OPTIMIZATION OF DELIVERIES FROM DISTRIBUTION POINTS

T. Ronkainen*, M.M. Ali† and M. Engelbrecht‡

Abstract

We solve an optimization problem involving the assignment of transport routes to a delivery point. The general and particular cases of the optimization problem are described. An algorithm for the simplified case and ideas for the general case is presented. The problem arose in CaterPlus, a major food catering industry in South Africa.

1 Introduction

Let us assume that we have a large geographical area filled by customers. Each customer has a unique demand for a product we are delivering. We want to organize the delivery process by splitting the customer base into smaller regional units served by *distribution points*. Each distribution point has some capacity of transportation to deliver products to the customers. The problem is to find the location of these distribution points and organize our transportation capacity to deliver products fulfilling the demand of every customer. The total transportation capacity is determined by the fixed number of *trucks*, where each truck can operate near one of the distribution points and can carry products up to its capacity.

Our goal is to minimize the cost of the delivery. For simplicity, we model the customers as points in $\mathbb{R}^2$ and use the total distance traveled by trucks

*School of Computational and Applied Mathematics, University of the Witwatersrand, Private Bag 3, Wits 2050, Johannesburg, South Africa. *e-mail: tommi.ronkainen@gmail.com*

†School of Computational and Applied Mathematics, University of the Witwatersrand, Private Bag 3, Wits 2050, Johannesburg, South Africa. *e-mail: mali@mail.cam.wits.ac.za*

‡CaterPlus, The BIDVest Group Ltd., Johannesburg, South Africa *e-mail: martin.engelbrecht@caterplus.co.za*

as the cost of delivery. For the mathematical formulation, we denote

	q	the number of delivery points,
	n	the number of customers,
	m	the number of trucks available,
$P(i)$,	$i = 1, \ldots, q$	location of delivery points $P(i) \in \mathbb{R}^2$,
$C(i)$,	$i = 1, \ldots, n$	location of customers $C(i) \in \mathbb{R}^2$,
$D(i)$,	$i = 1, \ldots, n$	demand of customers $D(i) \in \mathbb{R}, D(i) > 0$,
$T(i)$,	$i = 1, \ldots, m$	capacity of trucks $T(i) \in \mathbb{R}, T(i) > 0$.

Let us first look at the simplified case with only one fixed distribution point, i.e. assume that $q = 1$ and $P(1)$ is given. All trucks are operating from this distribution point. To model the route of a truck k we use an ordered subset $R^k = \langle r_1^k, \ldots, r_s^k \rangle \subset \{1, \ldots, n\}$, which defines the order of the customers visited by the truck. The total cost of such a tour from the delivery point $p \in \mathbb{R}^2$ is then

$$\boldsymbol{c}(p, R^k) = \boldsymbol{d}\left(p, C(r_1^k)\right) + \sum_{i=1}^{s-1} \boldsymbol{d}\left(C(r_i^k), C(r_{i+1}^k)\right) + \boldsymbol{d}\left(C(r_s^k), p\right),$$

where $\boldsymbol{d}(x, y)$ is the cost of transportation between two points x and y.

Then the problem is to find an ordered m-partitioning $\mathcal{R} = \{R^1, \ldots, R^m\}$ of the integer set $\{1, \ldots, n\}$ such that $R^i \cap R^j = \emptyset$, when $i \neq j$, and $\cup_{i=1}^m R^i = \{1, \ldots, n\}$, minimizing the function

$$\boldsymbol{f}(\mathcal{R}) = \sum_{k=1}^{m} (P(1), R^k) \tag{1}$$

so that for all $k = 1, \ldots, m$

$$\sum_{i=1}^{|R^k|} D(r_i^k) \leq T(k). \tag{2}$$

This partitioning corresponds to distributing customers to truck routes.

The general version of this problem involves the positioning of the distribution points and designating trucks and customers to each distribution point. Assume that we can solve the reduced case above with an algorithm $\mathcal{F}$ mapping a distribution point p and subset $S \subset \{1, \ldots, n\}$ of customers

using the subset of trucks $V = \{v_1, \ldots, v_t\} \subset \{1, \ldots, m\}$ to the optimal routes, i.e.

