Solving Min-Max Multi-Depot Vehicle Routing Problem*

John Carlsson†, Dongdong Ge‡, Arjun Subramaniam§,
Amy Wu¶, and Yinyu Ye‖

October 24, 2006

Abstract

The Multi-Depot Vehicle Routing Problem (MDVRP) is a generalization of the Single-Depot Vehicle Routing Problem (SDVRP) in which vehicle(s) start from multiple depots and return to their depots of origin at the end of their assigned tours. The traditional objective in MDVRP is to minimize the total length of all the tours, and existing literature handles this problem with a variety of assumptions and constraints. In this paper, we explore the notion of minimizing the maximal length of a tour in MDVRP ("min-max MDVRP"). We also introduce a heuristic method based on region partitioning, which is potentially useful for general network applications. A comparison of the computational implementations for three heuristics is included. Although this model is advantageous for real-world applications, no prior exploration into min-max MDVRP has been published to the best of our knowledge.

Keywords: vehicle routing problem, regionpartition, heuristic.

1 Introduction

The Vehicle Routing Problem (VRP) has been one of the central topics in optimization since Dantzig proposed the problem in 1959 [10]. A simple general model of VRP can be described as follows: a set of service vehicles need to visit all customers in a geographical region with the minimum cost. In the Single-Depot Vehicle Routing Problem (SDVRP), multiple vehicles leave from a single location (a "depot") and must return to that location after completing their assigned tours. The Multi-Depot Vehicle Routing Problem (MDVRP) is a generalization of SDVRP in which multiple vehicles start from multiple depots and return to their original depots at the end of their assigned tours. The traditional objective in MDVRP is to minimize the total length of all the tours, and

*This research is supported in part by the Boeing Company.

†Institute of Computational and Mathematical Engineering.

‡Department of Management Science and Engineering.

§Department of Management Science and Engineering.

¶Department of Computer Science.

‖Department of Management Science and Engineering and, by courtesy, Electrical Engineering, E–mail: yinyu-ye@stanford.edu; Stanford University, Stanford, CA 94305, USA.

existing literature handles this problem with a variety of assumptions and constraints. Common techniques include integer programming and clustering followed by routing with tabular search improvement. The paper [5] by Baltz et al. presented a probabilistic analysis of the optimal solution for the problem. Another paper [21] proposed a polynomial-time approximation scheme (PTAS) similar to Arora's PTAS algorithm for TSP.

In our paper, we explore the notion of minimizing the maximal length of a tour in a MDVRP ("min-max MDVRP") using both theoretical analysis and implementation. No prior exploration into min-max MDVRP has been published to the best of our knowledge, but this formulation is advantageous for a number of applications. Consider a network model in which depots represent servers and customers represent clients. A network routing topology generated by solving min-max MDVRP results in a set of daisy-chain network configurations that minimizes the maximum latency between a server and client. This can be advantageous in situations in which the server-client connection cost is high but the client-client connection cost is low.

The formulation of min-max MDVRP is as follows, with the assumption that all points are randomly and uniformly distributed in a square:

$$\begin{aligned} \text{minimize} \quad & \lambda \\ \text{subject to} \quad & TSP(S_i) \leq \lambda, \ \forall i \\ & \cup S_i = \mathcal{N}, \end{aligned}$$

where $\mathcal{N}$ is the set of all customers, $S_i \subset \mathcal{N}$ is the subset of customers assigned to vehicle i, and $TSP(S)$ is the minimal TSP tour-length to visit all customers in set S.

Our implementation results demonstrate the efficiency of the heuristic methods developed in this paper, as they reduce MDVRP to a sequence of smaller-scale TSP problems in polynomial time. In addition, the tour length sums that our methods generate are highly comparable to the minimized total lengths produced by traditional approaches, and our methods are also capable of quickly processing tens of thousands of customers, a scalability property which is becoming increasingly important as networks expand in size.

This paper is organized as follows: first, we give a theoretical analysis for the optimal solution of min-max MDVRP by developing lower and upper bounds. Then, by analyzing a region partition algorithm, we show that the optimal solution to MDVRP is asymptotically close to the optimal TSP tour of all customers divided by the number of depots constrained by a large customer-to-depot ratio.

Next, by making use of the fact that a convex equitable region partition yields an even division of points (i.e. if we divide the service region into a set of subregions with equal area, then each subregion will contain – asymptotically – the same number of points), we propose a fast approximation algorithm to generate good initial solutions for min-max MDVRP. It can then be improved with common local improvement procedures. Moreover, this region partition method for which we demonstrate both theory and implementation is potentially useful in many network design problems with regular topology structures, such as the Steiner tree and the minimum spanning tree problems.

We also implement a second approach to min-max MDVRP. By starting from a simple linear programming-based algorithm, we rapidly assign customers to depots. Then we generate TSP routes for each depot. By improving certain *global* parameters, the algorithm can efficiently generate feasible routes. In the performance analysis section, we compare the performances of these two algorithms and a routine heuristic. Finally, we summarize our results and provide a discussion of their significance.

2 Theoretical Analysis

2.1 Analysis of Optimal Solutions

Since MDVRPs are NP-hard, a great amount of efforts have been made to design heuristics while exploring the theoretical bounds is intriguing, too. The paper by Baltz et al. proved the sum of tour lengths asymptotically approaches $\alpha_k n^{\frac{d-1}{d}}$ asymptotically with the uniform distribution, and α_k depends on the number of depots. In this section, we provide a lower and upper bound for min-max MDVRP in a planar graph, and prove its asymptotic convergence for a broad class as the problem size expands. Before starting the work, we will review known results for the probabilistic TSP. The most celebrated discovery is the BHH theorem (1959) by Beardwood, Halton and Hammersley.

Theorem 1. *Suppose X_i's, $i \geq 1$, are independent and identically distributed random points uniformly distributed in a unit cube $[0,1]^d, d \geq 2$. Then with probability one, the length $TSP(X_1, X_2, \cdots X_n)$ of an optimal TSP tour traversing all points has the following property:*

$$\lim_{n\to\infty} TSP(X_1, X_2, \cdots, X_n)/n^{\frac{d-1}{d}} = \alpha(d),$$

where $\alpha(d)$ is a positive constant depending on d.

Considering that we are only interested in planar graphs, in our discussion we define $\alpha \equiv \alpha(2)$. Denote the set of depots by $\mathcal{D} = (D_1, D_2, \cdots, D_m)$, $|\mathcal{D}| = m$. Denote the set of vehicles by $\mathcal{V} = (V_1, V_2, \cdots, V_k)$, $|\mathcal{V}| = k \geq m$. Denote the set of nodes by $\mathcal{N} = (N_1, N_2, \cdots, N_n)$, $|\mathcal{N}| = n$. Denote the optimal TSP tour for a set S of points by $TSP(S)$. Denote the optimal value of min-max MDVRP by L, i.e., L is the length of the longest tour. Denote the largest distance between an arbitrary pair of points in two different sets A and B by $L_m(A, B)$, i.e., $L_m(A, B) = \max_{x\in A, y\in B} ||x - y||$. Then we have the following theorem:

Theorem 2. *For a general planar graph (points do not necessarily follow any distribution):*

$$\frac{TSP(\mathcal{D}\cup\mathcal{N}) - TSP(\mathcal{D})}{k} \leq L \leq \frac{TSP(\mathcal{N})}{k} + 2 * L_m(\mathcal{D}, \mathcal{N})$$

.

