Vehicle Routing on Road Networks: How Good is Euclidean Approximation?

Burak Boyacı* Thu Huong Dang† Adam N. Letchford*

To appear in *Computers & Operations Research*

Abstract

Suppose that one is given a Vehicle Routing Problem (VRP) on a road network, but does not have access to detailed information about that network. One could obtain a heuristic solution by solving a modified version of the problem, in which true road distances are replaced with planar Euclidean distances. We test this heuristic, on two different types of VRP, using real road network data for twelve cities across the world. We also give guidelines on the kind of VRP for which this heuristic can be expected to give good results.

Keywords: vehicle routing problems, traveling salesman problem, road networks, combinatorial optimisation.

1 Introduction

Vehicle Routing Problems (VRPs) are an important family of combinatorial optimisation problems, and there is a huge literature on them (see, e.g. the books [4, 25, 26, 40]). In most existing VRP models, the customers and depot(s) are represented by nodes in a complete graph, and it is assumed that the distance between each pair of nodes is known. In practice, however, VRPs are often defined on *road networks* (e.g. [20,28,33]). We follow [17,29] in calling these *Steiner* VRPs.

If one has access to detailed road network data, it is usually possible to transform a Steiner VRP into a standard VRP, by solving a series of shortest-path problems (see, e.g., [17, 20]). When the road network is huge, however, the shortest-path computations themselves may consume a significant amount of time and memory. Moreover, for a given pair of nodes, it may happen that the cheapest, shortest and quickest paths are not the same.

*Department of Management Science, Lancaster University, Lancaster LA1 4YX, UK. E-mail: {B.Boyaci,A.N.Letchford}@lancaster.ac.uk

†STOR-i Centre for Doctoral Training, Lancaster University, Lancaster LA1 4YR, UK. E-mail: T.H.Dang@lancaster.ac.uk

Thus, when a Steiner VRP involves more than one feature (cost, distance and time, respectively), such a transformation may not be possible. In that case, specialised approaches are needed (e.g. [7, 22, 28]).

A related issue is that some VRP heuristics are explicitly designed to work on "planar Euclidean" instances, in which the depot and customer nodes have known coordinates. Examples include the classical "sweep" and "petal" heuristics [21, 24], and variations of them (e.g., [38]).

Another related stream of literature is concerned with the difference between distances in road networks and Euclidean distances (e.g. [5, 8, 9, 13, 14, 30, 31, 37]). It is known that the distance between two random points in a road network is typically around 30% larger than the Euclidean distance [14, 30], though this varies from country to country [5, 8]. On the other hand, the correlation between true and Euclidean distances is typically very high, at over 0.98, for most cities and countries [30, 37]. This suggests that, for Steiner VRPs, Euclidean distances multiplied by 1.3 could be a reasonable surrogate for true distances.

The above considerations suggest the following three-phase heuristic approach to Steiner VRPs: (1) Create an approximation of the given instance, by replacing true distances with Euclidean distances multiplied by 1.3 (or some other suitable constant); (2) Solve the approximated instance, either exactly or heuristically, and (3) attempt to convert the solution into a feasible solution to the original instance. A natural question is whether this heuristic scheme can lead to solutions of reasonable quality in practice. To address this, we conduct extensive computational experiments, using real road network data. Specifically, we construct 96 instances of the Steiner versions of the *Travelling Salesman Problem* (TSP) and *Capacitated VRP* (CVRP), using road network data for twelve cities across the world. For the Steiner TSP, we are able to compare optimal solutions of the original problem with optimal solutions of the Euclidean version. For the Steiner CVRP, we compare heuristic solutions obtained using road and Euclidean distances, using the same heuristic in both cases.

The experimental results show that Euclidean approximation can work surprisingly well in some cities. The results also enable us to give guidelines concerning the kind of Steiner VRP for which Euclidean approximation can be expected to perform well (or badly).

The paper has the following structure. Section 2 contains a brief literature review. Section 3 explains how we extracted our road network data and created our test instances. Sections 4 and 5 present the computational experiments for the Steiner TSP and Steiner CVRP, respectively. Section 6 presents the guidelines and contains some concluding remarks.

2 Literature Review

We now briefly review the relevant literature. We cover the Steiner TSP in Subsection 2.1, other Steiner VRPs in Subsection 2.2, the planar Euclidean TSP in Subsection 2.3, and studies of road distances in Subsection 2.4.

2.1 The Steiner TSP

In the Steiner TSP, we are given a connected undirected graph $G = (V, E)$, a positive cost c_e for each $e \in E$, and a set $V_R \subseteq V$ of *required vertices*. The task is to find a minimum-cost closed walk that visits each required vertex at least once. Edges may be traversed more than once if desired.

Cornuéjols *et al.* [17] defined the Steiner TSP and presented some polyhedral results for it. In the same year, Fleischmann [20] proposed a cutting-plane algorithm for the Steiner TSP and obtained encouraging results. Some additional computational results are given in [16].

Interest in the Steiner TSP was revived by Letchford *et al.* [29], who explored several integer programming formulations. Xia *et al.* [41] proposed to solve the problem with branch decomposition and dynamic programming instead. A specialised branch-and-cut algorithm was given in Rodríguez-Pereira *et al.* [39]. Álvarez-Miranda and Sinnl [1] proposed instead to convert Steiner TSP instances to standard TSP instances, and then use a state-of-the-art TSP solver like `CONCORDE` [2].

2.2 Other Steiner VRPs

Apart from the Steiner TSP, the Steiner VRPs that have received most attention are *Arc Routing Problems* (ARPs). An ARP is a VRP in which demands are located along the edges or arcs of a network, rather than at nodes. In the literature on ARPs, it is common to model problems directly on road networks, rather than on a complete graph (see, e.g. the books [15, 18]).

Outside the ARP literature, a key paper is Garaix *et al.* [22]. They pointed out that, in a road network, the shortest and quickest paths between two vertices may differ. This led them to propose a specialised exact algorithm for the Steiner VRP with time windows, based on a data structure called a *multi-graph*. Letchford *et al.* [28] proposed an alternative algorithm that works on the original road network rather than the multi-graph. However, Ben Ticha *et al.* [7] showed that, in practice, the multi-graph approach usually works better.

Another work worth mentioning, which however was never published, is Fleischmann [19]. Fleischmann defined a "Steiner" version of the CVRP, in which a node in V is designated the depot, and there is a fleet of identical

vehicles, each of capacity Q, located at the depot. He also discussed various integer programming formulations of the problem.

Finally, we mention that Letchford *et al.* [29] presented integer programming formulations of some other Steiner VRPs, in addition to the Steiner TSP.

2.3 The planar Euclidean TSP

In the planar Euclidean TSP, we are given the coordinates of some points in the plane, and the cost of travel between any two points is equal to the Euclidean distance between them. The goal is to find a minimum-cost tour that passes through each point exactly once. The planar Euclidean TSP is a special case of the so-called *metric* TSP, in which the costs obey the triangle inequality.

Unfortunately, the planar Euclidean TSP is strongly $\mathcal{NP}$-hard [23]. On the other hand, there is some evidence that it is a 'relatively easy' special case of the TSP:

- The metric TSP is APX-hard [35], but there is a polynomial-time approximation scheme for the planar Euclidean TSP [3].
- The fastest known exact algorithm for the TSP takes $O\big(n^2 2^n\big)$ time [27], but the planar Euclidean TSP can be solved in $O\big(2^{\sqrt{n}}\big)$ time [10].
- Large-scale planar Euclidean instances can often be solved to proven optimality in a reasonable amount of time by branch-and-cut [2, 34].

