

ABSTRACT

Title of dissertation: Applications of Genetic Algorithms,
Dynamic Programming,
and Linear Programming
to Combinatorial Optimization Problems

Xia Wang, Doctor of Philosophy, 2008

Dissertation directed by: Professor Bruce Golden
Applied Mathematics and Scientific Program
Robert H. Smith School of Business

Combinatorial optimization problems are important in operations research and computer science. They include specific, well-known problems such as the bin packing problem, sequencing and scheduling problems, and location and network design problems. Each of these problems has a wide variety of real-world applications. In addition, most of these problems are inherently difficult to solve, as they are NP-hard. No polynomial-time algorithm currently exists for solving them to optimality. Therefore, we are interested in developing high-quality heuristics that find near-optimal solutions in a reasonable amount of computing time.

In this dissertation, we focus on applications of genetic algorithms, dynamic programming, and linear programming to combinatorial optimization problems.

We apply a genetic algorithm to solve the generalized orienteering problem. We use a combination of genetic algorithms and linear program to solve the concave cost supply scheduling problem, the controlled tabular adjustment problem, and

the two-stage transportation problem. Our heuristics are simple in structure and produce high-quality solutions in a reasonable amount of computing time.

Finally, we apply a dynamic programming-based heuristic to solve the shortest pickup planning tour problem with time windows.

Applications of Genetic Algorithm, Dynamic Programming
and Linear Programming
to Combinatorial Optimization Problems

by

Xia Wang

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
2008

Advisory Committee:
Professor Bruce Golden, Chair/Advisor
Professor Edward Wasil
Professor Denny Gulick
Professor Ali Haghani
Professor Konstanitina Trivisa

© Copyright by
Xia Wang
2008

To Andy, Yongjun, and My Parents

Acknowledgments

I want to show my gratitude to all the people who have made this thesis successful.

First and foremost I'd like to thank my advisor, Professor Bruce Golden, for giving me an invaluable opportunity to work on interesting topics in my field. Whenever I face the challenges and hesitate to move forward, he is always there to direct me toward the right road. Without his persistent guidance and knowledgeable insight, I would not have got so far.

I would also like to thank my co-advisor, Professor Edward Wasil. Without his extraordinary theoretical ideas and computational expertise, this thesis would have been a distant dream. Thanks are due to Professor Denny Gulick, Professor Konstantina Trivisa, and Professor Ali Haghan for agreeing to serve on my thesis committee and for sparing their invaluable time reviewing the manuscript. I had been a teaching assistant of Professor Gulick for two successive semesters, and learned a lot in teaching from him.

My appreciation are also shown to my colleagues. Si Chen car pooled with me to almost every Saturday meeting at Professor Golden's house, and we went to conference meeting. We had great fun working together. Yupei Xiong shared his expertise whenever possible, and he is always willing to help.

The staff members I would like to thank are Alverda McCoy, Ruth Zuba, Sharon Welton, Fletcher Kinne, William Schildknecht, and Tony Zhang. They offered numerous support financially, academically, and mentally.

My dearest parents have given me the deepest love unconditionally. They support me any time and I could not pay them back enough. My mother has sacrificed a lot to make sure my father and I have a comfortable life. My father is always my role model who has guided me through darkness.

Last but not least, I'd like to thank my husband, Yongjun Luo. His love and supports make everything possible in my life.

Table of Contents

List of Tables	vii
List of Figures	xii
1 Introduction	1
1.1 Overview	1
1.2 The Concave Cost Supply Scheduling Problem (SSP)	3
1.3 Controlled Tabular Adjustment (CTA)	4
1.4 Two-stage Transportation Problem (TsTP)	5
1.5 Generalized Orienteering Problem (GOP)	6
1.6 The Shortest Pickup Planning Tour Problem with Time Windows (SPTP-TW)	7
2 The Concave Cost Supply Scheduling Problem	8
2.1 Overview	8
2.2 Multiple Providers and a Single Manufacturing Unit	9
2.2.1 Problem Description	9
2.2.2 Literature Review	10
2.2.3 Outline of our Elite Genetic Algorithm	11
2.2.4 Computational Results	15
2.3 Multiple Providers and Multiple Manufacturing Units	16
2.3.1 Problem Description	16
2.3.2 Outline of EGA	18
2.3.3 Computational Results	26
2.3.4 Obtaining Robust Solutions Using EGA	29
2.3.5 Improving the Performance of EGA	30
2.4 Conclusions	31
3 Controlled Tabular Adjustment	34
3.1 Overview	34
3.2 Description of EGA-CTA	38
3.2.1 Details of EGA-CTA	39
3.3 Computational Experiments	46
3.3.1 Selecting the Probability of Mutation	47
3.3.2 EGA-CTA vs. Exact Method	48
3.3.3 QB-EGA (PWM) vs. Hybrid Heuristic	53
3.4 Conclusions	55
4 Two-stage Transportation Problem	57
4.1 Overview	57
4.2 Problem Description	60
4.2.1 Outline of pb-GA	63
4.2.2 Outline of TsTP-EGA	65

4.3	Computational Results	73
4.4	Conclusions	75
5	Generalized Orienteering Problem	77
5.1	Overview	77
5.1.1	Literature review	77
5.1.2	Description of the GOP	82
5.2	Problem Description	83
5.2.1	Structure of our Genetic Algorithm	85
5.2.2	Algorithm Details	86
5.3	Computational Experiments	90
5.4	Conclusions	96
6	The Shortest Pickup Planning Tour Problem with Time Windows	97
6.1	Overview	97
6.1.1	Introduction	97
6.2	Literature Review	99
6.3	Description of Dynamic Programming	106
6.3.1	Introduction to Dynamic Programming	106
6.3.2	Dynamic Programming in Pickup Planning Tour	108
6.3.3	Outline of Dynamic Programming	109
6.4	Computational Experiments	112
6.4.1	Exact Dynamic Programming	115
6.4.2	Computational Results	116
6.5	Conclusions	119
7	Conclusions	120
A	MATLAB Code for Computing Distances	123
B	Data for Ten New Concave Cost Supply Scheduling Problems	124
C	Data for Controlled Tabular Adjustment	158
D	An Example to Illustrate the Dynamic Program	195
E	Data for Four Two-stage Transportation Problems	206
	Bibliography	210

List of Tables

2.1	Comparison between four selected providers	14
2.2	Comparison between three available providers	14
2.3	EGA vs NPSM: Six providers and a single manufacturing unit	17
2.4	Summary of EGA	25
2.5	Short title	27
2.6	Results produced by EGA on 10 new problems	28
2.7	Running time approximation for EGA	29
2.8	Short title	32
2.9	Performance of EGA with New Mutation Strategy	33
3.1	Example to illustrate complementary cell suppression	35
3.2	Example to illustrate controlled tabular adjustment	37
3.3	Example to Illustrate EGA-CTA	44
3.4	Solution 1 after mutation	46
3.5	Solution 2 after mutation	46
3.6	Solution 3 after mutation	46
3.7	Solution 4 after mutation	46
3.8	Solution 5 after mutation	46
3.9	Solution 1 after selection	47
3.10	Solution 2 after selection	47
3.11	Solution 3 after selection	47
3.12	Solution 4 after selection	47
3.13	Solution 5 after selection	47
3.14	Results produced by four variants of EGA-CTA on nine problems	50

3.15	Results produced by QB-EGA (PWM) and EM on 15 problems with 10% sensitive cells	52
3.16	Results produced by QB-EGA (PWM) and EM on 15 problems with 30% sensitive cells	53
3.17	Results produced by QB-EGA (PWM) and HH on three problems with 10% sensitive cells	55
3.18	Results produced by QB-EGA (PWM) and HH on seven problems with 30% sensitive cells	55
4.1	Distance matrices with three plants, four DCs, and five customers . .	65
4.2	Data for test problems	74
4.3	Performance of TsTP-EGA and pb-GA	74
5.1	Locations and Scores of 27 Cities in China	92
5.2	EGA vs. ANN ($k = 1$)	92
5.3	EGA vs. ANN ($k = 3$)	93
5.4	EGA vs. ANN ($k = 4$)	93
5.5	EGA vs. ANN ($k = 5$)	94
5.6	EGA vs. ANN ($k = 10$)	95
6.1	Summary statistics for 20 routes	112
6.2	Performance of APS and HDP on 20 routes with original TW	113
6.3	Performance of APS and HDP on 20 routes with TW1	114
6.4	Performance of APS and HDP on 20 routes with TW2	115
6.5	Performance of APS, HDP, and EDP on 20 routes with original TW .	117
6.6	Performance of APS, HDP, and EDP on 20 routes with TW1	118
6.7	Performance of APS, HDP, and EDP on 20 routes with TW2	119
B.1	Ten Providers and Twelve Manufacturing Units(10P12U)	124

B.2	12P15U	125
B.3	15P18U	126
B.4	18P20U	128
B.5	20P25U	130
B.6	25P30U	132
B.7	30P35U	134
B.8	40P50U	138
B.9	45P50U	144
B.10	50P55U	151
C.1	10by10 table with 10% of sensitive cells	158
C.2	15by15 with 10% of sensitive cells	158
C.3	20by20 with 10% of sensitive cells	159
C.4	25by25 with 10% of sensitive cells	160
C.5	30by30 with 10% of sensitive cells	161
C.6	40by40 with 10% of sensitive cells	163
C.7	50by50 with 10% of sensitive cells	166
C.8	4by4by4 table with 10% of sensitive cells	171
C.9	5by5by3 table with 10% of sensitive cells	171
C.10	4by4by7 table with 10% of sensitive cells	171
C.11	5by5by5 table with 10% of sensitive cells	172
C.12	5by5by7 table with 10% of sensitive cells	172
C.13	10by10by3 table with 10% of sensitive cells	173
C.14	10by10by4 table with 10% of sensitive cells	174
C.15	10by10by4 table with 10% of sensitive cells	175

C.16 11by11 table with 30% of sensitive cells	176
C.17 16by16 with 30% of sensitive cells	176
C.18 21by21 with 10% of sensitive cells	177
C.19 26by26 with 30% of sensitive cells	178
C.20 31by31 with 30% of sensitive cells	179
C.21 41by41 with 30% of sensitive cells	181
C.22 51by51 with 30% of sensitive cells	184
C.23 4by4by4 table with 30% of sensitive cells	189
C.24 5by5by3 table with 30% of sensitive cells	189
C.25 4by4by7 table with 30% of sensitive cells	190
C.26 5by5by7 table with 30% of sensitive cells	190
C.27 5by5by7 table with 30% of sensitive cells	191
C.28 10by10by3 table with 30% of sensitive cells	191
C.29 10by10by4 table with 30% of sensitive cells	192
C.30 10by10by4 table with 30% of sensitive cells	193
D.1 Inputs for example with six nodes	197
D.2 State set	198
D.3 Initialization	202
D.4 Iteration 1	202
D.5 Iteration 2	203
D.6 Iteration 3	203
E.1 Three plants, four DCs, and five customers ($3 \times 4 \times 5$)	207
E.2 $4 \times 5 \times 10$	207
E.3 $4 \times 5 \times 15$	208

E.4	$8 \times 10 \times 20$	209
-----	-------------------------	-------	-----

List of Figures

2.1	Edges opened with respect to the chromosome in (2.35)	20
2.2	Flows on the edges as in (2.36)	21
3.1	PM vs Objective (POP=5)	49
3.2	PM vs Objective (POP=11)	49
3.3	PM vs Objective (POP=21)	49
3.4	PM vs Objective (POP=31)	49
3.5	PM vs Objective (POP=41)	49
3.6	PM vs Objective (POP=51)	49
4.1	A chromosome in pb-GA	64
4.2	The transportation tree after encoding	64
4.3	An example of a TsTP with three plants, four DCs, and five customers	65
4.4	Open edges based on chromosome in (4.9)	66
4.5	Flows from DCs to customers after Step 1	68
4.6	Flows from plants to DCs after Step 2	68
6.1	An example of a pickup route with six customers	98
6.2	Shortest Path Problem	107
D.1	An example with six nodes	196
D.2	Best Path	205

Chapter 1

Introduction

1.1 Overview

Combinatorial optimization is a key area in operations research and computer science. It includes specific, well-known problems such as the traveling salesman problem, the bin packing problem, sequencing and scheduling problems, and location and network design problems. Each of these problems has a wide variety of real-world applications. In addition, most of these problems are inherently difficult to solve, as they are NP-hard. That is, no polynomial-time algorithm currently exists for solving them to optimality. Therefore, we are interested in developing high-quality heuristics that find near-optimal solutions in a reasonable amount of computing time.

The general mathematical form of combinatorial optimization problem is given by (2.1) to (2.3).

$$\text{Optimize } Z = f(X) \tag{1.1}$$

subject to

$$g_i(X) \leq 0, \quad i = 1, 2, \dots, m \tag{1.2}$$

$$X \geq 0, \tag{1.3}$$

where $f(X)$ and $g(X)$ can be general functions of the vector $X = (x_1, x_2, \dots, x_n) \in R^n$. We need to find a column vector X that optimizes (minimizes or maximizes)

the objective function $f(X)$ subject to the m constraints $g_i(X) \leq 0, i = 1, 2, \dots, m$.

In this dissertation, we focus on applications of genetic algorithms, dynamic programming, and linear programming to combinatorial optimization problems. A genetic algorithm (GA) is a stochastic search procedure that works with a population of intermediate solutions over a number of generations, applying ideas such as natural selection, competition, and survival of the fittest, in order to obtain high-quality final solutions (see Michalewicz [38] for background on genetic algorithms). Often, GAs can quickly find good solutions, even for difficult search spaces. GAs can be applied to almost any optimization problem, and are especially useful for problems where calculus-based techniques do not work, such as when the objective function has many local optima, is not differentiable or continuous, or solution elements are constrained to be integers or sequences. Linear programs involve the optimization of a linear objective function $f(X)$, subject to linear equality and inequality constraints, where the decision variables are positive.

In the dissertation, we use a combination of GA and LP to solve (1) the Concave Cost Supply Scheduling Problem (SSP), (2) the Controlled Tabular Adjustment Problem (CTA), and (3) the Two-stage Transportation Problem (TsTP). Our heuristics are simple in structure and produce high-quality solutions in a reasonable amount of computing time.

Dynamic programming (DP) is a way of solving decision problems by finding an optimal strategy. The types of problems that can be solved by DP are problems that can be broken down into a sequence of single decisions made one after the other and problems where several decisions can be separated to give this same structure

(here the word “after” is being used in the sense of both time and space). We apply a dynamic programming-based heuristic to solve the shortest pickup planning tour problem with time windows (SPTP-TW).

Next, we describe the five problems that we are modeling and solving in this dissertation.