Proof. First we prove the lower bound holds.

Consider an optimal pattern of min-max MDVRP. Assume in this pattern the total length of all the tours is S_{opt}. Noticing that L is the longest tour, we have

$$k * L \geq S_{opt}.$$

Add one optimal TSP tour T for all the *depots* in the graph. Now each point(node or depot) in the graph has an even degree, which implies an Euler tour. This Euler tour can be reduced to a feasible TSP tour for the set of all the depots and nodes. Thus:

$$S_{opt} + TSP((D)) \geq TSP(\mathcal{D} \cup \mathcal{N})$$

.

Therefore:

$$L \geq \frac{TSP(\mathcal{D} \cup \mathcal{N}) - TSP(\mathcal{D})}{k}$$

.

For the upper bound, consider a tour partition heuristic:

1. Generate an optimal TSP tour for nodes.
2. Partition this tour into k equal subtours. Assume they are $t_1, t_2, \cdots, t_k$.
3. For each t_i, connect both the starting and ending node to vehicle V_i.

This generates a feasible solution for the MDVRP. The maximal tour-length, which is an upper bound of L, is at most

$$\frac{TSP(\mathcal{N})}{k} + 2 * L_m(\mathcal{D}, \mathcal{N}).$$

□

For the lower bound, the inequality holds even we change L to the average length of all the tours. So this lower bound is not very strong. However, in the underlying corollary, this estimation is still able to provide us with the possibility to predict the behavior of an optimal solution. Assuming nodes and depots are uniformly (randomly) distributed in a unit square, we can derive the following corollaries when the data size becomes large.

Corollary 3. *(a) When* $k = o(\sqrt{n})$,

$$\lim_{n\to\infty} \frac{L}{\sqrt{n}/k} = \alpha.$$

(b) For non-fixed MDVRP, i.e., a vehicle can return to an arbitrary depot instead of its originated depot, if $n = \Omega(k \log^{\frac{3}{2}} k)$ *and* $k = \Theta(m)$,

$$\lim_{n\to\infty} \frac{L}{\sqrt{n}/k} = \alpha.$$

Proof. (a) We only consider the case $m = \Omega(1)$ (the case $m = O(1)$ is trivial). In this case, when n, m are sufficiently large, for arbitrarily small ϵ,

$$\frac{\alpha((1-\epsilon)\sqrt{m+n} - (1+\epsilon)\sqrt{m})}{k} \leq L \leq \frac{\alpha(1+\epsilon)\sqrt{n}}{k} + 2L_m(\mathcal{D},\mathcal{N}) + \epsilon.$$

Noticing that $m \leq k$ and $k = o(\sqrt{n})$, we know $TSP(\mathcal{D})/TSP(\mathcal{D} \cup \mathcal{N})$ goes to zero as n goes to 0. On the other side, $L_m(\mathcal{D},\mathcal{N})$ is bounded by a constant while $\sqrt{n}/k$ goes to infinity.
(b) Using the inequality above, similarly, for the left side, $TSP(\mathcal{D})/TSP(\mathcal{D} \cup \mathcal{N})$ approaches zero as n goes to 0.
For the right side. instead of $L_m(\mathcal{D},\mathcal{N})$, we consider the min-max perfect matching problem for depots and both endpoints of subtours in the tour partition heuristic.
The min-max perfect matching for uniformly distributed points has the bound provided by Leighton and Shor in 1989 [16]: X_i's and Y_i's are independent and uniformly distributed points on a unit square for $1 \leq i \leq r$, then there exists a constant C, such that:

$$\min_{\sigma \in \mathcal{P}} \max_i ||X_i - Y_{\sigma(i)}|| \leq Cr^{-\frac{1}{2}} (\log r)^{\frac{3}{4}}$$

with high probability, where $\mathcal{P}$ is the set of all permutations of $\{1, 2, \cdots, r\}$.
First apply this theorem to depots and all starting points of subtours. Then apply it again to depots and all ending points of subtours. We get $L \leq \frac{\alpha(1+\epsilon)\sqrt{n}}{k} + C * k^{-\frac{1}{2}} (\log k)^{\frac{3}{4}}$. With $n = \Omega(k \log^{\frac{3}{2}} k)$, the desired inequality can be derived.

□

2.2 Bounds by a Region Partition Heuristic

From the discussion above, we conclude that the optimal solution to min-max MDVRP with uniform distributed points will numerically approach $\alpha\sqrt{n}/k$, the optimal TSP tour-length of nodes divided by the number of vehicles, under the constraint $k = o(\sqrt{n})$. This matches the natural intuition, although the constraint is restrictive. We would like to relax this constraint to a broader class and derive some nontrivial bounds by analyzing local performance in large size problems. In this section we will present a region partition heuristic with theoretically good performances. We also make an estimation on upper bounds for all cases $k = \Omega(\sqrt{n})$ by a grid region partition.
Before presenting the algorithm, we need several lemmas to review some facts:

Theorem 4. *(Convex Region Partition Theorem) Given k points in a convex bounded planar polygon, it is always possible to find a partition of the domain into k equal-area convex polygons, with exactly one point in each face.*

This convex region partition theorem was proved in [6] for both continuous and discrete versions and an algorithm for discrete version was also given. However, an algorithm for continuous version, i.e., a convex region partition in the theorem, has not been found until Carlsson and Armbruster

proposed it now [4]. In this paper, we propose a 3/2-approximation algorithm with time complexity $O(m^2 \log m)$, where m is the sum of the number of depots and sides of the polygon.
We will present the algorithm in next section. For the time being, we assume an equitable partition exists and develop a theoretical rationale for the desirability of such a partition.

Lemma 5. *(Occupancy Lemma) [19] Randomly throw n balls into m bins randomly with equal probability. If $n = \Omega(m \log^2 m)$, then the number of balls in the bin holding the most balls has an asymptotic performance as $\frac{n}{m}$.*

Proof. This fact can be proved by the Chernoff inequality. We will give a sketch of the proof here. For any $0 < x < 1$, the inequality $x - (1+x)\ln(1+x) \leq -\frac{x^2}{2} + \frac{x^3}{3}$ holds by checking Taylor expansion, which is equivalent to

$$\frac{e^x}{(1+x)^{1+x}} \leq e^{-\frac{x^2}{2}+\frac{x^3}{3}}.$$

Assume $n = m \log^2 m$. Consider one arbitrary bin, define x_i is 1 if the ith ball falls into this bin, and 0 otherwise. Let $S_i = x_1 + x_2 + \cdot + x_i$. Note that the expected number of balls falling into a bin is $\log^2 m$, from Chernoff inequality, we know

$$P = Pr(S_n \geq (1+\delta)\log^2 m) \leq [\frac{e^\delta}{(1+\delta)^{1+\delta}}]^{\log m}.$$

Let $\delta = \sqrt{\frac{2}{\log m}}$; we may ignore the item of δ^3 when m is large, without hurting our argument, as we can use a small perturbation on the constant of the item of δ^2 instead. Applying the inequality above to Chernoff inequality, we find that $P \leq \frac{1}{m^2}$.
Therefore, the probability that one bin has at least $(1+\sqrt{\frac{2}{\log m}})\log^2 m$ balls is at most $mP \leq \frac{1}{m}$. □

Simple scaling arguments show that BHH theorem still holds even if the unit cube is replaced by an arbitrary compact subset K in R^d. This fact suggests that the limit is independent of the shape of the compact set K.

