ABSTRACT

Title of dissertation: MODELING AND SOLVING ARC ROUTING PROBLEMS IN STREET SWEEPING AND SNOW PLOWING

Benjamin Dussault, Doctor of Philosophy, 2012

Dissertation directed by: Professor Bruce Golden
R.H. Smith School of Business

In arc routing problems, the goal is to determine an optimal path, or set of paths, that traverse a required subset of arcs on a graph with respect to a set of constraints and objective function. The Chinese Postman Problem (CPP) forms the basis for many arc routing problems. Let graph $G = (V, A)$, where V is a set of vertices and $A = \{(i, j)|i, j \in V\}$ is a set of arcs that each connect exactly two vertices, each with its own cost of traversal c_{ij}. The objective of the CPP is to construct a least cost path that traverses each arc in A at least once.

There are many practical applications for variants of the CPP, including winter street maintenance, and street sweeping that incorporate:

Rural Instances Rural Postman Problems (RPP) stipulate that only a subset $A_R \subset A$ require traversal, allowing for non-servicing traversal on the rest of the graph. In the context of street sweeping, a street sweeper isn't responsible for sweeping all the streets.

Windy Graphs In the CPP, the cost of traversal of an arc is the same, regardless of the direction of traversal. In the Windy Postman Problem (WPP), the cost of traversal is asymmetric. That is, it is possible for $c_{ij} \neq c_{ji}$. In the context of snow plowing, it is harder to plow uphill than downhill.

Multi-Vehicle Instead of a single vehicle with a single tour, multiple tours are found for multiple vehicles. This is often accompanied with an objective function that seeks to minimize the cost of the largest cost route. This is motivated by practical applications, which seek to balance the cost of each route. In the case where route cost is measured in time, route balancing minimizes, for example, paid overtime.

Turn Penalties UPS reported that it saved three million gallons of gasoline annually by avoiding unnecessary left-hand turns, which take longer to perform than going straight or turning right. Instances with turn penalties incorporate costs of turning, in addition to costs of traversal.

The Windy Postman Problem (WPP) incorporates windy graphs and the Rural Postman Problem (RPP) incorporates rural instances. The RPP can be extended to include turn penalties (RPPTP). The Windy Rural Postman Problem (WRPP) incorporates instances that are both windy and rural. The WRPP can be extended to the MM k-WRPP which adds k plows. In this dissertation, we extend these variants to new problems with new problem attributes that are practically motivated. Our new attributes are listed below.

Multi-Period The CPP solves for a single route, which can be interpreted to be traversed in a single day. It is possible that the set of required arcs is too long to service in a single day and therefore must be split among multiple days. In this case, we need to decide which day to assign service to each arc, before routing can take place.

Downhill Instances In street snow plowing, it is faster to deadhead (traverse without servicing) a street rather than plowing it. In this case, there are different costs for deadheading and plowing a street. Moreover, it takes longer to plow uphill,

resulting in four costs: plowing uphill, plowing downhill, deadheading uphill, and deadheading downhill.

Precedence When considering downhill instances, the snow may be so deep that it is impossible for a snowplow to deadhead a street before the street is plowed.

In this dissertation we present a variety of heuristics to solve these problems, all adaptations of the concept of cycle permutation based on Euclidean cycle decomposition. To our knowledge, the use of moving or permuting sub-cycles as a way to change and improve a Eulerian cycle is novel and we show that it is very robust at improving solutions.

MODELING AND SOLVING ARC ROUTING PROBLEMS IN STREET SWEEPING AND SNOW PLOWING

by

Benjamin Dussault

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2012

Advisory Committee:
Professor Bruce Golden, Chair/Advisor
Professor Edward Wasil
Professor Denny Gulick
Professor Radu Balan
Professor Paul Schonfeld

Dedication

I would like to thank my parents for encouraging me, since I was a small child, to pursue this degree, and I am truly grateful for their support and advice during my studies.

This dissertation, as well my my entire study of operations research, would not have been possible without my advisor, Dr. Bruce Golden. It was serendipity that I took his class, but Dr. Golden eloquently introduced me to the surprisingly difficult vehicle routing problem and immediately captured my interest. In addition to being an excellent teacher, he is an excellent advisor, and his expertise in the subject has been invaluable during my work.

Dr. Edward Wasil has thoroughly and thoughtfully edited each of the papers that we have worked on together and has single-handedly improved the clarity and style of my writing more than anything else. Despite nearly four years working together, my prose is still overly brief, but I hope my sincere appreciation is not. I thank Drs. Radu Balan, Denny Gulick, and Paul Schonfeld for taking the time to serve on my dissertation committee, Alverda McCoy for saving me multiple times from administrative disasters, and Dr. Larry Levy for providing test instances for some of my work.

Finally, and most of all, I thank my wife, Christine.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CPP	Chinese Postman Problem
RPP	Rural Postman Problem
DRPP	Directed Rural Postman Problem
RPP-TP	Rural Postman Problem with Turn Penalties
WPP	Windy Postman Problem
WRPP	Windy Rural Postman Problem
SPP	Street Sweeper Problem
m-PP	m-Plowing Problem
C-PP	Capcitated Plow Problem
SRAM	Snow Removal Asset Management
GA	Genetic Algorithm
LS	Local Search
CPLEX	A commercial Integer Programming Solver
DPP	Downhill Plow Problem
DPP-TP	Downhill Plow Problem with Turn Penalties
RDPP-TP	Rural Downhill Plow Problem with Turn Penalties
CARP	Capacitated Arc Routing Problem
WRPP	Windy Rural Postman Problem
k-WRPP	k-plow Windy Rural Postman Problem
RPPTP	Rural Postman Problem with Turn Penalties
MM k-DPP	Min-Max Downhill Plow Problem with Multiple Plows
MM k-WRPP	Min-Max Windy Rural Postman Problem
PPP	Plowing with Precedence Problem
MM k-DPPE	Min-Max Extended Downhill Plow Problem
IPX	IP formulation of the DPP
IPXE	IP formulation of the DPP-TP
CPLS	Cycle Permutation Local Search
CPLS-M	Cycle Permutation Local Search for Multiple Plows
CPLS-RHT	Cycle Permutation Local Search for Right Hand Turns

Chapter 1

Introduction

In arc routing problems, the goal is to determine an optimal path, or set of paths, that traverse a required subset of arcs on a graph with respect to a set of constraints and objective function. The Chinese Postman Problem (CPP) forms the basis for many arc routing problems. Let graph $G = (V, A)$, where V is a set of vertices and $A = \{(i, j)|i, j \in V\}$ is a set of arcs that each connect exactly two vertices, each with its own cost of traversal c_{ij}. The objective of the CPP is to construct a least cost path that traverses each arc in A at least once.

There are many practical applications for variants of the CPP, including winter street maintenance [48, 49, 50, 51] and garbage collection [30, 19]. The topic of arc routing has been studied extensively and there exist many comprehensive literature reviews, such as Eiselt et al. [26, 27] and Eksioglua et al. [28]. We summarize problem attributes that are used in variants of the CPP.

Rural Instances Rural Postman Problems (RPP) stipulate that only a subset $A_R \subset A$ require traversal, allowing for non-servicing traversal on the rest of the graph. A review is given by Eiselt et al. [27].

Windy Graphs In the CPP, the cost of traversal of an arc is the same, regardless of the direction of traversal. In the Windy Postman Problem (WPP), the cost of traversal is asymmetric. That is, it is possible for $c_{ij} \neq c_{ji}$.

Multi-Vehicle Instead of a single vehicle with a single tour, multiple tours are found

for multiple vehicles. This is often accompanied with an objective function that seeks to minimize the cost of the largest cost route. This is motivated by practical applications, which seek to balance the cost of each route. In the case where route cost is measured in time, route balancing minimizes, for example, paid overtime.

Turn Penalties UPS reported that it saved three million gallons of gasoline annually by avoiding unnecessary left-hand turns [43], which take longer to perform than going straight or turning right. Instances with turn penalties incorporate costs of turning, in addition to costs of traversal.

The Windy Postman Problem (WPP) incorporates windy graphs and the Rural Postman Problem (RPP) incorporates rural instances. The RPP can be extended to include turn penalties (RPPTP). The Windy Rural Postman Problem (WRPP) incorporates instances that are both windy and rural. The WRPP can be extended to the MM k-WRPP which adds k plows. In this dissertation, we extend these variants to new problems with new problem attributes that are practically motivated. Our new attributes are listed below.

Multi-Period The CPP solves for a single route, which can be interpreted to be traversed in a single day. It is possible that the set of required arcs is too long to service in a single day and therefore must be split among multiple days. In this case, we need to decide which day to assign service to each arc, before routing can take place.

Downhill Instances In street snow plowing, it is faster to deadhead (traverse without servicing) a street rather than plowing it. In this case, there are different costs for deadheading and plowing a street. Moreover, it takes longer to plow uphill, resulting in four costs: plowing uphill, plowing downhill, deadheading uphill, and

deadheading downhill.

Precedence When considering downhill instances, the snow may be so deep that it is impossible for a snowplow to deadhead a street before the street is plowed.

In Chapter 2, we discuss the street-sweeper problem (SSP), which adds the multi-period attribute to the RPP. In the SSP, we seek to sweep the sides of city streets in a way that minimizes the distance traveled by the street sweepers. Typically, street sweepers are blocked by parked cars that prevent the curb from being swept. The problem was posed by transportation managers for Washington, DC where the parking constraints are multi-period decision variables. For example, suppose the managers want to sweep streets over two days with available parking on at least one side of each street on each day. Before considering how to sweep streets, managers must first decide which street sides to make available for parking on each day in a way that obeys the parking constraint on both days. We present a genetic algorithm that generates high-quality solutions and discuss managerial implications.

In Chapter 3, we discuss the Plowing with Precedence Problem (PPP), which is motivated by the fact that, in winter street clearing operations, deadhead travel over streets that have already been plowed is significantly faster than the time it takes to plow the street. This problem differs from most arc routing problems because the cost of traversing a street changes depending on the order of the streets on a route. We develop a method called Cycle Permutation Local Search (CPLS) that generates near-optimal solutions to instances as large as 200 vertices.

In Chapter 4, we discuss the Downhill Plow problem (DPP) and the Min-Max Downhill Plow Problem (MM k-DPP). The DPP is a variant of the WPP that is motivated by the fact that deadhead travel over streets is significantly faster than the time it takes

to plow the street. In the MM k-DPP, we incorporate the use of multiple plows and the observation that, on steep streets, it is much more difficult, or impossible, to plow uphill. Therefore, we want to design routes that try to avoid plowing uphill on steep streets and take advantage of the faster traversal time for deadheading. We generalize this problem to include multiple plows and seek to minimize the maximum route length. We present an adaptation of CPLS, CPLS-M, that generates solutions that are very close to a lower bound.

In Chapter 5, we discuss the Extended Min-Max Downhill Plow Problem (MM k-DPPE), which extends the MM k-DPP to include rural graphs, turn penalties, and precedence. We demonstrate how CPLS-M can be modified to incorporate these additional constraints, and present results for turn penalties.

Figure 5.1 shows the hierarchy of the problems that are considered in this dissertation and illustrate how they are derived from existing work.

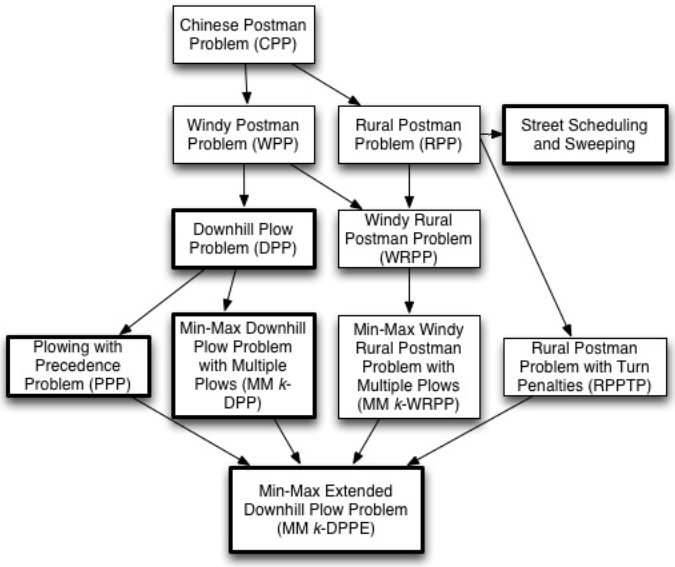


Figure 1.1: Hierarchy of arc routing problems. Arrows indicate increased generality or an additional constraint. Bolded problems are covered in this dissertation.

In optimization, it is often the case that it is computationally prohibitive to generate an optimal solution. This is the case for all problems considered in this dissertation. We develop heuristics that generate a good solution to each of the posed problems in a reasonable amount of time. Heuristics often rely on changing solutions in order to produce improved results and we do this in two ways: By breeding a population of solutions in a genetic algorithm (Chapter 2) or by investigating a neighborhood of nearby solutions for better ones in a local search procedure (Chapters 3-5).

There are three main contributions of this dissertation. First, we use Euclidean cycle decomposition in metaheuristics such as the genetic algorithm in Chapter 2 and CPLS/CPLS-M in Chapters 3, 4, and 5. To our knowledge, the use of moving or permuting sub-cycles as a way to change and improve a Eulerian cycle is novel and very robust at improving solutions. Second, we use the concept of a solution framework in Chapters 3, 4, and 5. This solution framework imposes significant restrictions on the set of solutions investigated by local search. Imposing restrictions on investigated solutions is uncommon for metaheuristics in the literature. Typically, a metaheuristic expands the set of investigated solutions in order to handle increasingly complex problems (often investigating poor solutions in the process) to ensure population diversity, in the case of genetic algorithms, or to avoid being trapped in a local minima, in the case of local search procedures. Our solution framework allows our local search algorithm to investigate promising solutions exclusively without sacrificing solution variety, producing near-optimal solutions. Finally, we provide several important managerial implications from our work. These observations provide insight into the problems at hand regardless of the solution methodology.

Chapter 2

The Street Sweeping and Scheduling Problem

2.1 Introduction

In the street-sweeper problem, we seek to sweep the sides of city streets in a way that minimizes the distance traveled by the street sweepers. Typically, street sweepers are blocked by parked cars that prevent the curb from being swept. We consider a problem posed to us by Washington, DC where the parking constraints are multi-period decision variables. For example, suppose the city wishes to sweep its streets over two days with available parking on at least one side of each street on each day. Before the city considers how to sweep its streets, it must first decide which street sides to make available for parking on each day in a way that obeys the parking constraint on both days.

Our paper is organized as follows. In Section 2, we provide a literature review. In Section 3, we show how our work extends the existing literature and provide the problem statements. In Section 4, we formulate our problem as an integer program. In Section 5, we introduce a genetic algorithm for solving our problem. In Section 6, we present the results produced by the genetic algorithm and compare these results to those produced by solving an integer program formulation. In Section 7, we offer concluding remarks and directions for future research.

2.2 Literature Review

Street sweeping has been modeled as a Directed Rural Postman Problem (DRPP). In the DRPP, we are given a directed graph $G = (V, E)$, a subset $E_R \subseteq E$ of required edges,

and non-negative costs associated with each edge of G. We need to determine a closed path with minimum total cost traversing the edges E_R at least once. Christofides et al. [20] developed a heuristic and mathematical programming formulation for the DRPP. Using their heuristic, the authors solved 23 instances within 1.3% of optimality. In general, the DRPP does not consider any additional constraints, such as parking, or objectives other than route length.

Bodin and Kursh [13, 14] described a computer-assisted system for scheduling and routing of multiple street sweepers. They modeled an urban setting with one-way streets and parking constraints. Parking constraints were modeled as time windows for a street side during a single day. Their algorithm assigned streets to sweepers and produced routes for the sweepers, while obeying parking regulations, balancing workload, and minimizing deadhead. The authors applied their algorithms to pilot studies in New York City and Washington, DC. In New York City, which had a total of 88 miles of streets, the authors reduced the deadhead travel from 65 to 38 miles.

Eglese and Murdock [25] described a street sweeping application in Lancashire County, England. In this application, there were no parking considerations and streets could be regarded as bidirectional (in rural areas, street sweepers are allowed to traverse a street against traffic). However, sweepers were constrained by the capacity of their bins. Sweepers must visit tip-off sites to empty their bins. The authors constructed routes over an entire year that swept each street at a desired annual frequency on a district with 624 intersections, 807 roads, and 4 tipping sites. The authors noted that the routes were "feasible and efficient" and "the amount of manual intervention required to produce good routes is minimal."

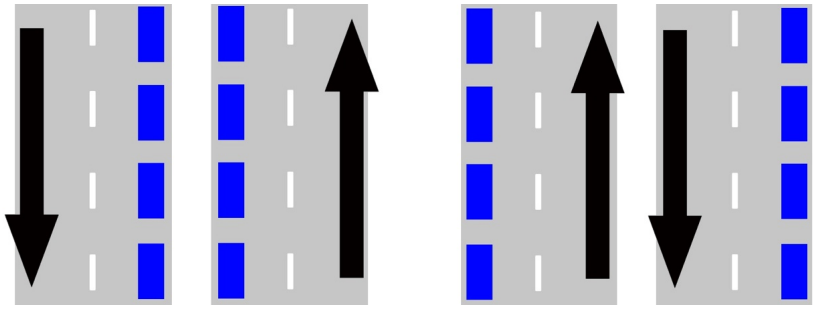
2.3 Problem Statement

Consider a city that needs to sweep a subset of its streets over two days with a single street sweeper (we designate the days as an even day and an odd day). The street sweeper cannot sweep a side of a street if there are cars parked on that side. Because the city requires parking to be available in certain areas, there are parking constraints associated with each street that dictate parking availability and, hence, sweeping ability.

The city has signs to enforce its parking restrictions. It is important to note that the collection of these signs need not be unique to satisfy the parking constraints. For example, a common parking constraint (and a central motivator for our problem) is the requirement that parking be available on at least one side of a street at all times. Assuming both sides must be swept, on each day one side of this street must be available for parking and the other side must be clear for sweeping. There are two choices for the enforcing signs to obey this: close only the left side of the street on even days and the right side on odd days, or close the right side of the street on even days and the left side on odd days. Given these two choices, there are two possible sweeping schedules: sweep the left side of the street on even days and the right on odd days with parking available on the other side of the street (Figure 2.1(a)), or sweep the right on even days and the left on odd days (Figure 2.1(b)).

It is possible that a street has no parking available. In this case, a sweeper is free to sweep either side on either day, or both sides on the same day. While there would only be a single street sign that must be obeyed for the street segment, namely a No Parking sign for both street sides, there still is a decision to be made as to when to sweep both sides of the street.

Thus, we have the following hierarchy of decisions where the decisions made in each



(a) Left Even Day, Right Odd Day (b) Left Even Day, Right Odd Day

Figure 2.1: Two possible ways to sweep both sides of a street over two days while maintaining parking availability on both days

level must obey the decisions made above it.

1 - Each street segment is assigned a parking constraint. This information is provided by the city.

2 - Each street has an enforcing street sign that obeys the parking constraints specified in 1.

3 - A street sweeper has a schedule that obeys street signs. This schedule specifies a sweeping day (even or odd) for each required street segment. A feasible schedule is one that obeys the parking signs specified in 2.

4 - The street sweeper is assigned a route that obeys the schedule given in 3.

In our street-sweeping problem, we consider parking constraints that require multi-period decision variables over the entire period in addition to the decision variables already associated with the routing portion of the problem. Choosing a street sign is a multi-period decision variable that affects any subsequent routing over both days. In order to transition between each level of the hierarchy, we need to make a decision. We now discuss each transition.

2.3.1 Initial Problem - Transition from Level 3 to 4

Once a sweeper knows which streets will be swept on each day (as provided by the schedule), the problem reduces to a DRPP on each day.

2.3.2 Variant 0 - Transition from Level 2 to 3

With the parking signs in place, we have the following problem which we call Variant 0. The city needs to route the street sweeper to minimize total distance traveled while obeying the signs (and hence the parking constraints). It is important to note this

problem is not simply a DRPP. For example, on streets that must be swept but never have available parking, there is a choice of which day to sweep each side of the street. A schedule incorporates these decisions and, after the schedule is assigned to the sweeper, the transition from level 2 to level 3 is made and the city can obtain an optimal sweeping route by solving the DRPP for each day. In Variant 0, we are solving for the optimal schedule given the street signs that are already in place.

We extend Variant 0 to the following two variants.

2.3.3 Variant 1 - Transition from Level 1 to 2

Suppose the city decides to redo all parking signs. How should the city schedule the closing and non-closing of each side of each street and route the street sweeper in order to minimize the total distance traveled by the sweeper while satisfying the parking constraints of each street?

Clearly, relaxing the schedule can only decrease the distance traveled by the sweeper. Consider the example given in Figure 2.2. All streets require available parking at all times.

Suppose the existing schedule is to sweep $(1, 2)$, $(4, 2)$, $(4, 3)$, and $(3, 1)$ on even days (Figure 2.3(a)) and $(2, 1)$, $(2, 4)$, $(3, 4)$, and $(1, 3)$ on odd days (Figure 2.3(b)). Then, for even days, the optimal route that begins and ends at node 1 is $1 \to 2 \to 4 \to 2 \to 4 \to 3 \to 1$ (Figure 2.4(a)). Similarly, the optimal route for odd days is $1 \to 3 \to 4 \to 2 \to 4 \to 2 \to 1$ (Figure 2.4(b)). If each edge has a cost of one, then the total deadhead (non-productive travel) cost is four.

Suppose the city exchanges $(4, 2)$ and $(2, 4)$. We have a new schedule with $(1, 2)$, $(2, 4)$, $(4, 3)$, and $(3, 1)$ being swept on even days (Figure 2.5(a)) and $(2, 1)$, $(4, 2)$, $(3, 4)$, and $(1, 3)$ on odd days (Figure 2.5(b)) results. The optimal solution is $1 \to 2 \to 4 \to 3 \to 1$

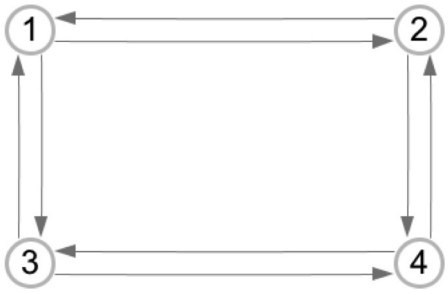
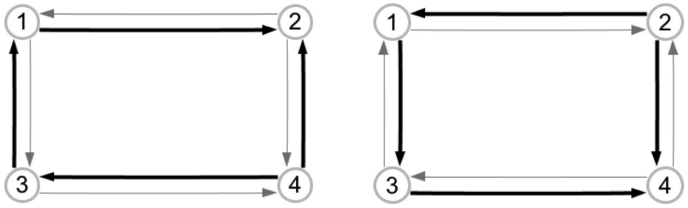
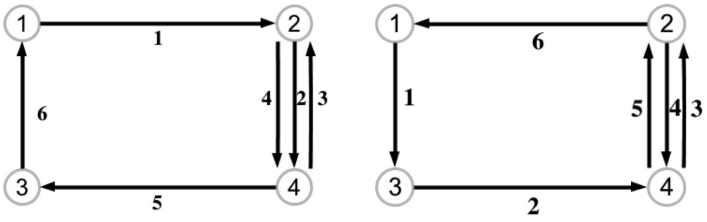


Figure 2.2: All streets must be swept and require available parking



(a) Even Day (b) Odd Day

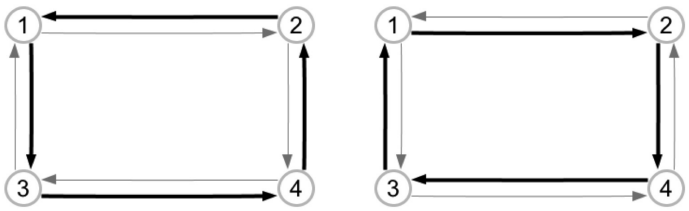
Figure 2.3: Bolded edges are street-sides that must be swept



(a) Even Day (b) Odd Day

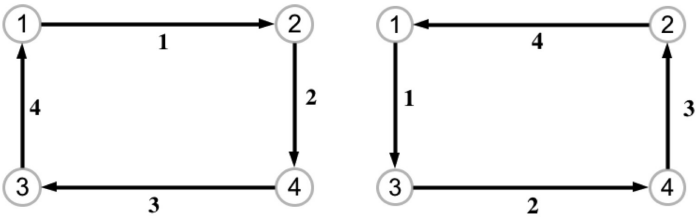
Figure 2.4: Optimal route given a schedule for the street sweeper

for even days (Figure 2.6(a)) and $1 \to 3 \to 4 \to 2 \to 1$ for odd days (Figure 2.6(b)) days. The total deadhead cost is zero.



(a) Even Day (b) Odd Day

Figure 2.5: Revised schedule: Bolded edges are street sides that must be swept



(a) Even Day (b) Odd Day

Figure 2.6: Optimal route for the revised schedule

2.3.4 Variant 2

Suppose the city has a set of street signs already in place. How can the city change as few signs as possible and produce a schedule that has a maximum decrease in the distance traveled by the street sweeper?

This variant is an extension of Variant 1. There are costs associated with changing a street sign. In addition to the physical cost of a placing a new sign, the change might confuse some residents who are familiar with the existing signs. It might be possible to achieve a significant fraction of the benefit of redoing all the signs by changing a smaller

number of signs. For example, the city might obtain a 10% reduction in route length by redoing all of the signs, but it might be possible to obtain a 9% reduction by changing only 15% of the signs. We note that Variant 2 will solve Variant 0 by simply not allowing any street signs to be changed.

2.4 Formulation of Variant 1

We represent a city as a directed graph $G = (V, E)$ where $E = \{(i, j, k) | i, j \in V, k = 0, 1\}$. Each street is represented by two edges representing the two sides of the street that must be swept. The representations are either $(i, j, 0)$ and $(i, j, 1)$ if the street is a one-way street from intersection i to intersection j, or $(i, j, 0)$ and $(j, i, 0)$ if the street is a two-way street. We assume that the directed graph G is strongly connected, that is, the sweeper can reach any intersection in the city from any other intersection. In addition, the street sweeper is responsible for sweeping a subset of G that is determined ahead of time. Thus, for each street (the associated pair of edges), we have four possible states.

1. Both sides never need to be swept. Travel along either side will result in deadheading.

2. One side does not need to be swept and the side other does. Travel along the former will result in deadheading, and the latter may be swept on either day.

3. Both sides need to be swept, but cannot be swept on the same day. This is the situation where parking is required to be available on one side of the street at all times.

4. Both sides need to be swept. This allows for both streets to be swept on the same day. This could occur if a street needs to be swept but there is no parking to consider.

These states are determined ahead of time and are requirements for a feasible schedule. If an edge (i, j, k) (and its associated pair representing the other side of the street) is subject to the constraint given by state l (given in 1, 2, 3, 4 above), we say that $(i, j, k) \in S_l$. We need to determine a feasible schedule that provides the Eulerian tour with the smallest deadhead. We note that the selection of the street signs is completely determined by a schedule, so we are only concerned with obtaining the optimal schedule.

We define the following variables and constants.

$\alpha \in \{0, 1\}$, where 0 denotes an even day and 1 denotes an odd day.

M is the number of vertices.

c_{ij} is the cost of traversing the edge from node i to node j.

$x^{\alpha}_{i,j,k}$ is the number of times the solution traverses edge (i, j, k) on day α.

$$u^{\alpha}_{i,j,k} = \begin{cases} 1, & \text{if } (i, j, k) \text{ is traversed on day } \alpha \\ 0, & \text{otherwise.} \end{cases}$$

$$r^{\alpha}_{i,j,k} = \begin{cases} 1, & \text{if } (i, j, k) \text{ is swept on day } \alpha \\ 0, & \text{otherwise.} \end{cases}$$

$$y^{\alpha}_{i} = \begin{cases} 1, & \text{if node } i \text{ visited on day } \alpha \\ 0, & \text{otherwise.} \end{cases}$$

Our formulation of Variant 1 follows.

$$\text{Minimize} \sum_{i,j,k,\alpha} c_{ij} x^{\alpha}_{i,j,k} \tag{2.1}$$

subject to

$$\sum_{j,k} x^{\alpha}_{0,j,k} \geq 1 \qquad \forall\ \alpha \tag{2.2}$$

$$\sum_{i,k} x^{\alpha}_{i,j,k} = \sum_{i,k} x^{\alpha}_{j,i,k} \qquad \forall\ j, \alpha \tag{2.3}$$

$$x^{\alpha}_{i,j,k} \geq r^{\alpha}_{i,j,k} \qquad \forall\ i, j, k, \alpha \tag{2.4}$$

$$x^{\alpha}_{i,j,k} \geq u^{\alpha}_{i,j,k} \qquad \forall\ i, j, k, \alpha \tag{2.5}$$

$$\sum_{j,k} x^{\alpha}_{i,j,k} \leq M y^{\alpha}_{i} \qquad \forall\ i, j, k, \alpha \tag{2.6}$$

$$r^{\alpha}_{i,j,0} + r^{\alpha}_{j,i,0} = 1 \qquad \forall\ i, j, \alpha \text{ where } (i,j,0), (j,i,0) \in S_3 \tag{2.7}$$

$$r^{\alpha}_{i,j,0} + r^{\alpha}_{i,j,1} = 1 \qquad \forall\ i, j, \alpha \text{ where } (i,j,0), (i,j,1) \in S_3 \tag{2.8}$$

$$\sum_{\alpha} r^{\alpha}_{i,j,k} = 1 \qquad \forall\ i, j, k, \alpha \text{ where } (i,j,k) \in S_2 \text{ or } S_4 \tag{2.9}$$

$$\sum_{i} y^{\alpha}_{i} = 1 + \sum_{i,j,k} u^{\alpha}_{i,j,k} \qquad \forall\ \alpha \tag{2.10}$$

$$v^{\alpha}_{j} \geq (v^{\alpha}_{i} + 1) - (M-1)(1 - u^{\alpha}_{i,j,k}) \qquad \forall\ i, j, i \neq j, \alpha \tag{2.11}$$

$$1 \leq v^{\alpha}_{j} \leq M - 1 \qquad \forall\ 1 \leq j \leq M, \alpha \tag{2.12}$$

$$\sum_{i,k} u^{\alpha}_{i,j,k} \leq 1 \qquad \forall\ j, \alpha \tag{2.13}$$

$$r^{\alpha}_{i,j,k},\ u^{\alpha}_{i,j,k} \in \{0,1\}\ ,\ x^{\alpha}_{i,j,k} \geq 0 \qquad \forall\ i, j, k, \alpha \tag{2.14}$$

In this formulation, we are solving the Rural Postman Problem over two days subject to the pre-determined constraints while allowing flexibility in the schedule. In (2.1), we simply add the costs of travel on all days. Constraint (2.2) requires the street sweeper to leave from the depot. Constraint (2.3) enforces flow balance on every node. Constraint (2.4) enforces coverage of a required edge. If $u^{\alpha}_{i,j,k}$ or $y^{\alpha}_{i} = 1$, then constraints (2.5) and (2.6) require that the sweeper traverse edge (i, j, k). The objective function ensures that

$x^{\alpha}_{i,j,k} = 0$ if $u^{\alpha}_{i,j,k} = 1$ and $y^{\alpha}_{i} = 1$. Constraints (2.7) and (2.8) require that exactly one side of the street must be swept each day when necessary. Constraint (2.9) requires that if a street is to be swept, it must be swept on exactly one day. Constraint (2.10) forces a Eurlerian cycle. Finally, constraints (2.11), (2.12), and (2.13) are the subtour elimination constraints [45].

Consider a city composed of one-way streets with a non-connected set of streets that must be swept. Because each street is composed of two edges that are in the same direction, the optimal solution is to solve the Directed Rural Postman Problem on day 0 and simply follow the same path on day 1. Since the Directed Rural Postman Problem is NP-hard, our generalization in Variant 1 is NP-hard. In the next section, we develop a heuristic to solve this problem.

2.5 Genetic Algorithm

We develop a heuristic procedure that uses a genetic algorithm (GA) to generate good solutions to Variant 1 in a short amount of time. The GA acts on a population of feasible schedules and uses a heuristic for the DRPP to return a tour whose length serves as the fitness function. Next, we describe each component of the genetic algorithm.

2.5.1 Initialization Procedure

It is possible that the set of required edges (those that must be swept) may be relatively small when compared to the entire graph. For example, a city sweeper may only be responsible for a small fraction of the city. However, any algorithm would consider each street in the city as a street that is possibly traversed. Our GA uses a random mutation to investigate a solution that requires the traversal of a particular random edge. However,

this random edge may be very far away from the required edges and is likely to be a poor choice. To remove these poor choices, we determine all required nodes, $R = \{r_i\}$, that are connected to at least one required edge. We calculate a shortest path between each required node. If there are multiple shortest paths, then we randomly select one. Any non-required edge that is not on a shortest path is removed from further consideration.

This process does not remove the optimal solution. Suppose a solution contains an edge that is not on any shortest path. Call the immediately preceding required node r_1 and the immediately following required node r_2. The shortest path from r_1 to r_2 does not contain the selected edge and the solution is not optimal.

In addition, we perform the following operations.

Step 1 - Find all shortest paths from one required node to another.

Step 2 - Remove those shortest paths that contain a third required node.

Step 3 - Collapse the remaining shortest paths into a single edge with length equal to the length of the original path.

The purpose of this operation is to further reduce the state space. Step 2 removes redundant shortest paths. If we have a shortest path from r_1 to r_3 containing r_2, then this shortest path is made redundant by the paths from r_1 to r_2 and r_2 to r_3. Step 3 reduces the remaining paths from potentially many edges to a single edge. If the schedule selects one of the edges in a shortest path to be required on a particular day, then the street sweeper must travel over that entire shortest path on the same day. Collapsing each shortest path to a single edge reduces the state space even further.

We illustrate these concepts with the following example. Consider the city graph in Figure 2.7. Bolded edges are required and all edges have a length of one. Nodes 1, 2, 3,

and 4 are required nodes. Because edges $(2,6)$, $(6,2)$, $(3,6)$, and $(6,3)$ are not on the any shortest path between required nodes, they are removed from consideration. The pairs of edges from $(4,5)$ and $(5,1)$ lie on the shortest paths from node 4 to node 1 and are collapsed into a single edge $(4,1)$ with length two. The resulting graph is given in Figure 2.8.

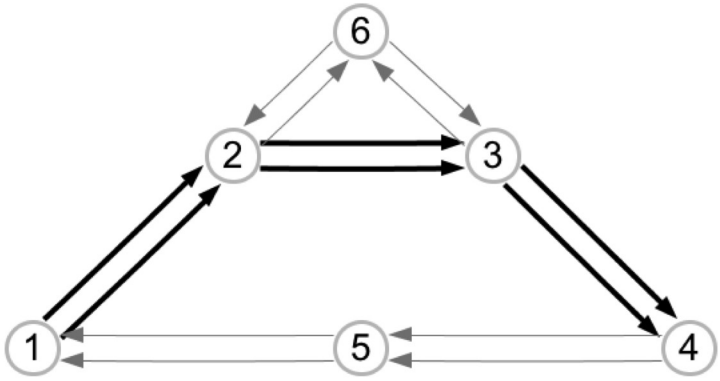


Figure 2.7: City graph: Bolded edges are required

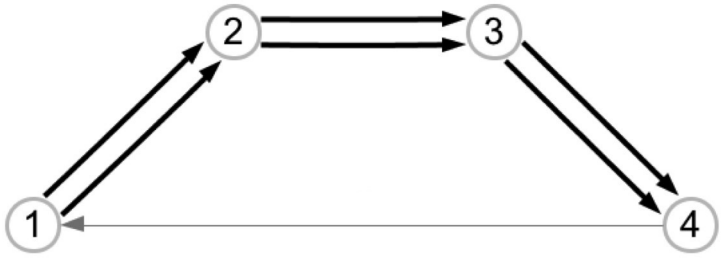


Figure 2.8: City graph after applying the initialization procedure: Bolded edges are required

In Figure 2.9, we show of a portion of the street network in Washington, DC. The bolded (required) streets are those that must be swept by the street sweeper over two days. The required streets are strongly connected within the set of bold and dim streets (the dim streets are those that the sweeper is allowed, but not required, to travel on). The

state space for the entire graph is very large. Applying our initialization procedure gives the graph in Figure 2.10. This graph is considerably smaller, by approximately a factor of two. In the remainder of this paper, the graph G refers to the graph produced by the initialization procedure.

2.5.2 Schedules

A chromosome (schedule) is represented as a $|E|/2$-dimensional vector. Each component represents the state of one street which is a pair of edges and explicitly states, by the element $((u^0_{i,j,0}, u^1_{i,j,0}), (u^0_{j,i,0}, u^1_{j,i,0}))$ or $((u^0_{i,j,0}, u^1_{i,j,0}), (u^0_{i,j,1}, u^1_{i,j,1}))$, when each edge must be traveled and when it does not have to be traveled. Recall that $u^\alpha_{i,j,k}$ indicates whether or not edge (i, j, k) requires travel on day α. A schedule does not explicitly prohibit traversing a an non-required edge.

Depending on what parking restrictions the component represents, the following restrictions are imposed.

S_1 : No restrictions are imposed. Either side may, or may not, be traversed.

S_2 : $u^0_{i,j,0} + u^1_{i,j,0} = 1$ on the required side. This forces the required side to be traversed (and hence swept).

S_3 : The left side is swept on even days and the right side on odd days, or the left side is swept on odd days and the right side on even days.

S_4 : $u^0_{i,j,0} + u^1_{i,j,0} = 1$ and $u^0_{j,i,0} + u^1_{j,i,0} = 1$. Similar to S_2, but both sides are forced to be swept.

The streets are randomly assigned a component of the vector except that two-way streets in S_3 are placed at the top of the schedule and one-way streets in S_3 are placed at

Figure 2.9: Portion of the Washington, DC street network before the initialization procedure is applied

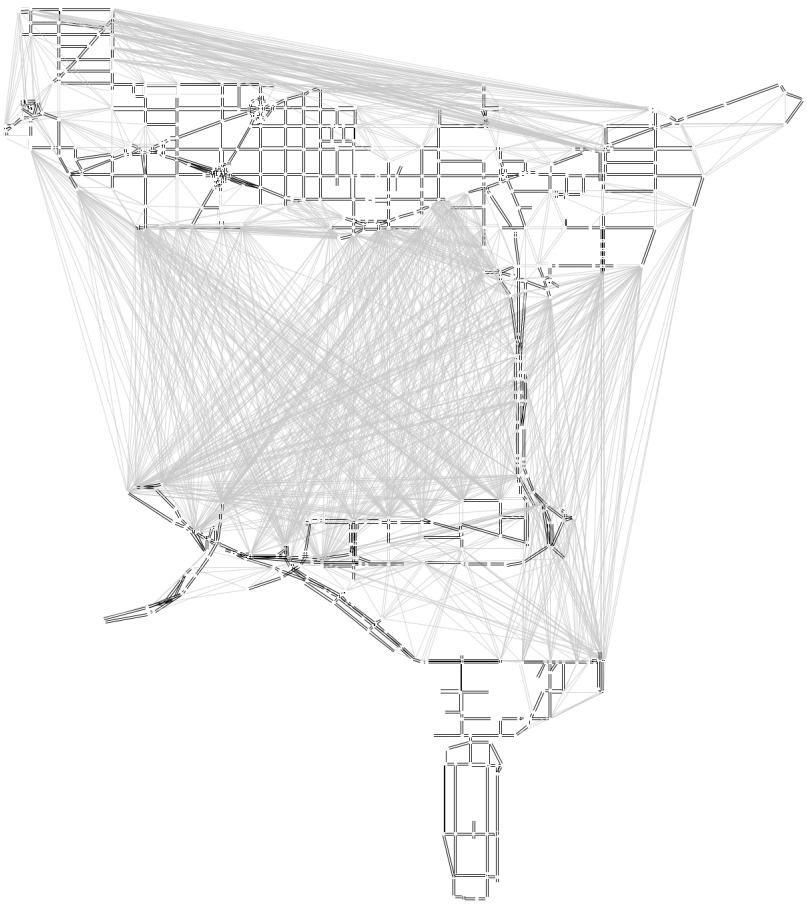


Figure 2.10: Portion of Washington, DC street network after the initialization procedure is applied

the bottom of the schedule. Each side must be swept on different days for one-way streets in S_3. We place them at the end of the vector since changing the days that the different sides are swept produces equivalent schedules.

A schedule induces non-unique daily Eulerian tours. A schedule assign a sweeping day to every required edge and only assigns a sweeping day to some unrequired edges (possibly none). The optimal Eulerian tour for a schedule is found by solving a DRPP.

2.5.3 Fitness

Given a schedule s, we generate an Eulerian tour by solving the DRPP whose length serves as the fitness of s. Because the fitness of a schedule is called often in our heuristic, we employ a modification of the following heuristic given by Christofides et al. [20].

1. Construct a shortest spanning arborescence for the connected components of required edges [24].

2. Solve the transportation problem by adding arcs in a least-cost manner so that the number of incoming arcs is equal to the number of outgoing arcs for each node [8].

3. Construct an Eulerian tour on the resulting graph.

In our heuristic, we construct a spanning arborescence in a greedy manner (this differs from step 1 of the Christofides et. al heuristic). We define the fitness $\mu(s)$ to be the length of the Eulerian cycle.

2.5.4 Breeding

A naive way to breed schedules would be to swap components of the schedules randomly. However, the induced route of a schedule is very sensitive to small changes in

the schedule (see the example in Section 1.1). For example, sweeping an edge on an even day rather than an odd day could result in a larger distance traveled for the sweeper. As a result, breeding two good schedules haphazardly could destroy a good solution.

To construct a good breeding algorithm, we note that all Eulerian cycles can be decomposed into sub-cycles. A short Eulerian cycle will have good sub-cycles, where good sub-cycles are those that will combine to form a short Eulerian cycle. Our breeding method attempts to swap sub-cycles between schedules.

A schedule does not have sub-cycles; the Eulerian cycle that it induces does. However, an optimal cycle is difficult to find. We make the reasonable extension that a good schedule allows for "good" sub-cycles. Our breeding algorithm determines a sub-cycle on a random day allowed by one schedule and adjusts the second schedule to allow the same sub-cycle on the same day.

We breed two schedules, s_1 and s_2 (denoted by $\beta(s_1, s_2)$) as follows.

Step 1 - Choose a random required edge in G. Call it (a_0, a_1) where $a_0, a_1 \in V$. According to the schedule s_2, this edge is either scheduled for day 0 or day 1. Without loss of generality, assume it is day 0. We initialize a list with (a_0, a_1) as the first entry.

Step 2 - Let (a_{t-1}, a_t) be the edge added to the list in Step 1. Randomly choose an edge (other than (a_t, a_{t-1})) scheduled for day 0 that begins from vertex a_t. If no such edge exists, choose a random edge (other than (a_t, a_{t-1})) regardless of the schedule. If no such edge exists, choose edge (a_t, a_{t-1}). Add the selected edge to the end of the list and denote it by (a_t, a_{t+1}).

Step 3 - If $a_{t+1} = a_{t'}$ where $t' \in 0, 1, 2, \cdots, t$, continue to Step 4. The sequence of edges $(a_{t'}, a_{t'+1}), \cdots, (a_{t-1}, a_t), (a_t, a_{t+1})$ defines a cycle. Otherwise, return to Step 2.

Step 4 - The schedule of s_1 is adjusted to allow for the cycle obtained in Step 3. It is important to note that if an edge from the list was scheduled for day 2 instead of day 1 in s_2, it is scheduled for day 1 instead of day 0 in s_1.

We motivate our breeding process with the following example where we restrict our attention to the small portion of a larger graph in Figure 2.11. Nodes 5 and 6 have additional incident edges that are not shown. Nodes 1, 2, 3, and 4 have no incident edges other than those shown. For this part of the graph, we assume the parking restrictions are type S_3, where both sides must be swept but only one side may be swept on each day. In Figures 2.12 and 2.13, we show possible schedules, where the bold edges indicate the edges swept on day 0. The non-bold edges are swept on day 1.

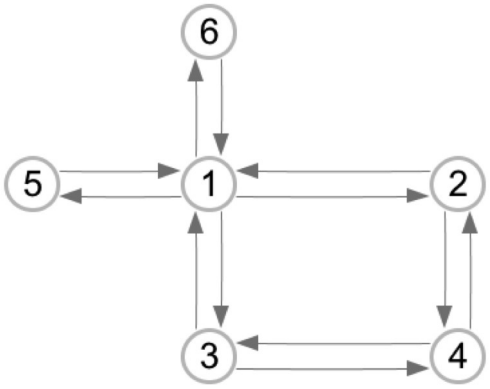


Figure 2.11: Portion of example city graph

We assume that on day 0, the sweeper enters from node 6 and exits at node 5 in schedule 1 and enters from node 5 and exits at node 6 in schedule 2. It is clear that schedule 1 has deadhead, with the tour $6 \to 1 \to 4 \to 3 \to 2 \to 1 \to 2 \to 3 \to 4 \to 1 \to 5$. Schedule 2 has no deadhead with the tour $5 \to 1 \to 4 \to 3 \to 2 \to 1 \to 6$. To construct the child of schedules 1 and 2, $\beta(s_1, s_2)$, we construct a cycle in schedule 2 on day 0. Suppose that the cycle is $1 \to 4 \to 3 \to 2 \to 1$. We adjust schedule 1 to allow this cycle in day

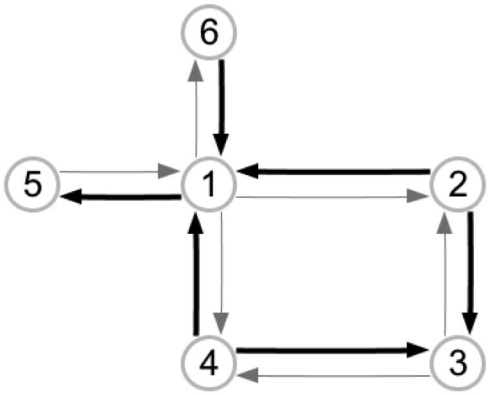


Figure 2.12: Schedule 1

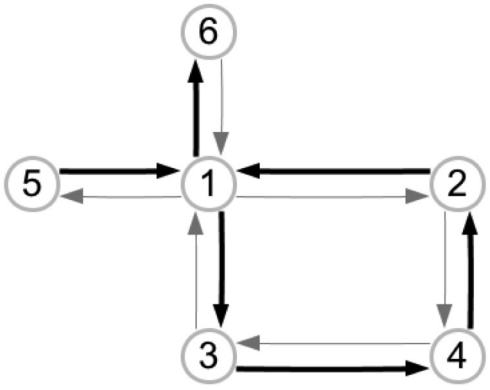


Figure 2.13: Schedule 2

0 (Figure 2.14). The result is set to be $\beta(s_1, s_2)$. Thus, the good cycle of schedule 2 is reflected in schedule 1 and we obtain a new low-cost solution.

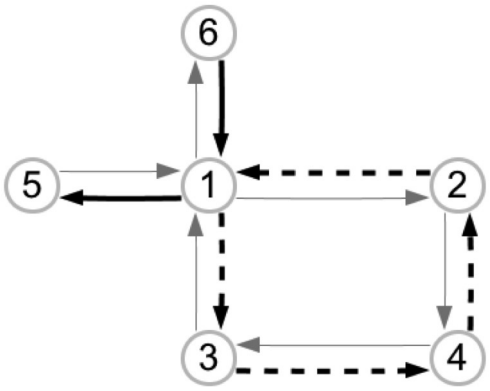


Figure 2.14: Breeding of schedule 1 and schedule 2

2.5.5 Mutation

We have three mutation operators that act on schedules:

Stable Vertex Mutation. One characteristic of schedules with good fitness values is that for a given day, a required node is balanced. That is, the number of required edges for that day entering the required node is equal to the number of required edges for that day exiting the required node. If a required node is not balanced, then it is clear that deadhead will occur. This mutation operator randomly selects a required node with unbalanced degree and tries to balance the node. If possible, the mutation changes the day assignment of an edge incident to the chosen node in a way that decreases the difference between the in- and out-degrees. In Figure 2.15, the difference between the in-degree and out-degree of node 5 is -2 so that deadhead travel occurs. Switching edge $(5, 6)$ with $(6, 5)$ on the same day balances the degree of node 5 and yields the right-hand graph with no deadhead travel.

Random Path Mutation. It is possible that a simple path, not necessarily a cycle, yields a lower-cost tour on an even day rather than on an odd day or vice versa. The breeding process will likely not change a path that is not a cycle because it deals exclusively with cycles. This mutation finds a random path of random length (2 to 5 edges long) on a random day and then adjusts the schedule to allow the same path on the other day.

Consider the example schedule in Figure 2.16, where each street is of type S_3 (required, but each side must be swept on different days). Bolded edges are required on day 0 and faded edges on day 1. In Figure 2.17, edges $(8, 7)$, $(7, 6)$, and $(6, 5)$ are undesirable in day 0 because they flow in the opposite direction from the rest of the schedule. This implies that edges $(5, 6)$, $(6, 7)$, and $(7, 8)$ are undesirable for day 1. The mutation operator tries to identify such a path and move it to the other day, as shown in Figure 2.18. This change might have been obtained by the breeding process. However, in this case, that would not occur unless the population already had the optimal schedule because there is only a single possible cycle in the graph.

Random Change Mutation. This mutation operator randomly changes the edge-day assignment of one or two streets in the schedule.

The mutation of schedule s, denoted $m(s)$, calls stable vertex mutation with 10% probability, random path mutation with probability 20%, random change of two streets with 20% probability, and random change of a single street with 50% probability.

2.5.6 Genetic Algorithm Summary

We describe our genetic algorithm in Table 2.1.

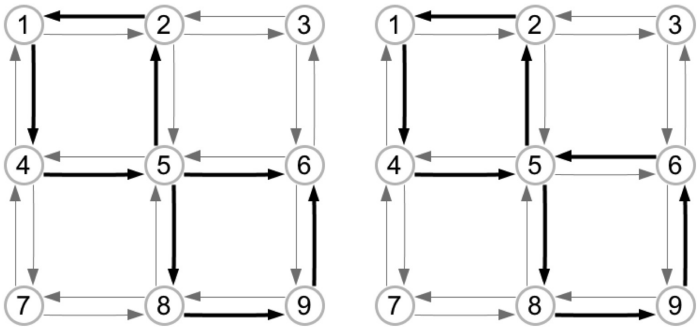


Figure 2.15: Schedule of required edges on one day before and after degree balancing with stable vertex mutation

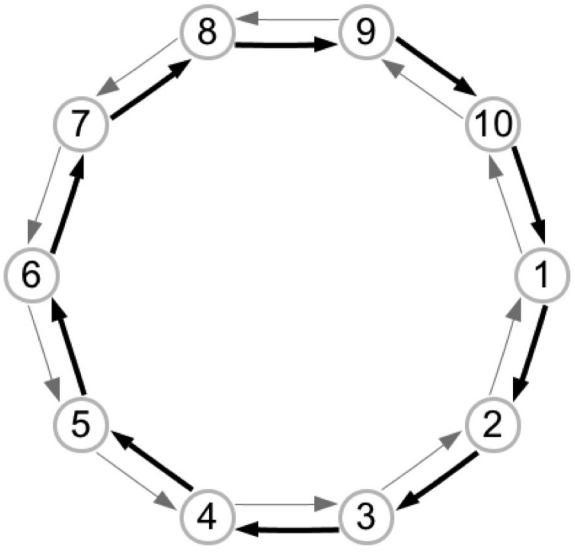


Figure 2.16: Bolded edges are required on day 0 and faded edges on day 1

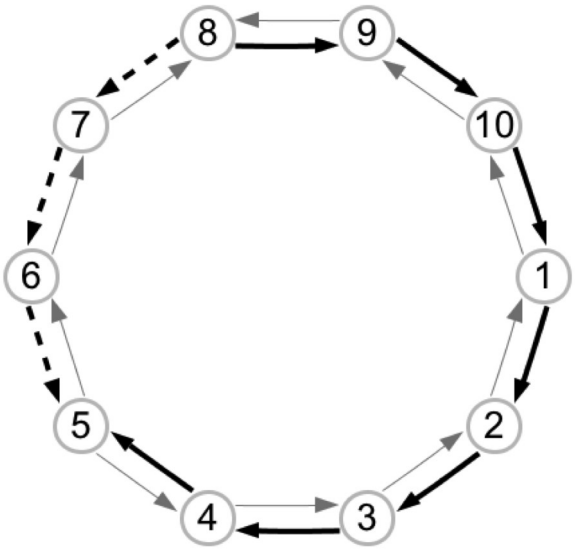


Figure 2.17: Bolded paths are undesirable for day 0

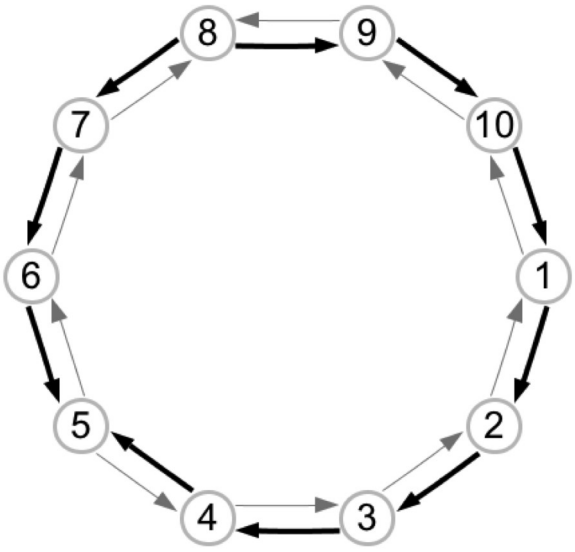


Figure 2.18: Revised schedule after random path mutation: bolded edges are required on day 0 and faded edges on day 1

Set:
k = size of population
h = number of best (ordered by fitness) schedules (set to be about $k/10$)
N = number of iterations without improvement
Let:
$S = \{s_i\}_i$ (the set of feasible schedules)
P_t = population at generation $t \in \mathbb{Z}^+$ (a vector with schedules as components and $|P_t| = K \ \forall \ t$)
$P_t[i]$ = schedule i of population P_t
$\mu(P_t[i])$ = fitness of $P_t[i]$
$\beta(P_t[i], P_t[i])$ = child of $P_t[i]$ and $P_t[j]$
$m(P_t[i])$ = mutation of $P_t[i]$

Create a random population of k individuals in S
Sort the random population with respect to the fitness function (best solution at the top)
Set the reordered population to be P_0
Set indices $n = 0$ and $t = 0$
Repeat
    For each i such that $k \leq i \leq h$
        Set $P_t[i] = \beta(P_t[i], P_t[j])$ where j is a random value in the range 1, ... $(i-1)$.
            This breeds schedule i that is not in the top h schedules with a random schedule
            better than it.
        If $P_t[i] = P_t[j]$ (after the breeding of the previous step), mutate $P_t[i]$ by setting
            $P_t[i] = m(P_t[i])$. Having duplicate schedules in the population is undesirable
            so, if duplication occurs, we mutate the duplicate.
        Calculate the fitness $\mu(P_t[i])$
    End-For
    For each i such that $k \leq i \leq 2h$
        Compute the mutation of $m(P_t[i])$
        If $\mu(m(P_t[i])) < \mu(P_t[i])$ set $P_t[i] = m(P_t[i])$. This mutates a schedule and
            replaces it with the mutation if the mutation is better.
    End-For
    Sort population with respect to the fitness function
    Set the reordered population to P_{t+1}
    If $P_{t+1}[0] = P_t[0]$ et $n = n + 1$, Else set $n = 0$
    Set $t = t + 1$
Until $n = N$
Return $P_t^{final}[1]$

Table 2.1: Genetic Algorithm

2.6 Results

2.6.1 Variant 1

We compare our genetic algorithm (GA) against a CPLEX implementation of the integer program (with a 7,200 second time limit) and a local search heuristic (LS). The local search acts on a randomly generated schedule by enumerating the fitness of all nearby schedules, accepting the best one, and repeating. Nearby schedules are defined to be those where a single component in the vector representation of the schedule is changed. We summarize LS in Table 4.2. All tests were performed on a 1.83 GHz Intel Core 2 Duo processor. Our test instances are both randomly generated (to emulate a city) and obtained from actual data of Washington DC. Instances are characterized by three attributes, the number of nodes (intersections), the number of edges (street segments), and the percentage of one-way streets. Our results are given in Table 4. A summary table of averages is given in Table 2.3. We summarize the results, focusing on comparisons for small (25 and 50 nodes) and large (100 and 225 nodes) instances. There are 40 small instances and 20 large instances.

2.6.1.1 Small Instances (25 and 50 nodes)

Our CPLEX implementation hits its time limit of 7,200 seconds on 25 of the 40 small instances. On two of these 25 instances, it fails to obtain a feasible solution. Fifteen of the 25 instances were solved optimally. Our GA obtains good solutions with respect to CPLEX on the 38 instances where CPLEX is able to produce a solution. On average, the solutions generated by our GA are 0.05% worse than the CPLEX solutions for 25-node instances and 0.27% worse on 49-node instances. GA outperforms LS by an average of 1.2% on 25-node instances and 1.1% on 49-node instances.

Set:
N = number of iterations without improvement
Let:
S_t = schedule at generation $t \in \mathbb{Z}^+$
$\mu(S_t)$ = fitness of S_t
$\epsilon(S_t)$ = Neighborhood of S_t (those who differ by a single component from S_t)

Set indices $n = 0$ and $t = 0$
Construct a random schedule S_0
Repeat
    For each schedule $s \in \epsilon(S_t)$ calculate $\mu(s)$
    Choose $s\prime \in \epsilon(S_t)$ such that $\mu(s\prime) \geq \mu(s)\ \forall s \in S_t$
    Set $S_{t+1} = s\prime$
    If $S_t = s\prime$ set $n = n + 1$, Else set $n = 0$
    Set $t = t + 1$
Until $n = N$
Return S_t

Table 2.2: Local Search Algorithm

Number of Nodes	25		49		100	225
Average Deviation from:	CPLEX	GA	CPLEX	GA	GA	GA
GA	0.05%	-	0.27%	-	-	-
LS	1.21%	1.20%	1.29%	1.10%	0.41%	0.34%

Table 2.3: Route Length Comparisons Between CPLEX, GA, and LS

CPLEX's running times range from 0.743 seconds to 6,580 seconds with an average of 548 seconds when it doesn't hit its time limit. In comparison, the average running time of GA is 5.4 seconds. LS is very fast with an average running time of 1.2 seconds.

2.6.1.2 Large Instances (100 and 225 nodes)

CPLEX fails to obtain a feasible solution to 18 of 20 instances. Thus, the CPLEX implementation is not a viable way of solving large instances.

GA outperforms LS by an average of 0.41% with respect to route length on 100-node instances and 0.34% on 225-node instances. For 100-node instances, the average running times of GA and LS are 44 and 34.4 seconds, respectively. For the largest instances with 225 nodes, the average running times for GA and LS are 708.7 and 3006.2 seconds, respectively. The running time performance of LS degrades substantially compared to GA on the largest problems.

2.6.2 Variant 2

In Variant 2, we seek to change a minimal number of existing street signs and achieve a maximum reduction in route length. A schedule is more restrictive than the street signs because it may stipulate traversal or non-traversal on streets that do not require sweeping. We modify our genetic algorithm by inserting a penalty in the fitness function

$$C\max\left(0, \left(\sum_{r_{i,j,k}\in I_0} r_{i,j,k}\right) - n\right) \tag{2.15}$$

where C is a large constant and I_0 is the set of $r_{i,j,k} \in S_3$ that are initialized to 0. Since streets of type S_3 correspond to street signs, the second term in equation (2.15) simply penalizes the fitness in proportion to the number of street signs changed over the desired

value, n.

For comparison purposes, we run GA and do not allow any changes to street signs to obtain a baseline solution, which is equivalent to solving Variant 0. It is interesting to note that, in this case, GA improves a random schedule an average of 38.4% by modifying the scheduling (and hence routing) on non-required streets.

We increase the number of allowed street sign changes (n) over several iterations and compare the resulting fitness values. Results are in Tables 5, 6, 7, and 8. We give an example that has 100 intersections and 179 street segments. In Figure 2.19, we see the percentage improvement compared to the baseline solution. When we allow changes to more than 7% of the street signs, there are diminishing returns in the fitness function. In Figure 2.20, we compare the same results against the best solution produced by GA in solving Variant 1. We see that applying Variant 2 to an existing solution gets us very close to the best solution generated by solving Variant 1. In this example, the difference in total distance traveled is essentially zero when changing approximately 12% of the street signs. This convergence of the fitness function for Variant 2 with Variant 1 occurs in nearly all test instances, with the average deviation being 0.004% at 12% change and 0.927% at 7% change. This is useful to city managers, because solution quality is essentially the same, compared to Variant 1 when we begin with an initial solution and change about 12% of the street signs.

2.7 Conclusions and Future Work

In this paper, we presented a new variant of the street sweeping problem that incorporated scheduling of parking constraints. We developed a genetic algorithm that produced high-quality solutions that were better than those produced by CPLEX and a

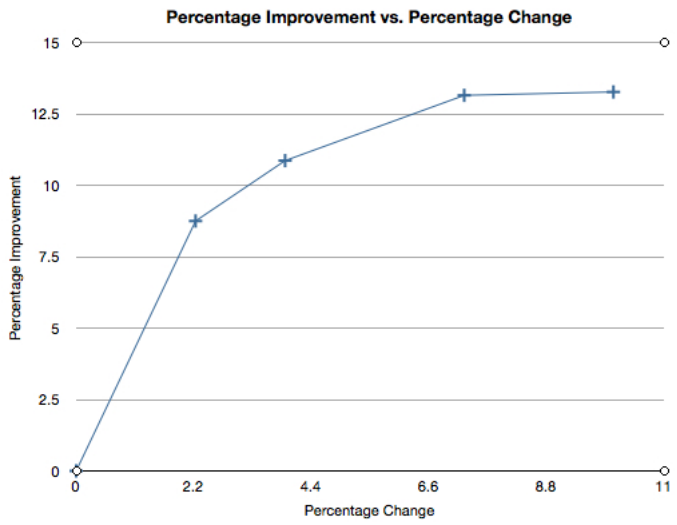


Figure 2.19: Percentage route length reduction vs. percentage of changed street signs

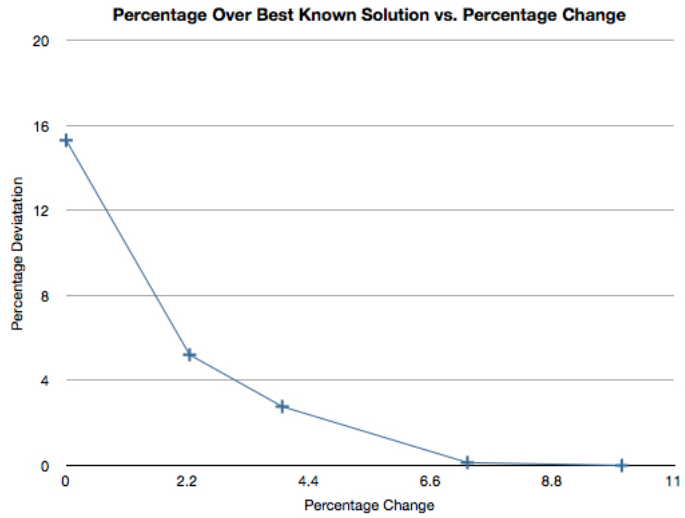


Figure 2.20: Percent Deviation of Variant 2 compared to best solution vs. percentage of changed street signs

naive local search. The running times of GA were much faster than CPLEX and LS for large instances. Finally, we presented a new variant that tried to minimally modify an existing schedule. We showed that the same solution quality can be achieved by allowing changes to approximately 12% of street signs of an initial solution as producing a solution from scratch. Our algorithm can be easily extended to incorporate additional objectives such as route balancing into the objective function.

In future research, we might consider multiple sweepers, which would require additional decision variables that assign street sides to a sweeper. We might also consider extending the planning period to more than two days. For example, there might be no parking on street sides on some days of the week, such as Monday, Wednesday, and Friday, or Tuesday and Thursday. We might incorporate practical concepts into the objective function, such as encouraging right-hand turns at an intersection because of their shorter duration compared to left-hand turns. Finally, real-world considerations such as sweeper capacity and intermediate dumping facilities could be taken into account.

				CPLEX		GA		Local Search	
Instance Name	**Nodes**	**Edges**	**One Way %**	**Solution**	**Time(sec)**	**Solution**	**Time(sec)**	**Solution**	**Time(sec)**
INSTX_5_5_88_3_1_50_1_1	25	88	50.00%	9023	1.19	9024	2.85	9286	0.22
INSTX_5_5_82_3_1_50_1_2	25	82	50.00%	8802	0.93	8802	1.72	8803	0.15
INSTX_5_5_94_3_1_50_1_3	25	94	50.00%	7931	TL	7931	1.64	7931	0.23
INSTX_5_5_82_3_1_50_1_4	25	82	50.00%	10655	1.09	10656	1.80	10799	0.14
INSTX_5_5_88_3_1_50_1_5	25	88	50.00%	8066	TL	8066	1.69	8066	0.17
INSTX_5_5_78_3_1_50_1_6	25	78	50.00%	8996	TL	8997	1.92	9059	0.13
INSTX_5_5_72_3_1_50_1_7	25	72	50.00%	8281	2.44	8282	2.10	8550	0.13
INSTX_5_5_84_3_1_50_1_8	25	84	50.00%	11511	0.74	11511	2.19	11511	0.20
INSTX_5_5_86_3_1_50_1_9	25	86	50.00%	10915	3.78	10916	2.24	10916	0.17
INSTX_5_5_82_3_1_50_1_10	25	82	50.00%	7623	TL	7624	2.80	7695	0.15
INSTX_5_5_82_3_1_35_1_1	25	82	35.00%	10017	TL	10103	2.35	10242	0.17
INSTX_5_5_92_3_1_35_1_2	25	92	35.00%	11397	4.08	11397	2.36	11397	0.23
INSTX_5_5_88_3_1_35_1_3	25	88	35.00%	8695	TL	8696	2.16	8696	0.17
INSTX_5_5_82_3_1_35_1_4	25	82	35.00%	11280	TL	11281	2.33	11281	0.17
INSTX_5_5_84_3_1_35_1_5	25	84	35.00%	9466	0.97	9467	2.17	9467	0.17
INSTX_5_5_78_3_1_35_1_6	25	78	35.00%	8180	TL	8181	2.44	8181	0.29
INSTX_5_5_84_3_1_35_1_7	25	84	35.00%	10800	1.43	10800	2.41	10800	0.21
INSTX_5_5_76_3_1_35_1_8	25	76	35.00%	5061	67.77	5062	2.57	5620	0.12
INSTX_5_5_84_3_1_35_1_9	25	84	35.00%	11625	TL	11626	2.42	11626	0.14
INSTX_5_5_76_3_1_35_1_10	25	76	35.00%	7723	TL	7724	1.99	7928	0.12
INSTX_7_7_168_3_1_50_1_1	49	168	50.00%	16940	20.14	16998	10.58	16941	1.82
INSTX_7_7_166_3_1_50_1_2	49	166	50.00%	14135	TL	14517	5.87	14874	2.61
INSTX_7_7_174_3_1_50_1_3	49	174	50.00%	17757	TL	17616	6.25	17616	1.87
INSTX_7_7_182_3_1_50_1_4	49	182	50.00%	23731	TL	23732	7.79	23942	2.24
INSTX_7_7_172_3_1_50_1_5	49	172	50.00%	22803	1509.49	22804	8.84	22804	3.41
INSTX_7_7_170_3_1_50_1_6	49	170	50.00%	21802	TL	21803	10.57	21879	3.73
INSTX_7_7_178_3_1_50_1_7	49	178	50.00%	18133	TL	17926	8.40	18183	2.21
INSTX_7_7_180_3_1_50_1_8	49	180	50.00%	14333	TL	14273	10.31	14275	2.16
INSTX_7_7_178_3_1_50_1_9	49	178	50.00%	22130	TL	22271	6.71	22209	2.16
INSTX_7_7_182_3_1_50_1_10	49	182	50.00%	21181	15.76	21195	15.02	21379	3.84
INSTX_7_7_152_3_1_35_1_1	49	152	35.00%	16334	TL	16535	8.03	16676	1.45
INSTX_7_7_174_3_1_35_1_2	49	174	35.00%	16393	TL	16715	10.01	17349	1.93
INSTX_7_7_172_3_1_35_1_3	49	172	35.00%	19591	TL	19655	6.99	19862	2.73
INSTX_7_7_146_3_1_35_1_4	49	146	35.00%	-1	TL	20121	13.05	20586	1.72
INSTX_7_7_166_3_1_35_1_5	49	166	35.00%	16842	TL	16906	7.18	17036	2.45
INSTX_7_7_182_3_1_35_1_6	49	182	35.00%	18676	TL	18677	8.47	19484	2.32
INSTX_7_7_166_3_1_35_1_7	49	166	35.00%	17753	TL	17679	4.77	17679	1.45
INSTX_7_7_184_3_1_35_1_8	49	184	35.00%	-1	TL	18312	13.81	18594	2.48
INSTX_7_7_162_3_1_35_1_9	49	162	35.00%	18288	6580.58	18291	4.92	18352	1.23
INSTX_7_7_168_3_1_35_1_10	49	168	35.00%	20894	15.92	20895	7.12	21241	1.87
INSTX_10_10_358_3_1_35_1_1	100	358	35.00%	-1	TL	43828	32.70	44027	30.80
INSTX_10_10_336_3_1_35_1_2	100	336	35.00%	-1	TL	31714	48.89	32151	39.12
INSTX_10_10_366_3_1_35_1_3	100	366	35.00%	-1	TL	39701	24.08	39707	20.71
INSTX_10_10_388_3_1_35_1_4	100	388	35.00%	-1	TL	36758	29.80	37055	27.09
INSTX_10_10_368_3_1_35_1_5	100	368	35.00%	-1	TL	43876	39.35	44076	27.03
INSTX_10_10_360_3_1_35_1_6	100	360	35.00%	-1	TL	40621	57.73	40486	49.58
INSTX_10_10_366_3_1_35_1_7	100	366	35.00%	36726	6535.87	37271	35.63	37671	33.10
INSTX_10_10_350_3_1_35_1_8	100	350	35.00%	41110	TL	41111	36.34	41111	25.24
INSTX_10_10_386_3_1_35_1_9	100	386	35.00%	-1	TL	39263	88.81	39623	69.19
INSTX_10_10_330_3_1_35_1_10	100	330	35.00%	-1	TL	40744	46.31	40484	21.76
INSTX_15_15_818_3_1_35_1_1	225	818	35.00%	-1	TL	89841	609.44	90204	1006.26
INSTX_15_15_820_3_1_35_1_2	225	820	35.00%	-1	TL	92160	604.99	91206	1601.33
INSTX_15_15_842_3_1_35_1_3	225	842	35.00%	-1	TL	91567	447.01	90947	8042.71
INSTX_15_15_840_3_1_35_1_4	225	840	35.00%	-1	TL	85490	359.32	87274	1200.03
INSTX_15_15_882_3_1_35_1_5	225	882	35.00%	-1	TL	84404	502.56	84901	12693.50
INSTX_15_15_836_3_1_35_1_6	225	836	35.00%	-1	TL	92552	616.14	92613	795.48
INSTX_15_15_836_3_1_35_1_7	225	836	35.00%	-1	TL	76648	2577.70	76975	917.87
INSTX_15_15_818_3_1_35_1_8	225	818	35.00%	-1	TL	88360	294.19	88597	647.01
INSTX_15_15_880_3_1_35_1_9	225	880	35.00%	-1	TL	100667	487.21	100501	1133.06
INSTX_15_15_838_3_1_35_1_10	225	838	35.00%	-1	TL	88921	588.95	90190	2024.50

Table 2.4: Variant 1 Results (TL = Time Limit)

Instance	Num Edges	GA Sol	Initial Sol	# Changed	% Changed	Improved Sol	% Imp	% Dev
INSTX_5_5_88_3_1_50_1_1	88	9024	10934	2	2.27	10266	6.11	13.76
INSTX_5_5_82_3_1_50_1_2	82	8802	9364	2	2.44	8803	5.99	0.01
INSTX_5_5_94_3_1_50_1_3	94	7931	9455	2	2.13	8691	8.08	9.58
INSTX_5_5_82_3_1_50_1_4	82	10656	12432	2	2.44	11062	11.02	3.81
INSTX_5_5_88_3_1_50_1_5	88	8066	8898	2	2.27	8273	7.02	2.57
INSTX_5_5_78_3_1_50_1_6	78	8997	9805	1	1.28	9599	2.10	6.69
INSTX_5_5_72_3_1_50_1_7	72	8282	10212	1	1.39	9603	5.96	15.95
INSTX_5_5_84_3_1_50_1_8	84	11511	13107	2	2.38	11925	9.02	3.60
INSTX_5_5_86_3_1_50_1_9	86	10916	12721	1	1.16	12369	2.77	13.31
INSTX_5_5_82_3_1_50_1_10	82	7624	7694	2	2.44	7624	0.91	0.00
INSTX_7_7_168_3_1_50_1_1	168	16998	19256	4	2.38	17825	7.43	4.87
INSTX_7_7_166_3_1_50_1_2	166	14517	17043	4	2.41	15841	7.05	9.12
INSTX_7_7_174_3_1_50_1_3	174	17616	20094	4	2.30	18648	7.20	5.86
INSTX_7_7_182_3_1_50_1_4	182	23732	26891	4	2.20	24346	9.46	2.59
INSTX_7_7_172_3_1_50_1_5	172	22804	24994	4	2.33	23544	5.80	3.25
INSTX_7_7_170_3_1_50_1_6	170	21803	23766	4	2.35	22575	5.01	3.54
INSTX_7_7_178_3_1_50_1_7	178	17926	20955	4	2.25	19355	7.64	7.97
INSTX_7_7_180_3_1_50_1_8	180	14273	16784	4	2.22	15780	5.98	10.56
INSTX_7_7_178_3_1_50_1_9	178	22271	24832	4	2.25	23370	5.89	4.93
INSTX_7_7_182_3_1_50_1_10	182	21195	23308	4	2.20	21925	5.93	3.44
INSTX_5_5_82_3_1_35_1_1	82	10103	10936	1	1.22	10574	3.31	4.66
INSTX_5_5_92_3_1_35_1_2	92	11397	12023	2	2.17	11479	4.52	0.72
INSTX_5_5_88_3_1_35_1_3	88	8696	10550	2	2.27	9803	7.08	12.73
INSTX_5_5_82_3_1_35_1_4	82	11281	13510	1	1.22	12686	6.10	12.45
INSTX_5_5_84_3_1_35_1_5	84	9467	10418	2	2.38	9877	5.19	4.33
INSTX_5_5_78_3_1_35_1_6	78	8181	9772	1	1.28	9356	4.26	14.36
INSTX_5_5_84_3_1_35_1_7	84	10800	11563	1	1.19	11273	2.51	4.38
INSTX_5_5_76_3_1_35_1_8	76	5062	5702	2	2.63	5476	3.96	8.18
INSTX_5_5_84_3_1_35_1_9	84	11626	13092	1	1.19	12744	2.66	9.62
INSTX_5_5_76_3_1_35_1_10	76	7724	8957	1	1.32	8612	3.85	11.50
INSTX_7_7_152_3_1_35_1_1	152	16535	20516	3	1.97	19101	6.90	15.52
INSTX_7_7_174_3_1_35_1_2	174	16715	19758	4	2.30	18220	7.78	9.00
INSTX_7_7_172_3_1_35_1_3	172	19655	22874	3	1.74	21487	6.06	9.32
INSTX_7_7_146_3_1_35_1_4	146	20121	22283	3	2.05	21479	3.61	6.75
INSTX_7_7_166_3_1_35_1_5	166	16906	19339	4	2.41	17962	7.12	6.25
INSTX_7_7_182_3_1_35_1_6	182	18677	21093	4	2.20	19848	5.90	6.27
INSTX_7_7_166_3_1_35_1_7	166	17679	21323	3	1.81	19859	6.87	12.33
INSTX_7_7_184_3_1_35_1_8	184	18312	20446	4	2.17	19074	6.71	4.16
INSTX_7_7_162_3_1_35_1_9	162	18291	22255	3	1.85	20738	6.82	13.38
INSTX_7_7_168_3_1_35_1_10	168	20895	22830	3	1.79	21648	5.18	3.60
INSTX_10_10_358_3_1_35_1_1	358	43828	50530	8	2.23	46110	8.75	5.21
INSTX_10_10_336_3_1_35_1_2	336	31714	35119	8	2.38	33311	5.15	5.04
INSTX_10_10_366_3_1_35_1_3	366	39701	45371	8	2.19	42368	6.62	6.72
INSTX_10_10_388_3_1_35_1_4	388	36758	42328	9	2.32	38935	8.02	5.92
INSTX_10_10_368_3_1_35_1_5	368	43876	49861	8	2.17	46825	6.09	6.72
INSTX_10_10_360_3_1_35_1_6	360	40621	44725	8	2.22	42371	5.26	4.31
INSTX_10_10_366_3_1_35_1_7	366	37271	42343	8	2.19	39107	7.64	4.93
INSTX_10_10_350_3_1_35_1_8	350	41111	47128	7	2.00	44213	6.19	7.55
INSTX_10_10_386_3_1_35_1_9	386	39263	45760	9	2.33	40703	11.05	3.67
INSTX_10_10_330_3_1_35_1_10	330	40744	45915	7	2.12	43228	5.85	6.10
INSTX_15_15_818_3_1_35_1_1	818	89841	100036	19	2.32	94339	5.69	5.01
INSTX_15_15_820_3_1_35_1_2	820	92160	103516	19	2.32	96858	6.43	5.10
INSTX_15_15_842_3_1_35_1_3	842	91567	104492	19	2.26	97139	7.04	6.09
INSTX_15_15_840_3_1_35_1_4	840	85490	100022	19	2.26	93994	6.03	9.95
Average					2.07		6.08	6.99

Table 2.5: Variant 2 Results: approximate maximum allowance of 3% schedule changes (Num Edges = Number of edges, GA Sol = Best solution obtained by GA, Initial Sol = Best solution obtained with an approzimate maximum allowance of 0% schedule changes, # Changed = number of schedule changed, % Changed = percentage of schedule changed, Improved Sol = Improved solution fitness after schedule changes, % Imp = Percentage Improvement over initial solution, % Dev = Percentage deviation from best solution obtained by GA)