1.2 The Concave Cost Supply Scheduling Problem (SSP)

Several providers are available to deliver materials to a manufacturing unit or several manufacturing units. The quantity delivered from each provider is between a minimum value and a maximum value, or it can be zero. The cost occurs if a positive quantity is delivered. Cost is a concave function of quantity. This problem is slightly different from the well-known transportation problem with concave costs. The main difference is that a provider can deliver nothing or a quantity that lies between a lower bound and an upper bound. Our goal is to select the providers that meet all demand while minimizing the total cost of allocating the materials.

We separate the problem into two cases. The first case has n providers and a single manufacturing unit. The second case has n providers and m manufacturing units. In the first case, we develop a genetic algorithm to select the providers and then assign the delivery quantities using a greedy method. In the second case, we develop a genetic algorithm to determine the providers and formulate a linear program to distribute the quantities to the manufacturing units.

1.3 Controlled Tabular Adjustment (CTA)

When government agencies and commercial organizations publish tabular data, they must withhold certain data elements that contain confidential information associated with the data respondents. In a table, a cell is considered sensitive if the publication of the true cell value will disclose a contributor's data to the public. Disclosure methods are then used to protect the sensitive cells. On the other hand, statistical agencies try to publish as much information as possible. This results into a trade-off between privacy rights and information loss, an issue of primary importance in practice.

Controlled tabular adjustment was introduced by Dandekar and Cox [18]. CTA introduces controlled perturbations (adjustments) into tabular data that satisfy the protection ranges and tabular constraints (additivity) while minimizing data loss as measured by one of several linear cell value adjustments. A tabular data protection range assures that all released tabular cells satisfy an appropriate disclosure rule.

We develop a two-phase algorithm that combines an Elite Genetic Algorithm (EGA) with a linear program (LP) (we denote our algorithm by EGA-CTA) to protect sensitive cells. We have exactly two options to perturb each sensitive cell. We use an Elite Genetic Algorithm for quick convergence. EGA sets values in sensitive cells and the LP sets values in nonsensitive cells for each generation.

1.4 Two-stage Transportation Problem (TsTP)

Typically, supply chain management (SCM) involves transportation network design, plant/distribution center (DC) locations, production schedule streamlining, and efforts to improve order response time. Transportation network design is one of the most important fields of SCM. It offers great potential to reduce costs and to improve service quality. The objective is to find the best way of transporting product from several sources to several destinations so that the total cost can be minimized.

The two-stage transportation problem aims to determine the transportation network to satisfy the customer demand at minimum cost subject to the plant and DCs capacity and also the maximum number of DCs that can be opened. Most companies have only limited resources to open and operate DCs.

We develop a two-phase algorithm that combines an elite genetic algorithm with a linear program (we denote our algorithm by TsTP-EGA). In the first phase, we determine the edges from plants to DCs and the edges from DCs to customers. In the second phase, there are two steps. In the first step, we determine the flows from DCs to customers by solving the corresponding LP model. In the second step, we consider DCs as the customers, and the total flow from each DC as the demand of the customer. We then assign the quantities to be delivered from the plants to the DCs by solving the corresponding LP model.

1.5 Generalized Orienteering Problem (GOP)

Let $G(V, E)$ be a complete graph with n points (or cities), where V is the set of points and E is the set of edges in the graph. Both the start and end points are the same (point 1), although this need not be the case. Each point in V has a score vector $S(i) = (S_1(i), S_2(i), \dots, S_m(i))$, where m is the number of independent goals, and $S_g(i)$ is the score of point i with respect to goal g . Given the dominant role played by the point with the largest score with respect to each goal, we define the total score of a path P starting and ending at point 1 as $Z = \sum_{g=1}^m W_g \cdot \{max_{i \in P}(S_g(i))\}$, where W_g is the weight that the traveler attaches to goal g , and $\sum_{g=1}^m W_g = 1$. Without exceeding the pre-defined distance limit, our goal is to determine a path from start to end through a subset of other points in order to maximize the total path score. This objective function is hard to work with because it is nonlinear and not continuously differentiable. We approximate it by a continuously differentiable function that approaches Z as k grows large.

In our GA, each chromosome is a path (tour) starting from and ending at point 1. Each path contains all the points and each point (except the first) is visited exactly once. A population of paths is selected, where each path starts and ends at point 1. In each generation, a queen-bee crossover and two-opt mutation are performed on all paths. Given the distance limit, each path is transformed into a feasible path and the next generation is selected based on proportional fitness. The algorithm is terminated if the number of generations reaches a predefined limit, or the current best solution stays the same for up to five passes.

1.6 The Shortest Pickup Planning Tour Problem with Time Windows (SPTP-TW)

A package delivery company needs to pick up packages from many commercial customers by the end of each day. The typical pickup starts when a driver finishes making deliveries. Drivers visit these customers everyday and, on some days, a customer has no packages for the driver.

The pickup time windows (TWs) are pre-defined in the company's contracts with the customers. We wish to sequence the pickups in order to minimize the total cost incurred while meeting each customer's time window.

We apply a dynamic programming-based heuristic to the shortest pickup planning tour problem with time windows (SPTP-TW). Our heuristic produces very good routes in a reasonable amount of computing time. Our routes are better on average than the existing routes produced by a package delivery company.

Chapter 2

The Concave Cost Supply Scheduling Problem

2.1 Overview

Several providers are available to deliver materials to a manufacturing unit or several manufacturing units. The quantity delivered from each provider is between a minimum value and a maximum value, or it can be zero. The cost occurs if a positive quantity is delivered. Cost is a concave function of quantity. This problem is slightly different from the well-known transportation problem with concave costs. The main difference is that a provider can deliver nothing or a quantity that lies between a lower bound and an upper bound. The lower bound is the economical production quantity imposed by the provider and the upper bound is a technical constraint. In particular, it is the maximum quantity the provider is able to produce during the period under consideration [8]. Our goal is to select the providers that meet all demand while minimizing the total cost of allocating the materials.

We separate the problem into two cases. The first case has n providers and a single manufacturing unit. The second case has n providers and m manufacturing units. In the first case, we develop a genetic algorithm (GA) to select the providers and, then, assign the delivery quantities using a greedy method. In the second case, we develop a genetic algorithm to determine the providers and formulate a mixed integer program (MIP) to distribute the quantities to the manufacturing units.

In Section 2.2, we consider the case of multiple providers and a single manufacturing unit. We develop a genetic algorithm to solve this problem. We apply our algorithm to a set of test problems and compare our results to those that appear in the literature. In Section 2.3, we consider the case of multiple providers and multiple manufacturing units. We solve this problem with our GA and present computational results. In Section 2.4, we give our conclusions.

2.2 Multiple Providers and a Single Manufacturing Unit

2.2.1 Problem Description

We are given a set of providers ($N = \{1, 2, \dots, n\}$) and each provider can deliver raw material to a manufacturing unit. The quantity demanded by a manufacturing unit is equal to D . Each provider supplies a quantity with certain restrictions: either the provider does not supply any material, or the quantity provided is between a minimum value and a maximum value. The minimum value represents the smallest quantity the provider is willing to supply, and the maximum value is the capacity of the provider.

The mathematical programming formulation of this problem (denoted by P1) is given by

$$\begin{aligned} & \min \sum_{i=1}^n k_i(x_i) & (2.1) \\ \text{subject to} & \sum_{i=1}^n x_i = D & (2.2) \\ & x_i \in 0 \cup [m_i, M_i] \quad \forall i \in N, & (2.3) \end{aligned}$$

where m_i is the minimum quantity that provider i is willing to deliver, M_i is

the maximum capacity of provider i , and D is the demand of the manufacturing unit. The decision variable x_i is the quantity delivered from provider i . The manufacturing cost is given by

$$k_i(x_i) = \begin{cases} 0 & \text{if } x_i = 0 \\ a_i + g_i(x_i) & \text{if } x_i > 0, \end{cases} \quad (2.4)$$

where $a_i \geq 0$, $\lim_{x \rightarrow 0^+} g_i(x) \geq 0$, and $g_i(x_i)$ is a concave, continuously differentiable, and increasing function of x_i .

2.2.2 Literature Review

Chauhan and Proth [8] develop a heuristic, called NPSM (N Providers, Single Manufacturing unit), that is based on two key results.

Result 1. There exists at least one optimal solution $X = \{x_1, x_2, \dots, x_n\}$, such that $x_i = m_i$ or $x_i = M_i$ or $x_i = 0$ for $i \in \{1, 2, \dots, n\}$, except perhaps for one $i^* \in \{1, 2, \dots, n\}$, where $m_{i^*} < x_{i^*} < M_{i^*}$.

Result 1 gives a necessary condition for a solution to be optimal. Result 2 (not discussed here) provides a way to check if a solution that satisfies the conditions of Result 1 is optimal.

NPSM has two main steps. In the first step, the provider i_0 with the smallest cost per unit at the maximum quantity is selected (the derivative of $k_{i_0}(x)$ is the smallest, where x is the maximum quantity that can be delivered). The demand is then reduced accordingly. NPSM then selects the provider with the second smallest cost per unit at the maximum quantity. This maximum quantity is assigned to this provider and the remaining demand is reduced accordingly, and so on. The first step

terminates when the maximum quantity that can be provided by the next cheapest provider is greater than the remaining demand.

In the second step, all remaining providers are considered and each one falls into one of the following three cases.

1. If the remaining demand (RD) is between the minimum quantity and the maximum quantity the next provider can deliver, then RD is tentatively assigned to the provider.
2. If RD is less than the minimum quantity that the provider can deliver, then this gap is filled by taking the appropriate quantity from the providers selected in the first step. Select those providers for whom the reduction of the quantity leads to the greatest reduction of cost.
3. If RD is greater than the maximum quantity that the provider can deliver, the maximum quantity is assigned to this provider, and the demand is reduced accordingly. Go back to Case 1. A provider can never be classified as Case 3 at the beginning of Step 2.

In each case, Chauhan and Proth [8] apply an adjustment process to verify Results 1 and 2, that is, shifting quantities to cheaper providers based on Result 2. At the end of the algorithm, a cost associated with each of the remaining providers is calculated. The cheapest solution is then selected.

2.2.3 Outline of our Elite Genetic Algorithm

The adjustment process of NPSM is complicated. In order to simplify the adjustment process without sacrificing solution quality, we use only Result 1 from Chauhan and Proth [8] to construct an elite genetic algorithm (EGA) to solve P1. In our elite genetic algorithm, we keep the current best solution from crossover and mutation to ensure we do not lose solution quality.

In the initialization step, we randomly select a subset of the providers, and always check for the feasibility by examining the following inequality

$$\sum_{i=1}^n M_i \times I(i) \geq D. \quad (2.5)$$

Inequality (2.5) ensures the total capacity of the selected providers will be greater than the demand. The indicator function $I(i) = 1$ if provider i is selected, and $I(i) = 0$ otherwise. Assume the population size is POP , then we randomly generate POP chromosomes that satisfy (2.5) as the initial population in our experiments. The following vector is a possible chromosome, with length equal to ten, which is the number of providers.

$$(1 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 0) \quad (2.6)$$

We now evolve the initial population. Each generation has two phases.

Phase 1: Perform EGA queen-bee crossover and mutation on the population (the details of these two operations are given in Section 2.3.2). We always check (2.5) for feasibility.

Phase 2: Greedy Method

1. First assign the minimum quantity to each provider selected in Phase 1.
2. Select the provider with the smallest derivative of the cost function at the quantity that can be provided (that is, the derivative at the value of minimum $(RD, M_i - m_i)$, where RD is the remaining demand for provider i). For example, provider i_0 is selected and assigned with quantities equal to $\text{minimum}(RD, M_{i_0} - m_{i_0})$.

3. We then calculate the remaining demand and capacity of providers accordingly.

Our algorithm terminates when the best solution is the same for five generations.

We use a small example to explain the process.

The demand of the manufacturing unit is 340. There are six providers and a single manufacturing unit. The capacities and cost functions of the six providers are given by

$$k_1(x) = \begin{cases} 0 & \text{if } x = 0 \\ 2 + 0.1x & \text{if } x > 0 \end{cases}, \quad m_1 = 20, \quad M_1 = 100 \quad (2.7)$$

$$k_2(x) = \begin{cases} 0 & \text{if } x = 0 \\ 4 - e^{-0.1x} & \text{if } x > 0 \end{cases}, \quad m_2 = 30, \quad M_2 = 80 \quad (2.8)$$

$$k_3(x) = \begin{cases} 0 & \text{if } x = 0 \\ 7 - e^{-0.3x} & \text{if } x > 0 \end{cases}, \quad m_3 = 10, \quad M_3 = 50 \quad (2.9)$$

$$k_4(x) = \begin{cases} 0 & \text{if } x = 0 \\ \frac{2+5x}{x+1} & \text{if } x > 0 \end{cases}, \quad m_4 = 40, \quad M_4 = 120 \quad (2.10)$$

$$k_5(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 + 0.3x & \text{if } x > 0 \end{cases}, \quad m_5 = 15, \quad M_5 = 70 \quad (2.11)$$

$$k_6(x) = \begin{cases} 0 & \text{if } x = 0 \\ 5 - e^{-0.2x} & \text{if } x > 0 \end{cases}, \quad m_6 = 5, \quad M_6 = 60. \quad (2.12)$$

After Phase 1, we have one chromosome like the following

$$(1 \ 1 \ 0 \ 1 \ 0 \ 1). \quad (2.13)$$

Phase 2:

1. We first assign the minimum quantity to each selected provider, and the resulting assignment is

$$(20 \ 30 \ 0 \ 40 \ 0 \ 5). \quad (2.14)$$

The remaining demand RD is then 245.

