On the Steiner Walk Problem

by

AMINA MAAROUF

B.S., Computer Science, Lebanese American University, 2005

Thesis submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science

Division of Computer Science and Mathematics

LEBANESE AMERICAN UNIVERSITY

June 2009

Lebanese American University
School of Arts & Sciences

Thesis Approval Form

Student Name: Amina Y. Maarouf I.D. #: 200102335

Thesis Title : On the Steiner Walk Problem

Program : Computer Science

Division/Dept : Department of Computer Science and Mathematics

School : Arts & Sciences

Approved by :

[redacted]

Faisal N. Abu Khzam, Ph.D. (Advisor)
Assistant Professor of Computer Science

[redacted]

Samer Habre, Ph.D.
Associate Professor of Mathematics

[redacted]

Nashat Mansour, Ph.D.
Professor of Computer Science

Date : June 26, 2009

Plagiarism Policy Compliance Statement

I certify that I have read and understood LAU's Plagiarism Policy. I understand that failure to comply with this Policy can lead to academic and disciplinary actions against me.

This work is substantially my own, and to the extent that any part of this work is not my own I have indicated that by acknowledging its sources.

Name: Amina Maarouf

Signature [redacted] Date: June 26, 2009

I grant to the LEBANESE AMERCIAN UNIVERSITY the right to use this work, irrespective of any copyright, for the University's own purpose without cost to the University or its students and employees. I further agree that the University may reproduce and provide single copies of the work to the public for the cost of reproduction.

To my beloved grandparents

Vladimir and Tamara

Your Memory fills me with strength and drives me forward.

Acknowledgment

It would be difficult to overstate my gratitude for my advisor, Dr. AbuKhzam. I will never forget his patience, support and valuable guidance throughout my graduate years. I would like to thank my committee members, Dr. Mansour and Dr. Habre for their constructive comments and advices.

I would like to express my sincere gratitude to the Lebanese American University whose financial support during my graduate studies made it all possible.

Sincere thanks to my family, special thanks for my mum and my best friend and sister who believe in me, support me and love me unconditionally. I will never be able to express my gratitude for my mum. You are the best mum ever; there is no one like you. I am so lucky to have you, and I know that no matter how much I tried, I will never be able to repay to you. Thank you mum, I love you. I want to thank my sister for staying up the nights with me just to make me feel better. Thank you for running my experiments, for offering help, for learning a lot of stuff just to help me. I am really lucky to have you and if I don't say it often, I really love you so much. Thanks to my friends for caring, for sharing and for being there. Special thanks to my best friend Samar El Haj Yusuf, who was always there for me, and who always pushed me forward, no matter what.

Thanks for all the people who tried hard to put obstacles in my way; they don't know how much they pushed me forward to achieve my goals.

Abstract

Given a Euclidean graph G = (V, E) with a special set of vertices, called the terminals, the Euclidean Steiner Walk problem asks for a walk through the vertices of the graph that starts and ends at a particular point, called the center, such that every terminal vertex is visited and the total cost of the walk (or distance) is minimized.

Despite its wealth of real applications, Euclidean Steiner Walk (ESW) seems to have been neglected thus far by the computing community. We show, easily, that the problem is NP-Complete and present a suite of heuristic approaches for computing efficient approximate solutions to the problem. A Graphical User Interface is also provided for users of our software and to allow developers to conduct further experimental work on the problem.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the Steiner Walk Problem, its applications and a number of related problems.

1.1 The Steiner Walk Problem

Consider the following problem: A postman needs to deliver post to specific mail boxes distributed over a known geographic area where post destinations are fixed and known. Each day, the postman has to deliver mail to a set of mail boxes. He/she would like to find an optimal tour starting from the post office, covering the needed set of addresses and then back to post office. The same problem applies to a delivery person with a potential set of clients from which only a subset needs to be served at a time.

Actually the above postman example can be transformed into a problem on a Euclidean weighted graph G = (V, E), where V is the set of vertices and E is the set of edges. In this graph, each mailbox in the given geographic area is represented by a vertex v. An edge e_{ij} is the road between vertex i and j with length w_{ij}. The set of mail boxes to be visited by the postman form a subset V' of V. The vertices in V' are called terminals.

Thus our postman problem can be treated as a Euclidean Steiner Walk problem. Figure 1 shows an example of a Euclidean Steiner Walk problem instance. The graph has three terminal nodes; any one of them can be the center. The problem is to find an optimal walk that contains all terminals and minimize the cost.

> **Definition 1.1** *Given a Euclidean weighted graph G = (V, E) with a set V' of marked vertices and a source node s. The Euclidean Steiner Walk problem asks to find an optimal tour that starts at s, visits all nodes in V' and ends at s. Vertices in V' are called terminals while those in V\V' are called Steiner vertices.*

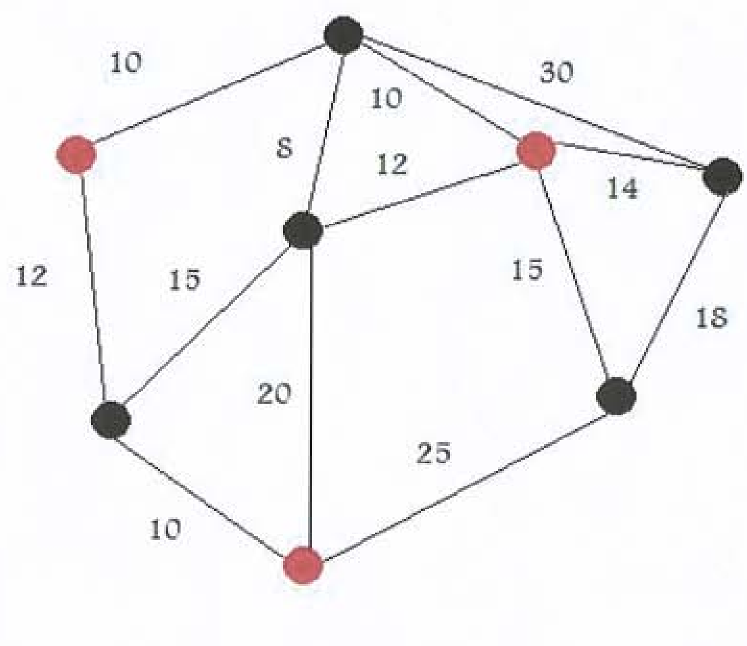


Figure 1.Graph with 3 terminal vertices

In the following section, we show that the Euclidean Steiner Walk problem is an NP-complete combinatorial optimization problem. The problem is named after the German mathematician Jacob Steiner, famous for his contribution to projective geometry. It arises in many contexts such as transportation, delivery and order picking. Despite its importance and applicability, this problem seems to have received little ot no attention from researchers so far.

1.2 NP Hardness

This section briefly presents preliminary definitions and concepts about complexity classes, and different types of algorithms. It also shows that the Euclidean Steiner walk problem is NP-hard.

1.2.1 Definitions and Notions

Problems can be broadly classified into two types; decision problems and optimization problems. A decision problem is a problem whose solution is either yes or no [Cormen, et al. 1990]. An example of decision problems is the graph coloring problem; given a graph G = (V, E) decided whether G can be colored using k colors. An optimization problem, on the other hand, is a problem where a quantity needs to be either maximized or minimized. An optimization graph coloring problem asks to color a given graph using the minimum number of colors. Every optimization problem has its decision version [Cormen, et al. 1990].

1.2.2 NP Class

P is the set of problems solvable on a deterministic sequential machine by a polynomial time algorithm, i.e. O (n^k) for some constant k where n is the input size to the problem. Even though many problems are in P, this is not the case for a large number of interesting problems. NP is the set of decision problems that have polynomial time verification algorithms, i.e. positive solutions can be verified in polynomial time. Every problem in NP can be solved in exponential time by exhaustive search. A problem is referred to be NP-complete if it is in NP and as hard as any problem in NP [Cormen, et al. 1990]. This means that every problem in NP can be reduced to a problem in NP complete class in polynomial time. Thus if any NP-complete problem can be solved in polynomial time, then every problem in NP has a polynomial time algorithm [Cormen, et al. 1990]. NP hard is the set of optimization problems whose decision version is in NP complete.

1.2.3 Proof for NP-Completeness

The travelling salesman problem (TSP) is one of the classical old and well known NP complete problems. In this section, we will use TSP to prove that Euclidean Steiner Walk problem is NP-complete. Given a Euclidean TSP instance G = (V, E), find a

Steiner Walk for that instance where all nodes are terminals; V'=V. Suppose that the Steiner walk for G has cost W_s and contains an ordered list of vertices. We can go over the list and delete any repeated vertex. The removal of repeated vertices is possible due to the Euclidean property of G, and given that G is a complete graph. Suppose that the solution has the following order of vertices:

$V_1, V_2, V_5, V_8, V_6, V_5, V_{10} \ldots V_n.$

Since the distance from V_6 to V_{10} is shorter than the distance from V_6 to V_5 and then from V_5 to V_{10}. We can safely remove V_5 in its second appearance, and shortcut the distance. Thus |cost of Optimal TSP| >= |cost of Optimal Steiner Walk|

Since by solving Euclidean Steiner Walk problem we can find the optimal travelling salesman tour, the Euclidean Steiner Walk problem is NP hard.

1.2.4 Overview of Heuristic Algorithms

Finding exact solutions for large instances of NP problems may not be always affordable. Heuristic algorithms try to find sub optimal solution in polynomial time [Kokash, 2006]. A heuristic algorithm is able to produce an acceptable solution to a problem in an acceptable time, but for which there is no formal proof of its correctness. Also, it may be correct but may not be proven to produce an optimal solution or to use reasonable resources [Wang, 2008]. There is a wide spectrum of different heuristic approaches and techniques, all try to find a near optimal solution while keeping the complexity of the algorithm within the polynomial range. Kokash (2006) and

Misevicius et al. (2004) summarize different heuristic approaches used to solve NP complete problems.

Metaheuristics algorithms combine heuristic techniques in the hope to get a better solution. Genetic algorithms (GA) are search based algorithms inspired from Darwin's theory of evolution and natural selection of species. A genetic algorithm starts with a set of solutions; each solution is represented by chromosomes and called population. Solutions from one population evolve to form a new population driven by a hope that the new solution will be better than the old one. Solutions are selected based on some fitness value. The more suitable they are the more chances they have to evolve. This is repeated until some condition is satisfied.

1.3 Applications

1.3.1 Transportation and Delivery

Many transportation problems seem to naturally map to Steiner Walk problem. The reason is pretty simple; in transportation problems we always look at optimizing a certain parameter like distance, time, costs, etc. In many transportation scenarios, we don't need to visit all nodes in the graph, but still look at optimal tour to cover the needed marked places to visit.

Consider a postman who goes to work every morning travelling from one post address to another to deliver parcels or letters in a certain assigned geographic area. It would be very convenient if the postman can get a list of address to deliver in the most optimal order. How about a delivery man who needs to deliver orders to a number of destinations distributed in his area? A musical band that is willing to go on a tour throughout the world? A salesman who travels from door to door marketing products? Would it be very efficient if one can get an ordered list of destination to minimize the total distance travelled or the total cost spent?

Consider a manufacturing company takes order to manufacture certain products. The company will need to buy supplies for an order. The order in which supplies are purchased is not important. Thus it would be convenient to have a list of stores to visit that minimize the time or distance travelled.

In the case of transportation company that wishes to find the best bus routes. A bus route has a number of stop stations where passengers wait. In would be highly suitable if the bus follows an optimal route minimizing the total distance travelled.

1.3.2 Manufacturing

Consider an assembly line machine whose purpose is to drill holes in a certain piece of material. The material could vary from a circuit board, to a frame of a vehicle or even a piece of wood to be used in building furniture. The machine contains a drill that is repositioned by special motors causing the drill to slide among the tracks enabling the drill to move to any point with the given material. It would be very efficient and time saving if the motors could reposition the drill among the given points in the optimal order.