Lemma 6. *[23](generalized BHH theorem) In particular if $X_i, i \geq 1$, are i.i.d with uniform distribution on a compact set K of Lebesgue measure one, then the limit in BBH theorem holds and is still $\alpha(d)$ almost surely.*

Now let's consider a simple region partition algorithm for min-max MDPVRP in the case $k = m$.

Algorithm 1 A region partition algorithm for min-max MDVRP.

1. Divide the region into k equitable convex polygons such that each polygon have exactly one vehicle inside.
2. Find an optimal TSP tour for each vehicle and the nodes in the same subregion.

With the fixed number of depots, similar to the discussion in last section, we have

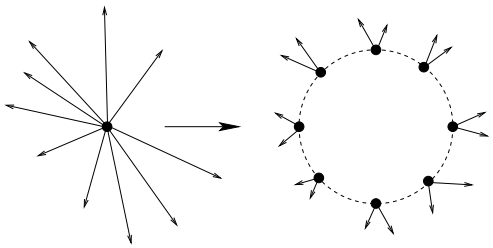


Figure 1: The perturbation method, and its subsequent effects on assignment and routes generation. Here the routes are suggested by arrows emanating from each depot node.

Lemma 7. *If k=O(1),*

$$\lim_{k\to\infty} \frac{L}{\sqrt{n}/k} = \alpha.$$

Proof. Assume the region with the most nodes has $N_{max} = n/k + t$ nodes. From the Occupancy lemma, we know with high probability, $\lim_{k\to\infty} \frac{t}{n/k} = 0$. Therefore, from the generalized BHH theorem, we know:

$$L \leq \alpha(1+o(1))\sqrt{N_{max}}\sqrt{\frac{1}{k}} \leq \alpha(1+o(1))\sqrt{(1+o(1))\frac{n}{k}}\sqrt{\frac{1}{k}} \leq \alpha(1+o(1))\frac{\sqrt{n}}{k}.$$

On the same hand, since the region with the most nodes has at least n/k nodes. Hence,

$$\lim_{k\to\infty} \frac{L}{\sqrt{n}/k} = \alpha$$

. □

The lemma actually claims that the length of the longest tour generated by this algorithm is asymptotically close to a lower bound when the size of the problem expands, which means this algorithm performs asymptotically optimally. As generated graphs in the implementation of our next algorithm, the LP-TSP algorithm, show, the LP-based approach often produces a good area subdivision. That algorithm also reveals the motivation of the region partition algorithm.

If we have more vehicles than depots available ($k > m$), we can use the same perturbing idea as in the latter LP-based algorithm to generate their routes. By a procedure similar to the LP-based algorithm, if there are two or more vehicles on the same depot, we only need to slightly relocate vehicles evenly distributed on a small circle centering that depot (figure 1).

For a sufficiently small perturbation the algorithm remains feasible and the optimal value won't change greatly.

If the gap between n and m is smaller, for example, $n = O(m \log m)$, it is harder to predict the optimal solution. Motivated by Occupancy Lemma, we still can derive a good upper bound. The following theorem discusses an extreme case:

Theorem 8. *Assume $k = m$, i.e., all vehicles are i.i.d. in the cube; if the number of vehicles is proportional to the number of nodes, i.e., $k = \lambda n$, then with high probability, as $k \to \infty$,*

$$L \leq (\frac{\alpha}{\lambda} + 2\sqrt{2}\log n)\frac{1}{\sqrt{n}}.$$

Proof. We divide the unit square into $n/\log^2 n$ smaller squares each of which has the side length $\log n/\sqrt{n}$. Then from the Occupancy lemma, we know, with high probability, any small square has at most $\log^2 n + O(\log n)$ nodes and at least $\lambda(\log^2 n - O(\log n))$ vehicles. Assume the square having the most nodes is R. Then by tour partition heuristic we used before, we know L is at most the length of the TSP tour of nodes in R divided by the number plus the maximal distance from starting and ending point to one depot in R. By the generalized BHH theorem, we have

$$L \leq \frac{\alpha\sqrt{\log^2 n + o(\log n)}}{\lambda(\log^2 n + o(\log n))}\frac{\log n}{\sqrt{n}} + 2\sqrt{2}\frac{\log n}{\sqrt{n}} \approx (\frac{\alpha}{\lambda} + 2\sqrt{2}\log n)\frac{1}{\sqrt{n}}.$$

. □

Therefore, by the generalized BHH theorem, we can see in the case $m = \lambda n$, the $m = \lambda n$ is the extreme case m reaches the largest value it can have. For the rest case, one can follow the procedure to get an upper bound similarly.

We already have an asymptotic estimation for the case $k = o(\sqrt{n})$, and applying the similar argument to the preceding theorem we have the following result for the case when $k = \Omega(\sqrt{n})$:

Corollary 9. *Assume all vehicles are uniformly distributed in the cube and $k = \Omega(\sqrt{n})$, then with high probability, as $k \to \infty$,*

$$L \leq \frac{\alpha\sqrt{n}}{k} + 2\sqrt{2}\frac{\log k}{\sqrt{k}}.$$

Many network structures, like minimum spanning trees and Voronoi graphs, have similar conclusions to that of the travelling salesman tour, i.e. their lengths converge asymptotically in a manner similar to the BBH theorem. Region subdivision is therefore a potentially powerful tool not only in vehicle routing but also in many network applications. It is worth extra effort to investigate how to implement subdivision quickly. Paper [6] gave the proof for the existence of a convex region subdivision and an algorithm with runtime $O(N^{\frac{4}{3}}\log N)$ for the discrete version in which the dividing subjects are red and blue points. Noticing that an area can be approximated within any ϵ factor by taking enough sampling points, we can implement their algorithm to get a near optimal convex subdivision. We have also been looking for a quick method to realize it. The current result is that we can realize a ham sandwich cut for two arbitrary polygons(discrete or continuous) in $O(n\log n)$ time, which motivates us to find a $\frac{3}{2}$-approximation algorithm in $O(mn\log m)$ time, m is the sum of the number of nodes and sides of the initial polygon.

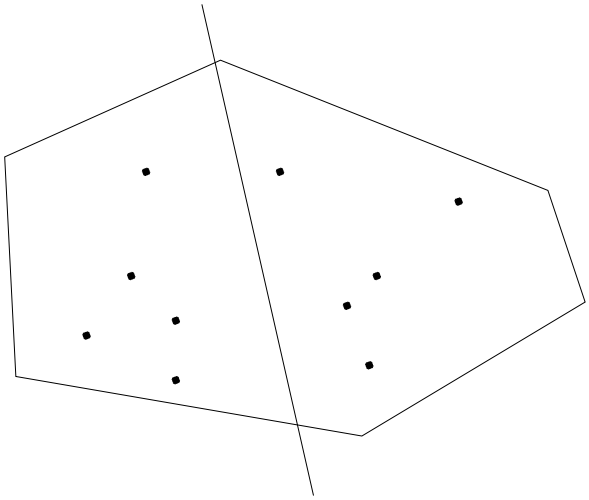


Figure 2: A mixed ham-sandwich cut.

3 Convex Region Partition

As our first heuristic, we describe an algorithm that takes as input a convex planar region C and a set of n points $X = x_1, \ldots, x_n$, and outputs a partition of C into n convex subregions $C_1, \ldots, C_n$, satisfying the property that

$$\frac{\max_i (\text{Area}\, C_i)}{\min_i (\text{Area}\, C_i)} \leq 3/2$$

Note that if we have an optimal solution to the problem, then all of the C_i have equal area, so we have

$$\frac{\max_i (\text{Area}\, C_i)}{\min_i (\text{Area}\, C_i)} = 1$$

so that we can treat $\max_i (\text{Area}\, C_i) / \min_i (\text{Area}\, C_i)$ as an objective function which is obviously bounded below by 1. We make a weak assumption that the points X lie in *general position*, i.e. that no three points are collinear.