Table 2.1: Comparison between four selected providers

Providers	P1	P2	P4	P6
$k_i(x)$	$2+0.1x$	$4 - e^{-0.1x}$	$\frac{2+5x}{x+1}$	$5 - e^{-0.2x}$
$k'_i(x)$	0.1	$0.1 * e^{-0.1x}$	$\frac{3}{(x+1)^2}$	$0.2 * e^{-0.2x}$
$\min(RD, M_i - m_i)$	80	50	80	55
$k'_i(\min(RD, M_i - m_i))$	0.1	$6.74 * e^{-4}$	$4.57 * e^{-4}$	$3.34 * e^{-6}$

Table 2.2: Comparison between three available providers

Providers	P1	P2	P4
$k_i(x)$	$2+0.1x$	$4 - e^{-0.1x}$	$\frac{2+5x}{x+1}$
$k'_i(x)$	0.1	$0.1 * e^{-0.1x}$	$\frac{3}{(x+1)^2}$
$\min(RD, M_i - m_i)$	80	50	80
$k'_i(\min(RD, M_i - m_i))$	0.1	$6.74 * e^{-4}$	$4.57 * e^{-4}$

2. From the last row in Table 2.1, the cost function of provider P6 has the smallest derivative. We then select provider 6 and assign an additional quantity of 55. The assignment is then

$$(20 \ 30 \ 0 \ 40 \ 0 \ 60). \quad (2.15)$$

3. Then we adjust the remaining demand, which is 190 after step 2. The remaining capacity of provider 6 is 0, and therefore is not available. We select the cheapest provider among the remaining three providers. From Table 2.2, the cost function of provider P4 has the smallest derivative. We then select provider 4 and assign an additional quantity of 80. The assignment becomes

$$(20 \ 30 \ 0 \ 120 \ 0 \ 60). \quad (2.16)$$

The remaining demand is 110. The next cheapest provider is P2, and we assign an additional 50 units to P2. The assignment is

$$(20 \ 80 \ 0 \ 120 \ 0 \ 60). \quad (2.17)$$

We then assign the remaining additional demand of 60 to P1, and the total demand is satisfied. The complete assignment is

$$(80 \ 80 \ 0 \ 120 \ 0 \ 60). \quad (2.18)$$

2.2.4 Computational Results

We compare the performance of EGA to NPSM from Chauhan and Proth [8]. We coded EGA in MATLAB and used a 3.06 GHz Pentium IV machine with 1GB RAM. We selected the 20 problems from [8], where the cost functions are concave. There are six providers and a single manufacturing unit in each problem. The capacities and cost functions of the six providers are given by

$$k_1(x) = \begin{cases} 0 & \text{if } x = 0 \\ 2 + p_1x & \text{if } x > 0 \end{cases}, \quad m_1 = 20, \quad M_1 = 100 \quad (2.19)$$

$$k_2(x) = \begin{cases} 0 & \text{if } x = 0 \\ 4 - e^{-p_2x} & \text{if } x > 0 \end{cases}, \quad m_2 = 30, \quad M_2 = 80 \quad (2.20)$$

$$k_3(x) = \begin{cases} 0 & \text{if } x = 0 \\ 7 - e^{-p_3x} & \text{if } x > 0 \end{cases}, \quad m_3 = 10, \quad M_3 = 50 \quad (2.21)$$

$$k_4(x) = \begin{cases} 0 & \text{if } x = 0 \\ \frac{2+p_4x}{x+1} & \text{if } x > 0 \end{cases}, \quad m_4 = 40, \quad M_4 = 120 \quad (2.22)$$

$$k_5(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 + p_5x & \text{if } x > 0 \end{cases}, \quad m_5 = 15, \quad M_5 = 70 \quad (2.23)$$

$$k_6(x) = \begin{cases} 0 & \text{if } x = 0 \\ 5 - e^{-p_6x} & \text{if } x > 0 \end{cases}, \quad m_6 = 5, \quad M_6 = 60. \quad (2.24)$$

We set the population size of EGA to 10. The results from EGA and NPSM are shown in Table 2.3. The optimal solutions are given by Chauhan and Proth [8]. To produce the optimal solutions, they enumerate all feasible solutions. According to Result 1, the optimal solution has $x_i^* \in \{0, m_i, M_i\}$ with the possible exception of one provider. There are n providers. They first select $n - 1$ providers (there are n

possible choices) and assign either 0, m_i , or M_i to each of them (there are 3^{n-1} such assignments). Thus, the number of trials required to produce an optimal solution is $n3^{n-1}$. The results in Table 2.3 show that EGA and NPSM find the optimal solution to each problem. EGA is very fast: typically, it finds the optimal solution in less than one second.

We point out that there is an error in the published results in Chauhan et al. [8]. The optimal solution value to problem 17 is not 30.2, but 39.0. The correct cost figure is shown in Table 2.3.

2.3 Multiple Providers and Multiple Manufacturing Units

2.3.1 Problem Description

We have a set of providers ($N = \{1, 2, \dots, n\}$) and a set of manufacturing units ($M = \{1, 2, \dots, m\}$). Each provider can deliver materials to each manufacturing unit.

The mathematical programming formulation of this problem (denoted by P2) is given by

$$\min Z = \sum_{i=1}^n \sum_{j=1}^m k_{ij}(x_{ij}) \quad (2.25)$$

$$\text{subject to} \quad \sum_{i=1}^n x_{ij} = D_j, \quad \forall j \in M \quad (2.26)$$

$$\sum_{j=1}^m x_{ij} \leq M_i, \quad \forall i \in N \quad (2.27)$$

$$x_{ij} \in 0 \cup [m_i, M_i] \quad \forall i \in N, j \in M. \quad (2.28)$$

The manufacturing cost is given by

$$k_{ij}(x) = \begin{cases} 0 & \text{if } x = 0 \\ a_{ij} + g_{ij}(x) & \text{if } x > 0, \end{cases} \quad (2.29)$$

Table 2.3: EGA vs NPSM: Six providers and a single manufacturing unit

Problem	Method	x_1	x_2	x_3	x_4	x_5	x_6	Cost	Time(sec)
1	EGA	80	80	0	120	0	60	23.98	0.24
	NPSM	80	80	0	120	0	60	23.98	
	Optimal	80	80	0	120	0	60	23.98	
2	EGA	0	80	50	120	30	60	30.98	0.3
	NPSM	0	80	50	120	30	60	30.98	
	Optimal	0	80	50	120	30	60	30.98	
3	EGA	100	80	50	0	50	60	44.00	0.27
	NPSM	100	80	50	0	50	60	44.00	
	Optimal	100	80	50	0	50	60	44.00	
4	EGA	80	80	0	120	0	60	23.975	0.31
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
5	EGA	80	80	0	120	0	60	23.975	0.20
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
6	EGA	80	80	0	120	0	60	23.975	0.24
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
7	EGA	80	80	0	120	0	60	23.975	0.25
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
8	EGA	80	80	0	120	0	60	23.973	0.19
	NPSM	80	80	0	120	0	60	23.973	
	Optimal	80	80	0	120	0	60	23.973	
9	EGA	0	80	50	120	30	60	27.973	0.23
	NPSM	0	80	50	120	30	60	27.973	
	Optimal	0	80	50	120	30	60	27.973	
10	EGA	0	80	50	120	30	60	72.60	0.31
	NPSM	0	80	50	120	30	60	72.60	
	Optimal	0	80	50	120	30	60	72.60	
11	EGA	0	80	50	120	30	60	27.973	0.23
	NPSM	0	80	50	120	30	60	27.973	
	Optimal	0	80	50	120	30	60	27.973	
12	EGA	0	80	50	120	30	60	27.975	0.35
	NPSM	0	80	50	120	30	60	27.975	
	Optimal	0	80	50	120	30	60	27.975	
13	EGA	0	80	50	120	30	60	27.975	0.29
	NPSM	0	80	50	120	30	60	27.975	
	Optimal	0	80	50	120	30	60	27.975	
14	EGA	0	80	50	120	30	60	27.975	0.30
	NPSM	0	80	50	120	30	60	27.975	
	Optimal	0	80	50	120	30	60	27.975	
15	EGA	80	80	0	120	0	60	23.975	0.28
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
16	EGA	80	80	0	120	0	60	23.975	0.27
	NPSM	80	80	0	120	0	60	23.975	
	Optimal	80	80	0	120	0	60	23.975	
17	EGA	100	80	50	0	50	60	39.00	0.28
	NPSM	100	80	50	0	50	60	39.00	
	Optimal	100	80	50	0	50	60	39.00	
18	EGA	0	80	50	120	30	60	72.60	0.28
	NPSM	0	80	50	120	30	60	72.60	
	Optimal	0	80	50	120	30	60	72.60	
19	EGA	0	80	50	120	30	60	72.60	0.31
	NPSM	0	80	50	120	30	60	72.60	
	Optimal	0	80	50	120	30	60	72.60	
20	EGA	0	80	50	120	30	60	72.60	0.26
	NPSM	0	80	50	80	70	60	72.60	
	Optimal	0	80	50	80	70	60	72.60	

where $a_{ij} \geq 0$, $\lim_{x \rightarrow 0^+} g_{ij}(x) \geq 0$, and $g_{ij}(x)$ is a concave, continuously differentiable, and increasing function of x . However, we only consider a linear function $g_{ij}(x)$, since we want to compare the results produced by EGA to the optimal solutions, which can be found only if $g_{ij}(x)$ is linear. Therefore, we define $g_{ij}(x) = b_{ij}x$, where b_{ij} is the variable cost per unit. M_i is the maximum capacity of provider i , m_i is the minimum quantity that provider i is willing to deliver to manufacturing unit j , and D_j is the demand of manufacturing unit j . The decision variable x_{ij} is the quantity delivered from provider i to manufacturing unit j .

2.3.2 Outline of EGA

We describe our elite genetic algorithm for solving the problem with n providers and m manufacturing units. The term elite implies that we keep the current best solution from crossover and mutation during the evolution.

Step 0. Initialization. Randomly select a manufacturing unit and use a greedy method (we use the same greedy method as in Phase 2 of EGA in Section 2.2.3) to meet its demand. For example, we select unit j^* , and order the delivery costs per unit, namely, $\frac{a_{ij^*} + b_{ij^*}M_i}{M_i}$, from smallest to largest, for all providers. Suppose provider i_0 has the smallest delivery cost per unit. Then we assign a quantity equal to $\min(D_{j^*}, M_{i_0})$ to this provider, and reduce D_{j^*} and M_{i_0} accordingly. We have to ensure that each provider delivers at least m_i . Once the demand of manufacturing unit j^* has been met, we continue to randomly select the remaining manufacturing units, until all demands have been met. In order to apply a genetic algorithm, we

encode the feasible solution into a chromosome. Let $P = P_{ij}$ be an $n \times m$ binary matrix, where $P_{ij} = 1$ if we assign provider i to manufacturing unit j , and $P_{ij} = 0$ otherwise. We now have the initial population which consists of binary matrices. All the initial chromosomes can produce feasible solutions in our LP, which is given by

$$\min Z = \sum_{i=1}^n \sum_{j=1}^m (a_{ij}P_{ij} + b_{ij}x_{ij}) \quad (2.30)$$

$$\text{subject to} \quad \sum_{i=1}^n x_{ij} = D_j, \quad \forall j \in M \quad (2.31)$$

$$\sum_{j=1}^m x_{ij} \leq M_i, \quad \forall i \in N \quad (2.32)$$

$$m_i P_{ij} \leq x_{ij} \leq M_i P_{ij}, \quad \forall i \in N, j \in M \quad (2.33)$$

$$x_{ij} \geq 0, \quad \forall i \in N, j \in M \quad (2.34)$$

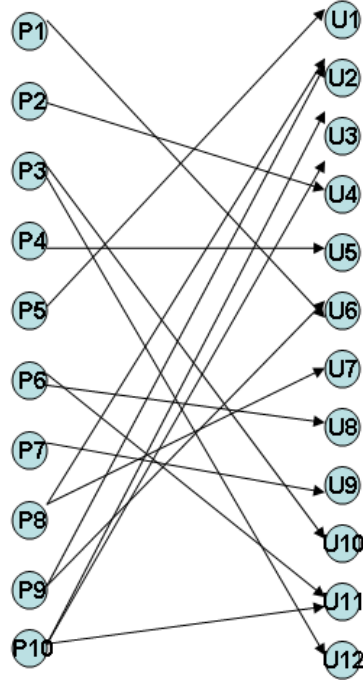
where x_{ij} are the decision variables and P_{ij} are the binary entries in the matrix, which are determined by the greedy method.

Suppose we have 10 providers and 12 manufacturing units, and we generate 50 chromosomes initially (the initial population size is 50). Each chromosome is a 10×12 matrix, such as the matrix in (2.35). The graph associated with this chromosome is displayed in Fig 6.1.

$$P_{10,12} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \quad (2.35)$$

Each member of the initial population is fed into the LP defined in (2.30) to (2.34) and the LP is solved. Suppose we have the LP solution in (2.36) corresponding

Figure 2.1: Edges opened with respect to the chromosome in (2.35)

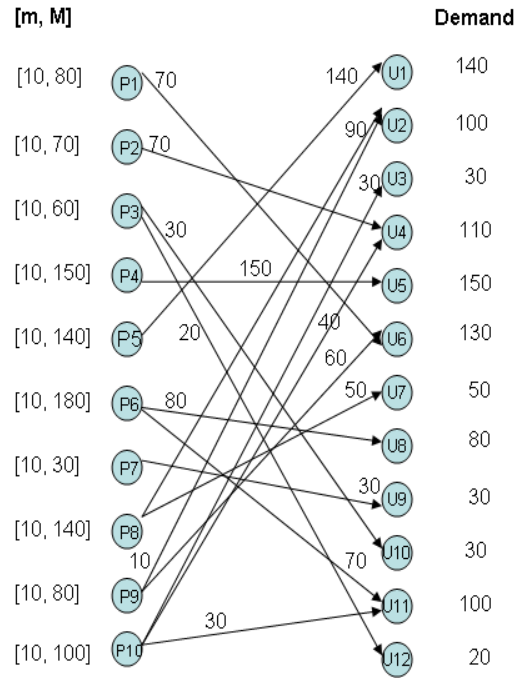


to the chromosome in (2.35), and the graph associated with this solution is displayed in Fig 6.2. We point out that the LP solution is never worse than and is often better than the solution produced by the greedy method.

Step 1. Evolution.

Step 1.1. Queen-Bee Crossover. We use the current best chromosome as the queen and select 25 chromosomes in proportion to their fitness, to mate with the queen, where fitness is the reciprocal of cost. We take the union of the two chromosomes as a base for the new chromosome. We open the route from provider i to unit j only if it is either open in the queen or in the (regular) bee, and close the route otherwise. We use the same greedy method as in the initialization phase on these open routes to select a subset of these routes. The chromosome that corresponds to this subset of routes is the new chromosome after crossover. Suppose

Figure 2.2: Flows on the edges as in (2.36)



we have queen in (2.37) and bee in (2.38).

$$\begin{pmatrix}
 0 & 0 & 0 & 0 & 0 & 70 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 70 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 0 & 20 \\
 0 & 0 & 0 & 0 & 150 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 140 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 80 & 0 & 0 & 70 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 0 & 0 & 0 \\
 0 & 90 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\
 0 & 10 & 0 & 0 & 0 & 60 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 30 & 40 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 0
 \end{pmatrix} \tag{2.36}$$

$$\text{Queen} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (2.37)$$

$$\text{Bee} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (2.38)$$

We close all routes that are not in the queen or the bee. This produces the following intermediate matrix.

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (2.39)$$

After applying the greedy method (see Section 2.2.3) using only the routes or edges that correspond to the nonzero entries in (2.39), we generate the following

matrix as a chromosome.

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (2.40)$$

Note that (2.40) has fewer nonzero entries than (2.39). For example, of the four possible routes from provider 10 in (2.39), only two of these are open in the greedy solution (2.40). Next, we see if we can obtain an improved solution by solving the LP in (2.30) to (2.34).

Step 1.2. Mutation. We apply point-wise mutation with probability $PM = 0.5$. For each chromosome, we flip every gene, that is, $P_{ij} = 1 - P_{ij}$. In general, feasibility will be maintained by mutation. This is due to the fact that there are fewer nonzero entries than zeros in every chromosome, and by flipping, we generate more nonzero entries. Therefore, more edges are open. Then we assign the quantity by only considering the open edges. We use the greedy method to reassign the providers and only consider the open edges. Suppose we apply mutation to the chromosome in (2.40), and produce the following matrix.

