Minimum Makespan Multi-vehicle Dial-a-Ride

Gørtz, Inge Li; Nagarajan, Viswanath; Ravi, R.

Published in:
Algorithms - ESA 2009

Link to article, DOI:
10.1007/978-3-642-04128-0_48

Publication date:
2009

Document Version
Peer reviewed version

Minimum Makespan Multi-vehicle Dial-a-Ride

Inge Li Gørtz* Viswanath Nagarajan† R. Ravi‡

12 April 2009

Abstract

Dial-a-Ride problems consist of a set V of n vertices in a metric space (denoting travel time between vertices) and a set of m objects represented as source-destination pairs $\{(s_i, t_i)\}_{i=1}^m$, where each object requires to be moved from its source to destination vertex. In the *multi-vehicle Dial-a-Ride* problem, there are q vehicles each having capacity k and where each vehicle $j \in [q]$ has its own depot-vertex $r_j \in V$. A feasible schedule consists of a capacitated route for each vehicle (where vehicle j originates and ends at its depot r_j) that together move all objects from their sources to destinations. The objective is to find a feasible schedule that minimizes the maximum completion time (i.e. *makespan*) of vehicles, where the completion time of vehicle j is the time when it returns to its depot r_j at the end of its route. We study the *preemptive* version of multi-vehicle Dial-a-Ride, where an object may be left at intermediate vertices and transported by more than one vehicle, while being moved from source to destination. Our main results are an $O(\log^3 n)$-approximation algorithm for *preemptive multi-vehicle Dial-a-Ride*, and an improved $O(\log t)$-approximation for its special case when there is no capacity constraint (here $t \leq n$ is the number of distinct depot-vertices). There is an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation known even for single vehicle capacitated Dial-a-Ride [19]. For uncapacitated multi-vehicle Dial-a-Ride, we show that there are instances where natural lower bounds (used in our algorithm) are $\widetilde{\Omega}(\log t)$ factor away from the optimum.

We also consider the special class of metrics induced by graphs excluding any fixed minor (eg. planar metrics). In this case, we obtain improved guarantees of $O(\log^2 n)$ for capacitated multi-vehicle Dial-a-Ride, and $O(1)$ for the uncapacitated problem.

1 Introduction

The *multi-vehicle Dial-a-Ride* problem (mDaR) involves routing a set of m objects from their sources to destinations using a set of q vehicles starting at respective depot vertices. Each vehicle has a *capacity* k which is the maximum number of objects it can carry at any time. Two versions of Dial-a-Ride are studied, based on whether or not object routes are preemptive. In this paper we consider the less-examined *preemptive version*, where preemption (also known as transshipment) of objects can occur at any vertex in the metric. The standard objectives in Dial-a-Ride problems are total completion time and maximum completion time (makespan). The multi-vehicle total completion

*Technical University of Denmark. E-mail: ilg@imm.dtu.dk. Sponsored in part by a grant from the Carlsberg Foundation.

†IBM T.J. Watson Research Center, Yorktown Heights, NY 10598. E-mail: viswanath@us.ibm.com.

‡Tepper School of Business, Carnegie Mellon University. E-mail: ravi@cmu.edu. Supported in part by NSF grant CCF-0728841.

time problem admits a simple reduction to single vehicle Dial-a-Ride, as discussed below. Here, we study the *makespan* version of the problem, which is more interesting from an algorithmic point of view. Our main result is a poly-logarithmic approximation ratio for this problem.

While the multiple qualifications may make the problem appear contrived, this is exactly the problem that models courier or mail delivery over a day from several city depots: preemption is cheap and useful for packages, trucks are capacitated and the makespan reflects the daily working time limit for each truck. Despite its ubiquity, this problem has not been as well studied as other Dial-a-Ride versions. One reason from the empirical side is the difficulty in handling the possibility of preemptions in a reasonable mathematical programming model. On the theoretical side which is the focus of this paper, the difficulty of using preemption in a meaningful way persists; it is further compounded by the hardness of the makespan objective.

The preemptive Dial-a-Ride problem has been considered earlier with a single vehicle, for which an $O(\log n)$ approximation [11] and an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation [19] are known; here n is the number of vertices. Note that the completion time and makespan objectives coincide in the single vehicle case.

Moving to multiple vehicles, the total completion time objective admits a straightforward $O(\log n)$ approximation along the lines of the single vehicle problem [11]: using probabilistic tree embedding [17], one can reduce to tree-metrics at the loss of an expected $O(\log n)$ factor, and there is a simple constant factor approximation for this problem on trees. The makespan objective, which we consider in this paper turns out to be considerably harder. Due to non-linearity of the makespan objective, the above reduction to tree-metrics does not hold. Furthermore, the makespan problem does not appear significantly easier to solve even on trees.

The multi-vehicle Dial-a-Ride problem captures aspects of both machine scheduling and network design problems. Unlike in the single-vehicle case, an object in multi-vehicle Dial-a-Ride may be transported by several vehicles one after the other. Hence it is important for the vehicle routes to be coordinated so that the objects trace valid paths from respective sources to destinations. For example, a vehicle may have to wait at a vertex for other vehicles carrying common objects to arrive.

Due to such scheduling issues, the multi-vehicle Dial-a-Ride problem is non-trivial even in the absence of capacity constraints. In this paper, we first provide an approximation algorithm for the uncapacitated mDaR problem, which is interesting in itself. Then we generalize the result to the capacitated setting. We also show that improved approximation ratios can be achieved in both cases, for metrics that exclude some fixed minor (eg. planar metrics).

We note that although our model allows any number of preemptions and preemptions at all vertices, our algorithms do not use this possibility to its full extent. The algorithm for the capacitated case preempts each object at most once. The algorithm in the uncapacitated case preempts objects only at depot vertices and at most a logarithmic number of times.

1.1 Problem Definition and Preliminaries

We represent a finite metric as (V, d) where V is the set of vertices and d is a symmetric distance function satisfying the triangle inequality. For subsets $A, B \subseteq V$ we denote by $d(A, B)$ the minimum distance between a vertex in A and another in B, so $d(A, B) = \min\{d(u, v) \mid u \in A, v \in B\}$. For a subset $E \subseteq \binom{V}{2}$ of edges, $d(E) := \sum_{e \in E} d_e$ denotes the total length of edges in E.

The *multi-vehicle Dial-a-Ride problem* (mDaR) consists of an n-vertex metric (V, d), m objects specified as source-destination pairs $\{s_i, t_i\}_{i=1}^{m}$, q vehicles having respective depot-vertices $\{r_j\}_{j=1}^{q}$,

and a common vehicle capacity k. A feasible schedule is a set of q routes, one for each vehicle (where the route for vehicle $j \in [q]$ starts and ends at r_j), such that no vehicle carries more than k objects at any time and each object is moved from its source to destination. The completion time C_j of any vehicle $j \in [q]$ is the time when vehicle j returns to its depot r_j at the end of its route (the schedule is assumed to start at time 0). The objective in mDaR is to minimize the makespan, i.e., $\min \max_{j \in [q]} C_j$. We denote by $S := \{s_i \mid i \in [m]\}$ the set of sources, $T := \{t_i \mid i \in [m]\}$ the set of destinations, $R := \{r_j \mid j \in [q]\}$ the set of distinct depot-vertices, and $t := |R|$ the number of distinct depots. Throughout, we consider the *preemptive* version, where objects may be left at intermediate vertices and carried by multiple vehicles, while being moved from source to destination.

Lower bounds for single vehicle Dial-a-Ride The following are simple lower bounds for the single vehicle problem: the minimum length TSP tour on the depot and all source/destination vertices (*Steiner* lower bound), and $\frac{\sum_{i=1}^{m} d(s_i,t_i)}{k}$ (*flow* lower bound). [11] gave an $O(\log n)$-approximation algorithm for this problem based on the above lower bounds. A feasible solution to preemptive Dial-a-Ride is said to be *1-preemptive* if every object is preempted at most once while being moved from its source to destination, i.e. for each object $i \in [m]$ there is one intermediate vertex v_i such that it is first moved non-preemptively from source s_i to v_i, and later non-preemptively from v_i to destination t_i. [21] showed that the single vehicle preemptive Dial-a-Ride problem always has a 1-preemptive tour of length $O(\log^2 n)$ times the Steiner and flow lower-bounds.

Lower bounds for mDaR The quantity $\frac{\sum_{i=1}^{m} d(s_i,t_i)}{qk}$ is a lower bound similar to the flow bound for single vehicle Dial-a-Ride. Analogous to the Steiner lower bound above, is the optimal value of an induced *nurse-station-location* instance. In the nurse-station-location problem [16], we are given a metric (V, d), a set $\mathcal{T}$ of terminals and a multi-set $\{r_j\}_{j=1}^{q}$ of depot-vertices; the goal is to find a collection $\{F_j\}_{j=1}^{q}$ of trees that collectively contain all terminals $\mathcal{T}$ such that each tree F_j is rooted at vertex r_j and $\max_{j=1}^{q} d(F_j)$ is minimized. [16] gave a 4-approximation algorithm for this problem. The optimal value of the nurse-station-location instance with depots $\{r_j\}_{j=1}^{q}$ (depots of vehicles in mDaR) and terminals $\mathcal{T} = S \cup T$ is a lower bound for mDaR. Two natural lower bounds implied by this nurse-station-location instance are:

1. $\frac{1}{q}$ times the minimum length forest that connects every vertex in $S \cup T$ to some depot vertex.
2. $\max_{i \in [m]} d(R, s_i)$ and $\max_{i \in [m]} d(R, t_i)$.

Finally, it is easy to see that $\max_{i \in [m]} d(s_i, t_i)$ is also a lower bound for mDaR.

We note that the approximation bounds for uncapacitated mDaR are relative to the above simple lower bounds. However, these do not suffice for the capacitated setting (see Section 3 for an example). Hence our algorithm for capacitated mDaR relies on stronger lower bounds from subproblems that are obtained by restricting to suitable subsets of depots and demands.

1.2 Results, Techniques and Paper Outline

Uncapacitated mDaR. We first consider the special case of multi-vehicle Dial-a-Ride where the vehicles have no capacity constraints (i.e. $k \geq m$). This problem is interesting in itself, and serves as a good starting point before we present the algorithm for the general case. The uncapacitated

mDaR problem itself highlights differences from the single vehicle case: for example, in single vehicle Dial-a-Ride, preemption plays no role in the absence of capacity constraints; whereas in the multi-vehicle case, an optimal non-preemptive schedule may take $\Omega(\sqrt{q})$ longer than the optimal preemptive schedule (see Section 2). Our first main result is the following:

Theorem 1.1. *There is an $O(\log t)$-approximation algorithm for uncapacitated preemptive* mDaR. *Additionally, the schedule preempts each object only at depot vertices, and at most $2\log_2 t$ times.*

The above algorithm has two main steps: the first one (in Subsection 2.1) reduces the instance, at a constant factor loss in the performance guarantee, to one in which all demands are between depots (a "depot-demand" instance). In the second step (Subsection 2.2), we use a *sparse spanner* [31] on the demand graph to construct routes for moving objects across depots.

We also construct instances of uncapacitated mDaR where the optimal value is $\Omega(\log t/\log\log t)$ times all our lower bounds for this problem (Subsection 2.3). This suggests that stronger lower bounds are needed to obtain a better approximation ratio than what our approach provides.

We then consider the special class of metrics induced by graphs excluding some fixed minor (such as planar or bounded-genus graphs), and obtain the following improved guarantee in Subsection 2.4.

Theorem 1.2. *There is an $O(1)$-approximation algorithm for uncapacitated preemptive* mDaR *on metrics induced by graphs that exclude any fixed minor. The resulting schedule preempts each object only at depot vertices, and at most thrice.*

The algorithm in Theorem 1.2 has the same high-level approach outlined for Theorem 1.1: the difference is in the second step, where we use a stronger notion of *sparse covers* in such metrics (which follows from the "KPR decomposition" theorem [26]), to construct routes for moving objects across depots.

Capacitated mDaR. In Section 3, we study the capacitated multi-vehicle Dial-a-Ride problem, and obtain our second main result. Recall that there is an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation for even single vehicle Dial-a-Ride [19].

Theorem 1.3. *There is an $O(\log^3 n)$-approximation algorithm for preemptive* mDaR. *Additionally, the schedule preempts each object at most once.*

This algorithm is considerably more complex than the one for the uncapacitated special case, and is the main technical contribution of this paper. It has four key steps:

1. We *preprocess* the input so that demand points that are sufficiently far away from each other can be essentially decomposed into separate instances for the algorithm to handle independently.

2. We then solve a single-vehicle instance of the problem that obeys some additional *bounded-delay* property that we prove (Theorem 3.4); This property combines ideas from algorithms for *light approximate shortest path trees* [5, 25] and *capacitated vehicle routing* [22]. The bounded-delay property is useful in *partitioning* the single vehicle solution among the q vehicles available to share this load This partitioning scheme serves to average out the effect of the cutting in the objective function. This is similar to the α-point rounding method in scheduling [23], and has also been used in network design, eg. [24], [6] and [26].

3. The partitioned segments of the single vehicle tour are assigned to the available vehicles. However, to check if this assignment is feasible, we solve a matching problem that identifies cases when this load assignment must be *rebalanced*. This is perhaps the most interesting step in the algorithm since it identifies stronger lower bounds for subproblems where the current load assignment is not balanced.

4. We finish up by *recursing* on the load rebalanced subproblem. An interesting feature of the recursion is that the fraction of demands that are processed recursively is not a fixed value (as is more common in such recursive algorithms) but a function of the number of vehicles on which these demands are served.

We prove the new bounded-delay property of single-vehicle Dial-a-Ride in Subsection 3.1. We present the algorithm for Theorem 1.3 in Subsection 3.2 and establish an $O(\log m \log n \log mn)$-approximation bound. In Subsection 3.3 we show how to remove the dependence on m (number of objects) to obtain the final $O(\log^3 n)$ approximation ratio. The main idea here is an $O(1)$-approximation algorithm for the *multi-vehicle stacker crane* problem (Theorem 3.9), which is the special case of mDaR when capacity $k = 1$. We note that [18] gave a constant-factor approximation for multi-vehicle stacker crane, when all depots are identical. To the best of our knowledge, ours is the first constant-factor approximation for minimizing makespan in the stacker crane problem with multiple distinct vehicles/depots.

When the underlying metric is induced by graphs excluding a fixed minor, we can establish a stronger bounded-delay property in step (2) of the above framework. The main idea here is the construction of *well-separated covers* in such metrics, which we show can be obtained using the KPR decomposition [26]. This leads to the following improved guarantee, proved in Subsection 3.4.