In a similar scenario, consider a robot arm assigned to solder a number of connections on a printed circuit board. The shortest tour that visits the assigned solder points defines the most efficient path for the robot. A similar application arises in minimizing the amount of time taken by a graphics plotter to draw a given figure.

1.3.3 Networking

Consider the case of a network architect who wants to design the most efficient ring topology that will connect a number of special nodes in a network graph.

1.4 Related Problems

The Euclidean Steiner Walk Problem was not investigated in literature the way it is formulated in this thesis although one can find in literature a number of related problems, which either ask a different question or require a different solution. The following section presents three problems related to Steiner Walk Problem, mainly universal TSP, Steiner tree, and Hamiltonian problems.

1.4.1 Universal TSP

In the universal TSP problem, we are given a complete weighted undirected graph G = (V, E) where V is the set of vertices and E is the set of edges. The objective is to find a universal tour that covers all nodes in V such that for all subsets S in V, the induced sub-tours resulting from following the universal tour and skipping the vertices that are not included in the subset is optimal. In other words, the goal is to minimize the ratio of the length of the induced sub-tour over a subset of vertices S divided by the length of the optimal tour on S for all subsets S in V.

1.4.2 Steiner Tree

The Steiner tree problem asks for a minimum cost tree interconnecting a subset V' of vertices in a weighted undirected graph G = (V, E) where V is the set of vertices and E is the set of edges. The solution to the problem is called Minimal Steiner Tree and it may contain vertices in the subset V – V'.

1.4.3 Hamiltonian Problem

The Hamiltonian problem is one of the classic NP-complete problems tracing its origins back to 1850's. Hamiltonian path problem poses a question of finding a path that visits every node in a non weighted undirected graph G = (V, E) exactly once. The solution to the problem is either by finding such a path or claiming that there is no such path.

1.5 Overview and Contributions

In this thesis, we focus on heuristic approaches for the Euclidean Steiner Walk problem. The Steiner Walk problem is an NP-complete problem that has not been formulated in the way it is presented in this thesis. The problem has many real world applications especially in the field of transportation and delivery. The aim of this thesis is to define the Euclidean Steiner Walk problem, shed light on its applications and importance, present heuristic approaches for the problem and develop a user friendly tool that implements the different heuristic approaches for the Steiner walk problem.

The structure of this thesis is as follows:

Chapter 2 reviews previous works done on three related problems; mainly the Steiner tree problem and the universal travelling salesman problem. The reason behind reviewing related problems is that no previous work is found on our problem.

Chapter 3 presents four heuristic approaches to the Euclidean Steiner Walk problem.

Chapter 4 summarizes the experimental results conducted and analyses the algorithms' performance.

Finally chapter 5 draws conclusion and focuses on future work and further studies that can be done in this domain.

Chapter 3

Proposed Solutions

We present a few preliminary lemmas and remarks, and then we present our algorithmic approaches.

3.1 The Steiner Walk Problem Overview

Lemma 1: No optimal Steiner walk uses an edge more than twice.

Proof:

Consider a graph G = (V, E). Suppose that the optimal Steiner walk uses an edge more than twice.

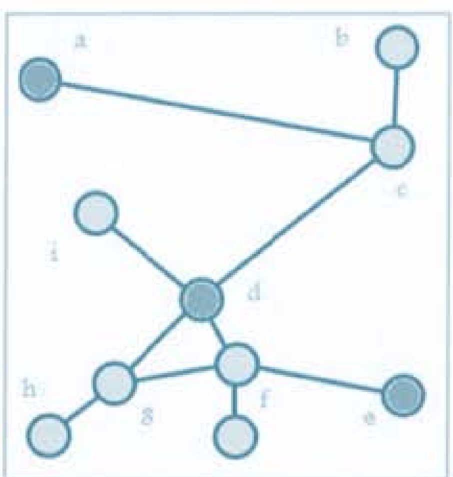


Figure 2. A graph with 3 terminal nodes

Let's say the optimal walk is a-c-d-f-e-f-e-f-d-c-a. The walk uses the edge (f, e) more than twice. We can get a smaller walk by deleting the repeated e-f pair from the above walk. If the edge (e,f) appears, but not successively, more than twice, then we distinguish two possibilities for the walk:

1. e-f-a ….x….b-e-f-c.. or
2. e-f-a ….x….b-f-e-c….y…d-e-f..

In the first case, we could replace the walk by the following: e-b …x…a-f-c. In the second, we replace the walk that covers the same vertices as follows: e-c…y…d-e-f-a…x…b-f… QED.

Thus an optimal walk cannot contain an edge more than twice (when e-f is followed, but not successively, by f-e).

Lemma 2: In any planar graph G= (V, E), number of edges <= 3|V| -6.

Proof: This is a well-known corollary of Euler's formula.

Corollary 1: In any planar graph G= (V, E), there exists a vertex v with degree <=5.

Proof: This is a well known corollary of Lemma 2.

Triangle inequality: for all sets of vertices A, B, C. the cost of travelling directly from A to B is lower than the cost of travelling from A to C and then from C to B.

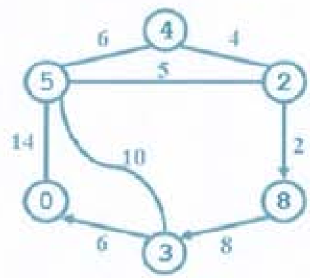


Figure 3. A graph with triangle inequality property (figure obtained from Chawla S.)

Lemma 3: Shortest paths satisfy the triangle inequality; i.e.

$\delta(s,v) \leq \delta(s,u) + w(u,v) \; for \; all \; (u,v) \in E$

Proof:

The shortest path from s to u followed by the edge (u, v) constitutes a path from s to v. The length of this path must be greater than or equal to the shortest path from s to v. Thus triangle inequality property holds true in all pairs shortest paths graph.

Lemma 4: The cost of the optimal Steiner walk is at most twice the cost of the Steiner tree.

Proof:

We can construct a Steiner walk by traversing a Steiner tree which will result in visiting each edge in the tree at most twice as shown in figure 3; thus the cost of optimal Steiner walk < 2|cost of optimal Steiner Tree|.

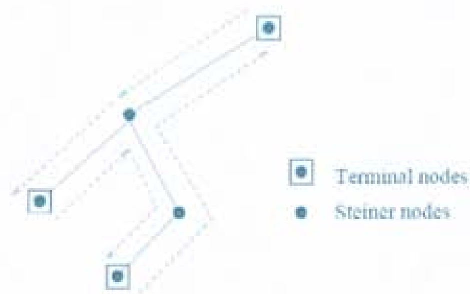


Figure 4. Constructing Steiner walk using Steiner tree (figure obtained from Chawla S.)

3.2 The Greedy Approach

The idea behind this algorithm is to try to find a Steiner walk that is a cycle, with no edge used more than once. The greedy approach constructs a shortest path matrix M for the terminals. The algorithm sorts the edges in M as pairs of vertices; each pair has a certain weight which is the weight of the corresponding edge. Then the algorithm starts with the pair having minimum weight, it then checks if adding the edge creates a cycle. The only cycle that is accepted is the one containing all terminals. In order not to get stuck in a local optimum, the algorithm iterates over all pairs, in each iteration, the algorithm select a different pair, paying no consideration to its cost as a starting pair. The algorithm will then select a Steiner walk with minimum total cost. The steps of the greedy approach are described below.

INPUT: Euclidean graph G= (V, E) and a set V' of marked vertices

OUTPUT: An ordered list V'' of vertices such that V'' contains V' and having a minimum cost.

The Greedy Approach:

STEP 1: Construct the shortest path matrix M for the terminals set. Sort the edges in M in increasing order.

STEP 2: REPEAT for all e in M

STEP 2.1: Start a new solution by adding e

STEP 2.2: Find an edge m with minimum cost

STEP 2.3: IF m does not create a cycle, THEN add m to solution

STEP 2.4: IF m creates a cycle AND there are still terminals not covered, THEN delete m.

STEP 3: IF walk found, THEN calculate cost and go to STEP 2.

STEP 4: Select a Steiner walk with minimum cost.

3.3 The MiniMAL Steiner Tree (ST) Walkthrough Approach

In this heuristic, we construct a Steiner tree T (not necessarily minimum), by iteratively adding Steiner nodes to T whenever there are terminal nodes that cannot be connected to T but through a Steiner node. Note that T will never have a Steiner leaf. A Steiner walk is then constructed by a walkthrough over the resulting ST. The walkthrough begins at a leaf (which must be a terminal node). It then traverses the ST by going to an unvisited node, when such a step is not possible anymore; the algorithm tries to find an unvisited terminal with the shortest distance to the current node. If no such terminal is found, the algorithm goes back until it finds a node that has unvisited neighbors. The steps of the algorithm are described below.

INPUT: Euclidean graph G= (V, E) and a set V' of marked vertices

OUTPUT: An ordered list V'' of vertices such that V'' contains V' and having a minimum cost.

ST_WALKTHROUGH

STEP 1: Find terminal spanning tree T.

STEP 2: Start with a leaf node s, add s to the walk.

STEP 3: Repeatedly go to an unvisited neighbor and add it to the walk.

STEP 3.1: If no unvisited neighbor is found:

STEP 3.1.1: Find an unvisited terminal t with shortest distance from the current node.

STEP 3.1.1.1: IF t is found, add t to the walk and go to STEP 3.

STEP 3.1.1.2: IF t is not found, go back to the previous node n, add n to walk and go to STEP 3.

The algorithm runs in O (n^2) time and constructs a walk whose cost is at most 2|cost of TST|.

The ST algorithm works best when traversing a Minimum Steiner Tree rather than our proposed ST. A Steiner tree algorithm gives the minimum tree spanning all terminal vertices, thus the cost of the Steiner walk resulting from the Steiner tree traversal will be better. The problem here is that Steiner tree problem is by itself an NP-hard problem. Solving the Steiner tree problem exactly requires exponential time, which is unaffordable in our case. On the other hand, solving the Steiner tree using approximation algorithms will result in double approximation, which is one level of approximation to find the Steiner tree and the second level of approximation to find the Steiner walk out of the resulted Steiner tree. Since Minimum Steiner tree has a 1.5 approximation algorithm, our ST-Walkthrouigh method yields a factor-3 approximation algorithm, mainly because a tree edge is not used more than twice.

3.4 The Traveling Salesman Approach

The travelling salesman (TSP) heuristic reduces the input graph to a TSP instance graph. Given a graph G = (V, E) with V' is the set of terminal nodes in G, we can easily obtain a complete graph G' = (V', E') where E' is the set of shortest paths between any pair of vertices in V'. Find a tour using Christofide's algorithms and then insert appropriate Steiner nodes to the walk to get a Steiner walk. Christofide's algorithm runs in O (n^3) during which an MST for the reduced graph G' is calculated, and then edges are added between MST nodes with odd degrees, as shown in figure 6. Then Euler walk is found by traversing each edge in the resulted MST.

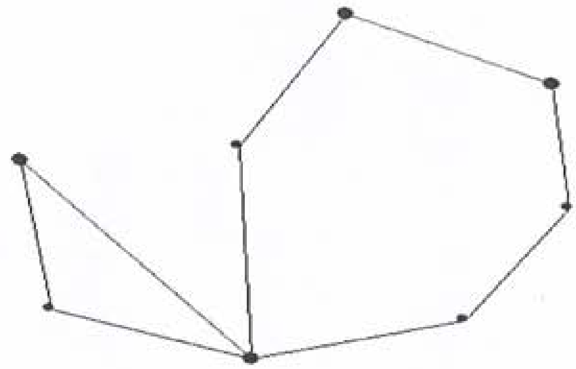


Figure 5. Adding minimum weight edges to MST

The steps of the travelling salesman approach are described below. The algorithms runs in $O(n^3)$ time.