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \quad (2.41)$$

After applying the greedy method, we generate the following matrix as a chromosome.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.42)$$

Using our LP, we obtain the fitness of the new chromosome. We now have 25 new chromosomes after queen-bee crossover and 25 new chromosomes after mutation.

Step 1.3. Selection. To generate the next population, 50 chromosomes are selected from 50 children and 50 parents using proportional selection. Since we use elite genetic algorithm, we keep the current best chromosome from crossover and mutation. The current best chromosome is updated if a new best chromosome is produced after crossover and mutation.

We stop the algorithm if the maximum step size is reached or the current best solution is the same for five generations.

Table 2.4: Summary of EGA

Step	EGA for Single Manufacturing Unit
0.	Initialization
1.	Apply queen-bee crossover to select the providers Use greedy assignment to produce the fitness of the chromosome Apply mutation to select the providers Use greedy assignment to produce the fitness of the chromosome
2.	Use proportional selection to select the next generation
3.	Stop when the best chromosome is the same for five generations
Step	EGA for Multiple Manufacturing Units
0.	Initialization
1.	Apply queen-bee crossover and greedy assignment to select the providers Solve LP to produce the fitness of the chromosome Apply mutation and greedy assignment to select the providers Solve LP to produce the fitness of the chromosome
2.	Use proportional selection to select the next generation
3.	Stop when the best chromosome is the same for five generations

The greedy method works as follows. For each manufacturing unit, say manufacturing unit j , select the provider i_1 , who offers the smallest cost per unit. Manufacturing unit j orders the maximum possible quantity from this provider, that is, $\min(RD_j, M_{i_1})$, where RD_j is the remaining demand of manufacturing unit j . The demand is reduced accordingly. We then select the provider i_2 who offers the second smallest cost per unit. Manufacturing unit j orders the maximum possible quantity from this provider, that is, $\min(RD_j, M_{i_2})$. Providers are selected until all of the demand of each manufacturing unit has been met. We order the providers dynamically, that is, we order the providers after every assignment has been made to the manufacturing units.

In Table 2.4, we summarize EGA for solving problems with single and multiple manufacturing units.

2.3.3 Computational Results

We compare the performance of EGA to the NPMU heuristic from Chauhan and Proth [8]. We coded EGA in MATLAB and used a 3.06 GHz Pentium IV machine with 1GB RAM. We point out that MATLAB is slow. So, one should focus on growth rates rather than just running times. We use the 10 problems from Chauhan and Proth [8], where each problem has three providers and four manufacturing units. Chauhan and Proth [8] specified linear cost functions, so that optimal solutions were easy to find. The results from EGA and NPMU are given in Table 2.5. The optimal solutions are given by Chauhan and Proth [8]. To produce the optimal solutions, they enumerated all $2^n - 1$ possible combinations of providers, where n is the number of providers. Some sets were infeasible. They solved a linear programming problem for each selected set. At the end of the $2^n - 1$ computations, they chose a feasible set of providers that produced the smallest cost. This can be done only because the costs are linear for strictly positive delivery amounts. For example, in the vector $(1, 1, \dots, 1)$, all providers are selected, while in $(1, 0, 0, \dots, 1)$, only the first provider and the last provider are selected. The vector $(0, 0, \dots, 0)$ is infeasible, so that we have $2^n - 1$ possible combinations of providers, and some of these will be infeasible. Chauhan and Proth solved an LP for each set. For example, they solved an LP based on $(1, 0, 0, \dots, 1)$ by considering the nonzero entries (providers). They selected the vector that produced the lowest cost to the LP. They were able to enumerate all possibilities because the problem size is small.

In Table 2.5, we see that EGA finds eight optimal solutions, while NPMU finds

Table 2.5: EGA vs NPMU: Three providers and four manufacturing units

Problem	Method	x_{11}	x_{12}	x_{13}	x_{14}	x_{21}	x_{22}	x_{23}	x_{24}	x_{31}	x_{32}	x_{33}	x_{34}	Cost	Time(sec)
1	EGA	0	0	0	80	70	0	90	0	0	50	0	0	294	12.3
	NPMU	0	0	80	0	70	0	0	80	0	50	10	0	309	
	OPT	0	0	0	80	70	0	90	0	0	50	0	0	294	
2	EGA	0	0	0	80	70	0	90	0	0	50	0	0	274	10.5
	NPMU	0	0	80	0	70	0	0	80	0	50	10	0	279	
	OPT	0	0	0	80	70	90	0	0	0	50	0	0	272	
3	EGA	0	0	0	80	0	0	90	0	70	50	0	0	294	11.2
	NPMU	0	0	80	0	70	0	0	80	0	50	10	0	309	
	OPT	0	0	0	80	0	0	90	0	70	50	0	0	294	
4	EGA	0	0	0	80	0	0	90	0	70	50	0	0	294	12.8
	NPMU	0	0	80	0	70	0	0	20	0	50	10	60	301	
	OPT	0	0	0	80	0	0	90	0	70	50	0	0	294	
5	EGA	0	0	0	80	0	0	90	0	70	50	0	0	294	10.2
	NPMU	0	0	80	0	70	0	0	80	0	50	10	0	309	
	OPT	0	0	0	80	0	0	90	0	70	50	0	0	294	
6	EGA	0	0	0	80	0	0	90	0	70	50	0	0	239	12.3
	NPMU	0	0	0	80	0	0	90	0	70	50	0	0	239	
	OPT	0	0	0	80	0	0	90	0	70	50	0	0	239	
7	EGA	0	0	0	80	0	0	90	0	70	50	0	0	239	10.5
	NPMU	0	0	0	80	0	0	90	0	70	50	0	0	239	
	OPT	0	0	0	80	0	0	90	0	70	50	0	0	239	
8	EGA	70	0	0	0	0	0	90	80	0	50	0	0	294	10.3
	NPMU	70	0	0	0	0	0	90	10	0	50	0	70	299	
	OPT	70	10	0	0	0	0	90	0	0	40	0	80	293	
9	EGA	0	0	0	80	70	20	0	0	0	30	90	0	253	10.5
	NPMU	70	10	0	0	0	10	0	80	0	30	90	0	265	
	OPT	0	0	0	80	70	20	0	0	0	30	90	0	253	
10	EGA	0	0	0	80	70	20	0	0	0	30	90	0	253	10.9
	NPMU	0	0	0	80	70	20	0	0	0	30	90	0	265	
	OPT	0	0	0	10	0	80	0	80	0	30	90	0	253	

Bold indicates optimal solution

two optimal solutions. The computation time of EGA is based on one run of the algorithm as the problems are small in size.

Next, we generate 10 new problems, with size ranging from 10 providers and 12 manufacturing units to 50 providers and 55 manufacturing units. The cost functions are piecewise linear, with variable cost uniformly distributed on $[0.1, 1]$ and fixed cost uniformly distributed on $[1, 20]$. The capacity of each provider is uniformly distributed on $[50, 200]$ and the minimum delivery quantity is 10. Demand is uniformly distributed between 40% of the minimum provider capacity and 75% of the maximum provider capacity. If the total demand exceeds the total capacity,

Table 2.6: Results produced by EGA on 10 new problems

Problem	Method	Cost	Gap (%)	Time(Sec)
10P12U	EGA	336	0.00	147.2
	OPT	336		0.7
12P15U	EGA	401	1.01	128.7
	OPT	397		1.1
15P18U	EGA	507	0.00	256.0
	OPT	507		0.9
18P20U	EGA	514	0.00	277.3
	OPT	514		6.6
20P25U	EGA	551	0.92	208.2
	OPT	546		4.4
25P30U	EGA	620	1.47	323.6
	OPT	611		69.1
30P35U	EGA	805	2.16	432.5
	OPT	788		3604.1
40P50U	EGA	745	5.08	452.8
	OPT	709		5045.2
45P50U	EGA	845	5.10	555.9
	OPT	804		6195.7
50P55U	EGA	876	4.66	610.2
	OPT	837		36606.3

1. 10P12U is a problem with 10 providers and 12 manufacturing units.
2. Cost and Time are the average of 10 runs for EGA.
3. OPT is the optimal solution produced by one run of Xpress.
4. GAP is the percentage above OPT for EGA.

we decrease the total demand so that it is less than the total capacity. The test problems are given in Appendix B.

We use Xpress [19] to solve the following mixed integer programming (MIP).

$$\min Z = \sum_{i=1}^n \sum_{j=1}^m (a_{ij}y_{ij} + b_{ij}x_{ij}) \quad (2.43)$$

$$\text{subject to} \quad \sum_{i=1}^n x_{ij} = D_j, \quad \forall j \in M \quad (2.44)$$

$$\sum_{j=1}^m x_{ij} \leq M_i, \quad \forall i \in N \quad (2.45)$$

$$m_i y_{ij} \leq x_{ij} \leq M_i y_{ij}, \quad \forall i \in N, j \in M \quad (2.46)$$

$$x_{ij} \geq 0, y_{ij} \in \{0, 1\}, \quad \forall i \in N, j \in M \quad (2.47)$$

where the costs are linear. The decision variable y_{ij} equals 1 if provider i makes a delivery to manufacturing unit j and 0 otherwise. The decision variable x_{ij} is the quantity delivered from provider i to manufacturing unit j .

In Table 2.6, we give the results produced by EGA on the 10 new problems. We

Table 2.7: Running time approximation for EGA

Problem	nm	\sqrt{nm}	Time(Sec)
10P12U	120	10.95	147.2
12P15U	180	13.41	128.7
15P18U	270	16.43	256.0
18P20U	360	18.97	277.3
20P25U	500	22.36	208.2
25P30U	750	27.39	323.6
30P35U	1050	32.40	432.5
40P50U	2000	44.72	452.8
45P50U	2250	47.43	555.9
50P55U	2750	52.44	610.2

used a 3.06 GHz Pentium IV machine with 1GB RAM. We performed 10 replications and averaged the 10 solution values and the total running times of the replications. EGA produced an optimal solution to three problems (10P12U, 15P18U, 18P20U). On average, a solution generated by EGA was 2.17% above the optimal solution. On average, EGA took 339.24 seconds to solve a problem while Xpress took 5153.4 seconds.

The running time of EGA for multiple manufacturing units is approximately a multiple μ of \sqrt{nm} , where n is the number of providers and m is the number of manufacturing units. The approximations are shown in Table 4.2. We point out that for μ between 12 and 13, a good fit results. The key observation here is that EGA's running time grows slowly in n and m .

2.3.4 Obtaining Robust Solutions Using EGA

We might want to generate solutions that are both cost-effective and robust. For example, suppose we want to protect against small interruptions (decreases) in supply from the providers. In particular, let's suppose the maximum supply of each

provider is reduced by 3%.

In Table 2.8, we present the results produced by EGA on the reduced capacity problems (we denote these by EGAK, where $k = 3, 5,$ and 7). The last column is the gap between EGAK and EGA in percentage, which indicates that by allowing for a slight (unplanned) reduction in supply, we sacrifice very little in cost, less than 5%. Therefore, building robustness into the solutions in this and similar ways seems like a very reasonable thing to do.

2.3.5 Improving the Performance of EGA

To improve the performance of EGA, we consider a new mutation strategy. After selecting 50 individuals from the children and parents of the current population, we mutate these individuals in the following way. We select the best m individuals. For each one of these m individuals, randomly select a row (this row represents a provider). In this row, change all zero entries to entries of one. Use the greedy assignment and the LP to find the optimal solution. We keep the improved individuals and discard the unimproved ones.

We record the results provided by EGA without the new mutation strategy, where the population size is 50. EGA1(POP- m) records the results with the new mutation strategy, where the population size is POP and the best m individuals are selected for mutation. From Table 2.9, we see that all four variants of EGAs produce optimal solutions on the first four problems. Overall, EGA1(80-10) seems to produce the best results.

2.4 Conclusions

We developed two genetic algorithms (GAs) to solve two versions of the concave cost supply scheduling problem. The first case has n providers and a single manufacturing unit, and we use a genetic algorithm to select the providers and assign quantities with the greedy method. The first GA produces optimal solutions to all of the benchmark problem instances. The second case has n providers and m manufacturing units. We use a genetic algorithm to select the providers, then formulate a linear program to distribute the quantities to be delivered, since the greedy method lacks the sophistication to handle the complexities of this problem. Our computational results show that the GA is simple in structure, quick in convergence, and yields nearly optimal solutions.

Table 2.8: Performance of EGA when provider capacity is decreased

Problem	Method	Cost	Time(sec)	Gap (%)
10P12U	EGA	336	147.2	
	EGA3	336	131.9	0.00
	EGA5	338	151.0	0.60
	EGA7	336	181.3	0.00
12P15U	EGA	401	128.7	
	EGA3	401	198.8	0.00
	EGA5	401	152.7	0.00
	EGA7	401	186.7	0.00
15P18U	EGA	507	256.0	
	EGA3	511	300.9	0.79
	EGA5	508	251.9	0.20
	EGA7	510	264.0	0.59
18P20U	EGA	514	277.3	
	EGA3	516	470.2	0.39
	EGA5	522	262.7	1.56
	EGA7	521	265.9	1.36
20P25U	EGA	551	208.2	
	EGA3	553	265.1	0.36
	EGA5	551	247.4	0.00
	EGA7	551	261.4	0.00
25P30U	EGA	620	323.6	
	EGA3	626	391.7	0.97
	EGA5	632	294.8	1.93
	EGA7	637	297.2	2.74
30P35U	EGA	805	432.5	
	EGA3	817	477.9	1.49
	EGA5	820	385.2	1.86
	EGA7	817	377.2	1.49
40P50U	EGA	745	452.8	
	EGA3	745	478.5	0.00
	EGA5	762	422.6	2.28
	EGA7	763	438.0	2.41
45P50U	EGA	845	555.9	
	EGA3	849	600.0	0.59
	EGA5	864	384.8	2.25
	EGA7	858	412.5	1.54
50P55U	EGA	876	610.2	
	EGA3	891	720.2	1.71
	EGA5	889	468.2	1.48
	EGA7	888	585.0	1.37

1. EGA is the genetic algorithm solutions to the original problem.
2. EGAK is the genetic algorithm solutions when capacity is reduced by k.
3. $GAP = \frac{EGAk - EGA}{EGA}$.