2.4 Road distances versus Euclidean distances

Cole & King [14] defined the "deviation factor" of a road network as the average, over all pairs of nodes, of the ratio between the road distance and the Euclidean distance. They stated that, for most cities and countries, the deviation factor ranges from 1.2 to 1.4.

Love & Morris [30, 31] proposed some more complex functions, based on ℓ^p norms, to estimate road distances from planar coordinates. Unfortunately, as pointed out by Berens & Körling [8, 9], the Love–Morris functions involve parameters that must be estimated, and the optimal parameter values can vary significantly from city to city.

Brimberg & Love [13] presented an alternative function, based on a linear combination of ℓ^1 and ℓ^2 distances. Again, however, the best parameters vary from city to city.

Phibbs & Luft [37] computed the correlation between road distances and Euclidean distances for several cities, using real data. The average correlation coefficient was remarkably high at 0.987. Interestingly, however, this reduced to 0.826 when the data was restricted to pairs of points that are no more than 15 miles apart in terms of Euclidean distance.

More recently, Ballou *et al.* [5] computed the deviation factor for several cities and countries across the world. They found that it ranged from 1.2 to 1.6, with mountains and rivers being the main cause of large values.

3 Data Collection and Instance Creation

In this section, we explain how we gathered our road network data, report some simple statistics for our selected road networks, and explain how we created our test instances.

3.1 Data collection

The first step was to select twelve cities from across the world. We selected London, Paris, Madrid, Barcelona, Moscow, Istanbul, New York, Mexico City, Hanoi, Seoul, Karachi and Johannesburg. Comprehensive data on the road networks of each of these cities are available from `OpenStreetMap` [32]. Road junctions and key landmarks are represented by nodes, and roads (or road segments) are represented by edges. The position of each node is given by its latitude and longitude, and the length of each road (or road segment) is given in kilometres.

We used the Python package `OSMnx` [12] to extract and process data from `OpenStreetMap`. For each city, we selected a key landmark as a "town centre", and then computed the size of the smallest square, with the given centre, that contained 2500 nodes. We then stored those nodes, together with all edges that connected pairs of the selected nodes. For a given city, we let V denote the set of 2500 nodes, E the set of edges and G the graph.

For reasons which will become clear, for each city, we also computed a smaller square, centred on the same point, that contained only 2000 nodes.

Table 1 gives the following for each of the twelve cities mentioned above: the length (and therefore also width) of the two squares, in metres; the name of the chosen town centre in `OpenStreetMap`; and the number of edges in E.

Figure 1 shows the maps for Paris, London and Mexico City, for the smaller squares (with 2000 nodes). These maps are based on the so-called *Universal Transverse Mercator* projection, which is the default projection in `OSMNX`. The two "holes" in the London map are caused by Buckingham Palace and two nearby parks, which more or less divide the given part of London into two districts. (A similar phenomenon occured with Madrid.) The "holes" in the Mexico City map are caused by sports facilities.

Now, let V' denote the set of 2000 nodes for each city, and note that $V' \subset V$. For each pair $\{u, v\} \subset V'$, we computed the Euclidean distance (in metres) between u and v, and the length (in metres) of the shortest path in G between u and v. We denote these quantities by $\delta(u, v)$ and $\Delta(u, v)$, respectively. To determine the Δ values, we used Dijkstra's single-source shortest-path algorithm.

City	Len 1	Len 2	Centre	$\lvert E\rvert$
London	1644.5	1896.3	Mayfair	5339
Paris	2135.0	2396.0	Eiffel	4932
Madrid	1845.4	2226.1	Puente de Vallecas	5093
Barcelona	1912.5	2205.0	Sant Gervasi - Galvany	4652
Moscow	3060.5	3608.5	Red Square	4929
Istanbul	1527.0	1743.7	Metrogarden Centre	7102
New York	3025.6	3301.6	Korean Town	5122
Mexico City	1637.3	1831.0	Granjas México	6146
Hanoi	1730.4	1959.5	National Cinema Center	5828
Seoul	1787.3	2095.5	The Plaza	6836
Karachi	1553.0	1814.5	Jinnahabad	7189
Jo'burg	2282.3	2670.0	Hillbrow	6037

Table 1: Extraction of twelve road networks with $|V| = 2500$

(a) Paris **(b) London** **(c) Mexico City**

Figure 1: Maps of smaller square regions

City	DF	r	Slope	Constant
Paris	1.174	0.989	1.094	129.126
Barcelona	1.201	0.986	1.119	112.312
Karachi	1.201	0.986	1.125	92.129
Moscow	1.255	0.983	1.124	292.372
London	1.268	0.976	1.200	79.442
Jo' burg	1.283	0.975	1.215	106.805
Istanbul	1.302	0.975	1.171	149.319
Madrid	1.306	0.944	1.265	37.712
New York	1.340	0.864	1.258	218.977
Hanoi	1.346	0.972	1.161	228.002
Seoul	1.358	0.954	1.189	191.468
Mexico City	1.403	0.966	1.188	272.295

Table 2: Euclidean distances versus true road distances for nodes in V'.

The reason for computing δ and Δ values for nodes in V', rather than V, is as follows. In the real road network, there exist many roads that connect nodes in $V \setminus V'$ with nodes outside of V. These roads are not included in E. Thus, if we computed shortest paths in G between nodes in $V \setminus V'$, there is a risk that the resulting Δ values would be over-estimates of the true road distances.

When computing the δ values, we regarded the latitude and longitude of each point as its horizontal and vertical coordinate, respectively. This induces some distortion in the computation of the δ values, given that the Earth is spherical. Fortunately, it can be shown that the distortion is less than 0.1% for each of our chosen cities.

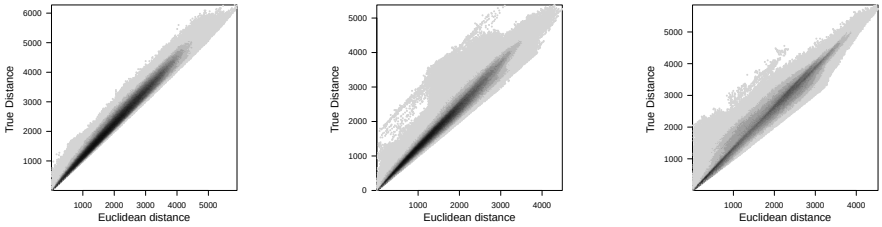
3.2 Road distances versus Euclidean distances

The next step was to compare road distances with Euclidean distances, for each city. Following Cole & King [14], we computed the *deviation factor* (DF) for each city, which we define as:

$$\binom{|V'|}{2}^{-1} \sum_{\{u,v\} \subset V'} \frac{\Delta(u,v)}{\delta(u,v)}.$$

We also performed a linear regression, comparing the Δ values with the δ values, again only for pairs of nodes in V'. Table 2 shows, for each city, the DF, the Pearson correlation coefficient, and the slope and constant in the regression. The cities are sorted in increasing order of DF.

Note that, in every case, the slope in the regression is less than the DF, and the constant term is positive. This suggests that the ratio between



(a) Paris **(b) London** **(c) Mexico City**

Figure 2: Scatterplots of road distance versus Euclidean distance

$\Delta(u, v)$ and $\delta(u, v)$ tends to decrease as $\delta(u, v)$ increases. This is confirmed by the scatterplots in Figure 2.