Instance	Num Edges	GA Sol	Initial Sol	# Changed	% Changed	Improved Sol	% Imp	% Dev
INSTX_5_5_88_3_1_50_1_1	88	9024	10934	3	3.41	9983	8.70	10.63
INSTX_5_5_82_3_1_50_1_2	82	8802	9364	3	3.66	8802	6.00	0.00
INSTX_5_5_94_3_1_50_1_3	94	7931	9455	3	3.19	8349	11.70	5.27
INSTX_5_5_82_3_1_50_1_4	82	10656	12432	3	3.66	10797	13.15	1.32
INSTX_5_5_88_3_1_50_1_5	88	8066	8898	3	3.41	8133	8.60	0.83
INSTX_5_5_78_3_1_50_1_6	78	8997	9805	3	3.85	9196	6.21	2.21
INSTX_5_5_72_3_1_50_1_7	72	8282	10212	2	2.78	9318	8.75	12.51
INSTX_5_5_84_3_1_50_1_8	84	11511	13107	3	3.57	11511	12.18	0.00
INSTX_5_5_86_3_1_50_1_9	86	10916	12721	3	3.49	11879	6.62	8.82
INSTX_5_5_82_3_1_50_1_10	82	7624	7694	3	3.66	7624	0.91	0.00
INSTX_7_7_168_3_1_50_1_1	168	16998	19256	7	4.17	17478	9.23	2.82
INSTX_7_7_166_3_1_50_1_2	166	14517	17043	7	4.22	14922	12.44	2.79
INSTX_7_7_174_3_1_50_1_3	174	17616	20094	7	4.02	18303	8.91	3.90
INSTX_7_7_182_3_1_50_1_4	182	23732	26891	7	3.85	23999	10.75	1.13
INSTX_7_7_172_3_1_50_1_5	172	22804	24994	6	3.49	23155	7.36	1.54
INSTX_7_7_170_3_1_50_1_6	170	21803	23766	6	3.53	22216	6.52	1.89
INSTX_7_7_178_3_1_50_1_7	178	17926	20955	7	3.93	18948	9.58	5.70
INSTX_7_7_180_3_1_50_1_8	180	14273	16784	7	3.89	15163	9.66	6.24
INSTX_7_7_178_3_1_50_1_9	178	22271	24832	7	3.93	22740	8.42	2.11
INSTX_7_7_182_3_1_50_1_10	182	21195	23308	7	3.85	21643	7.14	2.11
INSTX_5_5_82_3_1_35_1_1	82	10103	10936	3	3.66	10367	5.20	2.61
INSTX_5_5_92_3_1_35_1_2	92	11397	12023	3	3.26	11475	4.56	0.68
INSTX_5_5_88_3_1_35_1_3	88	8696	10550	3	3.41	9461	10.32	8.80
INSTX_5_5_82_3_1_35_1_4	82	11281	13510	3	3.66	11831	12.43	4.88
INSTX_5_5_84_3_1_35_1_5	84	9467	10418	3	3.57	9671	7.17	2.15
INSTX_5_5_78_3_1_35_1_6	78	8181	9772	2	2.56	9076	7.12	10.94
INSTX_5_5_84_3_1_35_1_7	84	10800	11563	3	3.57	10805	6.56	0.05
INSTX_5_5_76_3_1_35_1_8	76	5062	5702	3	3.95	5471	4.05	8.08
INSTX_5_5_84_3_1_35_1_9	84	11626	13092	2	2.38	12400	5.29	6.66
INSTX_5_5_76_3_1_35_1_10	76	7724	8957	3	3.95	8325	7.06	7.78
INSTX_7_7_152_3_1_35_1_1	152	16535	20516	6	3.95	18083	11.86	9.36
INSTX_7_7_174_3_1_35_1_2	174	16715	19758	7	4.02	17675	10.54	5.74
INSTX_7_7_172_3_1_35_1_3	172	19655	22874	6	3.49	20586	10.00	4.74
INSTX_7_7_146_3_1_35_1_4	146	20121	22283	5	3.42	21055	5.51	4.64
INSTX_7_7_166_3_1_35_1_5	166	16906	19339	6	3.61	17561	9.19	3.87
INSTX_7_7_182_3_1_35_1_6	182	18677	21093	7	3.85	19294	8.53	3.30
INSTX_7_7_166_3_1_35_1_7	166	17679	21323	6	3.61	18770	11.97	6.17
INSTX_7_7_184_3_1_35_1_8	184	18312	20446	7	3.80	18668	8.70	1.94
INSTX_7_7_162_3_1_35_1_9	162	18291	22255	6	3.70	19912	10.53	8.86
INSTX_7_7_168_3_1_35_1_10	168	20895	22830	6	3.57	21100	7.58	0.98
INSTX_10_10_358_3_1_35_1_1	358	43828	50530	14	3.91	45042	10.86	2.77
INSTX_10_10_336_3_1_35_1_2	336	31714	35119	14	4.17	32550	7.32	2.64
INSTX_10_10_366_3_1_35_1_3	366	39701	45371	13	3.55	41390	8.77	4.25
INSTX_10_10_388_3_1_35_1_4	388	36758	42328	15	3.87	37993	10.24	3.36
INSTX_10_10_368_3_1_35_1_5	368	43876	49861	13	3.53	45693	8.36	4.14
INSTX_10_10_360_3_1_35_1_6	360	40621	44725	14	3.89	41084	8.14	1.14
INSTX_10_10_366_3_1_35_1_7	366	37271	42343	14	3.83	38008	10.24	1.98
INSTX_10_10_350_3_1_35_1_8	350	41111	47128	13	3.71	42599	9.61	3.62
INSTX_10_10_386_3_1_35_1_9	386	39263	45760	15	3.89	40189	12.17	2.36
INSTX_10_10_330_3_1_35_1_10	330	40744	45915	12	3.64	42036	8.45	3.17
INSTX_15_15_818_3_1_35_1_1	818	89841	100036	32	3.91	92315	7.72	2.75
INSTX_15_15_820_3_1_35_1_2	820	92160	103516	32	3.90	94624	8.59	2.67
INSTX_15_15_842_3_1_35_1_3	842	91567	104492	32	3.80	94825	9.25	3.56
INSTX_15_15_840_3_1_35_1_4	840	85490	100022	32	3.81	91122	8.90	6.59
INSTX_15_15_882_3_1_35_1_5	882	84404	97302	35	3.97	88574	8.97	4.94
INSTX_15_15_836_3_1_35_1_6	836	92552	102847	32	3.83	95366	7.27	3.04
INSTX_15_15_836_3_1_35_1_7	836	76648	91036	33	3.95	81104	10.91	5.81
INSTX_15_15_818_3_1_35_1_8	818	88360	102718	31	3.79	92817	9.64	5.04
INSTX_15_15_880_3_1_35_1_9	880	100667	113182	34	3.86	102797	9.18	2.12
INSTX_15_15_838_3_1_35_1_10	838	88921	100270	33	3.94	92210	8.04	3.70
Average					3.67		8.62	4.06

Table 2.6: Variant 2 Results: approximate maximum allowance of 5% schedule changes

Instance	Num Edges	GA Sol	Initial Sol	# Changed	% Changed	Improved Sol	% Imp	% Dev
INSTX_5_5_88_3_1_50_1_1	88	9024	10934	7	7.95	9286	15.07	2.90
INSTX_5_5_82_3_1_50_1_2	82	8802	9364	5	6.10	8802	6.00	0.00
INSTX_5_5_94_3_1_50_1_3	94	7931	9455	7	7.45	7931	16.12	0.00
INSTX_5_5_82_3_1_50_1_4	82	10656	12432	6	7.32	10656	14.29	0.00
INSTX_5_5_88_3_1_50_1_5	88	8066	8898	6	6.82	8066	9.35	0.00
INSTX_5_5_78_3_1_50_1_6	78	8997	9805	6	7.69	8997	8.24	0.00
INSTX_5_5_72_3_1_50_1_7	72	8282	10212	5	6.94	8550	16.27	3.24
INSTX_5_5_84_3_1_50_1_8	84	11511	13107	3	3.57	11511	12.18	0.00
INSTX_5_5_86_3_1_50_1_9	86	10916	12721	6	6.98	11264	11.45	3.19
INSTX_5_5_82_3_1_50_1_10	82	7624	7694	4	4.88	7624	0.91	0.00
INSTX_7_7_168_3_1_50_1_1	168	16998	19256	11	6.55	17217	10.59	1.29
INSTX_7_7_166_3_1_50_1_2	166	14517	17043	12	7.23	14577	14.47	0.41
INSTX_7_7_174_3_1_50_1_3	174	17616	20094	13	7.47	17758	11.63	0.81
INSTX_7_7_182_3_1_50_1_4	182	23732	26891	12	6.59	23732	11.75	0.00
INSTX_7_7_172_3_1_50_1_5	172	22804	24994	12	6.98	22804	8.76	0.00
INSTX_7_7_170_3_1_50_1_6	170	21803	23766	13	7.65	21879	7.94	0.35
INSTX_7_7_178_3_1_50_1_7	178	17926	20955	14	7.87	18475	11.83	3.06
INSTX_7_7_180_3_1_50_1_8	180	14273	16784	15	8.33	14415	14.11	0.99
INSTX_7_7_178_3_1_50_1_9	178	22271	24832	14	7.87	22271	10.31	0.00
INSTX_7_7_182_3_1_50_1_10	182	21195	23308	13	7.14	21379	8.28	0.87
INSTX_5_5_82_3_1_35_1_1	82	10103	10936	6	7.32	10167	7.03	0.63
INSTX_5_5_92_3_1_35_1_2	92	11397	12023	7	7.61	11397	5.21	0.00
INSTX_5_5_88_3_1_35_1_3	88	8696	10550	6	6.82	8905	15.59	2.40
INSTX_5_5_82_3_1_35_1_4	82	11281	13510	5	6.10	11281	16.50	0.00
INSTX_5_5_84_3_1_35_1_5	84	9467	10418	6	7.14	9467	9.13	0.00
INSTX_5_5_78_3_1_35_1_6	78	8181	9772	5	6.41	8524	12.77	4.19
INSTX_5_5_84_3_1_35_1_7	84	10800	11563	6	7.14	10800	6.60	0.00
INSTX_5_5_76_3_1_35_1_8	76	5062	5702	6	7.89	5404	5.23	6.76
INSTX_5_5_84_3_1_35_1_9	84	11626	13092	5	5.95	11629	11.17	0.03
INSTX_5_5_76_3_1_35_1_10	76	7724	8957	6	7.89	7787	13.06	0.82
INSTX_7_7_152_3_1_35_1_1	152	16535	20516	12	7.89	16792	18.15	1.55
INSTX_7_7_174_3_1_35_1_2	174	16715	19758	13	7.47	16858	14.68	0.86
INSTX_7_7_172_3_1_35_1_3	172	19655	22874	13	7.56	20044	12.37	1.98
INSTX_7_7_146_3_1_35_1_4	146	20121	22283	11	7.53	20448	8.23	1.63
INSTX_7_7_166_3_1_35_1_5	166	16906	19339	13	7.83	16982	12.19	0.45
INSTX_7_7_182_3_1_35_1_6	182	18677	21093	14	7.69	18742	11.15	0.35
INSTX_7_7_166_3_1_35_1_7	166	17679	21323	13	7.83	17885	16.12	1.17
INSTX_7_7_184_3_1_35_1_8	184	18312	20446	15	8.15	18397	10.02	0.46
INSTX_7_7_162_3_1_35_1_9	162	18291	22255	12	7.41	18544	16.67	1.38
INSTX_7_7_168_3_1_35_1_10	168	20895	22830	13	7.74	21039	7.84	0.69
INSTX_10_10_358_3_1_35_1_1	358	43828	50530	26	7.26	43887	13.15	0.13
INSTX_10_10_336_3_1_35_1_2	336	31714	35119	28	8.33	31186	11.20	-1.66
INSTX_10_10_366_3_1_35_1_3	366	39701	45371	27	7.38	40057	11.71	0.90
INSTX_10_10_388_3_1_35_1_4	388	36758	42328	30	7.73	36827	13.00	0.19
INSTX_10_10_368_3_1_35_1_5	368	43876	49861	27	7.34	44296	11.16	0.96
INSTX_10_10_360_3_1_35_1_6	360	40621	44725	29	8.06	40954	8.43	0.82
INSTX_10_10_366_3_1_35_1_7	366	37271	42343	28	7.65	37667	11.04	1.06
INSTX_10_10_350_3_1_35_1_8	350	41111	47128	26	7.43	41115	12.76	0.01
INSTX_10_10_386_3_1_35_1_9	386	39263	45760	29	7.51	39934	12.73	1.71
INSTX_10_10_330_3_1_35_1_10	330	40744	45915	24	7.27	41435	9.76	1.70
INSTX_15_15_818_3_1_35_1_1	818	89841	100036	63	7.70	89622	10.41	-0.24
INSTX_15_15_820_3_1_35_1_2	820	92160	103516	64	7.80	93151	10.01	1.08
INSTX_15_15_842_3_1_35_1_3	842	91567	104492	65	7.72	91706	12.24	0.15
INSTX_15_15_840_3_1_35_1_4	840	85490	100022	65	7.74	87469	12.55	2.31
INSTX_15_15_882_3_1_35_1_5	882	84404	97302	70	7.94	84977	12.67	0.68
INSTX_15_15_836_3_1_35_1_6	836	92552	102847	64	7.66	93019	9.56	0.50
INSTX_15_15_836_3_1_35_1_7	836	76648	91036	67	8.01	76780	15.66	0.17
INSTX_15_15_818_3_1_35_1_8	818	88360	102718	62	7.58	89381	12.98	1.16
INSTX_15_15_880_3_1_35_1_9	880	100667	113182	68	7.73	101582	10.25	0.91
INSTX_15_15_838_3_1_35_1_10	838	88921	100270	66	7.88	89491	10.75	0.64
Average					7.29		11.29	0.95

Table 2.7: Variant 2 Results: approximate maximum allowance of 10% schedule changes

Instance	Num Edges	GA Sol	Initial Sol	# Changed	% Changed	Improved Sol	% Imp	% Dev
INSTX_5_5_88_3_1_50_1_1	88	9024	10934	16	18.18	9024	17.47	0.00
INSTX_5_5_82_3_1_50_1_2	82	8802	9364	3	3.66	8802	6.00	0.00
INSTX_5_5_94_3_1_50_1_3	94	7931	9455	11	11.70	7931	16.12	0.00
INSTX_5_5_82_3_1_50_1_4	82	10656	12432	12	14.63	10656	14.29	0.00
INSTX_5_5_88_3_1_50_1_5	88	8066	8898	7	7.95	8066	9.35	0.00
INSTX_5_5_78_3_1_50_1_6	78	8997	9805	7	8.97	9059	7.61	0.69
INSTX_5_5_72_3_1_50_1_7	72	8282	10212	8	11.11	8282	18.90	0.00
INSTX_5_5_84_3_1_50_1_8	84	11511	13107	5	5.95	11511	12.18	0.00
INSTX_5_5_86_3_1_50_1_9	86	10916	12721	12	13.95	10916	14.19	0.00
INSTX_5_5_82_3_1_50_1_10	82	7624	7694	11	13.41	7624	0.91	0.00
INSTX_7_7_168_3_1_50_1_1	168	16998	19256	23	13.69	16998	11.73	0.00
INSTX_7_7_166_3_1_50_1_2	166	14517	17043	17	10.24	14515	14.83	-0.01
INSTX_7_7_174_3_1_50_1_3	174	17616	20094	19	10.92	17616	12.33	0.00
INSTX_7_7_182_3_1_50_1_4	182	23732	26891	23	12.64	23732	11.75	0.00
INSTX_7_7_172_3_1_50_1_5	172	22804	24994	20	11.63	22804	8.76	0.00
INSTX_7_7_170_3_1_50_1_6	170	21803	23766	16	9.41	21803	8.26	0.00
INSTX_7_7_178_3_1_50_1_7	178	17926	20955	22	12.36	17926	14.45	0.00
INSTX_7_7_180_3_1_50_1_8	180	14273	16784	21	11.67	14334	14.60	0.43
INSTX_7_7_178_3_1_50_1_9	178	22271	24832	14	7.87	22271	10.31	0.00
INSTX_7_7_182_3_1_50_1_10	182	21195	23308	19	10.44	21257	8.80	0.29
INSTX_5_5_82_3_1_35_1_1	82	10103	10936	10	12.20	10103	7.62	0.00
INSTX_5_5_92_3_1_35_1_2	92	11397	12023	8	8.70	11397	5.21	0.00
INSTX_5_5_88_3_1_35_1_3	88	8696	10550	11	12.50	8696	17.57	0.00
INSTX_5_5_82_3_1_35_1_4	82	11281	13510	11	13.41	11281	16.50	0.00
INSTX_5_5_84_3_1_35_1_5	84	9467	10418	8	9.52	9467	9.13	0.00
INSTX_5_5_78_3_1_35_1_6	78	8181	9772	12	15.38	8181	16.28	0.00
INSTX_5_5_84_3_1_35_1_7	84	10800	11563	10	11.90	10800	6.60	0.00
INSTX_5_5_76_3_1_35_1_8	76	5062	5702	9	11.84	5202	8.77	2.77
INSTX_5_5_84_3_1_35_1_9	84	11626	13092	9	10.71	11626	11.20	0.00
INSTX_5_5_76_3_1_35_1_10	76	7724	8957	13	17.11	7724	13.77	0.00
INSTX_7_7_152_3_1_35_1_1	152	16535	20516	27	17.76	16395	20.09	-0.85
INSTX_7_7_174_3_1_35_1_2	174	16715	19758	28	16.09	16394	17.03	-1.92
INSTX_7_7_172_3_1_35_1_3	172	19655	22874	21	12.21	19657	14.06	0.01
INSTX_7_7_146_3_1_35_1_4	146	20121	22283	21	14.38	20121	9.70	0.00
INSTX_7_7_166_3_1_35_1_5	166	16906	19339	20	12.05	16850	12.87	-0.33
INSTX_7_7_182_3_1_35_1_6	182	18677	21093	21	11.54	18682	11.43	0.03
INSTX_7_7_166_3_1_35_1_7	166	17679	21323	21	12.65	17616	17.38	-0.36
INSTX_7_7_184_3_1_35_1_8	184	18312	20446	27	14.67	18315	10.42	0.02
INSTX_7_7_162_3_1_35_1_9	162	18291	22255	27	16.67	18289	17.82	-0.01
INSTX_7_7_168_3_1_35_1_10	168	20895	22830	23	13.69	20895	8.48	0.00
INSTX_10_10_358_3_1_35_1_1	358	43828	50530	36	10.06	43827	13.27	0.00
INSTX_10_10_336_3_1_35_1_2	336	31714	35119	27	8.04	31391	10.62	-1.02
INSTX_10_10_366_3_1_35_1_3	366	39701	45371	49	13.39	39707	12.48	0.02
INSTX_10_10_388_3_1_35_1_4	388	36758	42328	42	10.82	36551	13.65	-0.56
INSTX_10_10_368_3_1_35_1_5	368	43876	49861	42	11.41	44014	11.73	0.31
INSTX_10_10_360_3_1_35_1_6	360	40621	44725	33	9.17	40635	9.14	0.03
INSTX_10_10_366_3_1_35_1_7	366	37271	42343	43	11.75	37262	12.00	-0.02
INSTX_10_10_350_3_1_35_1_8	350	41111	47128	40	11.43	41317	12.33	0.50
INSTX_10_10_386_3_1_35_1_9	386	39263	45760	41	10.62	39384	13.93	0.31
INSTX_10_10_330_3_1_35_1_10	330	40744	45915	42	12.73	40610	11.55	-0.33
INSTX_15_15_818_3_1_35_1_1	818	89841	100036	100	12.22	89791	10.24	-0.06
INSTX_15_15_820_3_1_35_1_2	820	92160	103516	88	10.73	91383	11.72	-0.84
INSTX_15_15_842_3_1_35_1_3	842	91567	104492	93	11.05	91502	12.43	-0.07
INSTX_15_15_840_3_1_35_1_4	840	85490	100022	101	12.02	86273	13.75	0.92
INSTX_15_15_882_3_1_35_1_5	882	84404	97302	109	12.36	84811	12.84	0.48
INSTX_15_15_836_3_1_35_1_6	836	92552	102847	90	10.77	92665	9.90	0.12
INSTX_15_15_836_3_1_35_1_7	836	76648	91036	87	10.41	76536	15.93	-0.15
INSTX_15_15_818_3_1_35_1_8	818	88360	102718	104	12.71	88353	13.98	-0.01
INSTX_15_15_880_3_1_35_1_9	880	100667	113182	105	11.93	100797	10.94	0.13
INSTX_15_15_838_3_1_35_1_10	838	88921	100270	107	12.77	88236	12.00	-0.77
Average					11.87		12.10	0.00

Table 2.8: Variant 2 Results: approximate maximum allowance of 25% schedule changes

Chapter 3

Plowing with Precedence

3.1 Introduction

In winter, a common problem is to determine the optimal route that snow plows should take in order to minimize the distance traveled. Typically, this problem is represented as an arc routing problem, which seeks to find the route of minimum length that traverses the desired arcs of a graph G. For example, the Chinese Postman Problem (CPP) [41] is an arc-routing problem that seeks a route that visits each arc of a graph at least once while minimizing deadhead. The CPP, which has symmetric costs for each arc, can be extended to the Windy Postman Problem (WPP) [46], where the costs of traversal are not symmetric. We propose a variant to the WPP that is motivated by the fact that deadhead travel over streets that have already been plowed is significantly faster than the time it takes to plow the street. We incorporate the observation that, on some steep streets, it is much more difficult, or impossible, to plow uphill.

Therefore, we want to design routes that try to avoid plowing uphill on steep streets and take advantage of the faster traversal time on plowed streets. It is important to note that this assumption requires the possibility of plowing against the direction of traffic. If the direction of traffic is enforced, it is impossible to avoid plowing uphill. We assume that the snow conditions are sufficiently bad to warrant closing the streets. This assumption allows for the possibility of deadhead travel when only one side of the street has been plowed, regardless of the direction. For example, if a steep street has been plowed downhill, it is possible to traverse the street uphill without plowing, as long as the plow eventually

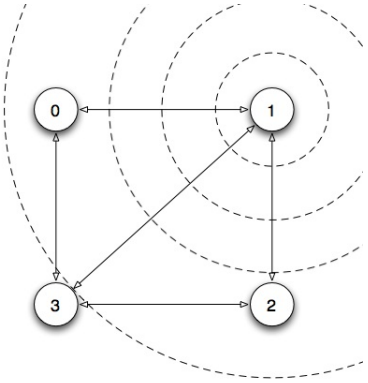
returns to clear the other side. If a street has not been plowed, however, then it is impossible to deadhead that street because the snow is too deep to traverse without plowing.

Deep snow on hills is a common occurrence after a large snow storm. While snow emergency routes are plowed first, side streets and less critical streets remain unplowed. In fact, streets in the suburbs surrounding a metropolitan area are divided geographically and assigned to individual plows before routes are determined (private communication with Cleveland's Division of Streets, 2011).

This notion of partitioning the residential streets first, taking into account both route duration and geographical considerations, and then routing snow plows within each partition is a common practice. For example, see the residential snow plowing routes for Spokane, Washington (www.spokanestreetdepartment.org/documents/snowroutemaps.pdf) and visit the Ann Arbor, Michigan snow plowing status web page chart (www.a2gov.org/snow) for visual confirmation.

We formulate the Plowing with Precedence Problem (PPP) in the following way. Consider an undirected graph $G = \{V, E\}$ where the set of vertices $V = \{v_i\}$ and the set of edges $E \subseteq \{(v_i, v_j) | v_i, v_j \in V, i < j\}$. Each edge (v_i, v_j) has four costs: c_{ij}^+, the cost of plowing from v_i to v_j, c_{ji}^+, the cost of plowing from v_j to v_i, c_{ij}^-, the cost of traversing from v_i to v_j after plowing, and c_{ji}^-, the cost of traversing from v_j to v_i after plowing. Assuming that vertex v_j has a higher altitude than vertex v_i then usually $c_{ij}^+ \gg c_{ji}^+ \gg c_{ij}^- \geq c_{ji}^-$. We seek the route that begins and ends at the depot (v_0), plows each edge twice (for each side of the street), and minimizes total cost. The order of a route is important since the cost of traversing an edge depends on whether or not the edge has been previously traversed. As a result, where the route begins is important and requires specifying a starting depot.

Consider the graph shown in Figure 5.2 with associated plowing and deadhead costs. Routes must begin and end at the depot (vertex 0). In this graph, the dashed lines indicate level sets with regard to elevation. Vertex 1 is at the top of a hill, and it is very difficult to plow to vertex 1 from either vertex 0 or vertex 2. It is easier to plow uphill from vertex 3 to vertex 1 because it is not as steep as the other approaches up the hill. It is easy to plow downhill. The cost of deadhead is the same for all edges and directions.



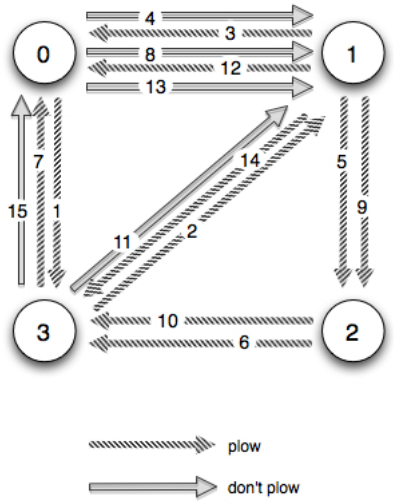
(a) Vertex 1 is on a hill

Edge	c^+_{ij}	c^+_{ji}	c^-_{ij}	c^-_{ji}
(0,1)	10	2	1	1
(2,1)	10	2	1	1
(2,3)	2	2	1	1
(3,0)	2	2	1	1
(3,1)	6	2	1	1

(b) Edge costs (c^+_{ij} plow uphill, c^+_{ji} plow downhill, c^-_{ij} traverse uphill, c^-_{ji} traverse downhill)

Figure 3.1: A graph with five edges (streets) with various grades that require plowing and their associated costs

In Figure 3.2, we show a route that plows the graph given in Figure 5.2. The cost of this route is 29. This route has a large amount of deadhead travel in order to avoid plowing uphill. Moreover, consider the edge $(0, 1)$. Initially, the snowplow avoids plowing uphill because of the large cost, but soon plows it downhill. The snowplow then travels from vertex 0 to vertex 1 (uphill) several times. The snowplow can travel uphill with the deadhead cost because it can travel up the side of the street that has been plowed already. Finally, the snowplow clears the other side of the street near the end of the route.



(a) A route that plows the graph given in Figure 5.2

Step	Edge	Plow	Cost
1	0,3	Yes	2
2	3,1	Yes	6
3	1,0	Yes	2
4	0,1	No	1
5	1,2	Yes	2
6	2,3	Yes	2
7	3,0	Yes	2
8	0,1	No	1
9	1,2	Yes	2
10	2,3	Yes	2
11	3,1	No	1
12	1,0	Yes	2
13	0,1	No	1
14	1,3	Yes	2
15	3,0	No	1
Total			29

(b) Cost of the route

Figure 3.2: Route for a graph with five edges and its associated cost

This emphasizes that, unlike most arc routing problems, precedence is important. In this paper, we develop a local search procedure that generates high-quality solutions to the PPP.

The rest of this paper is organized as follows. In Section 2, we review relevant articles in the arc routing literature. In Section 3, we provide an integer programming formulation for the PPP. In Section 4, we develop our solution methodology for the PPP. In Section 5, we present the results of our computational experiments. In Section 6, we offer concluding remarks and directions for future research.

3.2 Literature Review

Plowing with Precedence is an arc routing problem, specifically a variant of the WPP first described by Minieka [46]. Arc routing has been well studied in the literature. Assad and Golden [6] provide a review of arc routing problems with an emphasis on applications,

including street plowing and gritting, which is the process of laying salt and sand on a road to prevent snow accumulation and improve road traction. Eiselt et al. [26, 27] provide a review of arc routing. In [26], they cover variants of the CPP. In [27], they cover variants of the Rural CPP, which generalizes the CPP by only requiring a subset of the arc set to be traversed. A book on arc routing was edited by Dror [22]. This book describes arc routing problems and solution methodologies in various practical applications.

Winter street maintenance covers a wide variety of problems from routing snow plows to locating snow disposal sites. The literature on winter street maintenance is comprehensively reviewed by Perrier et al. [48, 49, 50, 51]. The four articles cover optimization algorithms for the design of winter road maintenance systems for plowing and gritting, system design problems for snow disposal, optimization algorithms for routing vehicles in gritting operations and vehicle depot location, and optimization algorithms for routing vehicles in plowing and snow disposal operations.

We provide a brief review of the arc routing literature that specifically deals with snow plowing. Marks and Stricker [44] consider the following problem. Given a fleet of m homogeneous plows, construct m routes that minimize the total deadhead, such that each street is plowed at least two or four times depending on the width of the street. We call this problem the m-Plowing Problem (m-PP). The authors account for the multiple pass requirements by duplicating each road segment the required number of times. Two approaches for generating routes are developed. The first approach partitions the graph into m subgraphs by solving a districting problem and then solves the CPP on each subgraph. In the second approach, the original graph is altered to form a Eulerian graph and then is partitioned into m subgraphs, taking care to make each subgraph Eulerian and approximately the same size. Three strategies were suggested to handle street priorities.

The first strategy reduces the length of those streets with higher priority so that solving the CPP will result in these streets being duplicated more often. The second strategy partitions the graph by priority class, ensures connectivity, and then routes the plows. The third strategy generates several Eulerian cycles after clustering and then chooses the best one. The authors apply their first approach to data from the city of Cambridge, Massachusetts.

The Bureau of Management Consulting, Transport Canada [15] provides a solution methodology to the m-PP by constructing a strategy for partitioning the graph in an attempt to reduce deadhead. By requiring that each subgraph have an even number of odd degree nodes and attempting to cluster the odd-degree nodes geographically, the amount of deadhead incurred when solving the CPP for each subgraph is reduced. The authors address street priority by modifying Fleury's algorithm used in the CPP to favor roads with higher priority (see Section 4.3 for a discussion of Fleury's algorithm). The Bureau of Management Consulting tested their heuristic on data from a major Canadian city.

Instead of having m vehicles as in the m-PP, Haslam and Wright [38] place a constraint on the maximum length of a plowing route in addition to street priorities. We call this problem the Capacitated Plowing Problem (C-PP). The authors calculate a lower bound on the number of vehicles and prompt the user to provide a number of seed nodes and associated classes in order to construct routes. A three-stage algorithm is used to construct initial routes. The first stage constructs a feasible path from the seed node to the depot and back. In the second stage, non-covered arcs are inserted iteratively as long as street priorities and maximum route lengths are obeyed. In the third stage, if required arcs have not been covered, then the street priority constraint is relaxed and the

second stage is repeated. The routes are improved by applying two procedures. The first procedure swaps arcs between routes. The second procedure removes a route of a given class and attempts to insert the arcs into other routes of the same class. The entire procedure was applied to a subdistrict of Indiana. The first improvement procedure reduced deadhead by 23% and the second by 9% over the existing method used by the subdistrict.

Wang and Wright [55] develop a tabu search method for a variant of the C-PP, where the objective function minimizes a combination of the total route length and the number of plows. An initial solution is created by a procedure (similar to the one in [38]) that requires no user input. The authors improved routes (in deadhead and number of plows) by an average of 8% over the existing method used in Indiana. We note that the instances in [55] are not the same as in [38] and the objective functions are different.

Kandula and Wright [39] develop a three-stage heuristic for the C-PP described in Wang and Wright [55]. The first stage selects seed nodes that are close to the depot. The second phase constructs routes one at a time from each seed node by adding arcs that travel away from the depot in a greedy manner. The third phase is an improvement procedure that swaps pairs of arcs between two routes or moves an arc from one route to another. The full procedure outperformed the tabu search of Wang and Wright [55] on five subdistricts of Indiana.

In 1990, there was a mathematical modeling competition that posed the following problem. Given an Eulerian, strongly connected, directed graph G, construct two plowing routes that begin and end at the same point, cover G, and minimize the maximum route length. The graph represented the roads of a district in Wicomico County, Maryland and had 139 vertices and 374 arcs. Atkins et al. [7] and Robinson et al. [52] develop a cluster-first, route second heuristic. Clustering is done by building two subgraphs iteratively by

adding streets from G to the subgraph that has the shortest cumulative length. Robinson et al. [52] give a route-first, cluster-second method that constructs an Eulerian cycle and attempts to split it into two routes. The authors note this method does not perform as well as clustering first in minimizing maximum route length. In addition, some routes have a large number of U-turns. Chernak et al. [18] extend the problem to include the notion of hierarchy (i.e., some streets are higher priority than others). The authors construct primary routes that service roads of highest priority and extend the primary routes to include all roads.

Salim et al. [53] propose the Snow Removal Asset Management (SRAM) system to route the plows of Black Hawk County, Iowa. SRAM takes into account service hierarchy and attempts to minimize maximum service time. In addition to decision rules derived from industry experts, SRAM constructs routes by extending partial routes of a given class in a greedy way by adding the nearest road of the same class that does not violate the route time limit. A 1.9% to 9.7% reduction in maximum traversal time is reported over the existing solution used by the county.

Gendreau et al. [32] use GeoRoute (a commercial software package) to solve a mixed rural postman problem with turn penalties where the objective is to minimize a linear combination of travel time and the number of U-turns, left turns, straight crossings, and street changes (effectively favoring right-hand turns). The software clusters first and routes second. The routing portion of the software is an adaptation of the GENIUS procedure of Gendreau et al. [31] which handles the additional routing objectives. The authors study the effects of the weights of the different objectives on the route pattern, specifically whether the route focused on plowing city blocks or plowing entire street lengths. Campbell and Langevin [16] also use GeoRoute in three Canadian cities.

A book by Omer [47] describes several variants of the CPP that relate to snow plowing operations. The author incorporates different constraints to the problem, such as rural instances, where only a subset of streets need to be plowed, distance limits for plows, time limits for plows, capacity constraints due to limited carrying capacity of chemicals, street service priority, and required frequency of service for different streets. Using an adaptive search procedure following by a simulated annealing procedure, the author matched the previous best known solution on 89 out of 115 benchmark instances, and found new best solutions for 18 of the benchmark instances.

Haghani and Qiao [35] developed a decision support system for the Maryland State Highway Administration Office of Maintenance to develop snow emergency routes for Calvert County, Maryland. The authors combine an integer programming formulation that takes into account time windows with slight adaptations to existing arc routing heuristics to solve the arc routing problem for snow removal. Total deadhead was reduced between 15 and 54%. Haghani and Qiao [36] extend this work to minimize the total number of trucks in addition to the total deadhead. Based on test results conducted for Calvert county, Maryland, it was found that the number of trucks was reduced by 15% and the total deadhead distance was reduced by 4%. Toobaie and Haghani [54] adds the constraint that each truck route must be connected, which reduces driver confusion. A three-step heuristic is presented that first minimizes the number of trucks, balances the routes, and then constructs connected routes. On an instance for Calvert County, Maryland, the number of trucks was reduced by 15% and the total deadhead was reduced by up to 70%.

The Plowing with Precedence Problem is a new variant of the WPP. The concept of precedence in the PPP, motivated by the option to deadhead a street after it has been

plowed, is a practical consideration that has not been addressed in the current literature on snow plowing and arc routing.

3.3 Formulation

We formulate the PPP as an integer program (IP). The formulation of the Windy Postman Problem given by Win [56] cannot be adapted completely because the formulation does not capture the precedence relations in the PPP. One key observation is that we do not know a priori how many steps (arcs traversed) are in an optimal solution to the PPP. In contrast, an optimal solution to a TSP on n nodes will have n steps or arcs. Since we do not know the number of steps in an optimal solution, we determine an upper bound in order to formulate the PPP as an IP. The bound is large. Therefore, the IP formulation has a large number of variables and constraints. As a result, the IP and its LP relaxation can become computationally burdensome to solve, even for small instances.

Consider an edge (v_a, v_b). For a given route, this edge is visited some number of times. The route can be decomposed into cycles where each cycle begins and ends by traversing edge (v_a, v_b). The worst case is that each cycle exists to plow another edge (v_i, v_j). A cycle cannot consist entirely of deadhead (if it did, we would remove it from the solution). This is illustrated in Figure 3.3. The bottom cycle is required because it is needed to plow edge (v_i, v_j). However, the top cycle, which is only deadhead, can be removed from the solution, and will not be part of the optimal solution. The maximum number of times that edge (v_a, v_b) can be traversed is equal to $|A|$, the total number of edges. For our problem, each edge must be plowed twice, so the maximum number of times an edge can be traversed is $2|E|$.

We consider a route to be a sequence of steps and the maximum number of steps is

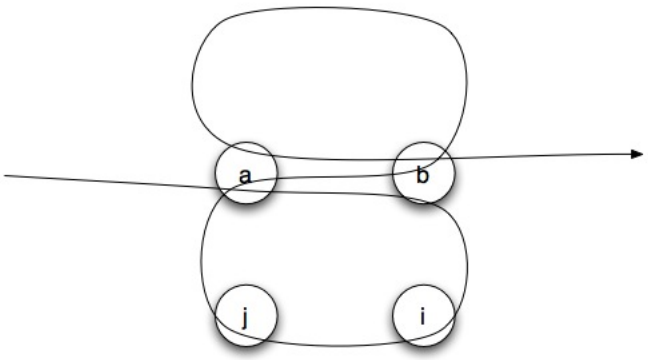


Figure 3.3: Required and non-required sub-cycles

equal to $L = 2|E| \cdot 2|E|$. For each step, we have three types of traversals: plow an edge, deadhead travel across an edge, or idle (do nothing). We use a dummy edge (v_0, v_0) for idling purposes.

We define notation and variables as follows.

1. V is the set of vertices
2. E is the set of edges
3. $t \leq L = (2|A|)^2$ is the index representing the t^{th} step traversed in the route
4. c_{ij}^{+} is the cost of plowing from vertex i to vertex j
5. c_{ij}^{-} is the cost of traversing from vertex i to vertex j after plowing
6. M is a sufficiently large number
7. $x_{ijt} = \begin{cases} 1, \text{ if edge } (v_i, v_j) \text{ is plowed at step } t \\ 0, \text{ otherwise.} \end{cases}$
8. $y_{ijt} = \begin{cases} 1, \text{ if edge } (v_i, v_j) \text{ is deadheaded at step } t \\ 0, \text{ otherwise.} \end{cases}$
9. $z_{ijt} = \begin{cases} 1, \text{ if the route is idling on edge } (v_i, v_j) \text{ at step } t \\ 0, \text{ otherwise.} \end{cases}$

10. $\gamma_{ijt} = x_{ijt} + y_{ijt} + z_{ijt}$ is equal to $\begin{cases} 1, \text{ if edge } (v_i, v_j) \text{ is traversed (plow,} \\ \text{ deadhead, or idle) at step } t \\ 0, \text{ otherwise.} \end{cases}$

11. $\phi_{ijt} = \begin{cases} 1, \text{ if edge } (v_i, v_j) \text{ or } (v_j, v_i) \text{ is first plowed at step } t \\ 0, \text{ otherwise.} \end{cases}$

Our formulation is given by:

$$\text{Minimize } \sum_{i,j,t} c_{ij}^{+} x_{ijt} + c_{ij}^{-} y_{ijt} \tag{3.1}$$

subject to

$$\sum_t (x_{ijt} + x_{jit}) = 2 \qquad \forall\ (i,j) \neq (0,0) \tag{3.2a}$$

$$\sum_i \gamma_{i,j,t-1} = \sum_k \gamma_{j,k,t} \qquad \forall\ j,t \tag{3.2b}$$

$$x_{00t} = y_{00t} = 0 \qquad \forall\ t \tag{3.2c}$$

$$z_{ijt} = 0 \qquad \forall\ (i,j) \neq (0,0), t \tag{3.2d}$$

$$M(1 - z_{00t}) \geq \sum_{i,j,t<t'} (x_{ijt'} + y_{ijt'}) \qquad \forall\ t \neq 0 \tag{3.2e}$$

$$M(1 - \phi_{ijt}) \geq \sum_{t'<t} y_{ijt'} \qquad \forall\ (i,j) \neq (0,0), t \tag{3.2f}$$

$$M(1 - \phi_{ijt}) \geq \sum_{t'<t} y_{jit'} \qquad \forall\ (i,j) \neq (0,0), t \tag{3.2g}$$

$$\sum_j x_{0j0} = 1 \tag{3.2h}$$

$$\sum_{i,j} \gamma_{ijt} = 1 \qquad \forall\ t \tag{3.2i}$$

$$x_{ijt} + x_{jit} \geq \phi_{ijt} \qquad \forall\ i,j,t \tag{3.2j}$$

$$\phi_{ijt} \leq \sum_{t<t'} (x_{ijt'} + x_{jit'}) \qquad \forall\ i,j,t \tag{3.2k}$$

$$\sum_t (\phi_{ijt} + \phi_{jit}) = 1 \qquad \forall\ i,j \tag{3.2l}$$

$$x_{ijt}, y_{ijt}, z_{ijt}, \phi_{ijt} \in \{0,1\} \qquad \forall\ i,j,t \tag{3.2m}$$

In (3.1), we seek to minimize cost. Constraint (3.2a) requires that each edge be plowed twice, once for each side of the street. In constraint (3.2b), if a route enters vertex j at step $t-1$, it must leave vertex j at step t by either plowing, deadheading, or idling. Constraint (3.2c) dictates that we do not plow or deadhead the dummy edge (v_0, v_0). Constraint (3.2d) requires that the route can only idle on the dummy edge (v_0, v_0). Constraint (3.2e) requires that if idling is performed at step t, then no plowing or deadhead

can take place afterwards. Constraint (3.2e) together with constraint (3.2d) force the route to end at the depot (vertex 0) by requiring that a plow idles at the depot after all plowing and deadheading is completed. Constraints (3.2f) and (3.2g) require that if edge (v_i, v_j) is first plowed at time t, then there is no deadheading across (v_i, v_j) or (v_j, v_i) before time t. Constraint (3.2h) requires that the route begin at the depot. Constraint (3.2i) requires that only one traversal is performed per step. Constraints (3.2j), (3.2k), and (3.2l) fix the definition of ϕ_{ijt}, which describes the first step that edge (v_i, v_j) is plowed. We illustrate this with an example. Consider edge (v_i, v_j). We can represent the set of $\{x_{ijt}\}_t$ as a vector whose elements are enumerated by the index t, where $0 \leq t < L$. Suppose $L = 7$. Let

$$[x_{ijt}] = [0, 0, 0, 0, 0, 1, 0] \tag{3.3}$$

$$[x_{jit}] = [0, 0, 1, 0, 0, 0, 0] \tag{3.4}$$

$$[x_{ijt}] + [x_{jit}] = [0, 0, 1, 0, 0, 1, 0] \tag{3.5}$$

Equation (3.3) says that edge (v_i, v_j) is plowed from v_i to v_j at step 6, and (3.4) says that edge (v_i, v_j) is plowed from v_j to v_i at step 3. In constraint (3.2j), if $x_{ijt} + x_{jit} = 0$ then $\phi_{ijt} = x_{ijt} + x_{jit} = 0$. Otherwise, if $x_{ijt} + x_{jit} = 1$, then constraint (3.2k) requires that $\phi_{ijt} \leq \sum_{t<t'} x_{ijt'} + x_{jit'}$. $x_{ijt} + x_{jit}$ is equal to 1 only two times. At $t = 2$, $\sum_{t<t'} x_{ijt'} + x_{jit'}$ is equal to 1 and at $t = 5$ the summand is equal to 0. Therefore, $\phi_{ijt} = 0$ for all t except for $t = 2$, in which case we have the constraint $\phi_{ijt} \leq 1$. Constraint (3.2l) forces one of the ϕ_{ijt}'s to be 1 for some t, which forces $\phi_{ijt} = 1$ for $t = 2$. This gives the following result

$$[\phi_{ijt}] = [0, 0, 1, 0, 0, 0, 0]. \tag{3.6}$$

This formulation has a large number of variables and constraints. In terms of the number of vertices ($|V|$), the number of edges ($|E|$), and the number of steps ($L = (2|A|)^2$), there are approximately $(2|A|+1)L = 8|A|^3+4|A|^2$ variables and $(4|A|+V+1)L+|A|+1 = 16|A|^3 + 4(V+1)|A|^2 + |A| + 1$ constraints.

3.4 Solution Methodology

As shown by Win [56], the Windy Postman Problem can be formulated and solved as an IP that gives the number of times each edge is traversed. A directed Eulerian graph is generated next and a route can be constructed using Fleury's algorithm [40] (described later).

Our solution methodology uses an adaptation of Win's IP whose solution provides the number of times each edge is to be traversed (we call this a partial solution). Using this partial solution, we construct an initial route using an alternative to Fleury's algorithm. This solution is optimal if precedence is ignored and would optimally solve any WPP instance that did not incorporate precedence. To incorporate precedence, we provide a heuristic to improve the solution while taking precedence into account. Our heuristic uses a local search procedure followed by reinitialization. Finally, we apply a pruning procedure to remove deadhead cycles and clean up the best routes. We call this method Cycle Permutation Local Search (CPLS).

3.4.1 Partial Solution

We formulate our integer program (denoted by IPX) that is adapted from the formulation given by Win [56]. Let

x_{ij} be the number of times edge (v_i, v_j) is plowed

y_{ij} be the number of times edge (v_i, v_j) is traversed after plowing

c_{ij}^{+} be the cost of plowing uphill

c_{ji}^{+} be the cost of plowing downhill

c_{ij}^{-} be the cost of traversing uphill after plowing

c_{ji}^{-} be the cost of traversing downhill after plowing

$\delta(i)$ be the set of edges incident to vertex v_i.

IPX is given by

$$\text{Minimize } \sum_{i,j} \left(c_{ij}^{+} x_{ij} + c_{ij}^{-} y_{ij} \right) \tag{3.7}$$

subject to

$$x_{ij} + x_{ji} = 2 \qquad \forall\ (v_i, v_j) \in A \tag{3.8}$$

$$\sum_{(v_i, v_j) \in \delta(i)} ((x_{ij} + y_{ij}) - (x_{ji} + y_{ji})) = 0 \qquad \forall\ v_i \in V \tag{3.9}$$

$$x_{ij}, y_{ij} \geq 0 \text{ and integer.} \tag{3.10}$$

The objective function (5.3) minimizes the cost of the route without regard to precedence. Constraint (5.4) requires each street to be plowed twice (once for each side). Constraint (5.5) requires that the degree of each vertex be balanced. Constraint (5.6) enforces non-negativity and integrality of the variables.

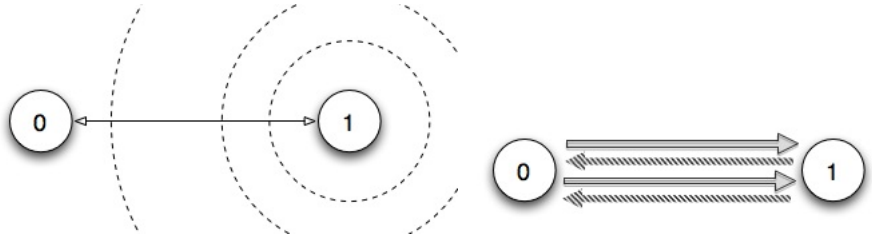
Solving this IP gives a number of times to plow and deadhead each edge in each direction and the associated directed Eulerian graph. By adding the number of times to

plow and number of times to deadhead, we can determine the number of times each edge must be traversed. We call this information a partial solution and use it to construct a route.

3.4.2 Lower Bound

The optimal solution of IPX (5.3) gives a lower bound to the PPP since it solves the same problem but ignores precedence. When considering the transition from the partial solution to a final route, due to the importance played by the order of edges on the route, it is not clear that there is a feasible route that produces the value of the objective function. To illustrate this, we note that there are three ways to plow edge (v_i, v_j): $(x_{ij}, x_{ji}) = (2,0), (1,1), (0,2)$. In the case where $(x_{ij}, x_{ji}) = (1,1)$, we plow once in each direction, i.e., we plow the first time we traverse each direction of the edge. However, a problem arises when we consider the case where $(x_{ij}, x_{ji}) = (0,2)$. In this case, we plow downhill for both sides of the street, which can pose a problem if the IP requires that we traverse uphill after plowing ($y_{ij} > 0$). In particular, there is no guarantee that we can plow a street downhill before traversing the street uphill. We illustrate this in the following example. Consider the graph in Figure 3.4(a) where vertex 1 is uphill from the depot (vertex 0) and $c^+_{0,1} = 10$, $c^+_{1,0} = 2$, $c^-_{0,1} = 1$, and $c^-_{1,0} = 1$. Solving IPX produces the solution $x_{0,1} = 0$, $x_{1,0} = 2$, $y_{0,1} = 2$, and $y_{1,0} = 0$ with objective function value $2(2) + 2(1) = 6$. The plow follows the route $\{0, 1, 0, 1, 0\}$ and plows downhill both times (Figure 3.4(b)). However, our route must begin at the depot (vertex 0). The first step in the route, traversing from vertex 0 to vertex 1, is supposed to be deadhead since $y_{0,1} = 2$. Because the edge (0,1) has not been plowed yet, it is necessary to plow uphill. This is not reflected in the IP solution and the corresponding objective function. This is

an infeasibility that occurs because the IP does not take precedence into account. The optimal route is $\{0, 1, 0\}$ with a cost of $10 + 2 = 12$. We note that the IPX solution, with an objective value of 6, is a lower bound.



(a) Vertex 1 uphill from the depot (vertex 0) (b) Dashed arrows indicate plowing and solid arrows indicate deadhead

Figure 3.4: Graph with two nodes and the associated Eulerian route

There are cases where plowing downhill is advantageous and it may not be possible to construct a route that obeys the partial solution given by the IP while simultaneously plowing downhill before deadhead travel uphill. In these cases, the lower bound is not tight. The gap between the optimal solution and the lower bound can be arbitrarily large by increasing the uphill plowing cost to a large value M. In this case, the solution to the IP is the same as above, with an objective function value of 6. The actual route cost is at least $M + 2 >> 6$, which is the cost of plowing uphill and plowing downhill.

Even if we place a restriction on the uphill plowing costs, the lower bound may not be tight. We expand the graph in Figure 3.4(a) so that the depot has arbitrarily high degree (n). If the cost of plowing away (uphill) from vertex 0 is 10, then the IP solution has each edge being plowed downhill twice. Plowing uphill once on each edge is unavoidable and the actual route cost is $n(10 + 2) >> n(6)$. So, the difference between the cost of the optimal route and the lower bound can be arbitrarily large.

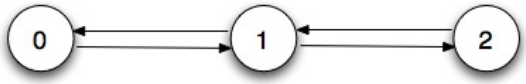
We note that the solution to IPX and the solution to the LP relaxation of the IP

formulation give a lower bound to the PPP. We do not know if one bound is consistently tighter than the other.

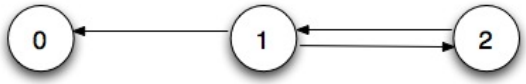
Given a partial solution, there are many possible routes that traverse the associated Eulerian graph. Since the order of a route is important, we seek to generate a route that avoids infeasibilities. This will bring the actual cost of a route closer to the lower bound given by the IP (or possibly exactly to the lower bound, where all infeasibilities are removed). Next, we construct an initial solution and then improve it.

3.4.3 Initial Route Construction

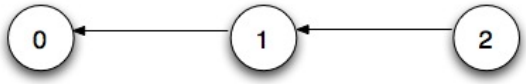
Fleury's algorithm [40] (see Table 3.1) can be used to construct a route on a directed or undirected Eulerian graph by following edges that do not disconnect the graph and removing those edges already traveled. Since we are working with a variant of the Windy Postman Problem, the partial solution provides a directed Eulerian graph. Consider the example in Figure 3.5(a). Beginning at vertex 0, Fleury's algorithm considers vertex 1 since it is the only choice and removes edge $(0, 1)$. This results in Figure 3.5(b). From here, there is a choice for the plow to travel back to vertex 0 or to vertex 2. Fleury's algorithm prohibits the plow traveling back to vertex 0 because it would disconnect the graph. Fleury's algorithm considers vertex 2 next. This results in Figure 3.5(c). From there, the plow travels back to vertex 1 (Figure 3.5(d)) and the algorithm terminates at vertex 0 (Figure 3.5(e)). The final route is $\{0, 1, 2, 1, 0\}$.



(a) Step 1



(b) Step 2

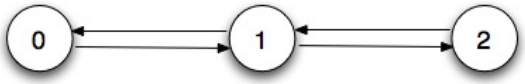


(c) Step 3

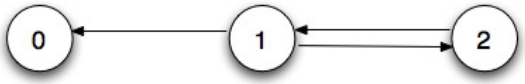
(d) Step 4

(e) Step 5

Figure 3.5: Progression of edge deletion in Fleury's algorithm



(a) Step 1



(b) Step 2

(c) Step 3

(d) Step 4

(e) Step 5

Figure 3.6: Progression of edge deletion in Find-Route method

```
Set:
T = route
u, v ∈ V
E = the set of (potentially duplicate) edges given by the partial solution

do {
        Append vertex u to T
        Choose an edge (u, v) that does not disconnect the graph
                (unless there is no choice, in which case choose any edge)
        Remove (u, v) from E
} while there are still edges to remove
```

Table 3.1: Fleury's algorithm

```
Set:
T = route
u, v ∈ V
E = the set of (potentially duplicate) edges given by the partial solution

Findroute(u) {
      For each edge (u, v) ∈ E
            Remove (u, v) from E
            Findroute(v)
       Prepend u to T
}
```

Table 3.2: FindRoute method for construction of an Eulerian cycle

We can use Fleury's algorithm to generate a route using the solution given by the partial solution. However, we use the method given in Table 3.2 which avoids the difficulty of determining whether removing an edge disconnects a graph.

Findroute(0) (see Table 3.2) produces a route T that starts at vertex 0 and visits each of the edges in the Eulerian graph associated with the partial solution exactly once. To motivate the FindRoute method, we note that if C is any cycle in an Eulerian graph $G = (V, E)$, then, after removing the edges of C from E, the remaining components of G will also be Eulerian. The FindRoute method finds a cycle in G, removes its edges, and repeats with each remaining connected component. An example on the same graph (Figure 3.5(a)) is described in Table 3.3 and Figures 3.6(a)-3.6(e).

Set:
T = route
$u, v \in V$
E = Figure 3.6(a)
Findroute(0) {
    For edge $(0, 1)$
        Remove $(0, 1)$ from E (Figure 3.6(b))
        Findroute(1) {
            For edge $(1, 0)$
                Remove $(1, 0)$ from E (Figure 3.6(c))
                Findroute(0) {
                    Prepend 0 to T
                }
            For edge $(1, 2)$
                Remove $(1, 2)$ from E (Figure 3.6(d))
                Findroute(2) {
                    For edge $(2, 1)$
                        Remove $(2, 1)$ from E (Figure 3.6(e))
                        Findroute(1) {
                            Prepend 1 **to** T
                        }
                    Prepend 2 **to** T
                }
        }
        Prepend 1 **to** T
    }
    Prepend 0 **to** T
}

Final route $T = \{0, 1, 2, 1, 0\}$

Table 3.3: FindRoute method applied to the graph in Figure 3.6

3.4.4 Local Search

It is possible that FindRoute will produce a route with infeasibilities. For such a route, we seek to reduce the number of infeasibilities by using a local search procedure. We define the fitness function and neighborhood used in this procedure.

3.4.4.1 Route Fitness

We represent a route as a sequence of vertices: $\{v_{\phi(1)}, v_{\phi(2)}, v_{\phi(3)}, \cdots, v_{\phi(k)}\}$ that satisfy the following constraints:

1. $v_{\phi(i)} \in V$
2. $v_{\phi(1)} = v_{\phi(n)} = v_0$
3. $(v_{\phi(i)}, v_{\phi(i+1)}) \in E$
4. $\cup_{i=1}^{k-1}\{(v_{\phi(i)}, v_{\phi(i+1)})\} = E$.

The cost of a route T is given by

$$\tau(T) = \sum_{i=1}^{k-1} c_{\phi(i)\phi(i+1)} \tag{3.11}$$

where

$$c_{\phi(i)\phi(i+1)} = \begin{cases} c^{+}_{\phi(i)\phi(i+1)}, \text{ if the edge is plowed in route } T \\ c^{-}_{\phi(i)\phi(i+1)}, \text{ if the edge is traversed with deadhead in route } T. \end{cases} \tag{3.12}$$

The decision to plow or not to plow an edge depends on the decision sequence described in Table 3.4. We illustrate this with the route $\{0, 1, 2, 1, 3, 1, 3, 1, 0\}$ for the graph in Figure 3.7 where vertex 1 is at the top of a hill. Each edge in the route is labeled by the type of decision in Table 3.4.

if edge has been plowed twice
  do not plow (1)
else if edge has not been plowed at all
  plow (2)
else if going downhill
  plow (3)
else if route is not going downhill later
  plow (4)
else do not plow (5)

Table 3.4: Decision sequence for plowing or deadheading (numbers in parentheses are used in Figure 3.7)

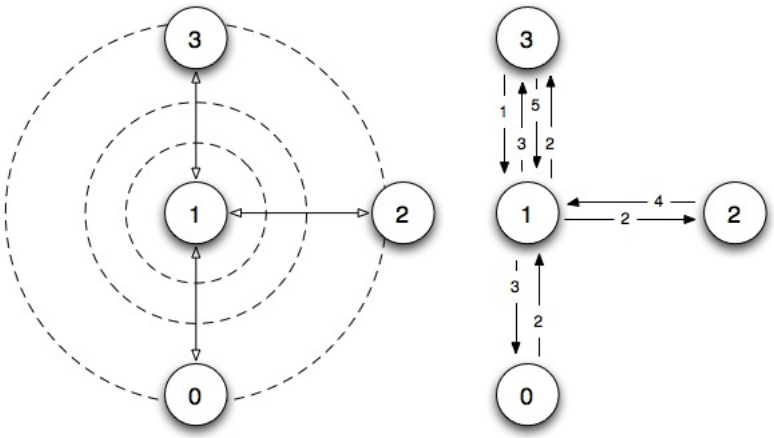


Figure 3.7: Left: Graph with vertex 1 on a hill. Right: A route on the graph

3.4.4.2 Neighborhood Definition

We improve the route generated by the Initial Route Construction with a local search procedure. The neighborhood of a route s, $N(s)$, is the set of all routes that can be obtained by a combination of the following two moves:

1. Cycles in the route are permuted
2. Cycles in the route are reversed.

To describe these two moves, we consider the following route (4.8) on the graph in Figure 5.2, where the v's have been suppressed.

$$\{0, \underbrace{1,3}_{1}, 0, \underbrace{3,1}_{2}, 0, \underbrace{1,2,3}_{3}, 0, \underbrace{1,3,2,1}_{4}, 0\} \tag{3.13}$$

The bracketed elements in (4.8) are cycles that begin and end at vertex 0. Because they begin and end at the same point, these cycles can be permuted, such as in (4.9).

$$\{0, \underbrace{3,1}_{4}, 0, \underbrace{1,2,3}_{1}, 0, \underbrace{1,3,2,1}_{2}, 0, \underbrace{1,3}_{3}, 0\} \tag{3.14}$$

This new solution visits each edge the same number of times, just in a different order. This permutation can be done with cycles that begin and end at any vertex. For example, the cycles of (4.8) that begin and end at vertex 1 are shown in (4.10).

$$\{0, 1, \underbrace{3,0,3}_{1}, 1, \underbrace{0}_{2}, 1, \underbrace{2,3,0}_{3}, 1, \underbrace{3,2}_{4}, 1, 0\} \tag{3.15}$$

Beginning with cycle (4.9), we can reverse cycle 2 and obtain the route in (4.11).

$$\{0, \underbrace{3,1}_{4}, 0, \underbrace{1,2,3}_{1}, 0, \underbrace{1,2,3,1}_{2}, 0, \underbrace{1,3}_{3}, 0\} \tag{3.16}$$

Suppose the number of cycles for a given vertex is k. Then the number of possible permutations is $k!$. Therefore, the size of the neighborhood of a solution can become very large if there are a moderate number of cycles. The number of cycles for a given vertex is related to the degree of the same vertex. For example, consider a four way intersection. To plow both sides of all four incident streets, a plow visits the intersection four times and produces four cycles for a total of $4! = 24$ permutations (see Figure 3.8). We note that, in practice, intersections rarely have more than four incident streets. In order to reduce the running time of our algorithm, if the number of cycles is larger than four, we simply restrict the neighborhood definition to be those cycles obtained by the first $4! + k$ random permutations, which allows for linear growth of the neighborhood size. We avoid progressing through the set of permutations lexicographically by generating a new permutation from a previous one by swapping two random cycles. Our local search heuristic is given in Table 4.2.

Begin with the solution s given by initial route construction
Let $\tau(x)$ be the fitness of solution x
Repeat:
For each solution $s' \in N(s)$
Calculate $\tau(s')$
Define s_{min} to be a solution where $\tau(s_{min}) \leq \tau(s') \ \forall \ s' \in N(s)$
If $\tau(s_{min}) < \tau(s)$ set $s = s_{min}$
Else break

Table 3.5: Local search heuristic

Recall the graph in Figure 5.2. If we begin with the initial solution in (4.8), then the solution in (4.11) is produced by computing the neighborhood of (4.8). We note that the route in (4.11) is exactly the low cost route given in the introductory example in Figure

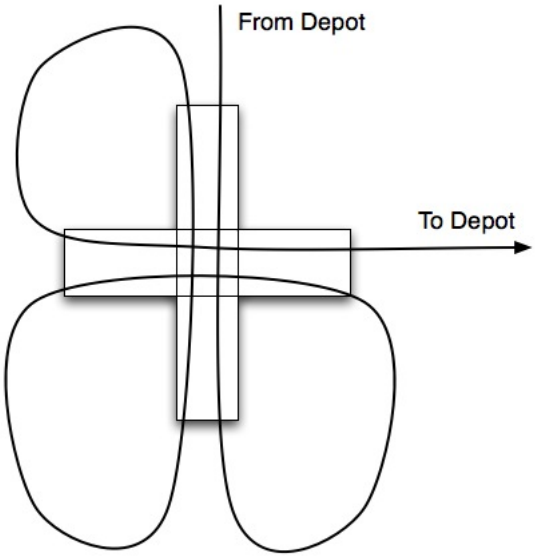


Figure 3.8: An intersection with four streets produces four cycles

3.2.

It is possible that no successive use of permutations and reversals will result in the optimal solution, because the length of the route (number of edges traversed) is fixed by the partial solution given by the initial IP. Consider the graph in Figure 5.2. The optimal route is

$$\{0, 3, 1, 0, 3, 1, 2, 3, 2, 1, 2, 1, 3, 2, 0\}. \tag{3.17}$$

The cost of the route in (3.17) is 28, which is less than the route generated by local search. The local search procedure never produces the optimal solution because the length of the route is fixed at traversing 15 edges, while the optimal solution traverses 14 edges. The lower bound provided by the partial solution is 27.

3.4.5 Reinitialization and Solution Pruning

In order to search more of the solution space, after the local search procedure has finished, we reinitialize the solution and run the local search procedure again. The reinitialization procedure is similar to the local search procedure, except that it chooses a new solution in the neighborhood randomly. This process is repeated several times to generate a new route.

The combination of local search and reinitialization is run for a specified number of iterations. The best solution is returned. This solution can be improved by pruning (removing) cycles that involve only deadhead travel. For example, consider the route $\{0, 1, 0, 1, 0\}$ on the graph in Figures 3.4(a) and 3.4(b). Plowing occurs on the first two edge traversals, $0 \rightarrow 1$ and $1 \rightarrow 0$. However, the last two edge traversals that constitute the cycle $\{0, 1, 0\}$ are deadhead and may be pruned. The result of pruning is the lower-cost route $\{0, 1, 0\}$.

3.4.6 Summary Example

We now give a summary example that fully demonstrates our heuristic. Consider the graph in Figure 3.9. The associated costs are given in columns 2-5 of Table 3.6. In column 6 of Table 3.6, we give the number of times that each edge should be traversed from the partial solution IP.

From the partial solution, the initial route is given below (cost 135)

$$\{0, 2, 1, 0, 2, 1, 0, 3, 2, 3, 2, 3, 2, 7, 2, 7, 2, 7,$$

$$6, 3, 2, 7, 6, 3, 5, 3, 5, 3, 6, 5, 4, 3, 6, 5, 4, 3, 1\}.$$

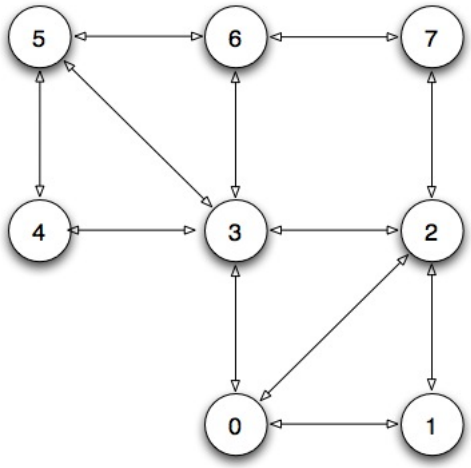


Figure 3.9: Graph with eight vertices (vertex 0 is the depot)

Arc	c_{ij}^{+} (plow ↑)	c_{ji}^{+} (plow ↓)	c_{ij}^{-} (traverse ↑)	c_{ji}^{-} (traverse ↓)	Number of Traversals
(0,1)	6	4	1	1	2
(0,2)	8	7	1	1	2
(0,3)	7	7	1	1	2
(3,2)	6	4	1	1	6
(2,1)	8	6	1	1	2
(3,6)	9	3	1	1	4
(6,7)	6	4	1	1	2
(2,7)	9	5	1	1	4
(3,4)	3	3	1	1	2
(4,5)	8	4	1	1	2
(3,5)	9	6	1	1	4
(5,6)	4	3	1	1	2

Table 3.6: Route cost and partial solution of the graph given in Figure 3.9 (↑ uphill, ↓ downhill)

The lower bound provided by the initial partial solution is 126. The route produced by the local search with reinitialization is given by

$$\{0, 2, 1, 0, 2, 3, 2, 7, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0,$$

$$3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}$$

and has a cost of 130. After pruning to remove the deadhead cycle $\{2, 7, 2\}$, the final route with cost 128 is given by

$$\{0, 2, 1, 0, 2, 3, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0,$$

$$3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}.$$

A summary of CPLS is given in Table 3.7.

Let:
T = Initial Solution
B = Best Solution
LocalSearch(T) = returns the best route in the neighborhood of the route T
Reinitialize(T) = returns a random route by reinitializing the route T
Prune(T) = returns a route that is the pruned version of T

do {
    Set T' = LocalSearch(T)
    If T' is a new best solution
        Set $B = T'$
    Set T = Reinitialize(T)
} until no new best solution has been obtained for 10 iterations
Return Prune(B)

Table 3.7: Summary of CPLS for the PPP

3.5 Computational Experiments

Our algorithm CPLS was implemented in C++ using CPLEX 12.2 on a single thread of a 1.86 GHz Intel Core2Duo Processor. To test CPLS, we modified instances of

the Windy Rural Postman Problem (a variant of the WPP where only a subset of the streets need to be traversed) given by Corberán et al. [21]. For each instance, we retained the overall structure, but removed the concept of required edges and forced every edge to be plowed twice. The two traversal costs for each edge were set to the plowing costs (c_{ij}^{+} and c_{ji}^{+}). The deadhead costs for the PPP (c_{ij}^{-} and c_{ji}^{-}) were generated in the following way:

$$c_{ij}^{-} = c_{ji}^{-} = \text{ceiling}[0.5 \times \text{rand}() \times \min(c_{ij}^{+}, c_{ji}^{+})] \tag{3.18}$$

The function rand() returns a random number between 0 and 1. The set of instances vary in the number of vertices and the number of edges. Our instances can be found at http://www.rhsmith.umd.edu/faculty/bgolden/ as well as in Appendix A.

We attempted to compute the lower bound given by the linear programming (LP) relaxation of the IP formulation for the PPP, but found that the computations were intractable because of excessive running times and memory requirements. We removed the variable ϕ and its associated constraints (3.2e, 3.2f, 3.2g, 3.2k, 3.2i, and 3.2l), to make the LP easier to solve. Using this LP, we found lower bounds for 43 of the 45 instances.We were unable to solve two instances and find lower bounds because of memory constraints. When compared to the lower bounds given by the partial solution (IPX), the LP relaxation produced the same lower bound on 20 of the 43 instances. On 23 instances where it did not produce this lower bound, the LP relaxation was worse than IPX for each instance by 0.31% on average. While the LP relaxation for the PPP was a relatively tight lower bound, it was always outperformed by the lower bound generated by IPX. Therefore, we compare our solutions to the lower bound given by IPX.

Our results are given in Table 3.8. Column 1 gives the instance name from Corberán

et al. [21]. Column 2 gives the number of vertices. Column 3 gives the number of edges. Column 4 gives the average deviation in cost, which we define to be the average difference between plowing up and plowing down, divided by the average of the plowing costs over all directions. It is a measure of the average discrepancy in cost between plowing up and plowing down. Column 5 gives the lower bound produced by the IP. Column 6 gives the lower bound produced by the LP relaxation. Column 7 gives the initial solution obtained by the FindRoute method. Column 8 gives the solution obtained after running the local search procedure with reinitialization. Column 9 gives the final solution generated by applying the pruning procedure. Column 10 gives the percentage improvement of the final solution over the initial solution. Column 11 gives the deviation of the final solution from the lower bound. Column 12 gives the running time in seconds.

The results from Table 3.8 show that the algorithm performs very well, averaging a 0.17% deviation from the lower bound, and achieving the lower bound (and hence guaranteed optimal solution) on 27 out of the 45 instances (denoted by 0.00 in column 11). For the solutions that did not achieve the lower bound, the average deviation was 0.43%. From Table 3.8 we make several observations. First, the running time can get very large as the number of nodes increases. Second, as seen in Figure 3.10, deviation from the lower bound generally increases as the average cost deviation increases. When the average cost deviation increases, we think there is more deadhead to avoid plowing uphill. When there is more deadhead, a route visits vertices more often, resulting in a larger number of cycles. When the number of cycles increases, our constrained neighborhood becomes smaller relative to the number of possible permutations. The local search procedure is less thorough and less accurate. It is also possible that the lower bound performs worse on the problems with larger deviations. Third, as the average cost deviation increases, the

number of infeasibilities in the initial solution increases, and the local search procedure produces larger gains over the initial solution, as shown in Figure 3.11. On average, the final solution has a 1.8% improvement over the initial solution.

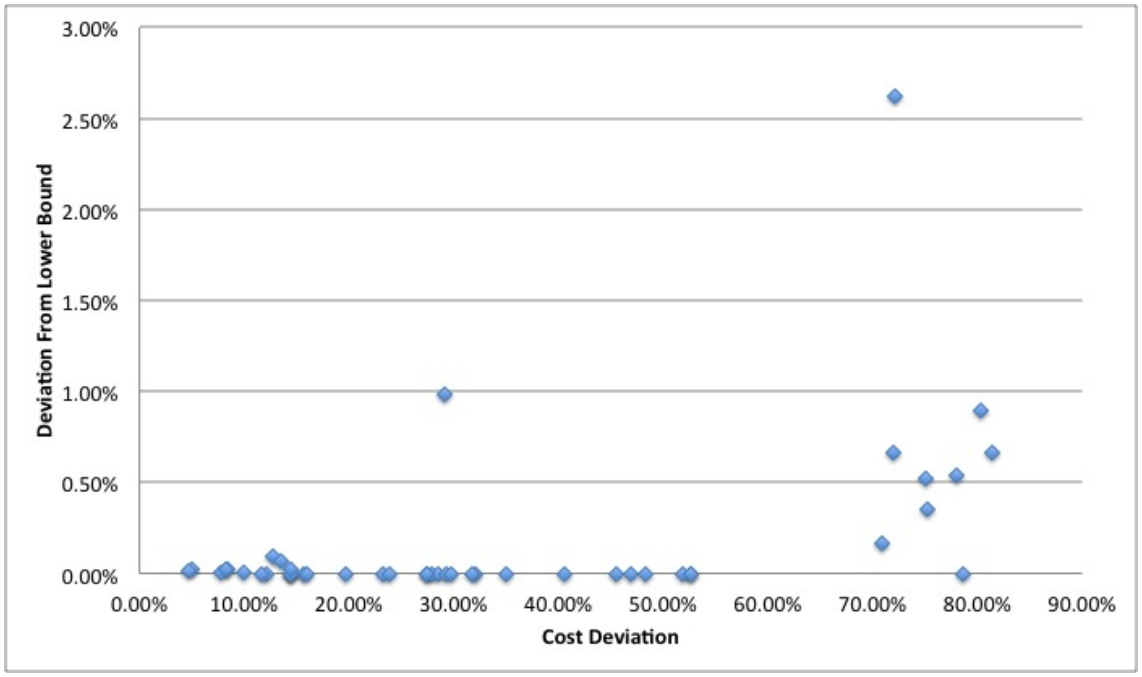


Figure 3.10: Deviation from lower bound vs. cost deviation (from Table 3.8)

To see how performance depends on an instance, we hypothesize that the deviation from the lower bound can be explained by the number of edges, the number of vertices, and the cost deviation of an instance. The fitted regression equation for data taken from Table 3.8 is given by:

$$\begin{aligned}\text{Deviation from Lower Bound} = 0.3038 &+ 0.0109 \times \text{Number of Vertices} \\ &- 0.00677 \times \text{Number of Edges} \\ &+ 0.02874 \times \text{Average Cost Deviation.}\end{aligned}$$

The p-values for the number of vertices, number of edges, and the average cost

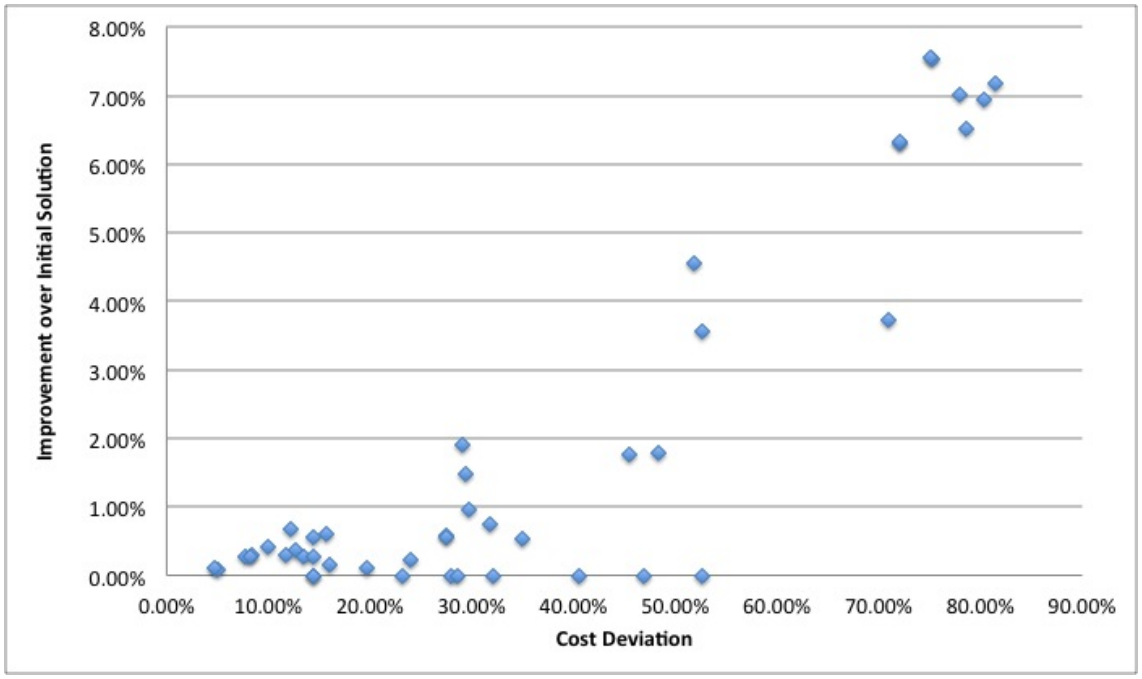


Figure 3.11: Percentage improvement over initial solution vs. cost deviation (from Table 3.8)

deviation were 1.28×10^{-1}, 1.32×10^{-1}, and 3.39×10^{-12}, respectively. It appears that all three variables can be used to explain the variability in deviation from the lower bound. We note that the average cost deviation has the largest impact (because of its largest coefficient) in deviation from the lower bound.

With this observation in mind, we seek to generate instances that will be difficult to solve using the PPP algorithm. We modified the costs in the two largest instances, A3101 and M3101, so that the average deviation ranged from 10% to 70% in the following way. Suppose $c_{ij}^{+} > c_{ji}^{+}$ and let q be the desired deviation in percent. The definition of c_{ij}^{+} is given by

$$c_{ij}^{+} = \text{ceiling}[c_{ji}^{+} + (1 + 2q/100) \times \text{rand}() \times c_{ji}^{+}]. \tag{3.19}$$

The definition in (3.19) sets the uphill plowing cost q percent higher than the down-

hill plowing cost, on average. The computational results for these new graphs based on instances A3101 and M3101 are given in Tables 3.9 and 3.10, respectively. We see that, in both cases, there is a roughly linear increase in running time with respect to average cost deviation and an increase in deviation from the lower bound with respect to average cost deviation. We see that there is a roughly linear increase in percentage improvement in the final solution over the initial solution. Incorporating our results for these instances, we find that the average deviation from the lower bound is now 0.49% over the combined 65 instances.

We note that running times can get very high, approximately 9.5 hours for the most difficult instance: M3101 with a cost deviation of 64.65%. We can reduce the running time by only investigating cycle permutations for a random subset of nodes thereby further reducing the neighborhood of the local search. We restrict the neighborhood by 5, 10 and 20 percent to study the effects of a less intensive local search on deviation from the lower bound. Figure 3.12 shows the asymptotic decrease in deviation from the lower bound as the running time of CPLS increases. We see that almost no reduction in accuracy occurs when sampling 20% of the neighborhood, which results in reducing the running time by two-thirds.