Table 2.9: Performance of EGA with New Mutation Strategy

Problem	Method	Cost	Gap (%)	Time (sec.)
10P12U	EGA	336	0.00	147.2
	EGA1(50-25)	336	0.00	612.4
	EGA1(80-10)	336	0.00	157.0
	EGA1(80-20)	336	0.00	297.5
	OPT	336	–	0.7
12P15U	EGA	401	1.01	128.7
	EGA1(50-25)	401	1.01	612.4
	EGA1(80-10)	401	1.01	134.2
	EGA1(80-20)	401	1.01	297.5
	OPT	397	–	1.1
15P18U	EGA	507	0.00	256.0
	EGA1(50-25)	507	0.00	612.4
	EGA1(80-10)	507	0.00	267.9
	EGA1(80-20)	507	0.00	543.2
	OPT	507	–	0.9
18P20U	EGA	514	0.00	277.3
	EGA1(50-25)	514	0.00	612.4
	EGA1(80-10)	514	0.00	334.3
	EGA1(80-20)	514	0.00	564.5
	OPT	514	–	6.6
20P25U	EGA	551	0.92	208.2
	EGA1(50-25)	551	0.92	612.4
	EGA1(80-10)	551	0.92	240.9
	EGA1(80-20)	548	0.36	564.5
	OPT	546	–	4.4
25P30U	EGA	620	1.47	323.6
	EGA1(50-25)	625	2.29	612.4
	EGA1(80-10)	612	0.16	780.2
	EGA1(80-20)	614	0.49	1076.4
	OPT	611	–	69.1
30P35U	EGA	805	2.16	432.5
	EGA1(50-25)	799	1.40	745.6
	EGA1(80-10)	801	1.65	846.9
	EGA1(80-20)	796	1.02	1076.4
	OPT	788	–	3604.1
40P50U	EGA	745	5.08	452.8
	EGA1(50-25)	743	4.80	723.5
	EGA1(80-10)	735	3.67	923.5
	EGA1(80-20)	740	4.37	1123.4
	OPT	709	–	5045.2
45P50U	EGA	845	5.10	555.9
	EGA1(50-25)	846	5.22	976.5
	EGA1(80-10)	843	4.85	1007.0
	EGA1(80-20)	844	4.98	1076.4
	OPT	804	–	6195.7
50P55U	EGA	876	4.66	610.2
	EGA1(50-25)	876	4.66	845.1
	EGA1(80-10)	872	4.18	921.2
	EGA1(80-20)	874	4.42	1076.4
	OPT	837	–	36606.3

For example, EGA1(50-25) has a population size of 50 and 25 individuals are selected in the new mutation strategy.

Chapter 3

Controlled Tabular Adjustment

3.1 Overview

When government agencies and commercial organizations publish tabular data, they must withhold certain data elements that contain confidential information associated with the data respondents. In a table, a cell is considered sensitive if the publication of the true cell value will disclose a contributor's data to the public. For example, in an economic survey, if a cell contains data from one respondent, then publication of the cell value would disclose confidential data pertaining to that respondent. This would breach the pledge of confidentiality made to the company by the agency collecting the data. Disclosure methods are then used to protect the sensitive cells. On the other hand, statistical agencies try to publish as much information as possible. This results into a trade-off between privacy rights and information loss, an issue of primary importance in practice.

Procedures to protect sensitive cells in tabular data have been developed over the last four decades. From the very beginning, national statistical agencies realized that simply withholding the values of sensitive cells was insufficient to protect sensitive information in tables containing marginal totals. Complementary cell suppression [10, 12] was introduced and practiced by statistical agencies to protect sensitive cells from disclosure through manipulation of additive relationships in statistical

Table 3.1: Example to illustrate complementary cell suppression

Activity	Region			Total
	A	B	C	
I	74	17	85	176
II	71	51	30	152
III	1	9	36	46
Total	146	77	151	374

(a) Original table

Activity	Region			Total
	A	B	C	
I	X	X	85	176
II	71	51	30	152
III	X	X	36	46
Total	146	77	151	374

(b) Published table

tables. Complementary cell suppression is aimed at assuring that exact interval estimates (lower and upper bounds) of the value of each suppressed sensitive cell are at a safe distance from the actual cell value, that is, they lie within an interval at least as broad as that defined by predetermined protection limits [13]. Cox [11] provides a theory and algorithms for computing protection limits.

Cell suppression is a widely used technique for data disclosure. We present a small example from Cox [14] as an introduction. In Table 3.1a, we show the original data corresponding to the investment of enterprises (per millions of guilders), classified by activity and region. Assume that the information in the cells corresponding to activity I and region B, activity III and region A, activity III and region B are considered confidential. These cells are sensitive cells to be suppressed. However, primary suppression is not enough. By using the marginal totals, we can easily recompute the missing values of the sensitive cells. Other table entries must be suppressed (this is secondary or complementary suppression). For example, with the missing entries in Table 3.1b, the values of the sensitive cells are not disclosed exactly, although we can still compute a range of the values for these cells that are consistent with the published entries.

Complementary cell suppression (CCS) [10] has been used to protect sensitive

cells from disclosure through the manipulation of additive relationships in tables. CCS suppresses both sensitive and nonsensitive cells to protect the confidentiality of the sensitive cells. As a result, it produces a table (such as Table 3.1b) with missing data and therefore the data in the table are difficult to analyze. Though suppressed entries can be replaced by interval estimates of their hidden values, interval data present analytical challenges and destroy additivity to totals. CCS is an NP-hard problem [32]. With CCS, it is possible to estimate expected values of the suppressed entries, and often these estimates are close to the original values.

Controlled tabular adjustment (CTA) was introduced by Dandekar and Cox [18]. CTA introduces controlled perturbations (adjustments) into tabular data that satisfy the protection ranges and tabular constraints (additivity) while minimizing data loss as measured by one of several linear cell value adjustments. A tabular data protection range assures that all released tabular cells satisfy an appropriate disclosure rule. CTA replaces each sensitive cell by either of the two endpoints of its protection range. These values are sometimes referred to as the minimally safe values. Selected nonsensitive cell values are then adjusted from their true values by small amounts to restore additivity. Additionally, nonsensitive cell perturbations are constrained to be small or insignificant, such as limiting them to be within a certain percentage of the sample variance. CTA satisfies the protection ranges and tabular constraints (additivity) while minimizing data loss as measured by one of several linear cell value adjustments.

The objective in generating synthetic tabular data is to closely mimic the original tabular data, subject to obscuring sensitive cell values to a sufficient extent.

Table 3.2: Example to illustrate controlled tabular adjustment

Activity	Region			Total
	A	B	C	
I	74	17[0,37]	85	176
II	71	51	30	152
III	1[0,21]	9[0,29]	36	46
Total	146	77	151	374

(a) Original table

Activity	Region			Total
	A	B	C	
I	74	0	85	159
II	71	51	30	152
III	0	29	36	65
Total	145	80	151	376

(b) Published table after CTA

The underlying concept is simple: The value of each sensitive cell is replaced by a synthetic value selected to be at a safe distance away from the true cell value. As a starting point, we set this distance to be either the sensitive cell’s lower protection limit or its upper protection limit. Some or all of the nonsensitive cell values are then adjusted from their true values by as small an amount as possible to restore additivity to totals within the tabular system.

The result of CTA is a tabular system without suppressions meeting the disclosure rule which is optimally close to the original system with respect to a distortion measure. Thus, CTA provides a safe, completely populated, and fully analyzable tabular system. An additional advantage is that the sensitive cells are not highlighted as they might be in CCS. In Tables 3.2a and 3.2b [14] we give the original data and the perturbed data from CTA. Sensitive cells are bold in Table 3.2a, with the predefined upper limits and the lower limits. Table 3.2b shows the perturbation after applying CTA.

In this chapter, we develop a two-phase algorithm that combines an Elite Genetic Algorithm (EGA) with a linear program (LP) (we denote our algorithm by EGA-CTA) to protect sensitive cells. We have exactly two options to perturb each sensitive cell. It is natural to use a binary variable to denote the choice of

perturbation, with 1 denoting way up and 0 denoting way down. It is well known that a Genetic Algorithm (GA) can be directly applied to binary vectors without any transformation. We generate binary vectors with length equal to the number of sensitive cells, which denotes the combination of perturbations on the sensitive cells. We then use a GA to find high-quality perturbations. We use an Elite Genetic Algorithm for quick convergence. EGA keeps the current best solution in every generation and exempts it from crossover and mutation. EGA sets values in sensitive cells and the LP sets values in nonsensitive cells for each generation. In a small number of generations and a reasonable amount of CPU time, EGA-CTA produces high-quality solutions.

In Section 3.2, we outline EGA-CTA and present a small example. In Section 3.3, we conduct experiments with four variants of EGA-CTA and then compare the results of the four variants to the results generated by exact and approximate methods including simple rounding. In Section 3.4, we give our conclusions.

3.2 Description of EGA-CTA

In CTA, the value of each sensitive cell is replaced by an adjusted value selected to be a safe distance from the original value. Some or all nonsensitive cell values are then adjusted from their true values by small amounts to restore additivity within the tabular system. There are lower and upper bounds on the adjustments of nonsensitive cells, i.e., the perturbations on nonsensitive cells cannot fall out of their lower and upper bounds.

EGA-CTA has two phases: Elite Genetic Algorithm and Linear Program. The EGA selection is after the LP phase, since we need the solution to the LP to determine the next generation. The solution to the LP is the fitness in the EGA, where we select the next generation based on the fitness of the chromosome.

The maximum number of generations in EGA is 50. EGA stops when the best solution stays the same for five generations. In EGA, we always keep the best solution of the current generation, and exempt it from crossover and mutation.

3.2.1 Details of EGA-CTA

Now we describe EGA-CTA in more detail.

Step 0. Initialization. POP is the population size in EGA and p is the number of sensitive cells. We randomly generate POP chromosomes (individuals) of length p , denoted as a binary matrix B . Therefore, B has POP rows and p columns, where each row is a chromosome representing a combination of perturbations on the sensitive cells. The value 1 denotes the selection of the upper bound on the sensitive cell, and 0 denotes the lower bound. The matrix B looks like

$$B = \begin{pmatrix} 1 & 1 & 0 & \cdots & 1 \\ 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & \cdots & 0 \end{pmatrix}. \quad (3.1)$$

Step 1. Evolution. Continue until the number of generations reaches Max step = 50, or the current best solution stays the same for up to five iterations.

Phase 1. Elite Genetic Algorithm. We use EGA, which keeps the best current

chromosome from crossover and mutation.

Crossover. With probability $PC = 0.95$, we use two-point crossover for each parent pair. For each parent pair, we randomly select two different positions on the chromosome and swap the genes.

For example, for the chromosome of length six in (3.2), we randomly select 2 and 4 as the crossover range, and swap the genes from the second to the fourth positions inclusive.

$$\begin{pmatrix} 0 & \underline{1} & \underline{1} & \underline{0} & 0 & 1 \\ 1 & \underline{0} & \underline{1} & \underline{1} & 1 & 0 \end{pmatrix} \implies \begin{pmatrix} 0 & \underline{0} & \underline{1} & \underline{1} & 0 & 1 \\ 1 & \underline{1} & \underline{1} & \underline{0} & 1 & 0 \end{pmatrix} \quad (3.2)$$

Mutation. We keep the best chromosome and mutate genes of each chromosome with probability PM , that is, we switch the gene from 1 to 0 or from 0 to 1 with probability PM . For example, in (3.3), we switch gene 2 from 0 to 1,

$$\begin{pmatrix} 0 & \underline{0} & 1 & 1 & 0 & 1 \end{pmatrix} \implies \begin{pmatrix} 0 & \underline{1} & 1 & 1 & 0 & 1 \end{pmatrix}, \quad (3.3)$$

where the underlined gene is the one that has been mutated. The mutation probability is small, such as 0.05 (later on we point out that a large probability does not necessarily improve the results significantly). After Phase 1, sensitive cells are set at their safe values, either at the lower bound or the upper bound.

Phase 2. Linear Program and EGA Selection. We introduce the following notation.

(m, n)	Dimension of input table
b_{ij}	Binary variable on perturbation direction of sensitive cell (i, j)
LP_{ij}, UP_{ij}	Lower, upper protection on sensitive cell (i, j)
x_{ij}^+, x_{ij}^-	Positive, negative adjustment on cell (i, j)
LB_{ij}, UB_{ij}	Lower, upper bound of change on nonsensitive cell (i, j)
c_{ij}	Cost per unit change on cell (i, j)

Sensitive cell values are forced to be integer through constraints that contain binary variables, For two-dimensional tables, the coefficient matrix is unimodular when the sensitive cells are integer and consequently nonsensitive entries are automatically assigned to integer values. Solutions for three-dimensional and other tables can exhibit fractional entries in nonsensitive cells. We round a fractional internal entry to the nearest integer and recompute totals. Therefore, we formulate a linear program (LP), solve the LP, and round any fractional values. The mathematical formulation of the linear program follows.

$$\text{minimize } \sum_{j=1}^n \sum_{i=1}^m c_{ij} (x_{ij}^+ + x_{ij}^-) \quad (3.4)$$

s.t.

$$\begin{aligned} \sum_{j=1}^{n-1} (x_{ij}^+ - x_{ij}^-) &= (x_{in}^+ - x_{in}^-) \quad \forall i = 1, \dots, m \\ \sum_{i=1}^{m-1} (x_{ij}^+ - x_{ij}^-) &= (x_{mj}^+ - x_{mj}^-) \quad \forall j = 1, \dots, n \end{aligned} \quad (3.5)$$

$$\begin{cases} 0 \leq x_{ij}^+ \leq UB_{ij} \\ 0 \leq x_{ij}^- \leq LB_{ij} \end{cases} \quad \text{for nonsensitive cells} \quad (3.6)$$

$$\begin{cases} x_{ij}^+ = UP_{ij} * b_{ij} \\ x_{ij}^- = LP_{ij} * (1 - b_{ij}) \end{cases} \quad \text{for sensitive cells} \quad (3.7)$$

$$x_{ij}^+, x_{ij}^- \geq 0 \quad \forall i, j \quad (3.8)$$

We consider two commonly used cost functions, which are usually defined over the perturbation variables x_{ij}^+ and x_{ij}^- . The first cost vector is $c_{ij} = 1$ for each cell. This corresponds to minimizing the total absolute adjustment. The second cost vector is $c_{ij} = 1/(\text{cell value})$. This corresponds to minimizing total percent absolute adjustment. Throughout this chapter, we focus on absolute adjustment.

Constraint (3.5) maintains tabular consistency, that is, the total adjustments on internal cells should be equal to the adjustment on the associated marginal

cell. Constraint (3.6) is used to control the nonsensitive cell deviations, that is, the perturbations on the nonsensitive cells cannot exceed their given bounds. For example, the bounds on the nonsensitive cells can be no more than 20% percent of their original values. Constraint (3.7) ensures that the sensitive cells are set at their safe values, either at the lower bound or the upper bound, while this perturbation is determined by EGA in Phase 1.

The solution to the LP is continuous and nonnegative, denoted by $x = [x^+, x^-]$. After solving the LP, the tabular cell value is $\bar{t}_{ij} = t_{ij} + x_{ij}^+ - x_{ij}^-$, where t_{ij} is the original cell value.

