3 *Algorithm minVR obtains a* $6(\lceil \log_2 \frac{D}{D-2\Delta+2} \rceil + 1)$*-approximation to DVRP.*

Proof: We first show that all the r-tours Π produced by algorithm minVR satisfy the distance constraint. For $j = 0$, each r-tour added in step 2 consists of two direct edges from r to V_0 and a path of length at most $\delta - 1$; so such a tour has length at most $2 \cdot \Delta + \delta - 1 = \Delta + \frac{D}{2} \leq D$. Now consider the r-tours corresponding to vertex sets V_j $(1 \leq j \leq t)$. Each path $\pi \in \Pi_j$ has length at most $2^j \cdot \delta - 1$, and every vertex of V_j (and hence the end points of π) is at distance at most $\frac{D}{2} - 2^{j-1} \cdot \delta$ from r. So each r-tour $(r \cdot \pi \cdot r)$ added in this step has length at most $2^j \cdot \delta - 1 + 2(\frac{D}{2} - 2^{j-1} \cdot \delta) \leq D$. Thus algorithm *minVR* produces a feasible solution to the DVRP instance.

We now prove the performance guarantee of this algorithm. Below OPT denotes the optimal number of r-tours for the DVRP instance.

Claim 4 *For each* $j = 0, \cdots, t$*, the optimal value of the unrooted DVRP instance defined in step 2a is at most* $2 \cdot OPT$.

Proof: Fix any $j \in [0, t]$. Let Γ denote an optimal solution to the original DVRP instance. Consider any r-tour $\sigma \in \Gamma$, and let σ_j denote the *path* induced by σ on the vertices in V_j. The length of σ_j is at most $D - 2(\frac{D}{2} - 2^j \cdot \delta + 1) = 2^{j+1} \cdot \delta - 2$. This is because every point in V_j (hence the end points of σ_j) is located at distance at least $\frac{D}{2} - 2^j \cdot \delta + 1$ from r. So the path σ_j can be split into two (unrooted) paths, each of length at most $2^j \cdot \delta - 1$. Splitting each tour in Γ in this manner gives us a set Θ of at most $2|\Gamma| = 2 \cdot OPT$ unrooted paths over V_j, that together cover all vertices of V_j. So Θ is a feasible solution to the unrooted DVRP instance on V_j with length bound $2^j \cdot \delta - 1$. Thus we have the claim. ■

Using Claim 4 and the 3-approximation to unrooted DVRP [3], we get $|\Pi_j| \leq 6 \cdot OPT$, for all $j = 0, \cdots, t$. Thus the total number of r-tours in the solution is at most $6(t+1) \cdot OPT$, giving the Theorem. ■

As a consequence of this algorithm, we also obtain a *bi-criteria* approximation algorithm for DVRP: if we are allowed to violate the distance bound D by a small factor $\epsilon > 0$, then we can cover all the vertices using $O(\log \frac{1}{\epsilon}) \cdot OPT$ r-tours. Here OPT is the minimum number of r-tours of length at most D, required to cover all vertices.

Corollary 5 *For every $0 < \epsilon < 1$, there is an $(O(\log \frac{1}{\epsilon}), 1 + \epsilon)$ bi-criteria approximation algorithm for distance constrained vehicle routing.*

Proof: Given an $0 < \epsilon < 1$ that denotes the allowed violation of the length bound, the claimed bi-criteria guarantee is achieved by algorithm *minVR* when the parameter δ is set to ϵD. It is easy to see that Claim 4 holds for any value of δ, and since $t = \lceil \log_2(D/2\delta) \rceil = \lceil \log \frac{1}{2\epsilon} \rceil$, the algorithm produces at most $O(\log \frac{1}{\epsilon}) \cdot OPT$ r-tours. One can also verify (as in Theorem 3) that the lengths of r-tours resulting from vertex-sets $V_1, \cdots, V_t$ are at most the length bound D. For vertex-set V_0, any r-tour has length at most $2\Delta + \delta - 1 \leq D + \epsilon D$. Thus the r-tours in the solution satisfy the length bound within a $1 + \epsilon$ factor.■

We note that the above bicriteria approximation was obtained independently in the preliminary version of this paper [23] and in Khuller et al. [24].

4 Integrality gaps

4.1 DVRP on trees

In this section, we study the LP relaxation $\mathcal{LP}$ (defined in Section 1.2) of the set covering formulation of DVRP on trees, and show that its integrality gap is a constant. This is achieved by showing that the lower-bound of Lemma 2 holds for all fractional solutions to $\mathcal{LP}$; Lemma 2 proved this for integral solutions. The next Lemma 7 is essentially a strengthening of Lemma 2 to fractional solutions of $\mathcal{LP}$. We first prove the following claim that gives a profit distribution scheme used in Lemma 7.