INPUT: Euclidean graph G= (V, E) and a set V' of marked vertices

OUTPUT: An ordered list V'' of vertices such that V'' contains V' and having a minimum cost.

TSP_HEURISTIC

STEP 1: Construct a complete graph G'=(V',E') where V' is the set of all terminal vertices and E' is the set of edges and shortest paths between any pair of vertices in V'

STEP 2: Apply Christofide's Algorithm to G' to get walk W

STEP 3: For every consecutive nodes v_1 and v_2 in W, if v_1 and v_2 are not adjacent in G, insert the nodes from the shortest path from v_1 to v_2 to the walk.

3.5 The Cluster and Conquer (CC) heuristic

The Cluster and Conquer heuristic starts by clustering the input graph into a specified number of small clusters. The algorithm then finds minimum Steiner walks in each cluster and connects the optimal walks. Here are the main steps:

- Construct a complete weighted graph G' consisting of the terminal nodes as vertices, such that the weight of a edge uv is the length of shortest path between terminals u and v in G.
- Find a minimum spanning tree in G'.
- Cluster the graph G:
  - Repeatedly select the minimum edge e_{ij}:
    - If vertices i and j are not clustered, create a new cluster and add i and j to the cluster.
    - If either one of the vertices i, j is clustered in a cluster c:
      - If c is not full, add the other vertex to c.
      - Else create a new cluster, add the other vertex to that cluster and store vertex e_{ij} in a vector of marked edges.
      - Delete e_{ij}

. The cluster size is specified by a strict upper bound. The reason behind this is the need to keep the cluster size small for acceptable time calculation of the optimal walk in the cluster. The algorithm then connects the clusters to construct a Steiner Walk. The steps of the algorithm are described below:

INPUT: Euclidean graph G= (V, E) and a set V' of marked vertices

OUTPUT: An ordered list V'' of vertices such that V'' contains V' and having a minimum cost.

CLUSTER_CONQUER

STEP 1: Cluster the graph into clusters with specific cluster size k.

STEP 2: Find minimum walk in each cluster.

STEP 3: Connect minimum walks in all clusters to get the Steiner walk.

Chapter 4

Experimental Results

A graphical user interface (GUI) tool is developed implementing the different proposed approaches. The tool is written in visual C#. We used Gengraph tool to generate random planar graphs. Since Gengraph generates graphML (graph markup language) files, our tool allows for the conversion from graphML to text file in the format needed. The tool also allows for the conversion from TSPLIB files and text files since many geographic graphs and maps are represented in the TSPLIB format. Appendix A shows snapshots of the developed tool.

To the best of our knowledge, there is no such tool or algorithm that finds Steiner Walk in Euclidean graphs, so we are not able to compare our results to some previous work. This section presents experimental results and compares the performance of the different algorithmic approaches presented in Chapter 3.

To test the efficiency of our algorithms, we computed the optimal Steiner Walk for small graph instances and compared the results with our proposed heuristics. The performance of the heuristics was within the factor 2 of the optimal for graphs of order less than 50.

The experiments were performed on a Pentium IV 2.00 GHz machine with 3.00 GB of RAM. Table 1 shows a summary of the results. The detailed experimental results are presented in Appendix B.

Table 1. Summary of the experimental results showing the cost of Steiner walk using the different proposed approaches

Graph		terminals	No. of test cases	TST wins	TSP wins	CC wins	PM wins
n	e						
200	584	10	18	1	0	4	13
200	586	20	19	0	0	8	11
200	586	30	10	0	0	3	7
200	586	40	5	0	0	2	3
200	586	50	9	0	0	7	2
300	884	30	30	0	0	12	18
300	884	40	30	0	0	16	14
300	884	50	30	0	0	25	5
300	884	60	30	0	0	25	5
300	884	70	20	0	0	15	5
300	884	80	20	0	0	17	3
300	884	90	20	0	0	18	2
400	1000	40	7	0	0	3	4
400	1000	50	5	0	0	1	4
400	1000	60	8	0	0	2	6
400	1000	70	5	0	0	4	1
400	1000	80	5	0	0	4	1
400	1179	40	10	0	0	9	1
400	1179	50	10	0	0	9	1
400	1179	60	10	0	0	8	2
400	1179	70	10	0	0	8	2
400	1179	80	10	0	0	8	2
400	1179	90	10	0	0	10	0
400	1184	40	5	0	0	3	2
400	1184	50	5	0	0	5	0
400	1184	60	5	0	0	5	0
400	1184	70	5	0	0	5	0
400	1184	80	5	0	0	3	2
400	1184	90	5	0	0	5	0
400	1178	40	5	0	0	4	1
400	1178	50	5	0	0	3	2

400	**1178**	**60**	**5**	**0**	**0**	**5**	**0**
400	**1178**	**70**	**5**	**0**	**0**	**4**	**1**
400	**1178**	**80**	**5**	**0**	**0**	**5**	**0**
400	**1178**	**90**	**5**	**0**	**0**	**4**	**1**

In most of the test cases, cluster and conquer approach outperformed the other three approaches. We have noticed that as the graph increases in size, Cluster and Conquer approach yields better results than its peers. Even though, in the above table, it may seem that the first two approaches have worse performance. We could not generalize this conclusion, since the results show that when the best cost is given by PM approach, it does not necessarily imply that the next to best cost is given by CC approach. In many cases, TSP approach gives next to best cost. Terminal spanning tree approach does not give optimal Steiner walks on large graph instances. However it gives good results on small graphs that have nodes less than 100. The terminal spanning tree approach gives better Steiner walks than TSP in some cases especially when the graph size is small and the terminals set is small.

The first approach (TST) runs faster than the other three approaches while the Cluster and Conquer approach takes the longest running time. The time taken by each experiment is recorded in Appendix B.

Figures 7 to 9, shows the number of wins of each proposed heuristics as the terminal size changes. Each figure use graphs of fixed order. Figures 10 to 12 show the average run times of the different heuristics on different terminal size and different graph orders.

The time reported is system time and is recorded in milliseconds. The TST and the TSP heuristics run much faster than the other two approaches. The C&C approach is the slowest and its run time lies within a bounded range for graphs of same size. With graphs of order 200, the C&C algorithm runs in 4 to 5 seconds, while with graphs of order 300, the C&C algorithm runs in 25 to 30 seconds. The average run times of the greedy approach increase as the terminal size increases. With graphs of order 300, as an example, the average run times of the greedy approach varies from less than 2 seconds for 30 terminals to about 25 seconds for 90 terminals.