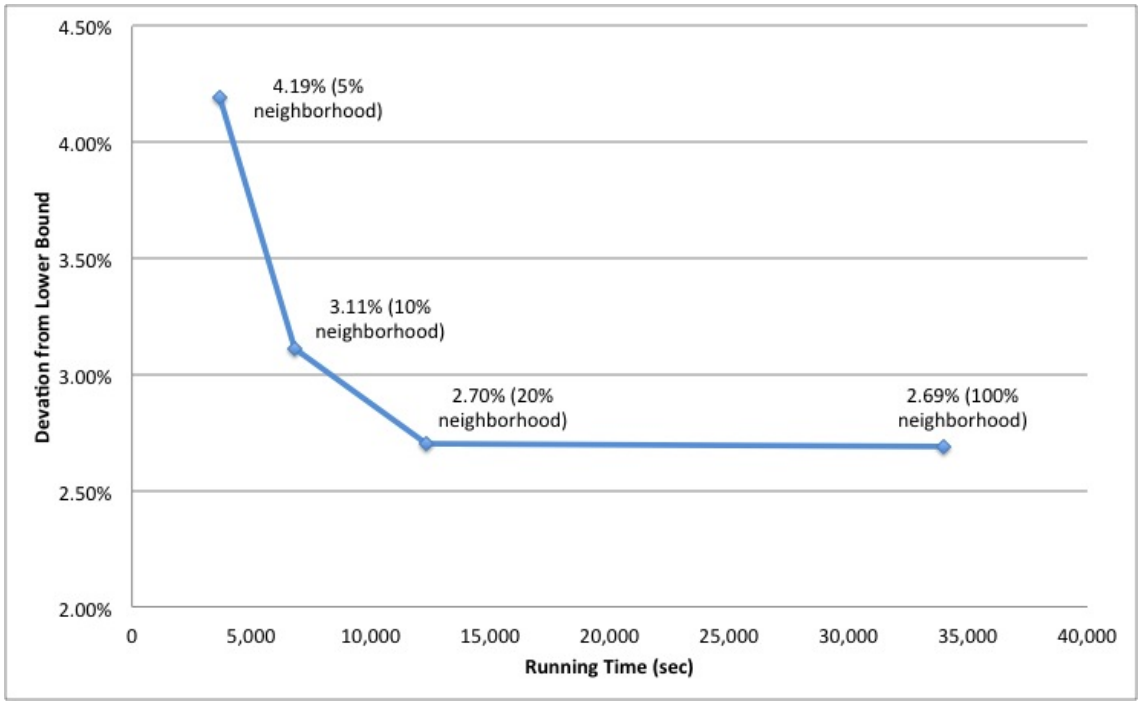


Figure 3.12: Deviation from Lower Bound vs. CPLS Running Time for Instance M3101 with 64.65% Cost Deviation

Instance	Vertices	Arcs	Cost Deviation (%)	LB (IP)	LB (LP)	Solution			IMP (%)	DEV (%)	Time(s)
						Initial	Local Search	Pruned			
A3101	116	174	4.96	24672	24672	24696	24685	24677	0.08	0.02	158.62
HD115	70	141	7.74	11453	11451	11484	11466	11454	0.26	0.01	438.08
HD215	70	162	8.41	13530	13529	13574	13533	13533	0.30	0.02	652.74
HD315	68	153	8.22	12449	12449	12487	12454	12452	0.28	0.02	775.46
HD415	88	171	9.93	12111	12111	12162	12116	12112	0.41	0.01	1677.44
HD515	86	178	12.13	10808	10804	10882	10810	10808	0.68	0.00	2964.12
HD615	88	181	12.71	10012	10008	10058	10025	10021	0.37	0.09	1783.23
HD715	100	188	15.75	8345	8341	8395	8345	8345	0.60	0.00	1140.42
HD815	95	193	11.68	10786	10783	10818	10786	10786	0.30	0.00	3221.27
HD915	96	189	13.50	9041	N/A	9071	9053	9047	0.26	0.07	2337.78
HD115	60	105	72.11	420	420	460	441	431	6.30	2.62	464.39
HG215	68	119	81.46	450	450	488	457	453	7.17	0.67	1783.44
HG315	64	116	72.00	455	453	489	462	458	6.34	0.66	1355.32
HG415	82	148	78.04	553	552	598	560	556	7.02	0.54	1141.52
HG515	92	165	70.92	592	592	616	599	593	3.73	0.17	1141.16
HG615	91	162	75.19	574	573	623	578	576	7.54	0.35	1938.28
HG715	97	174	80.30	557	556	604	566	562	6.95	0.90	267.36
HG815	100	180	75.06	572	572	622	583	575	7.56	0.52	344.09
HG915	97	175	78.67	588	585	629	590	588	6.52	0.00	1386.12
M3101	196	316	4.77	46567	N/A	46620	46575	46573	0.10	0.01	930.45
P0115	11	13	29.09	102	102	105	105	103	1.90	0.98	0.85
P1515	26	37	14.40	841	839	841	841	841	0.00	0.00	1.18
P0215	14	33	14.45	722	722	726	722	722	0.55	0.00	1.18
P0315	28	57	48.28	164	164	167	164	164	1.80	0.00	3.94
P0415	17	35	40.51	130	129	130	130	130	0.00	0.00	2.02
P0515	20	35	23.23	360	360	360	360	360	0.00	0.00	0.75
P0615	24	46	51.92	168	168	176	168	168	4.55	0.00	7.87
P0715	23	47	45.51	277	277	282	279	277	1.77	0.00	26.31
P0815	17	40	32.10	279	279	279	279	279	0.00	0.00	2.43
P0915	14	26	52.63	108	107	112	108	108	3.57	0.00	3.34
P1015	12	20	27.96	170	169	170	170	170	0.00	0.00	0.81
P1115	9	14	52.63	30	30	30	30	30	0.00	0.00	0.14
P1215	7	18	28.57	36	36	36	36	36	0.00	0.00	0.44
P1315	7	10	46.91	67	66	67	67	67	0.00	0.00	0.14
P1415	28	79	27.46	871	870	876	871	871	0.57	0.00	25.05
P1515	26	37	14.40	840	839	840	840	840	0.00	0.00	1.85
P1615	31	94	31.72	930	930	937	932	930	0.75	0.00	49.90
P1715	19	44	34.98	381	380	383	381	381	0.52	0.00	2.80
P1815	23	37	27.43	361	359	363	361	361	0.55	0.00	2.47
P1915	33	54	29.35	464	463	471	464	464	1.49	0.00	8.52
P2015	50	98	16.05	1310	1309	1312	1310	1310	0.15	0.00	10.23
P2115	49	110	29.71	931	931	940	933	931	0.96	0.00	215.03
P2215	50	184	14.45	4256	4255	4269	4259	4257	0.28	0.02	218.69
P2315	50	158	23.93	1781	1781	1785	1781	1781	0.22	0.00	14.02
P2415	41	125	19.63	1903	1903	1905	1903	1903	0.10	0.00	8.61
Average			34.91						1.83	0.17	588.89

Table 3.8: Results produced by CPLS on 45 instances of the PPP (LB = Lower Bound, IMP = Improvement of Pruned Solution over Initial Solution, DEV = Deviation of Pruned Solution from Lower Bound)

Instance	Vertices	Arcs	Cost Deviation (%)	LB (IP)	Solution			IMP (%)	DEV (%)	Time(s)
					Initial	Local Search	Pruned			
A3101	116	174	8.90	24869	24932	24884	24876	0.22	0.03	349.03
A3101	116	174	18.53	25434	25619	25464	25452	0.65	0.07	1026.14
A3101	116	174	24.67	25847	26341	25976	25938	1.53	0.35	1137.49
A3101	116	174	32.61	26150	26942	26492	26362	2.15	0.81	1117.51
A3101	116	174	40.77	26615	27717	26882	26776	3.40	0.60	961.59
A3101	116	174	45.21	26572	28030	27146	26996	3.69	1.60	1303.60
A3101	116	174	52.35	26744	28981	27399	27161	6.28	1.56	1767.63
A3101	116	174	56.91	26902	29502	28112	27675	6.19	2.87	1218.65
A3101	116	174	65.36	27083	30499	28170	27844	8.71	2.81	1115.12
A3101	116	174	63.81	26731	29589	27571	27301	7.73	2.13	1974.86
Average			40.91					4.06	1.28	1197.16

Table 3.9: Results produced by CPLS on 10 instances generated from A3101

Instance	Vertices	Arcs	Cost Deviation (%)	LB (IP)	Solution			IMP (%)	DEV (%)	Time(s)
					Initial	Local Search	Pruned			
M3101	196	316	9.08	46982	47081	46985	46983	0.21	0.00	4999.10
M3101	196	316	17.93	47688	47943	47759	47719	0.47	0.07	6158.20
M3101	196	316	24.32	48536	49338	48764	48632	1.43	0.20	10043.04
M3101	196	316	32.96	49121	50766	49648	49372	2.75	0.51	14560.55
M3101	196	316	40.58	49450	51612	50217	49883	3.35	0.88	40801.83
M3101	196	316	46.29	49724	52529	50518	50108	4.61	0.77	29689.19
M3101	196	316	50.98	49695	53110	50962	50483	4.95	1.59	21754.87
M3101	196	316	58.42	49789	53323	50928	50554	5.19	1.54	36036.32
M3101	196	316	62.16	50199	55065	52126	51663	6.18	2.92	48468.64
M3101	196	316	64.65	50152	55035	52103	51501	6.42	2.69	34369.44
Average			40.74					3.56	1.21	24688.12

Table 3.10: Results produced by CPLS on 10 instances generated from M3101

3.6 Conclusions and Future Work

We presented a variant of the Windy Postman Problem that addresses a practical consideration in winter street maintenance – traversing a street after it has been plowed is faster than plowing it when it is snow covered. This new consideration introduces the concept of precedence in a route, where the order of edges traversed in the route matters. We proposed an algorithm, CPLS, that constructs an initial solution and improves it iteratively using a local search and reinitialization procedure. CPLS generated very good results that were within 0.17% of the lower bound, on average, for 45 instances. We observed that the deviation from the lower bound increased linearly with respect to the average deviation in cost between plowing uphill and plowing downhill. Based on this observation, we generated 20 difficult instances of the PPP and solved them using the CPLS algorithm. For these 20 instances, the results were still within 1.3% of the lower bound, on average.

It is easy to adapt CPLS to handle generalizations of the PPP, such as the case where each street needs to be plowed only once (or multiple times). We briefly considered the case where the objective function is modified to incorporate penalties for left-hand turns, U-turns, and crossing straight at intersections, since each takes a longer amount of time and causes the plow to push snow into the intersection. We modified CPLS to favor right-hand turns (which we call CPLS-RHT) by incorporating turn penalties into the local search fitness function. When comparing CPLS-RHT and CPLS on a rectilinear test instance of 100 nodes with turn penalties, we found that CPLS-RHT generated tours that were 5.6% better than CPLS. This shows that CPLS can be easily and effectively modified to handle turn penalties.

In future work, we could consider the more general case where the possible traver-

sal times of a street are a function of the number of times that the street is traversed. Furthermore, it would be interesting to determine how tight the lower bound is on large instances with a large average deviation in cost. If the bound is tight, improvements could be made in accuracy and running time. Finally, the case with multiple snowplows could be considered, where one snowplow would have the choice of deadheading on a street that another snowplow has already plowed.

Chapter 4

The Min-Max Downhill Plow Problem with Multiple Plows

4.1 Introduction

In winter, a common problem is to determine the route that a snow plow should take in order to minimize the distance traveled. Typically, this is represented as an arc routing problem, where we seek to find the route of minimum length that traverses the desired arcs of a graph. In the Chinese Postman Problem (CPP) [41], we seek a route that visits each arc of a graph at least once while minimizing deadhead. The CPP, which has symmetric costs for each arc, can be extended to the Windy Postman Problem (WPP) [46], where the costs are not symmetric. We propose a variant, called the Downhill Plow Problem (DPP), that is motivated by the fact that, on steep streets, it is much more difficult, or impossible, to plow uphill. Because deadhead travel over streets is significantly faster than the time it takes to plow the street, we want to design routes that try to avoid plowing uphill on steep streets. We prefer to plow downhill and deadhead uphill. It is important to note that this assumption requires the possibility of plowing against the direction of traffic. If the direction of traffic is enforced, it is impossible to avoid plowing uphill. We assume that the snow conditions warrant closing the streets.

The DPP ignores the concept of precedence, presented in the Plowing with Precedence Problem (PPP) in Dussault et al. [23], which imposes the restriction that if a street is completely unplowed, a plow must clear the street before deadheading it. This constraint was motivated by the assumption that, if the snow is too deep, it is impossible to deadhead an unplowed street. In the DPP, the snow is shallow enough so that the

plow may have the option of deadheading any street, whether it has been plowed or not. Dussault et al. [23] formulate the DPP as an integer program based on Win's [56] model.

In this paper, we present the Min-Max DPP with k (where $k > 1$) plows (MM k-DPP). Consider an undirected graph $G = \{V, A\}$ where the set of vertices $V = \{v_i\}$ and the set of edges $E \subseteq \{(v_i, v_j) | v_i, v_j \in V, i < j\}$. Each edge (v_i, v_j) has four costs: c_{ij}^+, the cost of plowing from v_i to v_j, c_{ji}^+, the cost of plowing from v_j to v_i, c_{ij}^-, the cost of deadheading from v_i to v_j, and c_{ji}^-, the cost of deadheading from v_j to v_i. Assuming that vertex v_j has a higher altitude than vertex v_i, we have $c_{ij}^+ \gg c_{ji}^+ \gg c_{ij}^- \geq c_{ji}^-$. We seek k routes that each begin and end at the depot (v_0), plow each edge twice (for each side of the street), and minimize the maximum route length.

The rest of this paper is organized as follows. In Section 2, we provide a brief literature review. In Section 3, we describe our solution methodology for solving the MM k-DPP. In Section 4, we discuss our results. In Section 5, we offer concluding remarks and directions for future research.

4.2 Literature Review

Arc routing problems have been studied extensively in the literature. Assad and Golden [6] and Eiselt et al. [26, 27] provide a review of arc routing problems. A book on arc routing was edited by Dror [22].

In the last decade, research efforts have focused on the Capacitated Arc Routing Problem (CARP) that was first proposed by Golden and Wong [33]. In addition to costs, each arc has an associated demand. Multiple routes must be constructed so that the total demand served by each route does not exceed the truck capacity. A comprehensive summary of the CARP over the last decade is given by Wøhlk [57].

Winter street maintenance covers a wide variety of problems from routing snow plows to locating snow disposal sites. A comprehensive review of the literature on winter street maintenance is given by Perrier et al. [48, 49, 50, 51]. The four articles cover optimization algorithms for the design of winter road maintenance systems for plowing and gritting, system design problems for snow disposal, optimization algorithms for routing vehicles in gritting operations and vehicle depot location, and optimization algorithms for routing vehicles in plowing and snow disposal operations.

We provide a brief review of the arc routing literature that specifically deals with snow plowing with a fixed number of plows. Marks and Stricker [44] and The Bureau of Management Consulting, Transport Canada [15] consider the following problem. Given a fleet of homogeneous plows, construct routes that minimize the total deadhead, such that each street is plowed at least two or four times depending on the width of the street. The authors provide strategies based on partitioning the graph and solving the CPP on each subgraph. They also incorporate several strategies to handle street priorities. The authors apply their approach to data from the city of Cambridge, Massachusetts. The Bureau of Management Consulting tested their heuristic on data from a major Canadian city.

In 1990, there was a mathematical modeling competition that posed the following problem. Given an Eulerian, strongly connected, directed graph G, construct two plowing routes that begin and end at the same point, cover G, and minimize the maximum route length. The graph represented the roads of a district in Wicomico County, Maryland and had 139 vertices and 374 arcs. Atkins et al. [7] and Robinson et al. [52] develop a cluster-first, route-second heuristic. Clustering is done by building two subgraphs iteratively by adding streets from G to the subgraph that has the shortest cumulative length. Robinson et al. [52] also give a route-first, cluster-second method that constructs an Eulerian cycle

and attempts to split it into two routes. Chernak et al. [18] extend the problem to include street priority. The authors construct primary routes that service roads of highest priority and extend the primary routes to include all roads.

Laporte et al. [42] describe a snow plowing problem where multiple plows must plow a city with the restriction that certain streets, which are very wide, must be cleared by several plows at the same time. Therefore, routes must be synchronized so that enough plows appear at the same intersection at the same time to plow a wide street together. In this problem, streets can be either plowed or deadheaded with different costs, as in the MM k-DPP.

A special case of the MM k-DPP is the Min-Max k-Chinese Postman Problem (MM k-CPP) which does not incorporate different costs for servicing and deadheading. The MM k-CPP was introduced by Frederickson et al. [29]. They show that the problem is NP-hard and provide a $(2 - 1/k)$-approximation algorithm. Lower bounds and heuristics for the MM k-CPP are presented by Ahr and Reinelt [2]. They develop a tabu search algorithm [3]. Ahr [1] gives an exact solution method based on a branch-and-cut approach.

Benavent et al. [9, 10, 11, 12] extend the MM k-CPP to the Min-Max k-Plow Windy Rural Postman Problem (MM k-WRPP). This problem incorporates asymmetric traversal costs and only requires a subset of arcs to be traversed.

Benavent et al. [9, 11] formulate the MM k-WRPP as an integer program and introduce valid inequalities. They solve this problem using a branch-and-price algorithm [12] with an upper bound given by a heuristic [11]. The branch-and-price outperforms the algorithms developed by Benavent et al. [9, 11] for instances with 5 or 6 plows.

The MM k-CPP and MM k-WRPP are both similar to the MM k-DPP, in that each share a min-max objective function for multiple plows. However, the MM k-CPP

and MM k-WRPP do not incorporate different costs for traversing and servicing an arc. In this generalization, a plow needs to decide whether to service or deadhead each arc that it traverses.

4.3 Solution Methodology

We solve the MM k-DPP using the cycle permutation local search procedure (CPLS) from Dussault et al. [23] that we extend to handle multiple plows (CPLS-M). CPLS-M is described in Section 4.3.3.1. We represent a solution to the MM k-DPP as a single grand cycle that is split into k separate routes. First, we present the splitting procedure to convert a grand cycle into a set of separate routes. Then, we give a solution framework for the MM k-DPP. The framework provides an initial solution, a lower bound to our problem, and solution attributes that are used in CPLS-M. Next, we describe the basic local search procedure, the reinitialization procedure, and the different fitness functions for a grand tour that are used for the MM k-DPP. Finally, we summarize CPLS-M, where we incorporate a two-stage local search procedure with different size neighborhoods and fitness definitions.

4.3.1 Grand Cycle Splitting

We represent a solution to the MM k-DPP as a single grand cycle that we modify to obtain a tour for each plow. A grand cycle is a sequence of vertices that begins and ends at the depot (vertex 0) and traverses each edge at least twice. We split this grand cycle into smaller sub-cycles, that begin and end at the depot, and divide the sub-cycles among the k plows in a way that tries to balance the route costs. Consider an instance with vertices 0 through 7, plowing and deadhead costs of 1, and two plows. A possible

grand cycle is

$$\{0, 2, 1, 0, 3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0,$$

$$2, 3, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0\}.$$

This grand cycle is split into sub-cycles that begin and end at the depot. The result is the following set of sub-cycles.

$$\{0, 2, 1, 0\}$$

$$\{0, 3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}$$

$$\{0, 2, 3, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0\}$$

We define the current average cost of a plow to be the sum of the averages of the plowing cost and deadhead cost for each edge currently assigned to the plow. We add sub-cycles to the plow with the shortest current average cost. Therefore, the first step assigns the first sub-cycle to Plow 1.

$$\text{Plow 1: } \{0, 2, 1, 0\}$$

Plow 1 has a current average cost of 3 and Plow 2 has a current average cost of 0. Next, the second sub-cycle $\{0, 3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}$ is assigned to Plow 2.

$$\text{Plow 1: } \{0, 2, 1, 0\}$$

$$\text{Plow 2: } \{0, 3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}$$

The current average cost of Plow 1 is less than the current average cost of Plow 2, so the final cycle $\{0, 2, 3, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0\}$ is appended to Plow 1's current tour.

Plow 1: $\{0, 2, 1, 0, 2, 3, 2, 7, 2, 7, 6, 5, 3, 2, 1, 0\}$

Plow 2: $\{0, 3, 5, 3, 2, 3, 2, 7, 6, 3, 6, 5, 4, 3, 6, 3, 5, 4, 3, 0\}$

We summarize the splitting procedure in Table 4.1. We note that it is possible to have a plow with no sub-cycles if the number of sub-cycles is less than the number of plows (k). The rest of our procedure guarantees that there will be at least one sub-cycle for each plow by requiring it in the solution framework. A feasible grand cycle has at least k plows.

Let:
S = be the grand cycle
$\{S[i]\}_{0 \leq i \leq T}$ = be the set of T sub-cycles of grand cycle S
$\{P[j]\}_{0 \leq j \leq M}$ = be the set of k plows

for each sub-cycle $(0 \leq i \leq T)$ {
    Define $P[j']$ to be a plow with the lowest current average cost
        (choose the lowest plow lexicographically if there is a tie)
    Append $S[i]$ to $P[j']$
}

Table 4.1: Procedure for dividing the grand cycle among the k plows

4.3.2 Solution Framework, Initial Solution, and Lower Bound

The solution framework is an integer program (denoted by IPX and presented in Dussault et al [23]) that solves the DPP for a single plow and is adapted from the formulation of the WPP given by Win [56]. Let

x_{ij} be the number of times edge (v_i, v_j) is plowed

y_{ij} be the number of times edge (v_i, v_j) is traversed after plowing

c_{ij}^{+} be the cost of plowing uphill

c_{ji}^{+} be the cost of plowing downhill

c_{ij}^{-} be the cost of deadheading uphill

c_{ji}^{-} be the cost of deadheading downhill

$\delta(i)$ be the set of edges incident to vertex v_i

D_1 be the set of vertices with one degree of separation from the depot

D_2 be the set of vertices with two degrees of separation from the depot that are not in D_1.

IPX is given by

$$\text{Minimize } \sum_{i,j} \left(c_{ij}^{+} x_{ij} + c_{ij}^{-} y_{ij} \right) \tag{4.1}$$

subject to

$$x_{ij} + x_{ji} = 2 \qquad \forall\ (v_i, v_j) \in A \tag{4.2}$$

$$\sum_{(v_i,v_j)\in\delta(i)} ((x_{ij} + y_{ij}) - (x_{ji} + y_{ji})) = 0 \qquad \forall\ v_i \in V \tag{4.3}$$

$$\sum_{(v_0,v_j)\in\delta(0)} (x_{0j} + y_{0j}) \geq M \tag{4.4}$$

$$\sum_{\substack{(v_i,v_j)\in\delta(i) \\ v_j \neq v_0}} (x_{ij} + y_{ij}) \geq x_{0i} + y_{0i} \qquad \forall\ v_i \in D_1 \tag{4.5}$$

$$\sum_{v_i\in D_1} \sum_{\substack{(v_i,v_j)\in\delta(i) \\ v_j\in D_2}} (x_{ij} + y_{ij}) \geq \sum_{v_i\in D_1} x_{0i} + y_{0i} \tag{4.6}$$

$$x_{ij}, y_{ij} \geq 0 \text{ and integer.} \tag{4.7}$$

The objective function (5.3) minimizes the cost of a single route. Constraint (5.4) requires each street to be plowed twice (once for each side). Constraint (5.5) requires that the in-degree and the out-degree of each vertex be balanced. Constraint (4.4) requires that any grand cycle leave the depot at least k times, ensuring that there are enough sub-cycles to allocate among the k plows. Constraints (4.5) and (4.6) are optional. Constraint (4.5) requires that, for each vertex i incident to the depot, the grand cycle leaves i without returning to the depot at least as often as it travels from the depot to i. Constraint (4.6) requires that the number of times the grand cycle traverses from vertices in D_1 to D_2 is at least the number of times it traverses from the depot to vertices in D_1. When constraints (4.5) and (4.6) are included in the model, IPX no longer gives a lower bound. However, these constraints discourage solutions where a grand cycle leaves the depot, visits vertices that are only one degree away from the depot, and then returns to the depot. When this grand cycle is split, it results in a short sub-cycle being assigned to a plow, which is often undesirable. Constraint (5.6) enforces non-negativity and integrality of the variables.

Solving IPX gives a number of times to plow and deadhead each edge in each direction and the associated directed Eulerian graph. By adding the number of times to plow and number of times to deadhead, we can determine the number of times each edge must be traversed. We call this a solution framework and use it to construct an initial grand cycle using an alternative to Fleury's algorithm (described in Dussault et al. [23]) that iteratively removes cycles from the graph G and combines them.

The optimal value of (5.3) without (4.5) gives the exact solution to the DPP. Therefore, the objective value divided by k (the number of plows) gives a lower bound for the MM k-DPP. We use the lower bound to compare the solutions generated by CPLS-M.

Adding the optional constraint (4.5) will yield a grand cycle that may be suboptimal for the DPP. If this grand cycle could be split into k equal length routes, it would be near the optimal solution to the MM k-DPP. If constraint (4.5) is not tight, then an equally split grand cycle would attain the lower bound and hence be optimal. In CPLS-M, we assume this splitting into equal, or near equal, length routes is possible.

4.3.3 Local Search

In the MM k-DPP, we try to reduce route length and balance the routes. CPLS-M permutes the sub-cycles in a grand cycle. This does not change the cumulative route length of all plows. Instead, CPLS-M seeks only to balance the individual routes. Our local search heuristic is given in Table 4.2. In order to search more of the solution space, after the local search procedure has finished, we reinitialize the solution and run the local search procedure again. The reinitialization procedure is similar to the local search procedure, except that it chooses a new solution in the neighborhood randomly. This process is repeated several times to generate a new grand cycle.

Begin with the solution s given by initial grand cycle construction
Let
$\tau(x)$ be the fitness of solution x
$N(x)$ be the neighborhood of x

Repeat:
    For each solution $s' \in N(s)$
        Calculate $\tau(s')$
        Define s_{min} to be a solution where $\tau(s_{min}) \leq \tau(s') \; \forall \; s' \in N(s)$
    If $\tau(s_{min}) < \tau(s)$ set $s = s_{min}$
    Else break

Table 4.2: Local search heuristic

4.3.3.1 Neighborhood Definition

We improve the grand cycle generated from the solution framework with a local search procedure. The neighborhood of a grand cycle s, $N(s)$, is the set of all grand cycles that can be obtained by a combination of the following two moves:

1. Cycles in the grand cycle are permuted

2. Cycles in the grand cycle are reversed.

To describe these two moves, we consider the following grand cycle (4.8) where the v's have been suppressed.

$$\{0, \underbrace{1,3}_{1}, 0, \underbrace{3,1}_{2}, 0, \underbrace{1,2,3}_{3}, 0, \underbrace{1,3,2,1}_{4}, 0\} \tag{4.8}$$

The bracketed elements in (4.8) are cycles that begin and end at vertex 0. Because they begin and end at the same vertex, these cycles can be permuted as shown in (4.9).

$$\{0, \underbrace{3,1}_{4}, 0, \underbrace{1,2,3}_{1}, 0, \underbrace{1,3,2,1}_{2}, 0, \underbrace{1,3}_{3}, 0\} \tag{4.9}$$

This new solution visits each edge the same number of times, just in a different order. This permutation can be done with cycles that begin and end at any vertex. For example, the cycles of (4.8) that begin and end at vertex 1 are shown in (4.10).

$$\{0, 1, \underbrace{3, 0, 3}_{1}, 1, \underbrace{0}_{2}, 1, \underbrace{2, 3, 0}_{3}, 1, \underbrace{3, 2}_{4}, 1, 0\} \tag{4.10}$$

Beginning with cycle (4.9), we can reverse cycle 2 and obtain the grand cycle in (4.11).

$$\{0, \underbrace{3, 1}_{4}, 0, \underbrace{1, 2, 3}_{1}, 0, \underbrace{1, 2, 3, 1}_{2}, 0, \underbrace{1, 3}_{3}, 0\} \tag{4.11}$$

We note that the size of a neighborhood can be large. Suppose the number of cycles for a given vertex is k. Then the number of possible permutations is $k!$. Moreover, we investigate cycles around all vertices. Therefore, the size of the neighborhood of a solution can become very large if there are a large number of cycles or vertices. In order to reduce the running time of our algorithm, we restrict the neighborhood definition to those cycles generated by the first k random permutations, which allows for linear growth of the neighborhood size. We avoid progressing through the set of permutations lexicographically by generating a new permutation from a previous one by swapping two random cycles. We further restrict our neighborhood to only investigate cycles centered around a particular vertex with some probability p. Experiments conducted on the PPP showed that this significantly reduced running times without sacrificing solution quality (see Dussault et al. [23]).

4.3.3.2 Grand Cycle Fitness

CPLS-M operates the same way for the MM k-DPP and the MPP, only differing in the fitness function. After the grand cycle has been split, each plow has a route assigned to it. We now describe the fitness of the assigned routes for the MM k-DPP. The cost of a route r, ρ_r $(0 \leq r < k)$, is given by

$$\tau(\rho_r) = \sum_{i,j} (c_{ij}^{+} \hat{x}_{ijr} + c_{ij}^{-} \hat{y}_{ijr}) \tag{4.12}$$

where $\hat{x}_{ijr}$ is the number of times route r plows edge (i, j) and $\hat{y}_{ijr}$ is the number of times route r deadheads edge (i, j).

We can determine the optimal values of $\hat{x}_{ijk}$ and $\hat{y}_{ijk}$ by solving the following min-max integer program. Define n_{ijk} to be the number of times route k traverses edge (i, j), which is specified by the solution framework.

$$\text{Minimize } \max_r \{\tau(\rho_r)\} \tag{4.13}$$

subject to

$$\hat{x}_{ijr} + \hat{y}_{ijr} = n_{ijr} \qquad \forall\ i, j, r \tag{4.14}$$

$$\sum_r (\hat{x}_{ijr} + \hat{x}_{jir}) = 2 \qquad \forall\ i, j \tag{4.15}$$

$$\hat{x}_{ijr}, \hat{y}_{ijr} \geq 0 \text{ and integer} \qquad \forall\ i, j, r \tag{4.16}$$

This model can be reformulated as the following equivalent integer program (denoted by IP) using the method described in [34].

$$\text{Maximize } w \tag{4.17}$$

subject to

$$w \leq \sum_{i,j} c_{ij}^{-} n_{ijr} - \sum_{i,j} (c_{ij}^{-} - c_{ij}^{+}) \hat{x}_{ijr} \qquad \forall\ r \tag{4.18}$$

$$\sum_{r} \hat{x}_{ijr} + \hat{x}_{jir} = 2 \qquad \forall\ i, j \tag{4.19}$$

$$w, \hat{x}_{ijr} \geq 0 \text{ and integer} \qquad \forall\ i, j, r \tag{4.20}$$

The objective function (4.13) minimizes the maximum route length. Constraint (4.14) requires that, for each edge, the number of times an edge is plowed plus the number of times an edge is deadheaded equals the total number of traversals. Constraint (4.15) requires that each edge is plowed exactly twice.

In the reformulation, references to the variables $\hat{y}_{ijk}$ are removed in constraint (4.18) by using constraint (4.15). Constraint (4.18) restricts w to be less or equal than $\tau(r_k)$ for each k, giving the same convex hull imposed by the non-linear objective (4.13). Constraint (4.19) is the same as constraint (4.15). Splitting any grand cycle as described in the previous subsection, followed by solving IP and using its solution in (4.13), gives us the fitness for any feasible grand cycle. The optimal fitness of solution s is denoted by $\omega(s)$.

The running time for IP for large problems is prohibitively long for use in a fitness function, which can be called hundreds of thousands of times. We note that we have already determined an optimal number of times each edge (i, j) is plowed in the solution framework (given by x_{ij}). It only remains to optimally distribute the number of times an edge (i, j) is plowed between the plows that traverse edge (i, j). Instead of obtaining the

optimal distribution from IP, we estimate the cost of an optimal assignment by randomly assigning $\hat{x}_{ijk}$ so that $\sum_k \hat{x}_{ijk} = x_{ij}$. By comparing to the optimal solution given by IP on a variety of instances, this random assignment policy was empirically found to be a good approximation, as $x_{ij} \leq 2$ for all edges (i, j), and there is often no ambiguity on how to assign plowing duties for most edges because a plow usually only traverses a particular edge (i, j) once. After plowing duties are randomly assigned, the cost of each plow (and hence the largest cost) is obtained. This approximate fitness is denoted by ω_a. The optimal fitness ω is used to determine the fitness of the final solution found at the end of the local search procedure.

Instead of minimizing the length of the longest route, we minimized the difference between the longest route and the shortest route. This ultimately has the same effect, since the cumulative length of the streets that are plowed is fixed. We did this because we found our algorithm was most often presented with situations where some routes were significantly shorter or longer than the rest. Had we focused on minimizing the length of the longest route, we would have ignored solutions that increased the length of the shortest route without decreasing the length of the longest route or vice versa. Exploring the neighborhood of these solutions is beneficial in the long term.

4.3.4 Summary of CPLS-M

CPLS-M has two local search procedures. Often, when constructing the initial grand cycle, extremely short sub-cycles are generated. Consider

$$\{0, 2, 1, 0, 1, \cdots, 6, 2, 3, \cdots, 5, 2, 1, 0\}.$$

This tour has a very short sub-cycle $\{0, 2, 1, 0\}$. When split, this will result in one plow

having a route of $\{0, 2, 1, 0\}$, which is undesirable because the length of $\{0, 2, 1, 0\}$ is likely to be short and could possibly result in a plow only having $\{0, 2, 1, 0\}$ as its entire tour. Using cycle permutation around vertex 2, a more favorable split can be obtained by swapping sub-cycles 1 and 2 as seen below from the transition from (4.21) to (4.22).

$$\{0, 2, \underbrace{1, 0, 1, \cdots, 6}_{1}, 2, \underbrace{3, \cdots, 5}_{2}, 2, 1, 0\} \tag{4.21}$$

$$\{0, 2, \underbrace{3, \cdots, 5}_{2}, 2, \underbrace{1, 0, 1, \cdots 6}_{1}, 2, 1, 0\} \tag{4.22}$$

A similar cycle permutation around vertex 1 does not yield a similar favorable split. We found that it is very difficult to break out of these situations with small sub-cycles without investigating the largest neighborhood possible. However, this is not desirable, since calculating the fitness function for the MM k-DPP for a large neighborhood is computationally intensive.

CPLS-M begins with what we call a thorough local search procedure that investigates every cycle permutation around every vertex. Instead of using the standard MM k-DPP fitness, we maximized a new fitness function:

$$\tilde{\tau}(\text{shortest plow}) \times \tilde{\tau}(\text{second shortest plow}), \tag{4.23}$$

where $\tilde{\tau}$ gives the number of edges traversed (plowed or deadheaded) by the plow. The fitness function (4.23) attempts to maximize the two shortest routes simultaneously. This is done to break out of initial grand cycles that have very short plow tours. This thorough local search terminates immediately after the shortest route traverses at least five different

streets. We used the number of edges traversed ($\tilde{\tau}$) instead of the cost of a tour (τ) for running time considerations and because removing short sub-cycles resulted in larger and more flexible neighborhoods in the second local search procedure.

Then CPLS-M performs the standard local search using the fitnesses of the MM k-DPP as given above. The neighborhood is restricted so that:

1. Cycle permutations around a vertex are investigated 20% of the time.

2. If the number of sub-cycles around a given vertex is b, then the first b random permutations are investigated

A summary of CPSL-M is given in Table 4.3.

Begin with the solution s given by initial grand cycle construction
Let $\tilde{\phi}(x)$ be the output of the thorough local search procedure with initial grand cycle x
Let $\phi(x)$ be the output of the standard local search procedure with initial grand cycle x
Define s_{best} to be the best solution obtained
Repeat:
$s = \tilde{\phi}(s)$ Do thorough local search with large neighborhood and fitness ω_a
$s = \phi(s)$ Do standard local search with small neighborhood and fitness ω
If $\tau(s) < \tau(s_{best})$ set $s_{best} = s$
Reinitialize s
Until 10 iterations or s_{best} is within 0.1% of optimal

Table 4.3: Summary of CPLS-M

4.4 Computational Results

CPLS-M was implemented in C++ using CPLEX12.2 on a single thread of a 1.7 GHz Intel Core i5 Dual Core Processor. To test CPLS-M, we used instances presented in Dussault et al. [23] for the PPP with 2, 3, 4, and 5 plows. Our results are summarized in Table 4.4 and the details are given in Tables 4.5, 4.6, 4.7, and 4.8 for 2, 3, 4, and 5 plows, respectively. There are 20 different graphs for a total of $20 \times 4 = 80$ different instances.

We see that solution quality is quite good with respect to the lower bound, with an average deviation of 0.09%, 0.49%, 0.74%, and 1.92% for 2, 3, 4, and 5 plows, respectively. Furthermore, we obtained the optimal solution to at least 15 of the 80 instances by obtaining the lower bound (other solutions that are not equal to the lower bound may, in fact, be optimal).

In Table 4.4, we see that the CPLS-M's worst-case performance is quite good, never exceeding 5.5% deviation from the lower bound over all 80 instances. Additionally, there is general increase in deviation from the lower bound as the number of plows increases. We hypothesize that the increase in deviation from the lower bound when there are a larger number of plows occurs because it is more difficult to split an optimal grand cycle into a large number of evenly sized sub-cycles than a small number of evenly sized sub-cycles.

	Maximum Deviation from Lower Bound			
Node Size	2 Plows	3 Plows	4 Plows	5 Plows
60-69	0.48	2.00	2.83	5.49
70-87	0.03	1.08	2.41	2.68
88-94	0.34	1.56	0.56	3.45
95-99	0.09	0.07	0.71	2.10
100-196	0.07	1.36	1.61	4.67

Table 4.4: Maximum deviation from lower bound (in percent) for different size instances and number of plows

4.5 Conclusions

In this paper, we introduced a variant of the Windy Postman Problem called the Downhill Postman Problem, and generalized it to incorporate multiple plows. We proposed an algorithm (CPLS-M) that constructs an initial solution and improves it using a local search procedure followed by reinitialization. CPLS-M does not incorporate both

cumulative route length and route balancing in its objective function. Instead, it fixes cumulative route length and focuses on route balancing. This produces very high-quality solutions.

We showed that the lower bound, obtained by assuming an optimal tour for a single plow can be split equally among multiple plows, was actually attained in 15 of 80 cases. Cumulative route length and balanced routes are competing objectives. The apparent tightness of the lower bound in the MM k-DPP shows that one can optimize for cumulative route length (by ignoring the min-max objective) and obtain optimal route balancing while maintaining the same cumulative route length.

CPLS-M can be adapted to handle changes to the objective function by including penalties when making U-turns and left-hand turns, or going straight across an intersection, which takes additional time and pushes snow out into the middle of the intersection.

In future work, we could incorporate the concept of precedence, introduced in Dussault et al. [23]. This adds a constraint that a plow must clear (and not deadhead) an unplowed street. The temporal order of a plow's tour becomes important and allows for the possibility that one plow clears downhill on a street, allowing another plow to deadhead uphill later. While precedence can be incorporated easily into the objective function of CPLS-M, we still would need to determine the optimal plowing assignments for a set of plows even after all routes have been generated.

File Name	Vertices	Edges	Lower Bound	Initial	Plow1	Plow 2	Maximum	DEV (%)	IM (%)	Time (s)
A3101	116	174	12352	24622	12360	12360	12360	0.06	49.80	584.56
hd115	70	141	5727	9344	5725	5728	5728	0.02	38.70	117.94
hd215	70	162	6765	12550	6767	6763	6767	0.03	46.08	169.42
hd315	68	153	6225	8309	6230	6219	6230	0.08	25.02	49.86
hd415	88	171	6056	10628	6052	6059	6059	0.05	42.99	201.7
hd515	86	178	5404	9463	5403	5405	5405	0.02	42.88	404.94
hd615	88	181	5006	5959	5003	5009	5009	0.06	15.94	131.18
hd715	100	188	4173	7478	4169	4176	4176	0.07	44.16	266.4
hd815	95	193	5393	6905	5392	5394	5394	0.02	21.88	75.23
hd915	96	189	4521	6713	4516	4525	4525	0.09	32.59	215.06
hg115	60	105	210	416	211	209	211	0.48	49.28	17.23
hg215	68	119	225	441	226	224	226	0.44	48.75	64.69
hg315	64	116	228	423	228	227	228	0.00	46.10	28.48
hg415	82	148	277	542	277	276	277	0.00	48.89	21.89
hg515	92	165	296	372	295	297	297	0.34	20.16	136.94
hg615	91	162	287	539	287	287	287	0.00	46.75	237.43
hg715	97	174	279	455	278	279	279	0.00	38.68	272.24
hg815	100	180	286	488	286	286	286	0.00	41.39	274.33
hg915	97	175	294	336	294	294	294	0.00	12.50	26.24
M3101	196	316	23284	28271	23289	23278	23289	0.02	17.62	1429.11
Average								0.09	36.51	236.24

Table 4.5: Results of CPLS-M on the MM k-DPP for 2 Plows (File Name = the instance name from Dussault et al. [23], Vertices = the number of vertices, Edges = the number of edges, Lower Bound = the lower bound for the MM k-DPP. Initial = initial solution fitness for the MM k-DPP, Plow i = length of route traversed by plow i, Maximum = maximum cost over all plows, DEV = percent deviation of the maximum from the lower bound, IM = the percentage improvement from the initial solution, Time = running time.)

File Name	Vertices	Edges	Lower Bound	Initial	Plow1	Plow 2	Plow 3	Maximum	DEV (%)	IM (%)	Time (s)
A3101	116	174	8247	24657	8359	8120	8317	8359	1.36	66.10	1357.8
hd115	70	141	3818	10787	3818	3817	3818	3818	0.00	64.61	389.97
hd215	70	162	4510	12771	4507	4511	4512	4512	0.04	64.67	866
hd315	68	153	4150	9722	4158	4141	4150	4158	0.19	57.23	943.17
hd415	88	171	4037	11285	4037	4032	4042	4042	0.12	64.18	737.81
hd515	86	178	3603	8542	3592	3592	3624	3624	0.58	57.57	1256.29
hd615	88	181	3338	9312	3337	3338	3337	3338	0.00	64.15	1286.06
hd715	100	188	2782	5766	2779	2783	2783	2783	0.04	51.73	1566.93
hd815	95	193	3596	6594	3596	3592	3598	3598	0.06	45.44	1054.49
hd915	96	189	3014	7957	3016	3010	3015	3016	0.07	62.10	378.27
hg115	60	105	141	414	138	143	141	143	1.42	65.46	112.2
hg215	68	119	150	444	153	145	152	153	2.00	65.54	202.83
hg315	64	116	153	356	154	153	150	154	0.65	56.74	86.8
hg415	82	148	185	549	179	187	187	187	1.08	65.94	467.96
hg515	92	165	198	583	197	198	198	198	0.00	66.04	608.96
hg615	91	162	192	563	195	195	186	195	1.56	65.36	487.77
hg715	97	174	186	553	186	185	186	186	0.00	66.37	97.16
hg815	100	180	192	570	193	192	189	193	0.52	66.14	416.48
hg915	97	175	197	585	196	196	197	197	0.00	66.32	195.71
M3101	196	316	15527	35757	15487	15535	15449	15535	0.05	56.55	6508.85
Average									0.49	61.91	951.07

Table 4.6: Results of CPLS-M on the MM k-DPP for 3 Plows

File Name	Vertices	Edges	Lower Bound	Initial	Plow1	Plow 2	Plow 3	Plow 4	Maximum	DEV (%)	IM (%)	Time (s)
A3101	116	174	6195	24657	6209	6134	6158	6295	6295	1.61	74.47	2729.76
hd115	70	141	2864	10243	2877	2910	2933	2733	2933	2.41	71.37	2122.18
hd215	70	162	3383	8800	3343	3396	3412	3379	3412	0.86	61.23	2079.17
hd315	68	153	3113	10392	3098	3114	3123	3114	3123	0.32	69.95	1432.59
hd415	88	171	3028	6006	3029	3011	3026	3045	3045	0.56	49.30	2504.49
hd515	86	178	2702	5055	2714	2710	2689	2695	2714	0.44	46.31	2906.44
hd615	88	181	2503	4710	2508	2501	2498	2505	2508	0.20	46.75	2268.69
hd715	100	188	2087	3288	2075	2095	2091	2084	2095	0.38	36.28	2020.1
hd815	95	193	2697	3915	2699	2694	2695	2698	2699	0.07	31.06	526.87
hd915	96	189	2261	6395	2264	2260	2255	2262	2264	0.13	64.60	3209.05
hg115	60	105	106	278	109	108	102	103	109	2.83	60.79	151.03
hg215	68	119	113	223	113	113	112	112	113	0.00	49.33	257.53
hg315	64	116	115	248	113	117	111	118	118	2.61	52.42	217.82
hg415	82	148	139	541	138	139	139	139	139	0.00	74.31	650.91
hg515	92	165	149	575	149	149	148	149	149	0.00	74.09	554.8
hg615	91	162	145	548	145	145	145	145	145	0.00	73.54	986.94
hg715	97	174	140	287	139	139	141	140	141	0.71	50.87	783.09
hg815	100	180	144	562	144	145	143	142	145	0.69	74.20	466.65
hg915	97	175	148	578	149	147	147	147	149	0.68	74.22	515.77
M3101	196	316	11649	32344	11644	11676	11664	11609	11676	0.23	63.90	10835.05
Average										0.74	59.95	1860.94

Table 4.7: Results of CPLS-M on the MM k-DPP for 4 Plows

File Name	Vertices	Edges	OBJ	Lower Bound	Initial	Plow 1	Plow 2	Plow 3	Plow 4	Plow 5	Maximum	DEV (%)	IM (%)	Time (s)
A3101	116	176	24806	4962	15517	4944	5010	4876	4956	5020	5020	1.17%	67.65%	1380.45
hd115	70	141	11455	2291	4872	2269	2267	2291	2325	2303	2325	1.48%	52.28%	4548.33
hd215	70	162	13530	2706	10992	2735	2713	2668	2713	2701	2735	1.07%	75.12%	3490.75
hd315	68	153	12450	2490	6884	2481	2486	2511	2472	2500	2511	0.84%	63.52%	3423.62
hd415	88	171	12115	2423	5978	2447	2398	2413	2429	2428	2447	0.99%	59.07%	4630.96
hd515	86	178	10809	2162	8077	2163	2138	2198	2141	2169	2198	1.67%	72.79%	3800.97
hd615	88	181	10012	2003	5490	2010	2002	1994	1994	2012	2012	0.45%	63.35%	3800.84
hd715	100	188	8352	1671	6529	1747	1614	1623	1749	1657	1749	4.67%	73.21%	6741.63
hd815	95	193	10791	2159	5324	2159	2148	2149	2166	2169	2169	0.46%	59.26%	3061.77
hd915	96	189	9042	1809	4424	1765	1838	1780	1812	1847	1847	2.10%	58.25%	2918.74
hg115	60	105	424	85	322	88	83	87	84	86	88	3.53%	72.67%	181.2
hg215	68	119	452	91	440	80	86	95	96	95	96	5.49%	78.18%	406.7
hg315	64	116	461	93	439	94	96	90	91	90	96	3.23%	78.13%	268.7
hg415	82	148	557	112	437	111	114	110	115	110	115	2.68%	73.68%	520.65
hg515	92	165	596	120	589	120	118	121	121	119	121	0.83%	79.46%	901.27
hg615	91	162	580	116	508	120	117	114	118	115	120	3.45%	76.38%	1415.29
hg715	97	174	559	112	549	113	113	111	113	111	113	0.89%	79.42%	502.66
hg815	100	180	576	116	540	117	116	115	114	114	117	0.86%	78.33%	6078.37
hg915	97	175	592	119	570	119	120	118	118	118	120	0.84%	78.95%	1069.82
M3101	196	316	46609	9322	39154	9472	9431	9311	9014	9391	9472	1.61%	75.81%	9716.58
												1.92%	70.78%	2942.97

Table 4.8: Results of CPLS-M on the MM k-DPP for 5 Plows

Chapter 5

The Extended Min-Max Downhill Plow Problem with Multiple Plows

5.1 Introduction

The Min-Max Downhill Plow Problem with Multiple Plows (MM k-DPP) can be extended to include rural instances (where not all edges are required to be traversed), turn penalties, and precedence. We show the hierarchy in Figure 5.1, which culminates in the Extended Min-Max Downhill Plow Problem with Multiple Plows (MM k-DPPE). We describe the addition of rural instances, turn penalties, and precedence individually, and demonstrate how the solution framework (IPX) and Cycle Permutation Local Search (CPLS) can be modified to accommodate it.

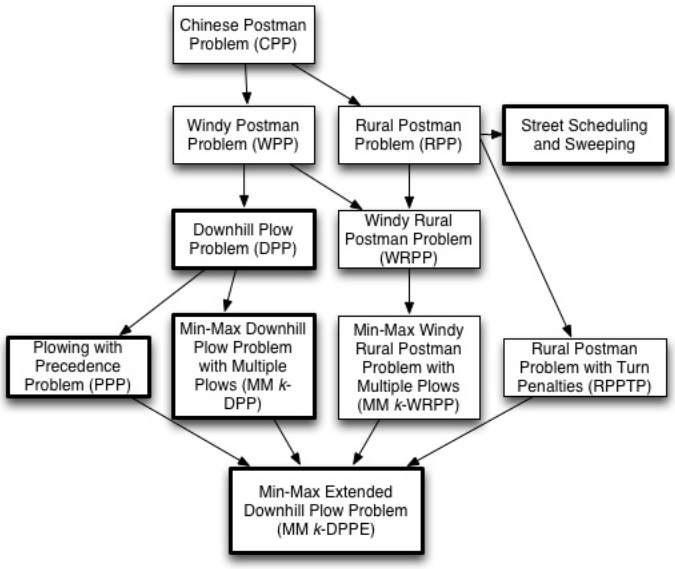


Figure 5.1: Hierarchy of edge routing problems. Arrows indicate increased generality or additional constraint

A recent paper by Carlsson and Ye [17] describes a similar problem. It consists of

extending the CPP with the following attributes.

1. Multiple vehicles (with a min-max objective);
2. Turning penalties;
3. A mixed graph (some edges may be directed, some may be undirected);
4. A desire to keep vehicle tours as geographically separate as possible.
5. Multiple depots, one for each vehicle

The MM $k-$DPPE incorporates items 1-3, but not 4 or 5. On the other hand, [17] does not incorporate different costs for servicing vs. deadheading, different costs for uphill or downhill, rural instances, or precedence. Carlsson and Ye [17] implement a metaheuristic on an actual (confidential) city with 235,570 intersections and 642,745 street segments. Their solution has 5% deviation from their lower bound in terms of cumulative route length, and a ratio between the longest and shortest route of 1.34. We cannot directly compare to the results of the MM k-DPP.

5.2 Turn Penalties and Rural Instances in IPX

5.2.1 Turn Penalties

Sometimes a certain type of turn at an intersection is preferred. A right-hand turn (or proceeding straight) is often preferable because it takes less time, is safer, and, in the case of plowing operations, does not push snow into the intersection. Left-hand turns and U-turns are often discouraged because they take more time and push snow into the intersection.

There are two ways to incorporate turn penalties. The first is to incorporate them into the objective function of CPLS-M. The neighborhood in CPLS is constructed by two moves: cycle permutation and cycle reversal.

Cycle permutation by itself will not change the objective with respect to turn penalties. When an Eulerian cycle (called the grand cycle) is decomposed into sub-cycles, the number of each type of turn is not changed when the sub-cycles are permuted. Cycle reversal changes left-hand turns to right-hand turns, and vice versa, which does change the turn penalties, but not the number of turns. Moreover, cycle permutation is unable to change the number of times a cycle goes straight through an intersection or makes a U-turn. Therefore, without modification to the definition of CPLS's neighborhood, CPLS will be ineffective at handling turn penalties.

One of the key attributes of CPLS and CPLS-M is that they both rely on the solution framework to handle a substantial portion of the optimization. In the PPP, the solution framework solved the problem optimally while ignoring precedence and used CPLS to handle precedence. In the MM k-DPP, the solution framework minimized the cumulative length of all plows, and CPLS-M only sought to optimize route balancing. This process has been extremely effective at producing near-optimal solutions. For the MM k-DPPE, it is reasonable to attempt to have the solution framework optimize for turn penalties, and then use CPLS to balance the routes and handle precedence.

5.2.2 Rural Postman Problem

The Rural Chinese Postman Problem (RPP) requires that a specified subset of edges $E_R \subset E$ is required for traversal. All other edges, E_R^c, are optional and may or may not be traversed. The RPP is well studied (for a review of the RPP literature, see Eiselt et

al. [27]), and a common solution methodology involves adding required edges to connect any disconnected subsets of E_R in an optimal way.

5.2.3 Modification of Solution Framework

The solution framework (IPX) of the MM k-DPP can be changed to handle both turn penalties and rural instances by modifying the formulation for the DPP. The integer programming (IP) formulation for the DPP relies on the observation that it suffices to convert the graph G into a Eulerian graph in an optimal way. From there, Fleury's algorithm can be applied to obtain an optimal solution. However, this formulation does not address turn penalties. Indeed, there are multiple solutions to the converted Eulerian graph that have different numbers of each type of turn. Consider the Eulerian graph in Figure 5.2. There are many cycles that obey this Eulerian graph, such as cycles (5.1) and (5.2).

$$\{0, 1, 2, 0, 2, 1, 0\} \tag{5.1}$$

$$\{0, 1, 0, 2, 1, 2, 0\} \tag{5.2}$$

Cycle (5.2) contains two U-turns at node 1 while cycle (5.1) has no U-turns at all. Therefore, solutions that are considered equivalent in the DPP are not equivalent when considering turn penalties.

We begin by stating the IP formulation for the DPP (IPX) and extend it to the IPXE for use in the solution framework of the MM k-DPPE to handle both turn penalties and rural instances. Let

x_{ij} be the number of times edge (v_i, v_j) is plowed

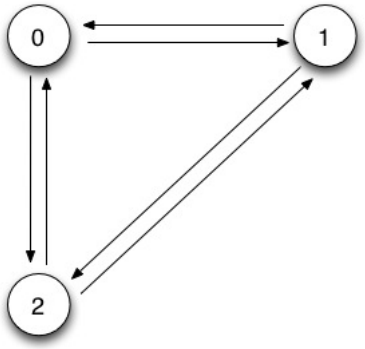


Figure 5.2: Eulerian Graph with three nodes

y_{ij} be the number of times edge (v_i, v_j) is traversed after plowing

c_{ij}^{+} be the cost of plowing uphill

c_{ji}^{+} be the cost of plowing downhill

c_{ij}^{-} be the cost of traversing uphill after plowing

c_{ji}^{-} be the cost of traversing downhill after plowing

$\delta(i)$ be the set of edges incident to vertex v_i.

The formulation for the DPP is given by

$$\text{Minimize } \sum_{i,j} \left(c_{ij}^{+} x_{ij} + c_{ij}^{-} y_{ij} \right) \tag{5.3}$$

subject to

$$x_{ij} + x_{ji} = 2 \qquad \forall\ (v_i, v_j) \in A \tag{5.4}$$

$$\sum_{(v_i,v_j)\in\delta(i)} ((x_{ij} + y_{ij}) - (x_{ji} + y_{ji})) = 0 \qquad \forall\ v_i \in V \tag{5.5}$$

$$x_{ij}, y_{ij} \geq 0 \text{ and integer.} \tag{5.6}$$

The objective function (5.3) minimizes the cost of the route. Constraint (5.4) requires each street to be plowed twice (once for each side). Constraint (5.5) requires that the degree of each vertex be balanced. Constraint (5.6) enforces non-negativity and integrality of the variables.

To formulate IPXE, we convert the graph G into a new graph G'. Instead of edges representing streets, the edges of G' represent turns. We illustrate this change on the graph in Figure 5.3(a). If the Eulerian cycle has the subtour $\{5, 3, 2\}$, then it performs a left-hand turn. We represent this as a new edge, shown in Figure 5.3(b). Similarly, the subtour $\{5, 3, 1\}$ is traveling straight through the intersection and is represented by a new edge, shown in Figure 5.3(c). Subtour $\{5, 3, 5\}$ performs a U-turn at the intersection and is represented by the edge shown in Figure 5.3(d). The entire converted graph G' is shown in Figure 5.4. Each edge in G', denoted by $a_{i,j,k}$, can be interpreted as traveling from the midpoint of edge (v_i, v_j) through vertex v_j to the midpoint of edge (v_j, v_k). However, for formulation purposes, it is convenient to interpret the edge $a_{i,j,k}$ as traveling from vertex v_i to vertex v_j and then turning in the direction of vertex v_k. We illustrate the cost conversions in Figure 5.5.

We now state the IP formulation for the DPP-TP. Let

N be the number of edges in G

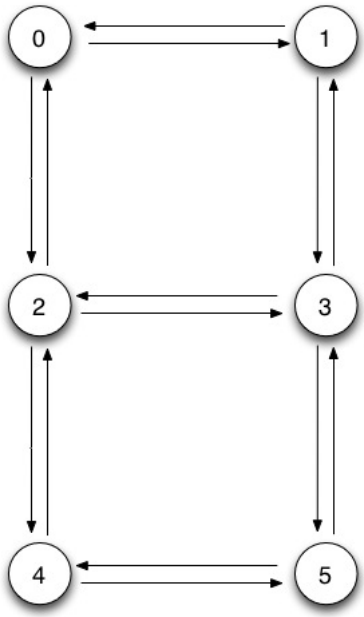
M be the number of plows

$m(i, j)$ be the midpoint of edge $(v_i, v_j) \in G$, which are the nodes of G'

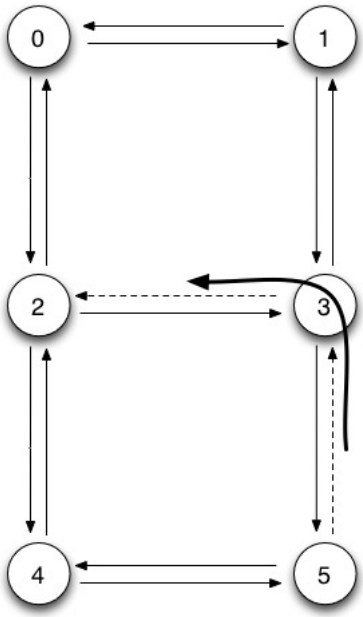
V' be the set of all $m(i, j)$. Note $|V'| = N$

B be a sufficiently large number

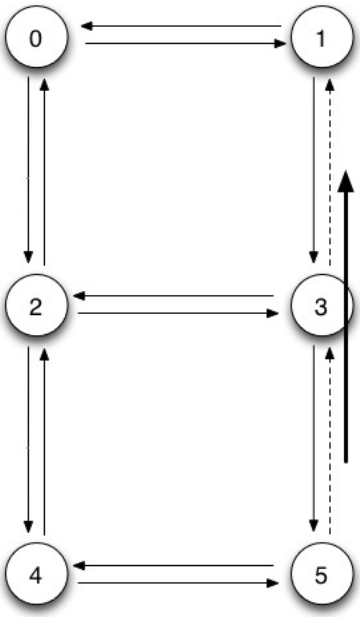
x_{ijk} be the number of times edge $a_{i,j,k}$ is plowed



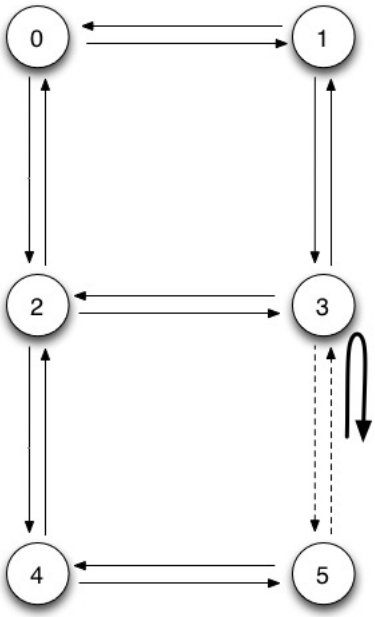
(a) A six-node graph



(b) The bold edge represents a left turn {5,3,2}

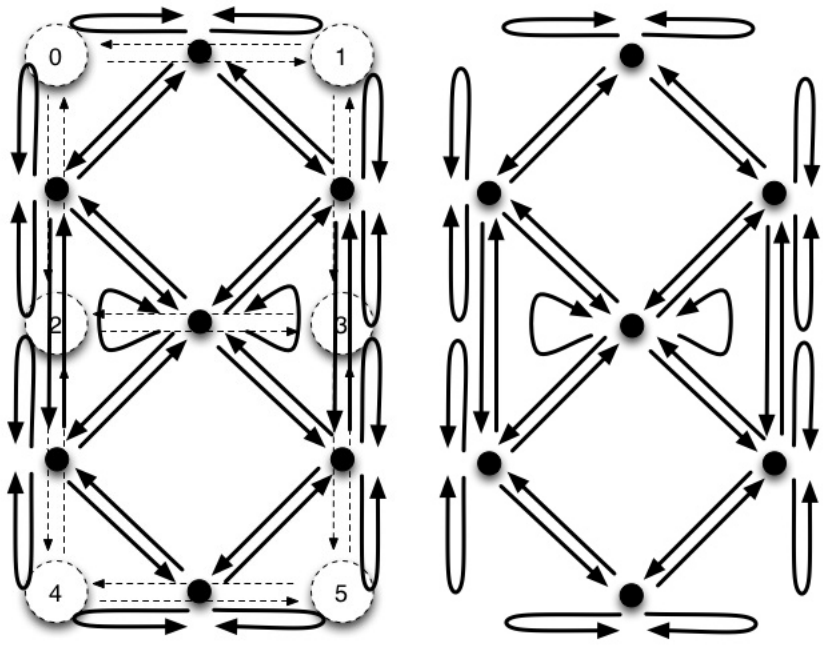


(c) The bold edge represents traveling straight {5,3,1}



(d) The bold edge represents a U-turn {5,3,5}

Figure 5.3: Conversions of different turns on a six-node graph



(a) Converted graph over the original instance

(b) Converted graph

Figure 5.4: Complete conversion of an six-node graph

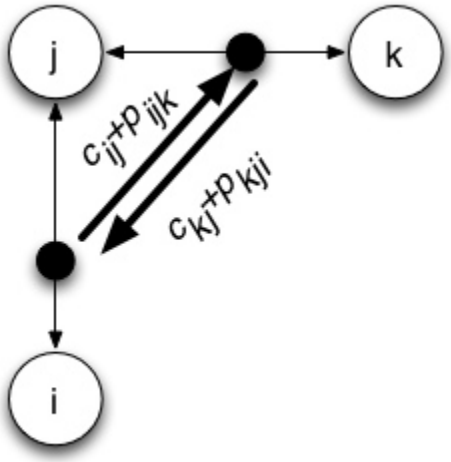


Figure 5.5: Turning costs (bolded arcs) consist of both the traversal (plowing or deadhead) costs for the first arc and the turn penalty

$\hat{x}_{ijk} = \begin{cases} 1, \text{ if edge } a_{i,j,k} \text{ is traversed} \\ 0, \text{ otherwise.} \end{cases}$

y_{ijk} be the number of times edge $a_{i,j,k}$ is deadheaded

p_{ijk} be the penalty of making the turn associated with traversing $a_{i,j,k}$

$c^+_{ijk} = c^+_{ij}$ be the cost of plowing $a_{i,j,k}$

$c^-_{ijk} = c^-_{ij}$ be the cost of deadheading $a_{i,j,k}$

α be a constant between 0 and 1 that serves as a weighting factor between distance and turn penalties (when $\alpha = 0$ only turn penalties count)

$u_{m(i,j)}$ be the order that node $m(i,j)$ is visited

The new formulation is as follows.

$$\text{Minimize } \sum_{i,j,k} \left[x_{ijk} \left(\alpha c^+_{ijk} + (1-\alpha) p_{ijk} \right) + y_{ijk} \left(\alpha c^-_{ijk} + (1-\alpha) p_{ijk} \right) \right] \tag{5.7}$$

subject to

$$\sum_{(v_0,v_j)\in\delta(0)} (x_{0jk} + y_{0jk}) \geq M \tag{5.8}$$

$$\sum_k (x_{ijk} + x_{kji}) \geq 2 \qquad \forall\ (i,j) \in A_R \tag{5.9}$$

$$\sum_k (x_{ijk} + y_{ijk}) = \sum_k (x_{kij} + y_{kji}) \qquad \forall\ i,j \tag{5.10}$$

$$x_{ijk} + y_{ijk} \leq B\hat{x}_{ijk} \qquad \forall\ i,j,k \tag{5.11}$$

$$x_{ijk} + y_{ijk} \geq \hat{x}_{ijk} \qquad \forall\ i,j,k \tag{5.12}$$

$$u_\beta = 0 \qquad \text{for some } \beta \in V' \tag{5.13}$$

$$1 \leq u_{m(i,j)} \leq N-1 \qquad \forall\ m(i,j) \neq \beta \tag{5.14}$$

$$(u_{m(i,j)} + 1) - (N-1)(1 - \hat{x}_{ijk}) \leq u_{m(j,k)} \qquad \forall\ m(i,j), m(j,k) \neq \beta \tag{5.15}$$

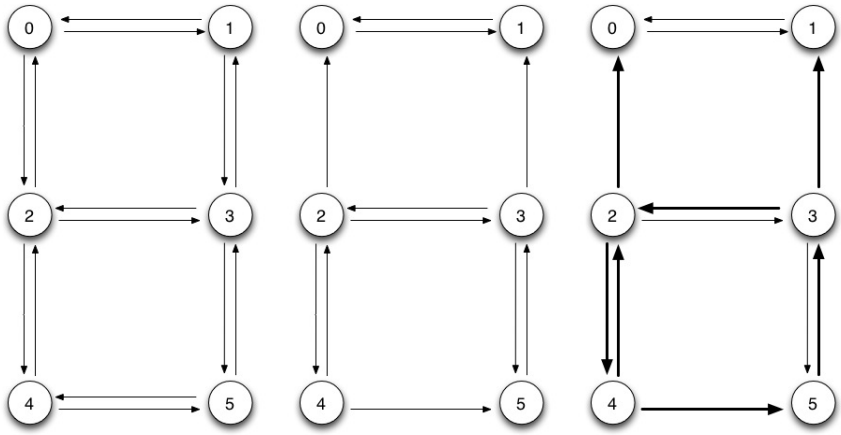
$$x_{ijk} \geq 0 \text{ and integer.} \tag{5.16}$$

The objective function (5.7) minimizes a linear combination of the plowing and deadhead costs and turn penalties. Adjustment of the parameter α can emphasize or deemphasize turn penalties. Constraint (5.8) requires that any grand cycle leave the depot at least M times, ensuring that there are enough sub-cycles to allocate among the M plows. Constraint (5.9) forces each required street to be serviced twice (once for each side). Constraint (5.10) requires that the degree of each vertex be balanced. That is, in the graph G', if you enter the first half of a street, you have to leave through the second half of the street. Constraints (5.11, 5.12) enforce the definition of $\hat{x}_{ijk}$. Constraints (5.13, 5.14, 5.15) are subtour elimination constraints that are required because not every edge in G' is traversed. We note that this constraint would be required even if an instance were not a rural instance and every street needed to be plowed. Constraint (5.16) enforces non-negativity and integrality of the variables.

5.2.4 Preliminary Results

Using Fleury's algorithm, IPXE can be used to solve the Rural DPP with Turn Penalties (RDPP-TP). By varying the parameter α, we can study the effects of turn penalties on the route length.

We generate instances with a 5 by 10 rectangular grid with 50 vertices. Each edge has a random service cost between 0 and 2, a deadheading cost equal to half of the service cost, and a turning penalty of 1 for all turns except going straight. These instances favor going straight, which is managerially useful since clearing the length of a street without turns is logistically easier than performing many turns. Then, we randomly delete each edge with a 20% probability. Finally, we randomly make edges required with an 80% probability. This sequence is demonstrated on a six-node instance in Figure 5.6.



(a) A six-node graph (b) Random edges are removed (20% probability) (c) Random edges are required (80% probability)

Figure 5.6: Generating a random 2 by 3 six-node graph

We solve 100 randomly generated instances with $\alpha = 0, 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 99,$ and 100%. Using a single thread of a 1.8 GHz Intel i5 processor, the average run time

per instance was 0.21 seconds per instance. The results are shown in Figure 5.7. We see that when turn penalties are ignored and the objective consists of only route length ($\alpha = 100\%$), the average percentage of straight turns is 31.9%. For all other values of α, that is, whenever turn penalties are taken into account with any non-zero weight, the percentage of straight turns increases to a single value for all instances, on average 60.9%. When route length is ignored, and the objective consists of only turn penalties ($\alpha = 0\%$), the average deviation of the route length from the lower bound is 8.2%. For all other values of α, the route length is optimal for all instances because they match the objective value when only route length is considered ($\alpha = 1$). This three-phase behavior for $\alpha = 0\%$, $0 < \alpha < 100\%$, and $\alpha = 100\%$ was observed for many different sized instances, ranging from 2 by 3 to the aforementioned 5 by 10. The conclusion is that one can often optimize for turn penalties without any negative impact to route length, unless one ignores route length completely. Therefore, for the RDPP-TP, route length and turn penalties are not generally competing objectives.

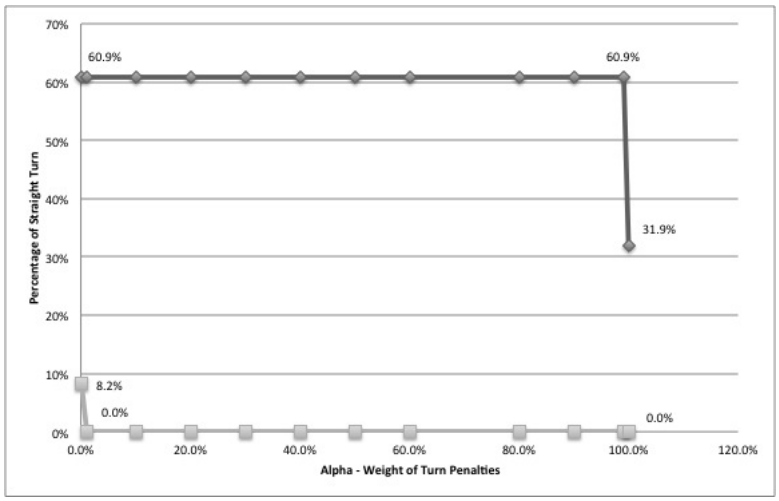


Figure 5.7: Aggregate results for 100 randomly generated instances. Dark line indicates average percentage of straight turns. Light line indicates average percentage deviation from the lower bound with respect to total length.

To gain additional insight, we investigate the distribution of the proportion of straight turns over all route-length-optimal tours on a sample rectilinear instance. We selected one of the random 5 by 10 instances mentioned earlier in this section, converted it to an optimal Eulerian graph using the solution framework, and then generated a tour using the FindRoute method from Chapter 3. By randomizing the order in which edges are investigated at each step in FindRoute, we can produce a new, randomly generated tour each time FindRoute is used. We sample 1,000 random tours, without repetition, from the set of all tours that obey the solution framework and compute the proportion of straight turns. The resulting distribution is shown in Figure 5.8 and the effect on route length and turn penalties is shown in Figure 5.9.

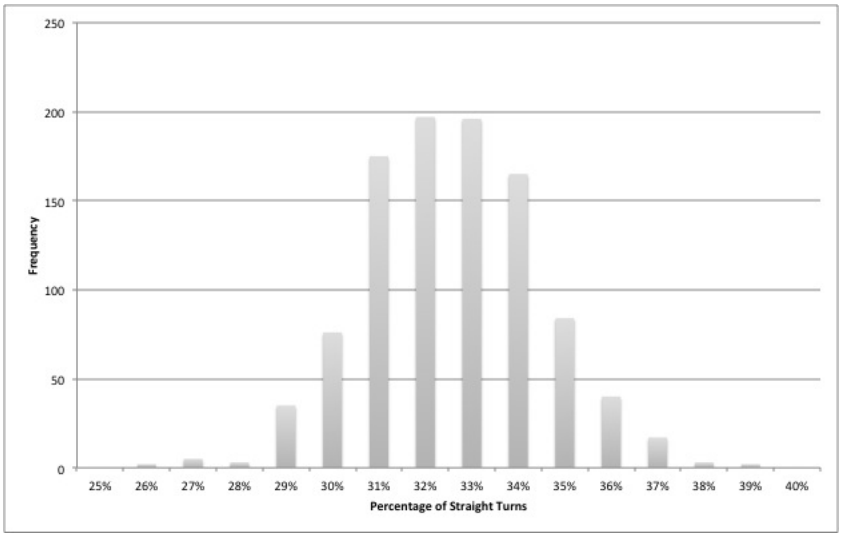


Figure 5.8: Distribution of straight turns.

The average proportion of straight turns in the sample instance is 32.5%, which aligns with the value of 33.3%, obtained when $\alpha = 100\%$. We see that the probability that the proportion of straight turns is greater than 40% is extremely small, indicating that one must actively optimize for turn penalties to obtain a large proportion of straight turns on the order of 52.0%.

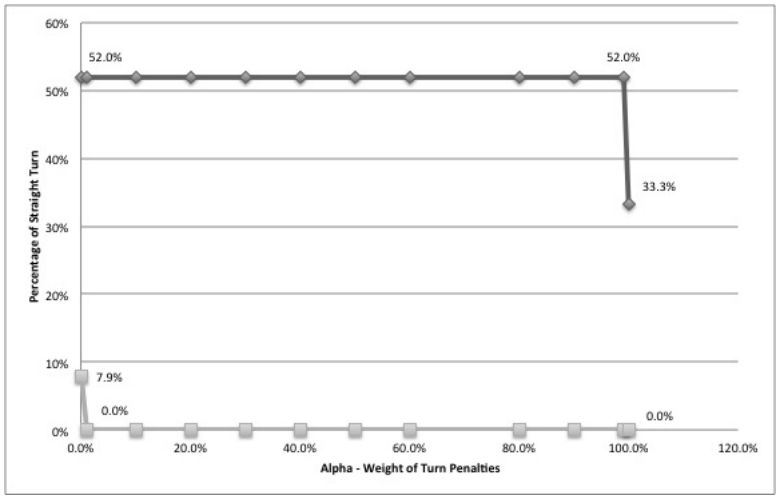
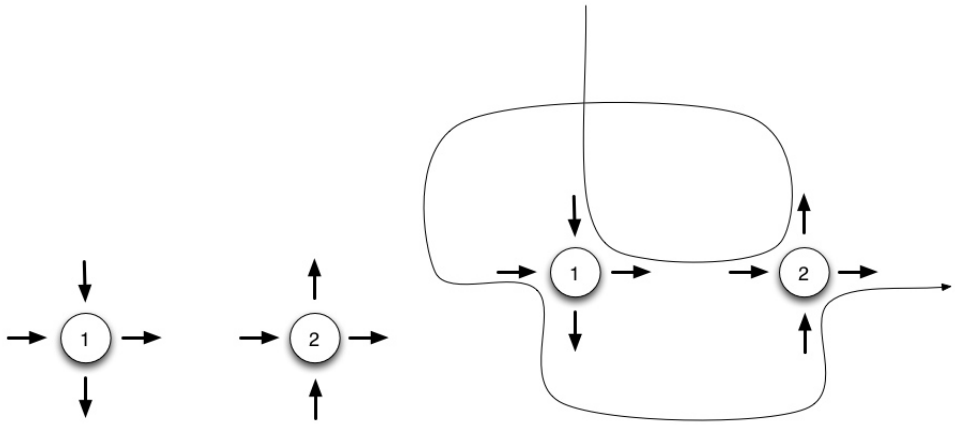


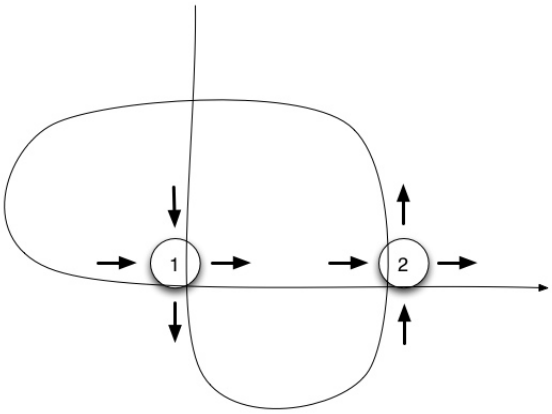
Figure 5.9: Results for a randomly generated instance. Dark line indicates average percentage of straight turns. Light line indicates average percentage deviation from the lower bound with respect to total length.

Finally, we attempt to explain two attributes seen in Figure 5.7. First, there is a jump in the number of straight turns when α transitions from $\alpha = 1$ to $0 < \alpha < 1$. Consider two nodes shown in an arbitrary graph shown in Figure 5.10(a). The arcs indicate the direction of travel as indicated by the solution framework when optimizing for route length. In Figure 5.10(a), both nodes are what we call conformal, meaning that the solution framework allows straight turns at both nodes. In Figure 5.10(b) we give a possible optimal cycle segment. Note that the cycle segment has no straight turns at nodes 1 and 2. We can, however, reorder cycle sub-segments to increase the number of straight turns, as seen in Figure 5.10(c), while preserving the optimal route length by maintaining adherence to the solution framework. Similar reorderings can be performed for pairs of degree 3 nodes. This process can, therefore, be repeated with any pair of conformal nodes, allowing for the construction of an optimal route with respect to route length with more straight turns than an initial optimal solution.



(a) Two 4-degree nodes in an arbitrary graph. Arcs indicate direction of travel indicated by solution framework for $\alpha = 1$

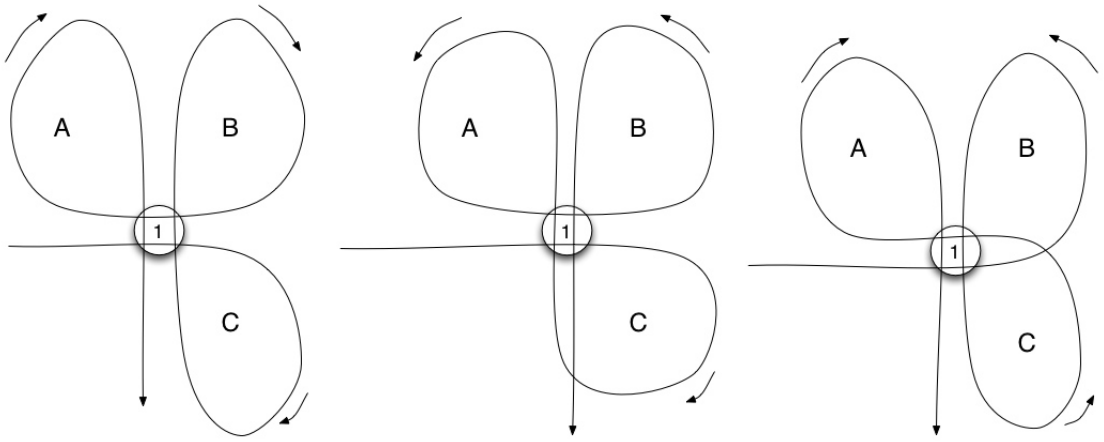
(b) A possible cycle that is optimal with respect to route length with no straight turns



(c) A different cycle that is also optimal with respect to route length with four straight turns (vertical through 1, vertical through 2, horizontal through 1, horizontal through 2)

Figure 5.10: Reordering cycle order to increase the number of straight turns while maintaining optimal route length

There is also a decrease in the route length when α transitions from $\alpha = 0$ to $0 < \alpha < 1$. Consider the example cycle in Figure 5.11(a), which has four straight turns. We note that it is important that in the DPP we traverse each street twice, giving each node of degree four at least four sub-cycles. In Figures 5.11(b) and 5.11(c), we show two of six possible ways to perform four straight turns. In each case, a combination of sub-cycles A, B, and C are reversed, while preserving the number of straight turns (since reversing sub-cycles does not change a straight turn into a non-straight-turn anywhere else). By reversing sub-cycles, there is the chance of reducing the overall route length because plowing and deadhead costs are not symmetric. Similar sub-cycle reversals that preserve the number of straight turns can be performed on nodes of lower degree. This process can be repeated on any node, allowing for the construction of an optimal route with respect to turn penalties with a lower route length than the initial solution.



(a) An example cycle that has four straight turns at node 1

(b) A new cycle with four straight turns where sub-cycles A and B are reversed

(c) A new cycle with four straight turns where sub-cycles B and C are reversed

Figure 5.11: Reversing sub-cycles to reduce route length while maintaining number of straight turns

While explaining the jumps in both objectives as α transitions from $\alpha = 0$ to $0 < \alpha < 1$ and $0 < \alpha < 1$ to $\alpha = 1$, the above analysis does not explain why both objectives are optimal when $0 < \alpha < 1$ for our set of test instances. We hypothesize that this is because the number of optimal solutions with respect to route length is very large with substantial variety with respect to the number of straight turns. Since the optimal number of straight turns is discreet, it seems reasonable that one of the numerous route-length-optimal solutions attains it. We suggest that this is an interesting avenue for future research.

5.3 Precedence in CPLS-M

Precedence, introduced in the PPP, was incorporated in the objective function of the CPLS. In addition to route balancing, CPLS-M could be modified to incorporate the impact of precedence on the objective function.

This modification is not trivial when incorporating multiple plows. With a single plow, as in the PPP, the decision of whether or not to plow a street could be decided optimally by a greedy algorithm. However, with multiple plows, this is not possible.

Suppose plow A and plow B both plow street s and, without loss of generality, plow A reaches s first. If s is plowed 0 or 2 times, the action of A is determined. However, if one side of s is plowed, than A has the option of plowing or deadheading s. If A plows, then A's route is made longer. If A deadheads, then B's route is made longer. It is not clear which option is preferable, since we are minimizing the maximum route. This same decision is encountered in the MM k-DPP, and the same method of random assignment or IP formulation could be used for this problem.

Chapter 6

Conclusions

In this dissertation, we developed metaheuristics to solve four new arc routing problems. In street scheduling and sweeping, managers in Washington DC wanted to know how to change their parking signs to allow for shorter routes for their street sweepers over a two-day period. We developed a genetic algorithm that took sub-cycles from good solutions and incorporated them into lesser solutions in order to improve them. We tested our genetic algorithm (GA) against a CPLEX implementation and a naive local search algorithm (LS) on a set of 40 small test instances, 20 large test instances, and a map of Washington DC.

Our CPLEX implementation hit its time limit of 7,200 seconds on 25 of the 40 small instances, and ranged from 0.743 seconds to 6,580 seconds with an average of 548 seconds when it did not hit its time limit. In comparison, the average running time of GA was 5.4 seconds and the average running time of LS was 1.2 seconds. Our GA produced good solutions with respect to CPLEX on the 38 instances where CPLEX produced a solution. On average, the solutions generated by our GA were 0.16% worse than the CPLEX solutions and 1.2% better than LS for small instances.

CPLEX failed to obtain a feasible solution to 18 of 20 large instances. GA outperformed LS by an average of 0.37% with respect to route length on large instances. For 100-node instances, the average running time of GA and LS was 44 and 34.4 seconds, respectively. For the largest instances with 225 nodes, the average running time for GA and LS was 708.7 and 3006.2 seconds, respectively, showing that the running time performance

of LS degrades substantially compared to GA on the largest instances.

Additionally, we showed that the same solution quality can be achieved by allowing changes to approximately 12% of street signs of an initial solution when compared to producing a solution from scratch. This is useful to a manager because replacing a parking sign has both a physical cost and inconvenience cost. In future research, we might consider multiple sweepers, which would require additional decision variables that assign street sides to a sweeper. We might also consider extending the planning period to more than two days. For example, there might be no parking on street sides on some days of the week, such as Monday, Wednesday, and Friday, or Tuesday and Thursday. We might incorporate practical concepts into the objective function, such as encouraging right-hand turns at an intersection because of the shorter traversal time compared to left-hand turns. Finally, real-world considerations such as sweeper capacity and intermediate dumping facilities could be taken into account.