Theorem 1.4. *There is an $O(\log^2 n)$-approximation algorithm for preemptive* mDaR *on metrics induced by graphs excluding any fixed minor. Additionally, the schedule preempts each object at most once.*

1.3 Related Work

Dial-a-Ride problems form an interesting subclass of Vehicle Routing Problems that are well studied in the operations research literature. [15] provided a classification of Dial-a-Ride problems using a notation similar to that for scheduling and queuing problems: preemption is one aspect in this classification. [32] and [14] surveyed several variants of non-preemptive Dial-a-Ride problems that have been studied in the literature. Most Dial-a-Ride problems arising in practice involve making routing decisions for multiple vehicles.

Dial-a-Ride problems with transshipment (the preemptive version) have been studied in [28, 29, 30]. These papers considered a more general model where preemption is allowed only at a specified subset of vertices. Our model and that of [11] is the special case when every vertex can serve as a preemption point; these results are nevertheless useful under restricted preemptions when the set P of preemption-points forms a "net" of the underlying metric (i.e. every vertex in V has a nearby point in P). It is clear that preemption only reduces the cost of serving demands: [30] studied the maximum decrease in the optimal cost upon introducing one preemption point. [28, 29] also modeled time-windows on the demands, and obtained heuristics and a column-generation based exact approach; they also described applications (eg. courier service) that allow for preemptions.

For single vehicle Dial-a-Ride, the best known approximation guarantee for the preemptive version is $O(\log n)$ [11], and an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation (for any constant $\epsilon > 0$) is shown in [19]. The non-preemptive version appears much harder and the best known approximation ratio is $\min\{\sqrt{k}\log n, \sqrt{n}\log^2 n\}$ [11, 21]); however, to the best of our knowledge, APX-hardness is the best lower bound. There are known instances of single vehicle Dial-a-Ride where the ratio between optimal non-preemptive and preemptive tours is $\Omega(\sqrt{n})$ in general metrics [11], and $\tilde{\Omega}(n^{1/8})$ in the Euclidean plane [21]. A 1.8-approximation is known for the $k = 1$ special case of single vehicle Dial-a-Ride (the *stacker-crane* problem) [18].

[10] studied the related *k-delivery TSP* problem, which involves transporting a number of *identical objects* from supply to demand vertices, using a single capacity k vehicle. The key difference from Dial-a-Ride is that an object can be moved from any supply vertex to any demand vertex. [10] gave an approximation algorithm for k-delivery TSP that outputs a non-preemptive tour of length at most five times an optimal preemptive tour. They also showed that for any k-delivery TSP instance, the optimal non-preemptive tour is at most four times the optimal preemptive tour.

The *truck and trailer routing problem* [8, 33] is another problem where preemption plays a crucial role. Here, a number of capacitated trucks and trailers are used to deliver all objects. Some customers are only accessible without the trailer. The trailers can be parked at any point accessible with a trailer and it is possible to shift demand loads between the truck and the trailer at the parking places. The papers [8, 33] gave some heuristics for this problem.

Single vehicle preemptive Dial-a-Ride is closely related to the uniform *buy-at-bulk* problem [4, 20], under the cost function $\lceil \frac{x}{k} \rceil$ where k is the vehicle capacity. Such a connection was formally used in [19] to establish the hardness of approximation for the single vehicle problem. Approximation algorithms for several buy-at-bulk variants have been studied recently, eg. non-uniform buy-at-bulk [9, 12], buy-at-bulk with node-costs [12] and buy-at-bulk with protection [3]; poly-logarithmic approximation guarantees are known for all these problems. However the techniques required to solve the *multi-vehicle* Dial-a-Ride problem appear quite different from these buy-at-bulk results.

The uncapacitated mDaR problem generalizes the *nurse-station-location* problem, for which a 4-approximation algorithm was given in [16]. In fact we also use this algorithm as a subroutine for uncapacitated mDaR. Nurse-station-location is the special case of uncapacitated mDaR when each source-destination pair coincides on a single vertex. In this paper, we handle not only the case with arbitrary pairs (uncapacitated mDaR), but also the more general problem (capacitated mDaR) with finite capacity restriction.

2 Uncapacitated Multi-Vehicle Dial-a-Ride

In this section we study the uncapacitated special case of mDaR, where vehicles have no capacity constraints (i.e. capacity $k \geq m$). We give an algorithm that achieves an $O(\log t)$ approximation ratio for this problem (recall $t \leq n$ is the number of distinct depots).

As a warm-up for the multi-vehicle setting, we first present an example that demonstrates a large gap between preemptive and non-preemptive schedules for uncapacitated mDaR.

Example (Preemption gap in Uncapacitated mDaR) Consider an instance of uncapacitated mDaR where the metric is induced by an unweighted star with q leaves (where q is number of vehicles), all q vehicles have the center vertex as depot, and there is a demand between every ordered pair of leaf-vertices. A preemptive schedule having makespan 4 is as follows: each vehicle

$j \in [q]$ visits leaf j and brings all demands with source j to the root, then each vehicle j visits its corresponding leaf again, this time delivering all demands with destination j. On the other hand, in any non-preemptive schedule, one of the q vehicles completely serves at least $q-1$ demands since there are $q(q-1)$ demands in all. The minimum length of any tour containing the end points of q demands is $\Omega(\sqrt{q})$, which is also a lower bound on the optimal non-preemptive makespan. Thus there is an $\Omega(\sqrt{q})$ factor gap between optimal preemptive and non-preemptive tours. This is in contrast to the *uncapacitated single vehicle* case, where it is easy to see that the optimal preemptive and non-preemptive tours coincide.

The algorithm for uncapacitated mDaR proceeds in two stages. Given any instance, it is first reduced (at the loss of a constant factor) to a "depot-demand instance" where all demands are between depot vertices (Subsection 2.1). This reduction uses the nurse-station-location algorithm from [16]. Then the depot-demand instance is solved using an $O(\log t)$-approximation algorithm (Subsection 2.2); this is the main step in the algorithm, and is based on constructing a sparse-spanner on the demand graph. For metrics excluding a fixed minor, we show (in Subsection 2.4) that a constant-factor approximation can be achieved using a different construction based on the KPR decomposition [26].

2.1 Reduction to Depot-demand Instances

We define *depot-demand instances* as those instances of uncapacitated mDaR where all demands are between depot vertices. Given any instance $\mathcal{I}$ of uncapacitated mDaR, the algorithm UncapMulti (Figure 1) reduces $\mathcal{I}$ to a depot-demand instance $\mathcal{J}$. We now argue that the reduction in UncapMulti only loses a constant approximation factor. Let B denote the optimal makespan of instance $\mathcal{I}$. Since the optimal value of the nurse-station-location instance solved in the first step of UncapMulti is a lower bound for $\mathcal{I}$, we have $\max_{j=1}^{q} d(F_j) \leq 4B$.

Claim 2.1. *The optimal makespan for the depot-demand instance $\mathcal{J}$ is at most $17B$.*

Proof: Consider a feasible schedule for $\mathcal{J}$ involving three rounds:

1. Each vehicle traverses (by means of an Euler tour) its corresponding tree in $\{F_j\}_{j=1}^{q}$ and moves each object i from its source-depot (the source in instance $\mathcal{J}$) to s_i (source in original instance $\mathcal{I}$).

2. Each vehicle follows the optimal schedule for $\mathcal{I}$, which moves each object i from s_i to t_i.

3. Each vehicle traverses its corresponding tree in $\{F_j\}_{j=1}^{q}$ and moves each object i from t_i to its destination-depot (the destination in $\mathcal{J}$).

Clearly this is a feasible schedule for $\mathcal{J}$. From the observation on the nurse-station-location instance, the time taken in each of the first and third rounds is at most $8B$. Furthermore, the time taken in the second round is the optimal makespan of $\mathcal{I}$ which is B. This proves the claim. □

Assuming a feasible schedule for $\mathcal{J}$, it is clear that the schedule returned by UncapMulti is feasible for the original instance $\mathcal{I}$. The first and third rounds in $\mathcal{I}$'s schedule require at most $8B$ time each. Thus an approximation ratio ρ for depot-demand instances implies an approximation ratio of $17\rho+16$ for general instances. In the next subsection, we show an $O(\log t)$-approximation algorithm for depot-demand instances (here t is the number of depots), which would imply Theorem 1.1.

Input: instance $\mathcal{I}$ of uncapacitated mDaR.

- Solve the nurse-station-location instance with depots $\{r_j\}_{j=1}^{q}$ and all sources/destinations $S \cup T$ as terminals, using the 4-approximation algorithm [16]. Let $\{F_j\}_{j=1}^{q}$ be the resulting trees covering $S \cup T$ such that each tree F_j is rooted at depot r_j.
- Define a depot-demand instance $\mathcal{J}$ of uncapacitated mDaR on the same metric and set of vehicles, where the demands are $\{(r_j, r_l) \mid s_i \in F_j \ \& \ t_i \in F_l,\ 1 \leq i \leq m\}$. For any object $i \in [m]$ let the *source depot* be the depot r_j for which $s_i \in F_j$ and the *destination depot* be the depot r_l for which $t_i \in F_l$.
- Output the following schedule for $\mathcal{I}$:
  1. Each vehicle $j \in [q]$ traverses tree F_j by an Euler tour, picks up all objects from sources in F_j and brings them to their source-depot r_j.
  2. Vehicles implement a schedule for *depot-demand instance* $\mathcal{J}$, and all objects are moved from their source-depot to destination-depot (using the algorithm in Section 2.2).
  3. Each vehicle $j \in [q]$ traverses tree F_j by an Euler tour, picks up all objects having destination-depot r_j and brings them to their destinations in F_j.

Figure 1: Algorithm UncapMulti for uncapacitated mDaR.

2.2 Algorithm for Depot-demand Instances

Let $\mathcal{J}$ be any depot-demand instance: note that the instance defined in the second step of UncapMulti is of this form. Let $\widetilde{B}$ denote the optimal makespan of instance $\mathcal{J}$. Our algorithm works on the induced metric (R, d) of depot vertices, and only uses one vehicle at each depot in R.

Consider an undirected graph H consisting of vertex set R and edges corresponding to demands: there is an edge between vertices r and s iff there is an object going from either r to s or s to r. Note that the metric length of any edge in H is at most $\widetilde{B}$ the optimal makespan of $\mathcal{J}$. We further reduce $\mathcal{J}$ to the following instance $\mathcal{H}$ of uncapacitated mDaR: the underlying metric is shortest paths in graph H (on vertices R), with one vehicle at each R-vertex, and for every edge $(u, v) \in H$ there is a demand from u to v and one from v to u. Note that any schedule feasible for $\mathcal{H}$ is also feasible for $\mathcal{J}$: since the demands in $\mathcal{J}$ are a subset of that in $\mathcal{H}$, and the vehicles in $\mathcal{J}$ are a superset of that in $\mathcal{H}$. Moreover, any schedule for $\mathcal{H}$ with makespan β corresponds to one for $\mathcal{J}$ with makespan $\beta \cdot \widetilde{B}$: this uses the fact that each edge of H has length at most $\widetilde{B}$ in $\mathcal{J}$'s metric. The next lemma finds a low makespan schedule for $\mathcal{H}$, which in turn implies an $O(\log |R|)$ approximation for depot-demand instances.

Lemma 2.2. *There exists a poly-time computable schedule for $\mathcal{H}$ with makespan $O(\log t)$, where $t = |R|$.*

Proof: Let $\alpha = \lfloor \log_2 t \rfloor$. We use the well-known *sparse spanner* construction from [2] to obtain subgraph A of H as follows: consider edges of H in an arbitrary order, and add an edge $(u, v) \in H$ to A iff the shortest path between u and v using current edges of A is more than 2α. It is clear

from this construction that the girth of A (length of its shortest cycle) is at least $2\alpha+2$, and that for every edge $(u,v) \in H$, the shortest path between u and v in A is at most 2α.

We now assign each edge of A to one of its end-points such that each vertex is assigned at most two edges. Repeatedly pick any vertex v of degree at most two in A, assign its adjacent edges to v, and remove these edges and v from A. We claim that at the end of this procedure (when no vertex has degree at most 2), all edges of A would have been removed (i.e. assigned to some vertex). Suppose for a contradiction that this is not the case. Let $\tilde{A} \neq \emptyset$ be the remaining graph; note that $\tilde{A} \subseteq A$, so girth of $\tilde{A}$ is at least $2\alpha+2$. Every vertex in $\tilde{A}$ has degree at least 3, and there is at least one such vertex w. Consider performing a breadth-first search in $\tilde{A}$ from w. Since the girth of $\tilde{A}$ is at least $2\alpha+2$, the first $\alpha+2$ levels of the breadth-first search form a tree (where level one consists of the singleton $\{w\}$). Furthermore, every vertex has degree at least 3, so each vertex in the first $\alpha+1$ levels has at least two children (the root w has three). This implies that $\tilde{A}$ has at least $3 \cdot 2^{\alpha+1} - 2 > t$ vertices, which is a contradiction! For each vertex $v \in R$, let A_v denote the edges of A assigned to v by the above procedure; we argued that $\cup_{v \in R} A_v = A$, and $|A_v| \leq 2$ for all $v \in R$.

The schedule for $\mathcal{H}$ involves 2α rounds as follows. In each round, every vehicle $v \in R$ traverses the edges in A_v (in both directions) and returns to v. Since $|A_v| \leq 2$ for all vertices v, each round takes 4 units of time; so the makespan of this schedule is $8\alpha = O(\log t)$. The route followed by each object in this schedule is the shortest path from its source to destination in spanner A; note that the length of any such path is at most 2α. To see that this is indeed feasible, observe that every edge of A is traversed by some vehicle in each round. Hence in each round, every object traverses one edge along its shortest path (unless it is already at its destination). Thus after 2α rounds, all objects are at their destinations. □

Combined with the reduction to depot-demands (UncapMulti), it follows that the final schedule for the uncapacitated mDaR instance $\mathcal{I}$ has makespan $O(\log t) \cdot B$. Additionally, this schedule preempts each object at most $2\alpha + 1 = 2\lfloor \log_2 t \rfloor + 1$ times (and only at depots). This completes the proof of Theorem 1.1.

2.3 Tight Example for Uncapacitated mDaR Lower Bounds

We note that known lower bounds for uncapacitated mDaR are insufficient to obtain a sub-logarithmic approximation guarantee. The lower bounds we used in our algorithm are the following: $\max_{i \in [m]} d(s_i, t_i)$, and the optimal value of a nurse-station-location instance with depots $\{r_j\}_{j=1}^{q}$ and terminals $S \cup T$. We are not aware of any lower bounds stronger than these two bounds.