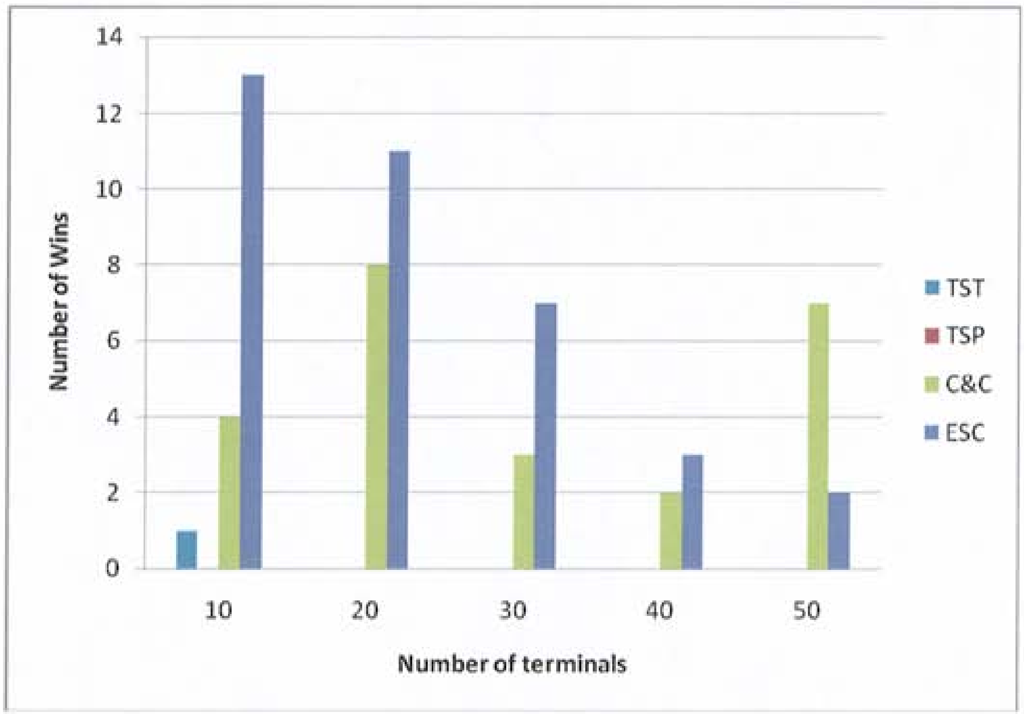


Figure 6. Number of Wins in graphs of order 200

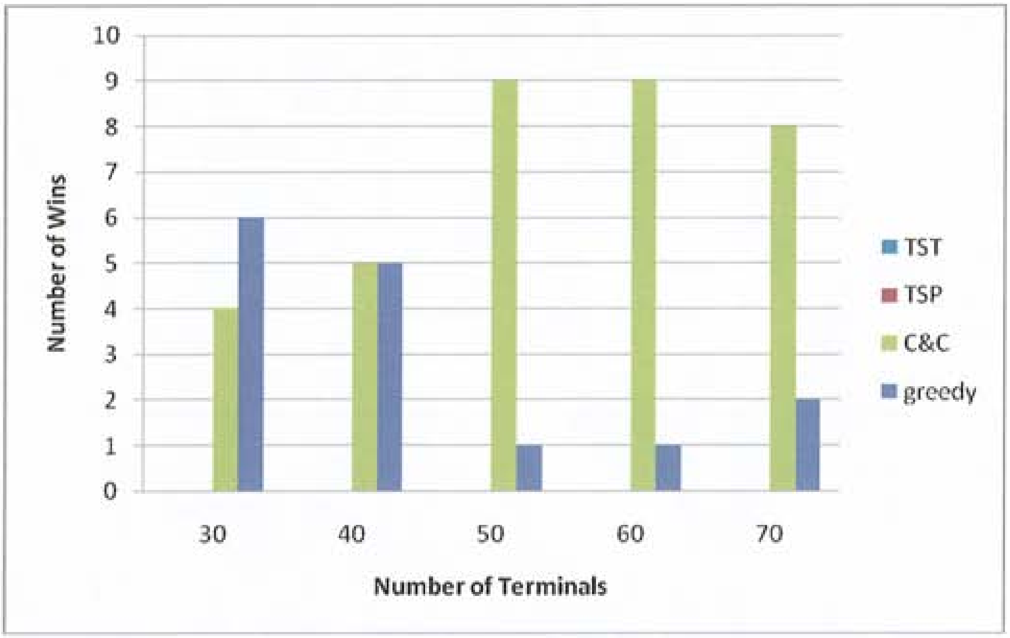


Figure 7. Number of Wins in graphs of order 300

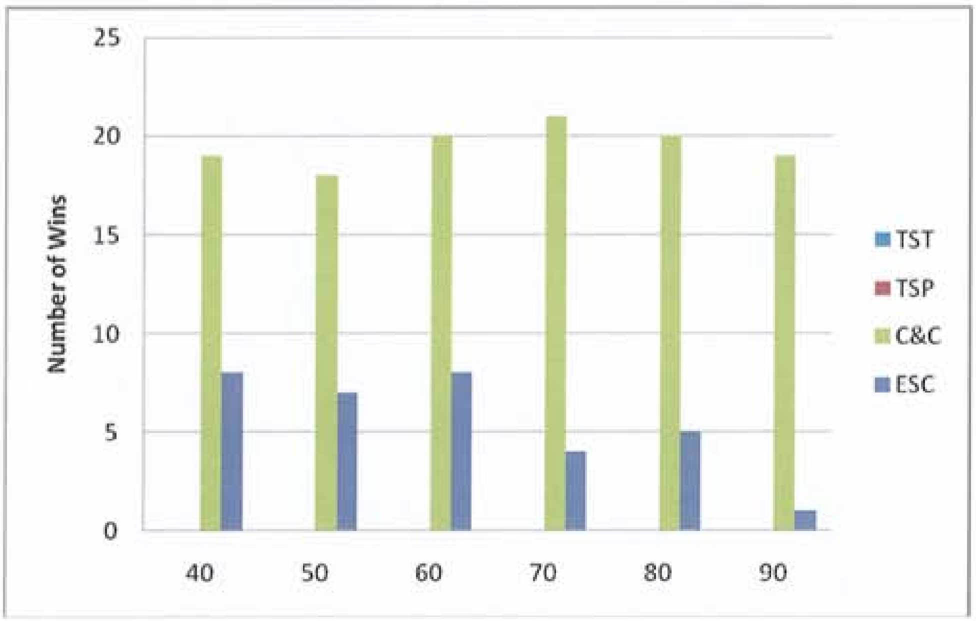


Figure 8. Number of wins in graphs of size 400 nodes

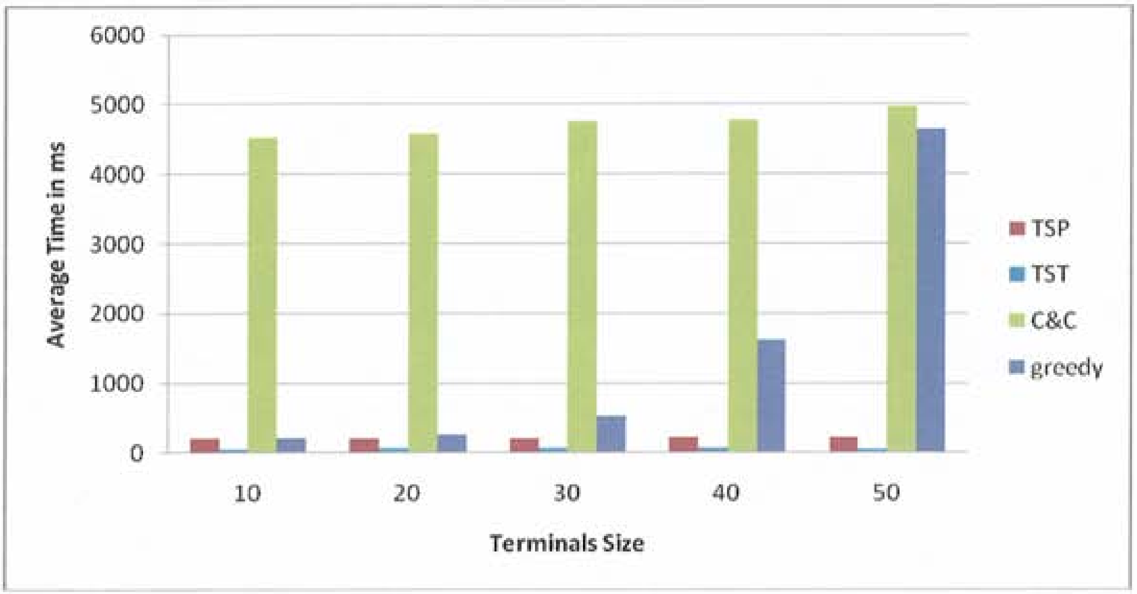


Figure 9. Average run time for graphs of order 200

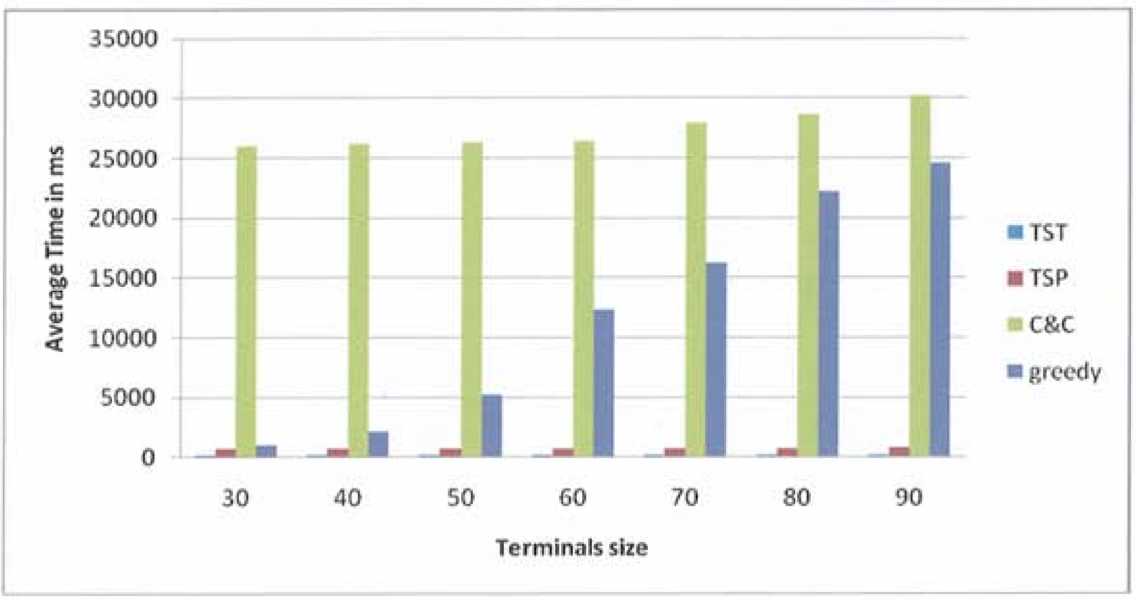


Figure 10. Average run time for graphs of order 300

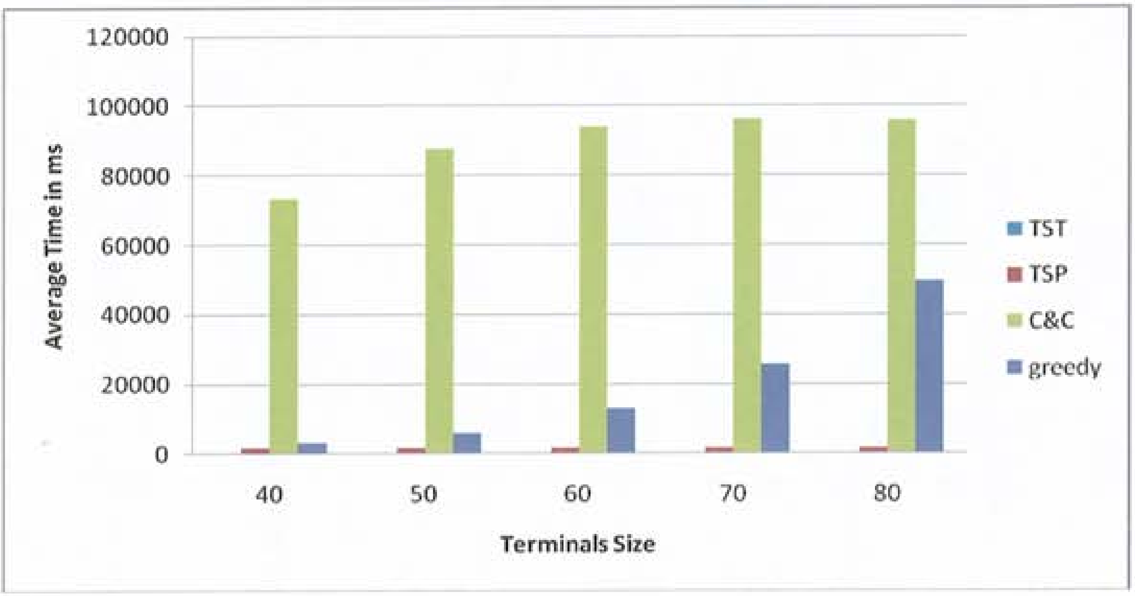


Figure 11. Average run time for graphs of order 400

Chapter 5

Conclusion and Future Work

In this thesis, we introduced an interesting problem that has not received attention from the computer science community thus far. The Euclidean Steiner Walk problem naturally arises in different practical applications. We defined the problem, proved that it is NP complete and presented four different heuristic approaches to the problem. We also developed a tool that allows users to find Steiner walks using any of the proposed approaches. Experimental results are summarized and compared.

A lot of work can be done in the future. One main track is to design exact algorithms for the Euclidean Steiner Walk problem. Another track is to design fixed parameter algorithms for the problem, which seems to be a good candidate for parameterized complexity. In this thesis we restricted our attention to the Euclidean version of Steiner Walk. We believe the general problem is also very important in many practical settings.

References

Altman, M. (1997). The computational complexity of automated redistricting: is automation the answer?. Rutgers Computer & Technology Law Journal.

Chawla S. (2007). Approximation Alogrithms. Lecture Notes.

Christopher, G., Farach, M. and Trick, M. (1996). The structure of circular decomposable metrics. *Algorithms*. (Barcelona), 1136, 486 -500. Springer, Berlin.

Cormen, T., Leiserson, C. E., and Rivest, R. L. (2001). *Introduction to Algorithms* (2nd ed.). Cambridge, McGraw Hill.

Deineko, V. G., Rudolf, R. and.Woeginger, G. J. (1998). Sometimes travelling is easy: the master tour problem. SIAM J. Discrete Math., 11(1), 81- 93.

Downey, R. G. and Fellows, M. R. (1999). *Parameterized Complexity*. Springer-Verlag.

Fomin, F., Grandoni, F. and Kratsch, D. (2008). Faster Steiner Tree Computation in Polynomial-Space. ESA. 430-441.

Fuchs, B. and Kern, W. and Mölle, D. and Richter, S. and Rossmanith, P. and Wang, Xinhui (2007) Dynamic programming for minimum steiner trees. Theory of Computing Systems, 41 (3). 493-500.

Garey, M. and Johnson, D. (1983). *Computers and Intractability: A Guide to the Theory of NP-completeness.* San Francisco, W. H. Freeman and Company.

Halperin, S. (1996). An Optimal Randomized Logarithmic Time Connectivity Algorithm for the EREW PRAM. Journal of Computer and System Sciences.

Hisashi H. (2006). Fitness function for finding out robust solutions on time-varying functions. Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO 06 GECCO.

Hu, J. (2001). A survey on multi-net global routing for integrated circuits. Integration, the VLSI Journal.

Kagaris, D. "Graph Theory", Wiley Encyclopedia of Electrical and Electronics Engineering.

Kokash, N. (2006). *An introduction to heuristic algorithms.* University of Trento, Italy.

Misevičius, A., Blažauskas, T., Blonskis, J. and Smolinskas, J. (2004). An Overview of

Some Heuristic Algorithms for Combinatorial Optimization Problems. Informacines technologijos Ir Valdymas,21-31.

Niedermeier, R. (2006). *Invitation to Fixed-Parameter Algorithms*. Oxford University Press.

Ribeiro, C.(2005). A Hybrid GRASP with Perturbations for the Steiner Problem in Graphs. INFORMS Journal on Computing.

Roughgarden, T. (2006). On the severity of Braess's Paradox: Designing networks for selfish users is hard. Journal of Computer and System Sciences, 2.