3.0.1 Ham-Sandwich Cuts

A well-known geometric result is the *ham-sandwich* theorem, which says that any two regions in the plane can be simultaneously broken into two regions of equal area by a single line. In particular, a region C and a set of points P of even cardinality can be simultaneously bisected by a single line. [3] gives an algorithm for finding such a line(see figure 2).
If the number of points n satisfies $n = 2^k$, $k \in \mathbb{Z}$, then we can find a partition of C into n regions of equal area each containing a single point by recursively taking mixed ham-sandwich cuts, as shown in figure 3. For general n, it is not hard to show that taking recursive ham-sandwich cuts (rounded to the nearest even number) gives a 2-approximation to a convex region partition.

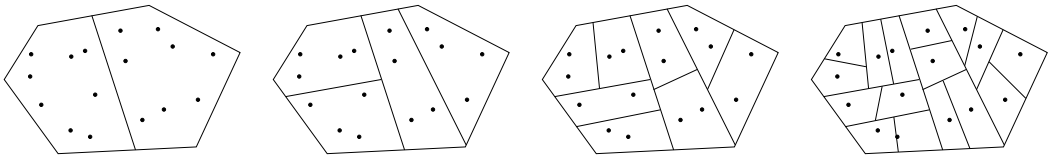


Figure 3: Recursive ham-sandwich cuts for the case $n = 16$.

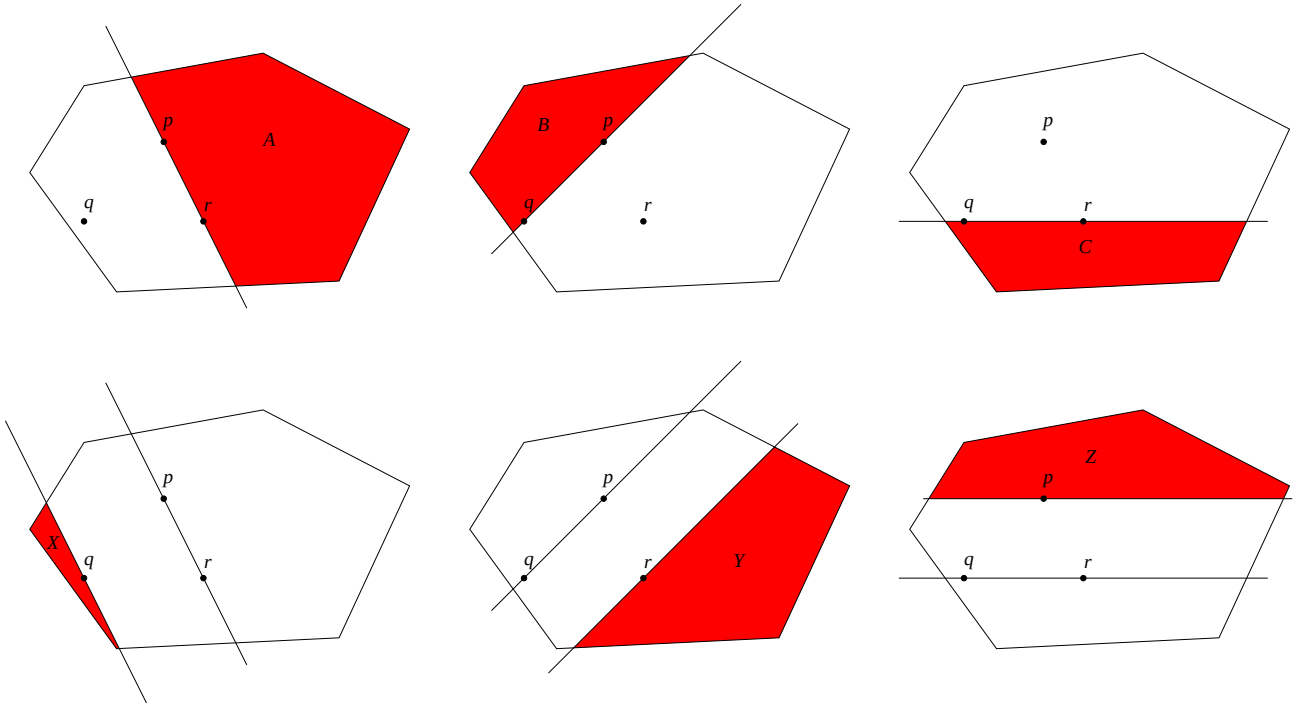


Figure 4: Setup for solving the $n = 3$ instance of CPP.

3.0.2 Extensions

We can improve our approximation ratio by explicitly finding a convex region partition for the case $n = 3$. We can obtain a 3/2-approximation to CPP if we can explicitly find a solution for the case $n = 3$ by performing recursive ham-sandwich cuts and solving the case $n = 3$ if it is needed. In this section we discuss the solution.

Given a polygon with area $A = 1$ containing three points p, q, and r, consider the subregions A, B, C, X, Y, Z shown in figure 4. Assuming none of the subregions has area exactly equal to 1/3 or 2/3, one of the following must be true:

1. All regions A, B, C, X, Y, Z have area less than 1/3.
2. There exist two overlapping regions, one of which has area less than 1/3 and one of which has area greater than 1/3.
3. All regions A, B, C, X, Y, Z have area between 1/3 and 2/3.

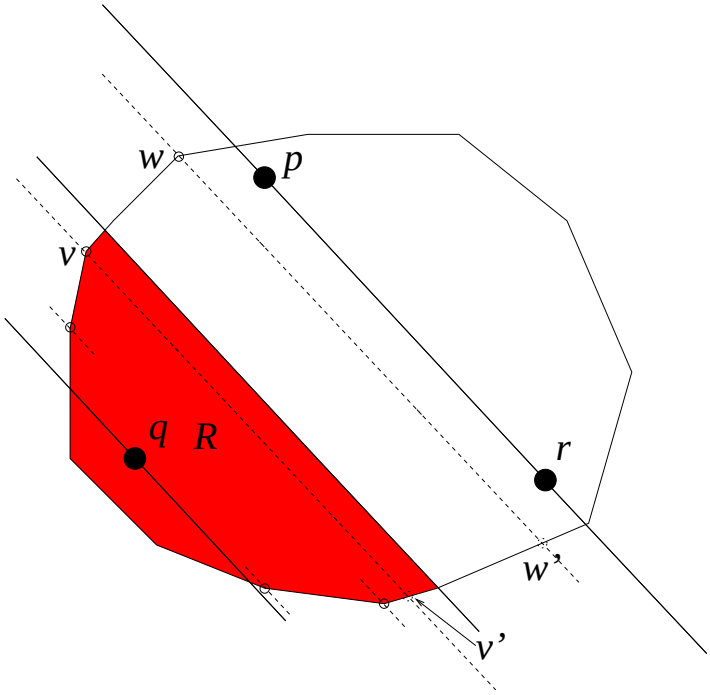


Figure 5: Case 1. After sorting the vertices lying between ℓ_q and ℓ_{pr} (indicated with bubbles in the diagram) we find that the line for which the area of R is 1/3 lies between vertices v and w. We form the quadrilateral $\left\{v, w, v', w'\right\}$ and solve analytically.