We show an instance $\mathcal{G}$ of uncapacitated mDaR where the optimal makespan is a factor $\Omega(\frac{\log t}{\log\log t})$ larger than both the above lower bounds. In fact, the instance we construct is a depot-demand instance that has the same special structure as instance $\mathcal{H}$ in Lemma 2.2. That is, the demand graph is same as the graph inducing distances. Take $G = (V, E)$ to be a t-vertex regular graph of degree $\theta(\log t)$ and girth $g = \theta(\log t / \log\log t)$; there exist such graphs, eg. [27]. Instance $\mathcal{G}$ is defined on a metric on vertices V with distances being shortest paths in graph G. For every edge $(u,v) \in E$ of graph G, there is an object with source u and destination v (the direction is arbitrary). There is one vehicle located at every vertex of V; so the number of vehicles $q = t$.

Observe that both our lower bounds are $O(1)$: the optimal value of the nurse-station-location instance is 0, and maximum source-destination distance is 1. However as we show below, the optimal makespan for this instance is at least $g - 1 = \Omega(\log t / \log\log t)$. Suppose (for contradiction) that there is a feasible schedule for $\mathcal{G}$ with makespan $M < g - 1$. A demand $(u,v) \in E$ is said to be

completely served by a vehicle j iff the route of vehicle j visits both vertices u and v. The number of distinct vertices visited by any single vehicle is at most $M < g$: so the number of demands that are completely served by a single vehicle is at most $M - 1$ (otherwise these demand edges would induce a cycle smaller than the girth g). Hence the number of demands that are completely served by some vehicle is at most $t \cdot g < |E|$. Let $(u, v) \in E$ be a demand that is *not* completely served by any vehicle, i.e. there is no vehicle that visits both u and v. Since we have a feasible schedule of makespan M, the path π followed by demand (u, v) from u to v (or vice versa) in the schedule has length at most M. The path π can not be the direct edge (u, v) since demand (u, v) is not completely served by any vehicle. So path π together with edge (u, v) is a cycle of length at most $M + 1 < g$ in graph G, contradicting girth of G.

2.4 Improved Algorithm for Metrics Excluding a Fixed Minor

We now give a constant-factor approximation algorithm for uncapacitated mDaR on metrics induced by K_r-minor free graphs, for any fixed r (Theorem 1.2). This improvement comes from using the existence of good 'sparse covers' in such metrics, as opposed to the spanner based construction in Lemma 2.2. This guarantee is again relative to the above mentioned lower bounds.

Consider an instance of uncapacitated mDaR on metric (V, d) that is given by shortest paths in a K_r-minor-free graph $G = (V, E)$, with edge lengths $w : E \to \mathbb{R}_+$. We start with some definitions. A *cluster* is any subset of vertices. For any $\gamma > 0$ and vertex $v \in V$, $N(v, \gamma) := \{u \in V \mid d(u, v) \leq \gamma\}$ denotes the set of vertices within distance γ from v. As observed in [7] and [1], the partitioning scheme of [26] implies the following result.

Theorem 2.3 ([26])**.** *Given K_r-minor free graph $G = (V, E, w)$ and value $\gamma > 0$, there is an algorithm that computes a set $Z = \{C_1, \cdots, C_l\}$ of clusters satisfying:*

1. *The diameter of each cluster is at most $O(r^2) \cdot \gamma$, i.e. $\max_{u,v \in C_i} d(u, v) \leq O(r^2) \cdot \gamma$ for all $i \in [l]$.*
2. *For every $v \in V$, there is some cluster $C_i \in Z$ such that $N(v, \gamma) \subseteq C_i$.*
3. *For every $v \in V$, the number of clusters in Z that contain v is at most $O(2^r)$.*

The set Z of clusters found above is called a *sparse cover* of G.

The reduction in Section 2.1 implies that it suffices to consider depot-demand instances, where all demands are between pairs of depots. An $O(1)$ approximation for depot-demand instances would imply an $O(1)$ approximation for general instances. Let $\mathcal{J}$ be any depot-demand instance on metric (V, d) that is induced by K_r-minor-free graph $G = (V, E, w)$, with a set $R \subseteq V$ of depot-vertices (each containing a vehicle), and where all demands $\{s_i, t_i\}_{i=1}^m$ are between vertices of R. The algorithm is described in Figure 2.

Note that $\gamma = \max_{i \in [m]} d(s_i, t_i)$ is a lower bound on the optimal makespan of $\mathcal{J}$. We claim that the makespan of the above schedule is at most $O(r^2\, 2^r) \cdot \gamma$. Observe that each depot is contained in at most $O(2^r)$ clusters, and the distance from any depot to the center of any cluster containing it is at most $O(r^2) \cdot \gamma$. Hence the time taken by each vehicle in either of the two Steps (5a)-(5b) is at most $O(r^2 2^r) \cdot \gamma$. Since r is a fixed constant, the final makespan is $O(1) \cdot \gamma$.

We now argue the feasibility of the above schedule. Step (4) is well-defined: for all $i \in [m]$, we have $s_i, t_i \in N(s_i, \gamma)$ and there is some $j \in [l]$ with $N(s_i, \gamma) \subseteq C_j$ and we can set $\pi(i) = j$. It is now easy to see that each object $i \in [m]$ traces the following route in the final schedule: $s_i \rightsquigarrow c_{\pi(i)} \rightsquigarrow t_i$.

Input: Depot-demand instance $\mathcal{J}$ on metric $G = (V, E, w)$, depot-vertices $R \subseteq V$, demands $\{s_i, t_i\}_{i=1}^{m}$.

1. Let $\gamma = \max_{i \in [m]} d(s_i, t_i)$ be the maximum source-destination distance.
2. Compute a sparse cover $Z = \{C_j\}_{j=1}^{l}$ given in Theorem 2.3 for parameter γ.
3. For each cluster $C_j \in Z$, choose an arbitrary vertex $c_j \in C_j$ as its *center*.
4. For each demand $i \in [m]$, let $\pi(i) \in [l]$ be such that $s_i, t_i \in C_{\pi(i)}$.
5. Output the following schedule for $\mathcal{J}$:
   (a) Each vehicle $r \in R$ visits the centers of all clusters containing r, and returns to r; it carries the objects $\{i \in [m] \mid s_i = r\}$ having source r, and drops each object i at center $c_{\pi(i)}$.
   (b) Each vehicle $r \in R$ again visits the centers of all clusters containing r; it brings the objects $\{i \in [m] \mid t_i = r\}$ having destination r: each object i is picked up from center $c_{\pi(i)}$.

Output: An $O(1)$-approximate minimum makespan schedule for $\mathcal{J}$.

Figure 2: Algorithm for uncapacitated mDaR on K_r-minor free graphs.

Combining this algorithm for depot-demand instances with the reduction in Subsection 2.1, we obtain an $O(r^2\, 2^r)$-approximation algorithm for uncapacitated mDaR on K_r-minor free metrics. When r is a fixed constant this is a constant approximation, and we obtain Theorem 1.2. Note that each object in the resulting schedule is preempted at most thrice, and only at depot-vertices.

3 Capacitated Multi-Vehicle Dial-a-Ride

In this section we consider the general capacitated mDaR problem and obtain an $O(\log^3 n)$-approximation algorithm. Our algorithm uses a structural result on single-vehicle Dial-a-Ride tours, which is proved in Subsection 3.1. The main algorithm for mDaR is then described in Section 3.2 where we prove a slightly weaker approximation bound of $O(\log^2 m \log n)$. In Subsection 3.3 we show how to remove the dependence on m to obtain an $O(\log^3 n)$-approximation algorithm even for "*weighted* mDaR" (Theorem 1.3). Finally, in Subsection 3.4, we establish a stronger single-vehicle structure theorem (than the one in Subsection 3.1) for metrics excluding any fixed minor; this immediately leads to an improved $O(\log^2 n)$-approximation ratio for mDaR on such metrics (Theorem 1.4).

Bad Example for Simple Lower Bounds The lower bounds that were used in the uncapacitated case are too weak for capacitated mDaR. Consider a line metric on vertices $[q] = \{1, 2, \ldots, q\}$ where the distance between any pair i, j of vertices is $d(i, j) = |i - j|$. There are q vehicles, one at each vertex. The capacity $k = 1$. There are q demands from vertex 1 to vertex 2; and one demand from vertex i to $i + 1$, for each $i = 2, \ldots, q - 1$. The lower bounds are as follows: flow lower bound of $\frac{2q-2}{q} \leq 2$, nurse station location bound of zero, and maximum source-destination distance of one. However, the optimal makespan is at least $\sqrt{q}$. Suppose, for a contradiction, that the optimum is $M < \sqrt{q}$. Then, only vehicles $\{1, 2, \ldots, M\}$ can even reach vertex 1. Since there are q demands at vertex 1 and the capacity is one, it must be visited at least q distinct times. So some vehicle must

visit vertex 1 at least q/M times, and the tour length of this vehicle is at least $q/M > \sqrt{q} > M$, which is a contradiction to the optimal makespan. Hence the known lower bounds for mDaR are an $\Omega(\sqrt{q})$ factor off the true makespan.

As this example shows, it is important to balance the load assigned to vehicles carefully, taking the metric into account. Indeed, the key step in our algorithm is a "load rebalancing" step, where we derive stronger lower bounds from induced sub-instances.

Remark We note that the approach from the uncapacitated case does not seem useful here. Firstly, one can not reduce general capacitated mDaR instances to depot-demand instances: it is easy to construct examples (even with $q = 1$ vehicle) where the optimal solution is much shorter than any routing that transports each object via a depot. Secondly, even depot-demand instances of capacitated mDaR do not seem easy to solve (as shown above, the known lower bounds are too weak here).

3.1 Single Vehicle Dial-a-Ride with Bounded Delay

Here we prove a structural property of *single vehicle* Dial-a-Ride that has an additional "bounded delay" condition. This result is an important ingredient in the algorithm for capacitated mDaR, which appears in the next subsection.

First, we consider the simpler setting of the *capacitated vehicle routing problem* [22]. Capacitated vehicle routing problem (CVRP) is a special case of single vehicle Dial-a-Ride when all objects have the same source. Formally, we are given a metric (V, d), specified depot-vertex $r \in V$, and m objects all having source r and respective destinations $\{t_i\}_{i\in[m]}$. The goal is to compute a minimum length *non-preemptive* tour of a capacity k vehicle originating at r, that moves all objects from r to their destinations. In *CVRP with bounded delay*, we are additionally given a *delay parameter* $\beta > 1$, and the goal is to find a minimum length capacitated non-preemptive tour serving all objects such that the time spent by each object $i \in [m]$ in the vehicle is at most $\beta \cdot d(r, t_i)$, i.e. β times the distance from its source (r) to destination (t_i). Two lower bounds for the CVRP are as follows: minimum length TSP tour on $\{r\} \cup \{t_i \mid i \in [m]\}$ (Steiner lower bound), and $\frac{2}{k}\sum_{i=1}^{m} d(r, t_i)$ (flow lower bound). These lower bounds also hold for the (less constrained) preemptive version of CVRP. We will use the following known algorithm for CVRP.

Theorem 3.1 ([22])**.** *Consider any CVRP instance on metric (V, d) with depot r, object destinations $T \subseteq V$ and capacity k. Given any tour σ on $\{r\} \cup T$, there is a polynomial-time computable CVRP solution of length at most $d(\sigma) + \frac{2}{k} \cdot \sum_{v\in T} d(r, v)$. Moreover, the time spent in the vehicle by each object $v \in T$ is at most the length in σ from r to v.*

The idea is to move objects in groups of k each, where each group consists of contiguous vertices in tour σ. Note that setting σ to be a $\frac{3}{2}$-approximate TSP tour [13] leads to a 2.5-approximation algorithm for CVRP, which is the best approximation ratio known for it.

Theorem 3.2. *There is a $(2.5 + \frac{3}{\beta-1})$-approximation algorithm for CVRP with bounded delay, where $\beta > 1$ is the delay parameter. This guarantee is relative to the maximum of the Steiner and flow lower bounds.*

Proof: Our algorithm is basically a combination of the algorithms for *light approximate shortest path trees* [25], and capacitated vehicle routing (Theorem 3.1). Let LB denote the maximum of the Steiner and flow lower bounds.

The first step is to compute an approximately minimum TSP tour C on the set $\{t_i\}_{i\in[m]}$ of all destinations and r. The algorithm from [13] gives a 1.5-approximation, so we have $d(C) \leq 1.5 \cdot \mathsf{LB}$. Number the vertices in the order in which they appear in C, starting with r being 0. For any pair of vertices $i < j$, let $d_C(i,j)$ denote the length of the path in C from i to j. Next, we apply the following procedure, which is similar to one in [25].

1. Set $v_0 \leftarrow 0$.
2. For $p \geq 1$ do:
   (a) Set $v_p \leftarrow \min\{v \geq v_{p-1} \,:\, \beta \cdot d(0,v) < d(0,v_{p-1}) + d_C(v_{p-1},v)\}$.
   (b) If $v_p = \mathsf{NIL}$ (i.e. there is no v as above) then terminate loop.

Let t denote the number of iterations in the above loop. So we have vertices $0 < v_1 < v_2 < \cdots < v_t < |V|$. We claim that the following properties hold:

P1. For any $1 \leq p \leq t$ and vertex $v_{p-1} \leq u < v_p$, $d(0,v_{p-1}) + d_C(v_{p-1},u) \leq \beta \cdot d(0,u)$.

P2. $\sum_{p=1}^{t} d(0,v_p) \leq \frac{1}{\beta-1} \cdot d(C)$.

Property (P1) is immediate by the definition of vertex v_p, which is the first vertex v after v_{p-1} where $d(0,v_{p-1}) + d_C(v_{p-1},v) > \beta \cdot d(0,v)$. To see property (P2), again by definition of v_p, we have

$$\beta \cdot d(0,v_p) \;<\; d(0,v_{p-1}) + d_C(v_{p-1},v_p), \quad \forall 1 \leq p \leq t.$$

Adding these inequalities,

$$\beta \cdot \sum_{p=1}^{t} d(0,v_p) \;<\; \sum_{p=1}^{t} d(0,v_{p-1}) + \sum_{p=1}^{t} d_C(v_{p-1},v_p) \;\leq\; \sum_{p=1}^{t} d(0,v_p) + d(C).$$

Rearranging this inequality gives (P2).