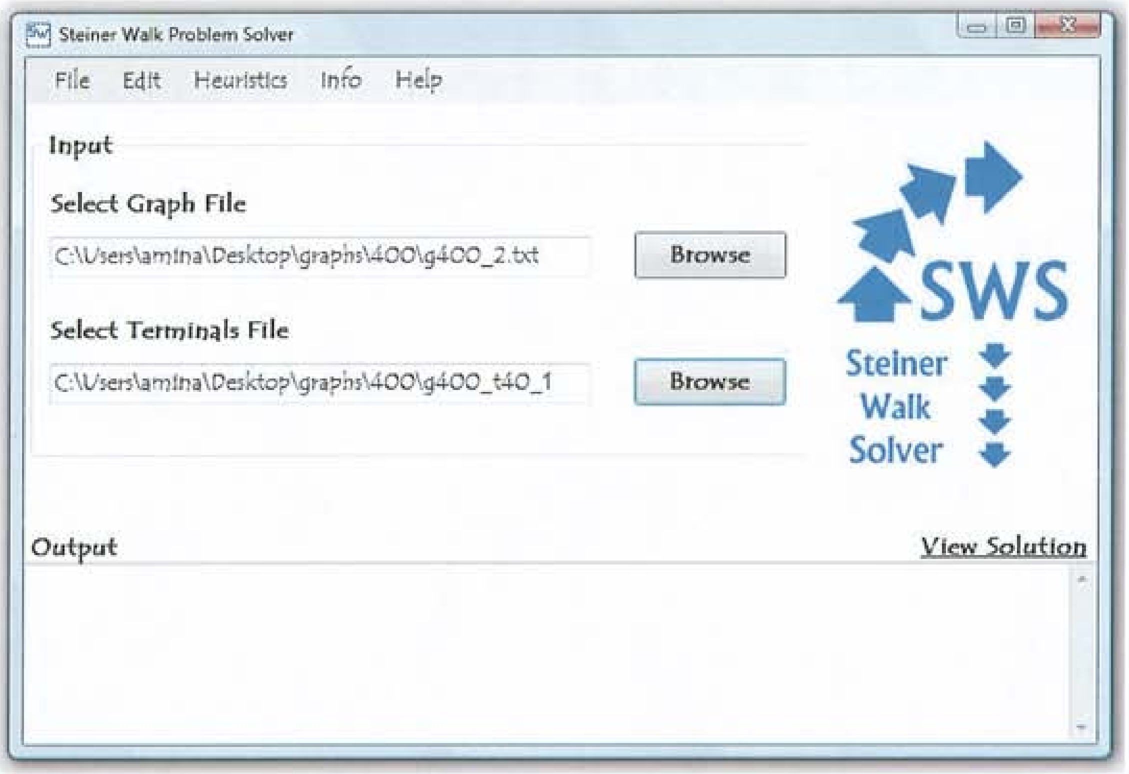
Takahashi, H. and Matsuyama, A. (1980). An approximated solution for the Steiner tree problem in graphs. Math. Japonica 254, 573–577.

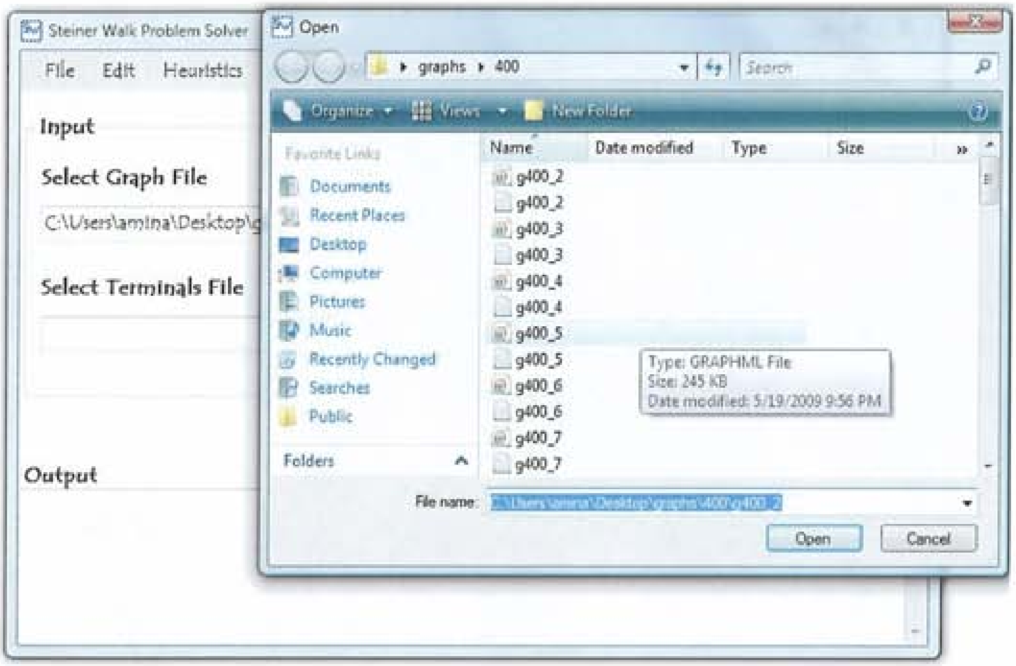
Verhoeven, M.. Severens, M. and Aarts, E. (1996). Local search for Steiner trees in graphs. In V.J.Rayward-Smith et al., editor, Modern Heuristics Search Methods, 117-129. John Wiley and Sons.

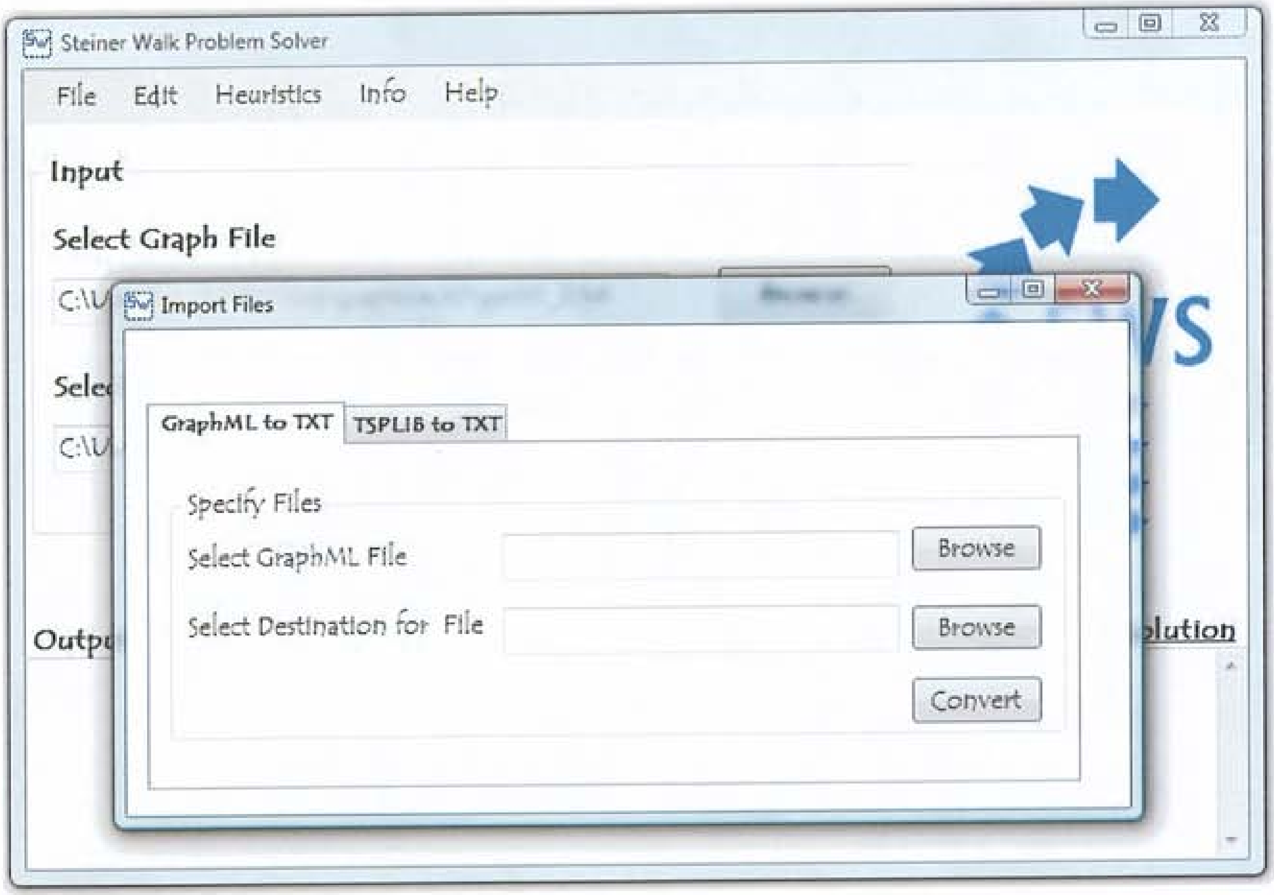
Wang, X. (2008). Exact Algorithms for the Steiner Tree Problem. Ph.D Dissertation.

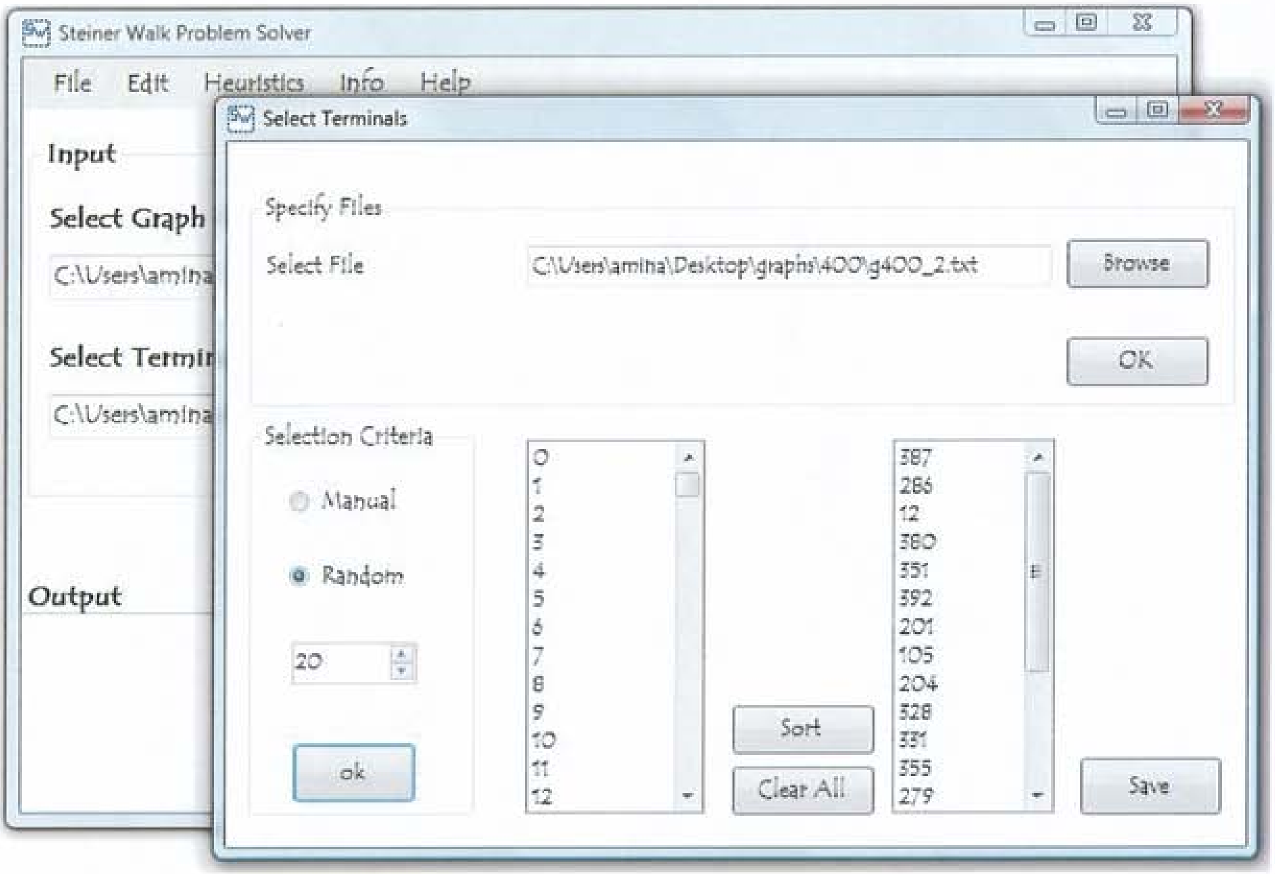
Zhenglong W. (2004). Using genetic algorithm for weighted fuzzy rule-based system. Fifth World Congress on Intelligent Control and Automation (IEEE Cat No 04EX788).