In plowing with precedence, we presented a variant of the windy postman problem that addresses a practical consideration in winter street maintenance – traversing a street after it has been plowed is faster than plowing it when it is snow covered. This new consideration introduces the concept of precedence in a route, where an edge cannot be deadheaded until it is serviced. We proposed an algorithm, CPLS, that constructs an initial solution and improves it iteratively using a local search and reinitialization procedure. CPLS generated near optimal results that were within 0.17% of the lower bound, on average, for 45 benchmark instances. We observed that the deviation from the lower bound increased linearly with respect to the average deviation in cost between plowing uphill and plowing downhill. Based on this observation, we generated 20 difficult instances of the PPP and solved them using the CPLS algorithm. For these 20 instances,

the results were still within 1.3% of the lower bound, on average.

In future work, we could consider the more general case where the possible traversal times of a street are a function of the number of times that the street is traversed. Furthermore, it would be interesting to determine how tight the lower bound is on large instances with a large average deviation in cost. If the bound is tight, improvements could be made in accuracy and running time.

The MM k-DPP is a new arc routing problem that incorporated multiple plows and the observation that the cost of service is not necessarily the same. To solve the MM k-DPP, we extended CPLS to CPLS-M, which does not incorporate both cumulative route length and route balancing in its objective function. Instead, it fixes cumulative route length and focuses on route balancing. This produced very high-quality solutions. We showed that the lower bound, obtained by assuming an optimal tour for a single plow can be split equally among multiple plows, was attained in 15 of 80 benchmark instances, with an average deviation of 0.09% for the small instances and an average deviation of 1.92% for the largest instances. The apparent tightness of the lower bound in the MM k-DPP shows that one can optimize for cumulative route length (by ignoring the min-max objective) and produce optimal route balancing while maintaining the same cumulative route length. Clearly, cumulative route length and route balancing are not competing objectives.

We extended the MM k-DPP to the MM k-DPPE by incorporating turn penalties, rural instances, and precedence. We demonstrated how CPLS-M can be adapted to handle these new variants by modifying the solution framework to handle rural instances and turn penalties, and by modifying the objective function of CPLS-M to incorporate precedence. By reformulating the solution framework, we presented preliminary results on turn

penalties for 100 randomly generated test instances, and showed that turn penalties can be minimized with no effect on route length.

The MM k-DPPE incorporates many of the natural extensions of our work. However, there are several further constraints and variants that could be considered in future research. In plowing operations, it is common for managers to pre-position plows in different locations in anticipation of a storm, creating a multi-depot, multiple plow problem. Determining the optimal location of depots in addition to any routing in the MM k-DPPE would be a new problem. In some plowing operations, snow cannot be simply pushed to the side of the road, but must be loaded and carried away. In these cases, the snow plows have capacity constraints where, once they accumulate enough snow, they must return to a depot or intermediate location to dump their load. This introduces a host of new problems on top of the MM k-DPPE, such as how to route plows given capacity constraints, where to place intermediate dumping facilities, and routing vehicles that don't plow but pick up gathered snow, relieving plows of their capacity. Finally, Laporte et al. [42] introduce a plowing problem where the routing of plows must be synchronized so that multiple plows can plow a major street at the same time. This problem could be extended to incorporate route balancing and precedence, which gives the possibility of one plow clearing a street and allows another to deadhead it later.

Euclidean cycle decomposition is a very flexible methodology and was used in different forms for each problem in this dissertation. To our knowledge, the use of moving or permuting sub-cycles as a way to change and improve a Eulerian cycle is novel and very robust at improving solutions. We hope that cycle permutation will be considered in solving future arc routing problems.

Appendix A

Instances

Instance: A3101

Number of Nodes: 116

Number of Arcs: 176

node1	node2	cost1	cost2	dh1	dh2
1	2	69	62	19	19
1	33	40	82	6	6
1	100	101	20	5	5
2	3	61	60	9	9
4	7	42	47	1	1
5	6	12	19	1	1
6	9	28	27	11	11
8	13	39	32	1	1
10	15	23	29	6	6
11	17	39	33	12	12
12	13	110	111	34	34
13	14	23	22	2	2
16	20	45	36	3	3
17	18	35	35	5	5
18	21	77	77	7	7
19	25	203	203	32	32
20	22	63	64	28	28
22	23	28	29	13	13
24	27	146	146	42	42
25	26	72	72	13	13
25	31	60	62	28	28
28	39	149	145	32	32
29	33	36	36	8	8
30	36	81	82	21	21
31	32	44	44	20	20
32	33	43	47	8	8
34	37	88	84	9	9
35	39	100	96	36	36
36	41	77	73	29	29
38	46	106	108	51	51
40	41	68	63	26	26
41	42	71	72	23	23
42	43	45	44	16	16
44	49	96	101	43	43
45	46	32	28	11	11
47	85	88	83	11	11
48	87	83	88	23	23
50	56	216	220	61	61
50	58	97	91	36	36
51	59	55	51	16	16
52	53	29	33	15	15
52	60	47	49	7	7
53	61	57	50	14	14
54	62	39	41	13	13
55	62	36	31	13	13
57	66	98	95	14	14
61	62	50	53	1	1
63	64	24	29	12	12
65	66	33	27	14	14
65	71	40	39	17	17
66	72	44	39	16	16

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
67	97	141	141	14	14
68	107	34	35	15	15
69	94	113	110	44	44
70	75	78	77	10	10
70	80	85	76	12	12
73	77	40	48	2	2
74	75	129	121	17	17
76	77	55	62	17	17
78	81	63	61	26	26
79	82	135	129	1	1
80	81	51	43	15	15
83	94	269	270	31	31
83	106	383	385	31	31
84	85	97	93	29	29
86	87	29	29	7	7
87	88	53	56	26	26
88	91	54	53	17	17
88	92	124	128	13	13
89	90	39	40	6	6
93	94	86	89	24	24
95	96	26	25	2	2
97	98	10	12	2	2
99	100	49	40	20	20
100	101	59	53	22	22
102	103	37	37	11	11
102	104	45	36	10	10
103	105	57	57	2	2
108	109	63	55	3	3
108	110	234	231	116	116
110	111	21	25	8	8
110	112	92	83	38	38
112	113	50	50	12	12
112	114	57	62	6	6
115	116	30	28	8	8
3	10	124	128	31	31
3	12	240	241	9	9
4	5	60	63	24	24
5	8	37	35	11	11
6	7	14	14	2	2
7	10	33	36	16	16
8	9	23	13	4	4
9	10	24	28	9	9
9	14	44	45	8	8
10	11	31	35	8	8
10	16	39	41	16	16
11	18	68	70	26	26
12	30	322	319	5	5
14	15	25	22	4	4
15	16	25	25	4	4
15	19	45	38	3	3
16	17	11	8	2	2
17	21	39	36	6	6
19	20	34	32	9	9
20	21	28	24	9	9
22	27	62	65	26	26
24	28	50	45	15	15

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
25	32	38	39	10	10
26	29	20	17	3	3
27	29	88	81	9	9
27	34	39	41	16	16
30	40	105	100	39	39
31	36	46	49	18	18
33	34	84	88	24	24
36	37	136	140	18	18
37	38	77	80	28	28
37	44	84	80	37	37
38	39	30	25	6	6
38	45	110	113	37	37
39	46	128	126	12	12
40	56	210	216	6	6
41	84	221	220	77	77
42	47	101	99	5	5
43	44	38	40	13	13
44	48	98	98	24	24
46	49	62	57	19	19
47	48	71	76	6	6
48	49	62	60	14	14
49	88	97	103	10	10
50	51	56	52	14	14
51	52	28	37	14	14
53	54	55	47	2	2
54	55	35	37	18	18
55	63	45	44	5	5
56	57	62	67	22	22
56	63	48	49	13	13
57	84	116	110	52	52
58	63	161	169	23	23
59	60	31	35	9	9
60	61	31	33	13	13
63	65	49	56	16	16
67	68	99	105	39	39
67	102	56	61	25	25
69	106	181	182	59	59
70	71	36	40	11	11
71	72	28	27	9	9
71	76	38	32	4	4
72	73	37	38	11	11
75	79	39	43	3	3
76	81	50	47	3	3
79	80	50	46	10	10
81	82	40	48	9	9
82	83	97	92	1	1
84	93	112	113	24	24
84	94	132	133	54	54
85	86	39	37	1	1
85	93	88	89	44	44
86	89	32	37	7	7
87	90	49	52	17	17
89	96	40	40	11	11
90	91	54	48	10	10
90	100	104	97	11	11
91	92	38	40	10	10

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
91	100	97	101	7	7
93	95	49	42	3	3
95	97	42	41	11	11
96	98	47	46	22	22
96	99	53	60	13	13
98	99	29	25	7	7
100	103	164	169	67	67
104	105	48	46	11	11
105	108	104	105	23	23
106	107	63	66	19	19
107	108	55	47	18	18
112	116	83	88	13	13
114	115	43	45	6	6

instance hd115

numNodes 70

numArc 141

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
3	4	12	11	2	2
5	6	1	1	1	1
7	8	13	15	1	1
7	9	24	28	5	5
10	11	5	13	3	3
12	13	37	33	9	9
4	14	23	16	6	6
13	15	33	29	13	13
16	17	6	8	2	2
9	18	21	20	8	8
19	20	63	55	27	27
21	22	22	25	10	10
23	24	22	23	4	4
25	26	8	7	2	2
25	27	19	20	4	4
24	28	14	12	5	5
20	28	28	32	13	13
29	30	14	16	4	4
31	32	30	22	8	8
26	33	18	17	3	3
26	27	8	13	3	3
22	32	18	16	7	7
33	34	10	16	2	2
27	34	18	20	4	4
30	35	31	27	12	12
36	37	11	14	1	1
33	38	9	10	2	2
39	40	26	28	1	1
41	42	49	52	19	19
43	44	11	12	1	1
45	46	1	4	1	1
47	48	34	43	2	2
46	49	2	11	1	1
50	51	15	13	1	1
52	53	24	29	2	2
54	55	50	48	5	5
54	56	14	9	4	4
57	58	27	20	2	2
59	60	2	1	1	1
61	62	30	23	8	8
53	63	5	8	1	1
64	65	1	2	1	1
58	66	10	13	2	2
60	67	16	11	2	2
55	68	15	14	1	1
69	70	9	1	1	1
56	65	24	25	9	9
62	70	6	5	1	1
63	70	11	11	6	6
1	9	31	28	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
1	12	30	31	13	13
1	17	36	37	2	2
2	3	63	63	17	17
2	12	27	28	11	11
2	30	70	75	1	1
2	46	67	70	22	22
2	49	59	67	26	26
3	9	38	41	3	3
3	14	20	19	8	8
3	30	47	48	15	15
3	42	177	176	76	76
4	5	14	15	1	1
4	6	13	21	2	2
5	11	13	14	2	2
6	10	10	12	5	5
6	11	14	12	3	3
7	18	11	13	6	6
7	30	37	35	13	13
8	16	20	11	4	4
8	18	18	13	7	7
9	30	43	47	19	19
10	15	14	15	4	4
13	35	101	106	32	32
13	48	164	172	32	32
14	15	8	6	3	3
14	42	167	170	13	13
16	18	15	16	5	5
16	42	185	184	27	27
16	47	119	124	6	6
16	53	72	70	7	7
16	57	117	116	26	26
17	18	28	25	12	12
17	49	42	41	2	2
19	29	23	23	4	4
19	34	13	15	6	6
20	35	10	12	4	4
20	41	218	219	82	82
20	42	184	178	23	23
21	38	9	15	3	3
22	31	18	25	6	6
23	29	53	56	16	16
23	31	16	13	1	1
23	32	32	30	15	15
24	31	5	7	2	2
25	37	24	15	3	3
28	29	17	19	8	8
30	47	157	162	24	24
30	53	114	112	27	27
30	57	162	160	42	42
35	41	216	223	27	27
35	42	179	178	75	75
36	38	28	35	13	13
37	58	31	27	12	12
37	66	24	25	3	3
39	43	7	14	1	1
39	44	19	23	8	8

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
40	44	23	23	5	5
40	45	30	34	3	3
40	46	39	38	4	4
41	48	10	9	1	1
42	47	66	67	7	7
42	48	54	59	17	17
42	57	238	232	26	26
43	60	35	34	12	12
44	45	53	48	16	16
44	46	51	50	10	10
45	49	15	9	2	2
47	57	168	167	7	7
47	61	92	94	12	12
50	52	4	5	2	2
50	69	8	4	2	2
51	52	2	8	1	1
51	62	34	28	5	5
51	66	17	17	3	3
52	69	9	9	2	2
53	61	22	20	7	7
54	57	18	16	2	2
54	65	31	33	4	4
55	59	57	50	12	12
55	61	58	66	20	20
55	67	68	64	14	14
56	64	20	20	1	1
59	67	7	9	1	1
59	68	63	58	16	16
61	67	31	29	12	12
61	68	70	68	28	28
62	63	13	14	6	6
63	69	8	8	2	2
64	68	20	10	2	2
67	68	72	64	13	13

instance hd215

numNodes 70

numArc 162

node1	node2	cost1	cost2	dh1	dh2
1	2	2	9	1	1
1	3	3	8	1	1
1	4	6	9	1	1
2	3	1	1	1	1
2	4	7	11	3	3
5	6	95	94	35	35
7	8	22	26	3	3
9	10	2	6	1	1
9	11	43	35	14	14
12	13	1	9	1	1
10	14	5	13	2	2
15	16	7	9	3	3
15	17	8	10	4	4
16	18	4	7	1	1
19	20	20	17	2	2
19	21	15	17	3	3
22	23	15	14	4	4
23	24	17	17	5	5
25	26	16	22	5	5
25	27	21	23	11	11
28	29	26	27	7	7
28	30	21	21	4	4
27	28	1	5	1	1
27	31	28	20	10	10
32	33	13	16	3	3
34	35	22	18	9	9
36	37	24	18	9	9
35	36	50	49	3	3
38	39	25	28	10	10
37	40	9	4	1	1
41	42	49	49	19	19
40	43	26	20	10	10
39	44	16	14	5	5
44	45	35	34	12	12
46	47	4	7	2	2
48	49	6	4	1	1
50	51	9	10	1	1
50	52	19	28	9	9
53	54	28	21	9	9
55	56	26	22	1	1
55	57	10	14	5	5
54	58	38	30	12	12
59	60	5	12	1	1
57	59	15	14	2	2
61	62	14	20	7	7
63	64	22	20	2	2
52	62	27	22	10	10
54	62	18	18	9	9
57	60	14	16	6	6
65	66	4	6	2	2
51	67	31	30	7	7

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
51	68	23	27	3	3
68	69	23	17	3	3
64	70	3	1	1	1
54	68	23	25	11	11
2	41	157	158	38	38
2	65	210	213	51	51
3	12	8	12	3	3
3	13	13	7	1	1
4	12	22	18	5	5
4	41	164	159	29	29
4	65	214	208	54	54
5	7	49	52	2	2
5	13	42	42	13	13
5	22	55	63	11	11
5	26	64	64	24	24
5	31	102	104	17	17
5	32	76	83	32	32
5	35	74	69	15	15
5	42	127	120	58	58
5	49	55	56	25	25
5	50	88	84	12	12
5	53	129	130	45	45
5	69	78	85	3	3
6	11	42	43	9	9
6	65	256	256	105	105
7	35	78	84	24	24
7	47	31	34	12	12
7	49	65	68	2	2
7	50	95	87	37	37
7	69	95	86	26	26
8	9	35	36	5	5
8	47	29	31	4	4
10	11	37	42	13	13
10	13	51	47	6	6
11	13	74	81	16	16
11	14	39	31	12	12
11	22	85	82	31	31
11	24	80	83	21	21
13	14	47	45	9	9
13	22	65	68	21	21
13	26	68	74	12	12
13	31	115	119	12	12
13	53	144	145	59	59
15	23	17	19	6	6
15	24	14	13	3	3
16	23	27	27	10	10
17	20	70	72	5	5
17	22	36	28	11	11
17	24	6	11	3	3
18	21	11	11	5	5
18	23	20	16	8	8
19	29	13	14	4	4
20	29	24	23	5	5
22	26	87	89	44	44
22	53	163	157	20	20
25	28	18	19	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
25	30	35	35	13	13
26	31	45	52	6	6
26	50	65	58	13	13
26	53	73	68	2	2
26	68	54	52	3	3
26	69	40	44	10	10
27	30	21	17	8	8
29	30	7	10	3	3
31	53	52	49	12	12
32	35	30	33	14	14
32	41	45	40	14	14
32	42	56	52	7	7
32	43	59	56	26	26
32	69	97	102	36	36
33	46	9	8	1	1
33	47	10	4	1	1
34	36	44	38	7	7
34	38	42	37	16	16
34	48	8	11	2	2
34	49	19	15	1	1
35	42	83	77	38	38
35	48	9	15	2	2
36	38	52	49	2	2
36	43	10	4	2	2
36	48	47	51	8	8
37	43	26	21	6	6
37	44	6	12	2	2
38	48	40	42	5	5
38	63	46	41	7	7
38	67	21	20	4	4
39	70	38	32	3	3
40	44	9	14	3	3
40	45	21	24	2	2
41	46	23	28	1	1
41	65	186	185	14	14
42	43	47	46	9	9
42	45	18	13	1	1
42	69	142	144	27	27
49	50	41	43	8	8
49	69	44	47	15	15
50	69	37	39	8	8
51	52	5	14	1	1
52	67	28	25	13	13
53	58	11	14	3	3
55	61	14	17	2	2
56	58	17	18	2	2
56	61	7	14	1	1
58	62	43	43	6	6
58	66	57	62	13	13
59	63	4	4	1	1
60	63	3	8	2	2
60	64	19	16	3	3
62	66	99	103	29	29
66	70	61	61	5	5
67	70	40	43	11	11

instance hd315

numNodes 68

numArc 153

node1	node2	cost1	cost2	dh1	dh2
1	2	30	28	9	9
3	4	3	10	1	1
5	6	15	15	1	1
7	8	7	11	3	3
9	10	23	25	2	2
11	12	9	8	2	2
12	13	33	29	8	8
14	15	19	24	5	5
2	16	16	16	4	4
4	15	25	20	3	3
17	18	23	23	5	5
19	20	31	32	8	8
19	21	25	28	9	9
18	22	9	4	2	2
18	21	1	5	1	1
23	24	31	30	9	9
25	26	34	36	9	9
20	27	15	14	7	7
28	29	26	28	3	3
20	29	11	9	4	4
30	31	5	14	1	1
29	32	31	28	10	10
33	34	19	23	8	8
35	36	22	18	7	7
37	38	6	10	1	1
37	39	23	16	1	1
37	40	19	16	2	2
41	42	2	5	1	1
42	43	6	13	2	2
42	44	27	28	13	13
36	45	68	59	11	11
36	46	22	16	1	1
40	47	6	7	2	2
44	48	8	1	1	1
49	50	7	7	1	1
51	52	8	8	2	2
51	53	11	13	6	6
54	55	8	8	2	2
56	57	9	18	1	1
50	56	18	25	8	8
56	58	10	10	2	2
59	60	12	22	5	5
59	61	13	11	2	2
62	63	22	20	8	8
50	55	20	19	9	9
57	58	13	17	2	2
64	65	32	25	9	9
50	58	19	19	2	2
66	67	21	13	6	6
67	68	32	27	6	6
1	3	65	59	25	25

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
1	9	32	27	3	3
1	16	14	15	6	6
2	13	74	71	12	12
2	27	83	84	13	13
2	28	64	56	12	12
2	32	67	67	17	17
3	9	72	71	27	27
3	15	24	30	3	3
3	35	59	60	16	16
4	11	31	29	1	1
4	14	26	19	4	4
5	7	24	24	8	8
5	13	27	21	1	1
5	16	80	80	22	22
6	7	19	16	4	4
6	24	72	76	25	25
6	30	25	24	8	8
6	31	25	29	12	12
6	33	29	32	5	5
6	49	106	107	7	7
8	33	18	23	4	4
10	11	20	24	5	5
12	14	35	30	3	3
13	16	73	71	31	31
13	24	90	97	30	30
13	26	116	116	30	30
13	28	43	44	4	4
13	32	52	45	17	17
14	35	42	42	8	8
14	36	50	57	1	1
15	35	37	32	12	12
15	36	47	50	6	6
16	24	148	149	33	33
16	26	170	166	17	17
16	53	231	241	100	100
17	19	30	31	10	10
17	22	23	22	2	2
17	23	1	1	1	1
19	22	30	26	12	12
19	65	115	113	44	44
20	24	84	82	12	12
20	65	129	134	55	55
23	25	38	37	10	10
23	26	13	14	1	1
24	26	43	36	17	17
24	30	53	47	6	6
24	31	52	55	22	22
24	49	69	65	2	2
24	54	62	59	17	17
24	55	62	70	29	29
25	53	78	82	13	13
25	64	47	39	14	14
27	29	11	13	3	3
27	30	56	46	20	20
27	32	38	35	11	11
28	32	16	14	7	7

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
30	32	39	35	3	3
30	49	87	90	32	32
30	54	78	73	24	24
30	55	80	80	36	36
31	48	138	137	4	4
31	49	91	86	36	36
31	54	77	82	8	8
31	55	91	81	19	19
31	68	115	115	16	16
34	35	70	62	9	9
34	39	5	6	2	2
34	55	34	33	7	7
34	60	91	92	22	22
37	55	66	58	12	12
37	60	120	120	16	16
38	39	23	29	6	6
38	40	18	18	8	8
38	55	63	65	28	28
38	60	119	118	43	43
40	46	22	19	5	5
41	43	10	12	3	3
41	44	30	26	5	5
43	46	38	32	9	9
44	45	7	2	1	1
45	47	28	35	12	12
45	48	3	9	1	1
47	68	69	70	19	19
48	68	70	69	15	15
49	54	31	27	13	13
49	56	10	15	2	2
49	58	14	16	2	2
51	59	34	32	15	15
51	61	46	40	14	14
52	53	7	6	3	3
52	59	46	41	7	7
55	60	89	85	12	12
57	64	24	22	7	7
57	65	6	7	1	1
60	61	10	7	3	3
60	63	52	45	1	1
60	64	50	49	8	8
60	65	60	66	27	27
62	66	1	2	1	1
63	64	55	56	20	20
63	65	72	72	23	23
63	67	14	11	1	1

instance hd415

numNodes 88

numArc 171

node1	node2	cost1	cost2	dh1	dh2
1	2	6	6	3	3
3	4	7	7	1	1
3	5	26	28	11	11
3	6	37	46	14	14
4	7	50	43	18	18
8	9	12	18	3	3
10	11	3	1	1	1
9	10	21	22	5	5
12	13	10	8	4	4
12	14	29	27	13	13
15	16	12	7	3	3
5	17	34	31	8	8
2	18	11	5	1	1
9	19	15	21	6	6
17	20	33	39	11	11
6	21	11	4	1	1
21	22	17	15	6	6
14	23	13	23	4	4
22	23	5	2	1	1
9	11	18	16	3	3
24	25	58	62	11	11
26	27	20	17	1	1
26	28	7	12	4	4
29	30	27	28	5	5
31	32	14	20	2	2
31	33	12	16	4	4
33	34	22	18	2	2
34	35	29	27	4	4
36	37	12	11	1	1
36	38	16	22	5	5
28	36	23	19	10	10
32	33	12	20	4	4
32	39	22	17	6	6
25	32	24	23	10	10
39	40	19	14	3	3
30	41	19	21	2	2
35	41	31	22	8	8
37	42	25	15	4	4
28	37	14	8	4	4
30	35	13	17	3	3
27	42	11	9	1	1
27	28	16	13	6	6
40	43	31	28	1	1
38	44	12	15	4	4
44	45	29	35	6	6
45	46	9	10	3	3
42	45	16	15	3	3
47	48	13	14	1	1
47	49	9	16	2	2
49	50	16	7	2	2
50	51	20	23	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
52	53	46	47	17	17
48	54	10	11	3	3
51	54	11	13	3	3
55	56	10	9	2	2
57	58	1	8	1	1
57	59	7	6	3	3
60	61	39	36	15	15
60	62	16	15	4	4
60	63	34	34	16	16
58	64	10	4	2	2
58	59	13	16	4	4
65	66	12	18	6	6
67	68	25	30	9	9
66	67	5	2	1	1
69	70	9	18	3	3
69	71	44	45	8	8
69	72	16	15	7	7
73	74	17	15	6	6
73	75	4	12	1	1
70	76	22	24	10	10
74	77	11	10	1	1
77	78	30	35	4	4
77	79	8	9	4	4
80	81	25	30	5	5
68	82	14	7	3	3
83	84	8	3	1	1
76	85	9	10	5	5
74	79	11	10	4	4
71	86	10	12	3	3
78	86	19	24	4	4
75	78	29	31	4	4
87	88	6	4	1	1
72	84	21	18	1	1
81	88	11	9	3	3
82	88	7	7	2	2
1	4	31	27	7	7
1	18	13	17	2	2
1	58	43	42	2	2
2	6	42	37	13	13
2	64	30	28	7	7
3	14	31	23	10	10
4	5	24	26	2	2
5	18	53	57	14	14
6	18	53	50	21	21
6	22	20	21	1	1
6	64	40	42	3	3
7	8	24	25	8	8
7	14	20	19	10	10
7	40	26	27	7	7
8	19	21	15	5	5
8	53	175	173	63	63
10	16	9	13	4	4
11	15	11	10	1	1
11	16	4	6	1	1
12	23	13	20	3	3
12	40	41	40	13	13

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
13	21	14	16	4	4
13	22	5	7	1	1
13	23	11	12	6	6
15	20	17	9	1	1
17	43	108	104	13	13
17	63	168	166	58	58
19	20	5	13	1	1
19	53	168	174	7	7
21	60	122	115	23	23
21	61	160	155	15	15
21	62	124	130	2	2
21	68	74	71	18	18
21	70	118	118	34	34
24	27	13	15	5	5
24	39	27	33	10	10
25	43	9	3	1	1
25	52	220	224	14	14
25	53	179	179	17	17
26	36	18	19	6	6
26	37	20	14	4	4
29	46	9	9	5	5
31	34	28	32	5	5
31	39	25	16	4	4
33	41	6	12	3	3
34	41	18	16	1	1
38	76	24	23	3	3
38	85	25	18	2	2
40	60	159	158	5	5
40	61	199	201	72	72
40	62	166	169	29	29
40	68	116	111	8	8
40	70	159	158	22	22
43	52	215	216	57	57
43	53	176	178	48	48
47	56	8	6	1	1
48	51	16	12	2	2
49	51	18	16	4	4
49	56	14	8	4	4
50	55	13	12	3	3
50	56	23	19	5	5
52	63	13	11	4	4
53	61	30	23	11	11
54	59	11	16	2	2
55	79	38	37	14	14
57	64	16	6	2	2
60	70	166	169	59	59
61	63	23	29	12	12
61	70	210	201	43	43
62	70	170	174	19	19
62	80	83	77	5	5
65	67	9	10	4	4
65	87	4	13	1	1
66	81	35	35	13	13
66	85	15	21	3	3
67	87	16	9	4	4
68	80	21	17	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
69	84	32	36	5	5
71	78	23	14	7	7
72	83	20	18	3	3
74	75	15	17	7	7
75	80	17	17	3	3
81	82	14	14	3	3
82	87	9	6	2	2
83	86	17	19	1	1

instance hd515

numNodes 86

numArc 178

node1	node2	cost1	cost2	dh1	dh2
1	2	17	17	4	4
2	3	36	41	5	5
3	4	24	27	6	6
5	6	4	6	1	1
7	8	2	10	1	1
7	9	12	13	6	6
7	10	1	1	1	1
7	11	14	6	2	2
8	10	4	7	2	2
8	11	6	9	3	3
12	13	88	86	32	32
2	14	28	22	7	7
15	16	21	21	10	10
16	17	5	5	2	2
2	17	42	39	16	16
6	17	43	38	1	1
9	18	1	11	1	1
16	19	7	13	1	1
6	13	38	45	2	2
20	21	6	12	1	1
20	22	10	11	2	2
21	23	13	14	2	2
21	24	7	14	1	1
25	26	23	25	12	12
25	27	17	19	4	4
28	29	42	46	5	5
30	31	18	12	2	2
31	32	17	16	7	7
22	32	9	11	2	2
33	34	19	24	5	5
33	35	21	20	8	8
33	36	15	22	8	8
35	37	20	17	1	1
35	38	24	21	11	11
35	36	7	1	1	1
26	37	22	21	8	8
23	31	11	18	3	3
23	39	5	14	1	1
36	40	28	19	4	4
24	39	5	9	1	1
41	42	27	23	8	8
43	44	27	29	12	12
43	45	15	13	2	2
43	46	22	20	6	6
47	48	19	16	1	1
47	49	9	15	3	3
50	51	15	22	6	6
50	52	10	9	3	3
48	50	52	51	3	3
53	54	30	32	10	10
46	55	37	37	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
51	56	5	8	2	2
57	58	25	23	12	12
57	59	54	52	25	25
52	56	19	20	4	4
54	60	14	24	2	2
54	61	58	63	8	8
60	62	26	32	10	10
56	62	17	16	3	3
61	62	8	10	1	1
45	58	4	8	2	2
58	63	2	6	1	1
56	60	8	11	3	3
48	64	8	18	4	4
49	64	12	4	1	1
65	66	10	13	5	5
65	67	28	23	8	8
68	69	19	13	2	2
68	70	25	28	12	12
68	71	21	26	10	10
72	73	21	18	7	7
72	74	13	11	6	6
69	73	17	17	5	5
69	71	38	31	15	15
75	76	3	5	1	1
75	77	14	6	2	2
74	75	14	11	3	3
78	79	15	17	6	6
76	80	23	22	5	5
67	79	31	22	9	9
71	79	16	13	6	6
74	77	23	16	2	2
77	80	19	19	8	8
81	82	4	1	1	1
66	83	32	29	4	4
66	84	21	28	10	10
84	85	19	20	1	1
80	86	1	1	1	1
83	86	36	44	5	5
71	84	17	17	2	2
1	14	25	16	2	2
1	63	9	9	2	2
3	12	37	40	7	7
3	14	22	17	5	5
4	12	31	27	11	11
4	46	32	32	10	10
4	49	26	30	8	8
4	65	56	57	25	25
4	85	49	55	18	18
5	15	47	40	2	2
5	16	38	35	6	6
5	19	24	28	1	1
6	30	77	83	3	3
6	32	80	87	9	9
8	44	144	137	10	10
8	81	213	216	94	94
9	10	14	13	3	3

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
9	11	13	19	6	6
10	18	13	7	3	3
11	44	140	146	49	49
11	81	217	214	31	31
12	14	48	53	1	1
12	18	40	45	11	11
12	28	46	50	22	22
12	30	54	58	3	3
12	42	128	131	37	37
13	81	251	253	38	38
15	18	23	28	12	12
15	19	22	18	3	3
18	28	57	56	9	9
18	30	71	66	10	10
20	31	11	13	1	1
20	32	13	17	7	7
22	30	38	28	2	2
22	42	40	45	9	9
23	24	7	3	1	1
24	27	14	13	5	5
25	37	16	16	8	8
25	41	12	11	1	1
26	41	25	23	1	1
26	42	28	21	8	8
27	39	6	16	1	1
27	41	14	14	6	6
28	30	72	78	31	31
28	34	14	19	5	5
29	34	24	34	10	10
29	40	14	23	5	5
29	70	17	22	7	7
33	38	37	33	11	11
34	65	60	60	3	3
34	84	47	54	10	10
34	85	44	41	8	8
36	38	22	23	5	5
37	38	6	8	3	3
40	70	25	20	1	1
43	55	29	30	11	11
44	57	17	13	4	4
44	61	79	74	8	8
45	63	5	7	1	1
46	48	17	13	3	3
47	50	39	36	18	18
47	53	38	39	4	4
47	64	7	13	1	1
49	65	37	40	15	15
49	85	44	49	11	11
50	53	52	51	8	8
50	64	46	43	5	5
51	52	27	17	3	3
51	60	13	5	1	1
52	55	26	25	8	8
53	64	41	43	14	14
53	76	46	46	20	20
53	83	13	18	6	6

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
54	86	38	38	11	11
55	59	24	23	4	4
57	81	183	185	5	5
59	61	15	17	1	1
59	62	17	17	9	9
65	85	39	29	3	3
66	67	5	5	3	3
67	83	28	31	14	14
69	79	35	35	14	14
69	82	60	67	6	6
72	78	10	17	4	4
73	78	16	7	3	3
76	77	5	9	2	2
79	82	98	96	25	25
82	86	61	60	11	11

instance hd615

numNodes 88

numArc 181

node1	node2	cost1	cost2	dh1	dh2
1	2	26	24	10	10
3	4	26	25	9	9
3	5	24	24	3	3
6	7	11	15	4	4
6	8	30	30	10	10
9	10	7	6	1	1
2	11	15	10	4	4
2	12	26	20	10	10
13	14	6	8	1	1
15	16	17	21	6	6
5	15	6	11	1	1
15	17	16	22	7	7
4	18	3	6	2	2
4	12	36	36	8	8
19	20	26	26	9	9
21	22	16	9	5	5
7	8	22	23	3	3
17	22	16	25	7	7
10	20	23	24	11	11
23	24	21	15	3	3
23	25	13	14	6	6
26	27	29	29	10	10
26	28	24	26	9	9
24	29	10	7	2	2
24	28	1	1	1	1
30	31	34	33	9	9
30	32	43	40	11	11
33	34	25	25	8	8
25	32	30	38	3	3
35	36	24	22	6	6
27	37	15	14	4	4
36	38	13	12	4	4
36	39	28	25	5	5
27	39	4	11	2	2
40	41	9	6	1	1
29	42	6	5	1	1
38	39	24	24	10	10
43	44	25	23	3	3
45	46	13	16	4	4
45	47	18	19	6	6
48	49	21	16	3	3
50	51	20	20	10	10
52	53	22	22	10	10
54	55	13	3	1	1
51	54	17	23	5	5
54	56	21	19	9	9
46	47	6	6	2	2
57	58	22	24	9	9
47	59	11	10	3	3
55	60	11	10	3	3
55	61	16	24	7	7
44	62	24	28	2	2
47	63	26	24	9	9
53	64	61	69	4	4
53	65	20	20	7	7
49	51	14	14	3	3
56	61	9	3	1	1
63	66	8	9	4	4
67	68	12	13	2	2
67	69	7	7	4	4
70	71	9	3	2	2
70	72	11	16	3	3
73	74	8	15	4	4
68	75	8	14	2	2
68	69	20	26	1	1
68	76	7	11	4	4
77	78	19	22	5	5
77	79	10	14	3	3
71	72	15	9	4	4
71	79	52	45	19	19
80	81	19	12	3	3
80	82	27	25	4	4
80	83	6	7	1	1
69	74	14	22	1	1
75	76	13	20	4	4
84	85	27	35	9	9
69	76	29	22	9	9
81	86	18	13	2	2
81	87	25	28	3	3
78	79	12	12	1	1
82	88	35	28	8	8
1	7	62	59	12	12
1	12	6	9	3	3
1	25	109	110	41	41
1	31	81	83	26	26
2	13	23	18	8	8
3	16	17	23	2	2
3	18	21	23	1	1
4	19	35	30	7	7
5	16	13	9	2	2
5	22	20	25	8	8
6	17	7	11	4	4
7	72	233	240	72	72
8	12	77	79	21	21
8	36	56	63	1	1
8	37	92	90	37	37
8	38	74	71	18	18
9	20	32	27	5	5
9	21	37	44	9	9
9	52	63	60	7	7
10	18	31	34	3	3
10	19	26	25	7	7
11	13	22	22	7	7
11	33	61	62	10	10
11	40	24	30	1	1
11	41	29	29	8	8
11	48	36	29	11	11

continued . . .

. . . continued

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
12	36	43	39	10	10
12	38	45	53	23	23
14	48	20	22	6	6
15	22	20	15	2	2
16	18	20	18	8	8
17	21	16	15	7	7
19	52	37	42	15	15
19	53	51	58	8	8
20	52	36	39	10	10
20	53	48	44	6	6
23	26	32	32	11	11
23	29	27	18	9	9
23	30	3	6	2	2
25	30	8	14	4	4
25	31	44	43	11	11
26	29	34	25	12	12
26	85	115	112	29	29
27	31	83	79	9	9
27	85	132	134	34	34
31	33	15	19	7	7
32	72	81	75	25	25
32	84	41	38	1	1
33	40	37	39	9	9
33	41	40	34	3	3
34	67	27	32	8	8
34	73	16	17	8	8
34	74	25	22	11	11
35	37	22	18	1	1
35	38	16	8	3	3
35	40	25	24	10	10
37	39	17	11	6	6
41	43	89	81	39	39
43	66	52	59	11	11
43	83	19	23	10	10
44	57	27	27	3	3
44	87	16	8	4	4
45	59	7	8	4	4
45	65	34	37	13	13
46	59	13	11	3	3
46	63	35	30	15	15
49	52	63	63	6	6
49	74	37	33	9	9
49	78	91	83	16	16
50	60	21	13	4	4
50	74	26	36	13	13
50	78	81	86	24	24
50	83	17	13	6	6
51	60	14	15	3	3
54	60	12	19	2	2
55	56	16	14	4	4
56	65	20	21	1	1
57	62	1	1	1	1
57	87	34	30	3	3
58	61	22	25	4	4
58	62	19	15	2	2
58	66	20	20	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
59	65	37	34	16	16
61	64	33	27	1	1
62	87	26	25	10	10
63	64	1	7	1	1
64	66	1	8	1	1
67	73	30	30	2	2
67	76	21	21	2	2
70	77	36	41	14	14
70	79	40	36	5	5
71	77	41	36	3	3
74	78	83	89	2	2
75	84	22	25	11	11
75	85	2	12	1	1
78	88	20	22	4	4
80	86	5	2	1	1
81	82	8	10	2	2
84	88	34	25	5	5
85	88	40	43	3	3

instance hd715

numNodes 100

numArc 188

node1	node2	cost1	cost2	dh1	dh2
1	2	13	18	6	6
3	4	4	4	1	1
5	6	1	1	1	1
5	7	26	31	2	2
5	8	43	41	18	18
6	9	50	40	11	11
2	10	21	20	6	6
10	11	10	12	1	1
12	13	1	3	1	1
11	12	13	18	5	5
14	15	15	12	3	3
14	16	19	28	8	8
17	18	3	10	2	2
17	19	17	13	1	1
18	19	22	25	7	7
7	20	31	37	12	12
21	22	1	1	1	1
4	22	11	8	2	2
2	11	24	15	3	3
19	20	29	34	4	4
8	23	8	10	2	2
23	24	13	7	1	1
16	25	13	16	2	2
8	24	21	17	1	1
24	25	3	6	1	1
11	13	18	22	4	4
26	27	14	22	3	3
26	28	58	56	8	8
28	29	1	6	1	1
30	31	12	11	5	5
27	30	21	19	4	4
30	32	8	9	1	1
33	34	26	23	2	2
33	35	39	36	2	2
36	37	28	29	12	12
36	38	11	16	5	5
36	39	14	15	3	3
36	40	16	20	2	2
41	42	18	24	2	2
37	39	19	23	3	3
35	37	29	31	1	1
31	43	12	5	3	3
43	44	18	20	9	9
32	43	17	17	2	2
38	39	17	16	7	7
38	40	17	16	7	7
28	38	27	25	1	1
40	42	19	16	2	2
34	45	27	18	5	5
35	45	30	27	6	6
31	46	21	24	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
31	32	11	11	4	4
34	47	23	24	12	12
34	35	14	22	6	6
27	46	8	14	2	2
27	32	20	14	3	3
42	48	27	33	5	5
44	49	10	19	3	3
49	50	26	27	12	12
47	50	9	5	2	2
46	50	15	10	1	1
51	52	15	12	5	5
51	53	15	19	7	7
51	54	12	16	2	2
54	55	7	13	1	1
55	56	25	26	10	10
57	58	59	64	26	26
57	59	43	46	16	16
52	60	6	13	2	2
60	61	18	10	2	2
56	60	7	14	3	3
62	63	22	17	2	2
62	64	17	9	2	2
65	66	11	7	2	2
53	65	1	3	1	1
67	68	2	3	1	1
69	70	4	7	1	1
61	69	3	3	1	1
52	61	22	23	7	7
52	56	9	15	2	2
53	66	5	9	1	1
71	72	42	42	7	7
58	71	17	9	4	4
71	73	39	42	3	3
71	74	2	9	1	1
72	74	44	44	5	5
70	75	5	7	1	1
61	70	9	7	4	4
76	77	4	6	2	2
76	78	15	18	6	6
76	79	19	11	3	3
77	80	21	20	2	2
77	81	11	13	5	5
77	79	10	8	4	4
82	83	15	11	2	2
82	84	42	48	19	19
82	85	15	12	5	5
86	87	15	17	2	2
86	88	6	13	1	1
83	89	22	25	9	9
87	90	16	16	7	7
90	91	33	32	3	3
90	92	7	5	2	2
93	94	24	24	5	5
88	93	11	21	1	1
80	95	6	11	1	1
85	96	23	14	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
96	97	18	9	1	1
96	98	9	1	1	1
89	99	13	7	2	2
87	92	14	12	1	1
87	88	15	16	2	2
84	91	22	22	6	6
84	100	16	11	1	1
91	100	22	22	5	5
88	91	26	33	8	8
78	81	3	7	1	1
81	95	15	16	5	5
85	98	19	21	5	5
78	94	11	7	2	2
78	95	13	3	1	1
1	10	26	24	11	11
1	26	91	96	32	32
1	59	156	146	26	26
2	19	13	7	3	3
3	6	24	21	7	7
3	21	11	8	3	3
3	70	40	43	13	13
4	63	17	13	4	4
4	75	32	32	12	12
5	16	26	28	2	2
6	7	26	26	6	6
7	64	41	47	12	12
8	63	20	30	5	5
9	10	17	18	1	1
9	16	21	17	6	6
9	42	26	25	6	6
12	18	10	7	3	3
13	17	8	9	2	2
13	18	8	8	2	2
14	25	17	10	2	2
14	41	17	10	3	3
15	23	13	11	4	4
15	24	9	10	1	1
15	25	8	9	3	3
20	48	109	104	37	37
20	73	171	170	46	46
21	64	6	8	3	3
22	62	11	9	4	4
22	64	9	15	5	5
23	68	31	32	12	12
26	40	30	33	10	10
29	48	1	2	1	1
29	57	211	218	1	1
29	59	180	172	80	80
30	43	21	24	10	10
33	47	7	1	1	1
37	45	18	16	1	1
39	45	8	4	1	1
41	67	61	60	4	4
41	68	53	49	18	18
44	89	30	29	9	9
44	99	19	17	6	6

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
51	66	3	8	1	1
53	55	12	13	1	1
54	56	11	10	1	1
54	66	8	6	2	2
55	65	15	11	1	1
57	73	8	8	1	1
58	74	10	7	2	2
58	93	80	83	3	3
59	72	30	33	12	12
63	75	15	12	2	2
65	92	39	30	15	15
67	80	30	32	2	2
67	83	75	77	4	4
68	74	86	81	26	26
69	75	9	14	4	4
72	73	26	30	10	10
76	81	14	5	2	2
79	94	29	32	12	12
79	99	14	23	3	3
80	93	17	19	5	5
82	98	34	33	8	8
85	97	30	26	3	3
94	95	18	11	2	2
96	100	18	15	3	3
97	98	5	5	3	3

instance hd815

numNodes 95

numArc 193

node1	node2	cost1	cost2	dh1	dh2
1	2	21	24	4	4
3	4	10	10	1	1
5	6	22	17	8	8
2	5	39	36	2	2
5	7	20	19	3	3
8	9	9	8	3	3
10	11	4	10	2	2
10	12	11	10	2	2
10	13	6	5	2	2
10	14	6	6	3	3
11	13	6	1	1	1
11	14	14	14	6	6
15	16	86	88	1	1
2	6	22	18	7	7
17	18	25	19	5	5
18	19	2	1	1	1
2	19	40	42	9	9
19	20	14	14	1	1
9	19	45	41	5	5
12	21	6	8	2	2
18	20	5	15	2	2
13	21	13	7	3	3
9	16	39	40	16	16
22	23	5	7	1	1
22	24	11	12	4	4
23	25	11	13	4	4
23	26	9	11	3	3
27	28	16	21	7	7
27	29	21	22	3	3
27	30	10	14	2	2
27	31	19	20	6	6
32	33	45	42	3	3
34	35	16	15	1	1
35	36	11	10	2	2
24	36	14	14	6	6
37	38	15	21	3	3
37	39	20	21	4	4
37	40	39	33	13	13
37	41	15	21	8	8
28	39	26	24	7	7
39	40	22	17	5	5
39	41	1	3	1	1
28	29	20	22	1	1
42	43	23	24	5	5
25	35	9	16	1	1
25	44	7	10	2	2
25	26	5	4	1	1
41	43	23	20	3	3
33	43	14	22	3	3
26	44	4	1	1	1
29	45	22	20	8	8

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
30	45	21	30	1	1
46	47	33	24	2	2
46	48	14	20	5	5
46	49	22	15	6	6
50	51	19	17	4	4
50	52	5	6	1	1
50	53	16	16	8	8
54	55	20	21	5	5
54	56	8	5	1	1
51	54	43	49	8	8
57	58	26	26	11	11
57	59	28	23	2	2
56	60	21	22	6	6
49	60	44	43	9	9
55	61	7	10	1	1
55	62	13	6	3	3
63	64	22	30	11	11
63	65	48	50	6	6
56	62	24	26	8	8
58	61	17	16	4	4
58	66	62	61	17	17
61	67	26	27	9	9
62	67	20	16	1	1
65	67	15	10	1	1
66	67	15	15	1	1
68	69	12	10	1	1
48	64	10	10	3	3
48	70	1	6	1	1
64	70	5	7	1	1
61	62	16	15	7	7
51	52	16	17	4	4
65	66	19	11	4	4
52	53	10	5	1	1
71	72	11	18	3	3
71	73	25	18	7	7
74	75	19	16	4	4
74	76	23	22	2	2
74	77	27	26	2	2
78	79	9	7	4	4
78	80	10	13	5	5
78	81	26	26	12	12
78	82	10	13	2	2
75	81	19	21	1	1
75	83	7	2	1	1
75	77	38	33	6	6
79	82	22	15	3	3
84	85	9	11	3	3
84	86	10	10	1	1
82	84	18	13	3	3
84	87	33	27	11	11
80	88	13	20	4	4
85	86	1	6	1	1
85	87	23	15	7	7
83	88	38	40	1	1
73	88	28	25	1	1
77	88	23	17	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
82	86	23	21	7	7
86	87	21	19	4	4
89	90	3	5	1	1
72	91	22	30	6	6
72	73	13	11	3	3
72	92	24	25	9	9
93	94	3	5	2	2
92	93	16	15	2	2
87	95	1	1	1	1
91	95	35	37	7	7
77	92	17	22	8	8
1	6	21	16	5	5
1	70	16	10	3	3
3	68	7	5	3	3
3	69	21	13	3	3
3	93	37	36	15	15
4	7	12	9	1	1
4	53	21	15	5	5
4	68	11	13	4	4
5	15	31	31	1	1
6	15	48	53	14	14
7	15	31	32	6	6
7	49	28	30	14	14
8	17	42	39	15	15
8	18	32	34	14	14
8	20	30	29	9	9
9	34	77	77	16	16
9	36	80	79	7	7
11	47	138	140	43	43
11	89	211	216	42	42
12	13	9	8	4	4
12	14	16	14	5	5
14	47	144	137	14	14
14	89	207	217	35	35
15	21	47	39	18	18
15	32	45	50	20	20
15	34	57	62	15	15
15	45	128	131	54	54
16	89	251	259	98	98
17	20	19	22	10	10
17	21	28	23	8	8
21	32	51	54	23	23
21	34	70	72	35	35
22	35	13	16	6	6
22	36	11	11	2	2
24	34	33	37	6	6
24	45	45	45	19	19
26	31	14	6	1	1
28	40	8	3	1	1
29	30	31	23	6	6
30	31	17	23	2	2
31	44	14	14	3	3
32	34	74	69	12	12
32	38	16	11	1	1
32	42	73	74	32	32
33	38	25	26	10	10

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
33	76	23	14	7	7
38	94	40	33	7	7
40	41	18	23	8	8
42	76	39	38	3	3
42	90	115	112	49	49
43	76	26	27	1	1
46	60	31	38	4	4
47	63	16	17	5	5
47	66	71	71	13	13
49	51	9	16	1	1
50	59	17	18	7	7
52	59	19	21	3	3
53	68	5	15	1	1
54	59	23	26	1	1
55	56	19	18	2	2
57	85	47	41	7	7
57	91	21	21	5	5
58	95	37	28	8	8
60	65	28	26	2	2
63	89	185	180	70	70
69	71	18	21	2	2
69	93	28	20	7	7
71	94	24	33	4	4
73	91	30	32	4	4
79	80	23	22	2	2
79	81	23	19	1	1
80	81	6	16	3	3
83	90	59	61	9	9
90	95	64	61	20	20
92	94	19	18	8	8

instance hd915

numNodes 96

numArc 189

node1	node2	cost1	cost2	dh1	dh2
1	2	20	28	2	2
1	3	13	11	2	2
4	5	24	26	6	6
4	6	16	20	8	8
7	8	51	48	10	10
9	10	15	14	6	6
8	9	36	37	14	14
9	11	11	11	4	4
12	13	45	46	15	15
12	14	31	24	4	4
12	15	5	5	1	1
2	16	10	10	2	2
2	17	17	21	2	2
2	3	26	25	8	8
16	17	22	16	1	1
17	18	11	7	2	2
17	19	17	9	1	1
20	21	17	21	3	3
6	20	12	10	1	1
11	20	24	24	9	9
5	22	8	3	2	2
15	22	35	33	1	1
6	21	10	8	2	2
3	5	33	31	2	2
14	23	18	18	2	2
6	24	26	26	7	7
13	24	9	13	3	3
8	10	22	21	10	10
11	24	19	21	7	7
14	15	21	27	6	6
25	26	16	19	6	6
25	27	1	5	1	1
25	28	11	12	2	2
29	30	12	15	4	4
29	31	26	29	9	9
29	32	18	23	6	6
29	33	32	31	13	13
26	34	1	4	1	1
26	33	1	1	1	1
27	35	33	32	3	3
27	36	46	36	16	16
27	28	8	9	3	3
37	38	16	18	2	2
35	39	21	12	1	1
39	40	19	29	8	8
30	35	56	53	23	23
35	41	3	6	1	1
30	31	29	27	6	6
28	36	28	33	4	4
42	43	23	23	9	9
28	41	26	29	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
31	44	6	5	2	2
43	45	16	11	1	1
43	46	26	24	9	9
31	46	1	4	1	1
38	47	15	7	3	3
32	34	10	6	2	2
34	48	3	1	1	1
33	34	13	3	1	1
45	46	32	26	1	1
49	50	25	18	5	5
51	52	16	21	7	7
51	53	10	11	5	5
51	54	15	19	7	7
51	55	37	34	6	6
56	57	24	17	6	6
58	59	19	11	2	2
60	61	21	17	2	2
62	63	13	7	2	2
59	62	17	16	6	6
62	64	26	19	10	10
52	54	8	2	1	1
52	65	30	33	9	9
66	67	15	18	7	7
53	54	10	10	1	1
63	68	14	9	1	1
63	69	23	21	9	9
63	64	15	10	5	5
50	70	23	26	8	8
54	65	32	33	10	10
59	68	8	7	3	3
67	70	17	20	6	6
69	71	37	34	11	11
71	72	1	8	1	1
61	71	67	66	8	8
55	61	22	23	2	2
55	64	18	24	4	4
57	59	5	11	3	3
64	69	4	6	1	1
65	72	7	2	1	1
73	74	18	10	1	1
74	75	16	14	2	2
74	76	14	11	6	6
77	78	8	3	1	1
77	79	14	8	4	4
77	80	41	46	18	18
81	82	17	7	1	1
81	83	9	6	3	3
75	84	11	9	1	1
75	76	20	22	7	7
75	85	10	11	3	3
86	87	22	20	2	2
80	86	5	8	1	1
78	79	13	7	3	3
78	80	47	46	9	9
88	89	19	22	2	2
88	90	28	21	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
88	91	6	9	3	3
76	82	22	14	3	3
84	85	18	18	5	5
92	93	31	26	4	4
76	85	27	20	8	8
89	94	19	21	4	4
89	95	30	27	5	5
80	87	11	15	5	5
90	96	30	31	12	12
1	10	64	63	9	9
1	41	81	79	12	12
3	7	30	30	2	2
4	21	23	14	4	4
4	22	22	23	9	9
5	23	34	36	14	14
7	43	12	13	2	2
7	45	25	18	7	7
8	44	84	89	17	17
10	79	236	240	14	14
11	13	12	13	2	2
12	60	65	60	12	12
14	60	41	40	8	8
14	61	48	45	14	14
15	23	25	27	12	12
16	37	38	35	10	10
16	38	24	27	9	9
16	47	25	25	12	12
16	56	35	33	12	12
18	19	9	7	3	3
18	56	23	21	9	9
19	56	14	16	1	1
20	24	19	17	3	3
21	22	22	25	10	10
23	60	43	46	11	11
23	61	57	56	8	8
25	32	8	8	3	3
30	93	95	104	17	17
36	79	81	75	12	12
36	92	41	44	17	17
37	39	17	25	4	4
37	47	13	18	3	3
38	42	27	29	9	9
40	73	17	14	4	4
40	83	13	12	3	3
42	44	18	28	8	8
42	45	9	10	1	1
44	46	15	17	1	1
47	49	90	86	12	12
49	72	57	53	23	23
49	91	23	16	1	1
50	66	21	21	5	5
50	95	16	15	6	6
52	53	14	7	3	3
53	55	38	33	10	10
57	60	66	66	3	3
57	82	34	31	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
57	87	86	86	8	8
58	68	18	18	9	9
58	82	30	26	2	2
58	87	82	80	14	14
58	91	14	11	5	5
62	68	16	18	6	6
65	71	3	1	1	1
66	70	1	6	1	1
66	95	33	30	9	9
67	69	25	16	8	8
67	72	19	17	3	3
70	95	33	30	9	9
73	81	10	12	2	2
73	83	7	16	1	1
74	85	20	18	9	9
77	86	36	37	7	7
78	86	45	37	12	12
82	83	12	15	6	6
82	87	84	85	34	34
84	92	22	23	9	9
84	93	2	5	1	1
87	96	19	18	5	5
88	94	5	1	1	1
89	90	13	16	2	2
92	96	31	31	9	9
93	96	48	44	12	12

instance hg115

numNodes 60

numArc 105

node1	node2	cost1	cost2	dh1	dh2
1	2	4	1	1	1
3	4	1	1	1	1
4	5	6	3	1	1
6	7	1	1	1	1
5	8	2	1	1	1
5	9	5	1	1	1
10	11	2	6	1	1
12	13	6	5	1	1
14	15	6	3	2	2
14	16	1	1	1	1
17	18	6	1	1	1
11	18	1	1	1	1
11	19	4	1	1	1
11	20	4	1	1	1
13	21	1	1	1	1
22	23	3	2	1	1
24	25	1	5	1	1
26	27	1	2	1	1
25	28	5	1	1	1
29	30	1	5	1	1
31	32	6	2	1	1
31	33	1	3	1	1
32	34	1	1	1	1
35	36	1	6	1	1
36	37	4	1	1	1
36	38	3	1	1	1
39	40	6	1	1	1
39	41	1	4	1	1
42	43	5	5	3	3
44	45	4	1	1	1
41	46	4	1	1	1
47	48	6	2	1	1
49	50	1	1	1	1
51	52	6	6	1	1
43	53	1	6	1	1
43	54	1	1	1	1
55	56	1	1	1	1
57	58	1	4	1	1
48	50	2	1	1	1
50	59	2	5	1	1
59	60	5	5	1	1
1	3	1	5	1	1
2	8	1	1	1	1
2	12	1	1	1	1
3	8	5	1	1	1
4	6	1	1	1	1
5	15	4	2	1	1
7	9	1	1	1	1
7	10	9	1	1	1
7	17	5	3	1	1
7	40	3	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
8	14	1	2	1	1
9	22	3	1	1	1
12	14	2	1	1	1
13	16	1	1	1	1
15	17	9	6	2	2
15	22	6	4	1	1
15	27	2	1	1	1
16	22	7	4	2	2
16	26	4	1	1	1
17	22	1	1	1	1
17	28	2	7	1	1
18	24	1	4	1	1
19	30	6	7	2	2
20	24	4	1	1	1
20	29	1	1	1	1
21	26	3	1	1	1
21	31	4	1	1	1
22	24	9	8	2	2
23	27	1	1	1	1
23	28	1	7	1	1
23	39	7	1	1	1
25	29	1	6	1	1
25	35	1	1	1	1
26	32	2	4	1	1
28	40	2	1	1	1
28	49	1	1	1	1
29	36	1	1	1	1
30	37	1	1	1	1
32	35	11	8	4	4
33	34	1	5	1	1
33	44	1	1	1	1
34	39	1	1	1	1
34	55	2	2	1	1
35	40	2	2	1	1
35	51	1	1	1	1
37	42	5	5	1	1
38	40	3	4	1	1
38	42	1	1	1	1
38	53	1	1	1	1
40	46	4	3	2	2
41	44	4	1	1	1
41	57	2	1	1	1
45	55	1	6	1	1
46	47	1	1	1	1
47	49	2	4	1	1
48	57	1	1	1	1
48	60	4	1	1	1
49	51	1	1	1	1
50	52	1	5	1	1
51	53	4	1	1	1
52	54	1	1	1	1
55	57	1	1	1	1
56	58	1	6	1	1
58	60	6	2	1	1

instance hg215

numNodes 68

numArc 119

node1	node2	cost1	cost2	dh1	dh2
1	2	3	1	1	1
2	3	6	1	1	1
4	5	4	1	1	1
6	7	6	4	2	2
7	8	4	1	1	1
9	10	3	3	1	1
10	11	6	3	1	1
12	13	1	1	1	1
5	14	6	1	1	1
15	16	1	1	1	1
17	18	1	1	1	1
11	19	6	4	1	1
19	20	1	1	1	1
13	21	1	4	1	1
22	23	5	1	1	1
24	25	5	4	1	1
24	26	4	1	1	1
25	27	2	5	1	1
18	28	1	1	1	1
20	29	1	4	1	1
21	30	6	1	1	1
26	27	1	1	1	1
31	32	1	1	1	1
33	34	1	4	1	1
35	36	1	6	1	1
37	38	1	3	1	1
39	40	3	1	1	1
41	42	5	1	1	1
41	43	1	1	1	1
36	44	1	1	1	1
45	46	1	3	1	1
46	47	1	6	1	1
43	48	4	1	1	1
49	50	1	1	1	1
51	52	3	6	2	2
48	53	3	5	1	1
53	54	1	4	1	1
48	55	1	1	1	1
48	56	3	6	1	1
55	57	2	5	1	1
57	58	1	1	1	1
57	59	4	2	1	1
60	61	1	1	1	1
61	62	1	1	1	1
62	63	1	1	1	1
56	64	1	5	1	1
59	65	5	1	1	1
66	67	2	1	1	1
63	68	6	1	1	1
1	6	6	1	1	1
2	7	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
3	4	8	1	1	1
3	9	1	2	1	1
5	12	3	1	1	1
5	22	1	1	1	1
6	17	6	1	1	1
7	9	5	4	2	2
8	17	1	4	1	1
8	19	6	1	1	1
8	28	1	1	1	1
9	19	4	6	2	2
10	12	4	5	1	1
11	13	5	1	1	1
11	29	1	5	1	1
13	22	1	1	1	1
14	24	1	1	1	1
14	35	3	1	1	1
15	17	5	2	1	1
16	18	1	5	1	1
16	39	1	1	1	1
18	31	1	2	1	1
20	28	3	2	1	1
20	57	6	10	2	2
21	23	1	5	1	1
21	29	3	6	1	1
22	24	7	1	1	1
23	26	1	1	1	1
23	35	1	5	1	1
23	44	1	2	1	1
26	37	4	1	1	1
27	38	6	1	1	1
28	32	1	2	1	1
29	58	3	5	2	2
30	32	2	5	1	1
30	33	1	1	1	1
31	41	4	6	2	2
32	42	5	5	2	2
33	35	1	7	1	1
34	42	1	1	1	1
34	44	1	2	1	1
34	49	5	1	1	1
35	37	2	1	1	1
36	45	5	1	1	1
36	51	6	6	3	3
37	45	1	3	1	1
38	46	1	3	1	1
39	41	1	3	1	1
40	43	4	1	1	1
40	53	3	2	1	1
42	55	3	1	1	1
43	49	7	3	2	2
44	67	7	8	4	4
45	52	1	3	1	1
47	52	3	1	1	1
47	62	1	1	1	1
49	51	6	1	1	1
50	58	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
50	60	2	1	1	1
50	66	6	4	2	2
51	60	3	1	1	1
52	61	4	1	1	1
54	56	1	1	1	1
55	64	4	1	1	1
58	65	3	1	1	1
59	64	2	6	1	1
60	67	1	7	1	1
61	68	1	1	1	1
65	66	1	1	1	1
67	68	1	1	1	1

instance hg315

numNodes 64

numArc 116

node1	node2	cost1	cost2	dh1	dh2
1	2	1	2	1	1
3	4	3	3	1	1
5	6	1	4	1	1
6	7	6	1	1	1
7	8	1	1	1	1
2	4	2	4	1	1
4	9	1	1	1	1
10	11	1	1	1	1
12	13	6	1	1	1
9	14	1	1	1	1
15	16	5	1	1	1
16	17	1	1	1	1
13	18	1	1	1	1
18	19	3	3	2	2
14	20	1	1	1	1
14	21	1	3	1	1
21	22	4	2	1	1
17	23	6	1	1	1
17	24	4	1	1	1
25	26	1	5	1	1
27	28	2	4	1	1
29	30	1	4	1	1
26	31	5	1	1	1
32	33	1	1	1	1
32	34	2	3	1	1
35	36	2	5	1	1
37	38	4	1	1	1
39	40	1	1	1	1
41	42	3	2	1	1
43	44	2	6	1	1
44	45	1	1	1	1
46	47	4	1	1	1
45	48	6	4	2	2
48	49	1	1	1	1
47	50	1	1	1	1
51	52	1	5	1	1
53	54	1	1	1	1
53	55	1	6	1	1
54	56	5	1	1	1
57	58	1	1	1	1
59	60	5	3	1	1
61	62	1	1	1	1
52	63	4	1	1	1
58	64	2	1	1	1
1	3	1	1	1	1
2	20	1	1	1	1
3	5	1	3	1	1
4	10	1	8	1	1
5	10	7	4	2	2
5	16	6	3	1	1
6	12	5	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
7	18	6	6	2	2
8	19	2	7	1	1
9	11	1	4	1	1
10	12	3	1	1	1
11	15	1	5	1	1
11	22	1	3	1	1
11	30	1	7	1	1
13	16	6	6	3	3
13	25	4	5	2	2
14	33	1	2	1	1
15	23	1	1	1	1
17	25	3	1	1	1
18	31	1	1	1	1
19	27	6	1	1	1
20	32	1	1	1	1
21	35	1	1	1	1
22	23	5	7	2	2
22	29	1	2	1	1
23	30	5	5	2	2
23	59	2	1	1	1
24	26	1	5	1	1
24	30	4	2	1	1
24	37	1	1	1	1
25	27	1	1	1	1
26	39	1	4	1	1
28	31	4	1	1	1
28	41	3	1	1	1
29	36	6	5	1	1
30	49	4	4	1	1
31	40	1	1	1	1
33	35	2	1	1	1
33	43	1	1	1	1
34	43	4	5	2	2
34	53	3	4	1	1
35	44	2	6	1	1
36	37	1	7	1	1
36	48	1	2	1	1
37	39	1	1	1	1
38	44	3	1	1	1
38	46	5	1	1	1
38	49	8	6	3	3
38	50	6	2	1	1
38	59	1	6	1	1
39	50	5	3	2	2
40	41	1	4	1	1
40	46	1	1	1	1
42	46	5	1	1	1
42	51	1	1	1	1
43	54	6	1	1	1
45	56	1	6	1	1
47	51	6	4	1	1
47	61	1	1	1	1
48	57	5	1	1	1
49	50	4	1	1	1
49	59	1	1	1	1
49	64	1	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
50	59	6	6	2	2
52	61	1	6	1	1
55	58	4	4	1	1
56	57	6	1	1	1
57	59	5	1	1	1
59	61	5	5	1	1
60	62	1	1	1	1
60	64	1	1	1	1
62	63	5	5	3	3

instance hg415

numNodes 82

numArc 148

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
3	4	1	1	1	1
3	5	2	1	1	1
4	6	1	1	1	1
4	7	1	1	1	1
6	8	1	4	1	1
9	10	6	1	1	1
11	12	1	5	1	1
12	13	3	1	1	1
14	15	1	1	1	1
15	16	1	1	1	1
5	7	5	4	2	2
7	8	6	1	1	1
8	10	1	2	1	1
13	17	1	1	1	1
16	18	6	1	1	1
19	20	1	1	1	1
21	22	1	5	1	1
21	23	3	1	1	1
22	24	2	4	1	1
22	25	1	5	1	1
26	27	5	2	1	1
26	28	3	2	1	1
27	29	4	1	1	1
27	30	5	1	1	1
18	29	1	1	1	1
29	31	1	1	1	1
18	20	6	1	1	1
18	32	1	3	1	1
23	25	3	1	1	1
23	33	5	1	1	1
25	34	4	3	1	1
35	36	1	5	1	1
36	37	1	2	1	1
36	38	3	1	1	1
31	32	1	1	1	1
31	39	1	5	1	1
34	40	1	6	1	1
41	42	6	1	1	1
39	42	1	3	1	1
43	44	1	1	1	1
45	46	2	5	1	1
45	47	5	1	1	1
46	48	1	1	1	1
46	49	4	2	1	1
50	51	3	3	1	1
51	52	6	2	1	1
51	53	2	1	1	1
47	54	1	5	1	1
49	55	1	1	1	1
56	57	2	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
56	58	1	2	1	1
59	60	6	1	1	1
54	61	1	2	1	1
55	62	3	2	1	1
55	63	5	1	1	1
58	62	1	6	1	1
58	64	1	1	1	1
65	66	6	2	1	1
67	68	1	1	1	1
69	70	6	1	1	1
60	71	6	1	1	1
71	72	2	1	1	1
60	73	1	1	1	1
61	74	1	1	1	1
63	75	3	3	1	1
76	77	1	1	1	1
66	68	5	5	3	3
68	78	1	2	1	1
73	79	1	1	1	1
75	80	4	2	1	1
77	81	1	1	1	1
78	82	6	3	2	2
1	3	1	3	1	1
2	5	6	1	1	1
2	21	1	2	1	1
5	22	5	1	1	1
6	9	1	1	1	1
7	24	1	1	1	1
8	36	2	3	1	1
9	11	1	5	1	1
10	13	5	6	1	1
10	26	4	6	2	2
10	37	7	2	1	1
11	26	3	1	1	1
12	14	5	1	1	1
13	27	3	1	1	1
14	17	1	1	1	1
16	17	6	1	1	1
16	19	1	1	1	1
17	29	4	1	1	1
20	44	2	5	1	1
24	26	5	4	2	2
24	35	4	1	1	1
25	35	3	1	1	1
28	30	1	1	1	1
28	37	4	1	1	1
28	41	1	1	1	1
30	31	1	1	1	1
30	42	1	5	1	1
32	43	5	3	2	2
33	34	1	1	1	1
33	45	6	1	1	1
34	46	4	5	1	1
35	40	1	6	1	1
37	57	8	4	1	1
38	40	1	5	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
38	41	2	6	1	1
38	56	7	5	3	3
39	43	4	3	2	2
39	50	3	6	2	2
40	48	2	6	1	1
41	57	8	5	2	2
41	65	6	7	3	3
42	67	1	1	1	1
43	51	2	1	1	1
44	52	1	1	1	1
47	49	6	1	1	1
48	50	7	4	1	1
48	56	1	3	1	1
48	62	2	1	1	1
49	56	1	1	1	1
50	57	4	9	1	1
50	69	1	1	1	1
52	59	1	5	1	1
53	57	4	3	2	2
53	59	1	1	1	1
53	71	1	4	1	1
54	55	1	1	1	1
57	64	3	1	1	1
58	76	2	2	1	1
61	63	1	1	1	1
62	80	3	7	1	1
63	76	6	4	1	1
64	65	1	1	1	1
64	81	1	1	1	1
65	67	5	3	1	1
66	76	1	4	1	1
66	82	4	1	1	1
67	69	4	1	1	1
68	70	1	1	1	1
69	71	1	4	1	1
70	72	1	6	1	1
72	73	5	5	1	1
74	75	4	6	2	2
77	80	1	1	1	1
78	79	8	1	1	1
81	82	2	1	1	1

instance hg515

numNodes 92

numArc 165

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
3	4	1	1	1	1
4	5	1	1	1	1
6	7	6	1	1	1
8	9	4	5	2	2
10	11	4	3	1	1
2	12	1	2	1	1
12	13	1	5	1	1
12	14	1	1	1	1
13	15	1	1	1	1
16	17	3	4	2	2
17	18	1	1	1	1
7	19	1	4	1	1
9	20	3	5	1	1
21	22	5	1	1	1
14	23	5	5	2	2
23	24	1	1	1	1
14	15	3	1	1	1
14	25	6	1	1	1
18	26	1	1	1	1
26	27	3	1	1	1
19	28	2	4	1	1
19	29	1	1	1	1
28	30	6	6	3	3
28	31	1	1	1	1
22	32	5	6	2	2
32	33	1	6	1	1
22	34	1	1	1	1
25	35	3	2	1	1
25	36	1	1	1	1
27	37	6	4	2	2
37	38	3	4	2	2
29	39	1	1	1	1
31	40	6	3	1	1
41	42	3	1	1	1
33	34	1	1	1	1
36	43	2	5	1	1
36	44	1	1	1	1
39	40	6	1	1	1
45	46	3	2	1	1
47	48	1	5	1	1
47	49	1	1	1	1
48	50	2	1	1	1
51	52	2	1	1	1
53	54	1	2	1	1
54	55	1	5	1	1
56	57	1	1	1	1
58	59	1	1	1	1
60	61	1	1	1	1
60	62	1	4	1	1
55	63	1	6	1	1
63	64	1	4	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
55	65	1	3	1	1
65	66	1	2	1	1
66	67	1	4	1	1
62	68	4	1	1	1
69	70	4	1	1	1
71	72	4	2	1	1
64	73	3	3	1	1
74	75	1	1	1	1
67	75	1	6	1	1
67	76	1	1	1	1
68	77	1	1	1	1
77	78	5	1	1	1
68	79	1	6	1	1
68	80	4	1	1	1
79	81	1	1	1	1
81	82	3	6	1	1
81	83	1	1	1	1
84	85	5	3	2	2
84	86	4	1	1	1
76	85	4	6	2	2
76	87	1	3	1	1
80	88	1	1	1	1
83	89	5	1	1	1
90	91	5	1	1	1
87	92	1	2	1	1
1	3	2	1	1	1
2	23	1	1	1	1
3	12	1	5	1	1
4	13	6	4	2	2
5	6	1	1	1	1
5	16	5	6	3	3
6	8	1	5	1	1
7	9	2	1	1	1
7	17	1	1	1	1
8	10	1	1	1	1
9	28	1	5	1	1
10	20	2	5	1	1
11	21	1	1	1	1
11	32	3	5	1	1
13	16	2	1	1	1
15	26	2	1	1	1
15	35	1	2	1	1
16	26	1	6	1	1
18	19	1	6	1	1
18	37	1	6	1	1
20	21	1	1	1	1
20	30	3	3	1	1
24	25	2	1	1	1
24	43	1	1	1	1
27	35	1	6	1	1
27	47	1	1	1	1
29	31	6	6	2	2
29	37	1	2	1	1
30	32	1	3	1	1
30	41	1	6	1	1
31	41	6	1	1	1
33	41	1	1	1	1
33	56	1	6	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
34	57	1	7	1	1
35	44	6	5	1	1
36	45	2	4	1	1
38	39	1	1	1	1
38	44	5	5	2	2
38	48	5	3	1	1
39	51	6	1	1	1
40	42	5	6	2	2
40	53	1	5	1	1
42	54	1	1	1	1
43	58	3	1	1	1
44	46	1	1	1	1
45	60	3	5	2	2
46	47	3	1	1	1
46	61	1	5	1	1
48	51	4	1	1	1
49	50	1	5	1	1
49	61	1	1	1	1
49	70	5	1	1	1
50	52	1	1	1	1
50	82	1	1	1	1
51	53	1	4	1	1
52	63	1	2	1	1
52	71	2	1	1	1
53	63	1	2	1	1
54	56	5	1	1	1
55	74	6	1	1	1
56	65	1	1	1	1
57	66	1	1	1	1
58	60	6	4	1	1
59	62	4	4	1	1
59	77	3	1	1	1
61	69	1	1	1	1
62	69	1	2	1	1
64	71	1	1	1	1
64	74	4	1	1	1
65	75	1	2	1	1
69	79	1	4	1	1
70	71	7	3	2	2
70	81	6	6	2	2
72	73	1	1	1	1
72	82	1	3	1	1
72	90	1	1	1	1
73	84	4	5	2	2
73	91	4	1	1	1
74	84	1	1	1	1
75	85	6	1	1	1
78	80	1	3	1	1
79	88	2	5	1	1
82	89	1	3	1	1
83	88	1	1	1	1
85	92	5	6	1	1
86	91	1	1	1	1
86	92	1	3	1	1
89	90	1	1	1	1

instance hg615

numNodes 91

numArc 162

node1	node2	cost1	cost2	dh1	dh2
1	2	3	2	1	1
3	4	1	1	1	1
3	5	1	1	1	1
4	6	4	3	2	2
6	7	3	1	1	1
8	9	1	4	1	1
8	10	5	1	1	1
9	11	1	3	1	1
9	12	6	1	1	1
11	13	1	3	1	1
2	5	1	1	1	1
2	14	2	1	1	1
5	15	3	6	2	2
16	17	2	1	1	1
18	19	2	1	1	1
12	20	6	3	2	2
21	22	6	1	1	1
15	23	5	1	1	1
24	25	2	3	1	1
25	26	2	2	1	1
19	27	6	6	3	3
20	27	2	1	1	1
27	28	1	1	1	1
20	29	4	4	1	1
22	29	1	6	1	1
29	30	1	2	1	1
23	31	1	6	1	1
23	32	1	3	1	1
26	32	5	6	1	1
32	33	4	1	1	1
26	34	1	3	1	1
35	36	5	1	1	1
28	37	1	5	1	1
28	38	5	4	2	2
39	40	1	6	1	1
30	41	3	3	1	1
42	43	2	5	1	1
44	45	1	6	1	1
34	36	1	1	1	1
40	41	1	4	1	1
41	43	4	4	1	1
46	47	5	3	1	1
46	48	6	3	2	2
49	50	1	1	1	1
50	51	4	1	1	1
52	53	1	1	1	1
52	54	2	1	1	1
53	55	1	5	1	1
55	56	1	1	1	1
56	57	1	1	1	1
48	58	6	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
58	59	1	6	1	1
59	60	4	1	1	1
51	61	5	1	1	1
62	63	6	1	1	1
54	64	1	1	1	1
65	66	1	3	1	1
67	68	1	1	1	1
60	69	5	3	1	1
69	70	1	1	1	1
60	71	3	1	1	1
71	72	5	1	1	1
71	73	1	1	1	1
74	75	1	1	1	1
74	76	4	1	1	1
66	75	1	1	1	1
66	77	1	1	1	1
78	79	1	2	1	1
68	70	2	3	1	1
68	80	5	2	1	1
70	81	5	1	1	1
70	82	1	1	1	1
81	83	1	1	1	1
73	84	1	1	1	1
73	85	3	4	1	1
84	86	1	1	1	1
87	88	1	2	1	1
77	89	1	1	1	1
77	90	1	1	1	1
79	91	1	4	1	1
80	82	1	1	1	1
85	86	1	2	1	1
1	3	5	5	2	2
4	24	2	7	1	1
5	16	3	7	1	1
6	25	1	3	1	1
7	8	4	1	1	1
7	16	4	5	2	2
10	12	1	1	1	1
10	18	1	4	1	1
10	27	4	1	1	1
11	29	1	5	1	1
12	21	1	5	1	1
13	21	4	3	1	1
14	15	1	5	1	1
14	31	2	1	1	1
15	24	1	1	1	1
16	18	1	2	1	1
17	19	6	3	1	1
17	25	2	2	1	1
17	35	1	6	1	1
19	37	1	3	1	1
20	39	1	1	1	1
22	42	1	1	1	1
23	45	1	5	1	1
24	32	3	1	1	1
26	35	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
28	39	1	1	1	1
30	39	4	2	1	1
30	42	6	6	3	3
31	44	1	3	1	1
33	34	1	1	1	1
33	45	1	4	1	1
33	49	1	2	1	1
34	50	1	2	1	1
35	37	1	1	1	1
36	38	5	5	2	2
36	61	1	7	1	1
37	62	8	1	1	1
38	40	2	1	1	1
38	52	5	1	1	1
40	53	1	1	1	1
41	55	1	1	1	1
43	56	1	5	1	1
44	46	5	1	1	1
45	47	1	1	1	1
47	49	4	1	1	1
47	58	1	5	1	1
48	67	1	4	1	1
49	59	1	2	1	1
50	52	6	6	2	2
51	59	1	1	1	1
51	71	1	5	1	1
53	64	1	3	1	1
54	62	4	1	1	1
54	74	4	1	1	1
55	65	4	4	1	1
57	65	1	1	1	1
57	78	2	1	1	1
58	69	1	2	1	1
60	81	1	1	1	1
61	62	1	1	1	1
61	72	2	6	1	1
63	72	5	4	2	2
63	74	5	2	1	1
63	87	6	1	1	1
64	65	2	1	1	1
64	75	1	1	1	1
66	78	3	1	1	1
67	69	4	5	1	1
72	84	2	4	1	1
73	81	1	1	1	1
75	89	5	1	1	1
76	87	1	1	1	1
76	89	5	1	1	1
77	79	2	1	1	1
82	83	1	1	1	1
83	85	6	6	3	3
84	87	1	4	1	1
86	88	1	4	1	1
88	90	8	1	1	1
90	91	1	1	1	1