Note that the scatterplot for Paris is very "smooth", as one might expect from the very high correlation coefficient. The scatterplots for London and Mexico City, on the other hand, are remarkably "spread out". This is probably due to the presence of "holes", that we mentioned above.

3.3 Creation of Steiner TSP instances

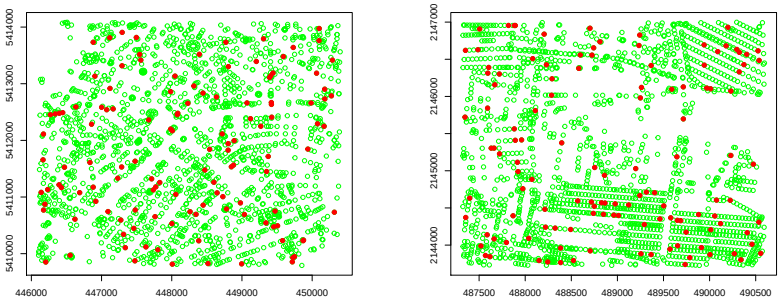
Next, we explain how we created our Steiner TSP instances. For each of the twelve cities, we constructed four instances, as follows. We took the corresponding graph $G = (V, E)$, and set $|V_R|$ to a value in $\{125, 250, 500, 1000\}$. To do this, we simply set V_R to a random subset of V' with the desired cardinality. (The reason for selecting required nodes from V' rather than V was to avoid over-estimation of the Δ values between pairs of required nodes; see Subsection 3.1.) The cost of each edge $e \in E$ was set to the length of the corresponding road, rounded to the nearest metre.

Figure 3 shows the instances for Paris and Mexico City, for the case $|V_R| = 125$. To aid visibility, only nodes in V' are displayed. The nodes in V_R and $V' \setminus V_R$ are represented by solid and hollow circles, respectively. (In the online version of the paper, the required nodes are in red and the others in green.)

For each city, we created an additional four instances, by setting V' to the 1000 closest nodes to the centre, instead of the 2000 closest nodes. The effect of this is that, for those instances, the required nodes become much closer together. This led to eight instances per city, i.e., 96 in total.

3.4 Creation of Steiner Capacitated VRPs

We also created 96 instances of the Steiner CVRP. To do this, we simply took each of the Steiner TSP instances, set the demand of each required node to one, and set the vehicle capacity Q to $|V_R|/5$. The node closest to the centre of the square was selected to be the depot. The objective function is to minimise the total distance travelled, and there is no limit on



(a) Paris **(b) Mexico**

Figure 3: Location of 125 required nodes for two cities

the number of vehicles used. (Of course, at least five vehicles are needed in any feasible solution.)

4 Experiments with the Steiner TSP

In this section, we describe our experiments with the Steiner TSP. For all experiments, we used a computer with an i5-8250U processor, running under Windows 10 at 1.6 GHz with 16GB of RAM.

4.1 Solution of Steiner TSP instances

The first step was to solve the Steiner TSP instances to optimality. Recall that an instance is given by a graph $G = (V, E)$ with $|V| = 2500$, a set of required nodes V_R, and a cost vector $c \in \mathbb{R}^E_+$. To solve it, we considered two options:

1. Use the "single-commodity flow" approach [29] to formulate the instance as an MILP with $O\big(|E|\big)$ variables and constraints, and then feed that MILP into `CPLEX`.

2. Convert the Steiner instance into a standard TSP instance with $|V_R|$ nodes, in which the cost of travel between nodes u and v is $\Delta(u, v)$. Then feed that TSP instance into `CONCORDE` [2].

We found that, for the instances considered, the second approach was faster.

From now on, for a given Steiner TSP instance, we let "OPT" denote the cost of the optimal solution (which is measured in metres).

4.2 The heuristic

The next step was to run the heuristic for each of the instances. In more detail, we did the following for each instance:

1. Construct a planar Euclidean TSP instance with $|V_R|$ nodes, in which the cost of travel between nodes u and v is $\delta(u, v)$. Then feed that TSP instance into `CONCORDE`.

2. Store the optimal TSP tour and let L be its cost.

3. Select an arbitrary starting node, and traverse the tour. Let v_k be the kth node visited in the tour, where k ranges from 1 to $|V_R|$.

4. For $k = 1, \ldots, |V_R| - 1$, run Dijkstra's algorithm to compute a shortest path in G from v_k to v_{k+1}. Also compute a shortest path from $v_{|V_R|}$ to v_1.

5. Replace each edge of the TSP tour with the corresponding shortest path, to obtain a heuristic solution to the original Steiner TSP instance.

6. Let U be the length of the heuristic solution, i.e., the sum of the lengths of the $|V_R|$ shortest paths.

We now make three remarks about this procedure:

- The shortest-path phase in step 3 takes very little time in practice, since v_k tends to be very close to v_{k+1} in G, and we abort the Dijkstra call as soon as the distance label for v_{k+1} becomes permanent.

- Let $t(e)$ denote the number of times that edge e is traversed in the heuristic solution. If $t(e) > 2$, the solution can be improved as follows: if $t(e)$ is even, set $t(e)$ to 2, otherwise, set it to 1. We let U^- denote the cost of the improved solution.

- For any given Steiner TSP instance, we have $L \leq \text{OPT} \leq U^- \leq U$.

4.3 Results

Table 3 shows, for each of combination of $|V'|$ and $|V_R|$, the average value of several ratios of interest. More details can be found in A.

An inspection of the ratios U/OPT and U^-/OPT reveals that, on the whole, Euclidean approximation performs reasonably well. In particular, in most cases, those ratios are much smaller than the corresponding DFs that we presented in Table 2.

On the other hand, both OPT/L and U/L tend to be larger than the corresponding DF. A possible explanation for this is the following two facts: (a) consecutive required nodes in an optimal Euclidean TSP solution tend to be close together and (b) as mentioned in Subsection 3.2, the ratio between $\Delta(u, v)$ and $\delta(u, v)$ tends to be higher when $\delta(u, v)$ is small.

It is also apparent that U/OPT, U^-/OPT, U/L and U^-/L tend to increase as $|V_R|$ increases, but decrease as $|V'|$ increases. Closer examination

$\lvert V'\rvert$	$\lvert V_R\rvert$	OPT/L	U/OPT	U^-/OPT	U/L	U^-/L
1000	125	1.403	1.092	1.078	1.538	1.518
1000	250	1.403	1.156	1.114	1.630	1.568
1000	500	1.391	1.207	1.136	1.815	1.676
1000	1000	1.315	1.353	1.183	1.799	1.562
2000	125	1.394	1.064	1.058	1.484	1.476
2000	250	1.417	1.106	1.082	1.571	1.535
2000	500	1.416	1.179	1.131	1.675	1.605
2000	1000	1.388	1.275	1.165	1.780	1.621

Table 3: Average ratios for Steiner TSP instances.