It is possible that the solution generated by CTA is infeasible when the number of sensitive cells in a particular row or column is large. Perturbations on this row or column may exceed the perturbation bound on the corresponding marginal cell. To avoid infeasibility, we relax the equality constraints and have the following inequalities:

$$\begin{cases} x_{ij}^+ \leq UP_{ij} * b_{ij} \\ x_{ij}^- \leq LP_{ij} * (1 - b_{ij}) \end{cases} \quad \text{for sensitive cells} \quad (3.9)$$

or

$$\begin{cases} UP_{ij} * b_{ij} \leq x_{ij}^+ \leq UB_{ij} * b_{ij} \\ LP_{ij} * (1 - b_{ij}) \leq x_{ij}^- \leq LB_{ij} * (1 - b_{ij}) \end{cases} \quad \text{for sensitive cells.} \quad (3.10)$$

Constraint (3.9) allows potential perturbations up or down on the sensitive cells. Constraint (3.10) forces the LP to perturb the sensitive cells away from the protection range, that is, it either perturbs the sensitive cell to a value that is bigger than its upper protection limit, or smaller than its lower protection limit.

If (3.9) or (3.10) are used, we need to force the sensitive cells back to their

original safe values after solving the LP, by replacing the sensitive cell (i, j) by its upper limit if b_{ij} is 1, and replacing it by its lower limit otherwise.

EGA Selection. The reason why we separate EGA selection from crossover and mutation is that we need the objective function in the LP to determine the next generation. We determine the next generation by selecting from both the parents and the children in proportion to their fitness, i.e., the probability we select pop_i is $P(pop_i) = fit(i) / (\sum_{i=1}^{POP} fit(i))$, where $fit(i)$ is the reciprocal of the objective function value of chromosome i . Since we are minimizing cost in the IP, the smaller the objective function value of the chromosome, the more likely it will survive to the next generation.

Step 2. Iteration. Go back to Step 1 for the next generation, and stop when the best current solution remains the same for five generations, that is, we only allow up to five non-productive generations.

We illustrate our algorithm in detail using the example from Section 3.1. Table 3.3a is the original table, where cells (I,B), (III,A), and (III, B) shown in bold have been identified as sensitive cells and the associated protection limits are shown in the brackets. There are 13 nonsensitive cells (marginal cells are treated as nonsensitive cells). The upper and lower bounds on the nonsensitive cells are set at 20% of the original cell values, while sensitive cells are set at either their lower or upper bounds.

After applying EGA-CTA, we find the solution in Table 3.3b. In Table 3.3b, the cells marked with an asterisk (*) are perturbed. We record the positive and negative adjustments in Table 3.3c. The cost of the total absolute adjustment is 80, which is the sum of absolute values of the adjustments in Table 3.3c. The solution

Table 3.3: Example to Illustrate EGA-CTA

Region				
Activity	A	B	C	Total
I	74	17[0,37]	85	176
II	71	51	30	152
III	1[0,21]	9[0,29]	36	46
Total	146	77	151	374

(a) Original table

Region				
Activity	A	B	C	Total
I	74	0*	85	159*
II	71	51	30	152
III	0*	29*	36	65*
Total	145*	80*	151	376*

(b) EGA-CTA solution

Region				
Activity	A	B	C	Total
I	-	-17	-	-17
II	-	-	-	-
III	-1	+20	-	+19
Total	-1	+3	-	+2

(c) Perturbations

in Table 3.3b is optimal, which is verified by solving the mixed integer program defined by

$$\text{minimize } \sum_{j=1}^n \sum_{i=1}^m c_{ij}(x_{ij}^+ + x_{ij}^-) \quad (3.11)$$

s.t.

$$\begin{aligned} \sum_{j=1}^{n-1} (x_{ij}^+ - x_{ij}^-) &= (x_{in}^+ - x_{in}^-) \quad \forall i = 1, \dots, m \\ \sum_{i=1}^{m-1} (x_{ij}^+ - x_{ij}^-) &= (x_{mj}^+ - x_{mj}^-) \quad \forall j = 1, \dots, n \end{aligned} \quad (3.12)$$

$$\begin{cases} 0 \leq x_{ij}^+ \leq UB_{ij} \\ 0 \leq x_{ij}^- \leq LB_{ij} \end{cases} \quad \text{for nonsensitive cells} \quad (3.13)$$

$$\begin{cases} x_{ij}^+ = UP_{ij} * y_{ij} \\ x_{ij}^- = LP_{ij} * (1 - y_{ij}) \end{cases} \quad \text{for sensitive cells} \quad (3.14)$$

$$x_{ij}^+, x_{ij}^- \text{ are nonnegative integers, } y_{ij} \in \{0, 1\} \quad \forall i, j \quad (3.15)$$

Now we illustrate how EGA-CTA generates a solution.

Step 0. Initialization. Since this is a small problem, we set the population size $POP = 5$ in EGA, that is, there are five chromosomes in each generation. There are three sensitive cells in the table, so we have $p = 3$. A binary matrix B is randomly generated, where each row represents a chromosome which determines

the perturbations on sensitive cells. For this example, we have

$$B = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}. \quad (3.16)$$

Step 1. Evolution.

Phase 1. Elite Genetic Algorithm.

Two-Point Crossover. We keep the current best chromosome and cross over the parent pairs. We use two-point crossover to swap the underlined genes shown below.

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & \underline{1} & \underline{0} \\ 0 & \underline{0} & \underline{1} \\ \underline{1} & \underline{0} & 1 \\ \underline{1} & \underline{1} & 0 \end{pmatrix} \implies \begin{pmatrix} 0 & 0 & 0 \\ 1 & \underline{0} & \underline{1} \\ 0 & \underline{1} & \underline{0} \\ \underline{1} & \underline{1} & 1 \\ \underline{1} & \underline{0} & 0 \end{pmatrix} \quad (3.17)$$

Point-Wise Mutation. We keep the best chromosome and mutate each gene on each chromosome with probability 0.05.

After mutation, we have the matrix \overline{B} ,

$$\overline{B} = \begin{pmatrix} 0 & 0 & 0 \\ \underline{0} & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}. \quad (3.18)$$

Phase 2. Linear Program. We solve our LP to generate five tables corresponding to the five chromosomes. These solutions are given in Tables 3.4 to 3.8.

EGA Selection. We select the next generation in proportion to each chromosome's fitness, where the fitness is the reciprocal of the objective function value, namely, the absolute value of the total perturbations. The new generation of chromosomes selected is B_{new} given by

Table 3.4: Solution 1 after mutation

74	0*	85	159*
71	51	30	152
0*	0*	36	36*
145*	51*	151	347*

Based on Chromosome (0 0 0)
Adjustment = 108

Table 3.5: Solution 2 after mutation

74	0*	85	159*
71	51	30	152
0*	29*	36	65*
145*	80*	151	376*

Based on Chromosome (0 0 1)
Adjustment = 80

Table 3.6: Solution 3 after mutation

74	0*	85	159*
71	51	30	152
21*	0*	36	57
166*	51*	151	368*

Based on Chromosome (0 1 0)
Adjustment = 126

Table 3.7: Solution 4 after mutation

74	37*	85	196*
71	51	30	152
21*	29*	36	86*
166*	117*	151	434*

Based on Chromosome (1 1 1)
Adjustment = 240

Table 3.8: Solution 5 after mutation

74	37*	85	196*
71	51	30	152
0*	0*	36	36*
145*	88*	151	384*

Based on Chromosome (1 0 0)
Adjustment = 82

$$B_{new} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (3.19)$$

The solutions from the new generation are given in Tables 3.9 to 3.13. As the probability of surviving is in proportion to each chromosome's fitness, the solutions in four tables (Tables 3.9, 3.11, 3.12, 3.13) are better than the corresponding tables (Tables 3.4, 3.6, 3.7, 3.8) before selection. Since the example is small, we find the optimal solution after the first generation.

3.3 Computational Experiments

We coded our algorithm in MATLAB. We used a machine with a 1.0GHz Pentium-III processor and 384MB RAM. For uniformity, the maximum number of generations is 50 and the size of the population (POP) is 51. All results are averaged from 10 independent runs to reduce the variance. We terminate the algorithm if the

Table 3.9: Solution 1 after selection

74	0*	85	159*
71	51	30	152
0*	29*	36	65*
145*	80*	151	376*

Based on Chromosome (0 0 1)
Adjustment = 80(Optimal)

Table 3.10: Solution 2 after selection

74	0*	85	159*
71	51	30	152
0*	29*	36	65*
145*	80*	151	376*

Based on Chromosome (0 0 1)
Adjustment = 80(Optimal)

Table 3.11: Solution 3 after selection

74	37*	85	196*
71	51	30	152
0*	0*	36	36*
145*	88*	151	384*

Based on Chromosome (1 0 0)
Adjustment = 82

Table 3.12: Solution 4 after selection

74	37*	85	196*
71	51	30	152
0*	0*	36	36*
145*	88*	151	384*

Based on Chromosome (1 0 0)
Adjustment = 82

Table 3.13: Solution 5 after selection

74	0*	85	159*
71	51	30	152
0*	29*	36	65*
145*	80*	151	376*

Based on Chromosome (0 0 1)
Adjustment = 80

best current solution remains the same for up to five generations.

3.3.1 Selecting the Probability of Mutation

Since crossover plays an important role in a genetic algorithm, we set the probability of crossover to be 0.98. Traditionally, mutation plays a minor role and its probability can be as small as 0.05. To verify this, we conduct an experiment with a 3-dimensional table ($6 \times 10 \times 4$) from Cox et al. [14]. This table contains 191 non-zeros cells, of which 24 cells are sensitive cells. We drop three sensitive marginal cells in the original table, as we consider only sensitive internal cells. We assume symmetric protection, that is, $LP_i = UP_i$, and the upper and lower bounds for the nonsensitive cells are set at 20% of the original cell values.

Results for different mutation probabilities on the $6 \times 10 \times 4$ table are shown in Figures 6.1 to 3.6. We used six different population sizes, where, in each figure, the probability of mutation (PM) is 0.05, 0.25, 0.5, 0.75 and 0.95. For each PM, we

average the objective values over 10 independent replications.

The six figures show the relationship between the probability of mutation and the objective, where we use different sizes of the population. We did not observe the objective value decreasing as the probability of mutation increases. Therefore, we select a small value for the probability of mutation (0.05).

3.3.2 EGA-CTA vs. Exact Method

We generate eight problems with sizes ranging from 64 to 625; four are 2-dimensional tables and four are 3-dimensional tables. All eight problems contain 30% sensitive cells, where the perturbation bounds on both sensitive cells and non-sensitive cells are 20% of the original cell values. All sensitive cells are internal cells. EGA-CTA recalculates the marginal cells after rounding the LP solution, and this may cause the violation of the sensitive constraint on the marginal sensitive cells. We also include the $6 \times 10 \times 4$ table and drop marginal sensitive cells.

We compare the results produced by four variants of EGA-CTA. We then compare the results of the best EGA-CTA variant with an Exact Method (EM). We now provide notation.

a. Two traditional EGA-CTAs

1. T-EGA(SPM). EGA with Traditional Two-Point Crossover and Single-Point Mutation (SPM). In SPM, we mutate one randomly selected gene on each individual (chromosome).

2. T-EGA(PWM). EGA with Traditional Two-Point Crossover and Point-

Figure 3.1: PM vs Objective (POP=5)

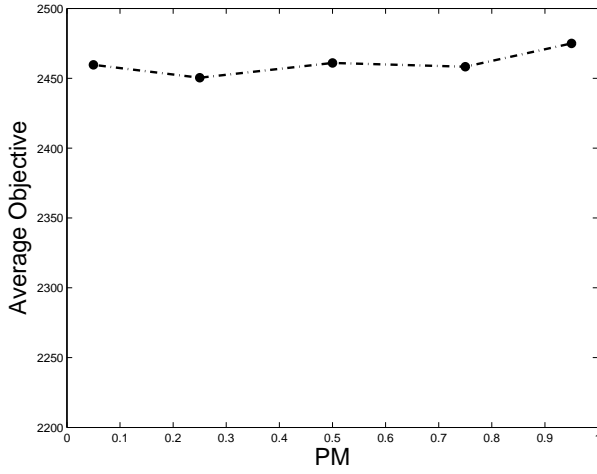


Figure 3.2: PM vs Objective (POP=11)

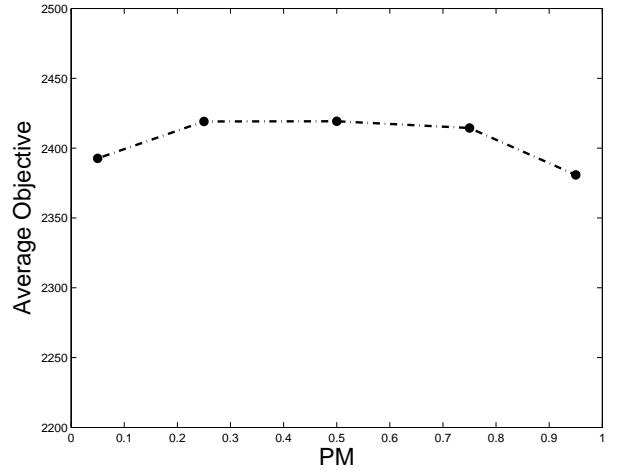


Figure 3.3: PM vs Objective (POP=21)

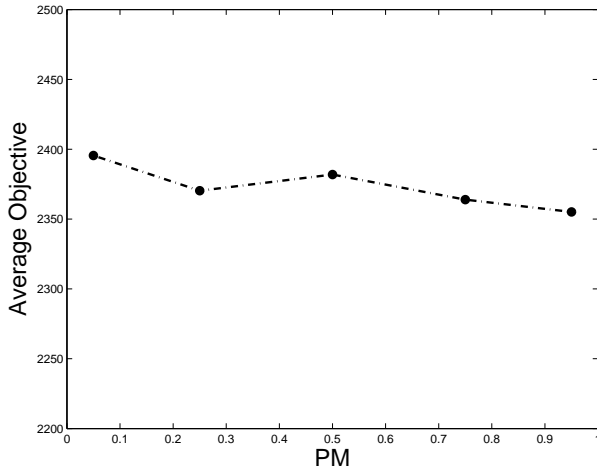


Figure 3.4: PM vs Objective (POP=31)

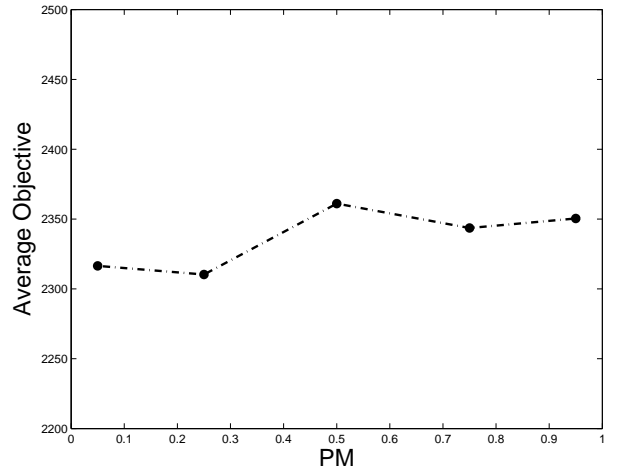


Figure 3.5: PM vs Objective (POP=41)