$$\mathcal{F}(p, S, V) = \arg\min_{\mathcal{R}} \left\{ \hat{f}(p, V, \mathcal{R}) : \mathcal{R} \text{ is an ordered } t\text{-partitioning of } S \right\},$$

where $\hat{f}$ is a generalization of the function f in (1), using only customers given in the index set S and trucks given in the index set V. The minimization over t-partitions of S is formally given by

$$\hat{f}(p, V, \mathcal{R}) = \sum_{k=1}^{t} (p, R^k) \tag{3}$$

constrained by

$$\sum_{i=1}^{|R^k|} D(R_i^k) \leq T(v_k) \quad \forall k = 1, \ldots, t. \tag{4}$$

Then the problem of positioning q delivery points is to locate the best coordinates for delivery points and distribute trucks and customers to each delivery point. Thus, we are looking for q points in $\mathbb{R}^2$ and two mappings $\mathcal{S} : \{1, \ldots, q\} \mapsto \mathcal{P}(\{1, \ldots, n\})$ and $\mathcal{T} : \{1, \ldots, q\} \mapsto \mathcal{P}(\{1, \ldots, m\})$ selecting the associated customers $\mathcal{S}(i)$ and trucks $\mathcal{T}(i)$ to each delivery point i respectively. The general problem is then to minimize

$$g\big(P(1), \ldots, P(q), \mathcal{S}, \mathcal{T}\big) = \sum_{i=1}^{q} \hat{f}\left(P(i), \mathcal{T}(i), \mathcal{F}\big(P(i), \mathcal{S}(i), \mathcal{T}(i)\big)\right)$$

over possible locations $P(1), \ldots, P(q) \in \mathbb{R}^2$ and over possible customer and truck mappings $\mathcal{S}$ and $\mathcal{T}$.

2 The algorithm

We will mainly concentrate on solving the simplified problem (1) and only sketch a few ideas for solving the general problem in the last section. We assume from now on a single fixed distribution point.

Solving this problem can be viewed as a combination of two separate processes. We need to cluster customers into groups, whose demand can be served by the capacity of one truck. We ignore now the situation where one truck can perform two trips faster than another truck can perform one trip. This is an unlikely possibility in the optimal case, if the capacities of trucks are approximately the same. We also assume that we have enough trucks

to cover all demand at any time, otherwise condition (2) cannot be fulfilled. The second part is to solve for the optimal cost for each single truck route. Because we are using the distance metric as a cost, this reduces the problem to the ordinary Euclidean TSP-problem.

Considering that the TSP-problem is NP-hard itself, it is clear that for practical use we want to find an approximate algorithm. Instead of solving a TSP-problem instance repeatedly in the clustering phase, we use simplified metrics to determine the fitness of the clustering. Note also that distributing customers to trucks is a bin-packing problem, which is also known to be NP-hard [2].

The approximate algorithm we implemented is as follows:

1. Choose the number of iterations N.
2. Denote trucks by $\mathcal{T}_i$, $i = 1, \ldots, m$, where each $\mathcal{T}_i$ is a set of customer indexes served by the truck i. Denote the center of mass of the truck $\mathcal{T}_i$ by

$$\boldsymbol{m}(\mathcal{T}_i) = \frac{1}{|\mathcal{T}_i|} \sum_{c \in \mathcal{T}_i} C(c).$$

   Denote also the *quality* of the cluster by

$$\boldsymbol{q}(\mathcal{T}_i) = \sum_{c \in \mathcal{T}_i} (\boldsymbol{m}(\mathcal{T}_i), C(c)).$$

3. Sort customer indexes into decreasing order $c'(1), \ldots, c'(n)$ by their demand.
4. For $i := 1, \ldots, m$:
5. Set $\mathcal{T}_i \leftarrow \{c'(i)\}$.
6. For $i = m+1, \ldots, n$:
7. Find the j minimizing $(\boldsymbol{m}(\mathcal{T}_j), C(c'(i)))$ constrained by