instance hg715

numNodes 97

numArc 174

node1	node2	cost1	cost2	dh1	dh2
1	2	1	6	1	1
1	3	1	1	1	1
2	4	1	1	1	1
2	5	1	1	1	1
4	6	1	1	1	1
4	7	1	1	1	1
6	8	1	1	1	1
9	10	4	2	1	1
11	12	1	1	1	1
11	13	1	4	1	1
12	14	6	1	1	1
12	15	4	5	1	1
14	16	1	1	1	1
16	17	5	1	1	1
16	18	1	4	1	1
3	5	1	5	1	1
5	7	1	1	1	1
7	8	5	1	1	1
7	19	6	2	1	1
8	10	1	2	1	1
8	20	3	2	1	1
15	21	1	2	1	1
21	22	1	2	1	1
18	23	4	5	1	1
18	24	1	6	1	1
23	25	2	2	1	1
26	27	1	1	1	1
26	28	2	6	1	1
19	27	5	4	1	1
27	29	1	1	1	1
19	30	1	1	1	1
31	32	1	1	1	1
31	33	2	1	1	1
22	32	1	6	1	1
32	34	1	2	1	1
22	24	1	5	1	1
22	35	6	2	1	1
24	25	2	1	1	1
24	36	1	1	1	1
25	37	1	5	1	1
28	29	1	5	1	1
28	38	2	6	1	1
29	39	2	1	1	1
30	40	1	6	1	1
30	41	1	1	1	1
40	42	2	1	1	1
40	43	1	6	1	1
42	44	3	1	1	1
35	36	4	1	1	1
35	45	1	5	1	1
36	37	2	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
36	46	1	6	1	1
37	47	1	1	1	1
38	39	6	1	1	1
39	41	1	1	1	1
48	49	1	2	1	1
45	49	6	1	1	1
45	46	1	2	1	1
46	47	1	2	1	1
50	51	5	1	1	1
50	52	1	4	1	1
51	53	1	1	1	1
51	54	2	1	1	1
55	56	4	5	2	2
57	58	1	6	1	1
59	60	4	1	1	1
60	61	3	2	1	1
60	62	4	1	1	1
61	63	1	1	1	1
61	64	1	1	1	1
52	54	2	1	1	1
52	65	1	1	1	1
54	66	5	5	2	2
54	67	1	2	1	1
56	68	1	5	1	1
56	69	1	2	1	1
58	70	1	1	1	1
62	64	1	5	1	1
62	71	1	2	1	1
64	72	1	2	1	1
73	74	1	2	1	1
65	67	1	5	1	1
65	75	1	1	1	1
67	76	1	1	1	1
67	77	1	1	1	1
69	76	3	1	1	1
69	78	2	1	1	1
69	79	2	5	1	1
78	80	4	1	1	1
70	81	6	1	1	1
71	82	6	1	1	1
82	83	6	1	1	1
71	84	1	1	1	1
72	74	2	1	1	1
72	85	2	1	1	1
74	86	1	6	1	1
75	77	1	1	1	1
75	87	2	5	1	1
77	88	1	1	1	1
89	90	1	5	1	1
79	80	1	1	1	1
79	91	4	1	1	1
81	83	1	1	1	1
81	92	1	1	1	1
83	93	5	1	1	1
84	85	1	5	1	1
84	94	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
86	95	4	1	1	1
87	88	6	2	1	1
88	90	1	3	1	1
91	96	1	1	1	1
92	93	1	5	1	1
94	97	1	1	1	1
3	26	1	1	1	1
5	27	2	2	1	1
6	9	3	1	1	1
9	11	1	1	1	1
10	13	2	6	1	1
10	42	1	3	1	1
13	15	3	1	1	1
13	31	3	1	1	1
14	21	1	5	1	1
15	32	1	1	1	1
17	23	5	1	1	1
18	21	1	1	1	1
19	20	3	1	1	1
20	31	1	1	1	1
20	40	5	5	3	3
29	30	5	1	1	1
33	34	1	1	1	1
33	42	1	1	1	1
33	48	5	3	1	1
34	35	6	1	1	1
34	49	1	1	1	1
38	50	3	4	1	1
39	51	1	1	1	1
41	43	5	1	1	1
41	53	6	4	2	2
43	44	2	3	1	1
43	55	1	1	1	1
44	48	1	1	1	1
44	68	1	1	1	1
45	60	3	1	1	1
46	61	1	1	1	1
47	63	2	1	1	1
48	57	5	2	1	1
49	59	4	1	1	1
53	55	2	1	1	1
53	66	1	1	1	1
55	57	6	7	3	3
56	66	2	1	1	1
57	59	1	6	1	1
58	62	4	2	1	1
58	68	1	1	1	1
59	82	2	1	1	1
63	73	1	2	1	1
64	73	6	1	1	1
66	76	5	6	1	1
68	78	1	3	1	1
70	78	2	4	1	1
70	82	1	4	1	1
71	72	1	6	1	1
76	89	1	6	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
77	89	6	6	2	2
79	89	1	3	1	1
80	81	1	1	1	1
80	96	1	5	1	1
83	84	5	6	3	3
85	86	1	2	1	1
85	97	1	1	1	1
90	91	2	1	1	1
92	96	1	2	1	1
93	94	6	2	1	1
95	97	1	1	1	1

instance hg815

numNodes 100

numArc 180

node1	node2	cost1	cost2	dh1	dh2
1	2	5	1	1	1
3	4	6	2	1	1
4	5	1	1	1	1
6	7	1	1	1	1
8	9	6	1	1	1
10	11	1	1	1	1
12	13	2	1	1	1
14	15	1	6	1	1
2	16	1	1	1	1
16	17	1	1	1	1
16	18	1	1	1	1
17	19	5	1	1	1
7	20	4	4	1	1
20	21	2	2	1	1
7	22	3	1	1	1
9	23	3	2	1	1
11	24	5	1	1	1
11	25	1	4	1	1
15	26	2	1	1	1
26	27	1	1	1	1
15	28	1	1	1	1
18	29	3	3	1	1
29	30	4	1	1	1
18	19	1	3	1	1
18	31	1	1	1	1
19	32	1	1	1	1
21	22	1	2	1	1
21	33	1	1	1	1
23	25	1	5	1	1
23	34	4	5	1	1
25	35	1	6	1	1
25	36	5	1	1	1
27	28	1	1	1	1
27	37	6	1	1	1
28	38	1	1	1	1
31	32	6	1	1	1
31	39	6	6	3	3
32	33	1	6	1	1
33	40	1	5	1	1
40	41	1	1	1	1
34	42	1	1	1	1
36	43	2	1	1	1
37	44	1	1	1	1
44	45	1	1	1	1
37	38	1	1	1	1
39	46	5	4	1	1
39	47	1	1	1	1
47	48	1	6	1	1
42	43	2	3	1	1
49	50	6	1	1	1
51	52	1	6	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
51	53	4	1	1	1
52	54	1	1	1	1
54	55	6	4	2	2
54	56	4	4	2	2
55	57	1	1	1	1
55	58	1	5	1	1
57	59	3	1	1	1
60	61	3	1	1	1
62	63	4	1	1	1
63	64	4	1	1	1
65	66	2	1	1	1
65	67	4	1	1	1
66	68	1	1	1	1
53	69	1	2	1	1
56	70	1	3	1	1
70	71	3	1	1	1
56	72	6	4	2	2
58	59	1	1	1	1
58	73	1	1	1	1
64	74	5	1	1	1
74	75	1	1	1	1
64	67	1	4	1	1
67	68	4	1	1	1
68	76	6	3	1	1
69	77	4	6	2	2
71	72	1	1	1	1
71	78	4	1	1	1
72	73	1	1	1	1
72	79	6	4	2	2
73	80	1	5	1	1
80	81	1	1	1	1
75	82	5	1	1	1
82	83	1	1	1	1
75	84	6	2	1	1
85	86	1	4	1	1
76	86	4	4	1	1
76	87	1	1	1	1
77	78	1	1	1	1
77	88	1	1	1	1
78	79	2	5	1	1
78	89	1	1	1	1
79	90	1	1	1	1
81	90	4	1	1	1
90	91	1	2	1	1
81	92	5	1	1	1
83	84	1	4	1	1
83	93	5	6	1	1
94	95	6	5	1	1
94	96	2	2	1	1
87	95	1	1	1	1
95	97	1	1	1	1
87	98	1	1	1	1
89	99	6	1	1	1
91	92	4	1	1	1
93	100	1	4	1	1
97	98	1	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
1	3	1	3	1	1
2	29	3	1	1	1
3	16	5	1	1	1
4	17	1	1	1	1
5	6	3	1	1	1
5	20	1	5	1	1
6	8	1	1	1	1
7	9	1	5	1	1
8	10	1	1	1	1
9	11	1	5	1	1
10	12	1	4	1	1
12	24	1	1	1	1
13	14	4	1	1	1
13	26	2	1	1	1
17	20	6	2	1	1
19	21	3	1	1	1
22	23	1	6	1	1
22	40	1	1	1	1
24	26	1	1	1	1
24	35	5	3	2	2
27	35	1	1	1	1
30	31	5	1	1	1
30	46	4	1	1	1
32	47	1	1	1	1
33	48	1	1	1	1
34	36	5	1	1	1
34	40	5	1	1	1
35	44	1	1	1	1
36	44	5	3	1	1
37	49	1	1	1	1
38	50	1	1	1	1
39	52	2	1	1	1
41	42	1	1	1	1
41	48	1	1	1	1
41	57	6	3	2	2
42	60	1	6	1	1
43	45	1	1	1	1
43	62	3	3	1	1
45	49	1	1	1	1
45	63	1	1	1	1
46	51	1	3	1	1
47	54	1	1	1	1
48	55	1	2	1	1
49	65	1	1	1	1
50	66	5	1	1	1
52	70	1	1	1	1
53	70	6	2	1	1
56	58	6	2	1	1
57	60	5	5	1	1
59	61	1	1	1	1
59	80	2	4	1	1
60	62	3	1	1	1
61	74	1	5	1	1
61	82	1	1	1	1
62	74	1	1	1	1
63	65	1	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
64	85	3	1	1	1
67	86	1	1	1	1
69	71	1	1	1	1
73	90	1	1	1	1
75	85	1	5	1	1
79	99	2	1	1	1
80	82	3	3	1	1
81	83	3	4	2	2
84	94	3	1	1	1
84	100	1	1	1	1
85	94	1	3	1	1
86	95	4	1	1	1
88	89	1	4	1	1
91	99	1	5	1	1
92	93	1	3	1	1
96	97	5	3	1	1
96	100	1	1	1	1

instance hg915

numNodes 97

numArc 175

node1	node2	cost1	cost2	dh1	dh2
1	2	1	3	1	1
3	4	1	1	1	1
3	5	1	5	1	1
4	6	4	3	1	1
4	7	3	4	2	2
6	8	1	1	1	1
6	9	5	1	1	1
10	11	1	4	1	1
10	12	1	3	1	1
11	13	5	1	1	1
11	14	1	2	1	1
13	15	2	3	1	1
13	16	3	5	1	1
2	5	1	1	1	1
2	17	2	6	1	1
5	7	3	4	1	1
5	18	1	1	1	1
7	19	1	2	1	1
20	21	6	1	1	1
22	23	4	1	1	1
12	24	2	1	1	1
14	16	2	5	1	1
14	25	1	1	1	1
26	27	1	1	1	1
17	18	1	2	1	1
18	19	3	6	1	1
18	28	1	5	1	1
19	29	1	6	1	1
29	30	4	5	2	2
23	24	2	1	1	1
23	31	3	4	2	2
24	25	1	1	1	1
24	32	1	1	1	1
25	33	1	1	1	1
27	33	1	1	1	1
33	34	1	4	1	1
28	35	3	1	1	1
28	36	4	3	2	2
28	37	1	2	1	1
30	36	1	1	1	1
36	38	3	1	1	1
30	39	1	6	1	1
40	41	1	4	1	1
31	32	1	1	1	1
32	42	1	1	1	1
32	43	1	1	1	1
42	44	6	6	2	2
34	45	1	3	1	1
34	46	1	1	1	1
45	47	1	1	1	1
37	48	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
37	38	5	5	3	3
39	41	1	1	1	1
44	46	6	1	1	1
46	47	1	1	1	1
49	50	2	1	1	1
49	51	1	5	1	1
52	53	1	4	1	1
53	54	1	2	1	1
55	56	1	1	1	1
57	58	1	1	1	1
57	59	3	1	1	1
58	60	1	5	1	1
58	61	1	1	1	1
60	62	4	5	2	2
60	63	1	5	1	1
62	64	2	1	1	1
51	65	1	4	1	1
65	66	1	1	1	1
66	67	4	1	1	1
54	68	5	3	1	1
54	69	5	1	1	1
68	70	3	1	1	1
56	71	5	1	1	1
59	61	3	2	1	1
61	63	5	4	2	2
63	64	1	1	1	1
63	72	3	1	1	1
73	74	3	1	1	1
67	75	6	6	2	2
75	76	6	3	2	2
67	69	1	3	1	1
69	70	1	4	1	1
69	77	1	1	1	1
70	78	5	3	1	1
71	79	4	1	1	1
71	80	2	1	1	1
79	81	1	5	1	1
79	82	1	2	1	1
72	81	6	1	1	1
72	83	6	1	1	1
84	85	1	1	1	1
74	76	3	1	1	1
74	86	1	1	1	1
76	87	1	1	1	1
76	88	4	1	1	1
87	89	4	3	1	1
77	78	1	1	1	1
77	90	1	4	1	1
78	91	1	1	1	1
80	82	1	4	1	1
80	92	1	2	1	1
83	93	1	3	1	1
83	94	1	1	1	1
85	95	3	1	1	1
86	88	1	1	1	1
90	91	1	5	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
92	96	1	5	1	1
96	97	1	1	1	1
1	3	5	4	1	1
7	9	1	1	1	1
8	10	2	6	1	1
8	20	1	5	1	1
9	20	5	1	1	1
9	29	1	1	1	1
12	14	1	4	1	1
12	22	3	1	1	1
15	26	4	1	1	1
16	26	1	1	1	1
16	33	5	1	1	1
17	35	2	1	1	1
19	36	1	3	1	1
20	22	1	5	1	1
21	23	6	1	1	1
21	29	1	4	1	1
21	40	6	4	1	1
25	42	3	1	1	1
27	45	1	2	1	1
30	40	5	1	1	1
31	40	1	6	1	1
31	55	1	1	1	1
34	42	5	1	1	1
35	48	1	1	1	1
37	50	5	5	1	1
38	39	4	1	1	1
38	52	1	6	1	1
39	53	5	3	1	1
41	43	7	1	1	1
41	55	1	6	1	1
41	68	1	1	1	1
43	44	1	1	1	1
43	57	1	5	1	1
44	58	1	1	1	1
46	60	4	1	1	1
47	62	5	1	1	1
48	49	1	3	1	1
50	52	1	1	1	1
50	65	1	1	1	1
51	73	1	3	1	1
52	66	1	2	1	1
53	55	4	1	1	1
54	66	5	1	1	1
55	57	4	5	1	1
56	59	6	1	1	1
56	68	1	1	1	1
59	79	3	1	1	1
61	81	1	3	1	1
64	84	1	2	1	1
65	75	1	1	1	1
67	87	1	1	1	1
70	71	1	1	1	1
72	84	4	5	1	1
73	75	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
77	87	4	1	1	1
78	80	1	1	1	1
81	93	1	6	1	1
82	93	1	1	1	1
82	96	1	2	1	1
83	85	4	6	2	2
88	89	1	2	1	1
89	90	1	5	1	1
91	92	4	1	1	1
93	97	1	3	1	1
94	95	1	1	1	1
94	97	1	3	1	1

instance M3101

numNodes 196

numArc 316

node1	node2	cost1	cost2	dh1	dh2
1	2	37	45	18	18
2	25	49	53	7	7
3	4	70	72	20	20
4	5	60	63	13	13
5	6	57	58	12	12
7	15	63	62	9	9
7	193	80	76	31	31
8	194	82	83	31	31
9	196	55	47	17	17
10	11	46	52	4	4
10	196	45	50	16	16
12	194	60	63	27	27
13	194	36	35	12	12
14	39	72	71	29	29
14	194	36	31	6	6
15	16	100	103	34	34
16	191	35	30	15	15
17	35	135	130	16	16
18	19	38	43	18	18
20	21	61	65	4	4
20	22	38	40	13	13
21	23	63	57	11	11
21	34	75	74	12	12
23	24	48	54	20	20
23	33	75	72	5	5
25	26	44	48	9	9
26	27	53	50	5	5
26	30	148	145	28	28
27	28	140	133	34	34
29	71	228	221	82	82
30	60	69	68	2	2
31	32	49	47	1	1
32	59	42	44	17	17
34	36	71	73	16	16
34	58	41	47	11	11
34	192	35	42	2	2
37	54	61	63	27	27
38	39	77	74	13	13
38	50	115	113	43	43
40	41	40	40	20	20
42	44	33	35	13	13
43	45	30	37	9	9
44	45	29	30	6	6
46	48	90	85	26	26
47	87	95	100	15	15
48	49	45	43	1	1
51	52	44	42	3	3
52	55	44	45	18	18
53	54	31	31	8	8
55	56	77	74	18	18
56	57	39	40	4	4

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
60	66	58	52	3	3
60	67	68	70	9	9
61	65	55	54	11	11
62	63	59	65	7	7
63	64	42	45	19	19
63	74	121	129	10	10
64	189	65	61	12	12
66	76	120	115	51	51
68	69	46	49	3	3
69	70	94	89	10	10
69	79	70	68	7	7
70	80	48	47	1	1
71	72	42	45	15	15
72	73	45	45	13	13
73	74	38	38	16	16
74	100	48	47	5	5
75	189	63	65	31	31
77	98	69	75	10	10
78	79	40	36	6	6
81	82	54	45	23	23
81	190	41	44	9	9
82	83	45	41	14	14
82	91	161	163	72	72
83	84	35	42	15	15
85	86	84	81	23	23
88	89	199	195	36	36
88	118	130	136	59	59
89	90	48	51	3	3
89	117	146	145	4	4
91	93	57	58	15	15
92	95	55	55	24	24
94	114	117	114	42	42
96	190	90	88	6	6
97	98	112	105	10	10
97	109	46	45	20	20
98	188	33	35	10	10
99	106	37	31	2	2
101	102	286	286	26	26
102	103	49	53	10	10
102	129	92	87	44	44
104	105	85	83	33	33
106	127	149	147	1	1
107	126	135	135	50	50
107	188	36	37	11	11
108	125	107	107	2	2
110	111	97	100	7	7
111	112	58	55	2	2
111	113	62	60	13	13
113	120	97	104	33	33
114	115	40	47	2	2
115	119	149	147	46	46
116	117	43	41	17	17
120	121	134	134	15	15
120	164	63	55	9	9
121	122	51	46	2	2
122	123	28	29	8	8

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
123	124	54	46	21	21
123	145	101	103	45	45
125	126	82	84	29	29
125	186	45	48	3	3
126	144	55	59	9	9
127	143	66	70	24	24
128	135	135	127	40	40
128	137	58	60	15	15
129	187	72	76	16	16
130	131	215	212	51	51
130	132	114	105	34	34
132	133	106	104	8	8
134	136	81	84	4	4
138	142	126	132	5	5
139	140	48	41	3	3
140	141	49	53	13	13
142	143	46	50	10	10
143	147	43	41	7	7
146	147	80	76	27	27
148	149	44	45	16	16
150	154	39	36	6	6
151	184	90	89	14	14
152	156	71	75	24	24
153	156	119	115	27	27
154	155	69	75	20	20
156	184	40	30	10	10
157	177	110	118	26	26
158	159	46	45	19	19
159	160	70	75	28	28
161	162	41	38	18	18
161	174	130	130	48	48
163	164	138	138	55	55
164	165	68	69	16	16
166	167	93	94	45	45
167	168	146	153	44	44
168	169	59	53	21	21
169	170	139	142	32	32
169	185	40	37	9	9
170	171	40	32	1	1
172	173	50	55	5	5
175	176	50	54	2	2
176	179	133	131	50	50
177	178	95	97	41	41
180	181	62	68	17	17
180	182	204	199	21	21
181	183	193	190	18	18
194	195	79	74	32	32
1	26	49	52	8	8
2	3	45	49	19	19
3	22	36	40	14	14
3	24	64	69	13	13
4	20	51	58	21	21
5	19	121	129	27	27
6	18	85	88	40	40
6	193	62	56	1	1
7	8	112	105	24	24

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
8	15	81	87	3	3
9	10	24	20	2	2
9	195	66	68	1	1
10	45	29	35	12	12
11	12	35	31	11	11
11	43	33	33	3	3
12	13	41	35	7	7
12	41	62	60	17	17
12	195	44	44	9	9
13	14	32	40	14	14
13	40	59	56	1	1
14	15	60	66	8	8
16	17	41	38	16	16
17	18	52	49	15	15
17	193	85	89	13	13
18	35	63	65	8	8
19	21	52	49	20	20
19	192	42	43	19	19
22	23	61	65	18	18
24	25	54	54	20	20
24	33	55	55	23	23
25	31	123	129	7	7
28	29	52	53	11	11
28	30	105	101	2	2
29	30	95	103	6	6
29	62	46	51	15	15
30	31	40	44	3	3
30	61	45	54	20	20
32	33	39	31	13	13
35	192	31	38	9	9
37	38	43	44	11	11
38	53	68	68	6	6
39	40	53	50	11	11
41	42	30	27	6	6
41	50	64	69	27	27
42	43	26	27	12	12
44	46	35	42	17	17
44	49	80	81	18	18
46	196	76	71	34	34
47	48	68	71	14	14
47	51	112	110	22	22
49	50	50	49	22	22
50	52	50	53	12	12
51	84	78	78	14	14
52	53	100	101	39	39
53	56	48	41	9	9
54	57	46	47	3	3
55	83	15	16	5	5
56	81	92	93	25	25
57	58	118	120	2	2
57	70	69	61	28	28
58	68	45	53	2	2
59	60	37	35	10	10
61	62	101	99	25	25
64	65	56	61	10	10
65	66	50	45	9	9

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
66	67	55	48	6	6
67	68	118	119	50	50
67	78	142	138	32	32
71	102	91	95	14	14
72	103	90	84	27	27
73	104	92	85	41	41
74	75	52	55	1	1
75	76	86	85	7	7
76	77	82	78	23	23
76	99	60	62	3	3
77	78	94	90	16	16
78	97	121	125	17	17
79	80	92	87	30	30
79	110	192	196	44	44
80	81	45	44	14	14
80	190	36	43	6	6
82	92	105	102	35	35
82	190	72	72	9	9
84	85	39	43	6	6
85	87	116	111	39	39
87	90	40	30	9	9
90	91	47	46	7	7
91	92	143	139	7	7
93	94	107	103	50	50
93	115	139	137	58	58
94	95	87	86	41	41
95	96	53	53	6	6
95	112	125	130	5	5
96	110	130	122	18	18
99	100	127	130	14	14
100	105	65	68	8	8
101	130	174	165	50	50
103	104	35	39	16	16
104	129	123	121	53	53
105	106	143	149	66	66
105	127	143	139	46	46
105	128	69	70	22	22
106	188	38	43	5	5
107	108	99	96	39	39
108	110	65	61	30	30
110	124	76	74	16	16
111	124	104	103	44	44
112	113	42	43	19	19
113	114	79	73	22	22
113	122	130	129	8	8
115	116	85	88	41	41
116	119	77	70	29	29
117	118	109	109	41	41
121	163	53	51	19	19
122	153	86	84	18	18
124	186	47	52	2	2
125	145	45	51	10	10
126	127	75	84	27	27
129	135	87	90	41	41
130	187	126	126	53	53
133	134	68	71	6	6

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
133	135	68	69	26	26
133	187	85	81	18	18
135	136	35	43	10	10
136	137	86	83	1	1
136	139	66	67	31	31
137	138	41	38	17	17
137	139	82	87	18	18
138	141	95	91	10	10
140	148	100	101	35	35
141	142	72	71	7	7
142	146	46	50	18	18
143	144	90	87	41	41
144	145	88	86	25	25
144	152	80	71	5	5
146	149	58	54	10	10
147	151	69	64	27	27
148	157	119	115	10	10
149	150	23	21	4	4
150	151	80	81	34	34
151	155	45	53	1	1
152	153	117	117	22	22
154	158	87	91	32	32
155	160	65	62	18	18
156	161	46	41	21	21
157	158	68	66	26	26
159	174	26	25	3	3
159	175	90	85	41	41
160	184	54	51	18	18
161	170	37	38	3	3
162	163	129	131	32	32
162	168	111	110	38	38
163	168	53	45	22	22
164	167	71	77	30	30
165	166	73	73	3	3
173	174	26	28	9	9
173	176	83	84	17	17
173	182	52	56	22	22
175	177	101	101	18	18
178	179	48	46	23	23
179	180	55	58	27	27
182	183	58	63	13	13

instance p0115

numNodes 11

numArc 15

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
3	4	1	4	1	1
3	5	1	1	1	1
5	6	1	1	1	1
7	8	4	7	1	1
9	10	1	1	1	1
10	11	4	1	1	1
1	10	3	3	1	1
2	3	8	6	3	3
4	5	1	1	1	1
4	8	1	4	1	1
5	11	18	19	4	4
6	7	9	8	1	1
1	6	3	3	3	3
1	4	3	3	3	3

instance P1515

numNodes 26

numArc 37

node1	node2	cost1	cost2	dh1	dh2
1	2	2	3	1	1
3	4	6	5	3	3
4	5	1	1	1	1
6	7	14	14	6	6
7	8	8	9	2	2
8	9	10	12	4	4
10	12	2	6	1	1
11	12	16	20	4	4
12	13	1	2	1	1
14	15	14	11	4	4
16	17	4	8	2	2
17	18	1	4	1	1
19	20	8	9	1	1
20	21	10	11	5	5
20	22	9	5	3	3
22	23	5	8	2	2
24	25	2	1	1	1
24	26	17	17	8	8
25	26	7	7	3	3
1	4	15	14	3	3
1	25	4	6	1	1
4	16	31	30	8	8
4	26	22	24	10	10
6	9	1	1	1	1
6	10	34	36	6	6
6	16	28	27	2	2
7	9	10	5	3	3
10	14	14	13	3	3
10	16	21	25	1	1
11	15	21	20	10	10
13	21	19	19	6	6
15	19	3	2	1	1
16	26	20	22	7	7
17	24	6	9	3	3
18	24	8	9	2	2
19	23	15	16	7	7
21	22	18	17	7	7

instance P0215

numNodes 14

numArc 33

node1	node2	cost1	cost2	dh1	dh2
1	3	4	1	1	1
2	3	8	4	1	1
3	4	9	10	2	2
5	6	3	7	1	1
6	7	1	1	1	1
5	7	10	7	3	3
8	9	1	1	1	1
9	10	5	7	2	2
10	11	1	1	1	1
11	12	2	2	1	1
9	12	9	10	1	1
13	14	1	1	1	1
1	4	4	7	1	1
1	5	15	16	3	3
1	7	19	18	7	7
1	8	24	22	3	3
1	10	20	20	5	5
1	13	8	10	1	1
2	5	14	13	6	6
2	7	20	16	7	7
2	8	27	25	7	7
2	10	24	21	4	4
3	5	13	13	2	2
3	7	21	22	2	2
3	8	26	26	8	8
3	10	21	26	11	11
5	10	18	16	2	2
5	13	22	22	7	7
7	8	9	7	1	1
7	10	12	9	5	5
10	12	2	1	1	1
10	13	5	5	3	3
10	14	6	10	3	3

instance P0315

numNodes 28

numArc 58

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	24	3	2	1	1
2	3	1	1	1	1
2	4	1	1	1	1
4	5	1	1	1	1
6	7	1	1	1	1
6	8	1	3	1	1
7	9	1	2	1	1
8	10	1	4	1	1
9	11	2	3	1	1
10	12	1	1	1	1
11	13	4	2	1	1
12	14	1	1	1	1
15	16	2	1	1	1
15	21	1	1	1	1
16	17	1	4	1	1
18	19	1	1	1	1
19	20	1	1	1	1
20	21	1	1	1	1
22	27	1	2	1	1
22	28	1	1	1	1
23	28	1	1	1	1
24	28	1	1	1	1
24	25	1	1	1	1
25	28	1	1	1	1
26	28	1	2	1	1
27	28	1	1	1	1
1	3	1	1	1	1
3	4	1	2	1	1
3	22	2	1	1	1
4	22	1	1	1	1
5	6	4	1	1	1
5	7	1	1	1	1
5	27	3	1	1	1
7	8	2	1	1	1
7	27	4	1	1	1
9	10	2	1	1	1
11	12	1	1	1	1
13	14	1	1	1	1
13	15	1	1	1	1
13	24	1	5	1	1
13	25	4	5	2	2
15	17	7	6	3	3
15	18	4	3	1	1
15	20	1	1	1	1
15	24	4	1	1	1
15	25	1	4	1	1
16	18	6	3	1	1
16	20	3	1	1	1
16	21	2	2	1	1
17	18	4	3	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
17	20	4	3	1	1
17	21	1	1	1	1
18	21	2	2	1	1
22	23	1	1	1	1
23	24	2	1	1	1
25	26	1	2	1	1
26	27	1	1	1	1

instance P0415

numNodes 17

numArc 35

node1	node2	cost1	cost2	dh1	dh2
1	2	3	1	1	1
1	4	1	3	1	1
2	3	1	1	1	1
2	4	1	1	1	1
3	4	1	1	1	1
4	5	2	1	1	1
6	7	1	2	1	1
6	8	2	2	1	1
7	8	1	1	1	1
7	10	1	1	1	1
7	12	1	3	1	1
9	10	1	1	1	1
9	12	1	1	1	1
11	12	1	1	1	1
13	14	1	1	1	1
13	17	1	1	1	1
14	15	1	1	1	1
14	16	4	2	1	1
14	17	1	3	1	1
15	17	1	1	1	1
15	16	1	1	1	1
16	17	2	1	1	1
1	16	7	7	3	3
3	6	1	5	1	1
4	17	4	4	1	1
5	6	2	1	1	1
5	8	1	1	1	1
5	13	11	8	3	3
5	17	7	4	2	2
6	13	8	5	1	1
7	9	1	1	1	1
8	10	1	1	1	1
9	11	1	1	1	1
10	13	4	3	2	2
12	13	2	6	1	1

instance P0515

numNodes 20

numArc 35

node1	node2	cost1	cost2	dh1	dh2
1	2	3	3	2	2
1	6	2	4	1	1
2	3	2	1	1	1
3	4	1	1	1	1
4	5	1	1	1	1
5	6	1	1	1	1
7	8	3	1	1	1
8	9	4	7	2	2
8	10	1	1	1	1
10	11	1	1	1	1
12	13	2	1	1	1
14	15	1	1	1	1
15	16	8	7	1	1
17	18	1	3	1	1
18	19	2	3	1	1
19	20	6	10	2	2
2	5	1	1	1	1
4	7	2	2	1	1
4	12	14	19	6	6
4	14	14	13	3	3
5	16	4	3	1	1
5	20	9	7	1	1
7	11	3	3	1	1
7	14	12	15	2	2
9	10	1	1	1	1
9	12	7	4	2	2
9	14	14	12	2	2
9	16	20	22	1	1
12	14	5	8	2	2
12	16	15	17	4	4
13	16	14	13	2	2
13	17	6	5	2	2
16	17	7	9	2	2
16	20	4	3	1	1
18	20	1	1	1	1

instance P0615

numNodes 24

numArc 46

node1	node2	cost1	cost2	dh1	dh2
1	2	2	1	1	1
2	3	4	2	1	1
4	5	1	1	1	1
6	7	3	1	1	1
7	8	3	3	1	1
7	9	1	2	1	1
8	9	1	5	1	1
10	11	2	1	1	1
12	13	1	1	1	1
12	14	3	1	1	1
13	14	1	1	1	1
15	16	2	4	1	1
15	17	1	1	1	1
15	19	1	1	1	1
16	17	1	1	1	1
18	19	1	1	1	1
20	22	4	1	1	1
21	22	1	1	1	1
22	23	2	6	1	1
22	24	3	4	1	1
1	3	4	3	2	2
1	18	5	1	1	1
3	4	1	1	1	1
3	16	4	2	1	1
3	18	4	4	1	1
4	20	1	2	1	1
5	6	5	3	1	1
5	20	2	1	1	1
5	21	5	5	2	2
6	9	4	8	1	1
6	21	1	1	1	1
8	10	1	1	1	1
9	10	3	2	1	1
10	23	3	1	1	1
11	12	1	1	1	1
11	14	1	2	1	1
13	15	1	1	1	1
14	15	4	2	1	1
14	16	1	1	1	1
14	23	3	1	1	1
16	20	4	4	1	1
16	24	1	1	1	1
17	18	2	1	1	1
20	21	1	1	1	1
21	23	2	6	1	1
23	24	4	7	2	2

instance P0715

numNodes 23

numArc 47

node1	node2	cost1	cost2	dh1	dh2
1	2	1	5	1	1
1	8	1	2	1	1
2	3	1	1	1	1
2	5	4	1	1	1
3	4	1	1	1	1
4	5	1	4	1	1
5	6	1	1	1	1
5	8	1	1	1	1
6	7	1	2	1	1
7	8	1	2	1	1
9	10	4	2	1	1
10	11	2	1	1	1
11	12	3	1	1	1
11	14	1	1	1	1
12	13	1	4	1	1
13	14	1	1	1	1
14	15	9	5	2	2
16	18	3	1	1	1
17	18	1	5	1	1
18	21	2	1	1	1
19	21	2	2	1	1
20	21	4	2	1	1
21	22	2	4	1	1
21	23	1	1	1	1
1	9	7	7	1	1
1	15	8	6	1	1
1	16	2	1	1	1
1	17	1	4	1	1
1	22	7	4	2	2
4	6	1	4	1	1
6	9	6	7	1	1
7	19	12	15	4	4
7	22	13	14	5	5
9	15	3	1	1	1
9	16	4	6	1	1
9	19	4	4	2	2
9	22	2	4	1	1
13	15	7	4	2	2
13	22	2	6	1	1
15	16	9	7	1	1
15	22	1	1	1	1
16	19	5	6	1	1
16	22	6	5	3	3
17	20	6	9	2	2
19	22	1	2	1	1
20	23	3	4	1	1
22	23	1	2	1	1

instance P0815

numNodes 17

numArc 40

node1	node2	cost1	cost2	dh1	dh2
1	2	2	1	1	1
1	9	1	1	1	1
1	10	1	1	1	1
2	3	1	1	1	1
3	4	2	4	1	1
3	10	2	1	1	1
4	5	1	3	1	1
5	6	1	4	1	1
5	8	4	2	1	1
6	7	1	1	1	1
7	8	4	8	2	2
7	9	1	3	1	1
7	10	2	1	1	1
8	9	1	1	1	1
11	12	5	4	1	1
12	13	5	6	1	1
13	14	1	1	1	1
13	16	1	1	1	1
14	15	1	1	1	1
14	16	1	2	1	1
15	17	3	1	1	1
16	17	1	1	1	1
11	17	4	1	1	1
11	16	8	6	1	1
2	4	1	1	1	1
3	6	1	1	1	1
4	11	10	10	1	1
4	12	3	5	1	1
4	17	12	10	3	3
5	11	7	11	1	1
5	15	10	10	1	1
5	17	10	7	4	4
8	11	9	9	4	4
8	12	7	9	3	3
8	15	9	10	1	1
8	17	6	7	2	2
9	15	11	7	3	3
9	17	6	9	2	2
12	16	1	2	1	1
15	16	2	1	1	1

instance P0915

numNodes 14

numArc 26

node1	node2	cost1	cost2	dh1	dh2
1	2	1	2	1	1
2	3	1	1	1	1
2	4	1	5	1	1
3	4	1	1	1	1
4	5	4	7	2	2
6	9	3	2	1	1
6	11	4	7	1	1
7	8	1	1	1	1
7	9	1	1	1	1
8	9	1	1	1	1
9	10	3	1	1	1
9	11	1	1	1	1
12	13	2	1	1	1
13	14	6	5	1	1
1	4	1	1	1	1
3	5	1	3	1	1
5	6	3	1	1	1
5	11	4	4	2	2
5	13	4	9	1	1
5	14	1	1	1	1
8	10	1	3	1	1
10	11	1	1	1	1
10	12	2	5	1	1
10	14	3	6	1	1
11	13	1	2	1	1
12	14	4	5	2	2

instance P1015

numNodes 12

numArc 20

node1	node2	cost1	cost2	dh1	dh2
1	2	4	5	2	2
1	3	1	1	1	1
2	3	3	4	1	1
4	5	3	3	1	1
6	7	1	4	1	1
7	8	4	4	2	2
8	9	2	2	1	1
10	11	2	1	1	1
10	12	4	3	1	1
11	12	4	1	1	1
1	4	8	7	3	3
1	5	6	8	3	3
1	6	7	8	1	1
2	5	10	12	4	4
2	11	1	2	1	1
5	6	11	8	1	1
5	10	13	11	5	5
6	8	8	5	3	3
7	10	1	1	1	1
9	12	1	2	1	1

instance P1115

numNodes 9

numArc 14

node1	node2	cost1	cost2	dh1	dh2
1	2	1	2	1	1
1	3	1	2	1	1
2	3	1	1	1	1
4	5	1	1	1	1
5	6	1	1	1	1
6	7	1	2	1	1
8	9	1	3	1	1
1	9	1	1	1	1
3	4	1	1	1	1
3	8	2	1	1	1
3	9	1	2	1	1
4	7	1	1	1	1
6	8	1	4	1	1
6	9	1	1	1	1

instance P1215

numNodes 7

numArc 18

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	3	3	1	1	1
2	3	1	1	1	1
4	5	1	2	1	1
6	7	1	2	1	1
1	7	1	1	1	1
2	4	1	1	1	1
2	5	1	1	1	1
2	6	1	1	1	1
2	7	1	1	1	1
3	4	1	1	1	1
3	5	1	1	1	1
3	6	1	1	1	1
3	7	1	1	1	1
4	6	1	2	1	1
4	7	1	1	1	1
5	6	1	1	1	1
5	7	2	1	1	1

instance P1315

numNodes 7

numArc 10

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
2	3	1	1	1	1
4	5	5	1	1	1
6	7	1	1	1	1
1	5	4	7	2	2
1	7	6	7	3	3
3	4	4	2	1	1
3	5	9	4	1	1
3	7	7	3	2	2
5	7	8	8	1	1

instance P1415

numNodes 28

numArc 79

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	4	3	1	1	1
2	3	4	1	1	1
2	4	4	2	1	1
3	4	3	1	1	1
5	6	1	2	1	1
6	7	3	3	2	2
6	8	5	5	1	1
9	10	2	1	1	1
9	14	8	8	3	3
10	11	1	2	1	1
11	12	2	3	1	1
11	14	10	8	4	4
12	13	4	9	1	1
12	14	1	1	1	1
13	14	1	1	1	1
15	16	4	1	1	1
16	17	3	1	1	1
18	19	6	5	2	2
18	20	3	1	1	1
19	20	1	3	1	1
20	21	1	2	1	1
22	23	2	1	1	1
22	24	8	4	2	2
23	24	5	1	1	1
24	25	1	1	1	1
24	26	6	2	1	1
25	26	1	1	1	1
26	27	1	1	1	1
25	28	1	2	1	1
27	28	1	1	1	1
1	28	1	3	1	1
2	28	1	3	1	1
3	5	5	5	2	2
3	8	2	2	1	1
3	22	8	7	3	3
3	25	5	8	1	1
4	22	8	6	2	2
4	25	4	5	2	2
5	8	3	1	1	1
5	22	11	9	1	1
5	25	10	6	3	3
6	9	9	9	4	4
6	10	5	8	2	2
6	13	19	18	4	4
6	19	17	19	2	2
6	21	12	15	3	3
7	8	1	1	1	1
7	9	4	4	2	2
7	10	4	7	2	2
7	13	9	13	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
7	14	12	8	3	3
7	19	12	10	5	5
7	21	8	11	3	3
7	22	9	9	5	5
8	22	7	9	3	3
8	25	7	6	2	2
9	19	14	12	1	1
9	21	12	11	5	5
9	22	10	9	4	4
13	15	1	1	1	1
13	19	10	10	5	5
13	21	15	12	1	1
13	22	12	16	4	4
14	19	7	8	2	2
14	21	13	14	3	3
14	22	16	15	5	5
15	17	5	1	1	1
15	19	9	8	2	2
16	19	5	10	1	1
17	18	6	7	1	1
17	19	8	10	3	3
17	20	8	6	3	3
17	26	16	16	1	1
18	26	17	14	3	3
20	23	4	5	2	2
20	26	10	9	4	4
21	23	5	1	1	1
22	25	7	3	2	2

instance P1515

numNodes 26

numArc 38

node1	node2	cost1	cost2	dh1	dh2
1	2	2	3	1	1
3	4	6	5	3	3
4	5	1	1	1	1
6	7	14	14	4	4
7	8	8	9	1	1
8	9	10	12	1	1
10	12	2	6	1	1
11	12	16	20	6	6
12	13	1	2	1	1
14	15	14	11	1	1
16	17	4	8	1	1
17	18	1	4	1	1
19	20	8	9	4	4
20	21	10	11	1	1
20	22	9	5	1	1
22	23	5	8	2	2
24	25	2	1	1	1
24	26	17	17	3	3
25	26	7	7	1	1
1	4	15	14	2	2
2	6	8	14	3	3
1	25	4	6	1	1
4	16	31	30	13	13
4	26	22	24	10	10
6	9	1	1	1	1
6	10	34	36	8	8
6	16	28	27	12	12
7	9	10	5	1	1
10	14	14	13	2	2
10	16	21	25	8	8
11	15	21	20	2	2
13	21	19	19	6	6
15	19	3	2	1	1
16	26	20	22	10	10
17	24	6	9	2	2
18	24	8	9	4	4
19	23	15	16	4	4
21	22	18	17	3	3

instance P1615

numNodes 31

numArc 94

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	3	3	4	1	1
2	3	5	1	1	1
3	4	2	2	1	1
4	5	1	1	1	1
6	7	5	2	1	1
7	8	9	8	4	4
8	9	2	1	1	1
10	11	1	2	1	1
10	12	3	1	1	1
11	12	1	1	1	1
12	13	1	1	1	1
12	14	4	4	2	2
13	14	1	1	1	1
15	16	2	1	1	1
15	17	1	1	1	1
16	17	3	3	1	1
16	20	1	3	1	1
17	20	1	1	1	1
17	18	1	2	1	1
17	19	1	1	1	1
18	19	2	1	1	1
19	20	1	3	1	1
19	21	1	3	1	1
20	21	1	1	1	1
22	23	4	6	1	1
24	25	3	1	1	1
25	26	4	6	2	2
27	28	4	5	1	1
28	29	4	1	1	1
28	30	5	1	1	1
28	31	1	1	1	1
29	30	7	8	3	3
30	31	3	1	1	1
1	6	12	7	1	1
1	7	8	5	1	1
1	23	9	9	4	4
2	5	6	5	1	1
2	30	9	4	2	2
2	31	2	7	1	1
3	30	6	6	1	1
3	31	6	7	2	2
4	7	4	8	2	2
5	6	6	3	2	2
5	23	5	5	1	1
5	30	6	5	2	2
5	31	7	4	2	2
6	8	1	1	1	1
6	23	3	1	1	1
6	29	18	13	5	5
6	30	11	7	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
7	9	1	1	1	1
8	11	3	8	1	1
8	14	7	4	2	2
8	18	10	14	5	5
8	21	15	12	1	1
8	22	5	7	2	2
9	10	2	6	1	1
9	14	6	6	2	2
9	18	11	13	3	3
9	21	14	10	4	4
9	22	3	5	1	1
10	13	2	3	1	1
11	14	1	2	1	1
11	18	5	7	3	3
11	22	7	5	2	2
13	15	1	1	1	1
14	18	6	6	3	3
14	22	8	3	1	1
18	22	4	6	2	2
18	24	11	9	4	4
18	26	10	7	4	4
18	29	9	14	1	1
18	30	15	16	5	5
21	22	8	5	1	1
21	24	9	11	1	1
21	26	13	12	6	6
21	29	14	15	1	1
21	30	19	17	9	9
22	24	7	10	3	3
22	26	4	5	2	2
22	29	10	10	5	5
22	30	13	10	1	1
23	24	10	11	2	2
23	26	9	10	2	2
23	29	12	9	3	3
23	30	10	6	3	3
24	26	9	9	5	5
24	29	9	6	1	1
24	30	11	13	2	2
26	27	3	4	2	2
26	29	5	5	1	1
26	30	8	11	3	3
27	29	3	1	1	1

instance P1715

numNodes 19

numArc 44

node1	node2	cost1	cost2	dh1	dh2
1	2	1	2	1	1
2	3	1	1	1	1
2	4	4	5	2	2
3	4	1	1	1	1
5	6	1	1	1	1
5	8	4	1	1	1
6	7	6	3	2	2
7	8	1	3	1	1
7	9	1	1	1	1
8	9	1	5	1	1
10	11	1	1	1	1
10	12	2	3	1	1
13	14	2	6	1	1
14	15	2	1	1	1
14	16	3	1	1	1
17	18	9	7	2	2
18	19	1	1	1	1
1	19	2	2	1	1
3	5	5	7	2	2
3	6	8	3	2	2
3	14	4	1	1	1
3	17	2	1	1	1
4	14	1	6	1	1
4	17	3	2	1	1
4	19	12	13	2	2
5	13	6	4	2	2
5	14	4	7	1	1
6	14	7	6	2	2
8	13	6	5	3	3
9	10	1	4	1	1
9	12	3	3	1	1
9	15	7	8	2	2
9	16	10	6	3	3
9	17	15	11	5	5
11	12	1	4	1	1
11	16	11	14	5	5
11	17	19	16	8	8
12	16	13	13	2	2
12	17	16	15	7	7
13	15	3	3	2	2
14	17	2	6	1	1
14	19	9	9	4	4
15	16	4	2	1	1
17	19	9	8	3	3

instance P1815

numNodes 23

numArc 37

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
3	4	1	3	1	1
4	5	1	1	1	1
5	6	1	1	1	1
5	7	1	1	1	1
8	9	1	1	1	1
10	11	1	2	1	1
10	12	1	1	1	1
11	12	1	1	1	1
12	13	2	1	1	1
14	15	1	1	1	1
16	17	13	13	3	3
17	18	7	11	2	2
18	19	2	4	1	1
20	21	1	1	1	1
22	23	3	1	1	1
1	23	1	3	1	1
2	7	11	9	4	4
2	19	6	11	2	2
2	22	1	3	1	1
3	7	1	1	1	1
4	8	25	28	3	3
4	12	22	24	8	8
6	18	1	1	1	1
7	19	3	1	1	1
7	22	12	11	2	2
8	16	16	21	1	1
9	10	3	2	1	1
11	14	4	5	2	2
12	16	16	12	5	5
12	17	9	8	4	4
13	15	4	2	1	1
16	18	1	5	1	1
17	19	1	3	1	1
17	20	5	6	2	2
19	22	9	8	2	2
21	22	3	1	1	1

instance P1915

numNodes 33

numArc 54

node1	node2	cost1	cost2	dh1	dh2
1	2	2	1	1	1
2	3	1	1	1	1
2	4	1	2	1	1
3	4	6	6	3	3
4	5	6	5	3	3
6	7	4	6	1	1
6	8	3	1	1	1
6	9	4	1	1	1
10	11	5	7	1	1
11	12	5	6	2	2
12	13	3	1	1	1
12	14	1	1	1	1
15	16	7	8	3	3
15	17	1	2	1	1
18	20	1	6	1	1
20	21	1	1	1	1
19	21	1	1	1	1
21	23	2	1	1	1
22	23	3	1	1	1
24	26	3	1	1	1
25	26	1	3	1	1
26	27	1	3	1	1
28	29	1	4	1	1
29	30	4	4	1	1
29	31	2	2	1	1
31	32	1	1	1	1
28	32	2	2	1	1
32	33	1	1	1	1
8	9	1	2	1	1
1	6	12	7	3	3
2	25	11	13	5	5
3	5	1	3	1	1
4	25	13	14	1	1
5	33	3	2	1	1
7	10	14	13	3	3
9	10	9	8	4	4
9	31	5	5	1	1
10	12	1	1	1	1
11	16	20	16	4	4
13	14	3	3	2	2
14	16	13	11	6	6
14	30	4	4	1	1
14	31	14	14	4	4
17	18	12	8	1	1
18	19	1	1	1	1
19	30	10	8	2	2
20	23	3	1	1	1
22	25	13	14	5	5
24	25	6	3	1	1
25	27	3	2	1	1
25	30	2	5	1	1
25	32	9	6	1	1
28	31	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
30	32	4	6	1	1

instance P2015

numNodes 50

numArc 98

node1	node2	cost1	cost2	dh1	dh2
1	2	2	2	1	1
1	4	1	1	1	1
1	7	5	5	2	2
2	3	3	2	1	1
3	4	1	1	1	1
4	5	2	1	1	1
5	6	1	1	1	1
5	7	3	4	2	2
6	7	1	1	1	1
8	9	3	1	1	1
8	15	1	1	1	1
9	10	1	1	1	1
9	14	5	4	1	1
10	11	6	5	2	2
10	13	1	1	1	1
11	12	1	2	1	1
12	13	3	5	1	1
13	14	1	2	1	1
14	15	2	3	1	1
14	16	1	1	1	1
15	16	3	3	1	1
17	18	4	8	2	2
17	20	3	5	2	2
18	19	5	6	2	2
19	21	4	5	1	1
20	21	4	2	1	1
20	23	1	2	1	1
20	24	1	1	1	1
21	22	1	1	1	1
22	23	5	4	2	2
23	24	1	1	1	1
25	26	1	1	1	1
26	27	5	3	1	1
27	28	1	1	1	1
28	29	3	1	1	1
29	30	6	6	2	2
28	31	9	9	4	4
30	31	2	1	1	1
26	31	2	6	1	1
26	32	5	6	1	1
31	32	1	3	1	1
33	35	3	1	1	1
34	35	1	1	1	1
35	36	2	1	1	1
35	38	5	4	1	1
36	37	7	7	2	2
36	38	1	1	1	1
37	38	4	3	1	1
39	40	1	1	1	1
39	42	1	1	1	1
40	41	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
41	42	1	1	1	1
41	43	5	8	2	2
41	46	1	2	1	1
43	46	1	3	1	1
43	44	1	1	1	1
44	46	1	1	1	1
44	45	1	4	1	1
45	46	7	8	1	1
47	48	3	2	1	1
47	49	4	2	1	1
48	50	1	1	1	1
49	50	5	4	1	1
4	15	7	4	1	1
4	39	16	16	3	3
4	46	14	15	2	2
5	15	3	2	1	1
5	39	16	17	5	5
5	45	12	12	1	1
5	46	11	12	3	3
5	48	9	6	1	1
12	17	8	6	2	2
12	24	6	6	3	3
12	33	21	20	1	1
12	36	15	17	5	5
12	39	26	23	6	6
13	33	24	24	3	3
13	36	18	13	6	6
15	39	14	14	2	2
15	45	17	16	4	4
15	46	9	9	3	3
16	17	14	17	2	2
16	24	18	16	2	2
16	25	26	30	10	10
16	33	18	22	7	7
16	36	12	11	3	3
17	36	22	18	1	1
17	39	27	28	8	8
23	27	2	3	1	1
24	36	16	19	2	2
24	39	30	30	4	4
26	33	4	3	1	1
33	34	2	1	1	1
36	39	21	22	4	4
36	40	20	23	7	7
39	48	21	21	10	10
45	48	10	14	4	4
46	48	16	16	2	2

instance P2115

numNodes 49

numArc 110

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	6	3	1	1	1
2	3	5	2	1	1
2	6	4	4	2	2
3	4	1	3	1	1
4	5	1	1	1	1
4	6	7	7	2	2
5	8	2	1	1	1
6	7	6	4	1	1
6	8	1	1	1	1
7	8	1	2	1	1
9	10	3	5	1	1
9	13	5	3	1	1
10	11	1	3	1	1
10	12	1	1	1	1
10	13	1	1	1	1
11	12	2	5	1	1
11	16	1	1	1	1
12	13	7	7	2	2
12	16	3	5	1	1
13	14	1	3	1	1
14	15	2	1	1	1
15	16	1	1	1	1
17	18	1	1	1	1
17	20	2	1	1	1
19	21	1	1	1	1
20	21	1	2	1	1
20	23	1	1	1	1
21	22	2	1	1	1
18	22	2	1	1	1
23	24	4	5	2	2
23	25	4	8	2	2
24	26	2	3	1	1
25	26	3	1	1	1
27	28	3	7	1	1
27	29	3	2	1	1
28	29	2	4	1	1
28	30	1	1	1	1
28	31	1	2	1	1
29	30	1	1	1	1
29	32	1	3	1	1
30	31	3	1	1	1
30	32	6	2	1	1
31	32	5	2	1	1
31	33	1	1	1	1
32	33	1	1	1	1
32	34	4	2	1	1
33	34	1	3	1	1
35	37	1	1	1	1
36	38	1	1	1	1
37	38	1	1	1	1
38	39	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
39	40	3	3	1	1
40	41	4	1	1	1
41	42	1	1	1	1
43	44	1	1	1	1
43	47	1	1	1	1
43	49	8	6	3	3
44	45	1	1	1	1
44	49	2	1	1	1
48	49	3	2	1	1
45	48	1	1	1	1
45	46	1	3	1	1
46	48	3	4	1	1
46	47	1	1	1	1
47	48	1	4	1	1
47	49	1	2	1	1
1	7	4	3	2	2
1	17	12	12	4	4
1	18	13	12	3	3
7	17	13	15	3	3
7	18	8	9	2	2
8	9	3	4	2	2
8	18	7	12	4	4
8	22	12	13	4	4
8	26	11	11	4	4
8	27	9	7	1	1
8	29	10	8	4	4
8	34	17	15	7	7
8	45	13	16	2	2
9	18	13	12	1	1
9	22	20	19	5	5
9	26	13	14	6	6
9	27	5	7	1	1
9	35	14	11	2	2
9	45	21	21	11	11
15	35	6	3	1	1
18	26	9	4	2	2
18	29	6	8	2	2
18	34	16	14	7	7
18	45	13	12	3	3
21	24	1	1	1	1
22	24	1	1	1	1
22	29	7	10	1	1
22	34	15	12	6	6
22	45	9	14	3	3
26	29	2	7	1	1
26	34	6	7	1	1
26	43	4	3	1	1
26	45	10	6	3	3
28	35	1	1	1	1
29	45	11	6	2	2
31	35	1	1	1	1
31	39	1	1	1	1
34	41	8	9	1	1
34	45	1	1	1	1
35	36	4	2	1	1
38	40	6	3	1	1
39	41	2	6	1	1
41	46	14	17	3	3

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2

instance P2215

numNodes 50

numArc 184

node1	node2	cost1	cost2	dh1	dh2
1	2	2	6	1	1
1	10	1	2	1	1
2	3	3	3	2	2
2	10	3	3	2	2
3	4	5	1	1	1
3	10	8	7	2	2
4	5	1	2	1	1
4	7	4	6	1	1
4	9	4	5	1	1
5	6	2	1	1	1
6	7	8	6	3	3
7	8	7	6	3	3
8	9	1	3	1	1
9	10	1	2	1	1
11	12	5	5	1	1
11	14	10	11	3	3
11	15	4	6	2	2
12	13	3	4	1	1
13	15	2	3	1	1
13	16	9	6	2	2
14	15	1	1	1	1
14	17	5	4	1	1
15	16	4	7	2	2
16	17	2	4	1	1
17	18	1	3	1	1
17	19	6	3	1	1
20	21	7	8	3	3
20	26	6	7	2	2
20	27	2	3	1	1
21	22	4	5	2	2
21	25	10	10	3	3
21	26	5	2	1	1
22	23	1	1	1	1
22	25	6	7	1	1
23	24	6	4	1	1
23	25	3	3	1	1
24	25	3	1	1	1
25	26	3	5	1	1
26	27	2	2	1	1
28	29	3	2	1	1
28	31	6	10	3	3
28	32	5	4	2	2
28	35	4	3	2	2
29	30	6	2	1	1
30	31	1	3	1	1
31	32	5	2	1	1
31	33	11	8	2	2
32	33	1	2	1	1
32	34	10	5	2	2
32	35	4	2	1	1
33	34	4	4	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
34	35	2	2	1	1
36	37	3	5	2	2
36	39	11	14	5	5
36	40	1	3	1	1
37	38	1	6	1	1
37	39	7	8	4	4
38	39	6	5	3	3
39	41	4	9	1	1
40	41	3	7	2	2
42	43	5	5	1	1
42	45	2	5	1	1
43	44	6	7	3	3
43	45	4	4	2	2
43	46	9	8	4	4
45	46	1	3	1	1
45	47	3	1	1	1
45	48	3	4	2	2
45	50	9	5	3	3
46	48	5	7	1	1
47	50	1	1	1	1
48	49	7	3	1	1
48	50	8	7	2	2
49	50	5	1	1	1
7	11	11	15	3	3
7	14	16	15	3	3
7	18	14	18	6	6
7	20	25	24	10	10
7	31	16	18	1	1
7	33	24	24	11	11
7	39	32	28	7	7
7	42	34	35	13	13
7	43	37	36	1	1
7	45	36	40	12	12
7	47	34	36	16	16
8	11	12	10	4	4
8	14	15	13	2	2
8	18	11	13	1	1
8	20	20	19	7	7
8	30	13	12	6	6
8	31	12	12	1	1
8	33	21	20	4	4
8	39	26	26	9	9
8	42	33	33	8	8
8	43	30	32	15	15
8	44	33	38	7	7
8	45	30	34	7	7
8	47	34	29	8	8
9	14	16	16	3	3
9	18	14	17	6	6
9	20	23	19	4	4
9	29	4	2	1	1
9	31	11	12	5	5
9	39	23	23	2	2
9	42	32	32	5	5
9	43	33	32	13	13
9	44	36	33	3	3

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
9	45	32	33	1	1
9	47	31	31	4	4
10	28	3	1	1	1
11	29	26	26	2	2
11	31	17	19	4	4
11	33	23	25	4	4
11	34	29	34	9	9
11	39	29	28	1	1
14	18	1	3	1	1
14	31	7	12	4	4
14	33	18	16	5	5
18	19	6	4	1	1
18	21	1	1	1	1
18	31	7	10	1	1
18	33	13	15	6	6
18	34	16	16	5	5
18	39	16	19	6	6
19	21	9	10	3	3
19	23	13	16	7	7
20	31	18	16	1	1
20	33	8	12	4	4
20	34	15	13	5	5
20	39	9	10	2	2
25	33	23	24	2	2
25	34	20	20	2	2
25	39	19	20	2	2
25	42	11	11	6	6
25	43	13	11	3	3
25	44	7	10	2	2
25	45	16	15	3	3
25	47	13	11	4	4
26	33	15	19	5	5
26	34	19	17	3	3
26	39	15	14	1	1
26	42	7	5	2	2
26	43	8	4	1	1
26	44	9	10	2	2
26	45	6	8	1	1
26	47	7	10	3	3
27	34	15	12	2	2
27	39	12	13	4	4
27	42	9	6	1	1
27	43	8	11	2	2
27	44	14	14	3	3
27	45	11	10	5	5
27	47	7	7	1	1
28	37	10	9	1	1
31	39	17	17	2	2
31	42	22	22	9	9
31	43	28	27	7	7
31	44	32	27	11	11
31	45	29	29	11	11
31	47	24	25	11	11
33	39	12	14	3	3
33	42	20	16	8	8
33	43	20	19	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
33	44	22	24	5	5
33	45	22	23	1	1
33	47	17	21	3	3
34	38	8	10	1	1
34	39	5	3	1	1
34	42	16	17	2	2
34	43	19	17	4	4
34	44	24	23	6	6
34	45	18	19	1	1
34	47	17	16	2	2
35	37	6	3	2	2
39	42	13	16	7	7
39	43	19	17	7	7
39	44	19	19	8	8
39	45	15	16	6	6
39	47	16	17	5	5
41	47	14	14	2	2
41	50	11	13	1	1
42	47	5	2	1	1
44	46	8	8	1	1
44	47	16	14	2	2

instance P2315

numNodes 50

numArc 158

node1	node2	cost1	cost2	dh1	dh2
1	2	5	3	1	1
1	6	1	2	1	1
1	8	4	6	2	2
2	3	2	1	1	1
2	5	1	3	1	1
2	6	1	1	1	1
3	4	2	4	1	1
3	5	3	2	1	1
3	6	5	6	1	1
4	5	5	8	1	1
5	6	1	5	1	1
5	7	5	4	1	1
6	8	6	5	3	3
7	8	8	3	1	1
7	9	1	1	1	1
8	9	1	1	1	1
10	11	1	5	1	1
10	12	1	1	1	1
10	13	3	2	1	1
11	13	7	4	1	1
12	13	1	2	1	1
13	14	3	2	1	1
13	15	1	1	1	1
13	16	6	6	2	2
14	16	2	1	1	1
15	16	3	1	1	1
15	17	5	6	2	2
16	17	2	4	1	1
16	18	2	2	1	1
17	18	1	1	1	1
19	20	5	3	1	1
19	24	5	4	2	2
19	25	1	1	1	1
20	21	8	6	2	2
20	25	1	1	1	1
20	26	1	1	1	1
21	22	4	3	1	1
21	26	2	1	1	1
22	23	2	3	1	1
23	24	3	1	1	1
23	26	1	5	1	1
24	25	2	3	1	1
25	26	4	4	1	1
27	28	5	4	2	2
27	31	9	11	2	2
28	29	3	3	1	1
28	30	7	7	3	3
29	30	1	1	1	1
30	31	1	1	1	1
30	33	2	5	1	1
32	33	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
31	32	1	1	1	1
34	35	3	6	1	1
34	37	1	1	1	1
35	36	7	4	2	2
35	37	1	5	1	1
35	38	1	1	1	1
36	38	1	1	1	1
37	39	1	1	1	1
37	40	4	2	1	1
39	40	1	1	1	1
39	41	1	1	1	1
40	41	4	4	2	2
42	43	7	4	2	2
42	45	2	1	1	1
43	44	1	1	1	1
43	49	1	1	1	1
44	49	4	1	1	1
44	50	1	1	1	1
45	46	9	7	3	3
45	49	1	6	1	1
46	47	1	3	1	1
46	49	1	1	1	1
47	48	4	1	1	1
47	49	1	1	1	1
47	50	2	2	1	1
48	50	3	1	1	1
49	50	1	1	1	1
1	19	19	19	9	9
1	23	21	24	7	7
1	34	17	19	4	4
1	36	27	24	9	9
1	43	34	30	14	14
4	7	5	4	2	2
4	12	3	2	1	1
4	19	6	7	3	3
4	21	12	14	4	4
4	23	9	11	3	3
4	27	18	21	2	2
4	32	20	20	2	2
4	34	4	4	1	1
4	36	11	14	6	6
4	43	21	20	3	3
4	44	18	18	5	5
4	48	25	23	1	1
5	8	2	1	1	1
7	19	7	7	2	2
7	23	15	13	1	1
7	34	8	10	2	2
7	36	14	18	3	3
7	43	22	20	7	7
12	21	12	12	5	5
14	18	6	4	2	2
18	21	7	7	4	4
18	22	11	7	2	2
18	27	6	9	3	3
18	32	7	10	2	2

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
18	36	15	14	6	6
18	38	10	13	4	4
18	40	12	10	2	2
18	42	12	15	3	3
18	43	9	9	4	4
18	44	8	9	3	3
18	48	14	13	5	5
19	34	3	7	1	1
19	36	14	10	4	4
19	43	16	20	4	4
21	27	10	13	1	1
21	32	11	10	5	5
21	43	11	8	4	4
21	44	8	10	4	4
21	48	16	16	8	8
22	27	9	10	3	3
22	32	7	10	2	2
22	36	11	8	3	3
22	38	12	7	3	3
22	40	9	10	3	3
22	42	14	13	4	4
22	43	10	7	2	2
22	44	9	7	1	1
22	48	12	11	3	3
23	25	2	5	1	1
23	34	3	6	1	1
23	36	14	12	1	1
27	30	1	1	1	1
27	32	2	1	1	1
27	43	1	1	1	1
27	44	2	1	1	1
27	48	2	2	1	1
29	33	1	1	1	1
30	32	1	1	1	1
32	43	3	1	1	1
32	44	1	1	1	1
32	48	5	6	3	3
33	48	3	5	1	1
34	36	8	7	1	1
36	43	6	5	2	2
38	40	2	1	1	1
38	42	6	2	1	1
38	43	6	2	1	1
38	46	8	9	2	2
40	42	5	7	2	2
40	43	7	5	1	1
40	46	10	7	2	2
41	42	8	8	4	4
41	46	6	7	2	2
42	49	6	6	3	3
43	48	2	5	1	1

instance P2415

numNodes 41

numArc 125

node1	node2	cost1	cost2	dh1	dh2
1	2	1	1	1	1
1	3	5	4	1	1
2	3	3	5	1	1
2	4	7	5	2	2
3	5	1	1	1	1
6	7	2	3	1	1
6	9	3	4	1	1
7	8	3	6	1	1
8	9	1	3	1	1
8	10	1	1	1	1
9	10	3	1	1	1
11	12	2	3	1	1
12	13	1	1	1	1
12	14	1	1	1	1
13	14	3	1	1	1
14	15	3	4	1	1
14	16	6	9	3	3
15	16	6	5	2	2
16	17	4	4	2	2
18	19	4	1	1	1
18	21	1	1	1	1
19	20	1	1	1	1
19	21	5	2	1	1
19	22	4	3	1	1
20	22	5	4	1	1
21	22	1	1	1	1
21	23	1	3	1	1
22	23	1	1	1	1
24	25	12	8	1	1
24	26	8	9	1	1
25	27	3	6	2	2
25	28	3	4	1	1
26	27	6	7	1	1
26	29	3	5	2	2
27	28	5	2	1	1
27	29	1	1	1	1
28	30	6	3	2	2
29	30	1	2	1	1
31	32	2	4	1	1
31	33	3	1	1	1
32	33	2	1	1	1
32	34	1	1	1	1
32	35	2	1	1	1
33	34	2	3	1	1
33	35	6	4	2	2
34	35	4	8	2	2
36	37	3	3	1	1
36	39	1	3	1	1
37	38	3	1	1	1
37	39	3	6	2	2
37	40	1	1	1	1

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
38	40	3	1	1	1
38	41	1	1	1	1
39	40	6	6	1	1
40	41	7	3	2	2
3	18	7	5	1	1
4	5	9	9	2	2
4	8	18	15	3	3
5	20	1	2	1	1
7	9	3	1	1	1
7	11	7	8	2	2
7	13	9	10	2	2
7	16	10	11	4	4
7	17	11	14	2	2
7	25	9	9	1	1
7	30	20	18	3	3
7	31	11	14	6	6
7	38	25	27	11	11
8	20	7	4	2	2
8	24	10	9	2	2
8	29	16	15	5	5
8	36	17	19	8	8
10	11	8	11	2	2
10	16	9	6	3	3
10	17	9	4	1	1
10	25	3	6	1	1
10	30	16	15	3	3
10	31	6	9	1	1
10	38	22	20	4	4
11	13	8	7	1	1
11	16	11	14	4	4
11	17	11	13	4	4
11	20	13	13	3	3
11	24	21	20	7	7
11	25	7	8	4	4
11	30	22	18	1	1
11	31	14	10	3	3
11	36	24	24	3	3
11	38	25	25	11	11
13	15	1	2	1	1
15	17	1	3	1	1
16	20	14	11	1	1
16	24	21	17	1	1
16	25	5	7	1	1
16	30	16	18	8	8
16	31	13	11	2	2
16	36	21	21	1	1
16	38	23	22	5	5
17	20	11	12	2	2
17	24	17	20	8	8
17	25	6	6	1	1
17	29	14	16	7	7
17	30	10	11	2	2
17	31	7	2	1	1
17	36	19	20	8	8
17	38	15	18	4	4
20	24	10	13	5	5

continued . . .

. . . continued

node1	node2	cost1	cost2	dh1	dh2
20	25	11	11	1	1
20	29	7	5	3	3
20	31	14	15	3	3
20	36	9	9	4	4
20	38	12	12	1	1
23	29	9	8	2	2
23	31	23	23	10	10
23	36	10	11	5	5
23	38	20	17	2	2
24	31	19	18	1	1
24	36	23	22	2	2
25	31	7	11	3	3
29	31	7	11	4	4
29	36	8	11	3	3
29	38	6	6	2	2
30	31	11	7	1	1
31	36	15	15	4	4
33	38	10	8	4	4

Appendix B

PPP Solutions

filename	A3101	1 2 3 10 7 4 5 4 7 6 7 10 11 10 15 14 9 8 5 6 9 8 5 6 9 10 15 14 9 10 16 15
numNodes	116	16 17 11 18 17 11 18 21 17 18 21 20 16 17 21 20 16 20 19 15 19 15 19 25 31
numArcs	174	32 31 36 30 36 41 36 37 44 48 44 49 46 39 35 39 38 46 45 46 49 48 49 88 91
minimum	24672	90 87 86 85 47 42 47 48 87 88 92 91 100 99 98 96 99 98 97 95 97 98 96 99
initial	24696	100 101 100 103 105 104 102 67 97 67 102 103 105 108 109 108 110 111 110
LS	24685	112 113 112 116 115 114 112 116 115 114 112 110 108 107 106 83 82 79 75
Prune	24677	70 71 72 73 77 73 77 76 81 80 70 75 74 75 79 80 79 82 79 82 81 78 81 80 70
time	190.188	71 72 66 57 66 65 71 76 77 73 72 66 65 63 58 50 56 40 30 12 3 10 16 20 19
aveCost	71.592	25 31 36 37 38 39 28 24 27 22 20 22 23 22 27 29 26 25 32 33 29 26 25 32 33
aveDiff	3.55172	29 27 34 33 34 37 38 45 38 46 39 28 24 27 34 37 44 43 42 43 44 49 88 91 90
stdevCost	56.08	89 86 89 96 95 93 85 84 41 40 56 57 56 63 55 54 53 52 53 54 53 54 62 55 63
stdevDiff	2.50635	58 50 51 52 60 59 51 52 60 61 53 61 62 61 62 55 54 62 61 60 59 51 50 56 63
initialRouteL	369	64 63 65 71 76 81 82 83 82 83 94 69 106 83 94 69 106 107 68 67 68 107 108
routeLength	365	105 104 102 103 100 90 87 86 85 47 48 87 88 92 91 100 90 89 96 95 93 85
		84 57 84 93 94 84 93 94 84 41 42 41 40 30 12 13 8 13 8 13 14 13 12 3 2 1

filename	A3101	1 2 3 10 7 4 5 4 7 6 7 10 11 10 15 14 9 8 5 6 9 8 5 6 9 10 15 14 9 10 16 15
numNodes	116	16 17 11 18 17 11 18 21 17 18 21 20 16 17 21 20 16 20 19 15 19 15 19 25 31
numArcs	174	32 31 36 30 36 41 36 37 44 48 44 49 46 39 35 39 38 46 45 46 49 48 49 88 91
minimum	24672	90 87 86 85 47 42 47 48 87 88 92 91 100 99 98 96 99 98 97 95 97 98 96 99
initial	24696	100 101 100 103 105 104 102 67 97 67 102 103 105 108 109 108 110 111 110
LS	24685	112 113 112 116 115 114 112 116 115 114 112 110 108 107 106 83 82 79 75
Prune	24677	70 71 72 73 77 73 77 76 81 80 70 75 74 75 79 80 79 82 79 82 81 78 81 80 70
time	190.188	71 72 66 57 66 65 71 76 77 73 72 66 65 63 58 50 56 40 30 12 3 10 16 20 19
aveCost	71.592	25 31 36 37 38 39 28 24 27 22 20 22 23 22 27 29 26 25 32 33 29 26 25 32 33
aveDiff	3.55172	29 27 34 33 34 37 38 45 38 46 39 28 24 27 34 37 44 43 42 43 44 49 88 91 90
stdevCost	56.08	89 86 89 96 95 93 85 84 41 40 56 57 56 63 55 54 53 52 53 54 53 54 62 55 63
stdevDiff	2.50635	58 50 51 52 60 59 51 52 60 61 53 61 62 61 62 55 54 62 61 60 59 51 50 56 63
initialRouteL	369	64 63 65 71 76 81 82 83 82 83 94 69 106 83 94 69 106 107 68 67 68 107 108
routeLength	365	105 104 102 103 100 90 87 86 85 47 48 87 88 92 91 100 90 89 96 95 93 85
		84 57 84 93 94 84 93 94 84 41 42 41 40 30 12 13 8 13 8 13 14 13 12 3 2 1

filename	M3101	1 2 25 24 33 32 31 30 28 27 26 25 31 32 59 60 30 28 29 30 31 32 31 32 59
numNodes	196	60 67 66 60 67 68 69 70 80 79 110 108 107 126 127 105 104 129 135 128 137
numArcs	316	136 134 133 132 130 131 130 187 133 135 136 137 138 142 146 149 150 151
minimum	46567	147 146 142 141 140 139 136 137 139 136 135 128 105 106 99 100 105 128
initial	46620	137 136 134 133 132 130 101 102 129 187 133 135 129 187 130 101 102 71
LS	46575	29 62 63 74 75 76 99 100 74 73 72 103 104 129 102 103 104 73 74 75 189
Prune	46573	64 65 66 76 77 76 75 189 64 65 61 62 63 64 63 74 100 105 106 188 98 97 78
time	1100.37	67 66 60 30 26 1 2 3 4 20 22 3 22 23 24 33 23 24 3 24 25 31 32 33 23 21 34
aveCost	74.5	192 34 192 35 17 16 191 16 15 7 8 15 8 194 12 11 10 9 10 11 43 42 41 50
aveDiff	3.55063	49 44 42 43 45 44 46 44 49 48 46 196 9 195 12 194 13 12 195 194 13 14 39
stdevCost	38.5549	40 13 12 41 40 13 14 15 16 17 193 7 8 15 7 193 6 18 35 192 34 58 68 69 79
stdevDiff	2.40266	110 108 125 145 123 122 113 114 115 114 115 119 116 115 119 116 117 89
initialRouteL	671	90 89 88 118 117 89 88 118 117 116 115 114 113 111 110 96 95 112 113 111
routeLength	669	124 110 111 124 186 124 123 145 144 143 127 106 188 107 108 125 186 124
		186 124 123 122 121 122 121 122 113 120 121 163 162 161 174 159 158 154
		150 151 155 151 184 156 153 152 156 161 170 171 170 169 185 169 168 162
		168 163 162 161 156 152 144 126 125 145 144 126 125 186 124 110 96 190
		96 95 92 82 81 56 81 80 190 81 80 190 82 91 90 87 47 48 49 50 52 53 54 57
		56 53 52 55 83 82 91 90 87 85 84 51 52 51 47 87 85 86 85 84 51 47 48 49
		48 46 196 10 196 9 10 9 195 194 14 15 8 194 14 39 38 50 52 55 83 84 83 82
		81 190 82 92 91 92 95 94 93 115 114 94 93 91 93 115 114 94 95 112 111 112
		113 120 121 163 164 167 168 163 164 120 164 165 164 167 166 165 166 167
		168 169 170 161 174 173 172 173 182 180 179 178 177 175 176 175 176 175
		159 158 157 177 178 179 176 173 182 180 181 183 182 183 181 180 179 176
		173 174 159 160 155 154 158 157 148 157 177 175 159 160 184 160 155 151
		155 151 147 143 142 141 138 137 139 140 148 140 139 140 141 138 142 143
		147 146 149 148 149 150 154 155 151 184 156 153 122 153 152 144 143 127
		105 104 73 72 71 102 103 72 71 29 30 26 25 2 3 4 5 6 18 35 17 193 6 5 4 20
		22 23 21 20 21 19 192 34 36 34 21 19 5 19 18 17 18 19 192 34 58 57 58 68
		69 79 78 67 68 69 70 57 56 55 56 53 38 39 40 41 12 11 43 45 10 45 44 42 41
		50 38 37 54 53 38 37 54 57 70 80 79 78 77 98 97 109 97 78 77 98 188 107
		126 127 106 99 76 66 65 61 30 61 62 29 28 27 26 1

filename	P0215	1 4 3 10 5 3 2 3 1 4 3 7 1 13 5 2 3 8 1 13 5 2 3 8 7 2 8 7 2 8 9 10 5 3 2 3 1
numNodes	14	5 1 10 2 3 7 1 10 2 3 10 7 5 6 7 5 6 7 8 9 10 11 12 9 12 10 11 12 10 14 13
numArcs	33	10 14 13 10 7 8 1
minimum	722	
initial	726	
LS	722	
Prune	722	
time	1.375	
aveCost	11.5303	
aveDiff	1.66667	
stdevCost	8.21532	
stdevDiff	1.49071	
initialRouteL	75	
routeLength	75	

filename	P0315	1 2 3 1 2 3 4 2 4 5 4 22 3 4 22 23 22 27 7 9 10 12 14 13 11 13 24 15 20 16
numNodes	28	15 18 15 21 16 17 20 17 21 18 16 21 20 19 18 17 15 25 26 27 7 9 11 13 24
numArcs	57	23 28 24 25 28 27 5 7 6 8 7 6 5 6 8 10 12 11 12 14 13 25 26 28 24 15 13 15
minimum	164	20 16 15 21 18 16 17 21 20 19 18 17 15 25 24 23 28 22 27 5 6 5 7 8 10 9 11
initial	167	13 11 13 25 28 27 26 28 22 3 1
LS	164	
Prune	164	
time	4.422	
aveCost	1.7807	
aveDiff	0.859649	
stdevCost	1.30961	
stdevDiff	1.09926	
initialRouteL	121	
routeLength	121	

filename	P0415	1 4 2 1 4 2 1 16 14 15 16 14 15 17 5 4 3 2 3 6 5 4 3 6 5 4 17 5 4 17 13 5 8 6
numNodes	17	7 8 7 9 10 7 9 10 7 12 9 11 12 9 11 12 13 5 8 10 13 6 7 12 13 6 8 10 13 14
numArcs	35	17 13 14 17 16 15 17 16 1
minimum	130	
initial	130	
LS	130	
Prune	130	
time	2.312	
aveCost	2.25714	
aveDiff	0.914286	
stdevCost	2.15596	
stdevDiff	1.25063	
initialRouteL	73	
routeLength	73	

filename	P0515	1 6 5 2 1 6 5 4 3 2 5 4 7 4 12 9 8 7 11 7 14 4 12 9 10 8 7 14 9 10 11 10 8 9
numNodes	20	16 13 12 14 9 16 13 12 16 15 14 12 16 17 13 17 18 19 20 5 16 17 18 19 20 5
numArcs	35	16 20 18 20 16 15 14 4 3 2 1
minimum	360	
initial	360	
LS	360	
Prune	360	
time	0.906	
aveCost	5.65714	
aveDiff	1.31429	
stdevCost	5.50035	
stdevDiff	1.28222	
initialRouteL	71	
routeLength	71	

filename	P0615	1 3 18 1 3 4 20 5 21 23 10 8 7 6 5 21 6 9 10 23 14 13 15 16 3 2 1 18 19 15
numNodes	24	14 11 10 9 6 9 7 6 21 23 14 12 11 10 8 9 7 8 9 6 5 4 20 21 22 23 10 23 24
numArcs	46	16 20 21 22 23 24 22 20 16 17 15 14 12 11 14 16 14 13 12 13 15 19 18 17 15
minimum	168	16 24 22 20 5 4 3 18 1 18 17 16 3 2 1
initial	176	
LS	168	
Prune	168	
time	8.39	
aveCost	2.26087	
aveDiff	1.17391	
stdevCost	1.61427	
stdevDiff	1.32377	
initialRouteL	99	
routeLength	99	

filename	P0715	1 2 3 4 5 6 9 16 19 22 1 8 7 19 9 22 15 1 17 18 16 15 9 19 22 16 19 21 22 1
numNodes	23	17 18 21 22 23 22 15 14 11 12 13 12 13 22 7 19 21 20 23 21 20 17 20 23 21
numArcs	47	18 16 1 15 13 12 11 10 9 1 8 5 2 3 4 6 9 1 15 1 2 3 4 6 7 8 5 2 3 4 5 6 7 22
minimum	277	15 9 16 15 14 11 10 9 22 15 13 14 13 22 16 1
initial	282	
LS	279	
Prune	277	
time	28.532	
aveCost	3.55319	
aveDiff	1.61702	
stdevCost	3.08606	
stdevDiff	1.26412	
initialRouteL	107	
routeLength	105	

filename	P0815	1 2 1 10 3 4 2 3 4 2 3 6 3 10 7 8 5 6 7 8 5 6 7 9 1 10 7 9 8 9 17 4 5 11 4 5
numNodes	17	11 4 12 11 8 12 13 14 13 16 11 8 12 13 16 11 12 16 15 8 17 4 12 16 15 8 17
numArcs	40	5 15 9 17 5 15 14 16 17 11 17 11 17 15 14 16 17 15 9 1
minimum	279	
initial	279	
LS	279	
Prune	279	
time	2.734	
aveCost	4.05	
aveDiff	1.3	
stdevCost	3.47815	
stdevDiff	1.249	
initialRouteL	83	
routeLength	83	

filename	P0915	1 2 3 2 4 1 2 4 5 3 4 5 3 5 6 5 11 5 13 11 9 6 11 9 6 11 10 9 7 8 9 7 8 10 9 8
numNodes	14	10 12 10 12 10 14 5 13 11 10 14 13 12 14 13 12 14 5 3 4 1
numArcs	26	
minimum	108	
initial	112	
LS	108	
Prune	108	
time	4.156	
aveCost	2.55769	
aveDiff	1.34615	
stdevCost	1.99435	
stdevDiff	1.41264	
initialRouteL	61	
routeLength	57	