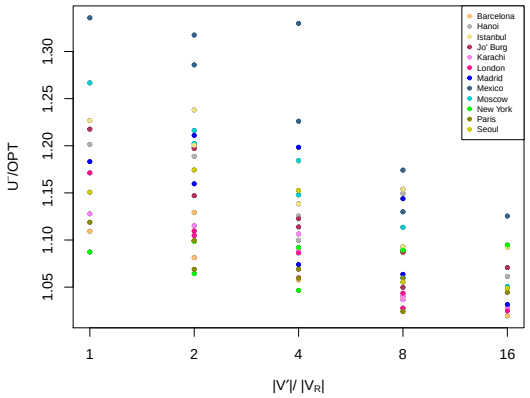
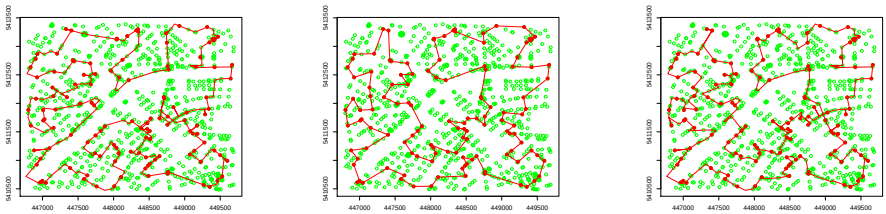


Figure 4: Scatterplot between $|V'|/|V_R|$ and U^-/OPT.

revealed that the quantity $|V'|/|V_R|$ plays a key role. For example, Figure 4 shows a scatterplot between U^-/OPT and $|V'|/|V_R|$ with different colors for different cities. It is apparent that the heuristic gets better as $|V'|/|V_R|$ increases. An explanation for this is that as $|V'|/|V_R|$ increases, the average distance between consecutive required nodes in the optimal Euclidean TSP solution increases. This in turn causes the average ratio between $\Delta(u, v)$ and $\delta(u, v)$ to decrease.

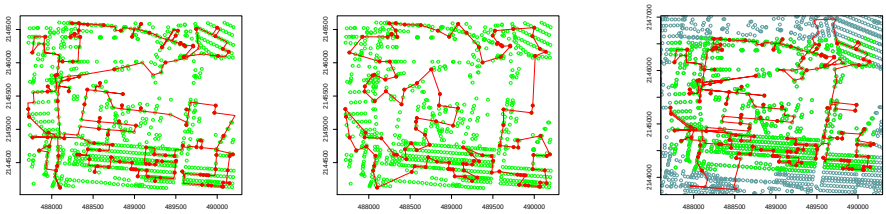
In Figure 5, we show the following for the Paris instance with $|V_R| = 125$ and $|V'| = 1000$: the optimal Steiner TSP solution, the optimal solution to the corresponding planar Euclidean TSP instance, and the Steiner TSP solution from the heuristic. As before, nodes in V_R and $V' \setminus V_R$ are represented by solid and hollow circles, respectively. It is clear that the heuristic solution is of excellent quality.

Figure 6 shows the same for the corresponding Mexico City instance. In this case, the tour found by the heuristic is of poor quality and passes through many nodes in $V \setminus V'$. (For this reason, in Figure 6(c), some nodes



(a) Steiner TSP solution **(b) Euclidean TSP solution** **(c)Solution from heuristic**

Figure 5: Paris with $|V'| = 1000$ and $|V_R| = 125$



(a) Steiner TSP solution **(b) Euclidean TSP solution** **(c)Solution from heuristic**

Figure 6: Mexico with $|V'| = 1000$ and $|V_R| = 125$

in $V \setminus V'$ are included. In the online version, they are cadet blue). In Figure 4, the solutions found by Euclidean approximation for Mexico City are the worst.

For completeness, we also present some results concerned with running times. (We emphasise, however, that our goal in this paper is to determine the loss of solution quality incurred by using Euclidean approximation, rather than to argue for the use of specific heuristics.) Table 4 shows the average running time, for each combination of $|V'|$ and $|V_R|$, taken for (a) computing shortest paths between all required nodes (T_S), (b) solving the TSP in `CONCORDE` (T_C), (c) solving the Euclidean TSP in `CONCORDE` (T'_C) and (d) computing shortest (s,t)-paths between consecutive nodes in the Euclidean TSP solution (T'_S). More details can be found in A.

Note that `CONCORDE` tends to solve the Euclidean instances more quickly than the non-Euclidean ones. Moreover, T'_S tends to be less than T_S. This is because consecutive nodes in the Euclidean TSP solution tend to be close together, and we abort each Dijkstra call as soon as the target node has been labelled permanently.

5 Experiments with the Steiner CVRP

In this section, we describe our experiments with the Steiner CVRP. We continued to use the same computer as described in Section 4.

$\lvert V'\rvert$	$\lvert V_R\rvert$	T_S	T_C	T'_C	T'_S
1000	125	0.219	0.294	0.510	0.325
1000	250	0.513	1.138	1.738	0.667
1000	500	1.502	17.553	13.034	1.371
1000	1000	4.408	8005.008	910.473	2.372
2000	125	0.441	0.343	0.147	0.276
2000	250	1.014	2.456	3.010	0.509
2000	500	2.632	27.858	11.641	1.016
2000	1000	5.215	276.058	103.762	1.563

Table 4: Average running time (in seconds) for Steiner TSP instances.

5.1 Heuristics

Since current exact CVRP algorithms tend to struggle when instances have more than 200 customers [36], we used heuristics both for the Steiner CVRP and for the Euclidean approximation. To make the comparison fair, we used exactly the same heuristic for both variants. In particular, we used a *route-first cluster-second* heuristic (see Beasley [6], Bodin [11]). In our experience, this kind of heuristic gives a good balance between solution quality and running time, while being easy to implement.

First we explain how the heuristic works when Euclidean approximation is *not* used. The first step is to solve (optimally) the Steiner TSP on the set V_R, using the same method that we used in Subsection 4.1. This yields a "giant tour" that passes through all nodes in V_R. The next step is to run Dijkstra's algorithm one more time, to compute shortest paths from the depot to each node in V_R.

Now, let v_k be the kth required node visited in the giant tour, where k ranges from 1 to $|V_R|$. For each pair $1 \leq i < j \leq m$, we create a trip that travels from the depot to v_i via a shortest path, then travels to $v_{i+1}, \ldots, v_{i+j}$, and then returns to the depot. If the trip is feasible, we store it in memory. We then use the standard method (see again [6]) to find the best CVRP solution that uses only trips that are in memory. Apart from the solution of the TSP in `CONCORDE`, the whole approach takes $O\big(|V||V_R| \log |V|\big)$ time. We let T be the total running time, and U denote the resulting upper bound.

For the Euclidean approximation, we proceed as follows. We solve the planar Euclidean TSP on the set V_R using `CONCORDE`. This gives a "giant tour" on V_R. We then use the standard method to convert the giant tour into a feasible solution to the planar Euclidean CVRP. Finally, we solve a series of (s,t)-path problems in G to convert the CVRP solution into a Steiner CVRP solution. Apart from the solution of the TSP in `CONCORDE`, the whole approach takes $O\big(|V||V_R| \log |V|\big)$ time. We let T_E denote the

$\|V\|$	$\|V_R\|$	U_E/U	T_E/T	U'_E/U	T'_E/T
1000	125	1.088	0.554	1.080	0.566
1000	250	1.131	1.338	1.106	1.340
1000	500	1.208	2.321	1.157	2.321
1000	1000	1.314	0.588	1.206	0.588
2000	125	1.055	1.446	1.050	1.457
2000	250	1.087	1.879	1.076	1.882
2000	500	1.163	0.952	1.141	0.953
2000	1000	1.243	0.859	1.174	0.859

Table 5: Average results for Steiner CVRP instances.