If the vertices of C are sorted in clockwise order, then we can clearly determine which of the above cases holds for a given instance in $\mathcal{O}(k)$ time, where k is the number of vertices of C. We consider each case separately below. Note that in cases 1 and 2, it suffices to find a single sub-region with area 1/3 that contains one of the three points.

Case 1 By assumption, A and X have area less than 1/3. Let ℓ_q denote the line parallel to ℓ_{pr}, i.e. the line defining region X. There must exist a line ℓ^* lying between ℓ_q and ℓ_{pr} (parallel to both of them) that defines a subregion R with area exactly equal to 1/3. In order to determine this line analytically, let V denote the set of vertices of the polygon lying between ℓ_q and ℓ_{pr}. We can sort the elements of V based on their distance to line ℓ_q. After performing a binary search we can conclude that ℓ^* must lie between two particular points v and w, on the boundary of the polygon, that share an edge (which may be vertices of the polygon). We can therefore reduce the problem to that of cutting off a portion of a quadrilateral $\left\{v, w, v', w'\right\}$ with a line of fixed slope, which can be accomplished in constant time.
This case is illustrated in figure 5.

Case 2 Without loss of generality, assume that A and B are the overlapping regions in question, with $\text{Area}\, A < 1/3$ and $\text{Area}\, B > 1/3$. Let V denote the (sorted) set of all vertices lying in A or

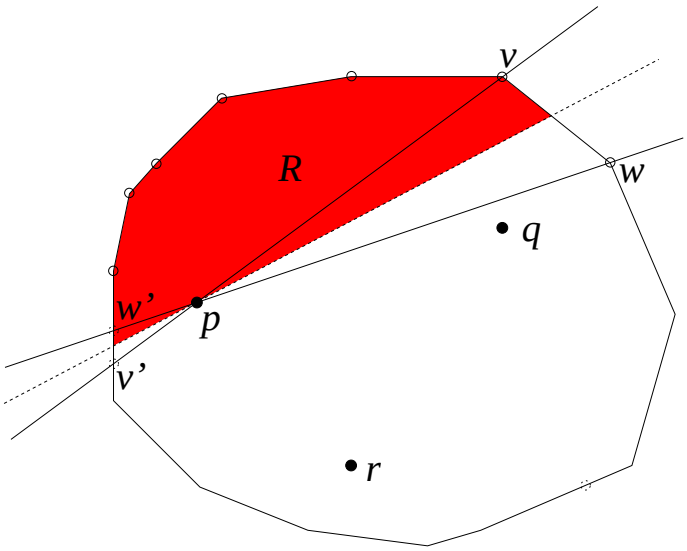


Figure 6: Case 2. After sorting the vertices lying in regions A or B (indicated with bubbles in the diagram) we find that the line for which the area of R is $1/3$ must intersect the boundary of the polygon somewhere between vertices v and w. We form the hourglass shape $\left\{v, w, p, v', w'\right\}$ and solve analytically.

B. By the intermediate value theorem there must exist a line ℓ^* through point p that defines a subregion R with area exactly equal to $1/3$. We will determine this region analytically by performing a binary search on the vertices $v \in V$. After the search we can conclude that ℓ^* must intersect the polygon somewhere between two points v and w, on the boundary of the polygon, that share an edge (which may be vertices of the polygon) . We can therefore reduce the problem to that of cutting off a fixed portion of an hourglass shape $\left\{v, w, p, v', w'\right\}$ with a line through point p, which can be accomplished in constant time.
This case is illustrated in figure 6.

Case 3 Without loss of generality, assume that we can project point p orthogonally onto side qr (if this is not possible we merely change the labels of the vertices). Let point m be the bisector of side qr.
We first desire a sector R based at m containing p with area $1/3$ for which line ℓ_{qr} is a supporting hyperplane, as shown in figure 7. By assumption, such a sector must exist, because $\text{Area}\, Z > 1/3$. We can find such a sector using algorithm 2.
Let w' be the point other than w at which line ℓ_{mw} intersects the boundary of the polygon. Note that by our choice of m, line ℓ_{mw} must have q and r on opposite sides. Let S be the sub-region defined by ℓ_{mw} that contains r. If $\text{Area}\, S > 1/3$, then by the original assumption we can perform a binary search to obtain a (unique) sector based at m, one of whose sides is segment mw, containing r with area $1/3$. If $\text{Area}\, S < 1/3$, then we essentially have to solve Case 2: by the intermediate

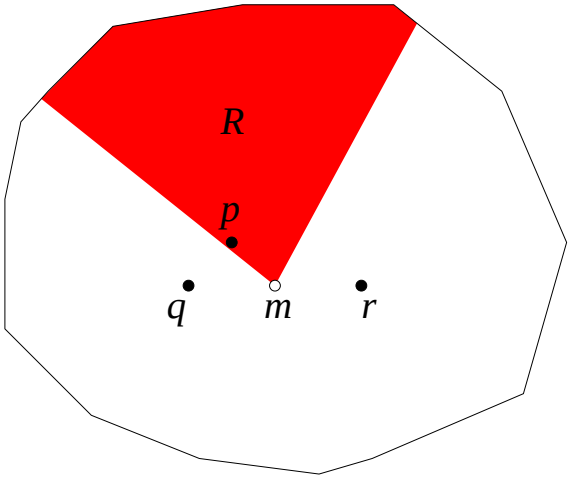


Figure 7: Case 3. By assumption, since $1/3 < \text{Area}\, Z < 2/3$, sector such as R exists.

Algorithm 2 Finding a sector based at m containing p with area $1/3$

1. Let s be the point where line ℓ_{mp} intersects the boundary of the polygon. Let x and y be the points at which ℓ_{pq} intersects the boundary of the polygon. Let t and u be the two vertices that share an edge with s that lie on the same side of ℓ_{pq} as s (it may be that $t = x$ or that $u = y$).

2. Compute the area A_{mtu} of triangle $\triangle mtu$. If $A_{mtu} > 1/3$, determine (non-unique) v and w lying on line segment tu analytically so that $A_{mvw} = 1/3$ and $p \in \triangle mtu$ and terminate, setting $R = \triangle mtu$. Otherwise, go to step 3.

3. Using the fact that $A_{mtu} < 1/3$ and that $\text{Area}\, Z > 1/3$, perform a binary search to determine (non-unique) v and w that define a sector R containing p with area $1/3$ and terminate.

Algorithm 3 A 3/2-approximation to CPP.

1. If $n = 1$ (i.e. only one point), assign x_1 to C and terminate. If $n = 3$, solve the problem explicitly using the case-by-case analysis and terminate.

2. Let ℓ be a mixed ham-sandwich cut that breaks C in half and evenly divides the points x_i (if n is odd, ignore one point chosen at random). Let X_1 be the set of points assigned to C_1 and X_2 the set of points assigned to C_2.

3. Apply steps 1 and 2 recursively to C_1, X_1 and C_2, X_2.

value theorem there exists a line going through point m that defines a region containing r with area exactly equal to 1/3.

The complete regional subdivision algorithm is described in algorithm 3. Noting that each case above can be solved in running time $\mathcal{O}(k \log k)$, where k is the number of vertices of the polygon, because the most expensive operation needed is a binary search whose objective function is the area of some sub-region of the polygon. Since we can compute a mixed ham-sandwich cut in $\mathcal{O}(m \log m)$, where $m = n + k$, and since we can introduce no more than one additional vertex to each sub-region at each step of the subdivision algorithm, we can conclude that the running time of algorithm 3 is $\mathcal{O}(mn \log m)$.