$$D(c'(i)) + \sum_{c \in \mathcal{T}_j} D(c) \leq T(j).$$

8. Set $\mathcal{T}_j \leftarrow \mathcal{T}_j \cup \{c'(i)\}$.
9. Repeat N times:
10. Select randomly i and j so that $1 \leq i, j \leq m$ and $i \neq j$.
11. Select randomly $c_1 \in \mathcal{T}_i$.
12. If $D(c_1) + \sum_{c \in \mathcal{T}_j} D(c) \leq T(j)$:
13. If $\boldsymbol{q}(\mathcal{T}_i \setminus \{c_1\}) + \boldsymbol{q}(\mathcal{T}_j \cup \{c_1\}) < \boldsymbol{q}(\mathcal{T}_i) + \boldsymbol{q}(\mathcal{T}_j)$:
14. Set $\mathcal{T}_i \leftarrow \mathcal{T}_i \setminus \{c_1\}$ and $\mathcal{T}_j \leftarrow \mathcal{T}_j \cup \{c_1\}$. Goto 10
15. Select randomly $c_2 \in \mathcal{T}_j$.
16. If $D(c_1) - D(c_2) + \sum_{c \in \mathcal{T}_j} D(c) \leq T(j)$ and
    $D(c_2) - D(c_1) + \sum_{c \in \mathcal{T}_i} D(c) \leq T(i)$:
17. If $\boldsymbol{q}((\mathcal{T}_i \setminus \{c_1\}) \cup \{c_2\}) + \boldsymbol{q}((\mathcal{T}_j \setminus \{c_2\}) \cup \{c_1\}) < \boldsymbol{q}(\mathcal{T}_i) + \boldsymbol{q}(\mathcal{T}_j)$:

18. Set $\mathcal{T}_i \leftarrow (\mathcal{T}_i \setminus \{c_1\}) \cup \{c_2\}$ and $\mathcal{T}_j \leftarrow (\mathcal{T}_j \setminus \{c_2\}) \cup \{c_1\}$.
19. For each $\mathcal{T}_i$:
20. Construct a TSP-problem $\mathcal{G}$ using points $P(1)$ and $C(c)$, $c \in \mathcal{T}_i$.
21. Find the minimum spanning tree of $\mathcal{G}$ using Kruskal algorithm.
22. Duplicate all edges of $\mathcal{G}$.
23. Find some Euler-path of $\mathcal{G}$ and sort the customers in $\mathcal{T}_i$ in the path order by skipping duplicate nodes on the Euler path.
24. Find all intersections in the path and remove them by reordering customers.

Lines 3–18 form the clustering part of the algorithm. We start by sorting the customers by their demand and fill m trucks with the m customers having the biggest demand. Then we assign the rest of the customers to the nearest truck with enough capacity. After this initialization step, we use random sampling to improve clusters either by moving or swapping customers. The clustering tries to minimize total distance of the customers from the mass center of the cluster. This gives a good starting point for the TSP-approximation [1] on lines 20–23 and some fine tuning by straightening path intersections.

We also tested a modified K-means [3] algorithm, but it appeared that this simple randomized algorithm gives better results.

3 Test results

The algorithm was implemented in C++ using GNU C++ compiler v4.0.2 and tested on an iBook G4 1333MHz Power PC using Ubuntu GNU/Linux. We tested the algorithm using uniformly distributed data with randomly placed circular holes. In Figure ?? there are 250 customers and 5 trucks. The capacity of the trucks was randomly selected between 25 and 35 and the demand for each customer was a random number between 0 and 1. We used $10\,000 \times n = 2\,500\,000$ iterations. This is usually enough for clusters to stabilize. In this example case, there were no changes after $2\,000 \times n = 500\,000$ iterations. When the algorithm finished, the capacities used in the trucks were 99.6%, 99.5%, 88.9%, 68.4% and 50.1%. Figure ?? shows how customers were distributed to trucks and Figure 2 shows the routes discovered by the TSP-approximation.

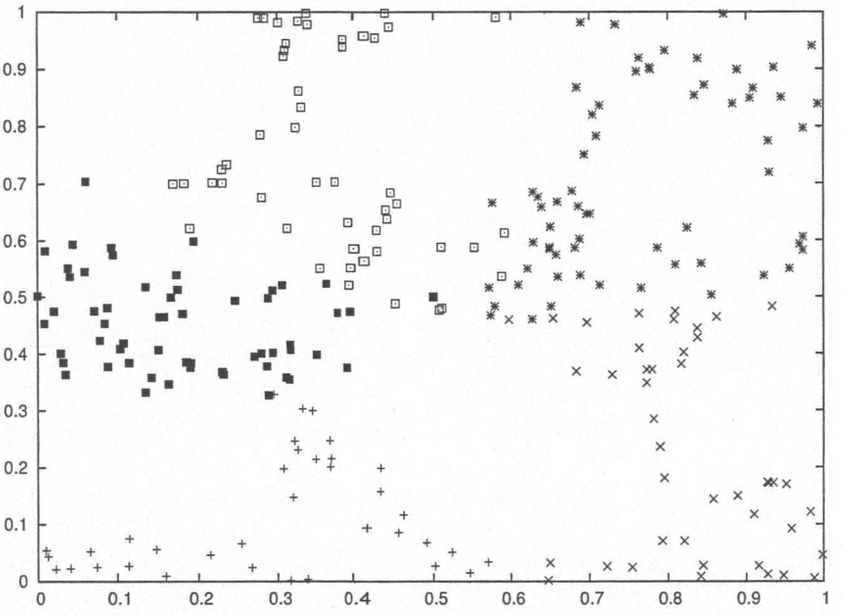

Figure 1: Customers after clustering. The clusters are denoted by different symbols.