filename	P1015	1 2 3 1 2 3 1 5 1 6 5 2 5 4 1 6 5 10 7 8 6 7 8 6 7 10 12 9 8 9 12 11 2 11 10
numNodes	12	12 11 10 5 4 1
numArcs	20	
minimum	170	
initial	170	
LS	170	
Prune	170	
time	0.891	
aveCost	4.65	
aveDiff	1.3	
stdevCost	3.43184	
stdevDiff	1.05357	
initialRouteL	41	
routeLength	41	

filename	P1115	1 2 1 3 2 3 4 3 8 3 9 1 3 9 6 7 4 5 6 7 4 5 6 8 9 6 8 9 1
numNodes	9	
numArcs	14	
minimum	30	
initial	30	
LS	30	
Prune	30	
time	0.156	
aveCost	1.35714	
aveDiff	0.714286	
stdevCost	0.717848	
stdevDiff	0.880631	
initialRouteL	29	
routeLength	29	

filename	P1215	1 2 3 1 2 3 1 7 2 4 5 2 4 5 3 4 6 2 5 3 4 6 3 7 2 6 3 7 4 7 5 6 7 5 6 7 1
numNodes	7	
numArcs	18	
minimum	36	
initial	36	
LS	36	
Prune	36	
time	0.546	
aveCost	1.16667	
aveDiff	0.333333	
stdevCost	0.440959	
stdevDiff	0.57735	
initialRouteL	37	
routeLength	37	

filename	P1315	1 5 3 2 1 5 3 2 1 7 3 5 4 3 5 4 3 5 7 3 5 7 6 7 1
numNodes	7	
numArcs	10	
minimum	67	
initial	67	
LS	67	
Prune	67	
time	0.172	
aveCost	4.05	
aveDiff	1.9	
stdevCost	2.80134	
stdevDiff	1.86815	
initialRouteL	25	
routeLength	25	

filename	P1415	1 4 1 28 27 26 24 22 7 21 9 7 10 11 14 12 13 22 9 22 14 21 13 14 7 10 11 12
numNodes	28	13 6 19 7 8 25 22 4 3 25 22 5 3 25 5 3 8 22 3 2 28 27 26 25 24 22 5 6 7 6 21
numArcs	79	20 23 21 9 14 19 20 18 17 26 18 17 16 15 13 19 9 7 13 6 8 5 6 10 9 6 19 7 8
minimum	871	25 5 6 8 5 6 10 9 6 21 13 14 7 13 22 7 21 20 23 22 14 21 20 26 24 23 21 20
initial	876	17 16 19 15 16 15 13 19 9 14 11 12 14 19 15 16 19 18 17 15 17 26 20 17 19
LS	871	18 17 15 17 19 20 18 26 25 24 23 22 4 3 8 22 3 2 1 28 25 4 2 28 25 4 2 1
Prune	871	
time	30.983	
aveCost	6.17722	
aveDiff	1.6962	
stdevCost	4.7381	
stdevDiff	1.40842	
initialRouteL	171	
routeLength	169	

filename	P1515	1 2 1 25 24 17 18 24 25 26 4 1 25 26 4 3 4 5 4 16 6 7 6 9 7 8 9 7 8 9 6 10 12
numNodes	26	11 12 13 21 20 19 15 11 15 14 10 12 13 21 22 20 21 22 20 22 23 19 20 22 23
numArcs	37	19 15 14 10 16 6 10 16 17 18 24 26 16 17 24 26 16 4 1
minimum	840	
initial	840	
LS	840	
Prune	840	
time	2.093	
aveCost	11.8243	
aveDiff	1.7027	
stdevCost	8.69973	
stdevDiff	1.3728	
initialRouteL	77	
routeLength	77	

filename	P1615	1 2 1 3 2 5 4 3 2 5 4 7 1 3 4 7 1 23 6 8 22 23 22 23 22 24 25 24 25 26 18 30
numNodes	31	2 31 3 30 2 31 5 30 3 31 5 30 6 5 23 22 11 12 10 11 12 10 11 14 8 7 6 5 23
numArcs	94	22 11 14 8 7 6 7 9 8 11 18 9 8 11 18 14 9 10 13 12 14 9 10 13 12 14 13 15
minimum	930	16 15 17 16 20 17 16 20 17 18 17 19 20 21 8 18 19 20 21 8 18 19 21 9 18 22
initial	937	14 13 15 17 19 21 9 22 14 18 22 21 24 18 29 28 29 23 24 26 22 26 27 26 29
LS	932	27 28 29 6 1 23 6 8 22 24 25 26 18 30 21 26 21 29 24 30 21 29 24 30 22 29
Prune	930	23 24 26 29 27 28 29 28 29 6 7 9 22 21 24 18 29 30 22 29 30 23 26 30 23 26
time	60.027	30 28 31 30 28 31 30 6 1
aveCost	5.70213	
aveDiff	1.80851	
stdevCost	4.24347	
stdevDiff	1.5387	
initialRouteL	203	
routeLength	201	

filename	P1715	1 2 1 19 17 3 2 4 3 2 4 14 3 4 14 3 5 6 3 5 13 8 5 13 8 5 14 6 3 17 4 19 17 4
numNodes	19	19 18 17 9 7 6 5 14 6 7 8 9 7 8 9 10 11 12 9 10 11 12 10 12 16 9 12 16 9 15
numArcs	44	13 14 16 14 17 9 15 13 14 17 11 16 15 14 19 18 17 12 17 11 16 15 14 19 1
minimum	381	
initial	383	
LS	381	
Prune	381	
time	3.016	
aveCost	5.06818	
aveDiff	1.77273	
stdevCost	4.37138	
stdevDiff	1.5055	
initialRouteL	91	
routeLength	89	

filename	P1815	1 23 22 7 3 4 8 16 12 10 9 8 16 12 10 9 8 16 17 16 18 6 5 4 8 16 18 6 5 4 12
numNodes	23	11 10 11 14 15 13 12 11 14 15 13 12 17 18 19 7 3 4 12 17 18 19 22 7 2 22 2
numArcs	37	1 23 22 21 20 17 19 22 2 19 2 19 2 22 21 20 17 19 7 5 7 2 1
minimum	361	
initial	363	
LS	361	
Prune	361	
time	2.907	
aveCost	5.41892	
aveDiff	1.48649	
stdevCost	6.47032	
stdevDiff	1.44487	
initialRouteL	81	
routeLength	81	

filename	P1915	1 2 25 24 25 24 25 26 24 26 24 26 27 25 26 27 25 30 14 13 12 10 9 6 7 10 11
numNodes	33	12 14 13 12 14 30 19 18 17 15 16 14 31 9 8 6 7 10 11 16 14 31 28 29 30 19
numArcs	54	18 20 18 20 21 19 21 23 20 21 23 22 23 22 25 30 29 31 32 25 30 32 25 30 32
minimum	464	28 29 31 32 33 5 33 32 28 31 9 8 6 1 2 3 2 4 3 5 4 3 5 4 25 2 4 25 22 23 20
initial	471	18 17 15 16 11 12 10 9 6 1
LS	464	
Prune	464	
time	10.218	
aveCost	4.73148	
aveDiff	1.38889	
stdevCost	4.36655	
stdevDiff	1.31116	
initialRouteL	119	
routeLength	119	

filename	P2015	1 4 1 7 5 4 3 2 1 7 6 5 4 39 4 46 5 7 6 5 39 12 11 10 9 8 15 4 46 15 5 39 12
numNodes	50	13 10 9 8 15 5 45 15 16 17 12 13 10 11 12 24 16 17 12 24 16 25 26 31 28 27
numArcs	98	23 22 21 20 23 24 36 13 14 9 14 15 16 25 26 31 30 29 28 27 26 32 31 30 29
minimum	1310	28 31 32 26 33 12 36 13 14 15 39 17 18 19 21 20 23 24 36 16 33 12 36 17 18
initial	1312	19 21 22 23 27 26 33 13 14 16 33 13 14 16 36 17 20 24 39 17 20 24 39 36 35
LS	1310	33 34 35 33 34 35 36 38 35 38 37 36 38 37 36 40 39 36 40 41 43 44 43 46 15
Prune	1310	39 40 41 43 46 44 45 46 44 45 48 5 45 48 5 46 48 39 42 41 46 48 50 49 47
time	11.656	48 50 49 47 48 39 42 41 46 45 15 4 3 2 1
aveCost	7.12245	
aveDiff	1.14286	
stdevCost	7.59063	
stdevDiff	1.23718	
initialRouteL	201	
routeLength	199	

filename	P2115	1 2 1 7 6 1 7 6 1 17 18 29 8 22 29 22 18 1 17 18 34 8 45 26 25 26 25 26 29
numNodes	49	22 24 26 34 22 29 22 9 10 11 12 13 9 10 11 12 13 14 13 14 13 10 12 16 11
numArcs	110	16 15 14 13 9 22 45 29 27 28 29 27 28 29 28 31 30 28 31 30 28 35 9 45 29
minimum	931	28 35 9 45 34 32 30 29 32 30 29 32 31 33 32 31 33 34 32 33 34 41 40 38 36
initial	940	35 15 14 13 10 12 16 15 35 15 35 31 35 37 38 36 35 37 38 39 31 39 40 38 39
LS	933	41 40 39 41 42 41 46 47 48 45 34 41 46 47 48 45 46 48 49 43 26 43 44 43 47
Prune	931	49 43 47 49 44 45 46 48 49 44 45 18 7 8 5 4 3 2 3 2 3 4 6 2 6 8 5 4 6 8 7 17
time	257.195	20 21 19 21 20 23 24 22 18 9 27 8 18 9 27 8 18 34 8 45 18 29 8 22 45 26 34
aveCost	4.77273	22 21 24 26 29 22 21 24 23 25 26 18 7 17 20 23 25 26 8 9 26 8 9 26 18 1
aveDiff	1.41818	
stdevCost	4.77143	
stdevDiff	1.39728	
initialRouteL	241	
routeLength	239	

filename	P2215	1 2 3 2 10 1 2 10 3 10 9 4 3 4 7 31 11 15 16 13 16 17 14 8 7 42 8 43 7 42 8
numNodes	50	45 25 24 23 22 23 19 18 31 8 7 18 19 17 18 34 20 7 11 34 27 43 26 47 9 42
numArcs	184	26 20 9 39 8 30 31 30 29 9 43 25 33 14 31 30 31 39 11 14 18 33 7 18 19 23
minimum	4256	25 24 23 25 39 34 25 39 26 33 34 25 26 20 33 8 45 25 44 31 45 34 27 39 37
initial	4269	38 37 35 28 10 9 42 25 21 20 33 8 18 39 7 6 5 4 3 4 9 14 7 6 5 4 7 11 8 9 14
LS	4259	11 33 31 42 26 21 19 17 18 33 7 45 9 18 39 7 45 9 31 20 8 44 9 39 8 43 34
Prune	4257	35 32 31 42 27 26 44 27 26 45 46 48 45 43 44 46 48 50 47 50 49 48 45 39 47
time	233.599	45 43 7 31 11 12 13 15 14 18 21 20 9 31 20 8 30 29 11 12 13 15 14 7 47 8 18
aveCost	12.2609	34 26 45 27 20 39 34 26 44 27 39 36 40 41 50 49 48 50 45 27 20 39 26 33 32
aveDiff	1.77174	28 31 47 27 43 44 46 43 39 42 45 46 43 33 34 20 7 47 8 44 9 47 25 44 31 47
stdevCost	9.5867	34 42 43 39 47 45 33 44 34 38 37 36 39 37 28 29 11 33 14 31 39 11 15 16 17
stdevDiff	1.38394	14 8 9 18 31 43 26 47 25 26 21 19 18 21 22 25 42 27 47 34 38 37 35 28 31 8
initialRouteL	383	11 34 35 32 28 29 9 43 33 39 38 37 38 37 36 40 41 47 42 33 47 42 33 47 44
routeLength	381	39 41 47 44 39 41 50 45 39 42 45 34 32 33 44 34 42 43 34 32 31 43 25 21 22
		25 33 31 45 33 39 38 37 28 10 1

filename	P2315	1 19 1 8 5 3 2 1 34 35 37 34 35 37 34 36 18 14 13 10 11 10 11 10 12 13 10
numNodes	50	12 13 11 13 11 13 15 13 16 17 18 16 15 17 16 15 17 16 17 18 14 16 14 13 16
numArcs	158	18 27 31 32 21 18 27 31 32 27 43 1 34 36 18 32 27 43 7 36 19 24 23 25 19
minimum	1781	34 36 19 34 36 22 18 32 33 29 28 30 33 29 30 33 48 4 43 7 36 22 18 38 22
initial	1785	21 18 38 22 21 20 19 43 21 20 19 43 21 27 43 22 23 26 20 25 24 25 26 20 25
LS	1781	26 21 27 43 22 23 26 21 32 33 48 4 43 32 44 4 44 18 42 22 27 43 32 44 22
Prune	1781	27 43 36 23 34 36 23 34 36 35 38 36 35 38 36 43 38 40 18 42 22 32 48 18 43
time	16.906	38 43 40 18 43 40 37 39 40 37 39 40 38 43 42 38 46 40 41 39 41 42 38 46 40
aveCost	6.26899	41 46 45 42 41 46 45 49 43 42 45 49 43 48 18 44 22 32 48 21 44 27 48 21 44
aveDiff	1.5	27 48 22 40 42 49 44 43 48 22 40 42 49 44 50 48 47 49 46 47 49 46 47 50 48
stdevCost	5.95769	47 50 49 50 44 43 1 23 24 23 25 19 7 34 4 36 1 23 7 4 5 3 2 5 6 2 5 6 3 4 5
stdevDiff	1.36757	7 4 19 4 21 12 4 21 12 4 23 7 5 8 6 3 4 23 24 19 7 9 8 7 9 8 7 34 4 27 28 27
initialRouteL	335	30 29 28 30 31 30 32 4 27 30 32 4 36 1 8 6 2 1 6 1
routeLength	335	

filename	P2415	1 2 1 3 5 20 16 7 8 9 6 7 8 9 7 17 10 11 16 10 11 16 17 29 20 25 31 20 38
numNodes	41	16 17 29 23 22 21 23 36 16 15 17 20 16 38 29 31 30 11 25 27 29 30 7 13 11
numArcs	125	12 14 13 15 14 16 7 31 11 25 28 25 7 11 7 38 10 25 24 8 36 11 38 23 31 24
minimum	1903	11 17 24 26 29 36 17 38 29 23 31 24 11 20 25 7 31 11 17 24 36 31 17 30 11
initial	1905	20 24 8 29 30 10 31 32 33 31 16 30 28 27 29 31 30 10 31 17 30 7 17 10 9 6
LS	1903	7 13 11 12 13 12 14 13 15 14 16 10 8 10 9 7 38 10 25 24 16 30 28 27 29 20
Prune	1903	31 16 25 17 25 27 26 27 29 8 36 11 38 23 22 21 23 36 16 15 17 20 19 22 20
time	10.328	24 26 29 36 17 38 33 31 32 34 33 32 34 35 32 35 33 34 35 33 38 37 36 20 38
aveCost	8.312	37 36 24 16 25 31 36 39 36 39 40 38 41 40 37 39 40 38 41 40 37 39 36 20 19
aveDiff	1.632	18 3 18 21 19 18 21 19 22 20 8 4 5 4 2 3 5 20 8 4 2 3 18 3 1
stdevCost	6.64399	
stdevDiff	1.29945	
initialRouteL	257	
routeLength	257	

filename	hd115	1 2 46 2 3 14 15 14 42 16 18 17 1 2 30 3 14 4 6 10 11 6 10 15 13 35 30 53
numNodes	70	63 62 51 50 51 50 51 66 58 57 58 37 25 26 33 34 33 34 33 38 21 38 36 37 36
numArcs	141	37 66 51 52 53 63 70 69 70 69 70 62 61 53 63 62 61 67 61 53 63 69 50 52 69
minimum	11453	50 52 69 63 70 62 51 52 53 16 18 17 16 42 48 42 20 19 34 27 25 26 27 34 33
initial	11484	38 21 22 31 23 32 31 23 29 30 47 48 41 42 3 9 1 12 2 30 7 8 7 18 9 30 57 42
LS	11466	47 57 16 47 42 3 2 49 45 44 40 39 44 46 40 39 43 39 40 39 43 44 43 60 59
Prune	11454	67 55 54 57 58 37 66 58 57 16 47 48 13 12 13 35 41 20 19 34 33 26 27 25 37
time	541.749	25 37 36 38 21 22 31 23 24 31 23 24 28 24 31 23 32 22 32 31 23 29 19 29 28
aveCost	41.5957	29 30 57 42 14 4 5 11 6 5 4 6 5 11 10 11 10 15 13 48 41 20 35 42 20 28 20
aveDiff	3.21986	35 41 42 35 30 53 16 8 7 18 8 7 30 47 57 54 65 56 64 65 56 54 55 61 47 61
stdevCost	50.0301	68 59 55 61 68 64 56 54 65 64 68 67 60 59 67 55 68 59 55 68 64 68 67 60 43
stdevDiff	2.56571	39 44 46 40 45 44 40 45 46 49 17 1 12 2 49 45 46 49 17 16 8 7 9 18 8 7 8 7
initialRouteL	325	9 30 3 4 3 9 1
routeLength	319	

filename	hd215	1 4 1 4 1 2 41 46 41 46 41 42 5 32 43 36 43 42 32 69 68 54 53 26 50 26 31
numNodes	70	5 7 8 47 33 46 41 32 69 68 26 53 31 13 31 27 25 26 53 31 5 32 35 5 69 50 5
numArcs	162	53 22 26 31 27 28 30 25 26 22 17 15 16 18 21 19 29 20 19 29 30 27 25 28 29
minimum	13530	30 27 28 30 25 28 29 20 19 21 18 16 15 23 24 23 22 17 20 17 24 11 13 53 26
initial	13574	69 7 49 50 7 49 34 49 48 38 34 48 36 43 37 44 45 40 37 44 45 42 32 33 46
LS	13533	41 4 12 13 10 11 24 15 17 24 15 23 18 16 18 16 23 18 16 23 22 11 13 14 13
Prune	13533	26 69 7 47 46 47 7 35 34 49 5 69 42 35 48 36 43 37 36 43 40 44 39 44 40 37
time	798.459	36 43 40 45 42 43 32 33 46 41 42 5 22 13 53 22 11 9 10 14 11 6 5 13 26 5 13
aveCost	42.9877	10 11 9 10 14 11 6 5 50 52 67 51 52 62 66 65 66 70 39 38 36 35 36 34 49 69
aveDiff	3.61728	50 52 51 67 70 64 63 38 67 52 51 50 7 35 5 53 58 66 70 39 38 67 70 64 60
stdevCost	44.0862	63 59 60 59 60 64 63 59 57 60 59 57 55 61 62 52 51 52 51 68 54 62 66 58 56
stdevDiff	2.37823	61 56 55 61 62 58 54 62 58 54 53 58 56 61 56 55 57 60 63 38 34 49 34 49 48
initialRouteL	367	49 50 51 68 26 5 7 8 9 8 47 33 46 41 32 35 34 49 69 42 35 48 38 36 43 36 34
routeLength	367	48 49 5 22 13 3 12 13 3 2 4 1 3 12 4 1 3 2 41 65 6 65 4 2 65 4 1 2 65 41 4 1

filename	hd315	1 9 3 4 3 9 10 11 12 13 24 23 17 19 17 22 18 21 18 17 23 26 13 24 23 26 13
numNodes	68	28 29 32 30 31 30 6 31 68 67 63 60 65 63 64 57 65 19 20 27 2 32 13 5 7 8 33
numArcs	153	34 55 30 6 33 34 39 34 55 30 31 48 44 45 44 45 47 68 67 63 60 55 37 60 55
minimum	12449	37 60 61 51 53 25 23 26 16 53 25 23 26 16 53 52 51 53 52 51 59 60 61 51 59
initial	12487	61 59 52 59 60 64 25 26 24 55 50 49 31 30 27 32 30 49 31 48 44 41 42 43 41
LS	12454	42 43 42 44 41 43 42 44 45 48 45 47 40 46 36 46 43 46 36 46 40 47 68 48 45
Prune	12452	48 45 36 35 34 60 38 55 31 30 49 56 58 50 56 58 49 6 49 24 31 54 30 27 29
time	850.229	20 65 19 21 18 21 18 17 22 18 22 19 21 18 22 19 20 29 32 28 13 12 11 4 3
aveCost	41.8856	15 4 3 4 3 15 4 3 35 14 36 35 34 60 38 39 37 38 39 37 38 40 37 38 40 37 38
aveDiff	3.44444	55 31 30 24 20 65 64 60 65 63 67 66 62 63 67 66 62 63 64 25 26 24 55 54 49
stdevCost	36.9781	50 56 57 58 49 24 54 49 56 57 65 64 57 58 50 55 54 24 31 68 48 45 36 14 15
stdevDiff	2.69807	36 14 15 35 15 36 14 12 14 4 3 35 14 36 14 4 3 1 9 10 11 4 3 1 16 2 1 16 5
initialRouteL	347	6 7 8 33 6 31 54 30 24 20 27 29 28 2 32 13 5 7 6 24 16 5 6 24 16 13 2 16 13
routeLength	345	2 27 32 28 2 1

filename	hd415	1 18 6 2 1 4 5 18 2 64 57 58 59 54 51 48 47 49 56 47 56 50 49 51 49 56 55
numNodes	88	50 51 54 59 54 48 47 49 50 49 50 51 48 47 56 50 55 79 74 77 78 71 69 84 72
numArcs	171	69 70 21 6 64 58 59 57 59 54 48 47 56 47 56 55 79 77 79 74 73 75 74 73 75
minimum	12111	74 77 78 71 69 84 72 84 83 72 84 83 86 83 86 78 75 80 68 80 62 80 81 66 85
initial	12162	66 85 38 76 85 66 65 87 67 66 67 65 66 67 66 81 82 68 80 68 80 81 88 87 82
LS	12116	88 87 82 88 81 82 68 67 65 87 67 68 80 75 78 86 71 86 83 72 69 70 21 61 60
Prune	12112	40 61 53 43 40 61 63 60 70 61 63 60 70 76 70 40 12 23 22 21 62 60 21 61 60
time	1972.53	40 62 70 40 62 70 61 53 8 9 19 20 19 53 8 7 4 5 18 2 64 57 58 1 18 6 64 58 1
aveCost	36.5585	4 3 5 17 63 52 43 25 43 40 7 14 23 13 22 21 62 60 21 68 40 39 31 34 33 41
aveDiff	3.63158	34 41 34 33 31 32 33 31 32 33 41 35 34 41 30 35 34 41 30 29 46 45 42 27 26
stdevCost	49.1893	36 38 76 85 38 44 45 42 37 26 36 37 28 36 38 44 45 42 27 26 28 27 24 39 31
stdevDiff	2.61551	34 41 35 30 29 46 45 42 37 28 36 37 26 28 27 24 25 43 25 52 53 25 52 53 43
initialRouteL	393	40 39 32 25 43 17 63 52 43 17 20 19 53 25 24 39 32 25 43 40 7 14 23 13 22
routeLength	391	13 21 6 22 13 12 14 3 5 17 20 15 11 16 15 11 10 11 10 16 15 20 15 20 19 20
		19 8 9 10 11 16 11 9 10 16 11 9 19 8 7 4 3 6 22 13 21 68 40 12 23 22 13 12
		14 3 6 2 1

filename	hd515	1 2 3 4 46 55 52 50 47 64 49 64 47 64 47 53 54 60 54 60 54 61 62 56 60 51
numNodes	86	60 62 59 61 62 59 61 44 57 81 13 12 14 3 12 14 1 2 17 16 19 15 16 17 6 30
numArcs	178	12 18 28 29 34 28 30 22 42 26 37 25 26 37 35 37 38 35 37 35 33 34 65 4 85
minimum	10808	34 65 4 85 34 84 85 49 85 65 49 64 53 50 48 46 43 45 63 45 58 63 1 14 3 12
initial	10882	14 2 17 6 13 6 32 22 20 31 20 31 20 21 23 21 24 23 39 23 31 20 32 31 30 18
LS	10810	15 16 5 19 16 19 16 5 19 15 5 6 32 31 30 18 28 12 18 10 9 18 10 9 18 15 5 6
Prune	10808	13 6 30 12 28 30 22 42 26 41 42 12 4 49 64 50 51 60 54 86 80 76 53 50 48
time	3670.25	47 48 47 48 64 47 53 54 86 80 76 53 83 66 67 65 66 67 83 86 80 77 74 72 73
aveCost	31.7219	69 82 79 67 65 49 64 53 83 66 84 85 65 66 84 71 69 68 71 68 70 40 70 29 40
aveDiff	3.84831	36 35 37 38 33 36 38 33 36 38 35 33 34 84 71 69 79 78 73 72 78 79 71 79 67
stdevCost	36.5932	83 86 80 77 74 75 76 77 75 76 77 75 74 72 78 73 69 82 81 82 81 82 86 82 79
stdevDiff	2.78933	69 68 70 29 34 28 29 40 70 40 36 35 37 25 26 41 25 27 41 25 27 39 27 24 39
initialRouteL	417	27 24 39 23 31 20 32 22 20 21 24 23 39 27 39 27 41 26 41 42 12 14 1 14 2 3
routeLength	415	4 46 55 59 57 58 63 45 58 57 44 43 55 59 57 81 13 12 4 49 64 47 49 64 49
		64 50 51 56 52 51 60 62 56 60 51 60 54 61 44 8 81 11 44 8 10 7 8 11 7 9 11
		7 8 10 7 9 11 8 81 11 44 43 55 52 51 56 52 50 47 48 64 47 49 64 47 48 46 43
		45 63 1

filename	hd615	1 12 2 1 12 2 1 25 1 31 25 30 23 29 26 23 25 32 30 31 33 11 13 14 13 2 11
numNodes	88	33 34 67 68 69 68 69 68 75 76 67 68 75 84 32 84 32 84 75 85 75 76 67 69 68
numArcs	181	76 69 68 76 69 74 34 67 69 74 34 73 67 73 74 49 48 11 40 11 40 11 41 33 34
minimum	10012	73 74 49 48 14 13 2 11 13 14 48 11 41 33 40 35 36 12 4 3 18 3 16 5 3 16 5 3
initial	10058	4 18 3 18 10 18 16 5 15 16 5 15 16 5 22 15 17 6 7 72 32 25 30 23 29 24 29
LS	10025	42 29 26 27 39 38 39 37 35 37 35 37 8 7 72 32 30 31 33 40 35 36 12 4 18 16
Prune	10021	5 22 15 17 6 8 12 38 8 12 38 8 36 38 35 37 8 36 38 35 37 27 39 37 27 85 26
time	1966.52	27 85 75 85 88 82 80 82 81 80 83 50 51 50 74 78 49 51 54 55 54 56 54 60 50
aveCost	28.9558	74 78 49 51 54 60 50 78 79 70 72 71 70 72 71 70 77 78 79 70 77 79 71 77 79
aveDiff	3.67956	71 77 78 88 82 81 80 86 80 86 81 87 44 43 41 40 41 40 41 43 66 43 83 80 86
stdevCost	27.1341	81 87 44 62 58 57 44 62 58 61 56 55 60 51 60 55 61 56 55 61 56 65 53 20 9
stdevDiff	2.70401	10 19 4 19 52 9 10 19 52 9 10 20 9 21 17 22 21 17 22 21 9 10 20 19 53 20 19
initialRouteL	403	53 52 20 52 49 52 53 64 61 56 65 53 64 61 58 66 63 46 47 45 46 47 45 46 59
routeLength	399	45 65 59 45 65 59 47 63 46 59 47 63 64 66 63 64 66 58 57 62 87 57 62 87 57
		44 43 83 50 78 88 84 85 88 84 85 26 28 24 23 26 28 24 23 25 32 25 31 27 39
		36 8 6 7 1 31 27 39 36 8 7 1

filename	hd715	1 26 40 38 39 36 38 28 29 28 26 27 30 32 27 30 43 31 30 32 43 44 49 50 46
numNodes	100	31 32 43 44 49 50 46 50 47 33 47 34 35 45 39 45 37 39 45 39 45 34 33 35 37
numArcs	188	36 38 28 29 57 58 74 68 23 8 63 62 22 64 62 22 4 75 63 4 3 70 61 52 60 52
minimum	8345	61 69 70 69 70 75 70 75 70 69 75 70 69 61 60 52 56 60 52 51 66 65 55 56 54
initial	8395	55 53 65 92 90 92 87 88 91 90 87 86 88 93 80 95 78 95 80 67 41 14 16 25 14
LS	8345	15 25 14 15 24 23 8 63 62 22 21 22 64 62 22 4 75 70 69 75 63 4 3 6 7 20 73
Prune	8345	57 58 74 72 73 57 59 29 28 26 40 38 39 45 34 33 35 37 39 36 40 42 48 29 57
time	1253.67	59 1 2 10 11 2 10 1 59 72 71 58 93 94 79 99 89 44 99 89 83 82 83 67 68 41
aveCost	23.3723	42 48 29 59 72 71 74 72 73 71 58 93 80 77 79 76 78 95 94 78 95 94 79 99 44
aveDiff	3.68085	89 83 82 84 91 100 84 100 96 85 98 96 85 82 98 97 85 98 97 96 97 96 97 85
stdevCost	28.2297	82 84 100 84 91 90 87 88 93 94 78 81 76 78 81 76 77 81 95 80 95 78 95 80
stdevDiff	2.69638	77 79 76 77 81 95 80 67 68 41 14 16 9 6 5 16 9 10 1 26 27 46 31 30 43 31 32
initialRouteL	431	27 46 50 47 33 47 34 35 45 37 36 40 42 9 6 5 7 64 21 3 6 7 20 19 2 11 13 18
routeLength	431	12 13 18 12 11 13 17 18 19 17 19 17 19 20 48 20 73 71 74 68 23 15 24 25 24
		23 15 25 24 25 24 8 5 16 25 24 8 5 7 64 21 3 70 61 60 52 60 52 56 54 66 51
		54 66 51 66 65 55 53 66 51 53 66 51 54 55 56 60 52 51 53 65 92 87 86 88 91
		100 96 98 96 98 82 83 67 41 42 9 10 11 2 11 12 13 17 18 19 2 1

filename	hd815	1 6 5 15 5 2 6 15 7 4 7 49 51 50 52 51 54 55 56 54 55 61 62 55 61 67 62 55
numNodes	95	56 62 67 62 67 65 60 49 46 60 56 54 59 57 58 61 67 65 63 64 48 70 1 6 15
numArcs	193	34 9 36 24 34 21 13 11 89 90 95 58 61 62 67 66 47 11 89 90 83 88 73 88 83
minimum	10786	88 77 92 93 69 68 69 71 94 93 94 92 72 91 95 58 66 65 60 49 51 54 59 50 53
initial	10818	68 69 71 72 73 71 94 38 33 43 41 40 28 29 27 28 40 39 41 39 37 41 39 41 39
LS	10786	28 29 27 30 45 30 29 45 30 31 27 28 40 37 41 39 28 40 28 40 37 38 33 32 42
Prune	10786	76 33 32 15 45 30 45 30 31 26 25 35 34 32 42 43 41 40 39 37 38 32 15 34 32
time	3686.47	21 17 20 8 18 20 19 9 8 17 8 18 17 20 19 18 17 21 15 7 5 2 6 5 7 49 46 48
aveCost	29.1477	64 70 1 2 19 18 20 8 9 16 89 90 83 75 74 76 33 43 76 74 77 75 81 79 78 82
aveDiff	3.40415	86 84 85 57 91 95 87 84 85 86 87 85 86 87 85 57 91 73 72 91 73 88 77 92 93
stdevCost	35.3273	3 68 53 68 53 4 7 4 3 93 69 3 4 68 69 3 68 53 52 59 50 53 4 68 69 68 53 52
stdevDiff	2.68799	51 50 52 59 57 58 66 47 46 48 70 1 2 19 9 36 24 22 23 25 44 26 23 25 35 36
initialRouteL	425	22 35 36 22 23 26 44 26 44 31 26 25 44 31 27 30 29 45 24 22 35 34 21 12 10
routeLength	425	13 12 13 11 14 89 90 42 43 76 42 90 95 87 84 82 86 84 82 79 78 80 81 78 80
		81 79 80 88 83 75 74 77 75 81 78 82 79 80 88 73 71 72 92 94 38 32 21 13 10
		14 89 63 47 14 12 21 15 16 15 45 24 34 9 16 15 16 89 63 47 14 10 11 14 12
		10 11 47 46 60 56 62 67 66 65 63 64 70 1

filename	hd915	1 2 3 5 4 5 23 61 14 23 14 12 13 11 9 11 13 24 11 20 6 21 4 6 21 4 6 24 11
numNodes	96	20 6 24 20 21 22 4 22 5 23 60 12 13 24 20 21 22 15 14 12 15 22 15 23 60 12
numArcs	189	15 23 61 55 53 51 54 52 51 54 52 51 55 53 51 55 64 69 71 65 54 53 52 65 54
minimum	9041	53 52 65 71 65 71 72 65 71 72 67 70 50 95 66 70 50 49 47 16 38 37 47 38 42
initial	9071	44 31 30 93 92 96 90 88 91 58 91 49 72 65 71 61 60 14 61 60 57 60 14 15 22
LS	9053	5 3 1 10 8 9 10 8 44 46 45 7 3 1 2 3 7 8 9 10 79 36 27 25 26 33 26 34 32 25
Prune	9047	28 36 27 25 28 36 92 84 93 84 75 74 73 40 83 82 81 82 81 82 87 86 78 80 87
time	2918.6	86 77 79 78 77 79 78 77 80 86 77 80 86 78 80 87 96 93 84 75 74 73 81 82 58
aveCost	25.0106	87 57 59 68 63 69 67 66 50 95 66 70 95 89 90 88 91 49 72 67 70 95 89 94 88
aveDiff	3.37566	89 94 88 89 90 96 87 82 76 75 85 76 74 73 81 82 57 56 18 19 17 16 2 17 19
stdevCost	24.1642	17 18 17 19 56 18 19 56 16 37 39 40 83 82 76 75 85 76 74 73 83 81 82 57 56
stdevDiff	2.57412	16 37 39 35 30 29 32 34 48 34 33 29 32 34 33 29 30 93 84 93 92 96 93 84 85
initialRouteL	419	74 73 40 39 35 27 25 26 34 32 25 27 28 41 1 10 79 36 92 84 85 74 73 83 81
routeLength	413	82 58 68 63 62 59 58 68 63 62 59 58 87 57 59 68 63 64 62 68 63 64 69 71 61
		55 64 62 68 63 69 67 66 50 49 47 16 2 17 16 38 37 47 38 42 44 31 29 31 46
		31 46 43 7 8 44 46 45 7 43 42 45 43 42 45 43 46 31 30 35 41 35 27 28 41 35
		41 1

filename	hg115	1 3 4 6 7 10 7 40 35 25 35 32 34 39 41 57 48 47 46 41 46 40 35 36 29 30 37
numNodes	60	36 38 40 39 23 22 15 27 23 22 15 5 8 3 8 14 12 13 16 26 16 26 27 15 14 16
numArcs	105	26 32 34 55 45 55 45 44 33 44 41 57 48 47 49 50 59 60 58 57 55 45 44 41 46
minimum	420	41 46 47 49 50 52 54 43 53 38 42 37 36 38 36 38 53 51 35 36 29 20 11 18 17
initial	460	28 17 22 9 5 9 5 4 6 7 10 11 18 24 20 24 20 24 25 29 30 19 11 19 11 19 30
LS	441	37 42 38 40 39 41 46 40 28 17 7 10 11 18 24 22 9 7 40 28 25 24 22 16 26 32
Prune	431	31 33 34 39 23 28 17 28 49 51 52 51 35 32 31 21 31 21 13 16 26 27 23 28 17
time	513.435	7 9 22 17 15 5 4 3 8 3 8 14 12 2 1 3 8 2 8 5 9 22 16 26 21 31 33 34 55 56 58
aveCost	2.68095	56 58 56 55 57 58 60 58 60 48 50 59 60 48 50 52 54 43 42 43 53 51 49 28 25
aveDiff	1.93333	24 25 29 20 11 18 17 15 14 16 26 21 13 12 2 1
stdevCost	2.17302	
stdevDiff	1.88865	
initialRouteL	263	
routeLength	253	

filename	hg215	1 2 7 9 19 8 7 8 17 6 17 15 16 18 31 18 28 20 29 20 29 58 65 59 64 55 57 58
numNodes	68	65 59 57 59 64 55 57 58 50 49 34 33 30 32 30 21 29 20 57 55 48 43 41 43 40
numArcs	119	53 54 56 64 55 48 56 48 56 48 53 54 56 64 55 42 41 31 32 30 33 34 33 35 36
minimum	450	51 49 43 48 43 40 39 16 39 41 42 41 42 34 44 67 66 65 58 50 60 51 49 34 33
initial	488	35 37 38 46 38 37 26 37 26 37 38 37 45 52 47 62 61 68 63 62 61 60 51 52 45
LS	457	46 47 46 38 46 47 46 38 27 26 37 35 14 24 25 27 25 24 22 13 11 29 20 57 59
Prune	453	57 55 42 34 44 23 22 13 21 29 58 65 66 50 60 67 68 61 60 67 66 50 49 43 48
time	2118.86	53 40 39 41 31 32 42 32 30 32 30 21 23 26 24 14 5 22 23 26 37 45 36 51 52
aveCost	2.65126	47 62 61 52 45 52 45 36 44 67 68 63 62 61 52 45 46 38 27 25 27 26 24 22 23
aveDiff	2.15966	35 14 5 4 3 2 7 6 17 18 17 15 16 18 28 8 28 32 28 20 19 11 10 12 5 22 13 12
stdevCost	2.02511	5 4 3 9 19 11 10 9 10 12 13 21 23 35 36 44 23 22 13 11 29 20 19 8 7 6 1 2 3
stdevDiff	1.81489	9 7 8 17 6 1 2 1 2 3 2 1
initialRouteL	301	
routeLength	297	

filename	hg315	1 2 4 3 5 3 4 10 11 22 21 35 44 38 44 35 44 35 36 37 36 48 49 64 60 59 61
numNodes	64	47 46 42 41 28 27 28 27 25 17 16 5 3 5 6 7 18 31 28 27 19 18 31 26 39 37 36
numArcs	116	37 36 48 49 59 57 56 54 43 33 14 9 11 30 24 17 16 15 23 22 23 30 49 38 37
minimum	455	24 17 16 15 23 17 16 5 6 7 6 7 8 19 8 19 8 7 18 13 25 26 31 26 31 40 41 28
initial	489	27 25 26 31 40 41 42 51 47 50 38 50 38 50 39 40 46 42 51 52 61 62 60 62 63
LS	462	52 61 47 50 39 40 46 38 59 57 56 54 43 44 35 33 32 34 43 33 32 20 32 34 43
Prune	458	44 45 56 54 53 34 53 55 58 64 60 59 61 62 63 52 51 52 51 47 46 38 59 50 49
time	1546.99	59 23 59 50 49 64 58 57 48 45 56 54 53 55 58 57 48 45 44 38 44 35 33 14 21
aveCost	2.69397	14 21 35 36 29 22 29 30 49 38 37 36 29 30 23 17 16 13 25 17 16 13 12 10 5
aveDiff	1.93966	3 1 2 4 9 4 10 11 22 21 14 20 2 20 14 9 11 15 11 15 11 30 24 26 39 37 24 26
stdevCost	1.99921	31 28 27 19 18 13 12 6 12 6 12 10 5 3 1
stdevDiff	1.88135	
initialRouteL	279	
routeLength	275	

filename	hg415	1 3 5 2 21 22 25 22 25 22 24 35 36 8 10 13 12 14 15 16 19 20 18 16 18 32 31
numNodes	82	29 17 29 18 29 17 16 18 16 19 20 44 43 32 31 30 27 13 17 29 27 13 12 14 17
numArcs	148	29 31 39 50 57 53 51 50 48 46 49 47 45 46 49 46 49 56 38 40 38 41 38 40 48
minimum	553	56 38 36 37 10 26 28 30 42 41 28 30 31 39 50 57 53 51 43 44 52 51 43 39 42
initial	598	41 28 26 11 26 10 9 11 12 14 17 16 19 20 18 32 43 39 42 67 65 64 58 62 80
LS	560	75 63 55 49 47 54 55 62 48 62 55 54 61 74 75 80 77 76 58 76 63 61 74 75 63
Prune	556	55 49 56 57 41 65 64 57 41 38 41 65 64 57 37 28 30 27 26 24 35 40 38 36 8
time	1281.51	10 13 17 29 27 26 11 26 24 7 4 6 8 7 5 3 4 7 24 35 40 35 24 35 36 37 10 9 6
aveCost	2.61486	9 11 12 14 12 14 15 16 19 20 44 52 59 52 51 50 69 71 72 73 60 59 53 71 69
aveDiff	2.04054	71 60 73 79 78 68 70 72 73 79 78 82 66 76 63 61 54 47 54 47 45 33 23 21 22
stdevCost	1.97629	5 22 5 2 1 3 4 6 8 7 5 22 24 35 25 23 33 23 33 34 25 34 40 48 56 58 62 80
stdevDiff	1.83022	75 80 77 81 82 66 65 66 65 66 68 66 76 77 81 64 58 56 57 37 28 30 42 67 65
initialRouteL	355	64 81 82 78 68 70 69 67 68 70 69 67 68 70 72 71 60 59 52 59 53 71 69 50 48
routeLength	349	46 45 33 34 46 45 46 34 40 35 25 23 21 2 1

filename	hg515	1 3 4 3 12 2 1 3 12 13 4 5 6 5 16 17 7 19 7 6 8 9 7 17 18 19 7 6 8 9 28 9 20
numNodes	92	21 11 10 8 10 20 21 11 10 20 30 32 22 21 11 32 22 21 11 32 33 34 22 34 57
numArcs	165	56 54 42 40 31 28 9 7 19 28 9 20 30 32 33 34 57 56 54 42 41 31 28 9 28 30
minimum	592	41 31 29 19 28 30 41 33 56 65 66 57 66 67 75 67 75 67 76 85 75 74 55 63 52
initial	616	50 48 38 39 29 31 40 31 40 53 40 39 40 42 41 33 56 65 66 67 76 87 92 85 75
LS	599	74 55 63 53 40 39 29 37 27 26 15 13 4 5 16 17 18 19 29 37 27 26 15 14 12
Prune	593	13 15 14 12 13 16 26 18 37 38 39 40 53 54 55 65 75 85 76 87 92 85 84 74 64
time	1336.02	71 52 51 39 51 39 51 48 38 44 35 15 35 25 14 23 2 12 13 16 26 18 37 38 44
aveCost	2.4697	35 25 14 23 24 25 36 43 24 25 36 43 36 44 36 45 46 44 46 45 60 58 43 36 45
aveDiff	1.75152	60 58 59 77 59 77 68 62 59 77 68 62 69 61 49 47 46 61 49 47 48 47 27 35 27
stdevCost	1.86043	35 27 47 46 61 60 62 69 79 81 70 49 50 48 47 48 51 48 51 53 54 55 65 75 85
stdevDiff	1.74566	84 74 64 71 52 51 53 63 64 73 72 71 70 49 50 52 63 64 73 84 86 84 86 91 73
initialRouteL	379	84 86 92 86 92 86 91 73 91 90 72 73 91 90 72 82 72 71 70 69 79 81 82 50 82
routeLength	373	72 82 89 83 81 82 89 90 89 83 88 80 68 79 88 80 78 77 78 77 78 80 68 79 88
		83 81 70 69 61 60 62 59 58 43 24 23 2 1

filename	hg615	1 3 5 16 18 19 37 19 27 10 18 10 8 7 6 25 26 34 50 49 33 45 33 45 47 58 48
numNodes	91	67 68 67 69 58 59 51 50 49 33 34 36 35 37 19 27 10 8 9 8 7 6 25 6 4 24 15
numArcs	162	23 45 23 15 23 32 24 15 23 31 44 45 23 31 14 2 5 15 23 45 33 32 26 34 36
minimum	574	61 51 71 60 59 51 71 60 81 70 69 60 59 49 47 46 48 67 69 60 81 70 82 80 68
initial	623	70 82 83 85 86 88 87 76 74 54 52 53 64 53 40 41 30 42 22 29 30 42 43 42 43
LS	578	56 55 56 57 78 79 91 90 77 79 91 90 88 90 77 89 76 74 63 72 71 73 85 86 84
Prune	576	73 84 86 88 90 88 87 63 62 37 19 37 19 17 35 26 32 33 32 33 34 50 52 53 55
time	2124.17	41 43 56 57 65 57 78 66 77 79 78 66 77 89 76 89 75 74 63 72 84 87 76 89 75
aveCost	2.46296	74 54 64 65 66 75 64 53 55 65 66 75 64 53 40 38 36 61 72 84 87 63 62 61 51
aveDiff	1.85185	50 52 38 28 39 40 38 36 35 26 25 17 16 7 16 18 10 12 9 8 9 11 13 11 29 20
stdevCost	1.84136	12 9 11 29 20 39 28 37 62 54 52 38 28 27 20 12 21 12 10 18 19 17 35 37 28
stdevDiff	1.8061	27 20 39 30 39 30 41 43 42 22 21 12 21 13 11 13 21 22 21 22 29 30 39 40 41
initialRouteL	377	55 65 64 53 64 54 62 37 28 37 62 54 62 61 72 71 73 85 83 81 73 81 83 82 80
routeLength	375	68 70 69 58 59 49 47 46 44 45 47 58 48 46 44 31 44 31 14 15 14 15 14 2 1 3
		4 24 25 17 16 5 16 5 15 23 32 24 25 6 4 3 5 2 1

filename	hg715	1 2 5 27 26 28 29 27 19 30 40 43 55 57 59 57 58 70 82 71 72 71 62 64 72 74
numNodes	97	73 74 86 85 72 74 86 85 97 94 84 94 93 83 84 85 72 64 62 58 68 44 43 55 57
numArcs	174	59 57 48 44 42 40 20 19 30 29 39 51 50 51 53 41 30 40 20 8 10 13 10 9 11
minimum	557	13 11 12 15 21 22 24 36 46 61 63 73 64 62 58 70 82 71 72 71 62 60 59 57 58
initial	604	68 78 69 79 80 78 80 81 70 78 80 96 91 79 80 78 69 56 68 44 42 40 43 41 39
LS	566	38 50 52 65 75 87 75 77 88 87 75 87 75 77 88 90 91 96 92 93 83 82 59 49 45
Prune	562	46 47 37 36 35 45 35 45 46 47 63 73 64 61 60 59 57 48 33 34 49 45 35 34 32
time	319.44	15 12 14 21 18 24 25 23 17 16 18 23 25 37 36 35 22 32 15 13 10 42 33 31 13
aveCost	2.27586	11 13 15 21 22 24 25 37 47 63 61 60 45 35 34 32 31 13 15 13 10 13 10 9 6 9
aveDiff	1.82759	6 9 11 12 14 12 14 16 14 21 18 23 17 16 18 24 36 46 61 64 62 64 62 60 45
stdevCost	1.78396	35 22 32 31 20 8 10 42 33 34 49 48 44 43 41 30 29 27 19 7 5 3 5 3 26 28 38
stdevDiff	1.7889	28 29 39 38 28 38 50 51 54 51 50 52 54 52 65 67 54 66 76 69 79 89 90 88 90
initialRouteL	399	91 79 89 77 67 76 89 77 67 76 89 90 88 87 75 65 67 54 66 56 55 53 66 56 55
routeLength	395	53 41 39 51 53 66 76 69 56 68 78 80 96 92 81 70 78 80 81 83 81 92 93 83 84
		71 84 85 97 95 86 95 86 95 97 94 93 83 82 59 49 48 33 31 20 19 7 4 6 8 7 5
		3 1 2 4 6 8 7 4 2 5 27 26 3 1

filename	hg815	1 3 16 3 16 18 31 39 46 30 29 2 1 3 16 2 29 2 29 18 19 32 31 30 46 51 52 54
numNodes	100	47 32 33 40 34 40 22 23 25 23 9 8 6 5 6 7 9 11 25 23 9 11 25 35 24 11 10 12
numArcs	180	13 12 24 11 10 12 24 26 27 35 44 45 63 62 60 61 60 61 74 61 82 80 81 83 84
minimum	572	75 74 61 59 80 59 80 59 57 55 58 59 58 56 58 56 70 53 69 77 88 89 78 77 88
initial	622	89 78 79 90 91 99 79 72 56 70 53 51 53 51 53 69 71 69 77 78 71 70 52 39 47
LS	583	48 33 40 41 48 55 54 56 58 73 90 79 72 73 90 81 83 82 75 74 61 82 75 85 94
Prune	575	84 100 84 75 85 86 85 94 95 97 98 87 95 97 96 97 96 94 84 83 93 100 96 94
time	376.533	95 86 85 86 76 68 66 50 49 45 63 65 49 37 27 35 24 26 13 14 13 14 15 28 27
aveCost	2.25	26 13 14 15 26 15 28 38 37 27 28 38 37 44 45 43 62 74 61 74 64 63 65 66 68
aveDiff	1.68889	67 65 66 50 38 50 49 37 44 36 25 23 34 40 34 42 43 62 74 64 67 86 85 64 67
stdevCost	1.72683	65 49 45 43 36 34 40 22 7 9 8 10 8 6 7 20 21 22 23 34 40 41 42 43 36 25 23
stdevDiff	1.72977	25 35 44 36 34 42 60 57 41 42 60 57 55 58 73 80 82 83 84 83 93 92 81 92 91
initialRouteL	415	90 81 92 81 92 93 100 96 97 98 87 76 87 95 86 76 68 67 86 85 64 63 62 60
routeLength	407	61 59 57 41 48 33 21 19 17 16 18 31 39 47 32 31 30 46 51 52 54 47 48 55 54
		56 72 71 72 73 80 81 92 91 99 89 78 79 99 89 78 71 70 52 39 46 30 29 18 19
		17 4 3 4 5 20 17 4 3 4 5 6 5 20 21 19 32 33 21 22 7 20 17 16 2 1

filename	hg915	1 2 17 18 28 37 50 49 51 73 75 67 66 54 66 67 69 54 66 67 87 89 90 89 90
numNodes	97	91 78 80 82 79 59 56 55 31 40 30 40 21 20 22 12 24 32 43 57 43 41 55 53 54
numArcs	175	69 70 68 54 69 77 90 89 88 89 87 76 74 73 74 86 88 76 75 76 87 77 78 70 68
minimum	588	41 39 38 37 48 49 48 35 28 37 48 35 17 35 28 18 19 36 30 39 38 37 50 49 48
initial	629	49 51 65 50 52 53 54 69 70 71 56 55 57 58 60 46 60 63 64 62 60 62 47 45 47
LS	590	46 60 46 44 58 61 81 93 97 96 82 79 71 56 68 41 43 41 39 38 52 66 67 66 54
Prune	588	66 65 51 73 74 86 88 76 74 73 75 65 51 65 50 52 53 39 38 36 30 40 41 55 31
time	1678.98	23 22 23 22 23 31 32 42 34 45 27 26 27 33 34 45 27 33 34 46 44 58 61 81 72
aveCost	2.31714	63 61 59 56 68 54 69 77 90 91 78 70 71 80 92 96 97 94 97 94 83 72 84 85 83
aveDiff	1.82286	93 82 79 71 80 92 91 92 91 92 96 82 79 81 72 84 64 84 85 83 93 82 79 81 93
stdevCost	1.72693	97 94 95 85 95 85 95 94 83 72 63 61 59 57 43 57 55 53 39 38 36 28 18 19 36
stdevDiff	1.67248	28 18 5 7 4 3 5 3 5 3 1 2 5 2 17 18 28 18 5 7 4 6 8 10 11 10 8 20 9 6 8 20 9
initialRouteL	417	7 19 29 30 40 30 40 41 43 44 42 32 43 44 42 34 42 25 24 23 21 20 22 12 14
routeLength	415	11 10 12 14 16 14 25 33 16 26 15 13 16 26 15 13 16 13 11 14 16 14 25 33 16
		13 11 10 11 10 8 10 12 24 23 21 29 30 39 38 52 66 67 69 54 66 65 75 67 87
		77 78 80 82 79 59 57 58 60 63 64 62 47 46 34 42 25 24 32 31 40 21 29 9 7
		19 29 9 6 4 3 1

filename	p0115	1 2 3 4 5 3 4 8 7 6 5 4 8 7 8 7 6 5 11 10 1 2 3 5 11 10 9 10 1
numNodes	11	
numArcs	13	
minimum	102	
initial	105	
LS	105	
Prune	103	
time	0.937	
aveCost	4.23077	
aveDiff	1.23077	
stdevCost	4.83821	
stdevDiff	1.30995	
initialRouteL	31	
routeLength	29	

Appendix C

MLPP Solutions

fileName	A3101	Plow 0: 0 1 2 9 15 16 17 16 10 9 6 3 4 5 6 5 8 7 12 7 4 3 6 3 6 9 14 13 8 7
numPlows	2	12 11 12 13 8 9 14 18 14 15 14 13 12 7 4 5 8 9 15 16 20 16 10 17 20 19 15
numNodes	116	19 18 24 30 31 32 28 26 21 19 18 14 18 24 31 30 35 36 43 42 41 42 43 47 86
numArcs	174	87 90 89 86 85 84 46 47 86 87 90 99 102 104 107 106 67 66 101 102 104 103
OBJ	24703	101 66 96 97 95 94 96 66 67 106 105 82 81 82 81 80 79 78 79 69 70 71 72 76
initial	-81	72 76 75 76 72 71 65 64 70 75 80 79 69 74 69 70 71 65 64 62 54 62 57 49 50
Plow0	12360	51 52 53 52 51 59 58 50 51 59 60 52 60 61 60 61 54 53 52 53 61 54 53 61 60
Plow1	12360	59 58 50 49 55 62 57 49 55 62 63 62 64 70 75 80 77 80 81 78 74 73 74 78 81
best	12360	78 81 82 93 83 56 65 56 55 39 29 11 2 1 0
time	1169.12	Plow 1: 0 1 2 9 10 17 20 19 15 19 21 22 21 26 33 36 37 45 38 27 23 26 28
aveCost	71.592	25 24 31 32 33 32 28 25 24 30 35 36 37 38 34 38 37 44 37 45 48 45 44 45 38
aveDiff	3.55172	27 23 26 33 36 43 48 87 91 90 89 86 85 88 95 98 97 95 98 99 98 97 96 94 92
stdevCost	56.08	84 46 47 43 48 47 48 87 91 90 99 100 99 89 88 85 84 83 92 84 83 40 39 55
stdevDiff	2.50635	56 83 92 93 68 105 106 107 108 107 108 107 109 110 109 111 115 114 113
		111 115 114 113 111 112 111 109 107 104 103 101 102 99 89 88 95 94 92 93
		68 105 82 93 83 40 35 40 41 46 41 40 39 29 35 29 11 2 1 0

fileName	A3101	Plow 0: 0 1 2 1 2 9 15 19 15 16 20 19 21 26 33 36 37 38 34 38 27 23 26 33
numPlows	3	32 28 25 24 30 31 32 33 36 43 47 48 87 90 89 86 85 84 46 47 86 87 91 90 99
numNodes	116	100 99 98 97 95 98 97 96 94 96 97 95 98 99 89 88 85 84 46 41 46 47 43 48
numArcs	174	87 90 99 102 104 107 106 67 66 67 106 105 82 81 78 74 69 70 75 76 72 76
OBJ	24741	72 71 65 56 83 40 39 55 62 54 53 52 51 52 53 52 53 61 54 62 63 62 57 49 55
initial	24657	62 57 49 50 51 59 60 61 60 52 60 61 54 53 61 60 59 58 50 51 59 58 50 49 55
Plow0	12326	39 29 35 40 41 40 35 36 37 44 37 45 38 37 45 44 45 38 27 23 26 28 25 24 31
Plow1	12369	32 28 26 21 22 21 19 18 14 13 12 11 2 1 0
Plow2	81	Plow 1: 0 1 2 9 15 19 18 14 18 24 31 30 35 36 43 42 41 42 43 48 45 48 47
best	12369	86 87 91 90 89 86 85 88 95 94 92 84 83 92 93 83 92 93 83 56 55 56 65 64 62
time	2715.59	64 70 71 65 64 70 75 80 77 80 79 69 70 71 72 76 75 80 79 69 74 73 74 78 79
aveCost	71.592	78 81 78 81 80 81 82 81 82 93 68 105 82 93 68 105 106 107 108 107 109 110
aveDiff	3.55172	109 111 112 111 115 114 113 111 115 114 113 111 109 107 108 107 104 103
stdevCost	56.08	101 102 104 103 101 66 96 66 101 102 99 89 88 95 94 92 84 83 40 39 29 11
stdevDiff	2.50635	12 7 12 13 8 7 4 3 6 3 6 5 6 9 6 3 4 5 8 7 12 7 4 5 8 9 10 17 16 17 20 16 10
		17 20 19 15 16 10 9 14 13 8 9 14 15 14 18 24 30 35 29 11 2 1 0
		Plow 2: 0 1 0

fileName	A3101	Plow 0: 0 32 28 25 24 31 32 28 25 24 31 32 33 26 28 26 21 22 21 19 15 9 10
numPlows	5	17 20 19 18 14 18 24 30 31 30 35 36 43 42 43 48 87 91 90 89 86 87 90 99
numNodes	116	100 99 102 104 103 101 102 104 107 109 110 109 111 115 114 113 111 115
numArcs	176	114 113 111 112 111 109 107 108 107 104 103 101 102 99 0
OBJ	24806	Plow 1: 0 99 98 97 95 94 96 66 101 66 67 106 105 82 93 83 56 55 62 54 53
initial	15517	61 60 61 54 62 57 49 50 51 59 58 50 51 59 60 61 54 53 61 60 52 51 52 53 52
Plow0	4944	53 52 60 59 58 50 49 55 39 29 35 40 35 36 43 47 43 48 87 91 90 89 88 95 98
Plow1	5010	97 96 97 95 98 99 0
Plow2	4876	Plow 2: 0 99 89 86 85 88 95 94 92 84 83 92 93 68 105 106 107 108 107 106
Plow3	4956	67 66 96 94 92 84 83 92 93 68 105 82 81 82 81 78 81 82 93 83 40 39 29 11 2
Plow4	5020	9 14 18 14 15 14 13 8 9 2 1 0
best	5020	Plow 3: 0 32 33 36 37 45 44 37 38 27 23 26 33 36 37 44 45 48 45 38 37 45
time	2622.86	38 34 38 27 23 26 21 19 15 16 17 20 16 10 9 6 3 4 5 6 3 6 5 8 9 14 13 8 7 4
aveCost	71.4688	5 8 7 12 13 12 7 4 3 6 9 15 16 10 17 16 20 19 18 24 30 35 29 11 12 7 12 11
aveDiff	4.21023	2 1 0
stdevCost	55.8782	Plow 4: 0 99 89 88 85 84 46 47 48 47 86 85 84 46 41 40 39 55 62 63 62 64
stdevDiff	6.94672	70 75 76 72 76 75 80 77 80 81 80 79 78 74 73 74 69 70 71 65 64 62 57 49 55
		56 65 64 70 75 80 79 69 74 78 81 78 79 69 70 71 72 76 72 71 65 56 83 40 41
		42 41 46 47 86 87 90 99 0

fileName	M3101
numPlows	2
numNodes	196
numArcs	316
OBJ	46567
initial	-18296
Plow0	23289
Plow1	23278
best	23289
time	2858.21
aveCost	74.5
aveDiff	3.55063
stdevCost	38.5549
stdevDiff	2.40266

Plow 0: 0 1 24 23 24 30 31 32 31 58 59 66 67 68 69 79 189 80 79 78 77 76 97 187 106 107 124 185 123 185 123 109 107 124 144 122 144 143 142 126 105 187 106 125 124 144 143 142 126 104 103 72 71 102 71 70 101 102 103 128 134 128 186 129 130 129 100 101 128 186 132 134 127 136 135 134 135 136 135 133 132 131 129 186 132 134 127 136 138 135 136 137 141 140 137 141 142 146 145 148 149 150 146 142 141 140 137 136 138 139 147 156 176 174 175 172 173 158 157 156 176 177 178 177 176 174 175 174 158 159 154 150 154 150 154 153 154 150 146 145 141 145 148 149 150 183 159 154 150 183 155 151 143 125 124 185 123 122 121 112 119 120 121 152 121 112 110 109 95 189 81 80 189 95 94 91 90 91 81 91 94 93 94 111 112 119 120 162 163 119 163 166 167 168 169 168 184 168 167 162 163 166 165 164 163 164 165 166 167 161 167 162 161 160 173 172 171 172 181 182 180 179 178 175 172 181 179 180 182 181 179 178 175 174 158 159 183 155 152 151 155 152 151 143 125 126 104 127 104 105 98 75 76 97 96 77 66 65 75 74 75 98 99 104 105 98 99 104 103 128 101 102 103 72 73 99 73 72 71 70 28 29 30 29 60 29 27 26 25 24 30 31 30 31 30 31 58 59 66 67 68 78 77 66 65 59 29 25 0

Plow 1: 0 1 2 3 4 5 17 34 16 17 16 192 6 192 5 17 18 191 34 191 33 191 33 35 33 57 67 68 69 56 55 80 79 78 109 110 123 185 123 122 121 120 121 120 162 161 160 155 160 169 170 169 170 169 160 173 158 157 153 149 153 157 156 147 148 147 139 140 139 138 139 138 135 133 132 131 129 100 101 70 28 29 27 28 27 26 25 24 1 2 3 19 21 22 23 2 23 32 22 23 32 22 20 18 4 3 19 20 19 21 2 21 22 20 33 20 18 17 34 16 15 190 15 14 6 7 14 6 7 14 7 193 13 38 39 12 13 38 37 49 51 50 46 47 45 195 8 9 195 9 44 43 48 47 48 47 45 195 8 194 193 13 14 7 193 12 11 10 9 8 9 10 42 41 40 11 193 12 13 14 15 16 192 5 4 18 191 33 191 33 57 56 55 54 82 81 90 92 114 113 112 113 93 92 90 89 88 87 117 116 115 114 118 115 114 113 114 113 93 92 114 113 114 118 115 116 88 87 117 116 88 89 86 84 85 84 83 50 51 54 55 52 37 36 53 56 69 79 189 81 90 89 86 84 83 82 83 50 46 86 46 47 48 49 37 38 39 40 39 12 11 40 49 48 43 45 43 41 42 44 9 8 194 11 194 193 11 10 42 44 43 41 40 49 51 52 51 54 82 81 80 55 52 53 52 37 36 53 56 57 67 68 78 109 95 94 111 112 110 111 110 123 109 107 106 125 126 105 187 97 96 108 96 77 76 75 65 64 60 61 62 63 64 60 61 28 61 62 73 74 188 63 62 73 74 188 63 64 65 59 29 25 0

fileName	M3101
numPlows	3
numNodes	196
numArcs	316
OBJ	46580
initial	35713
Plow0	15487
Plow1	15535
Plow2	15449
best	15535
time	13017.7
aveCost	74.5
aveDiff	3.55063
stdevCost	38.5549
stdevDiff	2.40266

Plow 0: 0 1 2 3 4 5 17 18 17 34 191 34 16 192 5 17 16 17 34 16 15 14 6 7 14 7 14 15 190 15 16 192 6 192 5 4 3 19 21 22 20 33 191 33 20 18 4 18 191 33 191 33 35 33 57 56 55 52 51 50 51 54 82 83 82 81 91 90 89 86 46 86 84 85 84 83 50 46 47 45 195 9 195 8 194 193 12 11 10 42 44 43 45 43 48 47 45 195 8 194 11 10 9 8 9 8 9 10 42 44 9 44 43 41 40 39 12 11 194 193 12 13 14 6 7 193 13 38 39 12 13 14 7 193 11 40 49 51 52 53 56 57 67 68 78 109 95 94 111 112 110 111 112 110 123 185 123 109 95 189 95 94 91 81 80 189 81 80 79 78 109 107 106 125 126 104 105 187 97 96 108 96 77 66 65 59 29 27 28 61 62 63 62 73 74 188 63 64 60 61 62 73 72 71 102 103 128 134 127 104 105 187 106 125 124 144 143 125 126 105 98 75 74 188 63 64 65 75 76 75 98 99 73 74 75 65 64 60 29 25 0

Plow 1: 0 25 24 1 24 23 2 23 24 30 29 60 61 28 27 26 25 24 30 31 32 22 20 19 20 18 191 33 57 67 68 69 56 69 79 189 80 55 54 55 80 79 78 77 76 97 96 77 66 65 59 66 67 68 69 79 189 81 90 91 94 93 92 90 89 86 84 83 50 46 47 48 47 48 49 48 43 41 42 41 40 11 193 13 38 39 40 49 37 36 53 52 37 36 53 56 55 52 37 38 37 49 51 54 82 81 90 92 114 113 93 94 111 110 109 110 123 122 144 143 125 124 185 123 109 107 124 185 123 185 123 122 121 112 113 114 113 93 92 114 113 114 118 115 114 118 115 116 88 89 88 87 117 116 88 87 117 116 115 114 113 112 119 163 164 163 166 165 164 165 166 167 162 163 166 167 168 184 168 169 170 169 160 173 158 157 153 154 150 183 155 152 121 112 119 120 121 120 162 161 160 155 151 143 142 126 105 98 99 104 103 72 71 70 28 29 27 26 25 0

Plow 2: 0 1 2 21 2 3 19 21 22 23 32 22 23 32 31 58 59 66 67 68 78 77 76 97 187 106 107 124 144 122 121 120 162 161 160 173 172 171 172 181 179 180 182 181 182 180 179 178 177 176 177 178 175 174 175 172 181 179 178 175 172 173 158 159 154 150 183 155 152 151 155 160 169 170 169 168 167 161 167 162 163 119 120 121 152 151 143 142 141 140 137 141 140 137 141 145 148 149 153 157 156 147 148 149 150 154 150 146 142 146 145 141 142 126 104 103 72 73 99 104 127 136 135 134 127 136 135 136 138 139 138 139 140 139 147 156 176 174 175 174 158 157 156 176 174 158 159 183 159 154 153 149 150 154 150 146 145 148 147 139 138 135 136 137 136 138 135 133 132 131 129 100 101 128 186 129 130 129 186 132 134 128 186 132 134 135 133 132 131 129 100 101 70 101 102 103 128 101 102 71 70 28 29 30 31 30 31 30 31 58 59 29 25 0

fileName	M3101	Plow 0: 0 1 24 30 31 58 59 66 67 68 78 109 110 111 112 110 109 95 94 91
numPlows	4	81 90 89 88 89 86 46 47 48 47 45 43 48 49 48 43 41 40 49 37 36 53 56 69 56
numNodes	196	55 52 37 49 51 50 46 86 84 83 50 46 47 48 47 45 195 9 44 9 8 194 193 13 38
numArcs	316	39 12 13 14 6 7 193 13 14 6 7 14 7 14 15 16 15 190 15 14 7 193 12 11 10 42
OBJ	46593	44 43 45 195 8 9 10 9 195 8 9 8 194 11 194 193 12 11 193 11 40 39 40 11 10
initial	34938	42 41 42 44 43 41 40 49 51 52 53 52 51 54 82 81 91 90 89 86 84 85 84 83 50
Plow0	11644	51 54 55 52 37 38 39 12 13 38 37 36 53 56 57 67 68 78 109 107 124 144 122
Plow1	11676	144 143 142 126 105 187 97 96 108 96 77 76 97 96 77 66 65 59 29 25 0
Plow2	11664	Plow 1: 0 25 24 1 2 21 2 3 4 18 4 5 17 34 16 17 18 191 33 191 33 57 67 68
Plow3	11609	69 79 78 77 66 67 68 69 79 189 81 90 92 114 118 115 114 113 114 113 93 94
best	11676	111 110 123 122 121 112 110 123 122 121 112 119 120 162 163 164 163 119
time	20332	120 121 120 121 120 162 161 160 173 172 173 158 159 154 153 149 150 146
aveCost	74.5	145 141 145 148 147 139 147 156 176 177 178 177 176 174 175 174 175 172
aveDiff	3.55063	181 182 181 179 178 175 172 171 172 181 179 180 182 180 179 178 175 174
stdevCost	38.5549	158 159 154 150 154 150 183 159 183 155 151 143 142 146 145 148 149 150
stdevDiff	2.40266	154 150 146 142 126 104 105 187 106 125 126 105 98 99 104 103 72 71 102
		71 70 28 29 60 29 25 0
		Plow 2: 0 1 2 3 19 21 22 23 2 23 32 22 20 18 17 34 191 33 35 33 20 18 191
		33 191 34 16 192 6 192 5 17 16 192 5 4 3 19 21 22 20 19 20 33 57 56 55 80
		55 54 82 83 82 81 80 189 81 80 79 189 95 189 80 79 78 77 76 75 98 99 104
		127 136 138 139 140 137 136 135 136 138 135 134 128 134 135 133 132 131
		129 186 132 131 129 130 129 100 101 70 101 102 103 128 186 129 100 101
		128 101 102 103 128 186 132 134 127 136 135 136 137 141 142 141 140 137
		141 140 139 138 139 138 135 133 132 134 127 104 103 72 73 74 188 63 64
		65 64 60 61 62 73 74 75 74 188 63 62 63 64 60 61 28 27 26 25 0
		Plow 3: 0 25 24 30 29 30 31 32 22 23 24 23 32 31 30 31 30 31 58 59 66 65
		75 76 97 187 106 107 106 125 124 144 143 125 124 185 123 185 123 109 107
		124 185 123 185 123 109 95 94 93 92 114 118 115 116 88 87 117 116 88 87
		117 116 115 114 113 114 113 93 92 90 91 94 111 112 113 112 119 163 166
		165 164 165 166 167 162 163 166 167 161 167 168 167 162 161 160 169 170
		169 170 169 168 184 168 169 160 155 160 173 158 157 153 157 156 176 174
		158 157 156 147 148 149 153 154 150 183 155 152 121 152 151 155 152 151
		143 125 126 104 105 98 75 65 59 29 27 28 61 62 73 99 73 72 71 70 28 29 27
		26 25 0

fileName	M3101
numPlows	5
numNodes	196
numArcs	316
OBJ	46609
initial	39154
Plow0	112
Plow1	11575
Plow2	11906
Plow3	11194
Plow4	11832
best	11906
time	18461.5
aveCost	74.5
aveDiff	3.55063
stdevCost	38.5549
stdevDiff	2.40266

Plow 0: 0 25 26 25 0

Plow 1: 0 1 2 3 19 21 22 23 2 3 4 5 17 34 16 15 190 15 14 7 193 12 11 10 42 44 43 45 195 9 10 9 195 8 9 44 9 8 9 8 194 193 13 14 7 14 6 7 14 6 192 5 4 3 19 21 2 21 22 23 32 22 20 19 20 33 191 33 191 33 20 18 17 18 4 18 191 33 57 67 68 69 56 57 56 55 52 53 52 51 52 37 38 39 12 13 14 15 16 17 16 192 5 17 34 16 192 6 7 193 12 13 38 37 36 53 56 55 80 55 52 37 49 37 36 53 56 69 79 189 81 80 189 80 79 78 109 107 124 144 143 142 126 104 103 128 101 102 103 72 71 102 103 72 71 70 101 128 186 129 186 132 131 129 130 129 100 101 70 28 29 25 0

Plow 2: 0 25 24 30 31 30 31 30 31 58 59 29 30 31 32 22 20 18 191 34 191 33 35 33 57 67 68 69 79 189 95 189 81 90 91 90 89 88 87 117 116 88 87 117 116 115 116 88 89 86 84 83 50 51 54 55 54 82 81 80 79 78 77 66 67 68 78 109 95 94 111 112 110 111 112 119 163 166 167 161 167 168 167 162 163 164 165 164 163 119 120 121 112 119 120 121 120 162 161 160 155 160 169 170 169 168 184 168 169 170 169 160 173 172 173 158 157 156 147 148 149 150 154 150 183 155 152 151 143 125 126 104 105 187 106 107 106 125 124 185 123 185 123 109 107 124 144 143 125 126 105 98 99 73 72 73 74 188 63 62 63 64 65 59 66 67 68 78 77 66 65 59 29 25 0

Plow 3: 0 25 24 23 24 1 2 23 32 31 58 59 66 65 75 76 97 96 77 76 97 187 97 96 108 96 77 76 75 98 99 104 127 104 105 187 106 125 124 185 123 109 110 109 95 94 91 81 90 92 114 118 115 114 113 93 92 114 118 115 114 113 114 113 112 113 114 113 93 94 93 92 90 89 86 46 47 45 43 41 40 11 193 13 38 39 12 11 194 193 11 10 42 44 43 48 47 45 195 8 194 11 40 39 40 49 51 54 82 83 50 46 47 48 47 48 49 48 43 41 42 41 40 49 51 50 46 86 84 85 84 83 82 81 91 94 111 110 123 185 123 122 144 122 121 112 110 123 122 121 152 151 155 151 143 142 126 105 98 75 74 188 63 64 60 29 27 26 25 0