Claim 6 *For any tree H with root s and length function w on edges, it is possible to distribute a total profit of 1 among the leaves of H such that the profit contained in any rooted (at s) subtree F of H is at most $\frac{w(F)}{w(H)}$.*

Proof: We assume, without loss of generality, that H is binary. Our distribution scheme is top-down. Initially the root s has profit 1. If p is the profit at a vertex v (obtained from its parent), it is distributed as follows:

- If v is a leaf, the profit p stays at v.
- If v is an internal vertex, it has at most two children u_1, u_2. For $j = 1, 2$, let c_j denote the length of the subtree at u_j plus the edge (v, u_j). Then the profit at u_j is $\frac{c_j}{c_1+c_2} \cdot p$.

It is clear that this distribution places profits only at leaves of H. For any vertex v, we denote the subtree of H rooted at v by H_v. We claim that for all vertices $v \in H$, the profit contained in any subtree F' of H_v rooted at v is at most $w(F')/w(H_v)$ times the total profit of H_v (setting $0/0$ to be 1). We prove this claim by induction on the depth of subtree H_v. This is trivially true when vertex v is a leaf. Consider an internal vertex v with p being the total profit in H_v, such that this claim is true at both its children u_1 and u_2. For $j = 1, 2$, let $e_j = w(v, u_j)$, $b_j = w(H_{u_j})$, and $c_j = e_j + b_j$ the length of the subtree at u_j plus its parent edge. Let F' be any tree rooted at v. First consider the case that v has degree 1 in F', say only edge (v, u_1) is present. In this case, the length of F' restricted to the subtree under u_1 is $w(F') - w(v, u_1)$; applying induction on u_1, the profit contained in F' is at most $\frac{w(F')-e_1}{b_1} \frac{c_1}{c_1+c_2} p \leq \frac{w(F')}{b_1+e_1} \frac{c_1}{c_1+c_2} p = \frac{w(F')}{w(H_v)} p$. Next consider the case that v has degree 2 in F', i.e. both edges (v, u_1) and (v, u_2) are present. For $j = 1, 2$, let a_j be the length of F' in the subtree H_{u_j}; it is clear that $a_j \leq b_j$. By induction on u_1 and u_2, the total profit in F' can be bounded as follows (again we set $0/0$ to be 1).

$$\begin{aligned} &\leq \frac{a_1}{b_1} \cdot \frac{c_1}{c_1+c_2} p + \frac{a_2}{b_2} \cdot \frac{c_2}{c_1+c_2} p &= \frac{p}{c_1+c_2} [a_1(1 + \frac{e_1}{b_1}) + a_2(1 + \frac{e_2}{b_2})] \\ &\leq \frac{p}{c_1+c_2} [a_1 + e_1 + a_2 + e_2] &= \frac{w(F')}{w(H_v)} p \end{aligned}$$

which proves the inductive step.■

Recall the definition of a heavy cluster from Section 2.

Lemma 7 *If $C_1, \cdots, C_k$ are k disjoint heavy clusters in the tree, then the optimal value of $\mathcal{LP}$ is at least $\frac{k}{10}$.*

Proof: The dual of $\mathcal{LP}$ is the following.

$$\begin{array}{ll}
\max \sum_{v \in V \setminus r} p_v & \\
s.t. & \\
\sum_{v \in \tau} p_v \leq 1 & \forall \tau : r\text{-tour of length at most } D \\
p_v \geq 0 & \forall v \in V \setminus r
\end{array}$$

We will construct an appropriate dual solution which has value at least $\frac{k}{10}$. Then the lemma would follow by weak duality. Recall the definitions of the internal and external parts of a heavy cluster from Lemma 2. The internal part (see Figure 1a) of each C_i is further divided into two parts: the edges that also lie on the external part of some other cluster constitute the *through part* of C_i (length denoted by t_i); and all other edges constitute the *local part* of C_i (length denoted by l_i). For any cluster C_i, traversing all the edges in its internal and external parts twice constitutes an r-tour covering it; and as C_i is a heavy cluster, its length $2(d(r, C_i) + t_i + l_i) > D$. We partition the k heavy clusters into two sets: $\mathcal{A}$ consisting of clusters C_i with $l_i \geq t_i$, and $\mathcal{B}$ consisting of clusters C_i with $l_i < t_i$. We consider the following two cases.