4 A Min-Max MDVRP Heuristic

In this section we present a two-stage LP-based heuristic, which uses *global* adjustment during post-refinement periods. Initially, we assign each node to exactly one depot; next, we simultaneously find TSP tours based on the assignment and minimize the maximal length of any single tour (*makespan*). We assume all points are uniformly distributed on a square and the locations of depots are randomly distributed. A depot is allowed to have more than one vehicle. Each vehicle must return to the depot from which it originates.

Although many algorithms have been applied to solve SDVRP, few have focused on MDVRP. We are unaware of any existing specific heuristic or theoretical results for min-max MDVRP, especially in the large-size case. For general literature, one main idea is a 2-stage heuristic by Wren and Holliday [22]: construct an initial solution, then apply a number of local improvements. The initial solutions to these algorithms are constructed quite naively: nodes are often assigned to their closest depots at the first stage. Gillet and Johnson [12] proposed a clustering heuristic which used sweep heuristic at each depot. Golden et al. [14] presented two heuristics for MDVRP; the second heuristic is to specifically solve the large-size problem by a 2-stage algorithm which assigns nodes to depots first and then built TSP tours for each vehicle. Chao et al. proposed a simple initialization heuristic followed by a refinement, which gained the best performance in many benchmark problems [8]. In

2002, Giosa et al. proposed a series of 2-stage heuristics to solve MDVRPTW [13]. Recently, Andrew Lim and Fan Wang [17] gave a one-stage approach to MDVRP with the constraint each depot only has fixed number of vehicles.

Our heuristic is based on *load balancing*: the load for each vehicle must be almost balanced in an optimal plan, so each vehicle is assigned the same working load, such as the number of nodes to serve. This natural guess is plausible considering that the distribution of nodes in our model is uniform. We carefully balance the number of nodes assigned to each depot by linear programming at the first step, then generate a near-optimal route for each vehicle by using the Concorde TSP solver [9]. The solution is also further improved by global adjustment of the number of nodes assigned to depots, rather than traditional local procedures to improve local tours.

The terms and heuristic are described as follows:

First assume each depot has exactly one vehicle, and no two depots share a location (this assumption will be generalized later). Suppose each node $X_j, 1 \leq j \leq n$, is located at (x_j, y_j), and each vehicle $V_i, 1 \leq i \leq k$, is located at (v_i, w_i). The distance c_{ij} between X_j and V_i is $||(x_j, y_j) - (v_i, w_i)||$.

Define x_{ij} to be a binary variable indicating if node X_j is assigned to vehicle V_i or not. We build the underlying linear program:

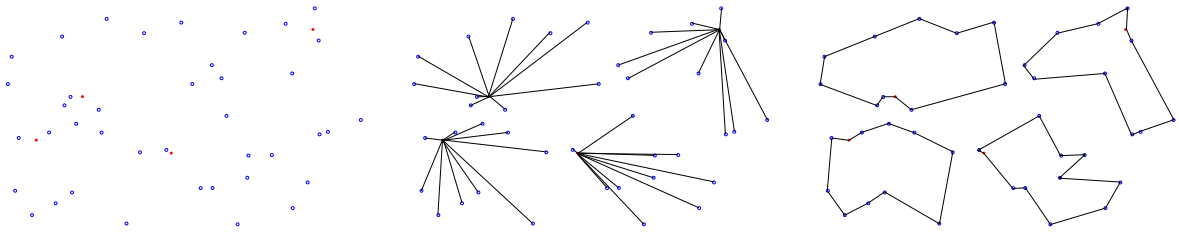
$$
\begin{aligned}
(LP): \text{maximize} \quad & \sum_{1 \leq i \leq k, 1 \leq j \leq n} c_{ij} x_{ij} \\
\text{subject to} \quad & \sum_{1 \leq j \leq n} x_{ij} = \frac{n}{k}, \ \forall i, \\
& \sum_{1 \leq i \leq k} x_{ij} = 1, \ \forall j, \\
& x_{ij} \geq 0, \ \forall i, j,
\end{aligned}
$$

where, without loss of generality, we assume that n/k is an integer.

A sketch of the heuristic is as follows:

Algorithm 4 A Load balancing algorithm for min-max VRP.

1. Build a linear program as above and solve it.
2. Build a TSP tour for each vehicle by the Concorde solver. Assume that their length is sorted in a descending order $\{\ell_1, \cdots, \ell_k\}$.
3. If we have $(\ell_1 - \ell_k)/\ell_k \leq r$, with r a small constant, or if we have run the integer program too many times, we stop and output a current best solution. Otherwise, we decrease the number of nodes assigned to vehicles having longer tours and increase the number of nodes assigned to vehicles having shorter tours (we will explain details later). This way, we generate a new revised LP. Return to step 1 and re-run the algorithm.



(a) (b) (c)

Figure 8: The initial input to the algorithm (a), the assignment (b), and the final construction of TSP tours (c).

A sketch of the algorithm in one loop is shown in figure 8.
There are several details we need to explain in our algorithm.
Note 1: At step 1, the LP program is a typical network flow model, so that any vertex solution to the program will be integral when the number of nodes on the right-hand-side are integers.
Note 2: r in step 3 is a ratio threshold to measure the output, which can be adjusted according to the need to control accuracy. In most cases, setting $r = 25\%$ is enough to achieve a satisfactory result.
Note 3: In step 3, if $(\ell_1 - \ell_K)/\ell_K \leq r$, we adjust the number of nodes assigned to each depot by the following procedure:

- Build the sets $L^+ = \{i : L_i - \bar{L} > 0\}$, $L^- = \{i : L_i - \bar{L} < 0\}$, where $\bar{L}$ is the average of tour length.
- Consider the set that has fewer elements. Without loss of generality, assume it is L^+, and $s = |L^+|$, then for any $i \in L^+$, define the *relative drift* $d_i^+ = \frac{L_i - \bar{L}}{\bar{L}}$. Decrease the number of nodes assigned to depot i by $[c * \frac{d_i}{r}]$(c is a positive constant). Sort elements in L^- in the descending order of their tour lengths. Then define $d_i^- = \frac{\bar{L} - L_i}{\bar{L}}$ for the first s elements in L^-. Increase the number of nodes in the same way.
- If the number of nodes removed from L^+ is more than the number of nodes added to L^-, add one node to each element in L^- sequentially starting from the depot which has the fewest nodes, keep doing that until nodes are balanced. Do the same procedure for the case the number of nodes removed from L^+ is less than the number of nodes added to L^-.

Note 4: The parameters r, c are empirically decided. We set $c = 2$ here. The numerical values we choose to use here are based on implementation results.

5 Performance and Analysis

There are no specific benchmark problems for min-max MDVRP in available literature. Most available MDVRP benchmarks have time window constraints, and their sizes are generally small, varying between $50 \sim 1000$(see [20]). Thus, we generated 12 MDVRP instances ourselves, and the results are reported in this section. These instances vary in size with respect to the number of depots (2 to 32) and to the number of nodes (100 to 2048), and the depot and node locations were randomly generated. Note that the data presented represent the average values obtained from repeating each experiment five times. Tests were conducted on a Pentium 4 1.8GHz/512 MB notebook. The implementation was done in Matlab and makes use of the COPL LP Solver and the Concorde TSP solver [9].