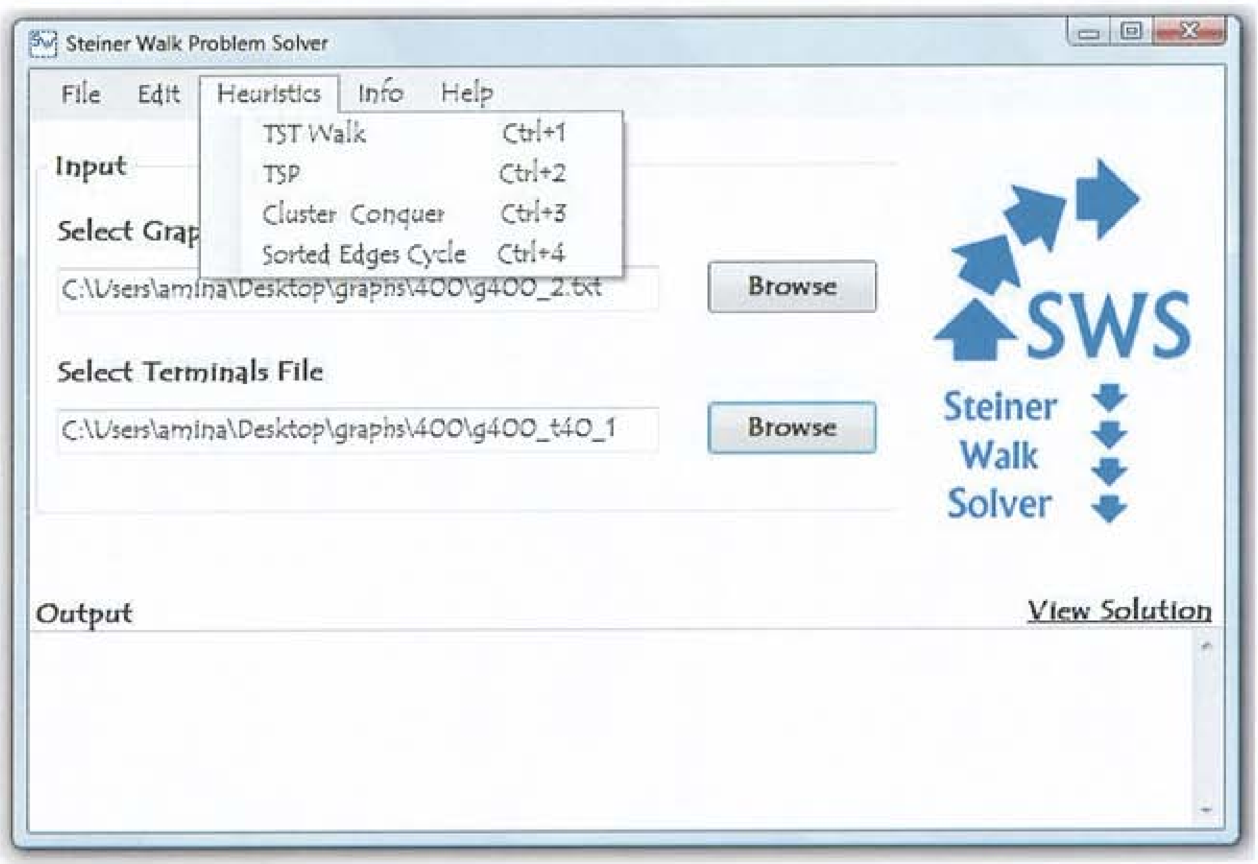
Appendix A

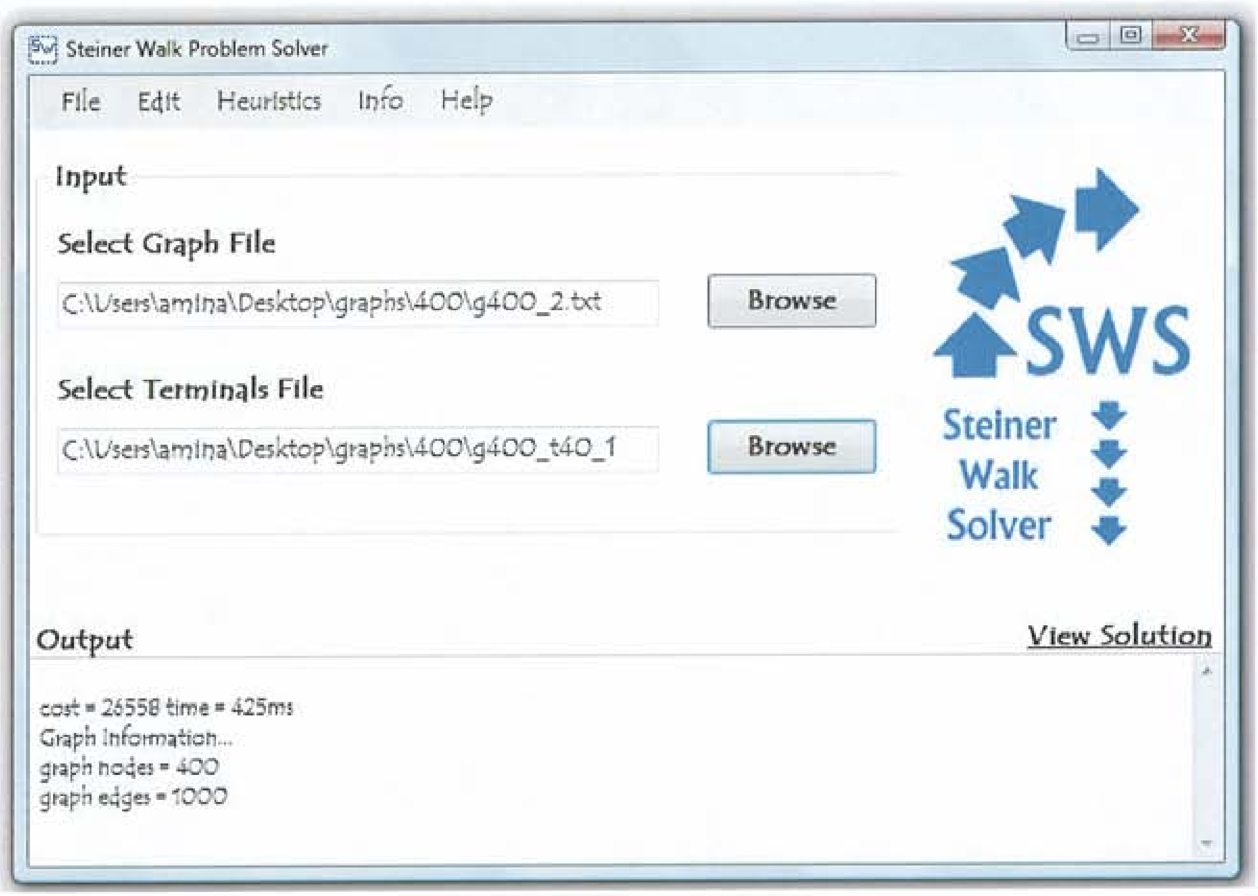

Steiner Walk Problem Solver
File Edit Heuristics Info Help
Input
Select Graph File
C:\Users\amina\Desktop\graphs\400\g400_2.txt
Browse
Select Terminals File
C:\Users\amina\Desktop\graphs\400\g400_t40_1
Browse
SWS
Steiner
Walk
Solver
Output
View Solution


Steiner Walk Problem Solver
File Edit Heuristics
Input
Select Graph File
Select Terminals File
Output
Open
graphs ▸ 400
Search
Organize
Views
New Folder
Favorite Links
Documents
Recent Places
Desktop
Computer
Pictures
Music
Recently Changed
Searches
Public
Folders
Name
Date modified
Type
Size
g400_2
g400_2
g400_3
g400_3
g400_4
g400_4
g400_5
g400_5
g400_6
g400_6
g400_7
g400_7
Type: GRAPHML File
Size: 245 KB
Date modified: 5/19/2009 9:56 PM
File name:
Open
Cancel


Steiner Walk Problem Solver
File
Edit
Heuristics
Info
Help
Input
Select Graph File
Import Files
GraphML to TXT
TSPLIB to TXT
Specify Files
Select GraphML File
Browse
Select Destination for File
Browse
Convert


Steiner Walk Problem Solver
File Edit Heuristics Info Help
Input
Select Graph
C:\Users\amina
Select Termi
C:\Users\amina
Output
Select Terminals
Specify Files
Select File
C:\Users\amina\Desktop\graphs\400\g400_2.txt
Browse
OK
Selection Criteria
Manual
Random
20
ok
0
1
2
3
4
5
6
7
8
9
10
11
12
Sort
Clear All
387
286
12
380
351
392
201
105
204
328
331
355
279
Save


Steiner Walk Problem Solver
File
Edit
Heuristics
Info
Help
TST Walk Ctrl+1
TSP Ctrl+2
Cluster Conquer Ctrl+3
Sorted Edges Cycle Ctrl+4
Input
Select Grap
C:\Users\amina\Desktop\graphs\400\g400_2.txt
Browse
Select Terminals File
C:\Users\amina\Desktop\graphs\400\g400_t40_1
Browse
SWS
Steiner
Walk
Solver
Output
View Solution


Steiner Walk Problem Solver
File Edit Heuristics Info Help
Input
Select Graph File
C:\Users\amina\Desktop\graphs\400\g400_2.txt
Browse
Select Terminals File
C:\Users\amina\Desktop\graphs\400\g400_t40_1
Browse
SWS
Steiner
Walk
Solver
Output
View Solution
cost = 26558 time = 425ms
Graph Information...
graph nodes = 400
graph edges = 1000