Case 1: $|\mathcal{A}| \geq k/5$. In this case, we assign dual-values to vertices of clusters in $\mathcal{A}$. The dual solution is constructed as follows: for each cluster $C_i \in \mathcal{A}$, consider the subtree $T[C_i]$ induced by it, contract all vertices in its through part to a root-vertex, and distribute a total profit of $1/2$ among the vertices in its local part using Claim 6 below. The dual value of this solution is $|\mathcal{A}|/2 \geq \frac{k}{10}$. To show that this dual solution is feasible, we argue that any r-tour with profit more than 1 has length more than D. In the following, an *r-tree* is a subtree of T containing the root r. We will show that any r-tree π with profit more than 1 has $d(\pi) > D/2$; this suffices since any r-tour on the tree T corresponds to traversing edges of some r-tree twice. For each cluster C_i, let α_i denote the fraction of the local part of C_i used in r-tree π. From Claim 6, we have $\frac{1}{2}\sum \alpha_i \geq$ (total profit of π) > 1. Let $C_m \in \mathcal{A}$ denote the cluster of minimum local length that is visited by π. Note that the path from r to C_m is disjoint from the local parts of all clusters that π visits; so we have $d(\pi) \geq d(r, C_m) + \sum \alpha_i \cdot l_i > d(r, C_m) + 2l_m \geq d(r, C_m) + l_m + d_m > D/2$.

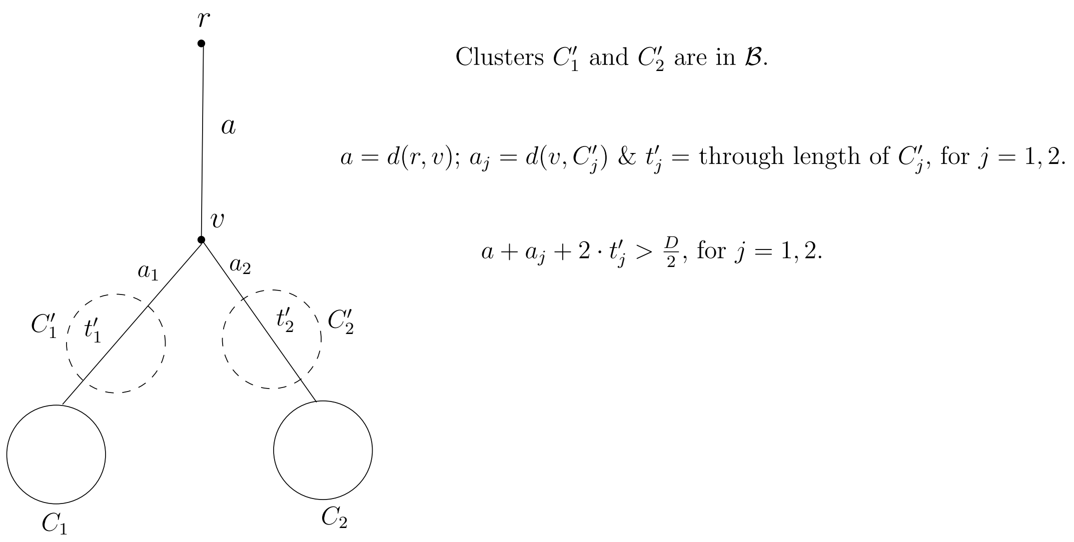

r
a
v
a_1
a_2
C'_1
t'_1
t'_2
C'_2
C_1
C_2
Clusters C'_1 and C'_2 are in $\mathcal{B}$.
$a = d(r,v)$; $a_j = d(v, C'_j)$ & t'_j = through length of C'_j, for $j = 1, 2$.
$a + a_j + 2 \cdot t'_j > \frac{D}{2}$, for $j = 1, 2$.