Now we split tour C at the vertices $0 < v_1 < v_2 < \cdots < v_t < |V|$ obtained above. For each $1 \leq p \leq t$, define sub-tour C_p which starts at r, goes to v_{p-1}, traverses C until v_p, then returns to r. Assign all objects with destinations in $\{v_{p-1}, \cdots, v_p - 1\}$ to C_p. Also define tour C_{t+1} which starts at r, goes to v_t, and traverses C until r; and assign all remaining objects to C_{t+1}. Using property (P2), the total length of these sub-tours is:

$$\sum_{p=1}^{t+1} d(C_p) \quad = \quad d(C) + 2\sum_{p=1}^{t} d(0,v_p) \quad \leq \quad (1 + \frac{2}{\beta-1})d(C). \tag{3.1}$$

By property (P1), it is clear that C_p (for each $1 \leq p \leq t+1$) visits each vertex u in it within time $\beta \cdot d(0,u)$. For each C_p, we serve the set D_p of objects assigned to it, using Theorem 3.1. Let γ_p denote the solution corresponding to C_p; note that its length $d(\gamma_p) \leq d(C_p) + \frac{2}{k} \cdot \sum_{z\in D_p} d(r,z)$. Furthermore, the time spent in the vehicle by each object $z \in D_p$ is at most the length from r to z in C_p, which (as noted above) is at most $\beta \cdot d(0,z)$.

The final solution γ is the concatenation of tours $\gamma_1, \cdots, \gamma_{t+1}$. Clearly, the time spent in the vehicle by any object $i \in [m]$ is at most $\beta \cdot d(r,t_i)$, which satisfies the bounded delay condition. The length of tour γ is

$$\sum_{p=1}^{t+1} d(\gamma_p) \;\leq\; \sum_{p=1}^{t+1} d(C_p) + \frac{2}{k} \cdot \sum_{p=1}^{t+1} \sum_{z\in D_p} d(r,z) \;\leq\; (1 + \frac{2}{\beta-1})d(C) + \frac{2}{k} \cdot \sum_{i=1}^{m} d(r,t_i).$$

The second inequality uses (3.1) and the fact that $\{D_p : 1 \le p \le t+1\}$ is a partition of the set of objects. Note that $\frac{2}{k} \cdot \sum_{i=1}^{m} d(r, t_i)$ is just the flow lower bound, so it is at most LB. Since $d(C) \le 1.5 \cdot \mathsf{LB}$, it follows that $d(\gamma) \le \left(2.5 + \frac{3}{\beta - 1}\right) \cdot \mathsf{LB}$. □

Now, we are ready to prove the bounded-delay property for the general Dial-a-Ride problem. This makes use of the previous Theorem 3.2. We will also use the following "tree embedding" result:

Theorem 3.3 ([17]). *Given any metric (V, d) and function $c : \binom{V}{2} \to \mathbb{R}_{\ge 0}$, there is a polynomial-time computable tree metric κ such that:*

- *κ is induced by a tree with $O(\log \frac{d_{max}}{d_{min}})$ levels, where d_{max} and d_{min} are the maximum and minimum positive distances in metric (V, d).*
- *For all $u, v \in V$, $\kappa(u, v) \ge d(u, v)$.*
- *$\sum_{u,v} c_{uv} \cdot \kappa(u, v) \le O(\log n) \cdot \sum_{u,v} c_{uv} \cdot d(u, v)$.*

The usual form of this tree embedding result involves a probability distribution over such tree metrics, where the expected length of each edge is only an $O(\log n)$ factor larger than its metric length. In order to obtain a deterministic algorithm for bounded-delay Dial-a-Ride, we use the above version of tree embedding, which guarantees an $O(\log n)$ factor bound (with probability one) for a single cost function on edges.

Theorem 3.4. *There is a polynomial time algorithm that, given any instance of single vehicle Dial-a-Ride on metric (V, d) with demand pairs $\{(s_i, t_i) : i \in D\}$ and capacity k, outputs a 1-preemptive tour τ satisfying the following conditions (here LB is the maximum of the Steiner and flow lower bounds, $n = |V|$ and $m \ge |D|$):*

1. *Total length: $d(\tau) \le O(\log n \cdot \log mn) \cdot \mathsf{LB}$.*
2. *Bounded delay: $\sum_{i \in D} T_i \le O(\log n) \sum_{i \in D} d(s_i, t_i)$ where T_i is the total time spent by object $i \in D$ in the vehicle under the schedule given by τ.*

Proof: Recall the two lower bounds for single vehicle Dial-a-Ride: minimum TSP on $\{s_i, t_i\}_{i \in D}$ (Steiner lower bound) and $\frac{1}{k} \sum_{i \in D} d(s_i, t_i)$ (flow lower bound). Let $n = |V|$ and $m \ge |D|$ be any value.

Using standard scaling arguments, we will first preprocess the instance to ensure $\frac{d_{max}}{d_{min}} = O(m^2 n^2)$, where d_{max} and d_{min} are the maximum and minimum positive distances in the metric. We restrict attention to the vertex set $U = \{s_i, t_i\}_{i \in D}$ of all sources/destinations. Let $M := \max\{d(u, v) : u, v \in U\}$ be the diameter of this set; so $d_{max} = M$. Note that the Steiner and flow lower bounds remain the same in the restricted metric (U, d), and $M \le \mathsf{LB}$. If $M > \sum_{i \in D} d(s_i, t_i)$ then we output the non-preemptive solution τ_0 that visits all sources $\{s_i : i \in D\}$ along a 1.5-approximate TSP tour [13], and when visiting any source s_i makes a direct roundtrip to/from its destination t_i. This solution τ_0 satisfies the two conditions:

1. $d(\tau_0)$ equals the length of the approximate TSP on the sources plus $2 \sum_{i \in D} d(s_i, t_i)$, which is at most $1.5 \cdot \mathsf{LB} + 2M \le 3.5 \cdot \mathsf{LB}$.
2. The time T_i spent by object i in the vehicle is $d(s_i, t_i)$, so $\sum_{i \in D} T_i \le \sum_{i \in D} d(s_i, t_i)$.

In the following we assume $M \leq \sum_{i \in D} d(s_i, t_i)$. Now we modify the metric by setting to zero the length of all edges smaller than $\frac{M}{4m^2n^2}$; the new distances d' are shortest paths under the modified lengths. So $d_{min} \geq \frac{M}{4m^2n^2}$ and $\frac{d_{max}}{d_{min}} = O(m^2n^2)$ as desired. Since any shortest path has at most n edges, any pairwise distance in d is at most $\frac{M}{4m^2n}$ more than that in d'. Let τ be a 1-preemptive tour in metric (U, d') satisfying the two conditions in the theorem. Then we claim that τ also satisfies these conditions in the original metric (V, d). Without loss of generality, any 1-preemptive tour has at most $4|D| \leq 4m$ edge traversals: each object is picked or dropped at most four times, and every vertex-visit must involve picking/dropping some object (otherwise one can obtain a shorter tour by shortcutting over such vertices). So:

1. $d(\tau) \leq d'(\tau) + 4m \cdot \frac{M}{4m^2n} \leq d'(\tau) + \frac{\mathsf{LB}}{mn} = O(\log n \log mn) \cdot \mathsf{LB}$.

2. The time T_i spent by any object i in the vehicle (under metric d) is at most $\frac{M}{mn}$ more than the time T'_i spent under metric d'. So $\sum_{i \in D} T_i \leq \sum_{i \in D} T'_i + m \cdot \frac{M}{mn} \leq \sum_{i \in D} T'_i + \frac{1}{n} \sum_{i \in D} d(s_i, t_i) = O(\log n) \cdot \sum_{i \in D} d(s_i, t_i)$.

Based on the above discussion, we assume that the input metric satisfies $\frac{d_{max}}{d_{min}} = O(m^2n^2)$. Let C denote a 1.5-approximate TSP tour [13] on all sources/destinations of the demands D; note that the length $d(C) \leq 1.5 \cdot \mathsf{LB}$. Define the following two cost functions on edges:

$$c^1_{uv} = \begin{cases} 1 & \text{if } (u,v) \in C \\ 0 & \text{otherwise} \end{cases}, \qquad \forall u, v \in V$$

$$c^2_{uv} = \begin{cases} 1 & \text{if } (u,v) = (s_i, t_i) \text{ for some } i \in D \\ 0 & \text{otherwise} \end{cases}, \qquad \forall u, v \in V$$

Let $A := \sum_{i \in D} d(s_i, t_i)$. Now define a combined cost function $\mathbf{c} := \frac{1}{d(C)} \cdot \mathbf{c^1} + \frac{1}{A} \cdot \mathbf{c^2}$. Observe that $\sum_{u,v} c_{uv} \cdot d(u, v) = 2$. We now use Theorem 3.3 on metric (V, d) and cost function c to obtain tree metric κ that is induced by tree $\mathcal{T}$ where:

- $\mathcal{T}$ has $l = O(\log \frac{d_{max}}{d_{min}}) = O(\log mn)$ levels.
- $\kappa(u, v) \geq d(u, v)$, $\forall u, v \in V$, and
- $\sum_{u,v} c_{uv} \cdot \kappa(u, v) \leq O(\log n) \cdot \sum_{u,v} c_{uv} \cdot d(u, v) = O(\log n)$.

The last property implies that $\sum_{e \in C} \kappa(e) = \sum_{u,v} c^1_{uv} \cdot \kappa(u, v) \leq O(\log n) \cdot d(C)$ and $\sum_{i \in D} \kappa(s_i, t_i) = \sum_{u,v} c^2_{uv} \cdot \kappa(u, v) \leq O(\log n) \cdot \sum_{i \in D} d(s_i, t_i)$. In other words, the Steiner and flow lower bounds under metric κ are only an $O(\log n)$ factor more than those under metric d. Let $\widetilde{\mathsf{LB}}$ denote the maximum of these two lower bounds in metric κ; then we have $\widetilde{\mathsf{LB}} = O(\log n) \cdot \mathsf{LB}$.

Given the bounded delay property (Theorem 3.2) for *capacitated vehicle routing* (CVRP), the rest of the proof is a simple extension of Theorem 4.1 from [21], which obtains a single vehicle 1-preemptive tour within an $O(\log^2 n)$ factor of the Steiner and flow lower bounds.

We partition the demands in D into l sets as follows. For each $p = 1, \cdots, l$, set D_p consists of all demands $i \in D$ having their *nearest common ancestor* (nca) in level p of tree $\mathcal{T}$ (i.e. the nca of s_i and t_i is some vertex in level p). We serve each D_p separately using a tour of length $O(\widetilde{\mathsf{LB}})$. Finally we concatenate the tours for each level p.

Serving D_p For each vertex v at level p in $\mathcal{T}$, let L_v denote the demands in D_p that have v as their nca. Let LB_v denote the maximum of the Steiner and flow lower-bounds (in metric κ) for the single vehicle Dial-a-Ride problem with demands L_v. Clearly the flow lower bounds are disjoint for different vertices v. Also, the subtrees under any two distinct level p vertices are disjoint, so the Steiner lower bounds are disjoint for different v. Thus $\sum_v \mathsf{LB}_v \leq \widetilde{\mathsf{LB}}$, where v ranges over all vertices in level p of $\mathcal{T}$. We now show how each L_v is served separately.

Serving L_v Consider the following two instances of the CVRP problem in metric κ: $\mathcal{I}_{src}$ with all sources in L_v and common destination v, and $\mathcal{I}_{dest}$ with common source v and all destinations in L_v. Clearly, the Steiner lower bound for both $\mathcal{I}_{src}$ and $\mathcal{I}_{dest}$ is at most LB_v. Moreover, the flow lower bound $\frac{2}{k}\sum_{i\in D} d(s_i, v)$ for $\mathcal{I}_{src}$ is at most $2\mathsf{LB}_v$, since $\kappa(s_i, t_i) = \kappa(s_i, v) + \kappa(v, t_i)$ (recall v is the nca of s_i and t_i) for all $i \in L_v$. Similarly, the flow lower bound for $\mathcal{I}_{dest}$ is at most $2\mathsf{LB}_v$.

Consider instance $\mathcal{I}_{src}$. Setting delay parameter $\beta = 2$ in Theorem 3.2, we obtain a *non-preemptive* tour σ_v that moves all objects in L_v from their sources to vertex v, such that (1) the length of σ_v is at most $11\,\mathsf{LB}_v$, and (2) the time spent by each object $i \in L_v$ in the vehicle is at most $2\,\kappa(s_i, v)$. Similarly for instance $\mathcal{I}_{dest}$, we obtain a *non-preemptive* tour τ_v that moves all objects in L_v from vertex v to their destinations, such that (1) the length of τ_v is at most $11\,\mathsf{LB}_v$, and (2) the time spent by each object $i \in L_v$ in the vehicle is at most $2\,\kappa(v, t_i)$. Concatenating these two tours, we obtain that $\sigma_v \cdot \tau_v$ is a *1-preemptive* tour serving L_v, of length at most $22 \cdot \mathsf{LB}_v$, where each object $i \in L_v$ spends at most $2(\kappa(s_i, v) + \kappa(v, t_i)) = 2 \cdot \kappa(s_i, t_i)$ time in the vehicle.

We now use a depth-first-search traversal (DFS) on $\mathcal{T}$ (restricted to the end-points of demands D_p) to visit all vertices v in level p that have some demand in their subtree (i.e. $L_v \neq \emptyset$), and use the algorithm described above for serving demands L_v when v is visited in the DFS. This is the tour serving D_p. Note that twice the length of the DFS is at most the Steiner lower bound on κ, so it is at most $\widetilde{\mathsf{LB}}$. Thus the tour serving D_p has length at most $\widetilde{\mathsf{LB}} + 22\sum_v \mathsf{LB}_v$, where v ranges over all vertices in level p of $\mathcal{T}$. Recall that $\sum_v \mathsf{LB}_v \leq \widetilde{\mathsf{LB}}$, so the tour serving D_p has length at most $23 \cdot \widetilde{\mathsf{LB}}$. Additionally, the time spent by each object $i \in D_p$ in the vehicle is at most $2\,\kappa(s_i, t_i)$.