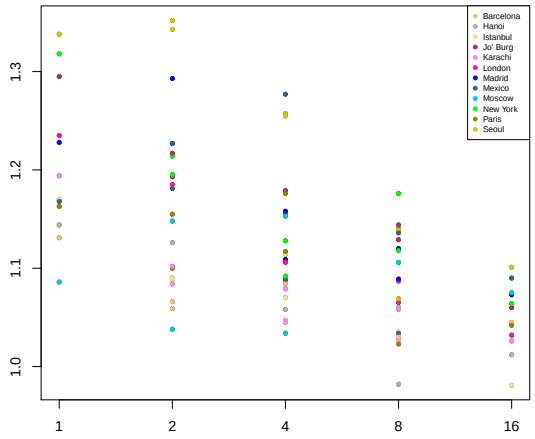


Figure 7: A scatterplot between $|V'|/|V_R|$ and U'_E/U

total running time, and U_E denote the resulting upper bound.

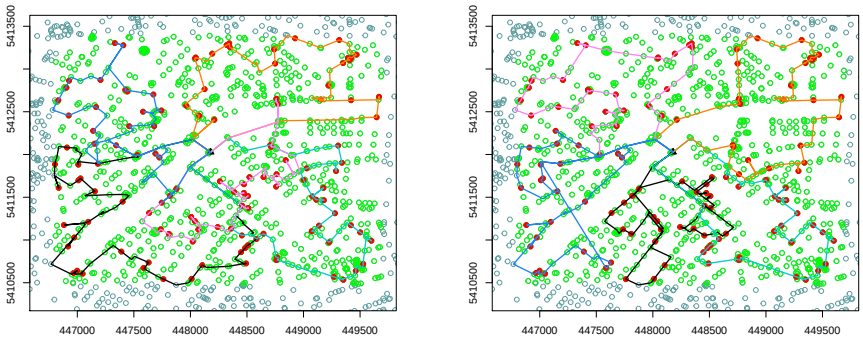
As in the case of the Steiner TSP, we can often improve U_E by checking if any of the vehicles traverse any edges more than twice. We denote the improve bound by U'_E. Note that $U \leq U'_E \leq U_E$.

5.2 Results

Table 5 shows, for each of combination of $|V'|$ and $|V_R|$, the average value of several ratios of interest. More details can be found in B.

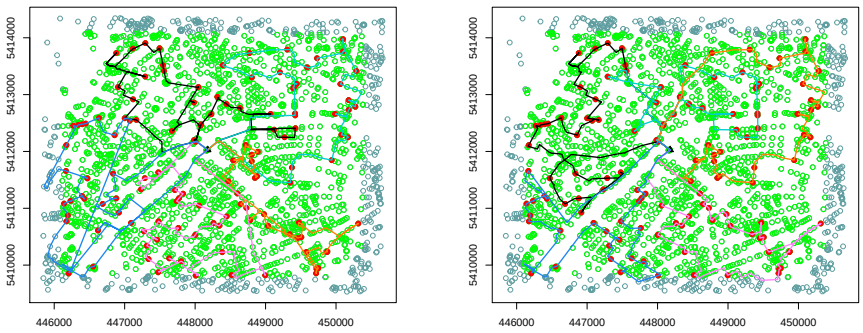
An inspection of the ratios U_E/U and U'_E/U reveals that, on the whole, Euclidean approximation performs reasonably well. As in the case of the Steiner TSP, it seems that the quality of approximation improves as $|V'|/|V_R|$ increases; see also the scatterplot in Figure 7. As for running times, there is no obvious pattern.

In Figures 8 and 9, we show, for the Paris instances with $|V_R| = 125$, the solutions that correspond to U and U_E. It is clear that, for these instances, Euclidean approximation yields solutions that are very close to the ones



(a) U **(b)** U_E

Figure 8: Paris with $|V'| = 1000$



(a) U **(b)** U_E

Figure 9: Paris with $|V'| = 2000$

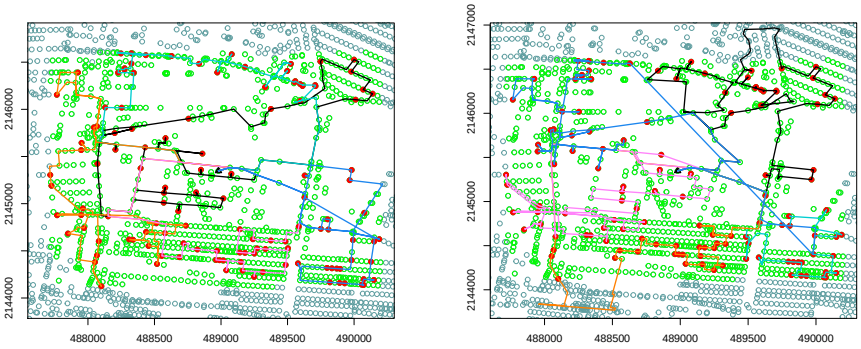
obtained without it.

Figures 10 and 11 show the same for the corresponding Mexico City instances. In this case, the solutions found by Euclidean approximation are noticeably different to (and worse than) the ones obtained without it. It is also comfirmed by Figure 7.

6 Concluding Remarks

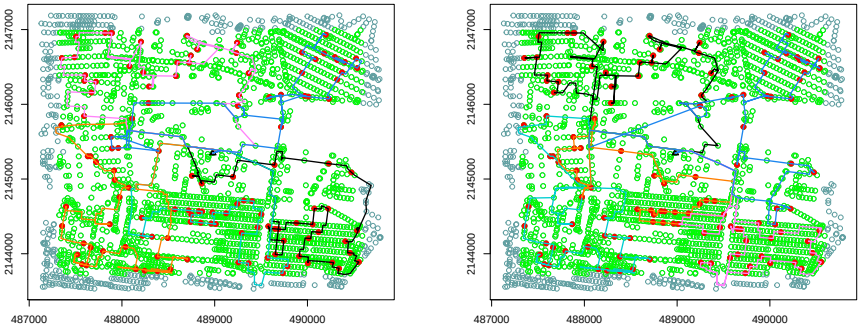
We have seen that, on the whole, using Euclidean distances instead of real road distances gives acceptable results for the Steiner TSP and Steiner CVRP. This is especially true when only a small proportion of nodes require service, which is the case in almost all real-life applications.

We believe that Euclidean approximation would also work well for some more complex Steiner VRPs, for example with split deliveries, backhauls, multiple depots, and/or demands located on edges as well as nodes. We are currently working on the use of Euclidean approximation to compute fast



(a) U **(b) U_E**

Figure 10: Mexico with $|V'| = 1000$



(a) U **(b) U_E**

Figure 11: Mexico with $|V'| = 2000$

upper bounds for arc routing problems.

It would be hard, however, to adapt the approach to problems with time windows. This is because a route that is feasible for the planar Euclidean version may become infeasible when the edges in the route are replaced with shortest paths in the road network. This difficulty could be alleviated somewhat by multiplying the Euclidean distances by 1.3 (or some other suitable constant, see Subsection 2.4) before solving the planar Euclidean version. Nevertheless, even if this is done, there is still a chance of obtaining routes that are infeasible for the original instance, especially if the time windows are narrow.

Acknowledgement: The second author gratefully acknowledges financial support from the EPSRC through the STOR-i Centre for Doctoral Training under grant EP/L015692/1.

References

[1] E. Álvarez-Miranda & M. Sinnl (2019) A note on computational aspects of the Steiner traveling salesman problem. *Int. Trans. Oper. Res.*, 26, 1396–1401.

[2] D. Applegate, R. Bixby, V. Chvátal & W. Cook (2006) *The Traveling Salesman Problem: A Computational Study.* Princeton NJ: Princeton University Press.

[3] S. Arora (1998) Polynomial-time approximation schemes for Euclidean TSP and other geometric problems. *J. of the ACM*, 45, 753–782.