Figure 2: r-tree π in case 2.1

Case 2: $|\mathcal{A}| < k/5$. Let $\mathcal{L}$ denote all the leaf-clusters (one that has no cluster below it in tree T); clearly $\mathcal{L} \subseteq \mathcal{A}$ since leaf-clusters have through length 0. Define a tree F as follows: first contract each cluster in $\mathcal{B} \cup \mathcal{L}$ to form *cluster vertices* and let F' be the subtree of T induced by r and these cluster vertices (F' may contain other non-cluster vertices from T); then shortcut edges over all degree 2 non-cluster vertices in F' to get tree F. Note that every vertex (other than r) in F is either a cluster vertex or a non-cluster vertex that has degree at least 3. Also every leaf in F is a cluster vertex of some cluster in $\mathcal{L}$. A cluster $C \in \mathcal{B}$ is called *high-degree* if it has has degree at least 3 in F. A cluster $C \in \mathcal{B}$ is called *marked* if its parent in tree F is r or has degree at least 3. Observe that the number of high degree clusters is at most the number of leaves in F, which is at most $|\mathcal{A}|$; similarly the number of marked clusters is at most twice the number of leaves in F i.e. $2|\mathcal{A}|$. Let $\mathcal{C} = \{C_i \in \mathcal{B} \mid C_i \text{ is neither high-degree nor marked}\}$, clearly $|\mathcal{C}| \geq |\mathcal{B}| - 3|\mathcal{A}| > \frac{k}{5}$. Note that for every $C \in \mathcal{C}$, its parent in F has degree 2 and hence is a cluster vertex from $\mathcal{B}$. The dual solution here assigns a profit of 1/2 to each cluster in $\mathcal{C}$, distributing it over the local part of the cluster using Claim 6 (as in case 1). The dual value is $\frac{1}{2}|\mathcal{C}| \geq \frac{k}{10}$. To show that this is a feasible dual solution, as in case 1 we argue that any r-tree π with profit more than 1 has length $d(\pi) > D/2$. There are two cases to consider:

1. π visits some two clusters $C_1, C_2 \in \mathcal{C}$ that are incomparable in tree F (one is not a descendant of the other). Let $C_1', C_2' \in \mathcal{B}$ be the parents (in F) of C_1 and C_2 respectively. From the definition of $\mathcal{C}$, it follows that the cluster vertices C_1' and C_2' have degree 2 in F: combined with the fact that C_1 and C_2 are incomparable, π contains a subtree as shown in Figure 2. We have $D < 2(d(r, C_j') + t_j' + l_j') = 2(a + a_j + t_j' + l_j') \leq 2(a + a_j + 2t_j')$ for $j = 1, 2$. So $d(\pi) \geq a + a_1 + t_1' + a_2 + t_2' \geq a + 2\min\{a_1 + t_1', a_2 + t_2'\} > D/2$.

2. π does not visit any pair of incomparable clusters in F. Let $C' \in \mathcal{C}$ denote the lowest cluster according to the ancestor-descendant relation in F, that is visited by π. Note that every cluster in $\mathcal{C}$ has degree 2 in F; together with the fact that all other clusters visited by π are ancestors of C', we obtain that π contains the through part of all clusters it visits other than C'. Ignoring profit from the cluster C', π gets a profit of at least 1/2 from clusters whose through parts are completely contained in it. As in case 1, if α_i denotes the fraction of the local part of C_i used in π, we have $\frac{1}{2}\sum \alpha_i \geq \frac{1}{2}$. So $\sum \alpha_i \geq 1$, where the summation is only over clusters whose through parts are completely contained in π. Considering the cluster with the smallest local length, we have $d(\pi) > D/2$.

In both cases above, we have a feasible dual solution of value $\frac{k}{10}$.■

Algorithm *minTVR* finds k^* disjoint heavy clusters such that there is an integral solution of value at most $2k^* + 1$. Using Lemma 7 on these k^* clusters, we obtain that the optimal value of $\mathcal{LP}$ is at least $\frac{k^*}{10}$, so its integrality gap is most 20. The worst example we are aware of has an integrality gap of 2. Thus we have the following theorem.

Theorem 8 *The integrality gap of $\mathcal{LP}$ on trees is* $\Theta(1)$.

4.2 Unrooted DVRP

In the unrooted version of DVRP, the goal is to cover all vertices in the metric using the minimum number of paths, each of length at most the bound D. As mentioned earlier, this problem was considered in Arkin et al. [3] where a 3-approximation was given. In this section, we show that the integrality gap of $\mathcal{LP}$ for unrooted instances is a constant. This will also be used in Section 4.3 to bound the integrality gap of $\mathcal{LP}$ on general metrics. We consider the following LP relaxation of the problem, which is a weakening of $\mathcal{LP}$ (restricted to unrooted DVRP).

$$
\begin{array}{lll}
 & \min \sum_{\tau} x_{\tau} & \\
 & s.t. & \\
(LP') & \sum_{\tau: v \in \tau} x_{\tau} \geq 1 & \forall v \in V \\
 & x_{\tau} \geq 0 & \forall \tau : \text{tree of length at most } D
\end{array}
$$