Finally, concatenating the tours for each level $p = 1, \cdots, l$, we obtain a 1-preemptive tour on $\mathcal{T}$ of length $O(\log mn) \cdot \widetilde{\mathsf{LB}}$. Additionally, the time spent by each object i in the vehicle is at most $2 \cdot \kappa(s_i, t_i)$. This translates to a 1-preemptive tour on the original metric having length $O(\log n \cdot \log mn) \cdot \mathsf{LB}$. Moreover, $\sum_{i\in D} T_i \leq \sum_{i\in D} 2 \cdot \kappa(s_i, t_i) \leq O(\log n) \sum_{i\in D} d(s_i, t_i)$ where T_i is the time spent by object i in the vehicle. This completes the proof of Theorem 3.4. □

3.2 Algorithm for Capacitated mDaR

We are now ready to present our algorithm for capacitated multi-vehicle Dial-a-Ride. The algorithm assumes an upper bound B on the optimal makespan (by enumerating over polynomially many choices) and returns either a feasible schedule of makespan $O(\log n \log m \log mn) \cdot B$, or a certificate that the optimal makespan is greater than B.

For any parameter B, the algorithm relies on a partial coverage algorithm $\mathsf{Partial}\langle Q, D, B\rangle$ that given subsets $Q \subseteq [q]$ of vehicles and $D \subseteq [m]$ of demands, outputs one of the following:

- a schedule for Q of makespan $O(\log n \cdot \log mn) \cdot B$ covering a constant fraction of demands D, or
- a certificate that the optimal makespan is more than B.

The main steps in Partial are as follows. First, we *preprocess* the input so that demand points that are sufficiently far away from each other can be decomposed into separate instances for the algorithm to handle independently (step 1). We then obtain a *single-vehicle* tour from Theorem 3.4 satisfying the 1-preemptive and bounded-delay properties (step 2). Then we *partition* the single vehicle tour into $|Q|$ equally spaced pieces (step 3). The bounded-delay property is useful in partitioning the single vehicle solution among the q vehicles available to share this load. Using the bounded-delay property we show that the number of objects that are carried in the vehicle over some cut edge is $O(\frac{1}{\log m}) \cdot |D|$. These objects will not be served by the schedule. Using a matching sub-problem, we assign *some* of these pieces to vehicles of Q, so as to satisfy a subset of demands in D (step 4 and 5). The matching problem identifies cases when the load assignment must be *rebalanced*. A piece can be matched with a vehicle if the distance between them is at most $2B$, and each vehicle can be assigned two pieces. All demands with both source and destination in pieces that got assigned to some vehicle will then be served (step 7 (a) and (b)). Finally, a suitable fraction of the unsatisfied demands in D are served recursively by unused vehicles of Q (step 7).

Formally, algorithm Partial$\langle Q, D, B\rangle$ is given in Figure 3. We set parameter $\rho = \Theta(\log n \log mn)$, the precise constant in the Θ-notation comes from the analysis. Let parameter $\alpha := 1 - \frac{1}{1+\log_2 m}$; all logarithms here are taken with base two. For any subset $P \subseteq [q]$, we abuse notation and use P to denote both the set of vehicles P and the multi-set of depots corresponding to vehicles P.

Lemma 3.5. *If there exists a schedule of vehicles Q covering all demands D, having makespan at most B, then* Partial *invoked on $\langle Q, D, B\rangle$ returns a schedule of vehicles Q of makespan at most $(16 + 16\rho) \cdot B$ that covers at least an $\alpha^{\log z}$ fraction of D, where $z := \min\{|Q|, 2m\} \leq 2m$.*

The final algorithm invokes Partial iteratively until all demands are covered: each time with the entire set $[q]$ of vehicles, all uncovered demands, and bound B. If $D \subseteq [m]$ is the set of uncovered demands at any iteration, Lemma 3.5 implies that Partial$\langle [q], D, B\rangle$ returns a schedule of makespan $O(\log n \cdot \log mn) \cdot B$ that serves at least $\frac{1}{4}|D|$ demands. If any call to Partial violates the condition in Lemma 3.5, we obtain a certificate that the optimum is more than B. Hence, a standard set-cover analysis implies that all demands will be covered in $O(\log m)$ rounds, resulting in an overall makespan of $O(\log m \log n \log mn) \cdot B$.

Proof of Lemma 3.5. We proceed by induction on the number $|Q|$ of vehicles. The base case $|Q| = 1$ is easy: the tour τ in Step (2) has length $O(\log n \log mn) \cdot B \leq \rho B$, and satisfies all demands (i.e. fraction 1). In the following, we prove the inductive step, when $|Q| \geq 2$.

Preprocessing Suppose Step (1) applies. By definition $d(Q_j, v) \leq B$ for all $v \in V_j$, $j \in \{1, 2\}$. So $d(Q_1, Q_2) \leq d(Q_1, v_1) + d(v_1, v_2) + d(v_2, Q_2) \leq 2B + d(v_1, v_2)$ for all $v_1 \in V_1$ and $v_2 \in V_2$. Since $d(Q_1, Q_2) > 3B$ this implies $d(V_1, V_2) > B$. Hence there is no demand with source in one of $\{V_1, V_2\}$ and destination in the other. So demands D_1 and D_2 partition D. Furthermore, in the optimal schedule, vehicles Q_j (any $j = 1, 2$) only visit vertices in V_j (otherwise the makespan would be greater than B). Thus the two recursive calls to Partial satisfy the assumption that there is some schedule of vehicles Q_j serving D_j having makespan B. Inductively, the schedule returned by Partial for each $j = 1, 2$ has makespan at most $(16 + 16\rho) \cdot B$ and covers at least $\alpha^{\log c} \cdot |D_j|$ demands from D_j, where $c \leq \min\{|Q| - 1, 2m\} \leq z$. The schedules returned by the two recursive calls to Partial can clearly be run in parallel and this covers at least $\alpha^{\log z}(|D_1| + |D_2|)$ demands, i.e. an $\alpha^{\log z}$ fraction of D. So we have the desired performance in this case.

Tour partitioning The more interesting part of the algorithm is when Step (1) does not apply: now the MST length on Q is at most $3|Q| \cdot B$. Note that when the depots Q are contracted to a single vertex, the MST on the end-points of D plus the contracted depot-vertex has length at most $|Q| \cdot B$ (the assumed makespan B schedule induces such a tree). Thus the MST on the depots Q along with end-points of D has length at most $4|Q| \cdot B$. Using the flow lower bound for mDaR, we have $\sum_{i \in D} d(s_i, t_i) \leq k|Q| \cdot B$ (again using the assumption that there is a schedule of makespan B). It follows that for the *single vehicle* Dial-a-Ride instance solved in Step (2), the Steiner and flow lower-bounds (denoted LB in Theorem 3.4) are $O(1) \cdot |Q|B$. Theorem 3.4 now implies that with high probability, τ is a 1-preemptive tour serving D, of length at most $O(\log n \cdot \log mn)|Q| \cdot B$ such that $\sum_{i \in D} T_i \leq O(\log n) \cdot |D|B$, where T_i denotes the total time spent in the vehicle by demand $i \in D$. The bound on the delay uses the fact that $\max_{i=1}^{m} d(s_i, t_i) \leq B$.

Choosing a large enough constant in $\rho = \Theta(\log n \log mn)$, the length of τ is upper bounded by $\rho|Q| \cdot B$. So, for any offset η, the cutting procedure in Step (3) results in at most $|Q|$ pieces of τ, each of length at most $2\rho B$. The objects $i \in C''$ defined in Step (3) are called a *cut objects*. Note that there are only polynomially many combinatorially distinct offsets, and we can just enumerate over them.

Claim 3.6. *The number of objects in C'' is at most $\sum_{i \in D} \frac{T_i}{\rho B} \leq O(\frac{1}{\log m}) \cdot |D|$.*

Proof: Consider choosing the offset η at random from $[0, \rho B]$. The probability that any object $i \in D$ is cut equals $\frac{T_i}{\rho B}$ where T_i is the total time spent by i in tour τ. So the expected number of cut objects,

$$\mathbb{E}[|C''|] \quad = \quad \sum_{i \in D} \frac{T_i}{\rho B} \quad \leq \quad \frac{O(\log n) \cdot |D|B}{\rho B} \quad \leq \quad O(\frac{1}{\log m}) \cdot |D|.$$

Step (3) chooses the offset minimizing $|C''|$, so the claim follows. □

We restrict attention to the objects $C' = D \setminus C''$ that are not cut. Claim 3.6 implies, again choosing a large enough constant in $\rho = \Theta(\log n \log mn)$, that:

$$|C'| \geq (1 - \frac{1}{2 \log m})|D| \geq \alpha \cdot |D| \text{ demands are } \textit{not cut}. \tag{3.2}$$

For each object $i \in C'$, the path traced by it (under single vehicle tour τ) from its source s_i to preemption-point and the path from its preemption-point to t_i are both completely contained in pieces of $\mathcal{P}$. Figure 4 gives an example of objects in C' and C'', and the cutting procedure.

Recall that the number of pieces $|\mathcal{P}| \leq |Q|$. A piece $P \in \mathcal{P}$ is said to be non-trivial if the vehicle in the 1-preemptive tour τ carries some C'-object while traversing P. Note that the number of non-trivial pieces in $\mathcal{P}$ is at most $2|C'| \leq 2m$: each C'-object appears in at most 2 pieces, one where it is moved from source to preemption-vertex and another from preemption-vertex to destination. Retain only the non-trivial pieces in $\mathcal{P}$; so $|\mathcal{P}| \leq \min\{|Q|, 2m\} = z$. The pieces in $\mathcal{P}$ may not be one-to-one assignable to the depots since the algorithm thus far has not taken the depot locations into account. We determine which pieces may be assigned to depots by considering a matching problem between $\mathcal{P}$ and the depots in Steps (4) and (5).

Load rebalancing The bipartite graph H defined in Step (4) represents which pieces and depots may be assigned to each other. Piece $P \in \mathcal{P}$ and depot $f \in Q$ are assignable iff $d(f, P) \leq 2B$, and

in this case graph H contains an edge (P, f). We claim that for any 'maximal contracting' subset $\mathcal{S} \subseteq \mathcal{P}$ chosen in Step (4), the 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \to Q \setminus \Gamma(\mathcal{S})$ in Step (5) is guaranteed to exist. Note that $|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{S}|}{2}$, but $|\Gamma(\mathcal{T})| > \frac{|\mathcal{T}|}{2}$ for all $\mathcal{T} \supset \mathcal{S}$. For any $T' \subseteq \mathcal{P} \setminus \mathcal{S}$, let $\widetilde{\Gamma}(T')$ denote the neighborhood of T' in $Q \setminus \Gamma(\mathcal{S})$. The maximality of $\mathcal{S}$ implies:

$$\frac{|\mathcal{S}|}{2} + \frac{|T'|}{2} \;=\; \frac{|\mathcal{S} \cup T'|}{2} \;\leq\; |\Gamma(\mathcal{S} \cup T')| \;=\; |\Gamma(\mathcal{S})| + |\widetilde{\Gamma}(T')| \;\leq\; \frac{|\mathcal{S}|}{2} + |\widetilde{\Gamma}(T')|, \qquad \forall\, T' \subseteq \mathcal{P} \setminus \mathcal{S}.$$

That is, $|\widetilde{\Gamma}(T')| \geq \frac{|T'|}{2}$, for all $T' \subseteq \mathcal{P} \setminus \mathcal{S}$. Hence by *Hall's condition*, there is a 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \to Q \setminus \Gamma(\mathcal{S})$. Clearly, the set $\mathcal{S}$ and 2-matching π can be efficiently computed. Figure 4 shows an example of this step.

Recursion In Step (6), demands C' are further partitioned into two sets: C_1 consists of objects that are *either* picked-up *or* dropped-off in some piece of $\mathcal{S}$; and C_2-objects are picked-up *and* dropped-off in pieces of $\mathcal{P} \setminus \mathcal{S}$. The vehicles $\Gamma(\mathcal{S})$ suffice to serve all C_1 objects, as shown below.

Claim 3.7. *There exists a feasible schedule of vehicles $\Gamma(\mathcal{S})$ serving demands C_1, having makespan B.*

Proof: Consider the schedule of makespan B that serves all demands $C' = C_1 \cup C_2$ using vehicles Q: this is implied by the assumption on instance $\langle Q, D, B\rangle$ in Lemma 3.5. We claim that in this schedule, no vehicle from $Q \setminus \Gamma(\mathcal{S})$ moves any C_1 object. Suppose (for a contradiction) that the vehicle from depot $f \in Q \setminus \Gamma(\mathcal{S})$ moves object $i \in C_1$ at some point in this schedule. Then, it must be that $d(f, s_i)$ and $d(f, t_i) \leq 2B$. But since $i \in C_1$, at least one of s_i or t_i is in a piece of $\mathcal{S}$, and this implies that there is some piece $P \in \mathcal{S}$ with $d(f, P) \leq 2B$, i.e. $f \in \Gamma(\mathcal{S})$, which is a contradiction! Thus the only vehicles participating in the movement of C_1 objects are $\Gamma(\mathcal{S})$, which implies the claim. □

In the final schedule, a *large fraction* of C_1 demands will be served recursively by vehicles $\Gamma(\mathcal{S})$, and *all* the C_2 demands are served by vehicles $Q \setminus \Gamma(\mathcal{S})$. Figure 4 shows an example of this partition.

Serving C_1 demands Based on Claim 3.7, the recursive call $\mathsf{Partial}\ \langle \Gamma(\mathcal{S}), C_1, B\rangle$ made in Step (7) satisfies the assumption required in Lemma 3.5. Since

$$|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{P}|}{2} \leq \frac{|Q|}{2} < |Q|,$$

we obtain inductively that $\mathsf{Partial}\ \langle \Gamma(\mathcal{S}), C_1, B\rangle$ returns a schedule of makespan $(16{+}16\rho)\cdot B$ covering at least $\alpha^{\log y} \cdot |C_1|$ demands of C_1, where $y = \min\{|\Gamma(\mathcal{S})|, 2m\}$. Note that $y \leq |\Gamma(\mathcal{S})| \leq |\mathcal{P}|/2 \leq z/2$ (recall $|\mathcal{P}| \leq z$), which implies that at least $\alpha^{\log z - 1}|C_1|$ demands are covered.

Serving C_2 demands These are served by vehicles $Q \setminus \Gamma(\mathcal{S})$ using the 2-matching π, in two rounds as specified in Step (7). This suffices to serve all objects in C_2 since for any $i \in C_2$, the paths traversed by object i under τ, namely $s_i \rightsquigarrow p_i$ (its preemption-point) and $p_i \rightsquigarrow t_i$ are completely contained in pieces of $\mathcal{P} \setminus \mathcal{S}$.