[4] M.O. Ball, T.L. Magnanti, C.L. Monma & G.L. Nemhauser (eds.) (1995) *Network Routing.* Handbooks in Operations Research and Management Science, vol. 8. Elsevier.

[5] R.H. Ballou, H. Rahardja & N. Sakai (2002) Selected country circuity factors for road travel distance estimation. *Transp. Res. A*, 36, 843–848.

[6] J.E. Beasley (1983) Route first–cluster second methods for vehicle routing. *Omega*, 11, 403–408.

[7] H. Ben Ticha, N. Absi, D. Feillet & A. Quilliot (2018) Vehicle routing problems with road network information: state of the art. *Networks*, 72, 393–406.

[8] W. Berens (1988) The suitability of the weighted ℓ_p-norm in estimating actual road distances. *Eur. J. Oper. Res.*, 34, 39–43.

[9] W. Berens & F.J. Körling (1985) Estimating road distances by mathematical functions. *Eur. J. Oper. Res.*, 21, 54–56.

[10] M. de Berg, H.L. Bodlaender, S. Kisfaludi-Bak & S. Kolay (2018) An ETH-tight exact algorithm for Euclidean TSP. In *Proc. FOCS 2018*, pp. 450–461. IEEE Computer Society.

[11] L.D. Bodin (1975) A taxonomic structure for vehicle routing and scheduling problems. *Comput. Urban Soc.*, 1, 11–29.

[12] G. Boeing (2017) OSMnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Comput. Environ. & Urban Systems*, 65, 126–139.

[13] J. Brimberg & R.F. Love (1992) A new distance function for modeling travel distances in a transportation network. *Transp. Sci.*, 26, 129–137.

[14] J.P. Cole & C.A.M. King (1968) *Quantitative Geography*. London: Wiley.

[15] Á. Corberán & G. Laporte (eds.) (2014) *Arc Routing: Problems, Methods, and Applications*. Philadelphia, PA: SIAM.

[16] Á. Corberán, A.N. Letchford & J.M. Sanchis (2001) A cutting plane algorithm for the general routing problem. *Math. Program.*, 90, 291–316.

[17] G. Cornuéjols, J. Fonlupt & D. Naddef (1985) The travelling salesman problem on a graph and some related integer polyhedra. *Math. Program.*, 33, 1–27.

[18] M. Dror (ed.) (2000) *Arc Routing: Theory, Solutions and Applications*. Dordrecht: Kluwer.

[19] B. Fleischmann (1982) *Linear programming approaches to traveling salesman and vehicle scheduling problems.* Presented at the XIth International Symposium on Mathematical Programming, Bonn.

[20] B. Fleischmann (1985) A cutting plane procedure for the traveling salesman problem on a road network. *Eur. J. Opl Res.*, 21, 307–317.

[21] B.A. Foster & D.M. Ryan (1976) An integer programming approach to the vehicle scheduling problem. *J. Oper. Res. Soc.*, 27, 367–384.

[22] T. Garaix, C. Artigues, D. Feillet & D. Josselin (2010) Vehicle routing problems with alternative paths: An application to on-demand transportation. *Eur. J. Oper. Res.*, 204, 62–75.

[23] M.R. Garey, R.L. Graham & D.S. Johnson (1976) Some NP-complete geometric problems. In *Proc. of the 8th ACM STOC*, pp. 10–22.

[24] B.E. Gillett & L.R. Miller (1974) A heuristic algorithm for the vehicle-dispatch problem. *Oper. Res.*, 22, 340–349.

[25] B.L. Golden & A. Assad (eds.) (1988) *Vehicle Routing: Methods and Studies*. Amsterdam: North-Holland.

[26] B. Golden, S. Raghavan & E. Wasil (eds.) (2008) *The Vehicle Routing Problem: Latest Advances and New Challenges*. Boston, MA: Springer.

[27] M. Held & R. Karp (1962) A dynamic programming approach to sequencing problems. *SIAM Journal*, 10, 196–209.

[28] A.N. Letchford, S.D. Nasiri & A. Oukil (2014) Pricing routines for vehicle routing with time windows on road networks. *Comput. & Oper. Res.*, 51, 331–337.

[29] A.N. Letchford, S.D. Nasiri & D.O. Theis (2013) Compact formulations of the Steiner traveling salesman problem and related problems. *Eur. J. Oper. Res.*, 228, 83–92.

[30] R.F. Love & J.G. Morris (1972) Modelling inter-city road distances by mathematical functions. *Oper. Res. Quart.*, 23, 61–71.

[31] R.F. Love & J.G. Morris (1979) Mathematical models of road travel distances. *Mgmt. Sci.*, 25, 130–139.

[32] `OpenStreetMap` available at `http://wiki.openstreetmap.org`

[33] C.S. Orloff (1974) A fundamental problem in vehicle routing. *Networks*, 4, 35–64.

[34] M. Padberg & G. Rinaldi (1991) A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.*, 33, 60–100.

[35] C.H. Papadimitriou & M. Yannakakis (1993) The traveling salesman problem with distances one and two. *Math. Oper. Res.*, 18, 1–11.

[36] D. Pecin, A. Pessoa, M. Poggi & E. Uchoa (2017) Improved branch-cut-and-price for capacitated vehicle routing. *Math. Program. Comput.*, 9, 61–100.

[37] C.S. Phibbs & H.S. Luft (1995) Correlation of travel times on roads versus straight line distance. *Med. Care Res. Rev.*, 52, 32–42.

[38] J. Renaud, F. F. Boctor & G. Laporte (1996) An improved petal heuristic for the vehicle routeing problem, *J. Oper. Res. Soc.*, 47, 329-336.

[39] J. Rodríguez–Pereira, E. Fernández, G. Laporte, E. Benavent & A. Martínez–Sykora (2019) The Steiner traveling salesman problem and its extensions. *Eur. J. Oper. Res.*, 278, 615–628.

[40] P. Toth & D. Vigo (eds.) (2014) *Vehicle Routing: Problems, Methods and Applications*. Philadelphia, PA: SIAM.

[41] Y. Xia, M. Zhu, Q. Gu, L. Zhang & X. Li (2016) Toward solving the Steiner travelling salesman problem on urban road maps using the branch decomposition of graphs. *Infor. Sci.*, 374, 164–178.

A Steiner TSP Results

Tables A1 and A2 give results for the 96 Steiner TSP instances. They show the following for each instance: the number of required nodes, the length of the optimal Steiner TSP solution in metres, and various ratios of interest. Dashes indicate instances for which U^- is the same as U. In the column headed "T_U/T_{OPT}", T_U and T_{OPT} denote the total time taken by the heuristic and exact methods, respectively. The cities are sorted in increasing order of DF.

B Steiner CVRP Results

Tables B1 and B2 give results for the 96 Steiner CVRP instances.