Appendix B

Graph ID	Graph Size (nodes)	No. of Terminals	Cost of Steiner walk given by			
			TST	TSP	CC	RPM
g_200_1	200	10	6707	5255	5108	5089
g_200_1	200	10	9862	8166	7344	6862
g_200_1	200	10	6841	6722	6877	6248
g_200_1	200	10	5026	5369	5188	4375
g_200_1	200	10	9472	6166	6111	5836
g_200_1	200	10	6591	5458	5061	5195
g_200_1	200	10	7485	4706	5638	4427
g_200_1	200	10	11498	8545	7287	6910
g_200_1	200	10	7295	5910	5199	5910
g_200_1	200	10	10120	7769	7022	6878
g_200_1	200	20	13477	10574	9656	8128
g_200_1	200	20	13065	8303	8498	7423
g_200_1	200	20	9833	10539	8500	7825
g_200_1	200	20	16238	10587	11122	8047
g_200_1	200	20	11762	10184	10024	8500
g_200_1	200	20	16856	12059	11009	11055
g_200_1	200	20	15463	10624	7689	8578
g_200_1	200	20	15247	8895	10227	7758
g_200_1	200	20	12865	8376	8193	7481
g_200_1	200	20	16390	11058	11042	9599
g_200_2	200	10	7091	5974	4844	5441
g_200_2	200	10	4780	5424	5415	4568
g_200_2	200	10	6129	6602	5455	4751
g_200_2	200	10	8044	6366	4272	5643

g_200_2	200	10	9996	6680	7630	5272
g_200_2	200	10	6531	5596	7078	5364
g_200_2	200	10	5999	7091	8785	6288
g_200_2	200	10	6701	6721	6468	5441
g_200_2	200	20	10140	7979	6273	8319
g_200_2	200	20	19531	10698	7465	8763
g_200_2	200	20	12732	9188	9416	7708
g_200_2	200	20	6770	7749	5435	6738
g_200_2	200	20	10767	7929	6192	7479
g_200_2	200	20	15268	9990	9080	7950
g_200_2	200	20	15852	11682	7817	9930
g_200_2	200	20	13036	10439	8067	9298
g_200_2	200	20	12117	9530	10151	8035
g_200_2	200	30	18487	12785	11992	10716
g_200_2	200	30	14553	12106	12208	10280
g_200_2	200	30	16407	13550	12026	11331
g_200_2	200	30	20078	12241	9689	10624
g_200_2	200	30	15474	11825	12869	10658
g_200_2	200	30	18908	12631	9945	10927
g_200_2	200	30	19199	11882	11155	10000
g_200_2	200	30	20509	13873	12330	11147
g_200_2	200	30	19946	12713	12872	10973
g_200_2	200	40	25217	16751	10653	13774
g_200_2	200	40	25010	15295	12456	13076
g_200_2	200	40	21152	15038	14318	12632
g_200_2	200	40	16674	13892	12340	11918
g_200_2	200	40	21864	17492	15926	15893
g_200_2	200	50	22735	16631	10171	14708
g_200_2	200	50	25645	17614	15450	15011
g_200_2	200	50	27108	17506	12028	14475
g_200_2	200	50	26162	16770	12632	15174
g_200_2	200	50	32760	16970	13339	14806
g_200_2	200	50	29341	18629	16340	16077
g_200_2	200	50	25947	16245	12411	14353

g_200_2	200	50	25843	17307	15499	15524
g_200_2	200	50	35546	18008	13028	13697
g_400_2	400	40	26558	19530	17241	16806
g_400_2	400	40	35075	20466	18410	18620
g_400_2	400	40	32084	20240	17720	17738
g_400_2	400	40	24522	19196	19057	16531
g_400_2	400	40	30842	20975	17730	18074
g_400_2	400	40	32867	19843	19261	17051
g_400_2	400	40	31233	21398	20002	18192
g_400_2	400	50	40639	24878	21979	21438
g_400_2	400	50	35747	24310	23454	22206
g_400_2	400	50	37821	28745	26462	24388
g_400_2	400	50	39257	21124	20947	19767
g_400_2	400	50	34081	23508	13522	21594
g_400_2	400	60	42891	30927	23794	27586
g_400_2	400	60	35223	23120	21477	20836
g_400_2	400	60	44787	25604	23619	23498
g_400_2	400	60	39687	28929	25091	26775
g_400_2	400	60	40368	27539	26983	23559
g_400_2	400	60	40368	26029	23797	23109
g_400_2	400	60	40120	25924	27110	24221
g_400_2	400	60	37874	29082	30110	24979
g_400_2	400	70	41799	28121	22036	26013
g_400_2	400	70	45011	31624	21103	26804
g_400_2	400	70	50138	35224	30280	31701
g_400_2	400	70	46791	28476	24537	25554
g_400_2	400	70	44042	29464	30453	26130
g_400_2	400	80	45194	30732	25843	27876
g_400_2	400	80	51866	37482	33163	33180
g_400_2	400	80	52420	33444	34712	30890
g_400_2	400	80	52835	35918	27034	31813
g_400_2	400	80	49278	32190	29340	29914
g_400_4	400	40	24536	17929	13592	14707

g_400_4	400	40	34184	19718	14889	17703
g_400_4	400	40	25794	19712	19432	17740
g_400_4	400	40	29872	20199	15316	17796
g_400_4	400	40	25948	16687	11667	15280
g_400_4	400	40	25338	16658	14086	15243
g_400_4	400	40	30990	18165	12045	15194
g_400_4	400	40	25405	19691	12017	17059
g_400_4	400	40	22934	16700	13287	13913
g_400_4	400	40	31134	19585	16266	17595
g_400_4	400	50	33966	20848	17041	18126
g_400_4	400	50	31061	20856	18870	17909
g_400_4	400	50	35656	21191	18294	18895
g_400_4	400	50	38268	22310	19116	20736
g_400_4	400	50	35901	20508	16394	18516
g_400_4	400	50	41915	22321	18986	19465
g_400_4	400	50	29781	19887	18102	18677
g_400_4	400	50	36642	22739	15442	19962
g_400_4	400	50	28392	18173	16378	16719
g_400_4	400	50	27606	18745	14806	17543
g_400_4	400	60	37133	24587	19259	21235
g_400_4	400	60	36398	21269	16693	18803
g_400_4	400	60	32777	20196	14374	17765
g_400_4	400	60	42671	26574	19189	23740
g_400_4	400	60	39193	23635	21452	22169
g_400_4	400	60	38728	23177	17703	21731
g_400_4	400	60	38280	28082	11780	23683
g_400_4	400	60	43499	24765	22630	21083
g_400_4	400	60	36924	23979	14502	20982
g_400_4	400	60	41438	23946	22508	20288
g_400_4	400	70	39804	24310	16729	21120
g_400_4	400	70	47500	29544	25919	24587
g_400_4	400	70	44035	25757	17499	21936
g_400_4	400	70	41295	25046	17349	21708
g_400_4	400	70	40952	28447	21793	25703

g_400_4	400	70	31180	24005	19834	22021
g_400_4	400	70	41373	25415	19673	22798
g_400_4	400	70	47296	28539	24579	24865
g_400_4	400	70	50262	31396	27146	26329
g_400_4	400	70	40535	25502	16894	23355
g_400_4	400	80	51307	30160	26739	26508
g_400_4	400	80	47315	29046	19234	25028
g_400_4	400	80	50580	30104	26395	26650
g_400_4	400	80	42999	29063	21470	25546
g_400_4	400	80	41333	25844	26233	23812
g_400_4	400	80	39221	27489	21438	24303
g_400_4	400	80	45481	28297	22499	25011
g_400_4	400	80	43647	28229	20220	25719
g_400_4	400	80	49443	32010	27143	28188
g_400_4	400	80	45749	30732	24694	27486
g_400_4	400	90	51781	30224	20568	28591
g_400_4	400	90	50616	31735	27800	29308
g_400_4	400	90	48006	30974	22924	27570
g_400_4	400	90	57177	33634	28369	29270
g_400_4	400	90	54998	32152	25899	28456
g_400_4	400	90	44656	29058	22474	26199
g_400_4	400	90	52337	31065	18728	27328
g_400_4	400	90	17943	17943	17943	26246
g_400_4	400	90	44456	31016	23621	27544
g_400_4	400	90	43798	31888	25268	29494
g_400_5	400	40	30781	18626	17273	15673
g_400_5	400	40	22668	17523	13730	15756
g_400_5	400	40	23634	16964	11795	15180
g_400_5	400	40	23730	18332	16867	16510
g_400_5	400	40	32547	17999	13250	16341
g_400_5	400	50	32037	23794	17517	20172
g_400_5	400	50	29495	19942	16303	18392
g_400_5	400	50	32850	21861	18198	19667
g_400_5	400	50	28602	22103	16230	19236

g_400_5	400	50	32899	24693	19752	21173
g_400_5	400	60	27430	20069	14751	17992
g_400_5	400	60	36723	24925	21100	21746
g_400_5	400	60	41278	26322	19178	21979
g_400_5	400	60	49038	27776	18282	25424
g_400_5	400	60	36142	23914	16300	20185
g_400_5	400	70	35539	29558	24713	25477
g_400_5	400	70	45471	28413	22917	24800
g_400_5	400	70	50282	30056	27088	27373
g_400_5	400	70	46820	30120	23168	26436
g_400_5	400	70	48544	30108	25922	27501
g_400_5	400	80	58750	32183	30050	28513
g_400_5	400	80	45301	28458	20949	25642
g_400_5	400	80	41905	27963	26301	24881
g_400_5	400	80	39596	28943	22436	26727
g_400_5	400	80	53337	32091	25700	29208
g_400_5	400	90	58408	30169	20551	27871
g_400_5	400	90	50504	31084	22295	26962
g_400_5	400	90	50617	31902	28305	29016
g_400_5	400	90	48855	32315	22940	31072
g_400_5	400	90	54497	30949	28208	28655
g_400_6	400	40	30035	18254	11760	16693
g_400_6	400	40	34631	20908	15900	17977
g_400_6	400	40	28651	19311	19396	16524
g_400_6	400	40	26257	19597	12981	17171
g_400_6	400	40	29488	20660	14339	17949
g_400_6	400	50	28638	21529	17431	18801
g_400_6	400	50	36136	23643	18780	21203
g_400_6	400	50	26470	18028	13471	16926
g_400_6	400	50	39355	21502	19958	18053
g_400_6	400	50	33978	21488	19410	19386
g_400_6	400	60	36435	24366	17285	22555
g_400_6	400	60	30549	22283	16051	19094
g_400_6	400	60	37516	23207	18133	20722

g_400_6	400	60	40461	25924	18743	23107
g_400_6	400	60	36317	22569	17959	20670
g_400_6	400	70	42389	24546	19896	22086
g_400_6	400	70	44830	27683	22898	23369
g_400_6	400	70	46645	26600	20708	23634
g_400_6	400	70	44528	26114	25824	24041
g_400_6	400	70	44748	25554	18970	22411
g_400_6	400	80	45504	30051	25255	25626
g_400_6	400	80	37504	28038	22093	24636
g_400_6	400	80	43998	29089	25787	26090
g_400_6	400	80	42553	27159	22325	23454
g_400_6	400	80	45915	31972	25694	28142
g_400_6	400	90	53037	34367	29097	30985
g_400_6	400	90	45514	30710	23253	28252
g_400_6	400	90	52370	33666	32762	28441
g_400_6	400	90	46226	30451	22826	26979
g_400_6	400	90	43117	28872	20384	26325
g_400_7	400	40	23377	15600	16495	13887
g_400_7	400	40	27741	18063	15036	16194
g_400_7	400	40	25619	18805	16620	15520
g_400_7	400	40	27746	15927	12664	14512
g_400_7	400	40	32881	19731	17584	17286
g_400_7	400	50	31170	17896	12215	15966
g_400_7	400	50	38570	19464	14851	18016
g_400_7	400	50	33773	21560	14993	18520
g_400_7	400	50	30879	20207	15521	16145
g_400_7	400	50	35729	19590	18008	18401
g_400_7	400	60	39716	26104	14595	23106
g_400_7	400	60	34191	24690	16757	21584
g_400_7	400	60	37951	22724	20763	20788
g_400_7	400	60	36026	24215	20015	22895
g_400_7	400	60	38341	20889	16815	18456
g_400_7	400	70	38917	23378	18954	21423
g_400_7	400	70	40960	24526	17530	22028