Claim 3.8. *The time taken in Step 7 by each vehicle $Q \setminus \Gamma(\mathcal{S})$ is at most $(16 + 16\rho) \cdot B$.*

Proof: Consider any vehicle $f \in Q \setminus \Gamma(\mathcal{S})$. Let $\pi^{-1}(f) = \{F_1, F_2\}$; recall that $|\pi^{-1}(f)| \leq 2$. The distance traveled by vehicle f in one round is at most

$$2 \cdot (d(f, F_1) + d(F_1) + d(f, F_2) + d(F_2)) \quad \leq \quad 4 \cdot (2B + 2\rho B).$$

So the time taken by this schedule is at most $2 \cdot 4(2B + 2\rho B) = (16 + 16\rho) \cdot B$. □

The schedule of vehicles $\Gamma(\mathcal{S})$ serving C_1, and vehicles $Q \setminus \Gamma(\mathcal{S})$ serving C_2 can clearly be run in parallel. This takes time $(16 + 16\rho) \cdot B$ and covers in total at least

$$|C_2| + \alpha^{\log z - 1}|C_1| \geq \alpha^{\log z - 1}|C'| \geq \alpha^{\log z}|D|$$

demands of D. This completes the proof of Lemma 3.5.

Using Lemma 3.5 repeatedly as mentioned earlier, we obtain an $O(\log m \log n \log mn)$-approximation algorithm. Using some preprocessing steps (described in Subsection 3.3), we obtain Theorem 1.3.

3.3 Weighted mDaR

Here we consider the *weighted* mDaR problem, where each object $i \in [m]$ has a weight w_i, and the capacity constraint requires that no vehicle carry a total weight of more than k at any time. The multi-vehicle Dial-a-Ride problem considered so far assumes that all objects have the same weight. We obtain an $O(\log^3 n)$-approximation algorithm for weighted mDaR. In particular, this would imply Theorem 1.3, by removing the dependence on m (number of demands) in the algorithm from Section 3.2.

The main idea is a constant-factor approximation algorithm for the *multi-vehicle stacker crane* (mSC) problem [18], i.e. the special case of mDaR when capacity $k = 1$. We present this first. The algorithm for weighted mDaR then proceeds in two phases. The first phase handles vertex-pairs that have more than k demand-weight between them, using the mSC algorithm; and the second phase is just the algorithm from Section 3.2 with the property that number of demands $m \leq n^2$.

Theorem 3.9. *There is an $O(1)$-approximation algorithm for the multi-vehicle stacker crane problem.*

Proof: An instance of the mSC problem is given by metric (V, d), m demand pairs $\{s_i, t_i\}_{i=1}^m$ and q vehicles with depots $\{r_j\}_{j=1}^q$. The goal is a minimum makespan schedule that moves each object from its source to destination, such that no vehicle carries more than one object at any time. We give a *non-preemptive* schedule that is an $O(1)$-approximation relative to the optimal preemptive schedule.

The algorithm for mSC is given in Figure 5. It follows the outline of the algorithm for general (unweighted) mDaR from Section 3.2, but is much simpler. The analysis is also similar to that of Partial in Section 3.2. To avoid repetition we only mention the changes required. We will show inductively that:

> If there is a makespan B schedule of vehicles $Q \subseteq [q]$ satisfying demands D then the mSC algorithm $\langle Q, D, B \rangle$ returns a schedule of makespan $72 \cdot B$.

The base case $q = 1$ follows directly from the single vehicle case [18]. If there is an edge of length larger than $3B$ in the MST, then we recurse with two disjoint subproblems just as in Partial. So, as discussed in Subsection 3.2, the MST on all sources/destinations in D is at most $4|Q|B$ (the

MST on vertices Q is at most $3|Q|B$, and the MST on the end-points of D is at most $|Q|B$ when all depots Q are contracted to a single vertex). That is,

$$\text{The TSP on all sources/destinations in } D \text{ has length } T \leq 8 \cdot |Q|B \tag{3.3}$$

Using the assumption on the makespan B schedule, we have that the single-vehicle flow lower bound $F := \sum_{i \in D} d(s_i, t_i) \leq |Q|B$. This means that the length of the single-vehicle tour $d(\tau) \leq 1.8 \cdot \max\{F, T\} \leq 15 \cdot |Q|B$. So we can cut tour τ into $|Q|$ disjoint pieces each of length at most $15B$. Moreover, if any object i is being carried over a cut edge, we can extend that piece to also visit the destination t_i: this is possible since the capacity $k = 1$. The increase in length of any piece is at most $\max_{i \in D} d(s_i, t_i) \leq B$. So each piece in $\mathcal{P}$ has length at most $16 \cdot B$. Thus Step (3) is well-defined.

Next, we construct the bipartite graph $\mathcal{H}$ with vertex sets $\mathcal{P}$ and Q, and choose a 'maximal contracting set' $\mathcal{S}$. Exactly as shown in the analysis of Partial, this implies a 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \to Q \setminus \Gamma(\mathcal{S})$ such that there exists a schedule of vehicles $\Gamma(\mathcal{S})$ serving all demands C in the pieces $\mathcal{S}$ with makespan B (c.f. Claim 3.7). The final schedule involves:

- Schedule for vehicles $Q \setminus \Gamma(\mathcal{S})$ given by π, which covers demands $D \setminus C$, and has makespan $4(16B + 2B) = 72 \cdot B$.
- Schedule for vehicles $\Gamma(\mathcal{S})$ obtained recursively to cover demands C, at makespan $72 \cdot B$. This follows by induction since $|\Gamma(\mathcal{S})| < |Q|$, and $\langle \Gamma(\mathcal{S}), C, B \rangle$ satisfies the makespan B assumption.

This proves the inductive step. □

Algorithm for Weighted mDaR For every $u, v \in V$ let $\mathsf{dem}_{u,v}$ denote the total weight of objects having source u and destination v. Define $H = \{(u, v) \in V \times V \mid \mathsf{dem}_{u,v} \geq k/2\}$ to be the *heavy* vertex-pairs, and $\hat{H}$ the set of demands between pairs of H. We handle the heavy demands using Theorem 3.9, and the remaining 'light' demands using the (unweighted) mDaR algorithm. Below, Opt denotes the optimal makespan of the weighted mDaR instance.

Heavy demands In this pre-processing step, we cover all demands $\hat{H}$ between heavy vertex-pairs. For each $(u, v) \in H$, we can partition all demands between them such that the total weight of each part is in the range $[\frac{k}{2}, k]$. For any $(u, v) \in H$, let $g_{u,v}$ denote the number of parts in this partition of (u, v) demands; note that $g_{u,v} \leq \frac{2}{k}\mathsf{dem}_{u,v}$. We now handle each of these $g_{u,v}$ parts as a single unsplittable object of weight k. This can be viewed as an instance of mSC (the weights and capacity are all k), which we solve using the $O(1)$-approximation algorithm in Theorem 3.9. An identical analysis yields a constant-factor approximation even relative to a solution that routes each object in a "splittable" manner (where the k units of demand between some pair of vertices may be sent on k different paths). So we obtain a non-preemptive routing for all heavy demands of makespan $O(1) \cdot \mathsf{Opt}$.

Light demands Let $L = \{(u, v) \in V \times V \mid 0 < \mathsf{dem}_{u,v} < k/2\}$ be the *light* vertex-pairs. The algorithm for these treats all (u, v) demands as a *single object* of weight $\mathsf{dem}_{u,v}$ from u to v; so there are $m = |L| \leq n^2$ distinct objects. The algorithm for this case is identical to Partial of Section 3.2 for the unweighted case: Theorem 3.4 generalizes easily to the weighted case, and the Steiner and flow lower-bounds stay the same after combining demands in L. Thus we obtain an $O(\log^2 m \log n)$ approximate schedule that covers all remaining demands. Since $m \leq n^2$, we have:

Theorem 3.10. *There is an $O(\log^3 n)$-approximation algorithm for weighted preemptive* mDaR.

3.4 Improved Guarantee for Metrics Excluding a Fixed Minor

In this section, we give an improved $O(\log^2 n)$-approximation algorithm for capacitated mDaR on metrics induced by graphs excluding any fixed minor (Theorem 1.4). The main ingredient is the following improvement in the single vehicle structure from Theorem 3.4.

Theorem 3.11. *There is a polynomial time algorithm that, given any instance of single vehicle Dial-a-Ride on a metric (V, d) excluding a fixed minor, with demand pairs D and capacity k, outputs a 1-preemptive tour τ satisfying the following conditions (here* LB *is the maximum of the Steiner and flow lower bounds, $n = |V|$ and $m \geq |D|$):*

1. *Total length: $d(\tau) \leq O(\log mn) \cdot \mathsf{LB}$.*
2. *Bounded delay: $\sum_{i \in D} T_i \leq O(1) \cdot \sum_{i \in D} d(s_i, t_i)$ where T_i is the total time spent by object $i \in D$ in the vehicle under the schedule given by τ.*

Using this 1-preemptive tour within the algorithm of Section 3.2, setting parameter $\rho = \Theta(\log mn)$, we immediately obtain an $O(\log m \log mn)$-approximation algorithm for capacitated mDaR on such metrics. Combined with the preprocessing in Subsection 3.3, we obtain an $O(\log^2 n)$-approximation algorithm for even weighted mDaR. In proving Theorem 3.11, we first show that metrics induced by fixed minor-free graphs allow a so-called *γ-separated cover* (Subsection 3.4.1). Then we show (in Subsection 3.4.2) how this property can be used to obtain the single-vehicle tour claimed in Theorem 3.11.

3.4.1 γ-Separated Covers

We are given a metric (V, d) that is induced by a graph $G = (V, E)$ with edge-lengths $w : E \to \mathbb{Z}_+$. For any pair $u, v \in V$ of vertices, the distance $d(u, v)$ equals the shortest path between u and v under edge-lengths w. We assume that the graph G does not contain any K_r-minor; here r is a fixed parameter. Recall that a *cluster* refers to any subset of vertices. We prove the following.

Theorem 3.12. *Given a K_r-minor free graph $G = (V, E, w)$ and integer $\gamma \geq 0$, there is a polynomial-time algorithm to compute a collection $\mathcal{C}$ of clusters along with a partition $Z_1, \ldots, Z_p$ of $\mathcal{C}$ such that:*

1. *$p \leq 3^r$.*
2. *For each $l \in [p]$ and distinct $A, B \in Z_l$, the distance between A and B, $d(A, B) \geq \gamma$.*
3. *The diameter of any cluster $S \in \mathcal{C}$ is $\max_{u,v \in S} d(u, v) \leq O(r^2) \cdot \gamma$.*
4. *For any pair $u, v \in V$ with $d(u, v) \leq \gamma$, there is some cluster $A \in \mathcal{C}$ having $u, v \in A$.*

Such a collection $\mathcal{C}$ is called γ-separated cover.

The proof of this theorem is based on the KPR decomposition algorithm [26]. In fact, without conditions (1-2), it is implied by Theorem 2.3. Achieving conditions (1-2) requires a further modification to the KPR decomposition, as described below.

We first remind the reader of the main property of the KPR decomposition [26]. We will consider graph G as having unit length edges, by subdividing each edge $(u, v) \in E$ to become a path of length w_{uv} between u and v. This increases the number of vertices, and we denote the new vertex set also by V. The distance function d remains unchanged by this modification. We show later (using standard scaling arguments) that one can always ensure $\max_{u,v} w_{uv}$ is polynomial in the original number n of vertices.

Definition 3.13 (Theorem 4.2, [26])**.** Let $G_1 = G$ and $\delta \in \mathbb{Z}_+$ a distance parameter. Let $G_1, G_2, \cdots, G_{r+1}$ be any sequence of subgraphs of G, where each G_{i+1} is obtained from G_i as follows:

1. Construct a breadth-first-search tree in G_i from an arbitrary root-vertex.
2. Select any set of δ consecutive levels in this breadth-first-search tree, and let G_{i+1} be any connected component of the subgraph in G induced by these levels.

Then, the diameter of G_{r+1} is $O(r^2) \cdot \delta$.

We are now ready to proceed with the proof of Theorem 3.12. We give a recursive procedure Split in Figure 6 to generate the desired γ-separated cover $\mathcal{C}$ of G. The input to Split consists of an induced subgraph $H = (V(H), E(H))$ of G, set $T \subseteq V(H)$ of terminals, depth i of recursion and color $\tau \in \{0, 1, 2\}^i$. We initialize $\mathcal{C} = \emptyset$, and invoke Split$\langle G, V, 0, \phi \rangle$.

At the end of the algorithm, $\mathcal{C}$ consists of several tuples of the form $\langle C, \tau \rangle$ where C is a cluster and $\tau \in \{0, 1, 2\}^r$ is its color. This naturally corresponds to partitioning $\mathcal{C}$ into $p = 3^r$ parts: clusters of the same color form a part. We now show that $\mathcal{C}$ satisfies the conditions in Theorem 3.12. Condition 1 clearly holds by the construction.

Condition 4. Let $u, v \in V$ be such that $d(u, v) \leq \gamma$, and let $P \subseteq V$ denote the vertices on any shortest $u - v$ path in G; note that $|P| \leq \gamma + 1$. Then, we have the following.

Claim 3.14. *For each $i \in \{0, \cdots, r\}$, there is some call to* Split *at depth i such that all vertices of P appear as terminals.*

Proof: The base case $i = 0$ is obvious since all vertices are terminals. Assuming the claim to be true for depth i, we prove it for $i + 1$. Let $\langle H, T, i, \tau \rangle$ denote the call to Split at depth i where $P \subseteq T \subseteq V(H)$. Since the P-vertices define a path of length at most γ (in G and also H), all P-vertices appear in some $\gamma + 1$ consecutive levels of any BFS on H. Thus there exist values $j \in \{0, 1, 2\}$ and $l \in \mathbb{Z}_+$ such that P is contained in levels $\{(3l + j)\gamma, \cdots, (3l + j + 2)\gamma\}$ of the BFS on H. This implies that one of the recursive calls (corresponding to this value of j and l) contains P as terminals. □

Using this claim when $i = r$ implies that there is some depth r call where both u and v (in fact all of P) are terminals; hence some cluster in $\mathcal{C}$ contains both u and v.

Condition 3. This is a direct consequence of the KPR decomposition. Note that every subgraph at depth r of algorithm Split is obtained from G via the sequence of operations in Definition 3.13 with $\delta = 4\gamma + 1$. Thus each such subgraph has diameter $O(r^2) \cdot \gamma$. Finally, any cluster in $\mathcal{C}$ is a subset of some subgraph at depth r of Split; so it also has diameter $O(r^2) \cdot \gamma$.