We first compare the performance of three heuristics after one iteration in Table 1 and 2: the LP-TSP approach, region partitioning and a traditional approach. The algorithm for the traditional approach can be specified as follows:

Algorithm 5 A nearest neighbor clustering and routing algorithm.

1. Assign each point to a single depot by the nearest neighbor method.
2. Find a TSP route for each cluster; get an initial solution for the problem.
3. Run the local improvement method to improve the solution.

We have also implemented a modified version of a local improvement procedure proposed by Chao, Golden and Wasil [8] called "1-point movement". In Table 3 and 4, we compare the performances of three heuristics. The first heuristic uses a LP based load balancing technique to distribute the load evenly amongst vehicles. The second heuristic starts with the same initial assignment as our heuristic and improves the solution using the modified 1-point movement method. The third heuristic starts from the nearest neighbor assignment and uses the modified 1-point movement improvement method.

The running time for the LP-TSP approach is primarily comprised of the execution time of the LP and TSP solvers. We use sparse matrices with the LP solver, which makes the execution much faster. For small to medium sized benchmarks, the main bottleneck is finding a near optimal TSP tour for each depot using the Concorde solver. As the number of vehicles for a given set of nodes increases, the running time for Concorde decreases, as there are fewer average number of nodes per TSP tour. For a smaller number of depots, this decrease surpasses the increase in time caused by the new, larger LP.

We can make several observations from the above tables. First, we can observe from Table 1 and 2 that both LP-TSP and region partition algorithms perform better than the traditional nearest neighbor approach. They are thus efficient and effective candidates for the initial solution.

Second, the LP-TSP with load balancing algorithm for large data sets generates the best solutions

Table 1: Summary of First Iteration Solutions for Three Heuristics

			LP-TSP			
	Depots	Nodes	Max	Min	Mean	Time
d2n100	2	100	464.1665	402.0456	433.1061	2.5545
d2n500	2	500	871.4619	842.9678	857.2148	22.961
d4n100	4	100	284.2732	205.7488	238.2282	1.9062
d4n500	4	500	460.8977	418.7652	439.5123	6.7656
d4n1000	4	1000	639.8514	591.98	613.6506	60.3594
d8n200	8	200	215.6308	134.7093	173.3228	3.075
d8n512	8	512	275.4057	207.7543	237.4244	7.5592
d16n256	16	256	145.9767	73.36452	103.5944	4.4876
d16n512	16	512	175.9382	106.4953	133.9108	6.4186
d16n1024	16	1024	204.026	147.1278	171.829	13.9064
d16n2048	16	2048	263.9013	210.0281	233.4694	49.6688
d32n1024	32	1024	137.6043	67.28527	94.68939	16.9215
d32n2048	32	2048	162.1593	104.0153	122.7029	30.164

Table 2: Continued: Summary of First Iteration Solutions for Three Heuristics

Region Dividing				Nearest Neighbor			
Max	Min	Mean	Time	Max	Min	Mean	Time
423.4482	405.3545	414.4013	2.344	515.1542	327.3245	421.2393	1.055
911.4612	790.7493	851.1053	12.6405	960.5076	764.5879	862.5478	11.3595
259.0749	201.4661	235.0596	2.4284	380.0658	92.77714	222.6629	1.1468
495.6325	400.4423	443.8642	11.0874	683.9169	198.0117	436.3651	20.8
714.2934	522.3308	614.3684	54.4968	967.1309	246.7237	606.5448	62.581
236.4221	139.9234	192.4091	8.2568	339.5696	67.13816	157.5154	2.1846
271.2728	225.9595	250.7157	20.1312	432.4621	84.85089	230.0775	6.5344
190.3163	76.68225	122.981	18.7396	196.367	25.85692	93.53754	3.7092
219.5802	113.4607	152.504	32.2056	244.1982	47.38766	123.5722	5.4874
227.1616	151.2489	184.0757	56.9036	324.0938	61.25875	163.9054	17.3968
285.9507	215.0933	244.4174	115.6698	535.0106	62.50485	225.7516	74.3128
182.5517	77.5453	110.9795	125.4785	209.842	23.38633	86.89184	11.8745
219.1573	109.1832	143.5964	242.655	285.9568	28.65197	117.1062	37.265

Table 3: Summary for Improved Heuristics

			LP-TSP+local Improvement				LP+Load Balancing			
	Depots	Nodes	Max	Min	Mean	Time	Max	Min	Mean	Time
d2n100	2	100	436.6301	433.1008	434.8655	7.9295	458.7227	402.0542	430.3884	9.172
d2n500	2	500	868.9795	865.5071	867.2433	138.7655	870.9441	841.7714	856.3578	69.6325
d4n100	4	100	264.0869	244.7366	255.2331	9.1062	260.2228	222.3239	240.8577	10.703
d4n500	4	500	456.3324	427.265	446.9734	66.6972	461.0018	418.0355	439.2781	65.8876
d4n1000	4	1000	636.5408	615.1136	629.4529	453.3094	638.6942	591.6303	613.6019	447.1312
d8n200	8	200	203.1164	148.3161	183.7135	19.3502	186.8439	154.7289	171.73	21.7656
d8n512	8	512	264.8996	226.2448	248.5112	63.1908	263.495	214.4003	237.3148	63.8188
d16n256	16	256	131.8973	75.03237	106.9231	31.9628	111.8126	89.28733	101.9054	34.8436
d16n512	16	512	164.1278	108.4291	141.9349	48.1784	145.8283	117.9964	132.2481	51.3344
d16n1024	16	1024	199.5094	149.6246	177.6315	89.8844	193.1127	154.2951	172.1586	106.5252
d16n2048	16	2048	261.9899	212.189	241.7618	415.3404	254.485	210.9747	232.6723	467.6626
d32n1024	32	1024	128.3784	66.37938	96.99964	96.344	101.0706	79.37021	92.39999	110.0465
d32n2048	32	2048	154.8267	103.1623	127.977	232.5	140.2472	111.9905	123.1741	228.3045

Table 4: Continued: Summary for Improved Heuristics

Region Dividing+Local Improvement				Nearest Neighbor+Local Improvement			
Max	Min	Mean	Time	Max	Min	Mean	Time
420.2927	418.882	419.5874	8.156	471.5656	427.8247	449.6951	10.14
868.9259	865.2076	867.0667	118.9685	893.0568	890.6041	891.8305	69.1325
249.4282	235.6096	244.0173	11.1188	315.1402	145.1332	248.2715	8.956
481.4977	427.9336	460.027	66.0092	573.5623	272.6562	470.2735	80.4152
637.6445	616.4023	630.5434	421.2174	917.552	345.6991	654.6088	788.1972
215.6192	171.7715	203.8352	29.3686	265.0315	76.03288	194.5665	40.4906
283.6925	207.1459	258.8974	69.003	387.5828	111.3567	263.5297	67.7752
173.0854	83.4317	133.4789	49.397	172.055	30.18939	104.9598	31.8874
204.8053	119.2954	171.3088	84.3312	220.886	47.30744	138.4406	43.1096
222.0949	157.2944	199.7946	157.6968	265.5577	69.27293	187.3292	134.4
287.1788	213.3255	255.3882	405.3216	467.5545	86.64765	255.156	554.4
186.8116	75.6875	126.9404	216.4065	185.7216	23.48184	95.35758	92.836
214.9572	108.9897	159.6786	456.586	252.2663	28.26846	130.5024	355.4055