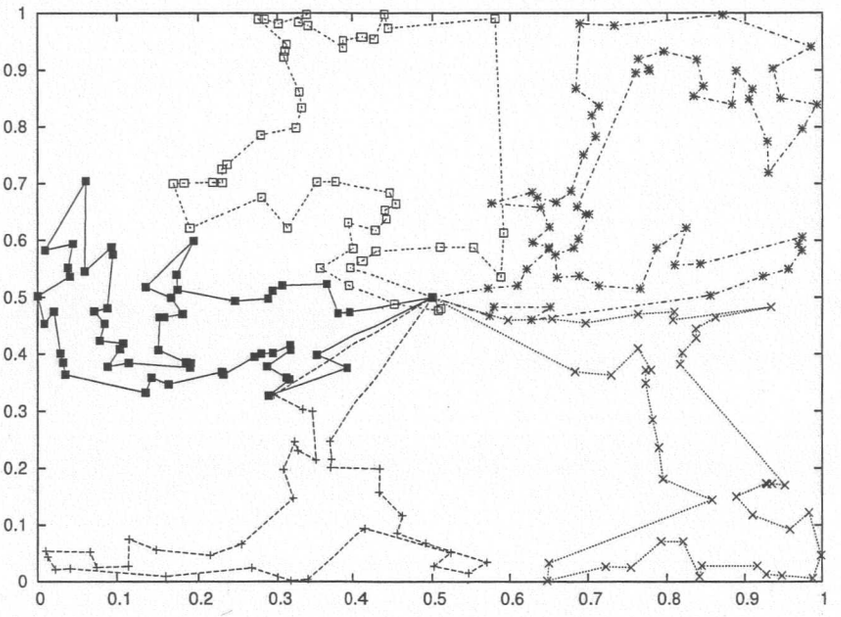

Figure 2: Routes of trucks.

Customers n	Trucks m	Average Time	Min. – Max.
50	5	2.4s	1.9s – 2.4s
100	5	7.1s	7.0s – 7.1s
250	5	28.6s	22.1s – 34.4s
500	5	125.3s	123.3s – 126.6s
1000	5	482.6s	474.3s – 487.1s
50	10	1.1s	0.9s – 1.3s
100	10	4.6s	3.6s – 5.4s
250	10	22.7s	22.3s – 23.0s
500	10	76.0s	74.6s – 78.1s
1000	10	95.6s	92.2s – 99.9s

Table 1: Running times.

We also made some performance testing. Table 1 summarizes the running times for uniformly distributed data using $10\,000 \times n$ iterations. Note that the algorithm is faster when there are more trucks. That is because clusters are smaller and thus TSP-problem instances are simpler to solve.

4 Conclusion

We have demonstrated that it is possible to obtain some solutions to the complex problem using relatively simple ideas. The solution presented here is probably adequate for some real life problems. It also forms a basis for further development. As we can see, the general approach of the algorithm is independent of the quality and the cost functions, $\boldsymbol{q}$ and $\boldsymbol{c}$. This leaves opportunities to improve heuristics used in the clustering as well as having more realistic modelling of the cost of transportation.

The generalized version of the problem was not in the scope of this paper. However, this algorithm might be a useful tool for that as well. The placement of the distribution points can be solved by the clustering part of the algorithm. That can be done by combining m/q trucks to one truck having the capacity of the sum of the capacities of the original trucks. Then we run the clustering part of the algorithm for this modified problem and use the centers of mass $\boldsymbol{m}(\mathcal{T}_i)$ as delivery points $P(i)$. The clusters also determine the customer to delivery point mapping $\mathcal{S}$.

References

[1] Kruskal, J.B. On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical Society, (1956) pp 48–50.

[2] Coffman, E.G., Garey, M.R. and Johnson, D.S. Approximation algorithms for bin packing: a survey. In Approximation Algorithms, editor D. Hochbaum, PWS Pubishing (1996), pp 46–93.

[3] McQueen, J.B. Some Methods for Classification and Analysis of Multivariate Observations. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, **1**, (1967) pp 281–297.