Condition 2. This well-separated property is the main reason for the use of 'terminals' in algorithm Split and the decomposition defined in Step (3). Let A and B be any two distinct clusters in $\mathcal{C}$

having the *same color* : we will show that $d(A, B) \geq \gamma$. Each of A and B corresponds to a root-leaf path in the recursion tree for algorithm Split, where the root is $\mathsf{Split}\langle G, V, 0, \phi\rangle$. Consider the call $\langle H, T, i, \tau\rangle$ to Split after which the paths for A and B diverge; let $\alpha_a = \langle H_a, T_a, i+1, \tau'\rangle$ and $\alpha_b = \langle H_b, T_b, i+1, \tau'\rangle$ denote the respective recursive calls from $\langle H, T, i, \tau\rangle$ where $A \subseteq T_a$ and $B \subseteq T_b$ (note that both have the same color τ' since A and B have the same color at the end of the algorithm). We will show that $d(T_a, T_b) \geq \gamma$ which in particular implies $d(A, B) \geq \gamma$. Note that both α_a and α_b are generated by the same value $j \in \{0, 1, 2\}$, since they have the same color τ'. Let α_a (resp. α_b) correspond to value $l = l_a$ (resp. $l = l_b$). Consider two cases:

1. $l_a \neq l_b$. In this case, we show that the terminal-sets T_a and T_b are far apart in the BFS on H. Observe that T_a appears within levels $\{(3l_a + j)\gamma, \cdots, (3l_a + j + 2)\gamma\}$; and T_b within $\{(3l_b + j)\gamma, \cdots, (3l_b + j + 2)\gamma\}$. If $l_a < l_b$ (the other case is identical), then $3l_b + j \geq 3l_a + j + 3$; i.e. T_a and T_b are separated by at least $\gamma - 1$ levels in the BFS of H. Using Claim 3.15 below, we have $d(T_a, T_b) \geq \gamma$ in G; otherwise H contains a path from T_a to T_b of length at most $\gamma - 1$, meaning that T_a and T_b are separated by $\leq \gamma - 2$ levels in the BFS, a contradiction!

2. $l_a = l_b = l$. In this case, it must be that H_a and H_b are two *disconnected components* of the subgraph (of G) induced on levels $\{(3l + j - 1)\gamma, \cdots, (3l + j + 3)\gamma\}$ of the BFS on H. Furthermore, T_a and T_b both appear within levels $\{(3l + j)\gamma, \cdots, (3l + j + 2)\gamma\}$. Again by Claim 3.15, we must have $d(T_a, T_b) \geq \gamma$ in G; otherwise H contains a path from T_a to T_b of length at most $\gamma - 1$, which would mean that H_a and H_b are connected within levels $\{(3l + j - 1)\gamma, \cdots, (3l + j + 3)\gamma\}$.

Claim 3.15. *For every call $\mathsf{Split}\langle H, T, i, \tau\rangle$, and terminals $x, y \in T$, if $d(x, y) \leq \gamma$ (in graph G) then H contains an $x - y$ path of length at most $d(x, y)$.*

Proof: We proceed by induction on i. The claim is obviously true for the call $\mathsf{Split}\langle G, V, 0, \phi\rangle$ at depth 0 since $H = G$. Assuming the claim for any depth i call $\mathsf{Split}\langle H, T, i, \tau\rangle$, we prove it for any call $\mathsf{Split}\langle H', T', i+1, \tau'\rangle$ generated by it. Let $x, y \in T'$ with $d(x, y) \leq \gamma$ in G. Clearly $x, y \in T$, and by the induction hypothesis, H contains an $x - y$ path π of length at most $d(x, y) \leq \gamma$. It suffices to show that π is also contained in H'. Let $j \in \{0, 1, 2\}$ and $l \in \mathbb{Z}_+$ denote the values that generated the recursive call $\mathsf{Split}\langle H', T', i+1, \tau'\rangle$. Since $x, y \in T'$, both these vertices lie within levels $\{(3l + j)\gamma, \cdots, (3l + j + 2)\gamma\}$ of the BFS on H. Furthermore, π is an $x - y$ path in H of length at most γ; so π is contained within levels $\{(3l + j - 1)\gamma, \cdots, (3l + j + 3)\gamma\}$ and so it lies in the connected component H' that contains x and y. □

Running time. Given an input graph $G = (V, E, w)$ and parameter γ as in Theorem 3.12, we first show how to ensure that $\max_{u,v} w_{uv}$ is polynomial in n (the original number of vertices). To this end, delete all edges in E with weight more than $\gamma \cdot n^2$, to obtain edge-set E'; note that (V, E') is also K_r-minor free. Set new weights $w'_{uv} = \lfloor n^2 w_{uv}/\gamma \rfloor$ for all $(u, v) \in E'$; and let d' be the resulting distance function (given by shortest paths under w'). Note that for all $u, v \in V$: (a) $d'(u, v) \geq \frac{n^2 d(u,v)}{\gamma} - n$, and (b) if $d(u, v) \leq \gamma n^2$ then $d'(u, v) \leq \frac{n^2 d(u,v)}{\gamma}$. This new instance has integer weights with $\max_{u,v} w'_{uv} \leq n^4$ as desired. Let $\mathcal{C}$ be the collection of clusters obtained from the above algorithm applied to (V, E', w') and parameter $\gamma' = n^2$, which satisfies the four properties in Theorem 3.12. We show below that $\mathcal{C}$ also satisfies all these properties w.r.t. the original input (V, E, w) and γ.

1. The first condition is obvious, since we use the same collection $\mathcal{C}$.

2. For any $A, B \in \mathcal{C}$, if $d(A,B) \le \gamma$ then $d'(A,B) \le \frac{n^2 d(uv)}{\gamma} \le n^2 = \gamma'$; i.e. $d'(A,B) \ge \gamma'$ implies $d(A,B) \ge \gamma$. This shows condition 2.

3. For any $u, v \in V$, if $d'(u,v) = O(r^2) \cdot \gamma'$ then $d(u,v) \le \frac{\gamma}{n^2} d'(u,v) + \frac{\gamma}{n} = O(r^2) \cdot \gamma$, which proves condition 3.

4. For any $u, v \in V$, if $d(u,v) \le \gamma$ then $d'(u,v) \le \frac{n^2 d(u,v)}{\gamma} = \gamma'$. This gives condition 4.

Since $\max_{u,v} w_{uv}$ is polynomial in n, it is easy to see that each call to Split generates only polynomial in n recursive calls. Since the depth of the recursion is at most r, the running time is polynomial for any fixed r.
This completes the proof of Theorem 3.12.

We note that the collection $\mathcal{C}$ also satisfies the 'sparse cover' property (i.e. condition (3) in Theorem 2.3), namely each vertex $v \in V$ appears in at most $O(2^r)$ clusters. It is easy to show (inductively) that the number of depth i calls to Split where any vertex v appears as a terminal is at most 2^i. However this property is not required in the proof of Theorem 3.11 that appears next.

3.4.2 Improved Bounded Delay Tour

Recall that we are given an instance of single-vehicle Dial-a-Ride on a metric (V,d) induced by an edge-weighted K_r-minor free graph $G = (V, E, w)$; here $w : E \to \mathbb{Z}_+$ denotes the edge-lengths and d the shortest-path distances under w. The vehicle has capacity k and $\{s_i, t_i\}_{i \in D}$ are the demand-pairs. Again r is a fixed constant. LB denotes the maximum of the Steiner (i.e. minimum TSP tour on all sources and destinations) and flow lower-bounds (i.e. $\frac{1}{k}\sum_{i \in D} d(s_i, t_i)$) for this instance.

Let Δ denote the diameter of the metric; as discussed in the proof of Theorem 3.4, we may assume, without loss of generality, that Δ is polynomial in $|D|$ and $n = |V|$. We group the demands D into into $\lceil \log_2 \Delta \rceil$ groups based on the source-destination distances: For each $j = 1, \cdots, \lceil \log_2 \Delta \rceil$, group G_j consists of those demands $i \in D$ with $2^{j-1} \le d(s_i, t_i) \le 2^j$. We will show how to service each group G_j separately at a cost of $O(1) \cdot \mathsf{LB}$, such that the time spent by each object $i \in G_j$ in the vehicle is $T_i = O(2^j) = O(1) \cdot d(s_i, t_i)$. Since the number of groups is $O(\log n|D|)$, this would immediately imply Theorem 3.11.

Serving group G_j Set distance parameter $\gamma = 2^j$, and obtain a γ-separated cover $\mathcal{C}$ along with its partition $\{Z_1, \cdots, Z_p\}$, using Theorem 3.12. For any demand $i \in G_j$, since $d(s_i, t_i) \le \gamma$, by property (4) of Theorem 3.12, there is some cluster of $\mathcal{C}$ containing both s_i and t_i: *assign* demand i to such a cluster. We further partition demands in G_j into $H_1, \cdots, H_p$ where each H_l contains all demands assigned to clusters of Z_l. The algorithm will serve demands in each H_l separately using a tour of length $O(1) \cdot \mathsf{LB}$, that also satisfies the bounded delay condition 2. This suffices to prove Theorem 3.11 since $p = O(1)$ by property (1) of Theorem 3.12.

Serving H_l For any cluster $S \in Z_l$ let $B(S) \subseteq G_j$ denote the demands assigned to S. Let $\mathsf{LB}(S)$ denote the maximum of the Steiner and flow lower bounds restricted to demands $B(S)$. A cluster $S \in Z_l$ is called non-trivial if $B(S) \neq \emptyset$. Choose an arbitrary vertex in each cluster as its *center*. Let $\mathcal{T}$ denote a 1.5-approximate TSP tour [13] containing the centers of all non-trivial clusters of Z_l.

Claim 3.16. *We have* $\sum_{S \in Z_l} \mathsf{LB}(S) \le O(1) \cdot \mathsf{LB}$, *and* $d(\mathcal{T}) \le O(1) \cdot \mathsf{LB}$.

Proof: It is clear that the sum of the flow lower-bounds over all clusters $S \in Z_l$ is at most the flow lower-bound on demands H_l, since $H_l = \cup_{S \in Z_l} B(S)$. Let $\mathsf{Tsp}(S)$ denote the minimum length TSP on the end points of demands $B(S)$, i.e. the Steiner lower bound on S. We show next that $\sum_{S \in Z_l} \mathsf{Tsp}(S) \leq O(1) \cdot \mathsf{Tsp}$ where Tsp denotes the minimum TSP on end points of H_l (i.e. Steiner lower bound on H_l). This would prove the first part of the claim.

For any cluster $S \in Z_l$, let $n(S)$ denote the number of connected segments in $\mathsf{Tsp} \cap S$ (i.e. Tsp restricted to vertices of S), and let $C_1^S, \cdots, C_{n(S)}^S$ denote these segments. Note that the tour Tsp travels from one cluster in Z_l to another at least $\sum_{S \in Z_l} n(S)$ times. By property (2) of Theorem 3.12, the distance between any two distinct clusters in Z_l is at least γ. So we obtain:

$$\mathsf{Tsp} \geq \gamma \sum_{S \in Z_l} n(S) \tag{3.4}$$

We define a TSP tour on end-points of demands $B(S)$ by connecting the end points of segments $C_1^S, \cdots, C_{n(S)}^S$. This requires $n(S)$ new edges, each of length at most $O(\gamma)$, the diameter of cluster S (by property (3) of Theorem 3.12). So the minimum TSP tour for any $S \in Z_l$ has length $\mathsf{Tsp}(S) \leq \sum_{a=1}^{n(S)} d(C_a^S) + O(\gamma) \cdot n(S)$. Hence,

$$\sum_{S \in Z_l} \mathsf{Tsp}(S) \leq \sum_{S \in Z_l} \sum_{a=1}^{n(S)} d(C_a^S) + O(\gamma) \sum_{S \in Z_l} n(S) \leq \mathsf{Tsp} + O(\gamma) \sum_{S \in Z_l} n(S) \leq O(1) \cdot \mathsf{Tsp} \; .$$

The second inequality follows from the fact that the C_l^Ss are disjoint segments in Tsp, and the last inequality uses (3.4).

To obtain the second part of the claim, augment Tsp as follows. For each non-trivial cluster $S \in Z_l$, add two edges to the center of S from some vertex in $\mathsf{Tsp} \cap S$. This gives a TSP tour visiting the centers of all non-trivial clusters in Z_l. Note that the number of non-trivial clusters of Z_l is at most $\sum_{S \in Z_l} n(S)$ since $n(S) \geq 1$ for all non-trivial $S \in Z_l$. Since the diameter of each $S \in Z_l$ is $O(\gamma)$ (property (3) of Theorem 3.12), the increase in length of Tsp is at most $O(\gamma) \sum_{S \in Z_l} n(S) \leq O(1) \cdot \mathsf{Tsp}$ by (3.4). So there is a TSP tour on the centers of Z_l's non-trivial clusters having length at most $O(1) \cdot \mathsf{LB}$. □

For each $S \in Z_l$, the demands $B(S)$ assigned to it are served separately. The flow lower-bound on S is at least $\frac{\gamma}{2k}|B(S)|$; recall that each demand $i \in G_j$ has $d(s_i, t_i) \geq \gamma/2$. Consider an instance $\mathcal{I}_{src}$ of capacitated vehicle routing (CVRP) with all sources from $B(S)$ and S's center as the common destination. Since the diameter of S is $O(\gamma)$, the flow lower bound of $\mathcal{I}_{src}$ is at most $O(\gamma) \cdot \frac{|B(S)|}{k} \leq O(1) \cdot \mathsf{LB}(S)$. The Steiner lower bound of $\mathcal{I}_{src}$ is also $O(1) \cdot \mathsf{LB}(S)$. So Theorem 3.2 (with delay parameter $\beta = 2$) implies a non-preemptive tour $\tau_{src}(S)$ that moves all objects from sources of $B(S)$ to the center of S, having length $O(1) \cdot \mathsf{LB}(S)$ such that each object spends at most $O(1) \cdot \gamma$ time in the vehicle. Similarly, we can obtain non-preemptive tour $\tau_{dest}(S)$ that moves all objects from the center of S to destinations of $B(S)$, having length $O(1) \cdot \mathsf{LB}(S)$ such that each object spends at most $O(1) \cdot \gamma$ time in the vehicle. Concatenating $\tau_{src}(S)$ and $\tau_{dest}(S)$ gives a 1-preemptive tour serving demands in S having length $O(1) \cdot \mathsf{LB}(S)$ such that each object $i \in B(S)$ spends at most $O(\gamma) \leq O(1) \cdot d(s_i, t_i)$ time in the vehicle.