Plow 4: 0 1 24 30 29 60 61 62 73 74 75 65 64 60 61 62 73 99 104 103 128 186 132 134 135 133 132 134 127 136 137 141 140 137 141 142 141 145 148 147 156 176 177 178 175 172 171 172 181 179 180 182 181 179 178 175 174 158 157 156 176 174 175 174 175 172 181 182 180 179 178 177 176 174 158 159 183 159 154 153 157 153 154 150 183 155 152 121 120 162 163 166 165 166 167 162 161 160 173 158 159 154 150 146 142 146 145 148 149 153 149 150 154 150 146 145 141 140 139 140 137 136 138 135 136 135 136 135 134 128 134 127 136 138 139 147 139 138 139 138 135 133 132 131 129 100 101 102 71 70 28 61 28 27 28 29 27 26 25 0

fileName	hd115	Plow 0: 0 11 12 34 29 2 8 17 8 29 56 15 17 16 0 11 1 29 6 7 6 17 16 15 7 6
numPlows	2	7 6 17 7 6 29 56 53 56 41 34 40 41 13 14 13 41 19 27 19 34 41 19 18 28 29
numNodes	70	46 47 40 41 46 56 57 36 35 37 20 37 20 21 30 22 28 18 33 32 37 20 37 20 21
numArcs	141	30 22 23 30 22 23 27 23 30 22 31 30 22 31 21 31 30 22 28 27 28 29 52 62 61
OBJ	11453	60 67 63 55 53 64 55 53 64 55 63 67 58 54 53 54 60 52 62 69 61 60 66 60 52
initial	-2109	15 7 6 8 0 1 29 2 8 29 52 62 69 68 69 68 49 51 68 49 50 65 50 49 50 51 52
Plow0	5725	62 68 69 61 50 49 51 68 62 61 50 51 52 15 17 7 6 8 0
Plow1	5728	Plow 1: 0 1 2 13 3 5 4 3 5 9 10 9 10 5 4 10 5 9 14 12 11 1 48 44 45 1 45 39
best	5728	38 43 39 38 42 38 43 42 59 58 66 59 58 54 60 67 58 66 54 67 63 64 63 67 66
time	235.871	54 67 66 59 42 38 39 38 39 38 42 43 39 44 43 45 39 44 43 45 48 44 45 48 16
aveCost	41.5957	15 46 60 46 56 15 46 47 12 47 40 19 34 29 46 41 15 41 47 41 2 3 2 13 3 4 10
aveDiff	3.21986	9 14 12 34 40 19 18 33 32 37 35 36 35 36 65 57 56 57 36 24 26 24 25 26 33
stdevCost	50.0301	32 33 32 33 32 25 26 33 32 25 24 36 24 36 65 57 56 41 2 1 48 16 0
stdevDiff	2.56571	

fileName	hd115	Plow 0: 0 11 12 34 29 46 56 53 54 53 56 57 36 24 36 65 50 49 50 49 50 51
numPlows	3	52 62 61 60 67 66 60 46 47 12 47 40 19 18 28 18 33 32 25 24 36 35 37 35 36
numNodes	70	35 36 24 25 26 33 32 25 26 24 26 33 32 33 32 33 32 37 20 21 30 22 23 30 22
numArcs	141	23 27 19 18 33 32 37 20 37 20 37 20 21 30 22 31 30 22 31 21 31 30 22 28 27
OBJ	11453	23 30 22 28 29 56 57 36 65 57 56 46 60 66 59 42 38 42 43 45 39 38 43 39 38
initial	10787	39 38 43 45 39 44 43 39 38 42 38 39 44 45 48 16 0
Plow0	3818	Plow 1: 0 1 29 2 1 29 2 13 3 4 3 5 4 10 5 9 10 5 4 10 9 10 9 14 12 11 1 48
Plow1	3817	44 45 48 16 15 7 6 7 6 17 8 29 46 41 19 34 40 41 34 41 13 14 13 41 2 13 3 2
Plow2	3818	8 29 6 29 52 15 17 16 15 46 41 2 3 5 9 14 12 34 29 52 15 7 6 8 0 1 2 8 17 7
best	3818	6 7 6 17 7 6 8 0
time	779.948	Plow 2: 0 11 1 45 1 48 44 43 42 59 58 66 54 60 52 62 68 49 51 68 49 51 68
aveCost	41.5957	62 61 50 51 52 62 69 68 69 68 69 61 60 67 63 67 66 54 67 58 54 67 63 55 53
aveDiff	3.21986	64 55 53 64 55 63 64 63 67 58 66 59 58 54 60 52 62 69 61 50 65 57 56 15 46
stdevCost	50.0301	47 40 41 47 41 15 56 41 19 34 40 19 27 28 29 56 41 15 17 16 0
stdevDiff	2.56571	

fileName	hd115	Plow 0: 0 11 12 47 40 41 47 40 41 46 47 41 34 29 52 62 69 68 62 68 49 51
numPlows	4	52 62 69 68 69 61 50 65 57 36 35 37 20 21 31 30 22 31 30 22 31 21 30 22 23
numNodes	70	30 22 28 29 6 8 29 2 3 2 13 41 13 3 5 4 10 9 14 13 14 12 34 41 15 46 47 12
numArcs	141	11 1 2 8 0
OBJ	11453	Plow 1: 0 1 29 52 62 61 60 67 58 54 67 63 67 66 59 58 66 54 60 52 62 61 50
initial	5778	51 52 15 7 6 7 6 8 17 7 6 17 8 29 46 41 19 18 33 32 37 20 21 30 22 23 27 23
Plow0	3000	30 22 28 27 28 18 28 29 56 41 2 1 48 44 43 45 39 38 42 38 43 39 38 39 38
Plow1	2803	39 44 45 48 16 15 7 6 7 6 17 16 0
Plow2	2783	Plow 2: 0 11 1 48 16 15 17 7 6 29 46 60 46 56 53 64 55 53 64 55 63 67 58
Plow3	2867	66 59 42 38 43 39 44 43 42 43 45 48 44 45 1 45 39 38 42 59 58 54 60 67 66
best	3000	60 52 15 46 56 57 36 24 26 33 32 25 24 36 65 50 49 51 68 49 50 49 50 51 68
time	5383.8	69 61 60 66 54 53 54 67 63 64 63 55 53 56 15 17 16 0
aveCost	41.5957	Plow 3: 0 1 29 56 57 56 15 41 19 27 19 34 29 2 13 3 4 3 5 9 10 5 4 10 5 9
aveDiff	3.21986	10 9 14 12 34 40 19 34 40 19 18 33 32 33 32 33 32 37 20 37 20 37 35 36 35
stdevCost	50.0301	36 24 25 26 33 32 25 26 24 36 65 57 56 41 2 8 0
stdevDiff	2.56571	

fileName	hd115	Plow 0: 0 1 29 56 53 64 55 53 56 41 19 18 28 29 52 62 68 49 51 52 62 61 60
numPlows	5	52 15 17 7 6 29 46 41 2 13 3 4 10 9 10 5 9 10 5 4 10 9 14 12 34 29 1 48 44
numNodes	70	43 45 39 38 43 42 43 45 48 16 0
numArcs	141	Plow 1: 0 1 29 52 62 61 60 66 54 60 67 63 64 63 67 63 55 53 54 53 64 55 63
OBJ	11455	67 66 60 46 56 57 36 24 36 65 57 36 35 36 65 50 51 68 49 50 49 51 68 62 69
initial	4872	68 69 68 69 61 50 65 57 56 41 2 13 14 13 41 15 7 6 17 7 6 7 6 7 6 8 0
Plow0	2269	Plow 2: 0 1 2 8 29 46 47 41 15 17 16 15 46 47 41 34 40 41 46 60 67 58 66
Plow1	2267	59 58 54 67 58 66 59 42 59 58 54 67 66 54 60 52 62 69 61 50 51 52 15 7 6 8
Plow2	2291	29 1 0
Plow3	2325	Plow 3: 0 11 12 47 40 41 19 34 41 13 3 5 4 3 5 9 14 12 47 40 19 34 29 6 17
Plow4	2303	8 17 16 15 46 56 57 56 15 56 29 2 3 2 8 0
best	2325	Plow 4: 0 11 1 45 48 44 45 39 38 42 38 43 39 38 42 38 39 38 39 44 43 39 44
time	8641.83	45 1 11 12 34 40 19 18 33 32 33 32 25 26 33 32 33 32 37 20 21 31 30 22 23
aveCost	41.5957	30 22 31 30 22 31 21 30 22 28 18 33 32 37 20 37 20 37 35 36 24 25 24 26 33
aveDiff	3.21986	32 25 26 24 36 35 37 20 21 30 22 28 27 19 27 23 30 22 23 27 28 29 2 1 48
stdevCost	50.0301	16 0
stdevDiff	2.56571	

fileName	hd215	Plow 0: 0 1 3 1 40 64 3 0 2 11 12 30 4 21 16 19 18 28 29 24 27 29 24 27 28
numPlows	2	29 26 24 25 30 26 24 25 4 31 68 49 6 34 47 48 68 49 51 66 50 51 50 51 50
numNodes	70	67 53 61 57 55 54 56 59 58 59 62 58 56 59 58 56 54 60 61 57 53 52 25 52 30
numArcs	162	4 6 48 49 4 52 25 52 30 26 27 29 26 27 28 19 18 20 17 15 22 17 15 22 23 14
OBJ	13530	22 17 15 17 15 17 15 14 22 21 10 23 22 21 10 8 9 13 12 25 49 25 68 41 42
initial	-980	35 42 39 44 39 43 38 37 33 48 47 48 33 48 49 51 50 49 4 21 16 19 28 18 20
Plow0	6767	17 15 14 16 23 14 16 23 10 8 7 46 32 45 40 31 32 45 40 31 34 33 48 4 68 6
Plow1	6763	34 33 48 4 31 68 67 53 52 12 2 11 12 13 10 5 4 5 64 40 3 0
best	6767	Plow 1: 0 1 40 45 40 45 46 45 40 3 0 2 1 64 65 61 65 69 63 62 58 59 62 37
time	338.835	66 69 38 37 33 48 68 41 34 4 6 46 6 48 33 47 35 34 35 33 47 37 66 50 67 25
aveCost	42.9877	52 57 53 61 51 66 69 63 59 63 62 37 35 33 48 47 35 42 35 42 31 32 45 40 41
aveDiff	3.61728	34 4 12 25 68 6 7 8 9 13 10 12 9 10 12 21 25 52 21 12 52 21 25 4 52 57 65
stdevCost	44.0862	57 55 54 60 55 60 55 60 61 51 50 49 6 7 46 32 45 40 41 4 12 9 10 5 64 65 69
stdevDiff	2.37823	38 43 44 41 31 42 41 31 34 47 37 35 42 39 36 35 42 36 35 42 36 43 39 36 43
		44 41 4 68 67 25 30 12 2 1 64 3 11 3 0 3 0 3 0

fileName	hd215	Plow 0: 0 1 64 3 11 12 25 4 52 30 12 25 4 12 52 21 12 30 4 31 42 35 42 36
numPlows	3	43 44 39 36 35 42 35 42 41 31 68 67 25 52 30 26 27 29 26 24 27 29 24 25 52
numNodes	70	25 21 16 14 15 17 15 14 16 19 16 23 22 21 16 23 14 22 17 15 22 23 10 8 7
numArcs	162	46 32 45 40 41 4 52 57 53 61 57 55 54 56 54 60 55 60 61 51 66 50 51 50 67
OBJ	13530	53 52 21 10 5 64 3 0
initial	12771	Plow 1: 0 3 0 2 11 12 52 25 30 26 27 28 19 18 28 19 18 20 17 15 17 20 18
Plow0	4507	28 29 24 27 28 29 26 24 25 49 51 50 67 25 68 49 51 50 51 50 49 4 68 6 48
Plow1	4511	33 48 4 21 10 12 2 1 40 45 40 45 46 32 45 40 3 1 3 0 2 1 64 65 57 53 61 65
Plow2	4512	69 63 62 37 66 69 38 37 35 34 47 37 33 48 47 48 68 49 6 48 49 4 6 34 4 12
best	4512	9 13 12 9 13 10 12 13 10 8 9 10 23 14 22 17 15 22 21 25 30 4 31 34 4 21 12
time	1731.99	2 11 3 0
aveCost	42.9877	Plow 2: 0 3 0 1 40 31 34 47 35 33 48 33 47 37 33 48 47 35 33 48 49 25 68 6
aveDiff	3.61728	34 35 42 31 32 45 40 64 65 61 51 66 50 49 6 46 45 40 41 34 33 48 4 68 41 4
stdevCost	44.0862	6 7 46 6 7 8 9 10 5 4 5 64 40 31 68 67 53 52 57 55 60 55 54 60 61 57 65 69
stdevDiff	2.37823	63 59 58 56 59 62 58 59 58 56 59 62 58 59 63 62 37 66 69 38 37 35 42 36 35
		42 39 44 41 42 39 36 43 39 43 38 43 44 41 34 33 47 48 68 41 31 32 45 40 3 0

fileName	hd215	Plow 0: 0 3 1 64 65 61 57 65 69 63 62 58 56 54 60 55 54 56 59 58 59 62 37
numPlows	4	33 47 35 42 31 68 49 51 50 67 53 61 65 69 38 43 38 37 35 42 35 42 41 4 68
numNodes	70	67 25 68 67 25 52 25 49 25 30 26 24 27 28 18 20 17 15 17 15 22 17 15 17 15
numArcs	162	14 16 19 28 29 26 27 28 19 18 28 29 26 24 27 29 24 25 4 5 64 40 3 11 3 0 3 0
OBJ	13530	Plow 1: 0 2 11 12 25 30 12 52 21 10 8 9 13 12 25 4 6 34 35 42 39 43 39 36
initial	8800	35 33 48 49 51 50 67 53 52 30 4 31 34 47 37 66 69 38 37 66 69 63 59 58 59
Plow0	3343	62 58 56 59 63 62 37 33 48 33 48 33 48 49 6 7 46 45 40 45 46 32 45 40 45
Plow1	3396	40 41 34 33 48 47 35 42 36 43 44 39 36 43 44 41 31 68 6 7 8 9 10 5 64 3 0
Plow2	3412	Plow 2: 0 1 64 65 57 55 60 61 51 66 50 51 50 49 6 34 4 68 41 4 21 12 13 10
Plow3	3379	12 21 25 68 41 42 36 35 33 48 4 6 48 47 48 68 49 4 12 30 26 27 29 24 25 52
best	3412	25 52 57 53 52 12 2 11 12 9 13 10 23 14 16 23 22 21 10 12 9 10 8 7 46 6 46
time	4158.33	32 45 40 3 0
aveCost	42.9877	Plow 3: 0 2 1 40 31 42 35 34 47 37 35 42 39 44 41 31 32 45 40 41 34 33 47
aveDiff	3.61728	48 68 6 48 4 52 30 4 52 21 25 52 57 55 60 55 54 60 61 57 53 61 51 50 51 66
stdevCost	44.0862	50 49 4 21 16 23 14 22 17 15 22 21 16 19 18 20 17 15 14 22 23 10 5 4 31 34
stdevDiff	2.37823	4 12 2 1 3 0 1 40 31 32 45 40 64 3 0

fileName	hd215	Plow 0: 0 3 0 1 64 65 69 63 62 58 59 58 56 59 62 58 56 59 58 59 63 59 62
numPlows	5	37 66 69 38 43 39 44 39 36 43 44 41 4 6 34 47 37 66 50 67 25 4 21 10 23 14
numNodes	70	16 23 22 17 15 14 16 23 14 22 23 10 5 4 12 9 13 10 8 9 10 8 7 46 6 48 4 52
numArcs	162	21 25 52 12 2 1 3 0
OBJ	13530	Plow 1: 0 2 11 12 52 25 68 67 53 61 57 65 69 63 62 37 33 47 48 47 35 42 39
initial	10992	43 44 41 4 5 64 65 57 55 54 60 55 60 61 65 61 57 53 61 51 50 49 6 34 33 48
Plow0	2735	49 51 50 49 4 68 41 31 32 45 40 45 40 64 3 0
Plow1	2713	Plow 2: 0 3 11 12 25 68 6 48 68 67 53 52 57 53 52 21 16 19 18 28 19 28 29
Plow2	2668	24 25 30 26 24 27 28 29 24 25 52 30 4 21 16 19 18 20 17 15 22 17 15 22 21
Plow3	2713	10 12 2 11 3 1 40 41 42 36 35 33 48 4 31 34 4 68 6 46 32 45 40 31 32 45 40
Plow4	2701	3 0
best	2735	Plow 3: 0 1 64 40 41 34 35 42 35 42 41 31 42 35 42 31 34 47 35 42 39 36 35
time	6632.43	33 48 33 48 49 4 52 57 55 54 56 54 60 55 60 61 51 50 51 50 67 25 49 51 66
aveCost	42.9877	50 51 66 69 38 37 33 48 33 47 48 68 41 34 33 48 47 37 35 42 36 43 38 37 35
aveDiff	3.61728	34 4 6 7 46 45 40 45 46 32 45 40 3 0
stdevCost	44.0862	Plow 4: 0 2 1 40 31 68 49 25 30 12 25 52 25 52 30 26 27 29 26 27 29 26 24
stdevDiff	2.37823	27 28 18 20 17 15 17 15 17 15 14 22 21 12 21 25 4 12 30 4 31 68 49 6 7 8 9
		10 12 13 12 9 13 10 5 64 3 0

fileName	hd315	Plow 0: 0 8 0 15 52 24 22 25 12 4 5 30 29 26 31 27 12 23 30 47 43 44 47 44
numPlows	2	47 44 46 39 46 67 47 44 43 40 42 45 42 40 41 43 44 43 40 41 42 41 42 41 43
numNodes	68	44 35 13 3 2 14 3 2 34 33 59 60 58 51 50 58 60 50 52 24 25 12 1 31 12 4 6 5
numArcs	153	30 29 30 29 23 19 64 63 56 64 18 19 28 31 27 1 15 52 51 58 59 64 18 20 17
OBJ	12449	16 22 16 18 20 17 16 21 18 16 21 17 21 17 20 17 20 17 21 18 19 64 62 66 65
initial	-4135	61 62 59 60 50 52 51 50 58 59 54 53 48 49 48 5 32 33 59 63 59 54 36 59 64
Plow0	6230	62 66 65 61 62 63 56 57 49 54 53 48 23 22 25 15 12 1 26 28 31 29 5 6 7 32
Plow1	6219	5 23 15 1 0
best	6230	Plow 1: 0 15 4 5 23 54 49 55 56 57 49 55 57 48 55 57 48 30 53 23 54 36 37
time	99.718	38 36 59 37 39 36 37 39 45 35 34 14 35 45 39 36 37 38 33 54 30 47 43 44 46
aveCost	41.8856	67 66 62 59 37 54 30 53 29 26 28 27 12 11 10 3 2 34 13 14 34 13 14 35 45
aveDiff	3.44444	35 34 33 54 29 48 30 67 66 62 63 24 25 23 22 25 23 30 29 30 67 47 44 35 13
stdevCost	36.9781	11 10 3 2 14 3 2 8 9 10 3 2 0 1 31 29 5 48 23 15 12 23 53 29 48 55 56 64 63
stdevDiff	2.69807	24 22 25 15 4 6 7 32 33 38 36 37 54 29 23 19 26 19 28 27 1 26 31 12 11 13
		35 13 35 13 3 2 8 9 10 3 2 3 2 3 2 0

fileName	hd315	Plow 0: 0 1 15 12 23 15 52 51 58 60 50 58 59 54 49 55 57 49 48 55 56 64 63
numPlows	3	24 25 15 12 23 30 47 43 40 41 43 40 42 40 41 43 44 43 44 47 44 47 44 46 67
numNodes	68	66 62 66 62 59 64 62 66 65 61 62 63 24 25 15 52 24 22 16 21 17 16 18 19 64
numArcs	153	18 20 17 20 17 20 17 21 18 19 64 62 59 64 18 20 17 21 18 16 21 17 16 22 25
OBJ	12449	12 1 31 29 26 19 28 27 1 0
initial	9722	Plow 1: 0 15 1 26 31 12 1 31 27 1 26 28 31 29 30 53 48 55 56 57 48 30 29
Plow0	4158	26 31 12 4 5 30 53 23 22 25 23 22 25 23 53 29 30 29 48 30 47 43 44 35 34
Plow1	4141	13 35 13 11 10 3 2 34 33 59 54 30 67 47 44 46 67 47 44 43 44 35 45 35 13
Plow2	4150	35 34 33 54 29 23 19 28 27 12 4 6 7 32 5 30 29 23 19 26 28 31 27 12 11 13
best	4158	3 2 3 2 34 14 34 13 3 2 8 9 10 3 2 3 2 14 3 2 0
time	1886.34	Plow 2: 0 8 0 15 4 5 32 33 38 33 54 29 5 23 54 53 48 49 54 36 37 39 45 42
aveCost	41.8856	41 42 41 42 45 35 13 14 35 13 14 35 45 39 36 37 54 30 29 5 23 15 4 6 7 32
aveDiff	3.44444	33 59 37 54 36 59 37 38 36 37 38 36 37 39 46 39 36 59 63 56 57 49 55 57 48
stdevCost	36.9781	5 6 5 48 23 54 53 29 48 23 30 67 66 65 61 62 63 56 64 63 59 60 58 59 60 50
stdevDiff	2.69807	52 51 50 58 51 50 52 24 22 25 12 11 10 3 2 14 3 2 8 9 10 3 2 0

fileName	hd315	Plow 0: 0 15 52 24 25 23 53 29 48 30 29 23 19 64 62 63 24 25 12 23 15 1 31
numPlows	4	27 12 1 31 29 5 32 5 30 47 43 40 41 43 44 43 40 42 45 35 34 33 59 37 54 30
numNodes	68	29 30 67 47 44 47 44 35 34 14 35 13 3 2 3 2 0
numArcs	153	Plow 1: 0 15 4 6 5 23 30 47 43 44 47 44 35 13 11 10 3 2 14 3 2 8 9 10 3 2
OBJ	12449	14 3 2 34 13 14 34 33 59 64 62 63 56 64 18 20 17 20 17 21 18 20 17 21 18
initial	10380	19 26 31 29 48 49 55 56 64 63 56 57 48 5 48 23 54 53 48 23 22 25 15 4 5 23
Plow0	3098	54 53 29 26 28 27 1 26 31 27 1 0
Plow1	3114	Plow 2: 0 1 15 52 24 22 25 23 22 16 21 17 20 17 16 21 17 16 18 16 22 25 12
Plow2	3123	4 6 7 32 33 54 30 29 30 67 47 44 46 67 66 65 61 62 66 62 59 64 18 19 64 63
Plow3	3114	59 37 54 36 37 38 33 54 49 55 57 49 54 29 26 19 28 27 12 23 15 12 1 26 28
best	3123	31 12 11 10 3 2 8 0
time	2865.18	Plow 3: 0 8 9 10 3 2 3 2 34 13 35 45 42 41 42 40 41 42 41 43 44 43 44 46
aveCost	41.8856	39 36 37 39 36 59 54 29 5 30 53 48 55 56 57 49 48 55 57 48 30 29 23 30 53
aveDiff	3.44444	23 19 28 31 12 4 5 6 7 32 33 38 36 37 38 36 59 60 58 51 50 52 51 58 59 60
stdevCost	36.9781	50 58 60 50 52 51 50 58 59 54 36 37 39 45 35 13 35 13 14 35 45 39 46 67 66
stdevDiff	2.69807	62 66 65 61 62 59 63 24 22 25 15 12 11 13 3 2 0

fileName	hd315	Plow 0: 0 8 9 10 3 2 14 34 33 38 36 37 39 46 39 45 39 36 59 60 58 59 64 63
numPlows	5	24 22 16 21 17 21 17 20 17 21 18 20 17 20 17 16 21 18 16 18 19 64 62 66 62
numNodes	68	59 37 38 33 59 60 50 58 60 50 52 24 25 15 4 5 30 47 44 35 13 35 13 35 45
numArcs	153	35 45 35 34 13 14 35 13 3 2 3 2 8 0
OBJ	12450	Plow 1: 0 1 26 19 28 27 12 4 6 5 48 23 30 53 29 23 54 53 48 30 29 5 23 15
initial	6884	1 31 12 23 19 64 18 20 17 16 22 25 23 53 23 19 28 31 27 1 15 12 23 22 25
Plow0	2481	12 11 10 3 2 0
Plow1	2486	Plow 2: 0 15 12 1 31 29 23 15 52 51 50 52 51 50 58 51 58 59 63 59 54 30 53
Plow2	2511	48 5 30 29 30 29 48 55 57 49 48 23 54 53 29 48 30 67 47 44 47 43 44 43 40
Plow3	2472	42 41 42 40 41 42 45 42 41 43 44 43 40 41 43 44 35 34 13 14 3 2 0
Plow4	2500	Plow 3: 0 15 52 24 25 23 30 67 66 65 61 62 66 65 61 62 63 56 57 48 55 56
best	2511	64 63 24 22 25 15 4 5 32 33 59 64 62 63 56 57 49 55 57 48 49 54 49 55 56
time	6504.88	64 18 19 26 31 29 30 29 26 31 27 1 26 28 31 12 11 13 11 10 3 2 3 2 8 0
aveCost	41.8856	Plow 4: 0 8 9 10 3 2 14 3 2 34 14 35 13 3 2 34 33 54 30 47 43 44 46 67 47
aveDiff	3.44444	44 47 44 46 67 66 62 59 37 54 29 26 28 27 12 4 6 7 32 5 6 7 32 33 54 36 59
stdevCost	36.9781	54 36 37 38 36 37 39 36 37 54 29 5 23 22 25 12 1 0
stdevDiff	2.69807	

fileName	hd415	Plow 0: 0 3 4 16 62 51 52 7 6 13 22 12 20 61 69 39 60 62 59 69 60 52 24 51
numPlows	2	42 39 11 13 2 5 21 20 5 21 12 11 13 22 21 12 21 12 21 12 11 22 21 20 67 66
numNodes	88	65 84 37 75 84 65 80 87 80 81 87 86 81 67 79 80 65 84 37 75 84 65 66 67 79
numArcs	171	67 79 74 73 76 78 73 76 77 74 73 72 74 79 67 39 61 59 20 60 59 39 11 22 12
OBJ	12111	20 61 59 69 75 69 60 52 7 8 18 52 42 39 60 59 39 6 3 2 4 17 5 1 63 56 58 53
initial	-1470	47 46 55 46 55 49 50 53 58 56 57 0 17 1 0
Plow0	6052	Plow 1: 0 17 5 63 57 58 53 47 46 48 55 54 78 76 77 85 77 70 68 83 82 71 83
Plow1	6059	71 83 71 68 69 20 60 62 51 52 24 23 24 42 24 42 24 42 16 19 18 7 8 9 10 15
best	6059	10 9 15 10 9 10 15 14 19 14 19 18 19 14 10 8 9 15 14 10 8 18 52 42 39 61 79
time	403.399	80 81 87 86 66 64 65 66 64 86 66 65 64 86 81 67 79 61 69 39 38 30 33 40 29
aveCost	36.5585	34 29 28 45 44 41 26 25 27 35 36 25 27 26 23 38 30 31 24 51 42 39 6 3 2 4
aveDiff	3.63158	17 1 63 56 57 58 53 58 53 50 48 55 49 50 47 46 55 54 49 48 50 47 46 55 46
stdevCost	49.1893	48 49 48 49 54 78 73 72 74 77 70 68 83 82 85 70 85 82 85 82 71 68 69 20 67
stdevDiff	2.61551	39 38 31 32 40 33 32 30 33 40 34 33 40 34 33 40 33 32 40 29 28 45 44 41 36
		27 35 37 43 44 41 36 25 35 36 27 26 25 35 37 43 44 41 26 23 38 31 32 30 31
		24 42 16 19 18 19 18 7 6 13 2 5 63 57 0 3 4 16 62 59 20 5 1 0

fileName	hd415	Plow 0: 0 3 4 17 1 0 17 5 63 56 57 58 53 47 46 55 49 50 48 55 54 49 54 78
numPlows	3	73 76 78 73 72 74 73 72 74 77 85 82 85 70 68 69 60 62 51 52 42 39 60 59 69
numNodes	88	20 60 52 42 16 19 18 7 8 18 19 18 7 6 3 2 4 16 19 14 10 8 9 10 9 15 10 9 15
numArcs	171	10 15 14 19 14 10 15 14 19 18 52 24 23 38 30 33 32 40 29 34 33 40 33 40 34
OBJ	12111	33 40 34 29 28 45 44 41 26 23 24 42 24 42 39 38 30 31 24 51 42 39 6 13 22
initial	11285	12 21 12 21 20 5 63 57 0
Plow0	4037	Plow 1: 0 3 2 4 16 62 59 20 61 59 39 60 59 69 75 84 37 43 44 41 36 25 27
Plow1	4032	26 25 27 26 25 35 36 27 35 37 43 44 41 26 23 38 31 32 40 33 40 29 28 45 44
Plow2	4042	41 36 25 35 36 27 35 37 75 84 65 64 65 66 64 86 66 64 86 66 65 66 65 80 81
best	4042	67 79 80 87 86 81 87 86 81 87 80 81 67 66 67 79 61 79 67 79 67 79 74 79 80
time	1475.62	65 84 65 84 37 75 69 20 60 52 7 8 9 10 8 18 19 18 52 24 42 24 42 16 62 51
aveCost	36.5585	42 39 6 13 2 5 21 20 5 1 0
aveDiff	3.63158	Plow 2: 0 17 5 21 12 20 61 59 20 67 39 11 22 12 11 13 2 5 1 63 57 58 53 58
stdevCost	49.1893	56 58 53 58 53 50 53 47 46 48 49 48 49 48 50 47 46 55 49 50 47 46 48 55 46
stdevDiff	2.61551	55 46 55 54 78 76 77 70 85 82 71 68 83 71 83 82 71 68 83 71 83 82 85 77 74
		73 76 77 70 68 69 39 61 69 60 62 59 39 61 69 39 11 22 21 12 11 13 22 21 12
		20 67 39 38 31 32 30 33 32 30 31 24 51 52 7 6 3 4 17 1 63 56 57 0

fileName	hd415	Plow 0: 0 17 1 63 56 58 53 47 46 55 49 54 78 76 77 70 68 69 60 52 42 16 62
numPlows	4	51 42 39 38 31 24 42 24 51 42 16 19 18 7 6 3 2 4 16 19 14 10 15 14 19 18 52
numNodes	88	7 8 9 15 14 19 18 19 18 52 42 24 42 39 11 22 12 11 13 2 5 1 63 56 57 0
numArcs	171	Plow 1: 0 3 4 17 5 21 20 5 21 12 11 22 21 12 21 12 21 20 60 59 39 60 59 20
OBJ	12111	60 52 24 42 39 11 13 22 21 12 20 61 69 20 67 79 67 39 61 79 80 81 87 80 81
initial	6033	67 39 61 59 39 6 3 4 17 1 0
Plow0	3029	Plow 2: 0 17 5 63 57 58 53 58 56 57 58 53 50 48 55 46 48 49 48 50 47 46 55
Plow1	3011	54 49 48 55 49 50 47 46 48 49 50 53 58 53 47 46 55 46 55 54 78 73 72 74 73
Plow2	3026	76 78 73 72 74 73 76 77 74 79 74 77 85 70 85 82 85 82 85 77 70 68 83 82 71
Plow3	3045	68 83 71 83 82 71 83 71 68 69 75 84 37 75 69 39 38 31 32 30 33 40 34 29 34
best	3045	33 40 33 32 30 31 32 40 33 40 34 33 40 29 28 45 44 41 36 27 26 23 38 30 31
time	5008.98	24 42 39 6 13 22 12 20 67 79 80 65 66 64 86 81 67 66 65 64 65 66 64 86 66
aveCost	36.5585	65 80 87 86 81 87 86 66 67 79 67 79 61 59 20 61 69 20 5 1 0
aveDiff	3.63158	Plow 3: 0 3 2 4 16 62 59 69 39 60 62 59 69 60 62 51 52 24 23 38 30 33 32
stdevCost	49.1893	40 29 28 45 44 41 36 25 27 35 36 25 35 36 27 26 25 27 35 37 75 84 65 84 65
stdevDiff	2.61551	84 37 43 44 41 26 25 35 37 43 44 41 26 23 24 51 52 7 8 9 10 9 10 8 18 19 14
		10 15 10 9 15 10 8 18 7 6 13 2 5 63 57 0

fileName	hd415	Plow 0: 0 3 2 4 16 19 18 52 24 42 16 19 14 10 15 10 9 10 8 18 7 8 9 15 10 9
numPlows	5	15 14 19 18 19 18 52 24 42 16 62 51 42 39 38 30 31 24 51 52 42 24 42 24 42
numNodes	88	39 11 13 22 21 12 11 13 2 5 63 57 0
numArcs	171	Plow 1: 0 17 5 21 12 21 12 11 22 21 20 67 79 74 79 80 81 67 79 67 39 38 31
OBJ	12115	32 30 33 40 33 40 33 32 40 29 28 45 44 41 36 27 26 25 27 35 37 75 84 65 66
initial	5978	64 86 81 87 80 65 84 65 84 37 75 84 37 43 44 41 26 25 27 26 23 24 23 38 30
Plow0	2447	33 32 40 29 34 33 40 34 33 40 34 29 28 45 44 41 36 27 35 36 25 35 36 25 35
Plow1	2398	37 43 44 41 26 23 38 31 32 30 31 24 51 52 7 6 13 2 5 1 0
Plow2	2413	Plow 2: 0 17 1 63 56 57 58 53 50 47 46 48 49 48 49 50 47 46 55 54 78 76 77
Plow3	2429	74 77 70 68 83 82 71 83 71 68 69 20 61 79 80 87 86 66 67 66 65 80 81 87 86
Plow4	2428	66 65 66 64 65 64 86 81 67 79 61 69 20 5 21 12 20 61 59 69 39 60 62 59 20
best	2447	5 63 57 58 53 58 53 58 56 57 0
time	8798.83	Plow 3: 0 17 1 63 56 58 53 47 46 55 46 48 55 46 55 49 50 48 50 53 47 46 55
aveCost	36.5585	49 48 55 54 49 54 78 73 72 74 73 76 78 73 72 74 73 76 77 70 85 82 85 77 85
aveDiff	3.63158	70 68 83 71 83 82 85 82 71 68 69 60 59 39 11 22 12 21 20 60 52 42 39 61 59
stdevCost	49.1893	39 6 13 22 12 20 67 79 67 39 6 3 4 17 5 1 0
stdevDiff	2.61551	Plow 4: 0 3 4 16 62 59 69 75 69 39 60 62 51 42 39 61 69 60 59 20 60 52 7 8
		9 10 15 14 19 14 10 8 18 19 18 7 6 3 2 4 17 0

fileName	hd515	Plow 0: 0 13 0 13 1 16 15 16 5 12 80 12 11 17 9 8 17 14 15 4 18 14 4 18 14
numPlows	2	4 5 29 17 14 15 18 15 18 15 4 5 29 17 27 29 21 31 30 19 30 19 20 23 38 22
numNodes	86	20 23 22 38 22 20 19 21 19 20 19 30 29 21 41 25 40 26 40 25 40 24 25 40 24
numArcs	178	25 36 37 34 36 24 26 23 38 26 23 22 30 19 31 30 19 31 5 12 11 41 11 27 28
OBJ	10808	39 35 37 34 36 34 32 33 64 48 63 52 53 60 43 7 9 8 10 6 8 10 43 42 44 57 56
initial	-1345	80 81 80 81 85 79 75 52 53 59 50 59 53 85 79 76 74 73 74 75 76 74 75 76 73
Plow0	5403	71 77 72 68 81 80 10 6 8 17 9 6 7 9 6 7 80 10 7 80 56 58 56 57 62 44 57 62 0
Plow1	5405	Plow 1: 0 1 16 5 31 21 41 40 41 25 40 25 36 37 32 33 83 70 78 77 78 66 82
best	5405	65 83 84 3 84 64 65 83 84 33 27 29 11 13 1 2 11 13 2 3 48 84 64 48 63 52 82
time	809.886	65 66 64 65 66 82 85 79 75 52 82 85 81 78 70 68 81 78 68 67 69 39 69 28 33
aveCost	31.7219	27 11 13 0 1 2 3 11 13 2 11 3 48 84 33 83 70 67 69 39 35 37 32 35 34 32 35
aveDiff	3.84831	34 36 34 36 24 26 38 26 38 22 38 22 30 29 11 17 27 28 39 69 28 33 64 3 45
stdevCost	36.5932	47 63 46 52 49 47 63 48 63 48 63 46 63 49 50 59 50 59 53 59 53 60 61 58 54
stdevDiff	2.78933	51 50 59 61 58 60 61 55 59 53 85 79 76 73 71 77 72 71 72 68 67 70 68 78 66
		64 3 45 42 54 51 50 55 59 61 55 51 49 50 55 51 49 47 46 48 63 49 46 47 46
		52 49 46 48 63 46 63 46 47 45 54 58 60 43 56 43 7 10 43 42 54 45 42 44 62 0

fileName	hd515	Plow 0: 0 1 2 11 17 14 4 18 15 18 14 15 4 18 14 15 16 15 18 15 4 5 12 11 3
numPlows	3	48 63 46 47 45 54 58 56 43 7 9 6 7 9 6 7 80 12 5 29 11 13 1 16 5 12 11 27
numNodes	86	29 21 41 25 36 37 34 36 37 32 35 37 34 36 24 25 36 34 36 34 32 35 37 32 33
numArcs	178	64 3 84 33 83 70 68 67 70 68 78 66 64 48 63 48 63 46 48 63 52 53 85 79 76
OBJ	10808	73 71 77 72 68 81 80 10 6 8 10 43 42 44 57 56 57 62 44 62 0
initial	8542	Plow 1: 0 1 16 5 29 11 3 48 63 46 47 45 54 58 56 80 10 6 8 10 43 56 80 81
Plow0	3592	85 79 75 52 82 85 79 75 76 74 75 52 53 85 81 78 77 72 71 72 68 81 80 81 78
Plow1	3592	70 78 66 82 65 83 84 64 65 66 82 85 79 76 73 74 75 76 74 73 71 77 78 68 67
Plow2	3624	69 28 39 69 39 69 39 35 34 36 24 25 40 25 40 24 26 23 22 30 19 30 29 17 9
best	3624	8 17 27 11 17 14 4 5 31 21 19 20 22 20 23 22 30 19 31 21 19 31 30 19 20 23
time	2512.57	38 22 38 26 38 22 38 26 38 26 23 38 26 40 25 40 24 26 40 41 25 40 41 11 13
aveCost	31.7219	2 3 45 42 44 57 62 0
aveDiff	3.84831	Plow 2: 0 13 2 11 13 0 13 1 2 3 45 42 54 51 49 46 63 48 63 46 52 82 65 66
stdevCost	36.5932	64 48 63 49 47 63 46 52 49 46 47 46 47 46 63 46 48 63 52 49 47 63 49 50 59
stdevDiff	2.78933	50 59 50 55 51 50 59 53 59 53 59 53 60 43 42 54 51 49 50 55 51 50 59 53 60
		61 55 59 61 55 59 61 58 60 61 58 60 43 7 10 7 80 12 5 31 30 19 30 29 17 9
		8 17 27 28 33 27 28 33 83 84 33 64 3 84 48 84 64 65 83 70 67 69 28 39 35
		34 32 33 27 29 21 41 11 13 0

fileName	hd515	Plow 0: 0 1 16 15 4 18 15 18 15 16 5 12 80 10 43 7 9 6 8 17 27 28 39 35 34
numPlows	4	36 24 25 40 26 38 26 40 25 40 24 26 23 38 26 38 22 38 22 20 19 20 23 22 30
numNodes	86	29 17 14 15 4 18 14 4 5 29 11 27 28 33 64 65 66 64 3 11 41 25 36 37 32 35
numArcs	178	34 32 35 37 32 33 27 29 21 19 30 29 17 27 11 13 0
OBJ	10808	Plow 1: 0 1 16 5 31 5 12 11 17 9 6 7 10 6 7 80 10 7 80 81 78 68 81 85 79 76
initial	5008	73 71 72 71 77 78 66 82 65 83 70 68 67 70 68 67 69 39 69 39 69 28 39 35 37
Plow0	2714	34 36 34 36 34 36 24 26 23 38 22 20 19 31 30 19 31 30 19 20 23 22 38 22 30
Plow1	2710	19 30 19 21 41 40 24 25 40 25 40 41 25 36 37 34 32 33 83 70 67 69 28 33 83
Plow2	2689	84 3 45 42 44 62 44 57 62 0
Plow3	2695	Plow 2: 0 13 1 2 11 13 2 11 13 1 2 3 48 63 52 49 47 46 47 63 49 50 59 61
best	2714	58 54 45 42 54 51 50 59 50 55 59 53 60 43 7 9 8 10 6 8 17 14 15 18 14 4 5
time	5812.88	29 11 3 45 54 51 50 59 53 59 53 59 53 60 61 58 56 57 56 58 60 61 55 59 50
aveCost	31.7219	55 51 49 46 63 48 63 46 63 49 50 59 61 55 51 49 47 63 52 82 85 79 76 74 75
aveDiff	3.84831	52 82 85 79 75 52 53 85 81 80 56 43 42 44 57 62 0
stdevCost	36.5932	Plow 3: 0 13 2 3 84 33 64 3 48 84 64 65 66 64 48 63 46 52 49 46 47 45 47
stdevDiff	2.78933	46 52 53 85 79 75 76 74 75 76 73 74 73 71 77 72 68 81 78 77 72 68 78 70 78
		66 82 65 83 84 64 48 63 48 63 46 48 63 46 48 84 33 27 29 21 31 21 41 11 17
		9 8 10 43 42 54 58 60 43 56 80 81 80 12 11 13 0

fileName	hd515	Plow 0: 0 1 2 11 3 84 64 65 66 82 65 83 70 68 81 78 70 78 68 67 69 28 33
numPlows	5	83 84 33 27 29 21 31 30 29 17 9 8 10 43 7 10 6 7 9 8 10 6 7 9 6 8 17 14 15
numNodes	86	18 15 18 14 15 4 5 31 30 29 17 27 29 11 17 14 4 18 15 4 18 14 4 5 16 1 0
numArcs	178	Plow 1: 0 13 2 3 48 63 46 63 49 50 55 51 50 59 50 59 53 60 61 55 51 50 59
OBJ	10809	53 60 43 42 44 57 56 43 42 54 51 49 50 59 50 55 59 53 59 61 55 59 53 85 79
initial	8077	75 76 74 75 76 73 71 77 78 77 72 68 78 66 82 85 79 76 74 73 74 75 52 82 65
Plow0	2163	66 64 65 83 84 64 48 63 46 48 63 52 53 59 61 58 60 61 58 56 80 10 43 56 57
Plow1	2138	62 44 62 0
Plow2	2198	Plow 2: 0 13 1 2 11 3 45 54 58 60 43 7 80 10 7 80 81 80 81 78 66 64 3 84
Plow3	2141	33 64 48 63 46 47 45 54 58 56 80 12 11 13 0
Plow4	2169	Plow 3: 0 13 2 3 48 63 49 46 48 84 48 63 52 82 85 79 75 52 49 47 46 63 46
best	2198	47 63 48 63 48 63 46 52 53 85 81 85 79 76 73 71 72 71 77 72 68 81 80 12 5
time	7221.85	29 11 41 40 25 36 24 26 40 41 25 40 24 25 40 25 40 24 26 23 22 38 26 38 22
aveCost	31.7219	20 23 38 26 38 22 38 22 30 19 30 19 30 19 20 19 20 23 38 22 20 19 31 21 41
aveDiff	3.84831	11 13 0
stdevCost	36.5932	Plow 4: 0 13 1 16 5 12 5 29 21 41 25 36 34 32 33 83 70 67 70 68 67 69 28 33
stdevDiff	2.78933	27 11 27 28 39 69 39 35 37 34 36 37 34 36 24 25 40 26 23 22 30 19 21 19 31
		5 16 15 16 5 12 11 17 9 6 8 17 27 28 39 69 39 35 34 36 34 32 35 37 32 35 34
		36 37 32 33 64 3 45 42 54 51 49 46 47 63 46 47 46 52 49 47 45 42 44 57 62 0

fileName	hd615	Plow 0: 0 24 30 24 0 30 26 84 74 83 84 74 84 25 26 38 36 7 6 71 31 24 31
numPlows	2	83 31 83 74 75 66 67 68 67 74 84 25 26 84 87 81 80 79 81 80 86 43 61 57 56
numNodes	88	43 61 57 56 43 42 65 62 63 65 62 45 46 62 45 46 44 45 58 46 44 45 58 46 62
numArcs	181	63 65 57 60 55 53 54 53 55 54 59 49 73 77 48 50 59 49 77 78 70 69 71 70 69
OBJ	10012	76 78 69 76 78 69 71 70 76 77 87 83 84 87 83 31 29 22 28 23 22 24 31 29 22
initial	-4062	25 28 25 27 23 28 41 28 22 25 27 23 22 24 29 30 32 33 66 68 67 75 68 67 68
Plow0	5003	73 33 72 73 48 47 10 32 39 40 39 10 40 39 10 39 40 42 82 79 85 80 79 85 80
Plow1	5009	86 56 61 86 56 61 86 43 42 40 32 33 72 66 67 75 66 68 67 74 75 68 73 33 66
best	5009	72 73 48 47 13 12 1 0
time	262.368	Plow 1: 0 11 1 10 12 1 10 40 32 10 12 13 12 13 47 10 39 34 35 7 6 71 31 24
aveCost	28.9558	29 30 32 39 34 35 7 11 3 17 2 3 2 17 15 4 2 17 2 15 4 2 15 4 21 14 16 21 20
aveDiff	3.67956	16 21 20 8 9 19 8 9 18 51 8 20 16 5 6 0 30 26 38 36 26 38 35 11 37 7 11 3
stdevCost	27.1341	17 9 19 18 3 18 52 19 51 52 63 60 57 65 42 82 49 77 78 70 76 77 48 51 48
stdevDiff	2.70401	50 53 54 60 55 64 52 63 60 55 54 53 59 50 49 73 77 87 81 79 85 79 85 79 82
		49 50 53 59 54 60 55 64 58 44 64 58 44 64 52 51 19 18 52 19 8 9 18 51 8 9
		17 15 4 14 15 4 21 14 15 4 14 16 5 7 5 6 0 11 37 34 36 26 38 35 37 34 36 34
		36 34 36 7 35 37 38 37 7 35 11 1 0

fileName	hd615	Plow 0: 0 30 32 10 12 13 47 10 12 13 12 1 10 40 42 82 49 77 78 70 69 76 78
numPlows	3	69 71 70 76 78 70 69 71 31 29 30 24 31 24 30 26 38 37 34 36 34 35 7 6 71
numNodes	88	70 76 77 78 69 76 77 48 51 8 9 18 52 63 65 62 45 58 44 64 58 44 45 46 62
numArcs	181	63 65 57 56 43 61 86 56 61 86 56 61 57 56 43 61 57 60 55 64 52 19 18 52 51
OBJ	10012	8 9 19 51 52 19 8 20 16 5 7 11 37 7 11 1 0
initial	9314	Plow 1: 0 24 31 29 22 24 29 30 32 39 40 39 34 35 37 7 35 37 34 36 26 38 35
Plow0	3337	11 37 38 35 11 1 10 39 10 39 40 32 33 66 67 68 67 75 66 68 67 75 68 73 48
Plow1	3338	47 13 12 1 0 11 3 17 15 4 14 16 21 20 8 9 17 15 4 2 15 4 2 17 2 3 17 2 17 9
Plow2	3337	18 51 19 8 9 19 18 3 18 51 48 50 53 59 54 53 54 59 49 73 77 87 83 84 74 83
best	3338	84 74 84 25 28 41 28 23 22 25 26 84 87 83 31 83 31 83 74 84 25 27 23 28 22
time	2572.11	24 29 22 25 27 23 22 28 25 26 38 36 26 38 36 7 35 7 5 6 0
aveCost	28.9558	Plow 2: 0 30 26 84 87 81 80 79 85 79 85 79 85 80 86 43 42 82 79 85 80 79
aveDiff	3.67956	82 49 50 49 73 33 72 66 67 68 67 74 75 68 67 74 75 66 72 73 33 72 73 48 50
stdevCost	27.1341	53 55 54 53 59 50 59 49 77 48 47 10 40 32 39 10 32 33 66 68 73 77 87 81 79
stdevDiff	2.70401	81 80 86 43 42 65 62 45 58 46 44 45 46 44 64 58 46 62 63 60 55 54 60 55 53
		54 60 55 64 52 63 60 57 65 42 40 39 34 36 34 36 7 6 0 11 3 2 15 4 14 15 4
		21 14 15 4 21 14 16 21 20 16 5 6 71 31 24 0

fileName	hd615	Plow 0: 0 24 0 11 3 2 3 18 52 63 60 55 64 58 46 44 45 58 44 45 46 62 63 65
numPlows	4	62 63 60 55 64 52 51 48 47 10 32 33 72 73 48 47 10 39 40 32 33 66 68 67 68
numNodes	88	67 75 66 67 75 68 73 33 72 73 77 48 50 49 73 48 50 53 59 49 73 33 66 68 67
numArcs	181	74 84 74 75 66 72 66 67 74 84 87 83 74 83 31 29 22 24 31 24 29 30 32 39 40
OBJ	10012	39 34 35 11 37 7 11 1 0
initial	4708	Plow 1: 0 30 24 30 26 38 36 26 38 37 38 36 34 36 26 38 35 37 7 11 1 10 40
Plow0	2508	42 82 49 50 59 54 53 55 54 59 50 53 54 53 54 60 55 54 60 55 53 59 49 77 87
Plow1	2501	83 84 25 28 22 25 26 38 35 7 5 7 35 37 34 35 11 3 17 15 4 14 16 21 14 15 4
Plow2	2498	14 16 21 20 16 5 6 71 31 24 29 22 24 31 29 30 32 39 10 12 13 12 13 47 13
Plow3	2505	12 1 10 39 10 12 1 0
best	2508	Plow 2: 0 30 26 84 25 27 23 28 41 28 25 27 23 22 28 23 22 25 26 84 87 81
time	4537.38	80 86 56 61 57 56 61 86 56 43 42 65 62 45 58 44 64 58 46 62 45 46 44 64 52
aveCost	28.9558	19 18 52 19 8 9 19 51 8 9 18 51 19 8 9 18 3 17 2 17 15 4 2 15 4 2 17 9 19
aveDiff	3.67956	18 51 8 9 17 2 15 4 21 14 15 4 21 20 8 20 16 5 6 0
stdevCost	27.1341	Plow 3: 0 11 37 34 36 7 6 71 31 83 31 83 84 74 75 68 67 68 73 77 78 69 71
stdevDiff	2.70401	70 69 76 77 78 70 69 76 78 69 71 70 76 78 70 76 77 87 81 79 85 79 81 80 79
		85 79 85 80 79 82 79 85 80 86 43 61 86 43 61 57 60 57 65 42 82 49 77 48 51
		52 63 65 57 56 43 42 40 32 10 40 39 34 36 34 36 7 35 7 6 0

fileName	hd615	Plow 0: 0 30 26 38 36 7 11 37 34 36 26 38 35 7 35 7 6 71 70 76 78 70 76 77
numPlows	5	87 83 74 75 66 67 75 68 73 77 48 51 52 19 51 48 50 49 77 78 69 76 78 69 71
numNodes	88	31 24 29 22 28 25 26 38 35 11 1 0
numArcs	181	Plow 1: 0 11 3 2 3 17 2 15 4 14 15 4 2 17 15 4 21 14 15 4 2 15 4 14 16 5 6
OBJ	10012	71 70 69 71 31 24 31 83 31 83 31 29 30 24 29 22 24 30 26 38 37 38 36 26 84
initial	5490	25 26 84 25 28 41 28 23 28 22 25 27 23 22 25 27 23 22 24 0
Plow0	2010	Plow 2: 0 24 31 29 30 32 10 12 1 10 40 39 40 32 33 72 73 48 47 10 39 10 32
Plow1	2002	39 40 39 34 35 11 3 18 51 8 9 19 8 9 19 18 51 19 18 52 19 8 9 18 52 63 65
Plow2	1994	62 63 65 62 45 46 44 64 58 46 62 63 60 57 56 43 42 82 49 77 48 47 13 12 13
Plow3	1994	12 1 0
Plow4	2012	Plow 3: 0 30 32 33 66 68 67 75 68 73 33 72 66 67 74 84 74 84 74 75 66 68
best	2012	67 68 67 68 67 74 83 84 87 81 80 86 56 61 57 65 42 65 57 60 55 54 60 55 53
time	7221.6	55 54 53 54 60 55 64 58 46 44 45 46 62 45 58 44 45 58 44 64 52 63 60 55 64
aveCost	28.9558	52 51 8 9 18 3 17 9 17 2 17 15 4 21 20 8 20 16 21 20 16 21 14 16 5 7 35 37
aveDiff	3.67956	7 6 0
stdevCost	27.1341	Plow 4: 0 11 1 10 39 10 12 13 47 10 40 42 82 79 81 80 79 85 80 79 82 49 73
stdevDiff	2.70401	33 66 72 73 48 50 53 59 54 53 54 59 50 59 49 50 53 59 49 73 77 78 70 69 76
		77 87 83 84 87 81 79 85 79 85 79 85 80 86 56 61 57 56 43 61 86 43 61 86 43
		42 40 32 39 34 36 7 11 37 34 36 34 36 34 35 37 7 5 6 0

fileName	hd715	Plow 0: 0 25 26 29 42 43 98 43 48 49 45 30 31 26 45 30 31 42 30 29 31 26
numPlows	2	29 42 30 29 31 42 43 88 82 66 40 13 14 23 22 14 23 24 23 7 4 15 24 13 14
numNodes	100	24 23 24 23 7 62 3 2 69 60 51 60 59 51 55 53 65 50 52 65 50 52 64 54 52 64
numArcs	188	91 86 85 87 90 89 91 89 86 87 90 99 83 90 99 83 99 95 84 97 81 83 99 95 84
OBJ	8345	81 97 96 84 97 95 97 96 95 97 95 96 95 96 84 81 82 81 83 90 89 86 87 92 79
initial	-867	94 93 77 94 79 66 67 22 7 4 15 8 5 4 6 63 20 2 69 74 69 68 74 62 3 2 5 4 6
Plow0	4169	63 61 21 63 61 21 20 21 3 74 69 68 69 74 69 68 60 68 74 69 68 69 60 59 51
Plow1	4176	59 51 55 59 51 50 53 54 55 59 51 59 51 50 65 50 53 54 55 53 65 50 65 64 54
best	4176	52 65 64 91 86 85 87 92 79 94 77 94 77 80 94 79 76 78 75 77 80 75 76 80 94
time	532.799	79 76 80 75 76 78 75 77 94 93 78 98 88 82 81 82 66 67 40 41 8 5 6 19 47 28
aveCost	23.3723	27 25 26 45 49 46 33 34 36 38 44 38 44 38 44 33 32 34 44 36 35 37 38 44 36
aveDiff	3.68085	35 37 27 25 39 41 8 9 0
stdevCost	28.2297	Plow 1: 0 58 0 1 9 10 11 12 17 18 16 18 19 72 56 58 28 58 71 70 73 71 72
stdevDiff	2.69638	70 57 73 71 70 57 92 93 78 98 88 43 48 49 45 49 46 32 34 44 33 32 46 32 46
		33 34 36 38 35 39 41 47 19 72 56 57 73 67 40 41 47 28 56 57 92 93 77 94 79
		66 40 13 15 8 9 0 25 39 37 38 35 39 37 27 28 27 28 56 58 71 72 70 73 67 22
		14 24 13 15 24 23 22 7 62 61 21 3 74 62 61 21 63 20 2 5 6 19 18 16 17 11 12
		16 18 1 9 10 1 10 12 16 17 11 10 12 17 18 1 10 1 0

fileName	hd715	Plow 0: 0 25 39 41 8 5 4 6 19 47 19 72 56 57 92 93 77 80 94 77 94 79 66 40
numPlows	3	13 14 24 13 15 8 5 4 6 19 18 16 18 19 72 56 57 73 67 22 7 62 3 2 69 68 74
numNodes	100	62 3 74 62 61 21 63 61 21 63 61 21 20 21 3 2 5 6 63 20 2 5 6 63 20 2 69 68
numArcs	188	74 69 68 69 74 69 74 69 60 51 55 53 65 50 53 65 64 91 86 85 87 90 99 83 90
OBJ	8345	89 91 86 87 90 99 83 99 95 97 96 95 96 84 97 95 84 81 82 66 40 41 8 9 0
initial	5766	Plow 1: 0 58 28 27 25 26 29 42 30 31 42 43 48 49 46 32 34 44 36 38 44 33
Plow0	2779	32 34 44 36 35 37 27 28 56 58 71 70 57 73 71 72 70 73 71 72 70 73 67 22 14
Plow1	2783	23 7 62 61 21 3 74 69 60 59 51 59 51 60 68 69 68 60 59 51 59 51 50 52 64
Plow2	2783	54 52 65 50 52 64 54 55 59 51 55 53 54 52 65 50 65 50 53 54 55 59 51 50 65
best	2783	64 91 89 86 85 87 92 79 94 93 78 75 77 94 79 94 77 94 79 66 67 40 13 14 23
time	3133.85	22 7 4 15 24 23 22 14 24 23 7 4 15 24 23 24 23 24 13 15 8 9 10 11 12 17 18
aveCost	23.3723	16 17 18 1 10 1 9 10 1 9 0
aveDiff	3.68085	Plow 2: 0 25 26 45 30 31 26 29 31 42 43 48 49 45 49 45 30 29 42 30 29 31
stdevCost	28.2297	26 45 49 46 33 32 46 32 46 33 34 36 35 39 37 38 44 38 44 33 34 36 38 44 38
stdevDiff	2.69638	35 37 27 28 27 25 39 37 38 35 39 41 47 28 56 58 71 70 57 92 93 77 94 79 76
		80 75 76 78 75 77 80 75 76 78 98 43 88 43 98 88 82 81 83 99 95 97 81 97 96
		95 96 84 97 95 84 81 82 81 83 90 89 86 87 92 79 76 80 94 93 78 98 88 82 66
		67 40 41 47 28 58 0 1 10 12 16 17 11 12 17 11 10 12 16 18 1 0

fileName	hd715	Plow 0: 0 25 26 45 49 45 49 46 32 46 32 34 36 35 39 37 38 44 33 32 46 33
numPlows	4	34 44 38 44 36 38 44 33 34 36 35 39 41 47 28 56 57 92 93 78 75 77 94 79 66
numNodes	100	40 13 14 23 22 7 62 3 74 69 68 60 51 60 59 51 50 53 65 64 91 86 85 87 92
numArcs	188	79 76 80 94 77 94 77 94 93 78 75 76 78 98 88 82 81 83 99 83 90 99 83 90 89
OBJ	8345	86 85 87 92 79 94 79 94 93 77 80 75 77 94 79 66 40 41 8 9 0
initial	3288	Plow 1: 0 58 71 72 70 73 71 72 56 57 73 71 70 57 73 67 22 7 4 15 24 23 7
Plow0	2075	62 61 21 20 2 5 6 63 61 21 63 61 21 63 20 2 5 4 6 19 47 28 27 25 39 37 38
Plow1	2095	44 36 38 35 37 27 28 27 28 56 58 28 58 0
Plow2	2091	Plow 2: 0 1 10 12 16 17 11 12 16 17 18 1 10 12 17 11 10 11 12 17 18 16 18
Plow3	2084	19 72 56 58 71 70 73 67 40 41 8 5 4 6 63 20 21 3 2 69 68 69 74 69 68 74 69
best	2095	68 69 74 69 60 68 74 62 3 74 62 61 21 3 2 69 60 59 51 59 51 59 51 55 53 65
time	4040.19	50 65 50 65 64 54 52 65 50 53 54 52 65 50 52 64 54 55 59 51 55 53 54 55 59
aveCost	23.3723	51 50 52 64 91 86 87 90 89 91 89 86 87 90 99 95 97 81 83 99 95 96 95 97 95
aveDiff	3.68085	84 81 82 66 67 40 13 14 24 23 7 4 15 8 9 10 1 9 10 1 0
stdevCost	28.2297	Plow 3: 0 25 39 41 47 19 72 70 57 92 93 77 80 75 76 80 94 79 76 78 98 43
stdevDiff	2.69638	48 49 45 30 29 42 30 29 31 26 29 31 26 45 30 31 42 43 48 49 46 33 32 34 44
		38 35 37 27 25 26 29 42 30 31 42 43 88 43 98 88 82 81 97 96 84 97 95 96 84
		97 96 95 84 81 82 66 67 22 14 24 23 22 14 23 24 13 15 24 23 24 13 15 8 5 6
		19 18 16 18 1 9 0

fileName	hd715	Plow 0: 0 58 71 72 56 58 28 47 19 72 56 57 92 87 90 89 91 86 89 86 85 86
numPlows	5	87 92 87 92 93 78 75 76 80 75 77 80 75 77 80 94 79 94 77 94 79 94 77 94 79
numNodes	100	76 78 98 88 82 66 67 40 13 15 24 23 24 23 22 14 24 23 7 62 3 74 3 21 3 2 5
numArcs	188	4 15 8 9 10 1 0
OBJ	8390	Plow 1: 0 25 39 37 27 28 56 57 73 67 22 7 4 15 24 13 14 23 22 67 40 13 14
initial	3091	23 24 23 7 62 3 2 69 60 68 69 74 62 61 21 63 61 63 61 63 20 2 69 74 69 60
Plow0	1747	59 51 55 51 60 59 51 59 51 55 59 51 59 51 50 53 55 59 51 60 68 74 69 68 69
Plow1	1614	68 74 62 61 21 20 21 63 20 2 5 6 5 4 6 63 6 19 18 1 9 0
Plow2	1623	Plow 2: 0 58 28 27 25 26 45 49 45 30 31 42 30 29 31 42 30 31 26 45 30 29
Plow3	1749	42 43 98 88 43 48 49 46 33 34 44 38 44 38 44 33 32 46 33 32 46 32 34 36 35
Plow4	1657	37 38 35 39 41 47 19 72 70 73 67 66 40 41 8 5 4 6 19 18 1 0
best	1749	Plow 3: 0 25 26 29 31 26 29 42 43 88 82 81 83 99 83 99 95 96 95 97 95 84
time	12809.1	81 83 90 99 83 90 89 91 64 52 50 65 53 65 50 65 50 53 54 55 53 54 55 51 50
aveCost	23.3723	52 64 54 52 65 64 52 65 64 54 52 64 91 86 87 85 87 85 87 90 99 95 97 96 84
aveDiff	3.68085	97 81 82 81 97 95 96 95 84 97 96 84 81 82 66 40 41 47 28 56 58 0
stdevCost	28.2297	Plow 4: 0 58 71 70 73 71 70 57 73 71 72 70 57 92 93 77 94 93 77 94 93 78
stdevDiff	2.69638	75 76 80 94 79 66 79 92 79 76 78 98 43 48 49 45 49 46 32 34 44 36 38 44 36
		38 44 33 34 36 35 37 38 35 39 37 27 28 27 25 39 41 8 5 4 7 22 14 24 13 15
		8 9 10 11 12 17 18 16 17 18 16 17 11 12 10 12 16 12 17 11 10 1 9 0

fileName	hd815	Plow 0: 0 5 14 6 3 2 92 68 67 68 70 71 91 92 68 67 68 70 93 91 93 92 2 67
numPlows	2	68 2 3 6 14 33 31 41 75 41 42 40 39 38 36 37 32 31 14 15 14 33 8 15 14 15
numNodes	95	88 89 41 89 94 86 84 85 83 81 85 83 84 85 86 83 81 78 77 81 78 77 81 85 86
numArcs	193	83 84 56 57 60 66 61 66 64 59 55 53 58 56 57 60 61 54 60 66 61 54 60 61 66
OBJ	10786	65 64 59 55 61 66 64 62 46 10 88 89 94 86 84 56 90 94 57 65 46 13 88 62 46
initial	-3881	13 9 12 11 20 16 7 17 16 20 12 10 46 45 59 48 50 53 58 49 51 58 56 90 94
Plow0	5392	57 65 46 45 59 48 50 49 51 58 49 52 3 67 68 2 67 52 51 50 53 54 55 53 54
Plow1	5394	55 61 66 65 64 62 63 47 63 69 0
best	5394	Plow 1: 0 5 4 6 3 67 52 51 50 49 52 67 52 67 52 3 6 48 45 47 69 0 1 18 8 15
time	150.45	88 89 82 74 80 78 79 80 77 79 80 77 79 87 72 70 71 90 72 71 72 70 93 37 31
aveCost	29.1477	41 42 40 39 38 40 38 27 28 26 29 30 26 27 39 27 28 44 29 44 29 28 44 29 30
aveDiff	3.40415	25 43 30 26 27 39 27 39 36 40 38 40 38 27 39 36 40 38 36 37 32 31 14 44 23
stdevCost	35.3273	21 22 24 43 25 24 43 25 43 30 25 22 25 24 34 33 20 14 44 29 28 26 29 44 23
stdevDiff	2.68799	33 31 20 11 12 10 13 88 62 63 69 0 1 18 8 35 21 34 33 8 7 16 19 18 17 16
		19 7 17 19 18 17 19 7 8 35 23 21 22 24 34 35 21 34 35 23 33 20 12 9 13 11
		9 10 13 11 9 10 88 89 82 87 82 74 73 76 91 71 90 72 87 76 74 73 75 32 42
		75 32 42 75 73 76 74 80 78 79 87 82 87 72 87 76 91 92 93 37 31 20 14 4 1 5
		4 1 5 14 4 6 48 45 47 69 0

fileName	hd815	Plow 0: 0 5 14 44 23 21 34 33 31 20 11 9 10 88 89 82 74 73 76 91 71 72 70
numPlows	3	93 91 93 92 68 67 68 70 71 90 72 87 82 87 72 87 76 74 73 75 32 31 41 42 75
numNodes	95	41 89 94 86 84 85 83 84 85 86 83 81 78 77 81 85 83 81 85 86 83 84 56 90 72
numArcs	193	71 90 94 86 84 56 90 94 57 65 64 59 48 50 53 58 49 52 67 52 67 68 2 67 68
OBJ	10786	2 92 2 67 52 3 67 52 3 6 48 50 53 58 56 57 65 46 10 88 89 94 57 60 61 66 64
initial	6587	62 46 45 59 55 61 66 64 59 48 45 47 63 69 0
Plow0	3596	Plow 1: 0 1 18 17 16 20 14 44 29 30 26 29 44 29 44 29 28 26 29 28 44 23 33
Plow1	3592	8 35 21 22 24 34 35 23 21 34 33 31 41 42 75 73 76 74 80 78 77 81 78 79 87
Plow2	3598	76 91 92 68 67 68 70 71 91 92 93 37 31 14 4 6 14 33 20 16 19 7 17 16 19 7
best	3598	17 19 18 8 15 88 62 63 69 0 1 5 4 1 18 17 19 18 8 7 16 7 8 15 14 15 88 62
time	2108.98	46 45 59 55 61 54 55 53 54 60 66 61 66 65 64 62 63 47 69 0
aveCost	29.1477	Plow 2: 0 5 14 33 8 35 21 22 24 34 35 23 33 20 14 4 1 5 4 6 3 2 3 67 52 51
aveDiff	3.40415	50 49 51 50 49 51 58 49 52 51 58 56 57 60 61 54 55 53 54 60 66 61 66 65 46
stdevCost	35.3273	13 88 89 82 74 80 77 79 80 77 79 80 78 79 87 82 87 72 70 93 37 31 20 12 10
stdevDiff	2.68799	13 11 20 12 11 12 10 13 9 10 46 13 11 9 12 9 13 88 89 41 75 32 42 40 38 40
		39 38 36 37 32 42 40 38 40 39 38 36 40 38 27 39 36 40 38 27 28 44 29 30 25
		43 30 25 22 25 24 43 25 24 43 25 43 30 26 27 28 26 27 39 27 39 27 39 36 37
		32 31 14 15 14 6 3 6 48 45 47 69 0

fileName	hd815	Plow 0: 0 5 4 6 3 2 3 6 14 4 6 48 50 53 58 49 51 50 49 51 58 49 52 51 50 53
numPlows	4	54 55 61 66 61 66 65 46 13 88 89 82 74 73 75 32 31 14 44 23 21 34 35 23 33
numNodes	95	8 15 88 62 46 13 9 10 13 11 20 11 9 13 11 9 12 11 12 10 13 88 62 63 47 63
numArcs	193	69 0
OBJ	10786	Plow 1: 0 1 18 8 7 17 19 18 17 16 19 7 16 20 16 19 18 8 35 21 34 33 20 14
initial	3910	33 8 35 21 22 25 43 25 22 24 43 30 26 27 28 44 29 28 26 29 44 29 28 26 27
Plow0	2699	39 27 39 36 40 39 27 39 36 37 31 20 14 6 48 50 49 52 3 6 3 67 52 67 52 67
Plow1	2694	52 3 67 68 2 67 68 2 67 68 70 93 37 31 41 75 32 42 40 38 27 39 38 36 40 39
Plow2	2695	38 40 38 40 38 36 37 32 42 75 41 89 82 74 80 78 77 81 85 83 81 85 83 84 56
Plow3	2698	90 94 57 65 64 62 63 69 0
best	2699	Plow 2: 0 5 4 1 5 14 4 1 18 17 19 7 17 16 7 8 15 14 15 88 89 41 42 40 38 27
time	1053.73	28 44 29 30 26 29 44 23 33 31 20 12 10 88 89 94 86 83 84 85 86 84 85 86 84
aveCost	29.1477	56 90 72 87 76 74 80 77 79 80 77 81 78 79 80 78 79 87 82 87 72 87 76 91 71
aveDiff	3.40415	90 94 57 65 64 59 48 45 59 55 53 58 56 57 60 61 54 60 61 66 61 66 65 46 45
stdevCost	35.3273	47 69 0
stdevDiff	2.68799	Plow 3: 0 1 5 14 33 31 41 42 75 73 76 74 73 76 91 92 2 92 93 91 92 68 67
		68 70 71 90 72 70 71 72 70 93 37 32 31 14 15 14 44 29 30 25 24 43 25 43 30
		25 24 34 35 23 21 22 24 34 33 20 12 9 10 46 10 88 89 94 86 83 81 78 77 79
		87 82 87 72 71 91 93 92 68 67 52 51 58 56 57 60 66 64 62 46 45 59 55 53 54
		55 61 54 60 66 64 59 48 45 47 69 0

fileName	hd815	Plow 0: 0 5 14 15 88 89 94 86 84 85 86 83 81 85 83 84 56 90 72 87 82 74 80
numPlows	5	77 79 80 78 77 81 78 77 81 85 83 81 78 79 80 77 79 87 72 70 93 92 93 91 93
numNodes	95	37 32 42 40 38 27 39 36 40 39 38 27 39 36 37 32 31 41 42 40 38 36 40 38 40
numArcs	193	39 27 28 44 29 28 26 29 30 25 43 30 26 27 28 44 29 28 26 27 39 27 39 38 40
OBJ	10791	38 36 37 31 20 14 4 6 48 45 47 69 0
initial	5324	Plow 1: 0 5 4 6 3 6 14 44 23 21 22 24 34 35 21 22 25 43 30 26 29 44 23 33
Plow0	2159	20 12 11 20 11 12 9 13 88 89 94 57 60 61 66 65 64 62 46 13 88 62 46 13 9
Plow1	2148	12 10 88 62 63 47 69 0
Plow2	2149	Plow 2: 0 5 4 1 5 14 15 14 33 8 7 17 19 7 16 19 7 17 16 20 16 7 8 35 23 21
Plow3	2166	34 35 23 33 20 12 10 13 11 9 10 13 11 9 10 88 89 82 74 73 75 73 76 74 80
Plow4	2169	78 79 87 76 91 92 68 2 3 6 3 2 92 68 67 68 2 67 68 67 68 70 93 37 31 41 42
best	2169	75 32 31 20 14 4 1 5 0
time	5817.37	Plow 3: 0 1 18 8 35 21 34 33 8 15 14 33 31 14 44 29 44 29 30 25 22 24 43
aveCost	29.1477	25 24 43 25 24 34 33 31 14 6 48 45 59 48 50 53 58 49 51 50 53 54 60 66 65
aveDiff	3.40415	46 10 46 45 59 48 50 49 52 51 50 49 51 58 49 52 67 52 3 67 52 3 67 52 67
stdevCost	35.3273	52 51 58 56 57 65 46 45 47 63 69 0
stdevDiff	2.68799	Plow 4: 0 1 18 17 19 18 17 16 19 18 8 15 88 89 41 75 32 42 75 41 89 82 87
		72 87 82 87 76 74 73 76 91 71 90 94 57 60 61 54 55 61 66 61 66 64 59 55 53
		54 55 53 58 56 90 72 71 91 92 2 67 68 70 71 72 70 71 90 94 86 84 85 86 83
		84 56 57 65 64 59 55 61 54 60 66 61 66 64 62 63 69 0

fileName	hd915	Plow 0: 0 1 16 18 16 15 36 46 37 36 46 37 36 38 34 40 34 26 27 40 34 40 0
numPlows	2	9 78 35 26 24 25 32 28 31 33 31 24 26 24 26 24 25 33 47 33 47 33 31 33 32
numNodes	96	25 33 32 28 29 92 83 74 84 73 72 82 80 81 86 81 80 81 56 58 67 62 61 67 62
numArcs	189	61 67 62 63 68 66 65 49 94 65 49 94 65 69 94 88 93 87 88 89 87 90 57 86 56
OBJ	9041	55 17 18 16 18 17 16 15 36 38 34 26 24 27 40 0 1 16 18 55 17 16 18 55 15 1
initial	-2337	2 4 3 5 20 3 21 4 2 6 7 43 45 30 45 30 45 44 6 7 8 9 7 8 10 19 5 23 19 20 21
Plow0	4516	3 5 20 3 4 22 59 13 22 60 59 13 11 14 21 14 21 4 22 59 11 12 23 10 19 20 21
Plow1	4525	14 22 13 14 13 11 14 22 60 54 52 50 54 63 61 58 57 67 62 68 70 71 66 69 49
best	4525	48 46 15 37 41 44 6 42 6 2 0
time	430.119	Plow 1: 0 9 78 35 91 95 86 85 76 78 77 79 86 85 76 79 85 77 76 78 77 79 85
aveCost	25.0106	77 76 79 86 95 92 83 74 84 73 72 80 81 75 73 72 39 82 81 75 74 73 72 80 81
aveDiff	3.37566	57 67 62 63 68 66 65 69 49 48 71 66 69 94 88 93 87 88 89 95 92 91 83 92 91
stdevCost	24.1642	95 89 87 90 48 71 64 70 64 70 71 64 53 51 50 54 52 50 53 51 50 53 52 51 64
stdevDiff	2.57412	70 64 53 52 51 64 70 60 13 60 59 11 12 23 19 5 23 10 12 10 8 9 7 43 45 42
		45 44 42 41 43 30 29 92 83 84 75 74 73 72 39 38 39 82 80 81 80 81 56 58 67
		62 68 70 60 54 63 61 58 57 90 48 46 15 37 41 44 42 41 43 30 29 34 29 28 30
		28 31 24 27 35 26 27 35 91 83 92 83 84 75 73 72 82 81 57 86 56 59 56 55 15
		1 2 0

fileName	hd915	Plow 0: 0 1 2 0 9 7 8 9 7 8 9 78 35 26 24 25 32 25 33 31 24 25 33 31 24 26
numPlows	3	24 26 24 27 35 26 24 27 35 91 83 74 73 72 80 81 57 86 56 55 17 16 18 16 18
numNodes	96	16 18 17 16 18 55 17 18 55 15 1 16 15 1 16 15 36 38 34 26 27 40 34 29 28
numArcs	189	31 33 47 33 47 33 32 28 29 92 91 95 92 83 84 73 72 80 81 56 58 67 62 63 61
OBJ	9041	67 62 63 61 67 62 68 66 65 69 49 94 65 49 48 46 15 37 36 46 37 41 43 30 45
initial	7957	30 45 42 6 2 6 7 43 45 44 6 42 41 44 42 41 44 42 45 30 29 34 40 34 40 0
Plow0	3016	Plow 1: 0 1 2 4 3 4 22 13 11 12 10 8 10 12 23 10 19 5 20 3 5 20 3 5 23 10
Plow1	3010	19 5 23 19 20 21 3 21 4 22 59 11 12 23 19 20 21 14 13 11 14 21 14 22 59 11
Plow2	3015	14 22 60 54 52 50 53 52 51 64 53 51 50 54 52 50 54 63 68 70 60 59 13 60 54
best	3016	63 68 70 64 53 51 50 53 52 51 64 70 64 70 71 64 70 71 66 69 49 94 65 49 48
time	756.544	46 15 37 36 46 37 41 43 45 44 6 7 43 30 28 31 33 32 28 30 29 92 83 84 73
aveCost	25.0106	72 39 38 34 26 27 40 0
aveDiff	3.37566	Plow 2: 0 9 78 35 91 83 74 73 72 82 81 75 74 84 75 74 84 75 73 72 82 81 80
stdevCost	24.1642	81 80 81 86 85 76 78 77 76 78 77 76 79 85 76 79 85 77 79 86 85 77 79 86 95
stdevDiff	2.57412	92 83 92 83 92 91 95 89 87 88 93 87 90 48 71 66 69 94 88 93 87 90 57 90 48
		71 64 70 60 13 22 60 59 56 55 15 36 38 39 82 80 81 57 67 62 61 58 57 67 62
		61 58 57 86 56 58 67 62 68 66 65 69 94 88 89 87 88 89 95 86 81 75 73 72 39
		82 80 81 56 59 13 14 21 4 2 0

fileName	hd915	Plow 0: 0 1 16 18 16 18 17 16 18 55 17 16 15 37 36 46 37 41 44 6 42 41 43
numPlows	4	45 44 42 45 44 42 41 44 6 7 8 10 19 20 3 5 20 21 14 21 14 13 11 14 21 4 22
numNodes	96	60 13 60 54 52 50 54 52 51 50 53 52 50 53 52 51 64 70 60 59 13 14 22 60 54
numArcs	189	63 61 58 57 90 48 46 15 36 38 39 82 80 81 56 55 17 18 16 15 37 41 43 30 45
OBJ	9041	30 28 31 24 26 24 25 33 47 33 47 33 32 28 29 28 31 33 31 33 32 28 30 45 42
initial	6412	6 2 0
Plow0	2264	Plow 1: 0 1 16 18 55 15 36 46 15 1 2 4 22 59 11 14 22 13 11 12 23 19 20 3
Plow1	2260	21 4 3 5 20 21 3 4 2 6 7 43 30 29 92 83 92 91 83 74 73 72 82 81 56 58 67 62
Plow2	2255	68 70 64 70 71 64 70 60 59 11 12 23 10 12 10 19 5 23 19 5 23 10 8 9 78 35
Plow3	2262	26 27 40 34 26 24 27 35 26 24 25 32 25 33 31 24 26 27 40 0
best	2264	Plow 2: 0 9 7 43 45 30 29 92 83 92 91 95 86 85 77 79 86 81 80 81 75 74 84
time	6418.09	73 72 80 81 86 85 77 76 78 35 91 83 84 75 73 72 39 38 34 26 24 27 35 91 95
aveCost	25.0106	92 83 74 73 72 39 82 80 81 57 67 62 63 68 70 64 53 51 64 53 51 50 54 63 68
aveDiff	3.37566	66 69 94 65 49 94 88 89 87 88 89 95 89 87 88 93 87 90 57 86 56 55 15 1 2 0
stdevCost	24.1642	Plow 3: 0 9 7 8 9 78 77 79 85 76 78 77 76 79 85 76 79 86 56 59 13 22 59 56
stdevDiff	2.57412	58 67 62 61 67 62 61 67 62 63 61 58 57 86 95 92 83 84 75 74 84 73 72 80 81
		75 73 72 82 81 80 81 57 67 62 68 66 65 69 94 88 93 87 90 48 71 64 70 71 66
		69 49 48 71 66 65 69 49 94 65 49 48 46 37 36 38 34 40 34 29 34 40 0

fileName	hd915	Plow 0: 0 9 7 43 30 29 28 30 45 44 42 41 44 6 7 8 9 7 43 45 42 41 43 45 42
numPlows	5	6 7 8 9 78 77 76 79 86 85 77 76 79 86 81 75 74 84 75 73 72 80 81 80 81 57
numNodes	96	90 57 86 95 86 56 55 17 18 55 17 16 18 16 18 16 15 1 0
numArcs	189	Plow 1: 0 1 16 18 55 15 37 36 46 15 36 46 37 41 43 30 29 92 91 83 92 83 74
OBJ	9042	84 75 74 73 72 80 81 56 55 15 1 2 4 22 59 56 58 57 86 85 77 79 85 76 78 35
initial	4424	26 24 26 27 35 26 24 25 33 31 24 27 35 91 95 92 91 83 74 73 72 39 38 34 40
Plow0	1765	34 26 24 25 33 31 24 26 24 27 40 34 26 27 40 0
Plow1	1838	Plow 2: 0 40 34 40 34 29 92 83 84 73 72 82 81 57 67 62 61 58 67 62 61 67
Plow2	1780	62 68 66 69 49 94 65 49 48 71 66 69 94 88 93 87 90 48 71 64 70 60 54 63 61
Plow3	1812	58 67 62 63 68 70 64 53 51 64 70 71 66 65 69 94 65 69 49 94 88 89 87 90 48
Plow4	1847	46 37 36 38 34 29 28 31 33 32 25 32 28 31 33 47 33 47 33 32 28 30 45 44 42
best	1847	6 2 0
time	5545.61	Plow 3: 0 9 78 77 79 85 76 78 35 91 95 89 87 88 93 87 88 89 95 92 83 92 83
aveCost	25.0106	84 73 72 82 80 81 56 59 13 60 59 13 22 60 13 11 12 23 19 20 21 3 5 20 3 5
aveDiff	3.37566	23 19 5 23 10 19 5 20 21 14 22 59 11 14 21 14 21 4 2 1 0
stdevCost	24.1642	Plow 4: 0 1 16 18 17 16 15 36 38 39 82 80 81 80 81 75 73 72 39 82 81 86 56
stdevDiff	2.57412	58 57 67 62 63 68 70 64 70 71 64 53 51 64 70 60 59 11 14 22 13 14 13 11 12
		23 10 12 10 8 10 19 20 3 4 3 21 4 22 60 54 52 51 50 54 52 50 53 52 50 53 52
		51 50 54 63 61 67 62 68 66 65 49 48 46 15 37 41 44 6 2 0

fileName	hg115	Plow 0: 0 2 7 1 7 13 15 25 20 12 11 12 15 25 31 33 38 22 27 16 6 9 10 17 23
numPlows	2	21 15 25 26 22 21 8 21 15 25 15 25 15 25 20 30 20 30 32 33 38 40 45 40 45
numNodes	60	40 45 39 34 35 28 29 36 41 37 35 37 35 28 29 36 41 37 39 34 31 33 54 44 54
numArcs	105	44 54 56 47 46 48 49 58 59 47 46 48 49 58 59 57 55 57 59 57 55 57 55 54 56
OBJ	420	47 49 51 53 42 52 50 51 50 34 35 36 35 36 35 37 39 27 16 27 16 14 13 11 1 0
initial	-4	Plow 1: 0 2 7 2 3 5 6 9 6 39 38 40 56 57 55 54 44 43 32 43 40 45 46 45 39
Plow0	211	38 22 21 8 21 14 13 15 25 31 30 32 33 54 44 43 40 56 57 59 47 49 51 53 42
Plow1	209	41 42 52 37 52 50 48 27 16 14 26 22 27 48 50 34 31 30 20 12 15 25 26 14 4
best	211	3 2 7 4 8 4 3 5 6 9 6 9 10 18 29 18 10 17 16 21 14 4 8 6 39 27 24 34 24 28
time	34.467	19 10 18 10 17 23 19 23 19 23 24 28 19 10 17 16 27 24 23 24 23 21 16 6 8 4
aveCost	2.68095	7 2 7 13 11 1 0
aveDiff	1.93333	
stdevCost	2.17302	
stdevDiff	1.88865	

fileName	hg115	Plow 0: 0 2 7 2 7 2 7 4 3 5 6 39 27 48 49 51 50 34 35 28 29 36 35 37 39 38
numPlows	3	22 27 16 6 9 6 9 6 8 21 8 21 14 13 15 25 26 14 13 11 12 15 25 15 25 15 25
numNodes	60	20 30 32 33 54 56 57 59 47 49 51 53 42 41 42 52 37 52 50 48 50 51 53 42 52
numArcs	105	50 34 31 30 20 12 11 1 0
OBJ	422	Plow 1: 0 1 7 13 15 25 26 22 21 14 26 22 27 24 28 19 23 24 28 19 23 24 34
initial	414	31 33 38 40 45 40 45 40 56 47 46 45 46 48 27 16 27 16 6 9 10 17 16 14 4 8
Plow0	138	6 39 27 24 23 19 10 17 23 21 15 25 31 33 54 44 54 44 43 32 43 40 45 39 34
Plow1	143	24 23 21 16 27 16 14 4 7 13 11 1 0
Plow2	141	Plow 2: 0 2 3 5 6 9 10 18 10 17 16 21 15 25 31 30 20 12 15 25 20 30 32 33
best	143	38 40 56 47 46 48 49 58 59 47 49 58 59 57 55 57 55 57 55 54 44 54 56 57 59
time	224.403	57 55 54 44 43 40 45 39 34 35 28 29 18 10 17 23 19 10 18 29 36 35 36 41 37
aveCost	2.68095	35 36 41 37 35 37 39 38 22 21 8 4 8 4 3 2 7 1 0
aveDiff	1.93333	
stdevCost	2.17302	
stdevDiff	1.88865	

fileName	hg115	Plow 0: 0 1 7 13 15 25 26 22 21 16 27 16 6 39 38 22 21 8 6 9 6 9 6 39 27 24
numPlows	4	23 24 28 29 18 29 36 35 37 35 28 29 36 35 37 39 27 16 27 24 28 19 23 24 23
numNodes	60	21 15 25 26 14 13 11 12 15 25 20 12 11 1 0
numArcs	105	Plow 1: 0 1 7 4 3 5 6 9 10 18 10 17 23 19 23 21 15 25 15 25 31 33 38 40 56
OBJ	422	47 46 45 39 34 31 33 54 44 43 40 45 40 45 40 45 39 34 35 36 41 42 52 50 48
initial	224	27 16 6 9 10 17 16 21 14 13 11 1 0
Plow0	109	Plow 2: 0 2 7 13 15 25 15 25 20 12 15 25 31 30 20 30 20 30 32 33 54 44 54
Plow1	108	44 54 44 43 40 45 46 48 49 58 59 57 59 57 59 47 49 58 59 47 49 51 53 42 52
Plow2	102	50 34 31 30 32 43 32 33 38 22 27 16 14 4 8 4 8 4 3 2 0
Plow3	103	Plow 3: 0 2 3 5 6 8 21 8 21 14 26 22 27 48 49 51 53 42 41 37 39 38 40 56
best	109	57 55 57 55 57 55 54 56 57 55 54 56 47 46 48 50 51 50 34 24 34 35 36 41 37
time	1431.17	52 37 35 28 19 10 17 23 19 10 18 10 17 16 14 4 7 2 7 2 0
aveCost	2.68095	
aveDiff	1.93333	
stdevCost	2.17302	
stdevDiff	1.88865	