Note that the variables in (LP') correspond to unrooted *trees* and not just paths. However, this is only a weaker relaxation that $\mathcal{LP}$, so the integrality gap of $\mathcal{LP}$ is at most that of LP'. We give an algorithm that rounds any fractional solution x to (LP') to an integral set of *paths*, that has a small cardinality. We assume (without loss of generality) that all trees in the support of the fractional solution (i.e. any tree τ with $x_{\tau} > 0$) are maximal: i.e., addition of any vertex to the tree makes its length more than D. We also assume that any tree in the support is a minimum spanning tree on the set of vertices that it covers. We construct a weighted graph G, with each edge e having weight $y_e = \sum_{\tau \ni e} x_{\tau}$, the sum of x-values of all trees containing e. Clearly $\sum_e y_e \cdot d_e \leq D \cdot \sum_{\tau} x_{\tau}$. For disjoint subsets of vertices $A, B \subseteq V$, $y(A, B)$ will denote the total y-weight of all edges having one end point in A and the other in B. For any subset $S \subseteq V$, $G|_S$ will denote the graph obtained from G by contracting the vertices of S into a single supernode. The rounding algorithm is as follows.

1. Initialize $S = \phi$.
2. While $G|_S$ has a cut of value less than 1, do:
   (a) Let $S' \subset V \setminus S$ be a minimal set such that $y(S', V \setminus S') < 1$.
   (b) Set $S = S \cup S'$.
3. Let $S_1, \cdots, S_t$ denote the vertex sets S' added in step 2b, over all iterations.
4. Compute an MST M on the contracted graph $H = G|_S$, and uncontract the supernode S in tree M to get a forest M' on G.
5. For each $i = 1, \cdots, t$, compute an MST M_i on S_i.
6. Construct the forest $F = M' \cup (\cup_{i=1}^{t} M_i)$ on G.
7. For each tree in F, take an Euler tour and split it into a set of maximal paths of length $\leq D$.

Note that the sets S' in step 2a can be easily obtained from the Gomory-Hu tree representing minimum cuts of $G|_S$. At the end of the while loop, $S = \cup_{i=1}^{t} S_i$ and $H = G|_S$ is the final graph with a supernode s consisting of all vertices in S. By the termination condition, $y(C, V \setminus C) \geq 1$ for all $C \subseteq V \setminus S$. So y induces a feasible fractional solution to the cut-based linear program for minimum spanning trees on H. Since the integrality gap of this LP relaxation is 2, the MST M in H has length at most $2 \cdot \sum_{e \in H} y_e \cdot d_e$. Note that none of the edges induced on S are included in this summation (in particular edges in any set S_i).

For each $i = 1, \cdots, t$, we have $y(S_i, V \setminus S_i) < 1$. By the minimality of S_i, for any $\phi \subsetneq C \subsetneq S_i$, $y(C, V \setminus C) \geq 1$ and $y((S_i \setminus C), V \setminus (S_i \setminus C)) \geq 1$. It is easy to verify that this implies $y(C, S_i \setminus C) \geq 1/2$ for all $\phi \subsetneq C \subsetneq S_i$. Thus $2y$ induces a feasible solution to the cut-based LP for minimum spanning trees on S_i. So the MST M_i in S_i has length at most $4 \cdot \sum_{e \in S_i} y_e \cdot d_e$ (for each $i = 1, \cdots, t$).

Now consider uncontracting the supernode s in tree M, and adding the trees $M_1, \cdots, M_t$, to get forest F. Clearly F contains exactly t trees, say $T_1, \cdots, T_t$ (with $T_i \supseteq M_i$ for all i). The preceding arguments imply that the length of forest F is $\sum_i d(T_i) \leq 2 \sum_{e \in H} y_e \cdot d_e + 4 \sum_{i=1}^{t} \sum_{e \in S_i} y_e \cdot d_e \leq 4 \cdot \sum_e y_e \cdot d_e$. In step 6 of the rounding algorithm, each tree T_i is split into a set of paths having length $\leq D$. The length of an Euler tour on T_i is at most $2 \cdot d(T_i)$; so the number of paths used to cover T_i is at most $\frac{2 \cdot d(T_i)}{D} + 1$. Thus the total number of (integral) paths generated by this rounding algorithm is $l \leq \frac{2 \sum_i d(T_i)}{D} + t \leq 8 \frac{\sum_e y_e \cdot d_e}{D} + t \leq 8 \sum_T x_T + t$.

In order to bound t, we partition the sets $S_1, \cdots, S_t$ into two groups: group 1 has all sets S_i such that the minimum edge length between S_i and $V \setminus S_i$ is at most $D/3$, and group 2 has all the remaining sets. Let t_1 and t_2 denote the number of sets in groups 1 and 2 respectively.