The final solution for demands H_l traverses the TSP tour $\mathcal{T}$ on centers of all non-trivial clusters of Z_l, and serves the demands in each cluster (as above) when it is visited. The resulting solution has length at most $d(\mathcal{T}) + O(1) \sum_{S \in Z_l} \mathsf{LB}(S) \leq O(1) \cdot \mathsf{LB}$, by Claim 3.16. Moreover, each object $i \in H_l$ spends $O(1) \cdot d(s_i, t_i)$ time in the vehicle.

Finally, concatenating the tours serving each H_l (for $l = 1, \cdots, p$), and then the tours serving each G_j (for $j = 1, \cdots, \lceil \log_2 \Delta \rceil$), we obtain the 1-preemptive tour claimed in Theorem 3.11.

4 Conclusion

We studied the preemptive multi-vehicle Dial-a-Ride problem and obtained an $O(\log n)$-approximation algorithm in the uncapacitated case and an $O(\log^3 n)$-approximation algorithm in the capacitated setting. Both guarantees improve by a logarithmic factor when the metric is induced by a graph excluding some fixed minor. While there is an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation for capacitated mDaR, there is no $\omega(1)$-factor hardness result for the uncapacitated case: obtaining a constant-factor approximation for uncapacitated mDaR is an interesting open question. Another interesting direction is the multi-vehicle Dial-a-Ride problem with non-uniform capacities: our approximation algorithm only works when the ratio of maximum to minimum vehicle capacities is a constant, and it would be interesting to obtain a poly-log ratio approximation algorithm for general non-uniform capacities.

Acknowledgements

We thank an anonymous reviewer for pointing out the derandomization of our earlier (randomized) algorithm for Dial-a-Ride with bounded delay. This lead to a deterministic approximation algorithm for the capacitated multi-vehicle Dial-a-Ride problem, that is presented here.

References

[1] I. Abraham, C. Gavoille, D. Malkhi, and U. Wieder. Strong-diameter decompositions of minor free graphs. *Theory Comput. Syst.*, 47(4):837–855, 2010.

[2] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

[3] S. Antonakopoulos, C. Chekuri, F. B. Shepherd, and L. Zhang. Buy-at-bulk network design with protection. *Math. Oper. Res.*, 36(1):71–87, 2011.

[4] B. Awerbuch and Y. Azar. Buy-at-bulk network design. In *FOCS*, pages 542–547, 1997.

[5] B. Awerbuch, A. Baratz, and D. Peleg. Cost-sensitive analysis of communication protocols. In *Proceedings of the ninth annual ACM symposium on Principles of distributed computing*, PODC '90, pages 177–187, 1990.

[6] B. S. Baker. Approximation Algorithms for NP-Complete Problems on Planar Graphs. *Journal of ACM*, 41:153–180, 1994.

[7] C. Busch, R. LaFortune, and S. Tirthapura. Improved sparse covers for graphs excluding a fixed minor. In *PODC*, pages 61–70, 2007.

[8] I.-M. Chao. A tabu search method for the truck and trailer routing problem. *Computer & Operations Research*, 29:469–488, 2002.

[9] M. Charikar and A. Karagiozova. On non-uniform multicommodity buy-at-bulk network design. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 176–182. ACM, 2005.

[10] M. Charikar, S. Khuller, and B. Raghavachari. Algorithms for capacitated vehicle routing. *SIAM J. Comput.*, 31(3):665–682, 2001.

[11] M. Charikar and B. Raghavachari. The Finite Capacity Dial-A-Ride Problem. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 458–467, 1998.

[12] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM J. Comput.*, 39(5):1772–1798, 2010.

[13] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. *GSIA, CMU-Report 388*, 1977.

[14] J.-F. Cordeau and G. Laporte. The Dial-a-Ride Problem (DARP): Variants, modeling issues and algorithms. *4OR: A Quarterly Journal of Operations Research*, 1(2), 2003.

[15] W. E. de Paepe, J. K. Lenstra, J. Sgall, R. A. Sitters, and L. Stougie. Computer-Aided Complexity Classification of Dial-a-Ride Problems . *Informs Journal on Computing*, 16(2):120–132, 2004.

[16] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Min-max tree covers of graphs . *Operations Research Letters*, 32(4):309–315, 2004.

[17] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[18] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. *SIAM Journal on Computing*, 7(2):178–193, 1978.

[19] I. L. Gørtz. Hardness of Preemptive Finite Capacity Dial-a-Ride. In *APPROX-RANDOM*, pages 200–211, 2006.

[20] F. Grandoni, T. Rothvoß, and L. Sanità. From uncertainty to nonlinearity: Solving virtual private network via single-sink buy-at-bulk. *Math. Oper. Res.*, 36(2):185–204, 2011.

[21] A. Gupta, M. T. Hajiaghayi, V. Nagarajan, and R. Ravi. Dial a ride from k-forest. *ACM Transactions on Algorithms*, 6(2), 2010.

[22] M. Haimovich and A. H. G. R. Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research*, 10:527–542, 1985.

[23] L. A. Hall, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line algorithms. In *Proceedings of the ACM-SIAM symposium on Discrete algorithms*, pages 142–151, 1996.

[24] D. Hochbaum and W. Maass. Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI. *Journal of ACM*, 32:130–136, 1985.

[25] S. Khuller, B. Raghavachari, and N. E. Young. Balancing Minimum Spanning Trees and Shortest-Path Trees. *Algorithmica*, 14(4):305–321, 1995.

[26] P. Klein, S. A. Plotkin, and S. Rao. Excluded minors, network decomposition, and multicommodity flow. In *STOC*, pages 682–690, 1993.

[27] F. Lazebnik, V. Ustimenko, and A. Woldar. A New Series of Dense Graphs of High Girth. *Bulletin of the AMS*, 32(1):73–79, 1995.

[28] S. Mitrović-Minić and G. Laporte. The Pickup and Delivery Problem with Time Windows and Transshipment. *Information Systems and Operational Research*, 44:217–227, 2006.

[29] C. Mues and S. Pickl. Transshipment and time windows in vehicle routing. In *8th Int. Symp. on Parallel Architectures, Algorithms and Networks*, pages 113–119, 2005.

[30] Y. Nakao and H. Nagamochi. Worst case analysis for pickup and delivery problems with transfer. *IEICE Trans. on Fund. of Electronics, Comm. and Computer Sci.*, E91-A(9), 2008.

[31] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *J. ACM*, 36(3):510–530, 1989.

[32] M. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation Science*, 29:17–29, 1995.

[33] S. Scheuerer. A tabu search heuristic for the truck and trailer routing problem. *Computer & Operations Research*, 33:894–909, 2006.

Input: Vehicles $Q \subseteq [q]$, demands $D \subseteq [m]$, bound $B \geq 0$.

Preprocessing

1. If the minimum spanning tree (MST) on vertices Q contains an edge of length greater than $3B$, there is a non-trivial partition $\{Q_1, Q_2\}$ of Q with $d(Q_1, Q_2) > 3B$. For $j \in \{1, 2\}$, let $V_j = \{v \in V \mid d(Q_j, v) \leq B\}$ and D_j be all demands of D induced on V_j. Run in parallel the schedules from $\mathsf{Partial}\langle Q_1, D_1, B\rangle$ and $\mathsf{Partial}\langle Q_2, D_2, B\rangle$. In the following, assume that every edge of the MST has length at most $3B$.

Tour partitioning

2. Obtain *single-vehicle* 1-preemptive tour τ with capacity k and demands D, by applying Theorem 3.4.

3. For any offset $\eta \in [0, \rho B]$, cut edges of tour τ at distances $\{p\rho B + \eta \mid p = 1, 2, \cdots\}$ along the tour to obtain a set $\mathcal{P}$ of pieces of τ. Let C'' be the set of objects $i \in D$ such that i is carried by the vehicle in τ over some cut edge; and $C' := D \setminus C''$. Choose η so that $|C'|$ is maximized.

Load rebalancing

4. Construct bipartite graph H with vertex sets $\mathcal{P}$ and Q and an edge between piece $P \in \mathcal{P}$ and depot $f \in Q$ iff $d(f, P) \leq 2B$. For any subset $A \subseteq \mathcal{P}$, $\Gamma(A) \subseteq Q$ denotes the neighborhood of A in graph H. Let $\mathcal{S} \subseteq \mathcal{P}$ be any *maximal* set that satisfies $|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{S}|}{2}$.

5. Compute a *2-matching* $\pi : \mathcal{P} \setminus \mathcal{S} \to Q \setminus \Gamma(\mathcal{S})$, i.e. function such that $(P, \pi(P))$ is an edge in H for all $P \in \mathcal{P} \setminus \mathcal{S}$, and the number of pieces mapping to any $f \in Q \setminus \Gamma(\mathcal{S})$ is $|\pi^{-1}(f)| \leq 2$.

Recursion

6. Define $C_1 := \{i \in C' \mid \text{ either } s_i \in \mathcal{S} \text{ or } t_i \in \mathcal{S}\}$; and $C_2 := C' \setminus C_1$.

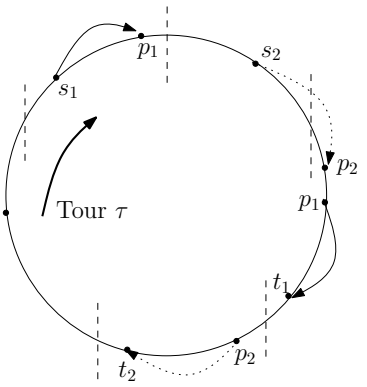
7. Run in parallel the *recursive* schedule $\mathsf{Partial}\langle \Gamma(\mathcal{S}), C_1, B\rangle$ for C_1 and the following for C_2:

   (a) Each vehicle $f \in Q \setminus \Gamma(\mathcal{S})$ traverses the pieces $\pi^{-1}(f)$, moving all C_2-objects in them from their source to preemption-vertex, and returns to its depot.

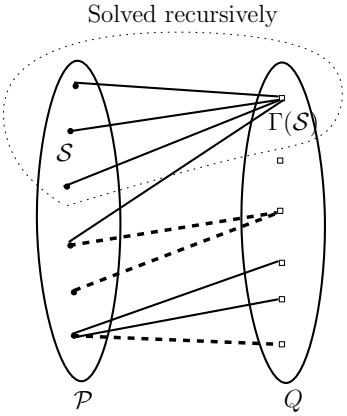
   (b) Each vehicle $f \in Q \setminus \Gamma(\mathcal{S})$ again traverses the pieces $\pi^{-1}(f)$, this time moving all C_2-objects in them from their preemption-vertex to destination, and returns to its depot.

Output: A schedule of Q of makespan $(16 + 16\rho) \cdot B$ that serves an $\alpha^{\log \min\{|Q|, 2m\}}$ fraction of D; or a certificate that the optimal makespan is more than B.

Figure 3: Algorithm $\mathsf{Partial}\langle Q, D, B\rangle$ for capacitated mDaR.



The 1-preemptive tour τ is cut at the dashed lines.
Object 1 is in C', it is not cut.
Object 2 is not in C', it is a cut object.



The bipartite graph H
The 2-matching π is shown by dashed edges.

Figure 4: Cutting and patching steps in algorithm Partial.

Input: Vehicles $Q \subseteq [q]$, demands $D \subseteq [m]$, bound $B \geq 0$.

1. If the minimum spanning tree (MST) on vertices Q contains an edge of length greater than $3B$, there is a non-trivial partition $\{Q_1, Q_2\}$ of Q with $d(Q_1, Q_2) > 3B$. For $j \in \{1, 2\}$, let $V_j = \{v \in V \mid d(Q_j, v) \leq B\}$ and D_j be all demands of D induced on V_j. Run in parallel the schedules from $\langle Q_1, D_1, B\rangle$ and $\langle Q_2, D_2, B\rangle$. In the following, assume that every edge of the MST has length at most $3B$.

2. Obtain single-vehicle (non-preemptive) stacker crane solution τ for demands D, by applying the algorithm from [18]. The length $d(\tau)$ is at most 1.8 times the Steiner and flow lower-bounds.

3. Cut τ into $|Q|$ disjoint pieces $\mathcal{P}$, each of length $\leq 16 \cdot B$, so that *no* object is carried over a cut edge in τ.

4. Construct bipartite graph H with vertex sets $\mathcal{P}$ and Q and an edge between piece $P \in \mathcal{P}$ and depot $f \in Q$ iff $d(f, P) \leq 2B$. For any subset $A \subseteq \mathcal{P}$, $\Gamma(A) \subseteq Q$ denotes the neighborhood of A in graph H. Let $\mathcal{S} \subseteq \mathcal{P}$ be any *maximal* set that satisfies $|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{S}|}{2}$.

5. Compute a *2-matching* $\pi : \mathcal{P} \setminus \mathcal{S} \to Q \setminus \Gamma(\mathcal{S})$, i.e. function such that $(P, \pi(P))$ is an edge in H for all $P \in \mathcal{P} \setminus \mathcal{S}$, and the number of pieces mapping to any $f \in Q \setminus \Gamma(\mathcal{S})$ is $|\pi^{-1}(f)| \leq 2$.

6. Define $C \subseteq D$ be the demands served in pieces of $\mathcal{S}$. Run in parallel the *recursive* schedule $\langle \Gamma(\mathcal{S}), C, B\rangle$ for C and the following for $D \setminus C$:
   Each vehicle $f \in Q \setminus \Gamma(\mathcal{S})$ traverses the pieces $\pi^{-1}(f)$.

Output: A schedule of makespan $72 \cdot B$; or a certificate that the optimal makespan is more than B.

Figure 5: Algorithm for multi-vehicle stacker crane.

Input: induced subgraph $H = (V(H), E(H))$, terminals $T \subseteq V(H)$, depth i and color $\tau \in \{0,1,2\}^i$.

1. If $i = r$ then add tuple $\langle T, \tau \rangle$ to collection $\mathcal{C}$, and stop.
2. Construct a breadth-first-search (BFS) tree from any vertex u of H. For any $v \in V(H)$ define level_v to be the length of the shortest path from u to v (so $\mathsf{level}_u = 0$).
3. For each $j \in \{0,1,2\}$ and integer $l \geq 0$, consider the subgraph of G induced by vertices $\{v \in V(H) : (3l + j - 1)\gamma \leq \mathsf{level}_v \leq (3l + j + 3)\gamma\}$; and for each connected component H' in this subgraph do:
   (a) Set $T' \leftarrow \{v \in T \cap V(H') : (3l + j)\gamma \leq \mathsf{level}_v \leq (3l + j + 2)\gamma\}$.
   (b) Set $\tau' \leftarrow \tau$ concatenated with j.
   (c) Recurse on $\mathsf{Split}\langle H', T', i + 1, \tau' \rangle$.

Figure 6: Algorithm for γ-separated cover.