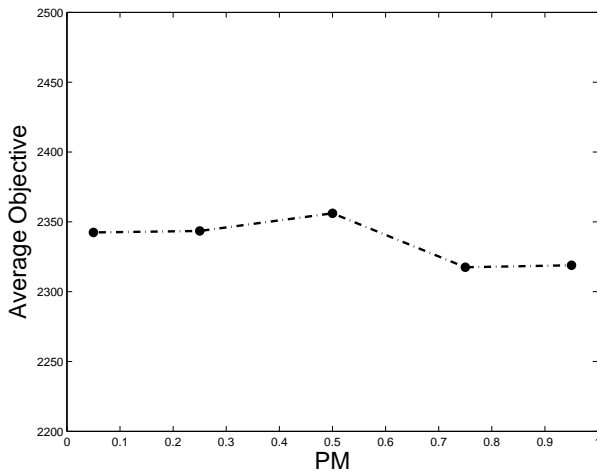


Figure 3.6: PM vs Objective (POP=51)

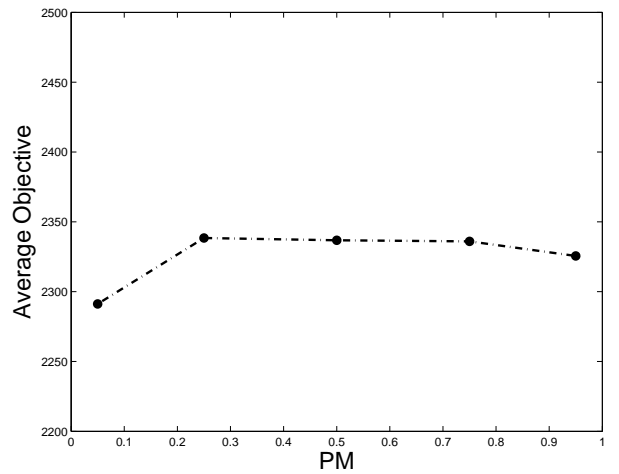


Table 3.14: Results produced by four variants of EGA-CTA on nine problems

Problem	T-EGA (SPM)	T-EGA (PWM)	QB-EGA (SPM)	QB-EGA (PWM)
10 × 10	3760.4	3855.0	3654.6	3517.8
15 × 15	8127.2	8185.0	7983.2	7648.0
20 × 20	14266.0	14532.0	13774.0	13467.0
25 × 25	24277.0	24409.0	23458.0	23133.0
4 × 4 × 4	4460.4	4470.6	4320.4	4272.4
5 × 5 × 3	4413.0	4361.8	4200.0	4119.6
4 × 4 × 7	6717.6	6641.6	6415.2	6259.0
5 × 5 × 7	9990.4	10238.0	9414.6	9136.6
6 × 10 × 4	2291.2	2274.6	2204.6	2164.2

-Bold indicates best solution

Wise Mutation (PWM). In PWM, we mutate all genes on each individual (chromosome).

Selection. We keep the current best solution, and select 50 individuals from 51 parents and 50 children in proportion to their fitnesses.

b. Two Queen-Bee EGAs

1. QB-EGA(SPM). EGA with Queen-Bee Selection in Crossover and Single-Point Mutation

2. QB-EGA(PWM). EGA with Queen-Bee Selection in Crossover and Point-Wise Mutation

Queen-Bee Selection in Crossover. We select the current best individual as the queen, and use proportional selection to select $POP/2$ number of individuals as bees to mate with the queen.

We use the same initialization, population size, probabilities of crossover and mutation, and selection mechanism in our experiments with the four variants of EGA-CTA.

In Table 3.14, we give the results of the four variants on nine problems. QB-EGA (PWM) produces the best solutions to all nine problems. Queen-Bee selection

in crossover performs better than the traditional two-point crossover, if we use the same mutation. This is due to that fact that Queen-Bee selection in the crossover tends to generate better-quality children. Furthermore, QB-EGA (PWM) generates better results than those by QB-EGA (SPM). The explanation is that point-wise mutation tends to provide more variety during the evolution. The running time of each variant on the same problem is comparable and ranges from a few seconds on the small problems to four hundred seconds on the large problems.

Next, we compare the results of QB-EGA(PWM) to the results produced by an Exact Method. In the Exact Method (EM), we set the perturbations on sensitive cells as binary variables, denoted by y_{ij} , and perturbations on nonsensitive cells as integer variables, denoted by x_{ij} . We formulate the problem as a Mixed Integer Programming (MIP), solve the MIP using Xpress, and compare the optimal solutions to the solutions generated by QB-EGA (PWM).

The MIP is given by

$$\text{minimize } \sum_{j=1}^n \sum_{i=1}^m c_{ij}(x_{ij}^+ + x_{ij}^-) \quad (3.20)$$

s.t.

$$\begin{aligned} \sum_{j=1}^{n-1} (x_{ij}^+ - x_{ij}^-) &= (x_{in}^+ - x_{in}^-) \quad \forall i = 1, \dots, m \\ \sum_{i=1}^{m-1} (x_{ij}^+ - x_{ij}^-) &= (x_{mj}^+ - x_{mj}^-) \quad \forall j = 1, \dots, n \end{aligned} \quad (3.21)$$

$$\begin{cases} 0 \leq x_{ij}^+ \leq UB_{ij} \\ 0 \leq x_{ij}^- \leq LB_{ij} \end{cases} \quad \text{for nonsensitive cells} \quad (3.22)$$

$$\begin{cases} x_{ij}^+ = UP_{ij} * y_{ij} \\ x_{ij}^- = LP_{ij} * (1 - y_{ij}) \end{cases} \quad \text{for sensitive cells,} \quad (3.23)$$

$$x_{ij}^+, x_{ij}^- \text{ are nonnegative integers, } y_{ij} \in \{0, 1\} \quad \forall i, j \quad (3.24)$$

In Table 3.15, we show the results for QB-EGA (PWM) and EM on 15 prob-

Table 3.15: Results produced by QB-EGA (PWM) and EM on 15 problems with 10% sensitive cells

Problem	EM	Time of EM (s)	QB-EGA (PWM)	Time of QB-EGA (PWM) (s)	$\frac{\text{QB-EGA(PWM)} - \text{EM}}{\text{EM}}$
10×10	2042	0.2	2042	2.1	0.00
15×15	3162	2.8	3162	8.2	0.00
20×20	5184	7.0	5230	22.6	0.89
25×25	8104	528.6	8250	43.1	1.80
30×30	NA	NA	11792	66.3	NA
40×40	NA	NA	20774	183.6	NA
50×50	NA	NA	31752	367.0	NA
$4 \times 4 \times 4$	1826	0.1	1826	2.7	0.00
$5 \times 5 \times 3$	3204	0.2	3204	3.7	0.00
$4 \times 4 \times 7$	3522	0.6	3522	6.1	0.00
$5 \times 5 \times 5$	3718	0.4	3718	8.9	0.00
$5 \times 5 \times 7$	4164	1.2	4164	65.2	0.00
$10 \times 10 \times 3$	7676	56.8	7786	98.3	1.43
$10 \times 10 \times 4$	8452	118.9	8620	156.1	1.99
$10 \times 10 \times 5$	9888	494.9	10012	170.2	1.25

lems with 10% sensitive cells. We record the solution values and running times (in seconds) for EM and QB-EGA (PWM). EM's running time increases quickly as the problem size increases, while QB-EGA (PWM)'s running time increases relatively slowly. EM is unable to generate the optimal solution for a couple of days on three problems (30×30 , 40×40 , and 50×50), while QB-EGA (PWM) generates a solution to all 15 problems in no more than six minutes. The gap between the optimal solution and the solution generated by QB-EGA (PWM) is less than 2% on the 12 problems solved to optimality by EM. QB-EGA (PWM) solves seven problems optimally.

In Table 3.16, we show the results for QB-EGA (PWM) and EM on 15 problems with 30% sensitive cells. We record the solution values and running times (in seconds) for EM and QB-EGA (PWM). EM's running time increases quickly as the problem size increases, while QB-EGA (PWM)'s running time increases relatively slowly. EM is unable to generate the optimal solution for a couple of days on seven problems (20×20 , 25×25 , 30×30 , 40×40 , 50×50 , $10 \times 10 \times 4$, and

Table 3.16: Results produced by QB-EGA (PWM) and EM on 15 problems with 30% sensitive cells

Problem	EM	Time of EM (s)	QB-EGA (PWM)	Time of QB-EGA (PWM) (s)	$\frac{\text{QB-EGA(PWM)} - \text{EM}}{\text{EM}}$
10 × 10	3266	0.63	3322	5.6	1.71
15 × 15	6894	142.5	7180	11.2	4.15
20 × 20	NA	NA	12760	16.7	NA
25 × 25	NA	NA	22360	23.8	NA
30 × 30	NA	NA	31178	40.4	NA
40 × 40	NA	NA	52808	107.1	NA
50 × 50	NA	NA	30590	214.5	NA
4 × 4 × 4	2616	0.05	2616	2.8	0.00
5 × 5 × 3	2668	2.3	2668	3.4	0.00
4 × 4 × 7	3312	0.1	3312	7.9	0.00
5 × 5 × 5	7472	2.5	7472	17.5	0.00
5 × 5 × 7	6356	3.4	6556	30.8	3.15
10 × 10 × 3	15194	1788.6	15526	100.9	2.18
10 × 10 × 4	NA	NA	22358	145.9	NA
10 × 10 × 5	NA	NA	26321	210.7	NA

10 × 10 × 5), while QB-EGA (PWM) generates a solution to all 15 problems in four minutes or less. The gap between the optimal solution and the solution generated by QB-EGA (PWM) is less than 4.15% on the eight problems solved to optimality by EM. QB-EGA (PWM) solves four problems optimally.

3.3.3 QB-EGA (PWM) vs. Hybrid Heuristic

For the three problems in Table 3.15 (30 × 30, 40 × 40, and 50 × 50) and the seven problems in Table 3.16 (20 × 20, 25 × 25, 30 × 30, 40 × 40, 50 × 50, 10 × 10 × 4, and 10 × 10 × 5) that cannot be solved to optimality within a reasonable amount of time, we apply the Hybrid Heuristic (HH) from Cox et al. [14] and compare the results produced by HH to the results produced by QB-EGA (PWM).

Outline of Hybrid Heuristic. Cox et al. [14] find that in good solutions to the CTA problem, approximately half of the sensitive cells are set to their high values and the remaining cells are set to their low values. Ordering the sensitive cells and

alternately setting them to their minimally low protection value or high protection value tends to produce a solution whose absolute total perturbation, and hence the overall mean, remains nearly unchanged.

Cox et al. [14] group the sensitive cells and assign a unique binary variable to the entire group, with the result that all cells in a group are adjusted in the same direction.

Specifically, let $M \geq 2$ be the number of groups. Cox et al. [14] add the following constraints to the MIP defined by 3.20 to 3.24.

For $i = 1$ to $M : B_i = B_{i+M} = B_{i+2M} = \dots = B_{i+kM}$, where $i + kM \leq p$, and p is the number of sensitive cells. B_i corresponds to the ordered sensitive cell i , that is, $B_i = 1$ if cell i is perturbed to its upper limit, and $B_i = 0$ if cell i is perturbed to its lower limit. Thus, all cells in group i are adjusted in the same direction.

Cox et al. [14] suggest a modified order heuristic, where sensitive cells are ordered from largest to smallest, and groups are created by skipping through the ordering, that is, assigning the ordered cells $i, i + M, i + 2M, \dots, i + kM$ to the same group, where $i + kM \leq p$. In other words, we try to mix large-valued cells with small-valued cells. This ensures greater group-to-group homogeneity so that large cells are less likely to be adjusted predominantly in the same direction. This tends to lower the overall perturbations as much as possible.

If $M = p$, then the solution produced by the MIP is optimal. If $M \leq 20$, the mathematical program can be solved in a reasonable amount of computing time [14].

In Table 3.17, we show the results produced by QB-EGA (PWM) and HH for

Table 3.17: Results produced by QB-EGA (PWM) and HH on three problems with 10% sensitive cells

Problem	HH	Time of HH (s)	QB-EGA (PWM)	Time of QB-EGA (PWM) (s)	$\frac{HH - QB-EGA(PWM)}{HH}$
30 × 30	11964	47.7	11792	41.7	1.44
40 × 40	21086	199.6	20774	99.5	1.48
50 × 50	33702	170.7	31752	164.5	5.79

Table 3.18: Results produced by QB-EGA (PWM) and HH on seven problems with 30% sensitive cells

Problem	HH	Time of HH (s)	QB-EGA (PWM)	Time of QB-EGA (PWM) (s)	$\frac{HH - QB-EGA(PWM)}{HH}$
20 × 20	13206	23.5	12760	16.7	3.38
25 × 25	23088	32.7	22360	23.8	3.15
30 × 30	32188	60.9	31178	40.4	3.14
40 × 40	54320	146.8	52808	107.1	2.78
50 × 50	33144	269.5	30590	214.5	7.71
10 × 10 × 4	23274	137.2	22358	145.9	3.94
10 × 10 × 5	28660	198.5	26321	210.7	8.16

three problems with 10% sensitive cells. For all three problems, QB-EGA (PWM) produced a better solution than HH and was slightly faster computationally. The solutions produced by HH were about 3% larger on average than the solutions produced by QB-EGA (PWM).

In Table 3.18, we show the results produced by QB-EGA (PWM) and HH for seven problems with 30% sensitive cells. For all seven problems, QB-EGA (PWM) produced a better solution than HH and was slightly faster computationally. The solutions produced by HH were about 4.6% larger on average than the solutions produced by QB-EGA (PWM).

3.4 Conclusions

Based on the results of our computational experiments, Queen-Bee EGA with Point-Wise Mutation produced optimal or near-optimal solutions on medium-size

problems in reasonable running times. QB-EGA (PWM) is easy to code and quick in convergence.

In these experiments, we treated marginal cells as nonsensitive, due to the fact that, in EGA-CTA, we recalculated the marginal cells after rounding the LP solution. This may cause the violation of the sensitive constraint on marginal sensitive cells. In future work, we will try to extend EGA-CTA to handle row totals and column totals as sensitive cells.

Chapter 4

Two-stage Transportation Problem

4.1 Overview

Supply chain management (SCM) is the term used to describe the management of materials and information across the entire supply chain, from suppliers to component producers to final assemblers to distribution (warehouses and retailers), and ultimately to the consumer [43].

Typically, supply chain management looks at the design of transportation networks locations, plants, and distribution centers (DCs), streamlining production schedules, and improving order response time. Transportation network design is one of the most important areas of SCM. It offers great potential to reduce costs and improve service quality. The transportation problem (TP) is a well-known basic network problem that was originally proposed by Hitchcock [31]. The objective is to find the best way of transporting product from several sources to several destinations so that the total cost is minimized. For real-world applications, the transportation problem can be extended to satisfy additional constraints such as two-stage distribution.