Suppose (for contradiction) that for some set S_i in group 1, the length of MST M_i is less than $2D/3$. Then due to maximality of trees in the support of x, none of them can be completely contained in S_i: the $D/3$ length edge leaving S_i can be added to the MST in S_i while still satisfying the distance constraint. So all the trees containing any vertex in S_i cross the cut $(S_i, V \setminus S_i)$. Since the total weight of trees containing any particular vertex is at least 1, $y(S_i, V \setminus S_i) \geq 1$, contradicting the choice of S_i. So for every S_i in group 1, MST M_i has length at least $2D/3$, and $d(T_i) \geq d(M_i) \geq 2D/3$. So $t_1 \leq \frac{3 \sum_i d(T_i)}{2D} \leq 6 \sum_T x_T$.

Now consider the sets in group 2; pick one vertex from each of these sets, say $v_1, \cdots, v_{t_2}$. Note that the pairwise distance between any two of these vertices is greater than $D/3$. So any D length tree may visit at most 3 of these vertices; i.e. any tree in the support contributes to covering at most 3 of the v_is. Since each of the vertices v_is is covered to an extent 1, we have the total coverage $t_2 \leq 3\sum_T x_T$. Using these expressions for t_1 and t_2, we get $l \leq 8\sum_T x_T + 6\sum_T x_T + 3\sum_T x_T = 17\sum_T x_T$. i.e., the integrality gap for unrooted DVRP is at most 17.

Theorem 9 *The integrality gap of $\mathcal{LP}$ for unrooted DVRP is* $\Theta(1)$.

4.3 General metrics

We now observe that algorithm *minVR* can also be used to bound the integrality gap of $\mathcal{LP}$ on general metrics. Let x be any feasible fractional solution to $\mathcal{LP}$, for an instance $\mathcal{I}$ of DVRP. We consider groups of vertices $\{V_j\}_{j=0}^{t}$ according to their distance from the root r (recall the definition from algorithm *minVR*). For each group V_j, we use x to obtain a feasible solution $y^{(j)}$ to the LP relaxation LP' of the unrooted DVRP instance on V_j (where the length bound is $2^j \cdot \delta - 1$). Solution $y^{(j)}$ is as follows: for each r-tour π in the support of x (i.e. $x_\pi > 0$), let σ' and σ'' be the unrooted paths on V_j derived from π as in Claim 4, and set $y^{(j)}_{\sigma'} = y^{(j)}_{\sigma''} = x_\pi$. Note that for each group V_j, the LP-value $\sum_\sigma y^{(j)}_\sigma = 2\sum_\pi x_\pi$. For each solution $y^{(j)}$, running the rounding algorithm from Section 4.2, we obtain a set of at most $34\sum_\pi x_\pi$ unrooted paths (each satisfying the length bound) covering V_j. Finally we append each of the unrooted paths (from all groups) with edges from r, as in step 2b to obtain an integral solution to $\mathcal{I}$ using at most $34t = O(\log_2 \frac{D}{D-2\Delta+2})$ r-tours. Combined with the $O(\log n)$ upper bound implied by a straightforward randomized rounding, we have the following.

Theorem 10 *The integrality gap of $\mathcal{LP}$ for general metrics is at most* $O(\min\{\log \frac{D}{D-2\Delta+2}, \log n\})$.

5 Extensions

In this section, we mention two natural extensions of DVRP, to which our algorithms apply and give similar performance guarantees.

5.1 Vehicle Routing with distance and capacity constraints

As defined, the distance constrained VRP does not consider vehicle capacities. However, it turns out that capacity constraints can be easily added to the formulation, and this incurs only a small increase in the approximation guarantee. An instance of the VRP with distance *and* capacity constraints consists of a metric space (V, d), a depot $r \in V$, a weight $q_u \in \mathbb{N}$ for each vertex $u \in V \setminus \{r\}$, a distance bound D, and a vehicle capacity Q. The objective is to find a *minimum cardinality* set of tours originating from r, that covers all the vertices $V \setminus \{r\}$, such that each tour has length at most D (the distance constraint), and serves a total demand weight at most Q (the capacity constraint). As in the capacitated VRP, there are two variants of this problem depending on whether or not demands can be split across vehicles: splittable or unsplittable demands.