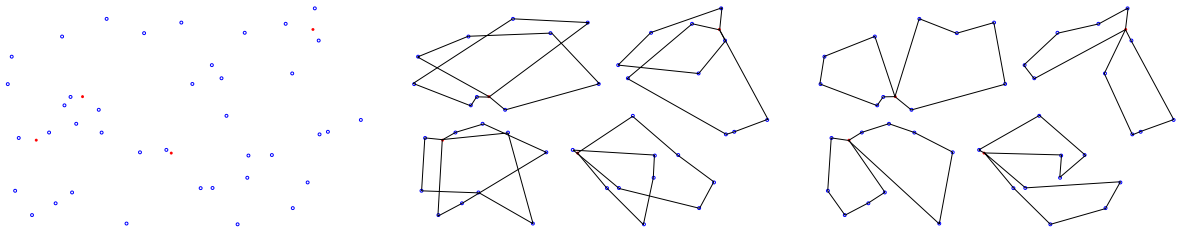


Figure 9: Problems encountered when multiple vehicles are assigned to a single depot. (a) is the initial node set given to the problem (we have 8 vehicles and 4 depots) and (b) represents a solution without the perturbation applied. (c) represents a solution with the perturbation applied; it is clearly preferable, as vehicle routes do not intersect.

while maintaining a competitive running time, as seen in Table 3 and 4. Third, LP-TSP with load balancing not only improves the maximal tour length as compared to the initial solution, but it also does not significantly increase the total tour length, suggesting that the adjustment will not lead to a major global increase in cost. Finally, the region partition algorithm tends to outperform all the other algorithms when the number of depots is small, though its running time will substantially increase for more depots.

In our present instances, we use depots of size 2^k because the 1.5-approximation region partition algorithm gives equitable convex partitions in this case. Recently, an exact solution to the convex region partition has been found [4], which would allow us to effectively use the region partition method on any number of depots.

5.1 Solving a More General Case

We have assumed each depot includes exactly one vehicle in the algorithm above. However, in a practical situation, multiple vehicles may lie on the same depot. We need to extend our algorithm to this general case. If we still follow the original algorithm and assign the same number of nodes to each vehicle instead of each depot, the solution is poor, because nodes are assigned to each such vehicle randomly – therefore, our initial geometric intuition for node assignment is lost (figure 9).

In order to work around this difficulty, for vehicles in a same depot, we perturb the locations of vehicles located at the same depot before generating c_{ij}'s. Each vehicle is relocated to a new location lying in a small circle around that depot , keeping them evenly distributed on that circle. This small change will dramatically improve the final solution (figure 1). We can generate good solutions with this technique.

6 Conclusion

In this paper, we make a first attempt to study min-max MDVRP. We predicted some properties of the optimal solution theoretically, and we proved the near-optimal performance for an algorithm based on computational geometry with a few constraints. In section 3, we discussed the details of the region partition algorithm, which may have applications beyond the vehicle routing problem. An LP-based algorithm with global improvement was implemented to solve very large-sized problems effectively. We concluded by reporting our simulation results.

Many interesting topics remain after this first attempt in min-max MDVRP. For example, from a theoretical point of view, accurate estimation of the optimal solution in large-scale problems for the general case is still open. Our theoretical and implemented work can serve as starting points for further exploration as well.

The techniques used in our algorithms provide advantages over traditional local search methods. For example, since the LP-TSP with load balancing algorithm does not inherently require that the points be scattered in Euclidean space, it can be adapted as a backend to existing map software to solve VRPs with actual geographic locations. The region partition algorithm can be applied to problems outside the scope of VRP as well. An exact convex region partition algorithm will be helpful not only for high-quality initial solutions to MDVRPs, but also for solving a class of network problems with similar structures and a min-max objective function.

References

[1] K. Altinkemer, B. Gavish, Heuristics with constant error guarantees fir the design of tree networks. *Management Science*, 3, 1988.

[2] E.M. Arkin R. Hassin, A. Levin, Approximations for Minimum and Min-max Vehicle Routing Problems. *Journal of Algorithms*, 2004.

[3] B. Armbruster, J. Carlsson. Finding efficient 2d ham sandwich cuts. 2006. Unpublished.

[4] B. Armbruster, J. Carlsson. Finding equitable convex partitions of points in a polygon efficiently. 2006. Unpublished.

[5] A. Baltz, D. P. Dubhashi, L. Tansini, A. Srivastav Soren Werth. Probabilistic Analysis for a Multiple Depot Vehicle Routing Problem. *Foundations of Software Technology and Theoretical Computer Science, 25th International Conference(FSTTCS).* pp 360-371, 2005

[6] S. N. Bespamyatnikh, D. G. Kirkpatrick, J. Snoeyink. Generalizing Ham Sandwich Cuts to Equitable Subdivisions. *Symposium on Computational Geometry,* 49-58, 1999.

[7] J. Carlsson, D. Ge. Methods For Vehicle Routing Problem With Distance Constraints. *Techniquel Report*, ICME, Stanford, 2005.

[8] I.-M. Chao, B. Golden, E. Wasil. A new heuristic for the multidepot vehicle routing problem that improves upon best-known solutions. *American Journal mathematical management Science*, Vol 13, No 3, pp 371-406, 1993.

[9] W. Cook. CONCORDE. Vers. 1.1. Computer software. `http://www.tsp.gatech.edu/concorde.html`

[10] G.B. Dantzig, J.H. Ramser The truck dispatching problem. *Management Science,* 6:60, 1959.

[11] J.F. Soumis, M. Desrochers A New Optimization Algorithm for the Vehicle Routing Problem With Time Windows by Column generation. *Networks* 14, 545-565,1984

[12] B. Gillet, J. Johnson Multi-terminal vehicle-dispatching algorithm. *Omega*, Vol 4, pp 711-718, 1976.

[13] I. Giosa, I. Tansini, I. Viera. New assignment algorithm for the multi-depot vehicle routing problem. *Journal of Operations Research Society*, Vol 53, pp 283-292, 1984.

[14] B.L. Golden, EL. Magnanti, H.Q. Nguyen. Implementing vehicle routing algorithms. *Networks*, 7:113-148, 1977.

[15] M. Haimovich, A. H. G. Rinnooy Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research* 10(1985) 527-542.

[16] T. Leighton, P.W. Shor. Tight Bounds for minimax grid matching with applications to the average case analysis of algorithms. *Combinatorica,* 9, 161-187, 1989

[17] A. Lim, F. Wang Multi-Depot Routing Problems: A Sone-Stage Approach. *IEEE transactions on automation science and engineering*, Vol 2, No 4, Octomber 2005.

[18] C-L li, D. Simichi-Levi, M. Desrochers. On The Distance Constrained Vehicle Routing Problem. *Operations Research* 40, No. 4 (1992), 790-799

[19] R. Motwani, P. Raghavan. Randomized Algorithms. *Cambridge University Press,* June 1995

[20] http://www.top.sintef.no/vrp/benchmarks.html

[21] L. Tansini. Department of Computer Science, Chalmers Univeristy, *PhD Thesis.* To appear.

[22] A. Wren, A. Holliday. Computer scheduling of vehicles from one or more depots to a number of delivery points. *Oper. Res. Q.,* Vol 23, pp. 333-344, 1972.

[23] J. E. Yukich. Probability Theory of Classical Euclidean Optimization Problems. *Lecture Notes in Mathematics,* 1998, vol. 1675.