City	$\lvert V_R\rvert$	OPT	T_{OPT}	U/OPT	U^-/OPT	OPT/L	U/L	T_U/T_{OPT}
Paris	125	30096	0.354	1.024	1.024	1.286	1.317	1.056
	250	40152	1.275	1.080	1.059	1.270	1.372	0.623
	500	53942	26.041	1.088	1.069	1.208	1.315	1.200
	1000	71682	331.798	1.178	1.119	1.198	1.411	0.072
Barcelona	125	26744	0.561	1.027	1.027	1.264	1.298	0.462
	250	39568	1.209	1.065	1.058	1.323	1.409	0.442
	500	52218	12.175	1.124	1.081	1.244	1.398	1.107
	1000	70638	5046.630	1.175	1.109	1.154	1.357	0.082
Karachi	125	25988	0.381	1.046	1.037	1.266	1.324	1.919
	250	35786	3.121	1.106	1.106	1.265	1.399	0.494
	500	46272	12.742	1.112	1.099	1.247	1.387	1.270
	1000	58323	247.505	1.181	1.128	1.187	1.401	5.142
Moscow	125	50085	0.384	1.081	1.060	1.493	1.614	1.594
	250	66486	0.779	1.211	1.148	1.400	1.695	2.338
	500	90680	2.755	1.326	1.202	2.440	3.236	1.187
	1000	114911	147.940	1.530	1.267	1.351	2.066	3.184
London	125	21736	1.034	1.038	1.028	1.319	1.368	0.548
	250	35164	0.771	1.081	1.060	1.432	1.547	1.791
	500	46528	19.245	1.165	1.105	1.408	1.640	0.135
	1000	62671	860.181	1.309	1.171	1.351	1.768	0.751
Jo' Burg	125	37002	0.327	1.058	1.050	1.430	1.512	1.719
	250	48524	2.170	1.144	1.114	1.395	1.595	2.169
	500	67491	36.416	1.256	1.197	1.388	1.743	0.416
	1000	85686	26315.294	1.369	1.218	1.335	1.828	0.001
Istanbul	125	24609	0.375	1.167	1.154	1.399	1.633	1.552
	250	31669	1.110	1.222	1.184	1.316	1.608	7.094
	500	43777	38.501	1.435	1.238	1.353	1.942	0.576
	1000	55225	40683.340	1.453	1.227	1.273	1.850	0.001
Madrid	125	25726	0.468	1.094	1.064	1.469	1.607	0.983
	250	36843	0.792	1.108	1.074	1.428	1.583	5.880
	500	43777	27.846	1.249	1.160	1.362	1.702	0.728
	1000	61255	1655.390	1.264	1.183	1.300	1.642	0.016
New York	125	38188	0.496	1.088	1.088	1.254	1.364	1.746
	250	48556	4.919	1.056	1.047	1.209	1.276	0.241
	500	65166	6.318	1.085	1.064	1.161	1.260	4.085
	1000	81534	12353.004	1.145	1.087	1.119	1.281	0.625
Hanoi	125	30949	0.820	1.160	1.149	1.704	1.977	0.491
	250	43837	1.007	1.160	1.100	1.806	2.094	0.982
	500	57353	14.019	1.299	1.198	1.663	2.160	0.305
	1000	78770	612.035	1.494	1.202	1.714	2.559	0.431
Seoul	125	21633	0.464	1.094	1.087	1.392	1.523	8.931
	250	30088	1.040	1.127	1.092	1.422	1.602	1.263
	500	39855	12.489	1.211	1.099	1.375	1.665	0.953
	1000	52281	1537.915	1.310	1.151	1.318	1.728	0.008
Mexico	125	31055	0.499	1.224	1.174	1.565	1.915	0.924
	250	41187	1.620	1.330	1.509	1.574	2.093	1.273
	500	55464	20.106	1.560	1.317	1.492	2.327	0.322
	1000	71650	6321.966	1.826	1.336	1.480	2.703	0.006

Table A1: Results for Steiner TSP instances with $|V'| = 1000$.

City	$\|V_R\|$	OPT	T_{OPT}	U/OPT	U^-/OPT	OPT/L	U/L	T_U/T_{OPT}
Paris	125	45523	0.527	1.046	1.045	1.282	1.341	0.767
	250	64232	6.741	1.064	1.060	1.292	1.375	0.124
	500	85955	41.998	1.079	1.069	1.295	1.398	0.141
	1000	112237	182.721	1.126	1.099	1.246	1.403	0.099
Barcelona	125	43880	0.381	1.019	1.019	1.306	1.332	0.591
	250	57636	3.106	1.044	1.040	1.326	1.384	0.599
	500	79516	8.290	1.104	1.090	1.321	1.459	0.545
	1000	113157	72.104	1.176	1.129	1.307	1.538	0.680
Karachi	125	34876	0.479	1.031	1.028	1.305	1.345	1.054
	250	49319	6.494	1.050	1.040	1.330	1.396	0.147
	500	65688	7.234	1.124	1.089	1.290	1.450	0.542
	1000	87759	200.307	1.138	1.115	1.238	1.409	2.604
Moscow	125	74460	0.490	1.053	1.051	1.432	1.508	0.902
	250	104426	2.048	1.157	1.114	1.496	1.731	1.490
	500	131120	4.272	1.217	1.184	1.424	1.733	5.212
	1000	182210	46.094	1.341	1.216	1.460	1.959	1.051
London	125	37193	0.428	1.026	1.025	1.376	1.411	0.871
	250	49487	0.939	1.054	1.044	1.416	1.492	1.167
	500	68414	13.314	1.099	1.086	1.365	1.501	0.144
	1000	89623	340.684	1.206	1.110	1.374	1.656	0.133
Jo' Burg	125	54215	0.874	1.096	1.071	1.387	1.520	0.523
	250	74904	7.781	1.102	1.087	1.475	1.626	2.495
	500	95815	4.048	1.150	1.123	1.386	1.594	1.120
	1000	130495	579.892	1.242	1.147	1.378	1.711	0.007
Istanbul	125	38105	0.636	1.111	1.092	1.458	1.620	0.619
	250	47075	1.956	1.112	1.093	1.362	1.514	2.327
	500	67996	231.774	1.190	1.138	1.440	1.714	0.100
	1000	87435	130.098	1.350	1.201	1.352	1.825	1.345
Madrid	125	35963	0.512	1.036	1.032	1.323	1.372	1.082
	250	56481	2.596	1.181	1.144	1.464	1.729	1.291
	500	78007	10.522	1.363	1.198	1.458	1.987	0.244
	1000	101852	424.388	1.402	1.211	1.414	1.983	0.100
New York	125	61746	0.497	1.099	1.095	1.422	1.563	0.948
	250	76981	1.753	1.108	1.089	1.333	1.476	2.315
	500	105543	7.706	1.108	1.092	1.295	1.436	2.845
	1000	137741	378.269	1.150	1.099	1.231	1.415	0.167
Hanoi	125	45157	0.706	1.065	1.061	1.589	1.692	0.467
	250	61001	3.184	1.141	1.090	1.577	1.798	0.456
	500	86562	17.374	1.201	1.125	1.683	2.021	0.809
	1000	117795	455.635	1.299	1.189	1.677	2.178	0.038
Seoul	125	30619	0.694	1.051	1.049	1.367	1.437	0.520
	250	46969	3.065	1.084	1.055	1.477	1.601	0.225
	500	66638	13.576	1.184	1.152	1.505	1.783	0.432
	1000	90786	377.850	1.294	1.174	1.461	1.890	0.699
Mexico	125	40018	3.183	1.130	1.125	1.481	1.673	0.173
	250	52156	1.977	1.180	1.130	1.460	1.723	0.465
	500	75101	5.761	1.321	1.226	1.533	2.025	7.143
	1000	103667	187.237	1.575	1.286	1.518	2.390	0.082

Table A2: Results for Steiner TSP instances with $|V'| = 2000$.