Theorem 11 *An α-approximation algorithm for DVRP implies an $(\alpha + 2)$-approximation algorithm for the unsplittable demands version of VRP with distance and capacity constraints, and an $(\alpha + 1)$-approximation algorithm for the splittable demands version.*

Proof: Let $\mathcal{A}$ denote the α-approximation algorithm for DVRP, $\mathcal{I}$ any instance of the VRP with distance and capacity constraints, and $OPT(\mathcal{I})$ the optimal value of the splittable demands version of $\mathcal{I}$. Note that the optimal value of the unsplittable demands version of $\mathcal{I}$ is at least $OPT(\mathcal{I})$. Dropping the capacity constraints, we get a DVRP instance $\mathcal{J}$ corresponding to $\mathcal{I}$; note that $OPT(\mathcal{J}) \leq OPT(\mathcal{I})$. The approximate solution $\mathcal{A}(\mathcal{J})$ consists of $t \leq \alpha \cdot OPT(\mathcal{J})$ r-tours that satisfy the distance constraint (but not necessarily the capacity constraint). For every r-tour $\tau \in \mathcal{A}(\mathcal{J})$ that does not satisfy the capacity constraint, a simple bin-packing algorithm (see eg. Vazirani [7]) can be used to partition the demands served by τ into pieces, each having total weight at most Q. It can also be ensured that the number of additional r-tours introduced in satisfying the capacity constraint is at most $\frac{2}{Q}\sum_{u \in V-r} q_u$ (when demands are unsplittable), and $\frac{1}{Q}\sum_{u \in V-r} q_u$ (when demands are splittable). Note that the capacity constraints alone imply that $OPT(\mathcal{I}) \geq \frac{1}{Q}\sum_{u \in V-r} q_u$, even in the case of splittable demands. So the number of r-tours in the final feasible solution is at most $t + 2 \cdot OPT(\mathcal{I}) \leq (\alpha + 2)OPT(\mathcal{I})$ in the case of unsplittable demands, and $(\alpha + 1)OPT(\mathcal{I})$ in the case of splittable demands. Observing that the optimal value of the unsplittable version is at most that of the splittable version, we obtain the result for both versions.■

5.2 Distance Constrained Vehicle Routing using paths

In the definition of DVRP, the objective is to find bounded length tours as vehicle routes. There are also situations where the objective is to find a set of bounded length *paths* (each starting at the depot) as the vehicle routes. An example is the common deadline special case of the VRP with time windows: all the demand locations have a common deadline D, and each location has to be served by some vehicle from the depot before this deadline. In this case, it is not important for vehicles to return to the depot before the deadline D, but only to serve all demands by this deadline. Here the problem of minimizing the number of vehicles is the following: find a minimum cardinality set of paths that covers all the locations, such that each path originates at r and has length at most D. A path originating at the depot r is referred to as an r-path. We refer to this problem (vehicle routes being paths) as *path-DVRP*, and the original DVRP (vehicle routes being tours) as *tour-DVRP*. The next theorem shows that both these problems are closely related.

Theorem 12 *An α-approximation algorithm for tour-DVRP implies a 2α-approximation algorithm for path-DVRP. Conversely, an α-approximation algorithm for path-DVRP implies a 2α-approximation algorithm for tour-DVRP*

Proof: The argument for both directions is similar, so we only prove one direction. Let $\mathcal{A}$ be an α-approximation algorithm for tour-DVRP. Given any instance $\mathcal{I}$ of *path-DVRP* with distance constraint D, we consider a *tour-DVRP* instance $\mathcal{J}$ with distance constraint $2D$. Note that by doubling each r-path in any solution to $\mathcal{I}$, we obtain a feasible solution to $\mathcal{J}$ (with the same number of vehicles). So $OPT(\mathcal{J}) \leq OPT(\mathcal{I})$. Also, any $2D$ length r-tour can be split in the middle to obtain two D length r-paths that together cover the same set of vertices. Applying this procedure to each r-tour in the α-approximate solution $\mathcal{A}(\mathcal{J})$, we obtain a solution to $\mathcal{I}$ using at most $2\alpha \cdot OPT(\mathcal{J}) \leq 2\alpha \cdot OPT(\mathcal{I})$ vehicles.■

Remark: $OPT = 1$ **promise problem for path-DVRP.** We note that on instances of path-DVRP that are known to have an optimal solution using a single r-path, there is a simple polynomial-time algorithm that *finds* a solution using at most *two* r-paths. Note that even testing whether all vertices can be covered by one r-path is NP-complete (c.f. TSP). So unless P=NP, it is not possible to find in polynomial time, a single r-path that covers all the vertices, even if we know that there exists one such path. The following algorithm finds a solution covering all the vertices using at most two r-paths. Compute a minimum spanning tree on V, and obtain an Euler tour τ of this tree; note that the length of the MST is at most D (as there is a D length Hamilton path), and so $d(\tau) \leq 2D$. Now, as mentioned in the proof of Theorem 12, this r-tour τ can be split in the middle to obtain two r-paths (each of length at most D) that together cover all the vertices.