fileName	hg115	Plow 0: 0 1 7 13 15 25 20 30 32 33 54 44 43 40 45 39 38 22 27 16 27 16 21
numPlows	5	8 21 15 25 26 22 27 16 27 24 34 35 37 35 37 52 50 34 31 33 54 44 43 32 33
numNodes	60	38 40 56 47 49 51 53 42 52 37 39 38 22 21 14 26 22 21 15 25 26 14 4 3 2 0
numArcs	105	Plow 1: 0 2 3 5 6 9 10 18 29 36 41 37 35 36 35 28 29 18 10 17 23 24 28 19
OBJ	424	10 17 23 19 10 17 16 6 9 6 9 6 39 27 24 23 24 23 21 14 13 11 12 15 25 20 12
initial	322	15 25 15 25 31 30 20 30 20 12 11 1 0
Plow0	105	Plow 2: 0 2 7 13 15 25 15 25 31 33 38 40 45 46 45 40 56 47 46 48 49 51 53
Plow1	106	42 41 37 39 27 48 49 58 59 57 55 57 59 47 49 58 59 57 55 54 44 54 56 57 55
Plow2	106	57 55 54 44 54 56 57 59 47 46 48 27 16 14 4 3 5 6 8 4 8 4 7 2 0
Plow3	4	Plow 3: 0 1 7 2 0
Plow4	103	Plow 4: 0 1 7 4 8 21 16 6 39 34 35 28 29 36 35 36 41 42 52 50 51 50 48 50
best	106	34 31 30 32 43 40 45 40 45 39 34 24 28 19 23 19 23 21 8 6 9 10 18 10 17 16
time	344.272	14 13 11 1 0
aveCost	2.68095	
aveDiff	1.93333	
stdevCost	2.17302	
stdevDiff	1.88865	

fileName	hg215	Plow 0: 0 1 6 5 16 5 0 5 16 14 15 38 40 41 33 32 33 43 22 25 36 25 23 21
numPlows	2	22 34 13 4 21 12 20 28 19 28 19 56 58 63 54 56 58 63 54 41 40 30 31 29 31
numNodes	68	29 31 29 20 28 57 64 57 49 48 42 47 42 40 41 40 41 31 27 7 6 8 2 8 9 11 4
numArcs	119	21 12 20 22 25 36 44 51 44 35 43 66 65 64 65 49 59 66 67 62 67 60 51 44 51
OBJ	450	44 45 37 36 44 45 37 45 37 26 24 26 24 26 25 36 37 26 25 36 37 45 46 61 60
initial	-9	59 66 65 49 48 42 39 38 40 41 31 27 7 6 5 16 17 30 31 29 20 22 21 22 34 13
Plow0	226	4 3 2 1 0
Plow1	224	Plow 1: 0 5 16 17 30 40 42 47 52 53 55 47 42 39 38 15 17 27 19 28 57 49 59
best	226	50 48 33 32 29 32 34 36 34 35 43 22 21 12 11 4 3 2 1 6 8 18 7 16 14 15 17
time	129.372	27 19 18 10 28 19 18 10 28 19 56 57 64 58 56 57 64 57 64 58 56 54 56 54 47
aveCost	2.65126	55 63 54 47 52 39 52 53 55 47 55 63 54 41 33 43 66 67 60 59 50 48 33 32 33
aveDiff	2.15966	32 34 35 50 51 44 35 50 51 46 45 46 61 62 67 62 61 60 51 46 45 37 36 25 23
stdevCost	2.02511	24 23 13 23 21 12 10 9 11 12 10 9 8 18 7 16 5 0
stdevDiff	1.81489	

fileName	hg215	Plow 0: 0 5 16 17 30 31 27 19 18 10 9 11 4 3 2 8 9 8 18 7 16 14 15 17 27 7
numPlows	3	6 8 2 1 6 5 16 17 30 31 29 32 33 43 22 34 35 50 51 44 45 46 61 60 59 66 67
numNodes	68	62 61 60 51 46 45 46 45 37 36 44 51 46 61 62 67 62 67 60 59 66 67 60 51 44
numArcs	119	35 43 22 21 12 20 28 57 64 58 63 54 56 57 64 65 64 57 49 48 42 39 38 15 17
OBJ	450	27 7 6 5 0
initial	444	Plow 1: 0 5 16 5 16 14 15 38 40 41 40 42 47 55 47 55 47 52 39 52 53 55 63
Plow0	153	54 56 58 56 58 56 57 64 57 49 48 42 40 41 33 32 33 32 29 20 22 21 22 25 36
Plow1	145	25 36 37 26 24 26 25 36 37 26 24 23 13 4 21 12 10 9 11 4 21 12 10 28 19 28
Plow2	152	19 28 19 56 54 47 42 47 42 39 38 40 30 40 41 31 29 31 29 31 27 19 18 7 16
best	153	5 0
time	405.662	Plow 2: 0 1 6 8 18 10 28 19 56 54 47 52 53 55 63 54 41 40 41 33 43 66 65
aveCost	2.65126	49 59 50 51 44 51 44 35 50 48 33 32 34 36 44 45 37 45 37 45 37 36 25 23 24
aveDiff	2.15966	26 25 23 21 22 25 36 34 35 43 66 65 49 59 50 48 33 32 34 13 23 21 12 11 12
stdevCost	2.02511	20 28 57 64 58 63 54 41 31 29 20 22 34 13 4 3 2 1 0
stdevDiff	1.81489	

fileName	hg215	Plow 0: 0 1 6 8 18 10 28 19 56 58 63 54 47 55 47 52 39 38 15 17 30 40 42
numPlows	4	47 42 40 41 40 41 33 43 66 67 60 51 44 45 37 45 46 61 60 51 46 45 46 61 60
numNodes	68	59 50 48 42 47 42 39 52 53 55 47 55 63 54 56 57 64 57 49 48 33 32 34 13 4
numArcs	119	3 2 1 0
OBJ	450	Plow 1: 0 5 16 14 15 38 40 30 31 29 31 29 32 33 32 33 32 29 20 22 34 35 50
initial	238	51 44 51 44 51 46 45 37 36 37 26 24 26 25 23 24 26 25 36 37 45 37 26 24 23
Plow0	113	21 12 20 28 19 28 57 64 57 64 58 56 54 56 54 41 31 29 31 27 19 28 19 18 7
Plow1	113	16 5 16 5 0
Plow2	112	Plow 2: 0 1 6 5 16 14 15 17 30 31 27 7 16 17 27 19 18 10 28 57 49 48 33 32
Plow3	112	34 13 23 21 22 34 36 34 35 50 48 42 39 38 40 41 40 41 31 29 20 22 25 36 44
best	113	45 37 36 25 36 25 36 44 35 43 22 21 22 21 12 11 4 21 12 10 9 8 2 8 18 7 6 5
time	836.968	0
aveCost	2.65126	Plow 3: 0 5 16 17 27 7 6 8 9 11 12 10 9 11 4 21 12 20 28 19 56 57 64 65 49
aveDiff	2.15966	59 66 65 49 59 66 67 62 67 62 61 62 67 60 59 50 51 44 35 43 66 65 64 58 56
stdevCost	2.02511	58 63 54 47 52 53 55 63 54 41 33 43 22 25 23 13 4 3 2 1 0
stdevDiff	1.81489	

fileName	hg215	Plow 0: 0 1 6 8 9 11 12 20 28 19 28 19 56 54 47 42 47 55 63 54 47 55 63 54
numPlows	5	56 58 56 57 64 58 63 54 56 58 56 54 41 31 29 32 29 31 27 7 6 5 0
numNodes	68	Plow 1: 0 5 16 17 27 19 28 19 18 10 28 57 49 48 42 39 52 53 55 47 52 53 55
numArcs	119	47 42 39 38 15 17 30 31 29 31 29 20 22 34 35 43 66 65 49 59 66 65 64 58 63
OBJ	452	54 41 33 32 33 32 33 32 34 13 4 3 2 1 0
initial	440	Plow 2: 0 1 6 8 18 10 28 57 64 57 64 57 64 65 49 48 33 32 34 35 43 66 67
Plow0	80	60 59 66 67 60 51 44 45 37 26 25 36 37 26 25 36 34 36 44 35 50 51 46 45 46
Plow1	86	45 37 36 25 36 37 36 25 23 21 22 21 12 11 4 3 2 1 0
Plow2	95	Plow 3: 0 5 16 14 15 17 30 40 30 31 29 20 22 25 36 44 35 50 51 44 51 46 61
Plow3	96	62 67 62 67 62 61 60 51 44 51 44 45 37 45 37 45 46 61 60 59 50 48 42 47 52
Plow4	95	39 38 40 42 40 41 33 43 22 34 13 23 21 12 20 28 19 18 7 6 5 0
best	96	Plow 4: 0 5 16 17 27 7 16 14 15 38 40 41 40 41 40 41 31 27 19 56 57 49 59
time	772.728	50 48 33 43 22 21 12 10 9 11 4 21 22 25 23 24 26 24 26 24 23 13 4 21 12 10
aveCost	2.65126	9 8 2 8 18 7 16 5 16 5 0
aveDiff	2.15966	
stdevCost	2.02511	
stdevDiff	1.81489	

fileName	hg315	Plow 0: 0 1 3 8 13 8 10 14 10 29 48 63 59 58 56 47 48 58 49 48 58 60 61 62
numPlows	2	51 50 46 49 48 37 45 41 50 51 60 61 59 61 62 51 60 46 49 37 36 23 25 38 36
numNodes	64	23 16 15 14 10 14 22 58 22 21 20 13 20 13 20 34 43 44 55 53 52 54 57 56 55
numArcs	116	53 42 32 31 33 42 43 37 58 49 38 36 35 36 35 47 44 43 34 43 37 49 38 39 40
OBJ	455	41 50 51 50 46 45 39 45 41 40 27 26 27 26 27 26 18 7 18 7 6 5 6 5 6 17 12
initial	-32	11 5 11 9 10 21 22 16 15 12 11 9 4 2 0
Plow0	228	Plow 1: 0 1 19 31 33 52 54 57 56 47 48 63 57 63 59 58 56 55 53 52 33 42 32
Plow1	227	13 32 34 35 47 44 55 53 42 43 34 35 28 29 23 25 38 39 40 27 26 24 16 15 12
best	228	24 25 30 25 30 17 6 7 18 17 30 25 30 39 30 27 26 18 17 12 24 25 30 27 26
time	56.951	24 16 15 4 2 3 9 4 5 11 5 4 5 4 2 3 9 10 21 28 21 20 34 32 31 19 13 19 1 3 8
aveCost	2.69397	10 29 23 16 15 14 22 29 48 37 58 60 46 45 37 45 37 49 37 36 35 36 35 28 29
aveDiff	1.93966	22 16 15 4 2 0
stdevCost	1.99921	
stdevDiff	1.88135	

fileName	hg315	Plow 0: 0 1 3 8 10 14 10 14 10 21 20 13 8 10 21 20 13 19 1 3 8 13 20 34 32
numPlows	3	13 20 34 35 36 35 36 35 47 44 55 53 42 43 37 58 49 38 39 30 39 40 27 26 24
numNodes	64	25 38 39 40 41 50 46 45 39 45 41 50 46 49 48 58 56 47 48 58 56 47 48 63 57
numArcs	116	56 55 53 42 43 44 43 34 35 28 29 22 16 15 4 2 3 9 4 5 4 5 6 5 6 5 11 5 11 9
OBJ	457	4 2 0
initial	358	Plow 1: 0 1 19 13 32 31 19 31 33 42 32 31 33 42 32 34 43 34 43 37 45 37 45
Plow0	154	37 49 37 49 37 58 22 58 49 38 36 35 47 44 55 53 52 33 52 54 57 56 55 53 52
Plow1	153	54 57 63 59 58 60 46 49 48 63 59 58 60 61 59 61 62 51 50 51 50 51 60 61 62
Plow2	150	51 60 46 45 41 40 27 26 27 26 27 26 18 17 12 11 5 4 2 0
best	154	Plow 2: 0 2 3 9 10 29 23 16 15 12 11 9 10 29 23 16 15 12 24 16 15 14 22 21
time	173.59	28 29 48 37 36 23 25 30 17 6 7 6 17 12 24 16 15 14 22 29 48 37 36 23 25 30
aveCost	2.69397	25 30 25 30 27 26 18 7 18 7 18 17 30 27 26 24 25 38 36 35 28 21 22 16 15 4
aveDiff	1.93966	2 0
stdevCost	1.99921	
stdevDiff	1.88135	

fileName	hg315	Plow 0: 0 1 19 31 19 1 3 9 10 29 48 63 57 56 55 53 52 33 52 54 57 56 47 44
numPlows	4	55 53 42 32 13 20 13 32 31 33 42 32 31 33 42 43 34 43 34 35 47 48 58 60 46
numNodes	115	49 38 36 35 47 48 37 45 39 40 41 50 46 45 37 58 22 16 15 4 2 0
numArcs	114	Plow 1: 0 1 3 8 13 19 13 8 10 21 20 13 20 34 32 34 43 37 45 41 40 27 26 24
OBJ	115	16 15 14 22 29 23 16 15 14 22 16 15 12 24 25 30 25 38 39 30 39 45 37 36 35
initial	115	36 23 25 30 25 30 17 30 27 26 24 25 38 39 40 27 26 27 26 27 26 18 17 6 7
Plow0	115	18 17 12 11 9 4 2 0
Plow1	787.06	Plow 2: 0 2 3 9 4 5 11 5 4 5 11 9 10 29 22 21 20 34 35 28 21 28 29 48 58 49
Plow2	2.69397	48 37 58 49 37 36 35 36 23 25 30 27 26 18 7 18 7 6 5 6 17 12 24 16 15 12 11
Plow3	1.93966	5 6 5 4 2 0
best	1.99921	Plow 3: 0 2 3 8 10 14 10 14 10 21 22 58 60 61 62 51 60 61 59 58 56 55 53
time	1.88135	52 54 57 63 59 58 56 47 44 43 44 55 53 42 43 37 49 37 49 48 63 59 61 62 51
aveCost		50 51 50 51 60 46 45 41 50 46 49 38 36 35 28 29 23 16 15 4 2 0
aveDiff		
stdevCost		
stdevDiff		

fileName	hg315	Plow 0: 0 1 3 9 10 29 22 58 56 47 48 58 60 46 49 48 63 59 58 60 46 45 41
numPlows	5	40 41 50 46 49 37 49 48 63 57 56 47 44 43 34 32 34 43 34 43 44 55 53 52 54
numNodes	64	57 56 55 53 42 32 13 19 13 32 31 19 1 0
numArcs	116	Plow 1: 0 2 3 8 10 21 20 34 35 47 44 55 53 52 33 42 43 37 45 41 50 46 45
OBJ	461	37 36 35 47 48 37 58 22 16 15 14 10 21 28 29 23 16 15 14 22 16 15 12 24 16
initial	439	15 12 24 16 15 4 2 0
Plow0	94	Plow 2: 0 1 19 31 33 42 43 37 45 37 49 38 36 23 25 30 25 30 17 30 25 30 39
Plow1	96	30 27 26 24 25 30 27 26 24 25 38 39 40 27 26 27 26 18 17 12 11 5 6 17 6 7
Plow2	90	18 17 12 11 5 4 5 4 5 6 5 11 9 4 2 0
Plow3	91	Plow 3: 0 2 3 8 13 20 34 35 28 29 48 37 58 49 37 36 23 25 38 36 35 36 35
Plow4	90	36 35 28 21 22 21 20 13 20 13 8 10 14 10 14 22 29 23 16 15 4 2 0
best	96	Plow 4: 0 1 3 9 10 29 48 58 56 55 53 42 32 31 33 52 54 57 63 59 61 62 51
time	510.534	50 51 60 61 62 51 50 51 60 61 59 58 49 38 39 45 39 40 27 26 27 26 18 7 18
aveCost	2.69397	7 6 5 11 9 4 2 0
aveDiff	1.93966	
stdevCost	1.99921	
stdevDiff	1.88135	

fileName	hg415	Plow 0: 0 2 4 2 3 5 7 6 3 6 23 34 24 21 23 6 4 21 24 22 32 33 24 22 20 21 4
numPlows	2	21 24 33 45 48 55 56 36 9 25 9 12 11 13 11 13 14 15 17 28 16 15 17 15 18
numNodes	82	19 17 31 30 38 41 66 64 63 57 55 57 61 47 61 54 61 79 74 62 60 53 54 53 60
numArcs	148	73 74 79 76 75 76 80 63 80 81 65 75 62 54 48 55 56 36 27 29 30 29 41 40 64
OBJ	553	65 75 62 54 48 46 44 45 44 32 33 39 47 45 48 45 33 39 34 35 36 9 12 16 28
initial	-11	30 28 17 15 18 19 43 51 58 52 70 68 66 67 69 68 70 71 72 78 77 67 69 71 72
Plow0	277	59 58 51 58 51 50 42 43 51 50 49 47 45 44 45 48 45 48 46 53 46 53 46 44 32
Plow1	276	22 32 22 20 1 0
best	277	Plow 1: 0 2 3 5 7 6 4 1 20 21 23 34 35 7 9 8 5 8 10 25 27 29 26 12 16 28 16
time	43.779	28 26 12 11 13 16 28 26 25 23 34 23 34 39 34 39 47 55 37 40 37 39 37 40 37
aveCost	2.61486	39 37 35 36 27 29 41 66 67 65 64 65 67 69 71 70 68 49 68 66 64 63 56 40 27
aveDiff	2.04054	29 26 25 10 11 13 11 13 14 15 18 19 17 31 30 38 41 40 27 25 10 11 13 16 15
stdevCost	1.97629	18 19 43 42 31 42 38 49 56 40 64 63 57 75 57 61 79 74 62 60 73 74 79 76 80
stdevDiff	1.83022	81 65 64 63 56 52 50 49 56 52 70 59 72 78 77 81 77 67 69 68 70 59 58 52 50
		42 38 49 47 55 37 35 7 9 8 10 25 23 34 23 34 24 21 4 1 0

fileName	hg415	Plow 0: 0 1 20 21 23 34 24 22 32 22 32 33 45 33 39 37 39 37 35 36 9 12 16
numPlows	3	15 18 19 43 42 43 51 58 52 70 71 70 68 70 68 70 59 72 78 77 67 65 75 76 80
numNodes	82	81 65 67 69 71 72 78 77 81 77 67 69 68 66 67 69 68 49 68 66 67 69 71 72 59
numArcs	148	58 52 70 59 58 51 50 42 38 49 56 40 64 63 80 63 56 52 50 49 47 45 48 45 44
OBJ	553	45 44 32 22 20 21 23 6 4 2 0
initial	549	Plow 1: 0 2 3 6 3 5 7 9 8 5 8 10 25 9 25 10 11 13 11 13 16 15 18 19 17 31
Plow0	179	42 31 30 38 49 47 55 37 40 64 65 64 63 56 52 50 49 56 40 27 29 26 12 11 13
Plow1	187	11 13 14 15 18 19 17 31 30 28 17 28 30 29 41 40 37 39 47 55 56 36 27 25 23
Plow2	187	34 23 34 39 34 35 7 6 23 34 35 36 27 29 26 25 10 25 23 34 39 47 45 48 45
best	187	48 46 53 46 44 32 33 24 33 39 34 23 34 24 22 20 21 24 21 24 21 4 2 0
time	3316.46	Plow 2: 0 2 3 5 7 9 12 16 28 16 28 26 12 11 13 14 15 17 15 17 15 18 19 43
aveCost	2.61486	51 58 51 50 42 38 41 40 27 29 30 38 41 66 64 63 57 55 56 36 9 8 10 11 13
aveDiff	2.04054	16 28 16 28 26 25 27 29 41 66 64 65 64 63 57 61 47 61 54 61 79 74 79 74 62
stdevCost	1.97629	60 53 60 73 74 62 54 53 54 48 46 53 46 44 45 48 55 57 61 79 76 80 81 65 75
stdevDiff	1.83022	57 75 62 60 73 74 79 76 75 62 54 48 55 37 40 37 35 7 6 4 1 20 21 4 1 0

fileName	hg415	Plow 0: 0 2 3 6 3 5 8 9 12 16 15 18 19 43 51 50 49 68 70 68 66 64 65 75 62
numPlows	4	54 48 46 53 60 73 74 79 74 62 60 73 74 79 74 62 54 61 79 76 80 81 65 64 63
numNodes	82	56 52 50 42 38 49 47 55 37 39 47 45 33 39 37 40 64 63 80 63 56 40 37 39 34
numArcs	148	24 21 4 1 0
OBJ	555	Plow 1: 0 2 3 5 7 9 25 10 11 13 16 28 17 31 30 38 49 56 52 70 68 66 67 69
initial	402	68 70 59 72 78 77 67 69 71 72 59 58 51 58 52 50 49 47 45 48 55 57 61 79 76
Plow0	138	80 81 65 64 65 75 62 60 53 46 44 45 44 45 48 55 37 40 37 35 7 9 12 11 13
Plow1	139	16 28 26 25 23 34 39 37 35 7 6 23 6 4 2 0
Plow2	139	Plow 2: 0 1 20 21 24 22 32 22 32 33 45 48 46 53 46 44 32 33 24 21 23 34 35
Plow3	139	36 9 25 10 8 10 11 13 14 15 18 19 17 31 30 29 30 28 16 28 30 38 41 66 67
best	139	69 71 70 71 72 78 77 81 77 67 65 67 69 68 49 56 36 27 29 41 40 64 63 57 61
time	1680.04	47 61 54 53 54 48 45 48 45 44 32 22 20 21 24 22 20 21 4 2 0
aveCost	2.61486	Plow 3: 0 1 20 21 23 34 23 34 23 34 24 33 39 34 39 47 55 56 40 27 29 26 12
aveDiff	2.04054	16 15 17 15 18 19 17 28 16 28 26 25 27 25 23 34 35 36 27 29 41 40 27 29 26
stdevCost	1.97629	12 11 13 11 13 11 13 14 15 17 15 18 19 43 42 43 51 58 52 70 59 58 51 50 42
stdevDiff	1.83022	31 42 38 41 66 64 63 57 75 76 75 57 55 56 36 9 8 10 8 9 8 5 7 6 4 1 0

fileName	hg415	Plow 0: 0 2 4 21 23 34 39 37 40 64 63 56 52 70 71 72 78 77 81 77 67 69 71
numPlows	5	72 59 72 78 77 67 65 67 69 71 70 59 58 51 58 52 50 49 56 40 27 29 30 38 49
numNodes	82	56 36 9 8 5 7 6 3 2 0
numArcs	148	Plow 1: 0 1 4 21 24 22 32 33 39 34 24 34 35 7 9 8 9 25 10 11 13 16 28 26
OBJ	557	12 16 15 18 19 17 31 42 38 41 66 64 65 75 62 60 73 74 79 76 75 57 61 79 74
initial	488	62 54 48 55 56 40 37 35 36 27 29 26 25 27 29 26 25 23 34 23 6 4 21 4 2 0
Plow0	111	Plow 2: 0 2 3 5 8 10 11 13 14 15 18 19 17 28 16 28 26 12 11 13 16 15 17 15
Plow1	114	17 31 30 29 41 40 37 40 64 63 80 81 65 64 63 56 36 9 12 11 13 11 13 11 13
Plow2	110	14 15 18 19 43 51 58 52 50 49 68 66 67 69 68 66 67 69 68 49 47 45 48 45 48
Plow3	115	46 53 46 53 60 53 46 44 32 22 20 1 0
Plow4	110	Plow 3: 0 1 20 21 23 34 23 34 35 36 27 29 41 66 64 63 57 55 57 75 76 80 81
best	115	65 64 65 75 62 60 73 74 62 54 61 79 74 79 76 80 63 57 61 54 53 54 48 45 44
time	10356.3	45 33 24 34 39 37 39 34 24 33 39 47 61 47 55 37 39 47 45 48 46 44 32 22 20
aveCost	2.61486	21 4 1 0
aveDiff	2.04054	Plow 4: 0 1 20 21 24 22 32 33 45 44 45 48 55 56 52 70 68 70 68 70 59 58 51
stdevCost	1.97629	50 42 43 51 50 42 31 30 28 16 28 17 15 18 19 43 42 38 41 40 27 25 10 8 10
stdevDiff	1.83022	8 9 12 16 28 30 38 49 47 55 37 35 7 9 25 23 6 3 5 7 6 4 1 0

fileName	hg515	Plow 0: 0 2 11 13 22 1 11 12 3 4 15 16 17 18 6 5 4 5 7 8 19 29 31 32 33 56
numPlows	2	55 64 65 56 65 66 75 86 91 84 83 73 63 70 69 48 49 51 49 81 71 70 69 48 49
numNodes	92	47 37 38 28 36 26 25 17 36 37 43 45 44 59 57 58 76 67 61 58 76 67 78 80 81
numArcs	165	71 72 83 85 90 72 90 72 83 73 54 64 65 66 74 73 54 62 51 62 52 39 38 28 36
OBJ	592	26 25 14 12 3 2 11 12 15 16 6 18 27 8 6 18 28 30 28 18 6 5 7 9 19 20 10 9 7
initial	-220	8 27 8 19 20 10 9 19 29 31 21 33 21 20 10 31 21 20 10 31 32 55 64 74 73 63
Plow0	295	72 90 89 88 82 87 79 67 61 58 57 42 23 22 23 24 13 22 1 0
Plow1	297	Plow 1: 0 2 3 4 15 25 17 36 37 43 34 14 34 24 35 42 23 24 35 44 59 61 68
best	297	60 48 46 45 60 48 46 47 46 26 46 45 43 35 44 45 60 59 61 68 78 80 81 88 89
time	273.884	71 70 51 50 38 39 52 53 41 39 41 40 32 33 56 55 53 54 62 63 72 71 81 49 47
aveCost	2.4697	50 52 53 41 40 30 39 52 62 63 70 51 50 52 39 38 50 38 50 47 46 47 50 47 37
aveDiff	1.75152	38 39 30 27 29 40 30 39 30 27 8 6 16 17 18 27 8 27 29 40 32 55 53 54 64 74
stdevCost	1.86043	66 74 66 75 84 83 85 91 84 74 66 75 84 74 66 75 86 91 85 83 85 83 85 91 85
stdevDiff	1.74566	90 89 71 81 88 82 80 69 68 78 87 79 77 76 58 76 77 76 58 76 77 79 77 79 67
		78 87 82 80 69 68 60 59 57 42 35 42 35 43 34 26 34 26 34 24 13 14 13 14 12
		15 25 14 13 11 1 0

fileName	hg515	Plow 0: 0 2 11 12 15 16 6 16 17 18 27 8 27 29 40 32 33 56 55 53 41 40 32
numPlows	3	33 21 33 56 55 53 54 64 74 73 63 72 83 73 54 62 52 39 30 27 8 6 18 27 8 19
numNodes	92	20 10 9 7 9 19 20 10 9 19 29 40 30 28 30 39 41 40 30 27 8 6 18 28 36 37 38
numArcs	165	39 52 53 54 64 74 73 63 70 51 49 81 88 82 87 79 77 76 67 61 68 78 80 81 71
OBJ	593	72 90 89 71 70 51 50 47 46 26 25 17 36 26 46 45 60 59 57 42 35 43 34 26 34
initial	583	24 13 14 34 24 13 22 1 0
Plow0	197	Plow 1: 0 2 11 13 14 13 22 23 24 35 42 35 44 59 61 58 57 58 76 58 76 77 79
Plow1	198	67 61 58 76 58 76 67 78 80 81 49 47 37 43 45 44 45 60 48 49 51 50 52 53 41
Plow2	198	39 38 28 18 6 5 7 8 27 29 31 21 20 10 31 21 20 10 31 32 55 64 65 56 65 66
best	198	75 86 91 85 90 72 90 89 71 81 71 81 88 89 88 82 80 69 48 49 47 50 38 50 38
time	1217.91	50 52 62 63 72 71 70 69 68 60 48 46 47 46 45 43 35 44 59 57 42 23 24 35 42
aveCost	2.4697	23 22 1 0
aveDiff	1.75152	Plow 2: 0 2 3 4 5 4 15 25 14 13 11 1 11 12 3 4 15 25 14 12 15 16 17 18 6 5
stdevCost	1.86043	7 8 19 29 31 32 55 64 65 66 74 66 74 66 75 84 74 66 75 84 74 66 75 86 91
stdevDiff	1.74566	85 91 84 83 85 83 85 91 84 83 85 90 72 83 85 83 73 54 62 51 62 63 70 69 68
		60 59 61 68 78 87 79 77 76 77 79 67 78 87 82 80 69 48 46 47 50 47 37 38 39
		52 39 30 39 38 28 36 26 25 17 36 37 43 34 26 34 14 12 3 2 0

fileName	hg515	Plow 0: 0 2 3 4 15 16 17 36 26 46 45 44 59 61 58 76 58 57 58 76 67 78 87
numPlows	4	82 80 69 68 78 87 79 77 79 67 78 80 81 49 81 88 82 87 79 77 76 77 76 58 76
numNodes	92	67 61 58 76 77 79 67 61 68 78 80 69 48 49 51 62 63 72 83 85 91 84 74 66 75
numArcs	165	84 83 73 63 70 69 48 46 47 37 43 34 14 13 11 2 0
OBJ	594	Plow 1: 0 2 11 12 15 16 17 18 27 29 31 21 20 10 9 19 20 10 9 19 29 40 30
initial	581	27 8 27 8 6 18 6 16 6 18 28 36 37 43 34 24 35 44 59 61 68 60 48 46 47 37 38
Plow0	149	50 52 53 41 40 32 33 21 20 10 31 32 55 64 65 66 75 86 91 84 74 73 63 72 90
Plow1	149	72 83 73 54 62 63 70 51 50 47 46 45 43 45 60 59 57 42 23 22 1 0
Plow2	148	Plow 2: 0 2 11 12 3 4 5 4 15 25 17 18 27 8 19 29 40 32 33 56 55 64 74 73
Plow3	149	54 64 74 66 74 66 74 66 75 84 83 85 90 72 71 81 71 72 90 89 88 89 71 81 88
best	149	82 80 81 71 70 69 68 60 59 57 42 23 24 35 43 35 42 35 42 35 44 45 60 48 49
time	1903.63	47 50 52 62 51 50 38 39 30 39 38 50 47 46 26 25 14 34 26 34 24 13 22 23 24
aveCost	2.4697	13 14 13 22 1 0
aveDiff	1.75152	Plow 3: 0 1 11 1 11 13 14 12 15 25 17 36 37 38 28 30 28 18 6 5 7 8 27 8 6 5
stdevCost	1.86043	7 9 7 8 19 20 10 31 21 33 56 55 53 54 64 65 56 65 66 75 86 91 85 91 85 83
stdevDiff	1.74566	85 83 85 90 89 71 70 51 49 47 50 38 39 30 39 52 53 54 62 52 39 52 39 41 40
		30 27 29 31 32 55 53 41 39 38 28 36 26 34 26 25 14 12 3 2 0

fileName	hg515	Plow 0: 0 2 11 12 15 16 6 5 7 8 6 18 27 8 19 29 40 32 33 21 20 10 9 7 8 19
numPlows	5	29 31 32 55 53 41 40 30 27 8 27 29 31 21 20 10 31 32 55 53 54 64 65 66 75
numNodes	92	84 83 85 90 89 71 70 69 68 60 59 61 67 78 87 79 77 76 58 57 42 23 22 23 22
numArcs	165	1 0
OBJ	596	Plow 1: 0 1 11 12 15 25 17 16 6 5 4 15 25 17 36 26 34 26 46 45 60 48 49 81
initial	589	88 82 87 82 80 81 71 81 88 82 80 81 71 89 90 72 83 85 83 73 54 64 74 66 74
Plow0	120	73 63 72 83 85 83 85 90 89 88 89 90 72 71 81 49 51 50 38 39 52 53 54 62 51
Plow1	118	49 47 37 43 34 14 12 3 2 0
Plow2	121	Plow 2: 0 2 11 12 14 34 24 35 42 35 42 35 43 45 44 59 57 58 61 67 76 58 61
Plow3	121	67 61 68 78 80 69 68 78 87 79 67 61 67 76 77 76 77 79 67 78 80 69 48 46 47
Plow4	119	50 52 53 41 40 30 28 18 28 36 26 34 24 35 44 45 60 59 61 68 60 48 49 47 46
best	121	26 25 14 13 22 1 0
time	12550.1	Plow 3: 0 1 11 12 3 4 15 16 17 18 27 8 27 8 6 18 17 18 17 36 37 38 28 38 39
aveCost	2.4697	52 50 38 39 38 39 41 39 30 27 29 40 32 33 56 65 66 75 84 74 66 74 66 75 86
aveDiff	1.75152	91 84 83 73 54 62 63 72 71 70 69 48 46 47 46 45 43 34 26 25 14 13 11 2 0
stdevCost	1.86043	Plow 4: 0 2 3 4 5 7 9 19 20 10 9 19 20 10 31 21 33 56 55 64 65 56 55 64 74
stdevDiff	1.74566	66 75 86 91 85 91 84 74 73 63 70 51 62 52 50 47 50 52 62 63 70 51 50 47 37
		38 39 38 39 30 28 36 37 43 35 44 59 57 42 23 24 13 22 23 24 13 11 2 0

fileName	hg615	Plow 0: 0 2 4 14 22 31 23 24 5 3 23 24 16 34 36 27 26 19 11 20 11 8 10 28
numPlows	2	19 11 8 7 8 7 8 10 28 29 41 42 55 54 40 42 41 42 41 21 20 21 20 11 20 12 10
numNodes	91	12 10 12 20 21 28 29 38 39 40 29 38 29 40 42 55 56 77 78 90 89 87 86 75 88
numArcs	162	75 73 53 61 53 51 37 27 36 18 36 61 53 51 37 27 36 18 36 61 60 71 83 85 87
OBJ	574	86 62 71 83 72 84 82 81 82 80 69 68 59 80 69 68 57 58 48 46 57 47 45 43 44
initial	-35	46 57 58 50 49 48 32 33 35 34 25 33 49 51 52 63 53 63 52 54 55 56 64 56 77
Plow0	287	65 77 78 90 89 76 88 74 73 53 61 36 18 16 15 6 5 24 5 3 2 3 23 14 22 14 22
Plow1	287	30 13 14 13 14 13 1 4 14 22 14 22 31 23 14 22 30 13 1 0
best	287	Plow 1: 0 2 4 15 17 9 11 9 7 6 15 17 9 17 18 26 19 38 29 41 21 28 19 38 27
time	474.85	38 39 37 35 60 71 70 72 80 82 84 85 87 89 87 89 76 78 76 65 77 65 74 73 62
aveCost	2.46296	61 60 50 70 59 58 50 49 51 52 54 64 63 64 65 74 63 52 63 52 39 40 54 64 65
aveDiff	1.85185	76 88 74 63 52 39 37 35 34 25 33 49 48 32 44 32 31 25 31 32 31 32 44 22 44
stdevCost	1.84136	46 45 47 66 67 69 81 79 67 69 81 79 67 66 68 57 47 66 68 59 58 48 46 45 43
stdevDiff	1.8061	30 43 30 43 44 22 44 32 33 35 60 50 70 72 84 85 83 86 62 71 70 59 80 72 83
		86 75 88 75 73 62 61 36 18 26 9 17 18 16 34 36 27 26 9 7 6 5 24 25 24 16 15
		4 15 4 1 0

fileName	hg615	Plow 0: 0 1 4 15 6 5 24 5 24 5 3 23 14 22 44 46 45 47 66 67 69 68 59 80 82
numPlows	3	81 79 67 66 68 57 58 50 70 72 80 69 81 82 84 85 87 86 62 61 36 61 60 71 83
numNodes	91	72 84 85 83 85 87 89 87 89 76 88 74 73 53 63 64 65 74 63 52 39 40 54 55 56
numArcs	162	77 78 90 89 76 88 74 73 62 61 53 51 52 39 37 27 26 9 17 18 36 18 16 15 17
OBJ	576	9 7 8 10 12 10 28 19 11 20 21 20 11 20 11 9 17 9 7 6 15 4 14 22 30 13 14 22
initial	563	31 32 31 32 44 46 45 43 30 13 1 0
Plow0	195	Plow 1: 0 2 4 15 4 14 13 14 22 31 25 33 35 60 50 49 48 32 33 35 34 25 33
Plow1	195	49 51 37 27 26 9 11 8 10 12 20 21 28 29 41 42 55 54 40 29 38 27 36 27 36
Plow2	186	18 26 19 38 29 38 39 37 35 60 50 70 59 80 69 81 79 67 69 68 59 58 50 49 51
best	195	52 63 52 63 53 61 36 61 53 51 37 35 34 25 24 16 34 36 18 16 34 36 18 36 27
time	975.54	38 29 40 42 41 21 20 12 10 28 19 11 8 7 8 7 6 5 3 2 4 1 0
aveCost	2.46296	Plow 2: 0 2 3 23 24 25 31 23 24 16 15 17 18 26 19 38 39 40 42 41 21 28 29
aveDiff	1.85185	41 42 55 56 64 56 77 65 76 65 74 63 52 54 64 63 52 54 64 65 77 65 77 78 76
stdevCost	1.84136	78 90 89 87 86 62 71 70 72 84 82 80 72 83 86 75 73 62 71 83 86 75 88 75 88
stdevDiff	1.8061	75 73 53 61 60 71 70 59 58 48 46 57 58 48 32 33 49 48 46 57 47 66 68 57 47
		45 43 44 22 14 22 44 32 44 32 31 23 14 22 30 43 30 43 44 22 14 13 1 0

fileName	hg615	Plow 0: 0 2 4 14 22 44 32 33 35 60 71 83 85 87 89 76 88 75 88 74 63 52 63
numPlows	4	64 65 77 65 77 65 76 65 74 73 62 61 53 51 52 54 55 54 64 63 52 39 40 42 55
numNodes	91	56 77 78 90 89 87 86 62 71 83 86 75 88 75 73 62 71 70 59 80 82 84 85 87 86
numArcs	162	62 61 36 61 60 71 70 72 84 85 83 86 75 73 53 63 52 39 37 27 26 9 17 18 16
OBJ	578	15 4 1 0
initial	499	Plow 1: 0 1 4 14 22 30 43 44 32 33 35 34 25 33 49 51 37 27 38 27 36 61 36
Plow0	145	27 26 19 38 29 41 21 20 11 8 10 28 19 11 9 11 20 11 20 12 10 12 20 21 28
Plow1	145	29 40 54 64 65 74 63 52 54 40 29 38 39 40 42 41 42 55 56 64 56 77 78 76 78
Plow2	145	90 89 87 89 76 88 74 73 53 51 52 63 53 61 53 61 60 50 49 48 46 57 47 45 43
Plow3	145	30 13 14 13 1 0
best	145	Plow 2: 0 2 3 23 24 16 34 36 27 36 18 36 18 36 18 16 34 36 18 26 19 38 29
time	1516.12	41 42 41 21 20 21 28 29 38 39 37 35 34 25 33 49 51 37 35 60 50 49 48 46 57
aveCost	2.46296	58 50 70 59 80 72 83 72 80 69 81 79 67 69 68 59 58 50 70 72 84 82 80 69 81
aveDiff	1.85185	82 81 79 67 69 68 59 58 48 32 31 32 31 23 14 22 30 43 30 13 1 0
stdevCost	1.84136	Plow 3: 0 1 13 14 22 31 32 44 22 14 22 44 46 45 47 66 68 57 47 66 67 66 68
stdevDiff	1.8061	57 58 48 32 44 46 45 43 44 22 31 25 24 25 31 23 24 5 24 16 15 17 9 17 9 7
		8 10 12 10 28 19 11 8 7 8 7 6 15 17 18 26 9 7 6 5 24 5 3 2 4 15 4 15 6 5 3
		23 14 22 14 13 1 0

fileName	hg615	Plow 0: 0 1 13 14 22 31 32 33 35 60 71 83 86 62 71 83 86 75 88 74 63 52 39
numPlows	5	40 54 40 29 41 42 55 56 64 65 74 73 62 61 53 63 52 63 52 63 64 63 52 54 64
numNodes	91	56 77 65 77 65 74 63 53 61 60 50 49 48 32 44 32 31 25 31 23 14 22 31 23 24
numArcs	162	16 15 17 18 36 18 16 15 4 1 0
OBJ	580	Plow 1: 0 2 3 23 24 5 24 25 33 49 51 52 54 55 56 77 78 90 89 87 86 62 61
initial	508	36 27 26 19 11 8 10 12 10 12 20 21 28 29 38 39 37 35 34 25 33 35 60 50 70
Plow0	120	59 80 82 84 85 87 89 87 89 76 88 75 73 53 51 37 27 36 61 36 27 26 9 17 9 7
Plow1	117	6 5 3 23 14 22 30 13 1 0
Plow2	114	Plow 2: 0 1 4 15 6 15 17 9 7 8 7 8 10 28 19 38 39 37 27 38 29 40 42 55 54
Plow3	118	64 65 77 78 76 65 76 78 90 89 76 88 75 73 62 71 70 72 84 85 83 72 84 82 81
Plow4	115	79 67 69 68 57 58 48 32 31 32 44 22 14 22 44 22 14 13 1 0
best	120	Plow 3: 0 1 13 14 22 30 43 30 43 44 32 33 49 48 46 45 47 66 68 57 58 50 70
time	2689.06	72 80 69 68 59 80 69 81 79 67 66 67 69 81 82 80 72 83 85 87 86 75 88 74 73
aveCost	2.46296	53 61 53 51 52 39 40 42 41 21 20 12 10 28 19 11 9 11 20 21 28 29 38 27 36
aveDiff	1.85185	61 60 71 70 59 58 48 46 57 47 45 43 30 13 1 0
stdevCost	1.84136	Plow 4: 0 2 4 14 22 44 46 45 43 44 46 57 47 66 68 59 58 50 49 51 37 35 34
stdevDiff	1.8061	36 18 36 18 16 34 25 24 16 34 36 18 26 9 17 18 26 19 38 29 41 42 41 21 20
		11 20 11 8 7 6 5 24 5 3 2 4 15 4 14 13 1 0

fileName	hg715	Plow 0: 0 1 4 26 25 2 25 27 28 38 37 27 28 38 37 27 37 49 51 64 74 86 74
numPlows	2	86 74 64 66 75 68 55 67 43 42 40 29 28 26 25 27 37 49 50 52 40 38 50 52 40
numNodes	97	38 50 53 50 49 50 49 51 53 51 64 66 75 88 76 66 53 65 55 54 56 57 67 77 68
numArcs	174	55 67 77 68 78 88 89 87 86 74 76 87 86 74 76 66 53 65 55 54 52 65 75 68 78
OBJ	557	79 77 79 95 91 80 69 77 79 80 91 92 82 83 84 71 63 61 57 69 81 70 71 73 85
initial	-102	94 85 94 85 84 71 73 85 84 96 94 96 93 83 70 61 63 61 59 58 56 58 48 44 34
Plow0	278	21 23 35 34 44 45 46 36 35 45 46 62 60 59 44 34 44 34 33 31 14 12 9 12 9
Plow1	279	12 9 41 39 19 18 29 39 42 40 29 28 26 18 6 4 2 0
best	279	Plow 1: 0 1 4 2 4 2 4 26 18 29 39 42 54 56 57 67 43 41 32 33 48 47 43 41 32
time	544.473	30 12 14 20 17 22 16 15 17 23 24 22 24 36 35 34 33 31 14 20 21 23 24 36 46
aveCost	2.27586	62 72 63 61 59 58 56 47 32 33 48 47 32 30 19 7 6 3 5 8 5 8 5 7 9 41 39 19 7
aveDiff	1.82759	6 3 1 3 5 7 9 8 10 12 10 12 9 8 10 11 13 20 17 23 35 45 60 63 61 63 60 62 72
stdevCost	1.78396	73 72 63 61 70 83 93 92 82 83 84 96 93 92 82 81 58 56 47 43 42 54 52 65 75
stdevDiff	1.7889	88 76 87 89 90 78 88 89 87 89 90 95 91 92 82 81 70 71 63 61 57 69 77 79 77
		79 95 90 78 79 80 82 80 69 81 58 56 58 48 44 45 60 59 44 34 21 31 30 12 10
		11 14 12 14 11 13 11 13 15 17 22 16 15 13 11 13 20 21 31 30 19 18 6 4 2 0

fileName	hg715	Plow 0: 0 1 3 5 7 9 8 10 12 10 11 14 12 10 12 14 11 13 11 13 20 21 31 30
numPlows	3	12 9 8 10 11 13 11 13 20 17 23 35 34 33 48 47 43 42 54 56 47 32 33 48 44
numNodes	97	34 44 45 60 62 72 63 61 57 69 77 68 78 88 76 87 86 74 86 74 76 87 86 74 76
numArcs	174	66 53 50 49 50 49 51 64 74 86 74 64 66 53 65 75 88 89 87 89 90 78 88 89 87
OBJ	557	89 90 95 91 92 82 81 70 71 73 72 63 61 63 61 59 58 48 47 43 41 39 19 18 29
initial	546	28 26 18 6 3 1 0
Plow0	186	Plow 1: 0 1 4 26 18 6 3 5 8 5 8 5 7 9 41 32 30 12 14 20 17 22 16 15 13 15
Plow1	185	17 22 24 22 16 15 17 23 24 36 35 34 33 31 14 20 21 31 14 12 9 12 9 12 9 41
Plow2	186	32 33 31 30 19 18 29 28 38 50 53 65 55 54 52 65 75 88 76 66 75 68 55 67 43
best	186	42 54 52 40 38 37 27 28 38 50 52 40 38 37 27 37 49 50 52 65 55 67 77 68 55
time	194.325	54 56 47 32 30 19 7 6 4 2 4 2 25 27 28 26 25 2 0
aveCost	2.27586	Plow 2: 0 2 4 26 25 27 37 49 51 53 51 64 66 75 68 78 79 80 69 81 58 56 58
aveDiff	1.82759	56 57 69 77 79 77 79 80 69 81 58 56 57 67 77 79 77 79 95 90 78 79 95 91 80
stdevCost	1.78396	91 92 82 80 82 81 70 83 70 61 59 58 56 58 48 44 45 60 59 44 34 44 34 21 23
stdevDiff	1.7889	35 45 46 36 46 62 60 59 44 34 21 23 24 36 35 45 46 62 72 73 85 94 85 84 96
		93 83 84 96 93 92 82 83 93 92 82 83 84 71 73 85 94 96 94 85 84 71 63 60 63
		61 63 61 70 71 63 61 57 67 43 41 39 42 40 29 39 42 40 29 39 19 7 6 4 1 0

fileName	hg715	Plow 0: 0 2 25 27 37 49 51 64 74 86 74 64 66 75 88 76 66 53 50 52 65 75 88
numPlows	4	89 90 78 88 89 87 86 74 86 74 76 66 75 68 55 67 77 68 78 79 77 79 77 79 80
numNodes	97	91 80 69 81 58 56 58 56 58 48 44 34 44 45 60 59 58 56 47 43 42 40 29 39 42
numArcs	174	54 56 57 69 81 70 71 63 61 63 61 59 44 45 46 62 72 63 61 57 67 43 41 32 33
OBJ	557	31 30 19 18 6 4 1 0
initial	352	Plow 1: 0 1 3 5 7 9 8 5 8 5 8 10 11 13 20 21 23 24 36 35 34 33 48 47 43 42
Plow0	139	40 38 37 27 28 38 50 49 50 53 51 64 66 53 65 55 67 77 79 95 90 78 79 95 91
Plow1	139	92 82 81 70 61 70 71 73 85 84 96 94 85 94 96 93 83 93 92 82 80 69 77 68 55
Plow2	141	54 52 40 38 37 27 28 38 50 52 65 55 54 52 40 29 28 26 18 29 28 26 25 2 4 1
Plow3	140	0
best	141	Plow 2: 0 2 4 26 18 6 3 5 7 9 41 39 19 18 29 39 42 54 56 57 67 43 41 32 33
time	4922.39	48 47 32 30 12 9 8 10 11 13 15 17 22 24 22 16 15 13 11 13 11 13 20 17 23
aveCost	2.27586	35 34 33 31 14 12 10 12 9 12 9 12 10 12 14 11 14 20 17 22 16 15 17 23 24
aveDiff	1.82759	36 46 36 35 45 46 62 60 59 58 48 44 34 21 31 14 20 21 31 30 19 7 6 4 1 0
stdevCost	1.78396	Plow 3: 0 1 4 26 25 27 37 49 50 49 51 53 65 75 68 78 88 76 87 89 87 86 74
stdevDiff	1.7889	76 87 89 90 95 91 92 82 83 84 71 63 60 62 72 73 85 94 85 84 96 93 92 82 83
		70 83 84 71 73 72 63 61 63 61 59 44 34 44 34 21 23 35 45 60 63 61 57 69 77
		79 80 82 81 58 56 47 32 30 12 14 12 9 41 39 19 7 6 3 1 0

fileName	hg715	Plow 0: 0 2 25 27 37 27 37 27 28 38 50 49 51 53 51 64 66 75 68 78 79 77 79
numPlows	5	77 79 95 91 80 69 81 58 56 47 43 42 54 56 57 69 77 68 78 88 89 90 78 88 89
numNodes	97	90 95 91 92 82 83 84 96 93 92 82 80 69 81 58 48 44 45 46 36 35 34 21 31 30
numArcs	174	12 9 41 39 19 18 6 3 1 0
OBJ	559	Plow 1: 0 2 4 26 18 29 28 26 18 29 39 42 54 52 65 55 67 77 79 95 90 78 79
initial	549	80 82 81 70 71 73 85 94 85 94 85 84 71 63 61 57 67 77 68 55 54 56 47 32 33
Plow0	113	48 47 32 30 19 7 9 12 10 12 9 8 5 7 9 41 32 33 31 30 12 9 12 10 12 9 8 5 7
Plow1	113	6 4 1 0
Plow2	111	Plow 2: 0 1 3 5 8 10 11 13 11 14 12 14 12 14 20 21 23 24 36 35 45 60 59 44
Plow3	113	34 44 34 44 34 33 31 14 11 13 20 21 31 14 20 17 22 16 15 17 22 16 15 13 11
Plow4	111	13 20 17 23 24 22 24 36 46 62 72 63 60 59 44 45 46 62 60 63 61 59 58 56 58
best	113	56 58 48 47 43 41 39 42 40 29 28 26 25 2 0
time	955.051	Plow 3: 0 2 4 26 25 27 28 38 50 52 65 75 88 76 66 53 65 75 88 76 87 89 87
aveCost	2.27586	86 74 76 87 89 87 86 74 86 74 86 74 64 74 76 66 53 50 52 40 38 37 49 50 49
aveDiff	1.82759	51 64 66 75 68 55 54 52 40 29 39 19 18 6 3 1 0
stdevCost	1.78396	Plow 4: 0 1 3 5 8 10 11 13 15 17 23 35 34 33 48 44 34 21 23 35 45 60 62 72
stdevDiff	1.7889	73 85 84 96 94 96 93 92 82 83 70 83 93 83 84 71 63 61 63 61 59 58 56 57 69
		77 79 80 91 92 82 81 70 61 63 61 70 71 73 72 63 61 57 67 43 42 40 38 37 49
		50 53 65 55 67 43 41 32 30 19 7 6 4 1 0

fileName	hg815	Plow 0: 0 2 3 2 3 4 5 6 8 10 9 11 23 10 24 22 24 22 33 41 42 35 24 22 24 22
numPlows	2	33 41 42 35 33 39 33 39 40 47 54 53 55 69 51 53 55 71 72 79 80 91 92 91 80
numNodes	100	82 83 99 83 74 84 93 94 96 97 86 94 85 75 86 94 85 84 93 94 96 95 96 95 93
numArcs	180	83 82 81 79 80 91 90 98 88 77 78 98 88 77 78 89 78 71 55 57 55 57 55 69 51
OBJ	572	53 46 31 32 39 33 39 40 41 59 56 54 53 46 47 54 57 58 56 40 47 32 39 21 6
initial	-84	19 20 21 22 8 10 9 7 5 6 19 16 3 2 15 17 30 29 45 29 28 17 30 29 45 50 51
Plow0	286	38 46 47 32 20 18 31 30 38 46 31 30 38 45 50 51 38 45 29 28 17 18 31 32 20
Plow1	286	18 16 15 2 15 1 28 1 0
best	286	Plow 1: 0 2 15 17 18 16 3 4 5 4 19 20 21 6 8 7 9 11 23 10 24 34 43 35 24 34
time	548.658	43 44 42 61 59 56 40 41 59 60 81 74 84 63 62 61 59 60 59 60 73 60 81 74 73
aveCost	2.25	60 73 63 62 61 73 63 66 64 48 36 43 44 62 64 65 49 48 36 26 27 37 36 26 34
aveDiff	1.68889	23 25 12 11 12 25 14 13 14 13 12 13 12 13 14 27 26 34 23 25 26 25 14 27 37
stdevCost	1.72683	49 48 44 62 64 65 67 66 85 84 63 66 64 48 44 42 61 73 60 59 60 58 79 58 79
stdevDiff	1.72977	58 57 72 89 90 98 78 71 70 68 76 77 76 87 88 77 70 69 52 68 70 69 52 50 52
		50 52 68 76 87 88 77 70 71 72 89 80 82 92 99 95 93 83 82 83 74 73 60 58 56
		54 57 72 79 81 82 92 91 80 91 90 89 80 91 92 99 95 96 97 86 75 67 66 85 84
		85 84 85 75 67 65 49 37 36 43 35 33 39 21 22 8 7 5 4 19 16 15 2 15 1 0

fileName	hg815	Plow 0: 0 1 28 17 30 38 45 29 28 17 30 29 45 29 45 50 52 50 51 38 46 47 54
numPlows	3	53 55 57 58 56 54 57 55 57 72 79 80 91 92 91 80 91 80 91 90 89 78 71 70 69
numNodes	100	52 50 52 68 76 87 88 77 70 71 55 69 52 68 70 68 76 87 88 77 78 71 72 89 80
numArcs	180	91 92 99 83 74 84 63 66 85 84 93 94 85 84 85 75 67 65 49 37 36 26 27 37 49
OBJ	574	48 44 42 61 59 56 40 47 32 39 40 41 59 56 40 47 54 53 46 31 30 29 28 1 15
initial	570	1 0
Plow0	193	Plow 1: 0 2 15 17 18 31 32 20 21 22 33 41 42 61 73 60 81 74 73 60 59 60 73
Plow1	192	60 58 56 54 57 55 69 51 53 46 31 32 20 18 16 3 4 5 4 5 6 8 10 9 7 9 11 12 13
Plow2	189	12 13 12 11 23 25 12 25 14 27 26 25 14 13 14 13 14 27 37 36 43 44 62 61 59
best	193	60 59 60 73 60 81 79 81 82 83 82 92 99 95 93 83 74 84 85 84 93 83 82 92 91
time	832.951	90 98 88 77 78 98 78 89 80 82 81 74 73 63 62 64 65 67 66 64 48 36 43 44 62
aveCost	2.25	61 73 63 62 64 65 49 48 36 26 34 43 35 24 22 33 39 21 6 8 7 5 4 19 16 15 2
aveDiff	1.68889	3 2 0
stdevCost	1.72683	Plow 2: 0 2 3 2 15 17 18 16 3 4 19 20 18 31 30 38 45 50 51 38 46 47 32 39
stdevDiff	1.72977	33 39 33 39 21 22 8 10 24 34 23 10 9 11 23 25 26 34 43 35 33 41 59 60 58
		79 58 57 72 89 90 98 88 77 76 77 70 69 51 53 55 71 72 79 58 79 80 82 83 99
		95 93 94 96 97 86 75 86 94 96 95 96 95 96 97 86 94 85 75 67 66 85 84 63 66
		64 48 44 42 35 33 39 40 41 42 35 24 22 24 22 24 34 23 10 24 22 8 7 5 6 19
		20 21 6 19 16 15 2 15 1 0

fileName	hg815	Plow 0: 0 1 28 17 30 29 45 50 52 68 76 87 88 77 78 89 90 98 88 77 78 71 72
numPlows	4	79 80 91 92 99 83 74 84 85 84 85 84 63 62 64 65 67 66 85 75 67 66 64 48 44
numNodes	100	62 61 73 63 66 64 48 36 43 44 42 61 73 60 59 56 40 47 32 20 21 6 8 10 9 11
numArcs	180	23 25 12 13 12 11 12 13 14 13 12 25 14 27 37 49 37 36 43 35 33 39 33 39 21
OBJ	574	6 19 16 15 17 18 16 3 2 0
initial	568	Plow 1: 0 1 15 17 18 31 32 39 33 39 21 22 33 41 59 56 58 79 80 82 92 91 92
Plow0	144	99 95 96 97 86 75 67 65 49 48 44 42 35 33 39 40 41 42 61 59 56 54 57 55 69
Plow1	145	51 53 55 57 55 69 52 68 70 68 76 77 70 71 55 57 72 89 80 91 90 89 80 91 80
Plow2	143	82 83 74 84 63 62 64 65 49 48 36 26 34 23 10 24 22 8 10 9 7 5 6 19 16 15 2
Plow3	142	15 1 0
best	145	Plow 2: 0 2 3 4 19 20 18 31 30 38 45 29 45 29 28 17 30 38 45 50 52 50 51
time	2326.49	38 46 31 32 39 40 41 59 60 73 60 58 56 40 47 54 57 58 79 81 74 73 63 66 85
aveCost	2.25	75 86 94 85 84 93 94 96 97 86 94 96 95 96 95 93 94 85 84 93 83 82 92 91 80
aveDiff	1.68889	91 90 98 78 98 88 77 76 87 88 77 70 69 51 38 46 31 30 29 28 1 0
stdevCost	1.72683	Plow 3: 0 2 15 2 3 4 5 6 8 7 9 11 23 25 14 13 14 27 37 36 26 27 26 25 26 34
stdevDiff	1.72977	23 10 24 22 8 7 5 4 5 4 19 20 21 22 24 34 43 35 24 22 24 22 33 41 42 35 24
		34 43 44 62 61 59 56 58 56 54 53 46 47 54 53 55 71 72 79 81 82 83 99 95 93
		83 82 81 74 73 60 81 60 59 60 73 60 58 57 72 89 78 71 70 69 52 50 51 53 46
		47 32 20 18 16 3 2 0

fileName	hg815	Plow 0: 0 2 15 17 30 38 45 29 45 50 52 68 76 87 88 77 70 71 55 57 55 57 58
numPlows	5	79 80 91 80 82 83 82 81 74 84 85 75 86 94 96 97 86 94 85 84 85 75 67 66 85
numNodes	100	84 93 94 85 84 93 83 74 73 60 73 60 59 60 58 56 54 53 55 69 51 38 46 31 30
numArcs	180	29 45 29 28 1 0
OBJ	576	Plow 1: 0 1 28 17 18 31 32 39 33 39 40 47 54 53 55 69 52 68 70 68 76 77 78
initial	540	89 80 82 83 74 84 63 66 85 84 63 62 64 65 49 48 44 42 35 24 34 43 35 33 41
Plow0	117	42 35 24 22 8 10 24 22 33 39 21 22 24 34 23 10 24 22 8 10 9 7 5 4 19 20 18
Plow1	116	31 30 29 28 17 18 16 15 17 28 1 0
Plow2	115	Plow 2: 0 1 15 2 3 4 5 6 8 7 9 11 12 13 12 11 23 10 9 11 23 25 14 13 14 27
Plow3	114	37 49 37 36 43 44 42 61 73 60 81 82 92 99 95 96 95 93 94 96 95 93 83 82 92
Plow4	114	91 90 98 78 71 72 89 90 89 78 71 70 69 52 50 52 50 51 53 46 47 32 20 21 6
best	117	19 16 3 2 0
time	11548.9	Plow 3: 0 1 15 17 30 38 46 47 54 57 72 89 80 91 92 91 80 91 92 99 83 99 95
aveCost	2.25	96 97 86 75 67 65 49 48 36 26 27 26 34 23 25 14 13 14 27 37 36 43 44 62 61
aveDiff	1.68889	73 63 62 64 65 67 66 64 48 44 62 61 59 56 58 57 55 71 72 79 81 60 58 56 40
stdevCost	1.72683	41 59 56 40 47 32 20 18 16 15 2 0
stdevDiff	1.72977	Plow 4: 0 2 3 4 5 6 8 7 5 4 19 20 21 22 24 22 33 39 33 39 40 41 59 56 54 57
		72 79 81 74 73 60 59 60 73 63 66 64 48 36 26 25 12 13 12 25 26 34 43 35 33
		41 42 61 59 56 58 79 80 91 90 98 88 77 76 87 88 77 78 98 88 77 70 69 51 38
		45 50 51 53 46 31 32 39 21 6 19 16 3 2 0

fileName	hg915	Plow 0: 0 1 16 17 27 36 47 48 47 48 50 72 74 66 86 75 86 88 87 75 73 85 87
numPlows	2	75 73 72 73 72 73 85 87 88 89 88 86 76 89 88 89 90 91 90 91 95 96 95 81 78
numNodes	97	58 56 57 60 80 71 83 84 94 93 94 84 94 84 82 92 81 78 80 71 83 63 83 84 82
numArcs	175	71 62 60 80 92 81 78 80 92 96 93 96 93 82 92 96 93 96 93 82 71 62 60 58 56
OBJ	588	42 31 23 11 9 7 19 21 11 13 15 12 15 12 10 9 10 13 24 23 31 30 22 21 22 21
initial	-252	22 20 28 29 39 40 42 56 57 60 58 55 67 53 68 76 89 90 77 79 91 95 81 78 70
Plow0	294	79 81 78 70 79 91 90 77 69 70 55 54 30 39 40 42 31 41 43 57 59 45 43 42 56
Plow1	294	54 52 38 37 36 47 34 16 17 18 35 29 39 29 39 20 28 8 6 3 5 3 2 4 6 3 2 4 2 0
best	294	Plow 1: 0 1 16 34 27 17 18 35 27 17 4 6 18 28 8 6 18 28 29 38 37 35 29 39
time	52.484	29 38 37 51 65 53 68 69 70 55 54 30 31 41 43 57 59 45 33 41 33 41 24 32 33
aveCost	2.31714	41 24 23 22 30 39 20 19 21 11 23 22 20 19 8 5 7 19 8 5 7 9 10 9 7 9 11 9 11
aveDiff	1.82286	13 24 32 15 25 14 12 15 25 26 32 15 25 14 12 10 13 15 25 26 44 46 61 46 45
stdevCost	1.72693	33 44 26 32 33 41 33 44 46 61 59 62 63 61 59 62 63 61 46 45 43 42 40 54 56
stdevDiff	1.67248	42 40 38 37 51 65 53 65 64 49 51 52 53 68 69 67 40 54 52 53 68 76 77 69 67
		53 65 66 68 53 65 66 68 53 65 66 65 64 50 72 74 75 74 64 74 66 65 66 86 76
		77 79 81 78 58 55 67 40 38 37 36 49 48 50 64 50 64 49 48 47 34 27 36 49 51
		52 38 37 35 27 17 27 17 4 1 4 2 0

fileName	hg915	Plow 0: 0 1 16 34 16 17 4 6 18 28 29 39 40 42 56 54 56 57 59 45 33 41 43
numPlows	3	42 40 54 52 38 37 36 49 51 65 66 86 88 89 88 89 90 91 95 81 78 70 55 67 53
numNodes	97	68 76 89 90 77 69 70 79 81 78 80 71 62 63 83 84 94 84 94 84 82 92 81 78 58
numArcs	175	55 54 52 53 68 53 65 53 65 66 68 53 68 69 70 79 81 78 58 55 54 30 22 20 19
OBJ	589	21 22 20 28 8 5 7 19 21 22 21 11 23 31 30 31 41 33 41 33 41 24 23 11 13 24
initial	579	23 22 21 11 9 7 19 8 6 3 5 3 2 4 1 0
Plow0	196	Plow 1: 0 2 4 6 18 35 27 17 27 17 27 17 18 28 8 5 7 9 10 9 11 9 7 9 10 9 11
Plow1	196	13 24 32 15 25 26 32 33 44 46 45 43 57 60 58 56 57 59 45 33 44 26 44 46 61
Plow2	197	46 61 59 62 63 61 59 62 60 80 92 96 93 94 93 96 95 96 93 96 93 82 92 81 78
best	197	70 55 67 40 38 37 35 29 39 29 39 20 28 29 38 37 51 52 38 37 51 65 66 65 64
time	391.415	50 72 74 64 50 64 49 48 47 48 47 48 50 64 49 51 52 53 65 66 68 76 77 79 91
aveCost	2.31714	90 91 90 77 69 67 53 68 69 67 40 38 37 35 29 39 20 19 8 6 3 2 0
aveDiff	1.82286	Plow 2: 0 1 16 17 18 35 27 36 49 48 50 72 73 85 87 88 87 75 73 72 73 85 87
stdevCost	1.72693	75 73 72 74 66 65 53 65 64 74 75 74 66 86 75 86 76 89 88 86 76 77 79 91 95
stdevDiff	1.67248	81 78 80 92 96 93 82 71 83 63 61 46 45 43 42 40 54 30 39 40 42 56 42 31 41
		24 32 15 12 15 25 14 12 10 13 15 25 14 12 10 13 15 12 15 25 26 32 33 41 43
		57 60 80 71 83 84 82 71 62 60 58 56 42 31 23 22 30 39 29 38 37 36 47 34 27
		36 47 34 27 17 4 1 0

fileName	hg915	Plow 0: 0 2 4 6 18 35 29 38 37 36 47 48 50 64 74 75 86 88 89 90 91 95 81
numPlows	4	78 58 55 67 53 68 53 65 53 65 64 50 72 73 72 74 64 49 51 65 66 86 76 89 90
numNodes	97	91 95 81 78 58 55 67 40 42 31 23 31 41 43 42 56 57 60 58 56 54 30 31 41 33
numArcs	175	44 46 61 59 45 33 41 33 41 43 57 59 62 60 58 56 42 40 38 37 35 27 36 47 48
OBJ	590	47 34 16 17 4 1 0
initial	456	Plow 1: 0 2 4 6 3 5 7 19 21 22 20 19 21 11 23 22 30 39 40 42 40 54 52 53 68
Plow0	149	76 89 88 89 88 86 75 73 85 87 75 74 66 68 76 77 69 70 79 81 78 80 71 62 60
Plow1	147	80 92 81 78 80 71 83 63 83 84 94 84 94 84 82 92 96 93 96 93 96 95 96 93 94
Plow2	147	93 82 92 81 78 70 55 54 52 53 68 69 67 40 54 56 42 31 30 39 20 19 8 5 3 2 0
Plow3	147	Plow 2: 0 1 16 17 27 17 18 35 29 39 20 28 8 5 7 9 11 13 15 25 26 44 26 32
best	149	15 25 14 12 15 25 14 12 10 13 24 23 22 21 11 9 10 13 24 32 15 25 26 32 33
time	3206.1	44 46 61 59 45 43 57 60 80 92 96 93 82 71 83 84 82 71 62 63 61 46 45 43 42
aveCost	2.31714	56 57 59 62 63 61 46 45 33 41 24 32 33 41 24 23 11 9 10 9 7 9 11 13 15 12
aveDiff	1.82286	15 12 10 9 7 19 8 6 3 2 0
stdevCost	1.72693	Plow 3: 0 1 16 34 27 36 49 51 52 38 37 51 65 53 65 66 68 53 65 66 65 66 86
stdevDiff	1.67248	76 77 79 91 90 77 79 81 78 70 79 91 90 77 69 67 53 68 69 70 55 54 30 22 21
		22 20 28 29 39 29 39 29 38 37 51 52 38 37 35 27 17 18 28 8 6 18 28 29 39
		40 38 37 36 49 48 50 64 50 72 73 85 87 88 87 75 73 72 74 66 65 64 49 48 47
		34 27 17 27 17 4 1 0

fileName	hg915	Plow 0: 0 1 16 17 18 28 8 5 7 9 10 9 10 9 7 19 21 11 13 24 23 31 41 24 32
numPlows	5	33 44 46 61 59 45 33 44 26 32 15 25 26 32 33 41 33 41 33 41 43 57 60 58 55
numNodes	97	67 40 38 37 51 65 53 65 53 65 66 65 66 86 88 89 88 89 88 87 75 86 76 77 69
numArcs	175	67 40 54 52 38 37 51 52 38 37 35 27 17 27 34 16 1 0
OBJ	592	Plow 1: 0 1 16 17 27 36 47 48 50 64 50 64 50 72 73 85 87 75 73 85 87 88 86
initial	570	75 74 66 65 64 49 51 52 53 68 76 77 79 81 78 70 79 81 78 58 56 42 31 23 11
Plow0	119	23 22 21 22 21 11 13 15 25 26 44 46 45 33 41 24 32 15 25 14 12 15 12 10 13
Plow1	120	15 25 14 12 15 12 10 13 24 23 22 20 19 8 6 3 2 0
Plow2	118	Plow 2: 0 2 4 6 18 28 29 39 40 42 56 54 52 53 68 69 70 79 91 90 91 95 81
Plow3	118	78 80 71 83 84 82 71 83 84 94 93 96 93 96 93 82 92 81 78 70 55 54 30 22 20
Plow4	118	28 29 39 29 38 37 35 29 39 20 28 8 5 3 5 7 9 11 9 11 9 7 19 8 6 3 2 0
best	120	Plow 3: 0 2 4 17 18 35 29 39 20 19 21 22 30 31 41 43 42 56 57 59 62 60 80
time	2032.66	71 62 60 80 92 96 93 94 84 94 84 82 71 62 63 83 63 61 59 62 63 61 46 61 46
aveCost	2.31714	45 43 42 40 54 56 57 59 45 43 57 60 58 56 42 40 42 31 30 39 29 38 37 36 49
aveDiff	1.82286	48 47 34 16 1 0
stdevCost	1.72693	Plow 4: 0 1 4 6 18 35 27 36 49 48 47 48 50 72 74 75 73 72 73 72 74 64 49
stdevDiff	1.67248	51 65 64 74 66 68 53 65 66 68 53 68 69 67 53 65 66 86 76 89 90 91 90 77 69
		70 55 67 53 68 76 89 90 77 79 91 95 96 95 81 78 80 92 96 93 82 92 81 78 58
		55 54 30 39 40 38 37 36 47 34 27 34 27 17 4 1 0

Bibliography

[1] D. Ahr, "Contributions to Multiple Postmen Problems," PhD Thesis, University of Heidelberg (Germany), September 2004.

[2] D. Ahr, G. Reinelt, "New Heuristics and Lower Bounds for the Min-Max k-Chinese Postman Problem," in R. Möring and R. Raman, editors. *Algorithms - ESA 2002, 10th Annual European Symposium*, Rome, Italy September 17-21, 2002. *Proceedings, Lecture Notes in Computer Science,* 2461. Berlin: Springer. p 64-74.

[3] D. Ahr, G. Reinelt, "A Tabu Search Algorithm for the Min-Max k-Chinese Postman Problem," *Computers & Operations Research* 33, 3403-3422 (2006).

[4] M. Almeida and M. Mourao, "Lower-bounding and Heuristic Methods for a Refuse Collection Vehicle Routing Problem," *European Journal of Operational Research* 121, 420-434 (2000).

[5] L. Amado and M. Mourao, "Heuristic Method for a Mixed Capacitated Arc Routing Problem," *European Journal of Operational Research* 160, 1, 139-153 (2005).

[6] A. Assad and B. Golden, "Arc Routing Methods and Applications," in: M. Ball, T. Magnanti, C. Monma and G. Nemhauser Editors. *Handbooks in Operations Research & Management Science, Volume 8* Amsterdam, The Netherlands: Elsevier Science; 375-483 (1995).

[7] J. Atkins, J. Dierckman, and K. O'Bryant, "A Real Snow Job," *The UMAP Journal* 11, 231-239 (1990).

[8] E.L. Beltrami and L.D. Bodin, "Networks and Vehicle Routing for Municipal Waste Collection," *Networks* 4, 65-94 (1974).

[9] E. Benavent, Á. Corberán, I. Plana and J.M. Sanchis, "Min-Max K-vehicles windy rural postman problem," *Networks* 54, 4, 216226 (2009).

[10] E. Benavent, Á. Corberán and J.M. Sanchis, "An heuristic algorithm for the Min-Max K-vehicles Windy Rural Postman Problem," *Computational Management Science* 7, 269287 (2010).

[11] E. Benavent, Á. Corberán, I. Plana and J.M. Sanchis, "New facets and an enhanced branch-and-cut for the min-max K-vehicles windy rural postman problem," *Networks* (to appear).

[12] E. Benavent, Á. Corberán, G. Desaulniers, F. Lessard I. Plana, and J.M. Sanchis, "A Branch-Price-and-Cut Algorithm for the Min-Max k-Vehicle Windy Rural Postman Problem," (submitted for publication).

[13] L.D. Bodin and S.J. Kursh, "A Computer-Assisted System for the Routing and Scheduling of Street Sweepers," *Operations Research* 26, 525-537 (1978).

[14] L.D. Bodin and S.J. Kursh, "A Detailed Description of a Computer System for the Routing and Scheduling of Street Sweepers," *Computer & Operations Research* 6, 181-198 (1979).

[15] Bureau of Management Consulting, "Improving Snow Clearing Effectiveness in Canadian Municipalities," *Canada: Transportation Development Agency, Ministry of Transport* Catalogue No. T48-9/1975 (1975).

[16] J. Campbell and A. Langevin, "Roadway Snow and Ice Control," in: M. Dror editor. *Arc Routing: Theory, Solutions and Applications* Boston, MA: Kluwer; p. 389-418 (2000).

[17] J. Carlsson and Y. Ye, "Solving Min-Max Multi-Depot Vehicle Routing Problem," in: P. Pardalos and T. Coleman editors. *Lectures on Global Optimization* Providence, RI: American Mathematical Society; p. 31-51 (2009).

[18] R. Chernak, L. Kustiner, and L. Phillips, "The Snowplow Problem," *The UMAP Journal* 11, 241-250 (1990).

[19] J. Clossy, G. Laporte, and P. Soriano, "Solving Arc Routing Problems with Turn Penalties," *Journal of the Operational Research Society* 52, 4, 433-439(7) (2001).

[20] N. Christofides, V. Campos, A. Corberan, and E. Mota, "An Algorithm for the Rural Postman Problem on a Directed Graph," *Mathematical Programming Study* 26, 155-166 (1986).

[21] A. Corberán, I. Plana, and J.M. Sanchis, "A Branch and Cut Algorithm for the Windy General Routing Problem and Special Cases," *Networks* 49, 245-257 (2007).

[22] M. Dror, editor. *Arc Routing: Theory, Solutions and Applications.* Boston, MA: Kluwer (2000)

[23] B. Dussault, B. Golden, C. Groer, and E. Wasil, "A New Variant of the Windy Postman Problem," presented at ROUTE II, Sitges, Spain June (2011).

[24] J. Edmonds, "Optimum Branching," *Journal of Research of the National Bureau of Standards* 71, 233-240 (1967).

[25] R.W. Eglese and H. Murdock, "Routing Road Sweepers in a Rural Area," *The Journal of the Operational Research Society* 42, 281-288 (1991).

[26] H. Eiselt, M. Gendreau, and G. Laporte, "Arc Routing Problems, Part I: The Chinese Postman Problem," *Operations Research* 43, 3, 339-414 (1995).

[27] H. Eiselt, M. Gendreau, and G. Laporte, "Arc Routing Problems, Part II: The Rural Postman Problem," *Operations Research* 43, 2, 231-242 (1995).

[28] B. Eksioglua, A. Vuralb, and A. Reisman, "The Vehicle Routing Problem: A Taxonomic Review," *Computers & Industrial Engineering* 57, 4, 14721483 (2009).

[29] G. Fredrickson, M. Hecht, and C. Kim, "Approximation Algorithms for Some Routing Problems," *SIAM Journal on Computing* 7, 2, 178-193 (1978).

[30] L. Gelders and G. Cattrysse, "Public Waste Collection: A Case Study," *Belgian Journal of Operations Research, Statistics, and Computer Science* 31, 3-15 (1991).

[31] M. Gendreau, A. Hertz, and G. Laporte, "New Insertion and Post Optimization Procedures for the Traveling Salesman Problem," *Operations Research* 40, 1086-1094 (1992).

[32] M. Gendreau, G. Laporte, and S. Yelle, "Efficient Routing of Service Vehicles," *Engineering Optimization* 28, 263-271 (1997).

[33] B. Golden and R. Wong, "Capacitated Arc Routing Problem," *Networks* 11, 305-315 (1981).

[34] R. Gupta and S.R. Arora,"Programming Problem with Maximin Objective Function", Opsearch 14, 125-130 (1977).

[35] A. Haghani and H. Qiao, "Decision Support System for Snow Emergency Vehicle Routing," *Transportation Research Record* 1771, 172-178 (2001).

[36] A. Haghani and H. Qiao, "Snow Emergency Vehicle Routing with Route Continuity Constraints," *Transportation Research Record* 1783, 119-124 (2001).

[37] C. Hartman, K. Hogenson, and J. Miller, "Plower Power," *The UMAP Journal* 11, 261-272 (1990).

[38] E. Haslam and J. Wright, "Application of Routing Technologies to Rural Snow and Ice Control," *Transportation Research Record* 1304, 202-211 (1991).

[39] P. Kandula and J. Wright, "Designing Network Partitions to Improve Maintenance Routing," *Journal of Infrastructure Systems* 3, 160-168 (1997).

[40] A. Kaufmann, "Graph, Dynamic Programming, and Finite Games," New York, NY: Academic Press (1967).

[41] M-K. Kwan, "Graphic Programming using Odd or Even Points," *Chinese Mathematics* 1, 237-277 (1962).

[42] G. Laporte, M.A. Salazar, and A. Langevin,"Synchronized Arc Routing for Snow Plowing," presented at ROUTE 2011, Sitges, Spain, June (2011).

[43] J. Lovell, "Left-hand-turn Elimination," *New York Times* (June 1, 2007).

[44] H. Marks and R. Stricker, "Routing for Public Service Vehicles," *Journal of the Urban Planning and Development Division* 97, 165-78 (1971).

[45] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer Programming Formulation of Traveling Salesman Problems," *Journal of the ACM* 7(4), 326-329 (1960).

[46] E. Minieka, "The Chinese Postman Problem for Mixed Networks," *Management Science* 25, 643-648 (1979)

[47] M. Omer, "Efficient Routing of Snow Removal Vehicles: A Study of Capacitated Arc Routing Problem," Saarbrücken, Germany: VDM Publishing (2008).

[48] N. Perrier, A. Langevin, and J. Campbell, "A Survey of Models and Algorithms for Winter Street Maintenance. Part I: System Design for Spreading and Plowing," *Computers & Operations Research* 33, 209-238 (2006).

[49] N. Perrier, A. Langevin, and J. Campbell, "A Survey of Models and Algorithms for Winter Street Maintenance. Part II: System Design for Snow Disposal," *Computers & Operations Research* 33, 239-262 (2006).

[50] N. Perrier, A. Langevin, and J. Campbell, "A Survey of Models and Algorithms for Winter Street Maintenance. Part III: Vehicle Routing and Depot Location for Spreading," *Computers & Operations Research* 34, 211-267 (2007).

[51] N. Perrier, A. Langevin, and J. Campbell, "A Survey of Models and Algorithms for Winter Street Maintenance. Part VI: Vehicle Routing and Fleet Sizing for Plowing and Snow Disposal," *Computers & Operations Research* 34, 258-294 (2007).

[52] J. Robinson, L. Ogawa, and S. Frickenstein, "The Two-Snowplow Routing Problem," *The UMAP Journal* 11, 251-259 (1990).

[53] M. Salim, T. Strauss, and M. Ernch, "A GIS-Integrated Intelligent System for Optimization of Asset Management for Maintenance of Roads and Bridges," *Lecture Notes in Computer Science* 2358, 628-637 (2002).

[54] S. Toobaie1 and A. Haghani, "Minimal Arc Partitioning Problem: Formulation and Application in Snow Routing with Service Route Continuity," *Transportation Research Record* 1882, 167-175 (2007).

[55] J. Wang and J. Wright, "Interactive Design of Service Routes," *Journal of Transportation Engineering* 120, 897-913 (1994).

[56] Z. Win, "On the Windy Postman Problem on Eulerian Graphs," *Mathematical Programming* 44(1), 97-112 (1989).

[57] S. Wøhlk, "A Decade of Capacitated Arc Routing," in: B. Golden, S. Raghavan, E. Wasil Editors. *The Vehicle Routing Problem: Latest Advances and New Challenges* New York, NY: Springer; p. 29-48 (2008).