g_400_7	400	70	42557	23498	23007	21419
g_400_7	400	70	38918	26247	20373	23657
g_400_7	400	70	50109	27859	22404	24436
g_400_7	400	80	40574	25040	21247	23206
g_400_7	400	80	39776	26352	17516	23773
g_400_7	400	80	40731	27035	20914	24266
g_400_7	400	80	45620	27806	22654	25420
g_400_7	400	80	38996	27991	20342	25429
g_300_1	300	30	23989	14785	14742	12945
g_300_1	300	30	25294	16321	15956	13970
g_300_1	300	30	23393	15572	12082	12944
g_300_1	300	30	14485	14500	13427	12400
g_300_1	300	30	18394	12449	11453	10875
g_300_1	300	30	16079	13609	10762	11480
g_300_1	300	30	23561	13581	13942	12326
g_300_1	300	30	20981	14868	12535	13107
g_300_1	300	30	17479	14855	12398	11803
g_300_1	300	30	26001	16255	14083	14200
g_300_1	300	40	29883	19655	14337	16371
g_300_1	300	40	25402	16482	15176	14249
g_300_1	300	40	25317	18489	14987	15735
g_300_1	300	40	24977	16820	16003	14612
g_300_1	300	40	29357	17759	12492	14260
g_300_1	300	40	28789	17385	16414	14930
g_300_1	300	40	27340	15506	17263	15412
g_300_1	300	40	32135	17274	16012	15121
g_300_1	300	50	26821	19909	16313	14295
g_300_1	300	50	28845	20642	13728	18355
g_300_1	300	50	30892	19194	12663	17800
g_300_1	300	50	28833	20158	13931	18324
g_300_1	300	50	31566	20712	17462	18663
g_300_1	300	50	29775	22062	18248	19073
g_300_1	300	50	29572	21390	16761	17923
g_300_1	300	50	31212	21198	17121	19616

g_300_1	300	50	31546	19655	15992	17139
g_300_1	300	50	35687	22814	17671	20027
g_300_1	300	60	29037	22234	15164	18775
g_300_1	300	60	37706	25962	18797	22900
g_300_1	300	60	33717	23516	18138	21123
g_300_1	300	60	30967	22415	17285	19413
g_300_1	300	60	32224	24353	18745	21468
g_300_1	300	60	34096	23563	22343	20908
g_300_1	300	60	28379	21661	19403	19511
g_300_1	300	60	32812	20615	15532	19058
g_300_1	300	60	38222	25404	17752	22279
g_300_1	300	60	37964	24761	16821	21417
g_300_1	300	70	34262	22553	18693	20130
g_300_1	300	70	41158	24384	15042	22476
g_300_1	300	70	40749	26482	23659	24206
g_300_1	300	70	32494	27505	16430	24773
g_300_1	300	70	34992	21592	13176	19270
g_300_1	300	70	29912	24829	21850	22557
g_300_1	300	70	41776	24858	19074	21612
g_300_1	300	70	36315	27306	26953	23784
g_300_1	300	70	40745	25162	19043	23136
g_300_1	300	70	34042	25130	25785	21891
g_300_1	300	80	36791	25970	21058	23185
g_300_1	300	80	41555	29608	24089	26506
g_300_1	300	80	41874	28707	23254	26472
g_300_1	300	80	38464	26941	14866	23993
g_300_1	300	80	42434	27329	25975	24868
g_300_1	300	80	43002	27559	23042	23984
g_300_1	300	80	41989	27676	21879	24048
g_300_1	300	80	39572	25401	14617	22188
g_300_1	300	80	42000	27187	21538	23936
g_300_1	300	90	50977	26875	25168	25216
g_300_1	300	90	48106	29258	24545	26649
g_300_1	300	90	37725	29472	21379	26935

g_300_1	300	90	43558	28584	23107	25771
g_300_1	300	90	47357	27116	21996	24075
g_300_1	300	90	39976	26023	18932	23610
g_300_1	300	90	42455	27218	21789	23892
g_300_1	300	90	50068	31470	22231	27666
g_300_1	300	90	47549	31665	25426	28513
g_300_1	300	90	42424	25985	21138	23166
g_300_2	300	30	26890	15488	15102	14383
g_300_2	300	30	28281	17567	13428	14510
g_300_2	300	30	18152	16000	15361	13505
g_300_2	300	30	21137	14393	11617	11902
g_300_2	300	30	20982	10608	11910	9036
g_300_2	300	30	23892	14293	12428	12407
g_300_2	300	30	24693	16520	14987	14124
g_300_2	300	30	17108	12770	11627	11594
g_300_2	300	30	21887	14424	12635	11881
g_300_2	300	30	21836	16441	11631	14498
g_300_2	300	40	19618	14993	11814	12838
g_300_2	300	40	22049	17983	14580	15822
g_300_2	300	40	26194	15472	13868	13524
g_300_2	300	40	28780	19238	18378	15854
g_300_2	300	40	27128	18398	18434	15724
g_300_2	300	40	19814	16500	11759	14514
g_300_2	300	40	23276	16415	17699	14561
g_300_2	300	40	21721	16638	15017	14854
g_300_2	300	40	27919	17462	13480	15339
g_300_2	300	40	28983	18070	15040	16485
g_300_2	300	50	27479	18115	14810	16337
g_300_2	300	50	30368	20460	17226	18925
g_300_2	300	50	28022	20265	18151	17437
g_300_2	300	50	34562	20766	16773	18596
g_300_2	300	50	31589	19320	17458	18063
g_300_2	300	50	30442	20111	18115	17550
g_300_2	300	50	34564	22237	16593	18271

g_300_2	300	50	37929	24367	<u>20590</u>	20352
g_300_2	300	50	34599	22560	<u>17166</u>	19181
g_300_2	300	50	28815	19319	17785	<u>17483</u>
g_300_2	300	60	29532	24008	<u>21111</u>	21168
g_300_2	300	60	30688	19411	20954	<u>17915</u>
g_300_2	300	60	33317	26242	<u>22741</u>	23000
g_300_2	300	60	37017	24783	<u>17483</u>	21737
g_300_2	300	60	31503	21402	<u>12906</u>	19083
g_300_2	300	60	31175	22732	19538	<u>18902</u>
g_300_2	300	60	36420	26170	21969	<u>20853</u>
g_300_2	300	60	35452	24446	<u>17061</u>	21465
g_300_2	300	60	35930	21896	21285	<u>20203</u>
g_300_2	300	60	41231	24430	<u>19368</u>	21217
g_300_2	300	70	36398	25718	<u>21829</u>	23349
g_300_2	300	70	33678	24346	22879	<u>21629</u>
g_300_2	300	70	42011	25700	<u>21017</u>	23397
g_300_2	300	70	38124	23338	23497	<u>21482</u>
g_300_2	300	70	41359	24293	23837	<u>21786</u>
g_300_2	300	70	41648	22852	<u>14599</u>	21602
g_300_2	300	70	35140	27629	<u>17572</u>	21602
g_300_2	300	70	42544	26390	<u>21056</u>	23450
g_300_2	300	70	32077	24971	<u>20004</u>	22871
g_300_2	300	70	42609	27005	<u>21014</u>	23626
g_300_2	300	80	37257	24762	<u>16360</u>	21669
g_300_2	300	80	39933	23552	<u>18179</u>	20998
g_300_2	300	80	42457	26046	<u>18263</u>	22811
g_300_2	300	80	46795	27343	<u>22516</u>	23846
g_300_2	300	80	35629	24248	<u>18452</u>	22064
g_300_2	300	80	40860	27218	<u>21079</u>	23946
g_300_2	300	80	34919	25553	<u>21565</u>	21915
g_300_2	300	80	40780	28822	<u>21044</u>	25225
g_300_2	300	80	35395	25503	<u>20521</u>	22200
g_300_2	300	80	44909	26518	<u>17671</u>	23083
g_300_2	300	90	42036	28865	<u>22123</u>	26759

g_300_2	300	90	42000	28761	26353	25049
g_300_2	300	90	47422	30583	24089	26406
g_300_2	300	90	45638	28285	23530	25436
g_300_2	300	90	47325	29904	26415	26507
g_300_2	300	90	44415	27071	25555	23924
g_300_2	300	90	36085	25230	16633	22841
g_300_2	300	90	46865	30537	23661	27344
g_300_2	300	90	42671	29071	24751	25870
g_300_2	300	90	38057	26833	22819	23681
g_300_3	300	30	20915	17746	17016	15453
g_300_3	300	30	30736	18489	16984	15610
g_300_3	300	30	18535	14115	11208	10922
g_300_3	300	30	20894	15402	14346	13788
g_300_3	300	30	22550	13568	13343	12321
g_300_3	300	30	21129	14612	10572	12325
g_300_3	300	30	24189	16383	12457	14031
g_300_3	300	30	20416	14033	12891	12551
g_300_3	300	30	25493	16934	14405	14437
g_300_3	300	40	29205	20477	14923	17183
g_300_3	300	40	26482	19092	17171	15792
g_300_3	300	40	26441	16113	14366	14734
g_300_3	300	40	25358	16966	14935	15529
g_300_3	300	40	27255	19817	17254	16790
g_300_3	300	40	26696	19653	17812	16847
g_300_3	300	40	25364	18334	13987	16013
g_300_3	300	40	26141	18300	9508	15774
g_300_3	300	40	33396	19639	18187	16752
g_300_3	300	40	21114	18618	12021	16391
g_300_3	300	50	34778	22079	14211	19090
g_300_3	300	50	31551	21790	18161	18824
g_300_3	300	50	26700	20572	16593	18294
g_300_3	300	50	27924	21500	14420	18993
g_300_3	300	50	29054	20944	13943	17266
g_300_3	300	50	29238	21230	15062	18216

g_300_3	300	50	30173	16976	16456	15158
g_300_3	300	50	31055	22956	19336	20580
g_300_3	300	50	22387	18205	13285	16628
g_300_3	300	50	36676	23632	18350	21056
g_300_3	300	60	28850	20425	14384	17941
g_300_3	300	60	33657	21148	17294	17959
g_300_3	300	60	32172	23111	17277	20463
g_300_3	300	60	37516	22893	11988	21363
g_300_3	300	60	35372	23910	19158	20420
g_300_3	300	60	40247	22683	19088	20092
g_300_3	300	60	27981	24418	20033	21076
g_300_3	300	60	43361	24278	17734	21656
g_300_3	300	60	34306	22634	17389	21287
g_300_3	300	60	39442	23995	17330	21203
g_300_3	300	70	40090	25822	15732	22619
g_300_3	300	70	46336	26616	20076	22563
g_300_3	300	70	40985	26666	22051	23311
g_300_3	300	70	33810	25322	24168	22706
g_300_3	300	70	42844	24919	23702	22284
g_300_3	300	70	35960	23993	18812	21026
g_300_3	300	70	37307	24081	19813	20987
g_300_3	300	70	34779	25011	14052	23122
g_300_3	300	70	39951	23887	20861	21511
g_300_3	300	70	36229	23254	14369	21315
g_300_3	300	80	40703	27496	17251	24836
g_300_3	300	80	44637	27353	23399	25915
g_300_3	300	80	46819	28368	25794	25812
g_300_3	300	80	36862	26655	14026	23604
g_300_3	300	80	33961	24418	22022	22650
g_300_3	300	80	41811	27887	23989	25642
g_300_3	300	80	40177	26093	18930	23813
g_300_3	300	80	42037	26344	22278	24271
g_300_3	300	80	44158	27264	18654	25704
g_300_3	300	80	44124	28560	24129	25504

g_300_3	300	90	51300	29877	25386	27546
g_300_3	300	90	40732	27861	23849	25793
g_300_3	300	90	42520	30043	23077	26673
g_300_3	300	90	40095	28836	24331	26157
g_300_3	300	90	50170	31388	24165	27086
g_300_3	300	90	43308	28178	24776	25349
g_300_3	300	90	43216	28436	23062	25166
g_300_3	300	90	45771	29433	22785	26559
g_300_3	300	90	45311	26891	20898	23984
g_300_3	300	90	44186	28628	25584	26000