Acknowledgements: We thank Shuchi Chawla for helpful discussions.

References

[1] G. Laporte, M. Desrochers, and Y. Nobert, "Two Exact Algorithms for the Distance Constrained Vehicle Routing Problem," *Networks*, vol. 14, pp. 47–61, 1984.

[2] C.-L. Li, D. Simchi-Levi, and M. Desrochers, "On the distance constrained vehicle routing problem," *Operations Research*, vol. 40, pp. 790–799, 1992.

[3] E. M. Arkin, R. Hassin, and A. Levin, "Approximations for Minimum and Min-max Vehicle Routing Problems," *Journal of Algorithms*, 2005.

[4] A. Assad, "Modeling and Implementation Issues in Vehicle Routing," *Vehicle Routing: Methods and Studies*, pp. 7–45, 1988.

[5] P. Toth and D. Vigo, *The vehicle routing problem.* 2001.

[6] D. S. Hochbaum, ed., *Approximation algorithms for NP-hard problems.* 1997.

[7] V. V. Vazirani, *Approximation Algorithms* . 2001.

[8] M. Desrochers, J.Desrosiers, and M. Solomon, "A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows," *Operation Research*, vol. 40, pp. 342–354, 1992.

[9] N. Bansal, A. Blum, S. Chawla, and A. Meyerson, "Approximation Algorithms for Deadline-TSP and Vehicle Routing with Time Windows," *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pp. 166–174, 2004.

[10] B. Carr and S. Vempala, "Randomized meta-rounding," *Random Structures and Algorithms*, vol. 20(3), pp. 343–352, 2002.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* 1979.

[12] R. Jothi and B. Raghavachari, "Approximating the k-Traveling Repairman Problem with Repairtimes," *Journal of Discrete Algorithms*, vol. 5, no. 2, pp. 293–303, 2007.

[13] A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff, "Approximation Algorithms for Orienteering and Discounted-Reward TSP," *SIAM Journal on Computing*, vol. 37, no. 2, pp. 653–670, 2007.

[14] C. Chekuri, N. Korula, and M. Pal, "Improved Algorithms for Orienteering and Related Problems," in *SODA*, pp. 661–670, 2008.

[15] C. Bazgan, R. Hassin, and J. Monnot, "Approximation Algorithms for Some Vehicle Routing Problems," *Discrete Applied Mathematics*, vol. 146, pp. 27–42, 2005.

[16] M. Labbe, G. Laporte, and H. Mercure, "Capacitated Vehicle Routing on Trees," *Operations Research*, vol. 39(4), pp. 616–622, 1991.

[17] Y. Karuno, H. Nagamochi, and T. Ibaraki, "Vehicle scheduling on a tree with release and handling times ," *Annals of Operations Research*, vol. 69(1), pp. 193–207, 1997.

[18] S. Hamaguchi and N. Katoh, "A Capacitated Vehicle Routing Problem on a Tree," *Proceedings of the 9th International Symposium on Algorithms and Computation*, pp. 397–406, 1998.

[19] T. Asano, N. Katoh, and K. Kawashima, "A New Approximation Algorithm for the Capacitated Vehicle Routing Problem on a Tree," *Journal of Combinatorial Optimization*, vol. 5, no. 2, pp. 213 – 231, 2001.

[20] G. Frederickson and D. Guan, "Preemptive ensemble motion planning on a tree," *SIAM J. Comput.*, vol. 21, no. 6, pp. 1130–1152, 1992.

[21] I. Averbakh and O. Berman, "Sales-delivery man problems on treelike networks," *Networks*, vol. 25, pp. 45–58, 1995.

[22] A. Kohen, A. R. Kan, and H. Trienekens, "Vehicle Routing with Time Windows," *Operations Research*, vol. 36, pp. 266–273, 1987.

[23] V. Nagarajan and R. Ravi, "Minimum Vehicle Routing with a Common Deadline," *Proceedings of 9th Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pp. 212–223, 2006.

[24] S. Khuller, A. Malekian, and J. Mestre, "To Fill or not to Fill: The Gas Station Problem," *Proceedings of 15th Annual European Symposium on Algorithms*, pp. 534–545, 2007.