	Graph		Solution		Phase 1		Phase2	
City	$\|V'\|$	$\|V_R\|$	U	T	U_E/U	T_E/T	U'_E/U	T'_E/T
	1000	125	39387	0.288	1.027	0.389	1.025	0.403
	1000	250	49405	6.145	1.057	0.057	1.045	0.058
Paris	1000	500	64205	40.119	1.078	0.128	1.066	0.128
	1000	1000	81105	182.045	1.166	0.092	1.131	0.092
	1000	125	36189	0.409	0.989	0.352	0.982	0.362
	1000	250	46902	2.987	1.093	0.572	1.089	0.573
Barcelona	1000	500	59834	7.692	1.112	0.556	1.089	0.557
	1000	1000	78542	70.542	1.180	0.695	1.144	0.695
	1000	125	34706	0.286	1.038	0.671	1.035	0.689
	1000	250	44025	6.039	1.120	0.069	1.114	0.070
Karachi	1000	500	54695	5.770	1.105	0.515	1.089	0.515
	1000	1000	64549	197.549	1.219	2.638	1.170	2.638
	1000	125	62679	0.250	1.133	0.780	1.129	0.796
	1000	250	81481	1.352	1.172	1.950	1.156	1.953
Moscow	1000	500	108101	2.647	1.261	8.066	1.193	8.067
	1000	1000	132527	43.911	1.442	1.072	1.295	1.073
	1000	125	29374	0.316	1.074	0.443	1.058	0.456
	1000	250	42474	0.650	1.093	0.891	1.079	0.895
London	1000	500	55500	12.536	1.121	0.080	1.084	0.080
	1000	1000	70096	338.395	1.282	0.130	1.194	0.130
	1000	125	47098	0.342	1.069	0.471	1.065	0.482
	1000	250	59274	6.954	1.107	2.728	1.090	2.729
Jo' Burg	1000	500	77447	2.111	1.219	1.601	1.195	1.603
	1000	1000	98001	576.304	1.304	0.005	1.235	0.005
	1000	125	32398	0.332	1.131	0.434	1.120	0.443
	1000	250	40379	0.887	1.143	4.640	1.109	4.646
Istanbul	1000	500	51766	230.977	1.402	0.097	1.293	0.097
	1000	1000	64426	128.776	1.372	1.353	1.228	1.353
	1000	125	34763	0.300	1.041	1.003	1.034	1.017
	1000	250	44890	2.129	1.110	1.290	1.088	1.292
Madrid	1000	500	56146	8.675	1.235	0.157	1.181	0.158
	1000	1000	70813	422.553	1.212	0.097	1.168	0.097
	1000	125	50217	0.289	1.060	0.806	1.060	0.820
	1000	250	60523	1.285	1.042	2.795	1.034	2.798
New York	1000	500	78315	6.077	1.051	3.457	1.038	3.458
	1000	1000	94061	376.965	1.122	0.167	1.086	0.168
	1000	125	37036	0.392	1.187	0.355	1.176	0.365
	1000	250	51679	2.625	1.124	0.389	1.092	0.390
Hanoi	1000	500	65519	15.831	1.266	0.839	1.214	0.839
	1000	1000	85719	453.008	1.486	0.037	1.318	0.037
	1000	125	33792	0.286	1.161	0.839	1.140	0.853
	1000	250	40297	2.230	1.130	0.214	1.117	0.216
Seoul	1000	500	53020	11.842	1.169	0.462	1.100	0.462
	1000	1000	63617	377.450	1.269	0.699	1.163	0.699
	1000	125	44980	2.827	1.146	0.100	1.139	0.102
	1000	250	55517	0.930	1.384	0.456	1.257	0.460
Mexico City	1000	500	67630	3.380	1.478	11.893	1.343	11.894
	1000	1000	85090	184.903	1.711	0.074	1.338	0.074

Table B1: Results for Steiner CVRP with $|V'| = 1000$.

	Graph		Solution		Phase 1		Phase2	
City	$\|V'\|$	$\|V_R\|$	U	T	U_E/U	T_E/T	U'_E/U	T'_E/T
	2000	125	59555	0.331	1.050	0.834	1.045	0.843
	2000	250	78003	1.261	1.034	0.463	1.030	0.466
Paris	2000	500	98272	25.891	1.090	1.192	1.084	1.192
	2000	1000	126146	331.089	1.075	0.071	1.059	0.071
	2000	125	55457	0.563	1.016	0.284	1.012	0.290
	2000	250	68377	1.195	1.061	0.291	1.060	0.294
Barcelona	2000	500	92880	12.514	1.066	1.050	1.058	1.050
	2000	1000	126761	5045.895	1.152	0.082	1.126	0.082
	2000	125	48010	0.324	0.984	0.954	0.981	0.972
	2000	250	61408	2.947	1.031	0.182	1.026	0.184
Karachi	2000	500	78360	12.079	1.095	1.225	1.070	1.226
	2000	1000	100169	247.115	1.102	5.143	1.090	5.143
	2000	125	93548	0.364	1.064	1.011	1.060	1.033
	2000	250	121731	0.753	1.159	1.348	1.144	1.352
Moscow	2000	500	151465	2.582	1.192	0.716	1.179	0.718
	2000	1000	204989	147.940	1.289	3.172	1.217	3.172
	2000	125	48934	0.943	1.029	0.267	1.026	0.270
	2000	250	62347	0.680	1.038	1.204	1.026	1.210
London	2000	500	81685	18.868	1.055	0.081	1.047	0.081
	2000	1000	102689	859.985	1.166	0.751	1.102	0.751
	2000	125	70955	0.326	1.049	0.742	1.032	0.755
	2000	250	90505	2.018	1.102	2.041	1.087	2.043
Jo' Burg	2000	500	111526	36.197	1.118	0.378	1.106	0.378
	2000	1000	145352	26315.217	1.227	0.001	1.185	0.001
	2000	125	48660	0.311	1.088	0.955	1.073	0.968
	2000	250	57547	0.843	1.101	8.686	1.089	8.690
Istanbul	2000	500	78710	37.830	1.199	0.554	1.158	0.554
	2000	1000	98115	40683.938	1.315	0.001	1.227	0.001
	2000	125	48513	0.372	1.092	0.433	1.090	0.441
	2000	250	68328	0.686	1.138	6.044	1.136	6.050
Madrid	2000	500	89592	27.635	1.315	0.679	1.277	0.679
	2000	1000	114311	1654.951	1.362	0.015	1.227	0.015
	2000	125	79175	0.435	1.075	0.940	1.075	0.947
	2000	250	94726	4.794	1.115	0.109	1.106	0.110
New York	2000	500	119121	5.894	1.162	4.152	1.153	4.152
	2000	1000	152221	12353.404	1.169	0.625	1.148	0.625
	2000	125	55992	0.882	1.064	0.150	1.064	0.153
	2000	250	71724	0.966	1.130	0.322	1.118	0.325
Hanoi	2000	500	97896	13.977	1.169	0.221	1.128	0.221
	2000	1000	129269	610.425	1.269	0.432	1.195	0.432
	2000	125	45757	0.378	1.043	10.463	1.042	10.487
	2000	250	62760	0.847	1.045	1.138	1.023	1.145
Seoul	2000	500	79397	11.662	1.202	0.957	1.176	0.958
	2000	1000	106821	1537.491	1.229	0.007	1.155	0.007
	2000	125	56698	0.477	1.103	0.319	1.101	0.327
	2000	250	68801	1.583	1.094	0.717	1.069	0.720
Mexico City	2000	500	91527	19.801	1.289	0.223	1.255	0.223
	2000	1000	117344	6321.233	1.555	0.005	1.352	0.005

Table B2: Results for Steiner CVRP with $|V'| = 2000$.