Logistics is defined as the art of bringing the right amount of the right product to the right place at the right time and usually refers to supply chain problems [48]. The efficiency of the system is influenced by many factors, such as the number of

DCs. We want to find a good location to open, so that demand can be satisfied at minimum opening cost and minimum shipping cost.

In this chapter, we consider an extension of the two-stage transportation problem (denoted by TsTP). In the TsTP, we try to determine the transportation network that satisfies demand at minimum cost subject to the plant and DCs capacities and the maximum number of DCs that can be opened. Most companies have only limited resources to open and operate DCs. Limiting the number of DCs that can be located is important when a manager has limited available capital. For this reason, the maximum number of DCs that can be opened is considered as a constraint. The TsTP combines the transportation problem with the facility location problem. In this chapter, the demands are known with certainty.

Geoffrion et al. [26] were the first researchers to study the TsTP. They used Benders decomposition to solve the problem. Hindi et al. [30] added two constraints to the problem. First, each customer must be served with all products it requires from a single distribution center. Second, it must be possible to ascertain the plant origin of each delivery. The authors developed a mathematical model for this problem and applied a branch-and-bound algorithm. Lower bounds were calculated through a series of structural transformations and descent search was employed to generate good customer-DC assignments and upper bounds. Amiri [2] worked on a distribution network design problem in supply chain system involving the location of plants and warehouses. The author tried to determine the best strategy for distributing the product from plants to warehouses and from warehouses to customers. Multiple levels of capacities for the plants and warehouses were consid-

ered. A mixed integer programming model and a Lagrangean-based heuristic were used to solve this problem. Computational tests indicated that the procedure was both effective and efficient for a wide variety of problem sizes and structures.

Michalewicz et al. [37] were the first researchers to use a genetic algorithm (GA) for solving linear and nonlinear transportation problems. They used a matrix representation in a chromosome, a two dimensional structure for representing a chromosome, and matrix-based crossover and mutation. A matrix $V = (x_{ij})$ is a chromosome, where x_{ij} is the amount delivered from source i to destination j . Arithmetical crossover was used in the GA. Starting with two parents (matrices U and V), the arithmetical crossover produced two children $X = c_1 \cdot V$ and $Y = c_1 \cdot V + c_2 \cdot U$, where $c_1, c_2 \geq 0$ and $c_1 + c_2 = 1$. The mutation operator selected a submatrix W from a parent matrix X , and reassigned flows on the edges in W only. The results demonstrated the efficiency of the GA in finding the global optimum.

Syarif et al. [46] considered a production and distribution problem modeled by the TsTP and proposed a hybrid genetic algorithm to solve it. While a solution at each stage was represented by a Prüfer number (a Prüfer number is very suitable for encoding a spanning tree, such as transportation problems and minimum spanning tree problems [54]), a hybrid approach was used to enhance the performance of the GA. Double Prüfer numbers were used to represent the candidate solution to the problem [46]. The authors showed that the proposed method worked very well and also gave better performance with respect to computational time and memory usage.

Gen et al. [29] developed a priority-based Genetic Algorithm (pb-GA) with decoding and encoding procedures, and proposed a crossover operator known as

Weight Mapping Crossover (WMX). An experimental study was carried out in two-stages. While the effect of WMX on the performance of pb-GA was investigated in the first stage, pb-GA and another GA approach based on a different representation method were compared on solution quality and solution time in the second stage.

In this chapter, we develop a two-phase algorithm that combines an Elite Genetic Algorithm (EGA) with a linear program (LP) (we denote our algorithm by TsTP-EGA). In the first phase, we determine the edges from plants to DCs and the edges from DCs to customers. In the second phase, there are two steps. In the first step, we determine the flows from DCs to customers by solving the corresponding LP model. In the second step, we consider DCs as the customers, and the total flow from each DC as the demand of the customer. We then assign the quantities to be delivered from the plants to the DCs by solving the corresponding LP model.

In Section 4.2, we describe our mathematical model of the two-stage transportation problem and a priority-based genetic algorithm for solving it. In Section 4.3, we develop a genetic algorithm to solve the TsTP. We apply our genetic algorithm to a set of test problems and compare our results to pb-GA. In Section 4.4, we give our conclusions.

4.2 Problem Description

In the TsTP, we have a set of plants ($i = 1, 2, \dots, I$), a set of distribution centers ($j = 1, 2, \dots, J$), and a set of customers ($k = 1, 2, \dots, K$). We want to select a subset of plants and a subset of DCs, and determine the flows such that all demands of the

customers are satisfied, the capacities of the plants and the DCs are not exceeded, and the fixed cost and the delivery cost are minimized. We assume the demands of customers, the capacities of the plants, and the capacities of the DCs are fixed and known.

We introduce the following notation before describing the mathematical model.

Inputs

I	Number of plants ($i = 1, 2, \dots, I$)
J	Number of distribution centers ($j = 1, 2, \dots, J$)
K	Number of customers ($k = 1, 2, \dots, K$)
a_i	Capacity of plant i
b_j	Capacity of distribution center j
d_k	Demand of customer k
t_{ij}	Cost to ship a unit from plant i to distribution center j
c_{jk}	Cost to ship a unit from distribution center j to customer k
g_j	Fixed cost for operating distribution center j
W	Upper limit on number of DCs that can be opened

Decision variables

x_{ij}	Amount shipped from plant i to distribution center j
y_{jk}	Amount shipped from distribution center j to customer k
z_j	1 if distribution center j is opened and 0 otherwise

The mathematical model is given by the following.

$$\text{Minimize } \sum_{i=1}^I \sum_{j=1}^J t_{ij} x_{ij} + \sum_{j=1}^J \sum_{k=1}^K c_{jk} y_{jk} + \sum_{j=1}^J g_j z_j \quad (4.1)$$

s.t.

$$\sum_{j=1}^J x_{ij} \leq a_i, \forall i \quad (\text{plant capacity}) \quad (4.2)$$

$$\sum_{k=1}^K y_{jk} \leq b_j z_j, \forall j \quad (\text{distribution center capacity}) \quad (4.3)$$

$$\sum_{j=1}^J z_j \leq W, \quad (\text{number of distribution centers}) \quad (4.4)$$

$$\sum_{j=1}^J y_{jk} \geq d_k, \forall k \quad (\text{demand}) \quad (4.5)$$

$$\sum_{i=1}^I \sum_{j=1}^J x_{ij} = \sum_{j=1}^J \sum_{k=1}^K y_{jk} \quad (\text{flow balance}) \quad (4.6)$$

$$x_{ij}, y_{jk} \geq 0, \quad \forall i, j, k \quad (4.7)$$

$$z_j \in \{0, 1\}, \quad \forall j \quad (4.8)$$

Constraints (D.2) and (D.3) are the plant capacity constraints and the distribution center capacity constraints, respectively. Constraint (D.4) restricts the maximum number of DCs that can be opened. Constraint (D.5) is the demand constraint. Constraint (D.5) is the flow balance, that is, the total flow from plants to DCs equals the total flow from DCs to customers. Constraints (D.5) and (D.5) are the nonnegativity restrictions and binary restrictions on the decision variables, respectively. Without loss of generality, we assume that our model satisfies the balance condition, since an unbalanced problem can be changed to balanced one by introducing dummy suppliers or dummy customers.

4.2.1 Outline of pb-GA

Gen et al. [29] developed a priority-based Genetic Algorithm (pb-GA) with decoding and encoding procedures, and used Weight Mapping Crossover (WMX). An experimental study was carried out in two stages. While the effect of WMX on the performance of pb-GA was investigated in the first stage, pb-GA and another GA approach based on a different representation method were compared on solution quality and solution time in the second stage.

In pb-GA, a chromosome consisted of priorities of sources and depots to obtain a transportation tree with length equal to total number of sources ($|K|$) and depots ($|J|$), i.e. $|K| + |J|$. At each step, only one arc is added to the tree by selecting a source (depot) with the highest priority and connecting it to a depot (source) considering minimum cost.

Gen et al. used two priority-based encodings to represent the transportation trees on stages. Each chromosome in the population had two parts. While the first part (i.e., the first priority-based encoding) represented a transportation tree between plants and DCs, the second part (i.e., the second priority-based encoding) represented a transportation tree between DCs and customers.

In Figure 4.1, we show a chromosome in pb-GA with v_1 and v_2 . v_1 contains seven digits that represent the priorities of the nodes in the first stage, and v_2 consists of nine digits that represent the priorities of the nodes in the second stage.

To obtain priority-based encoding for any transportation tree, a cheapest insertion-based encoding algorithm was used. Starting with the second part, the

Figure 4.1: A chromosome in pb-GA

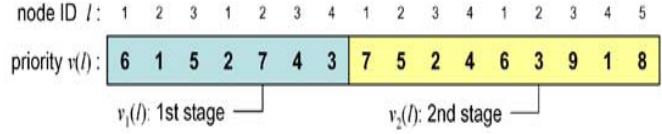
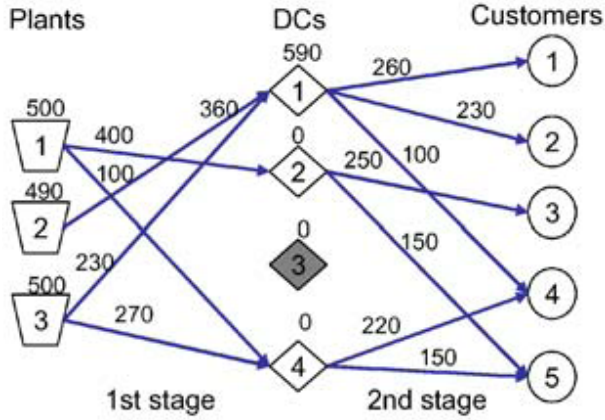


Figure 4.2: The transportation tree after encoding



authors selected one node with the highest priority and used the encoding algorithm to add an edge out of or in to the node at each iteration, until all demands had been met. In Figure 4.2, we show the transportation tree after encoding.

The crossover operator can be viewed as an extension of one-point crossover for permutation encoding. As in one-point crossover, after determining a random cut-point, Gen et al. generated the offspring by using the left segment of the cut-point and carrying out remapping on the right segment of its own parent. In the remapping process, after obtaining an increasing order of digits on the right segments of parents and mapping digits on the ordered parts, a new right segment of the first offspring was obtained using the original sequence of right segment of the second parent and its mapped digits on the first parent. When obtaining new right segment of second parent, the original sequence of right segment of the first parent and its

Figure 4.3: An example of a TsTP with three plants, four DCs, and five customers

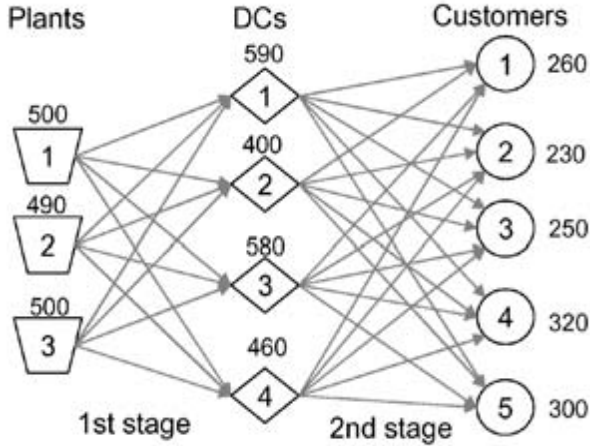


Table 4.1: Distance matrices with three plants, four DCs, and five customers

T_{ij}	1	2	3	4
1	34	29	38	29
2	23	34	30	27
3	35	31	32	30

(a) Distance matrix T

C_{ij}	1	2	3	4	5
1	15	18	16	19	21
2	17	20	12	18	15
3	25	23	15	21	19
4	20	16	13	14	18

(b) Distance matrix C

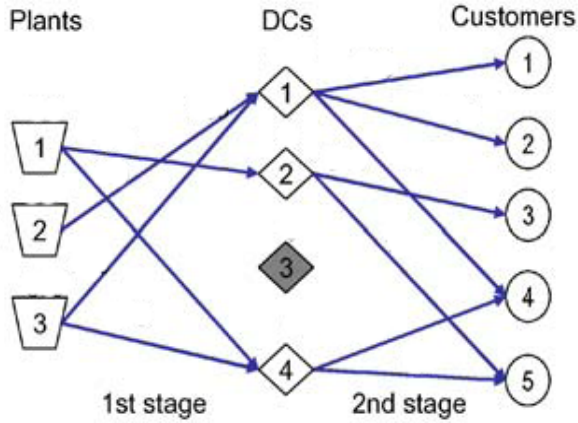
mapped digits on the second parent were used.

4.2.2 Outline of TsTP-EGA

We use a small example to describe our TsTP-EGA. Suppose we are given a set of three plants, a set of four DCs, and a set of five customers. The maximum number of DCs that can be opened is four. The demands and capacities are given in Figure 6.1. The distance matrices are given in Table 4.1a and 4.1b.

The chromosome consists of two parts. The first part is a 3×4 binary matrix

Figure 4.4: Open edges based on chromosome in (4.9)



that corresponds to the edges from plants to DCs. The second part is a 4×5 binary matrix that corresponds to the edges from DCs to customers. We use the transpose of the second part in the chromosome so that the number of columns in both parts is the same, which is the number of DCs. Therefore, each column represents flows in and out of a DC. The first part of the column represents the edges in to the DC, and the second part of the column represents the edges out of the DC.

$$P = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix} \quad (4.9)$$

The chromosome in (4.9) corresponds to the open edges in Figure 6.2.

Step 0. Initialization. We set the population size to $POP = 20$ and generate 20 chromosomes.

To generate a chromosome, we start with the second part, that is, we first determine the edges from DCs to customers.

(a) Stage 2 assignment (from DCs to customers)

We randomly select one customer at a time and use the greedy method to meet its demand. The greedy method is similar to the greedy method given in Chapter 2. For example, we select customer k^* , and order the delivery costs per unit, namely, $\frac{z_{k^*} + c_{jk^*} b_j}{b_j}$, from smallest to largest, for all DCs $j = 1, 2, \dots, J$. Suppose DC j_0 has the smallest delivery cost per unit. Then we assign a quantity equal to $\min(D_{k^*}, b_{i_0})$ to this DC, and reduce D_{k^*} and b_{i_0} accordingly. Once the demand of customer k^* has been met, we continue to randomly select the remaining customers, until all demands have been met.

After applying the greedy method, the assignment from DCs to customers is given in (4.10), which corresponds to the transpose of the second part in (4.9). In Figure D.1, we show the flows from DCs to customers. We point out that DC_3 is not open.

$$\left(\begin{array}{c|ccccc} & C_1 & C_2 & C_3 & C_4 & C_5 \\ \hline DC_1 & 260 & 230 & 0 & 100 & 0 \\ DC_2 & 0 & 0 & 250 & 0 & 150 \\ DC_3 & 0 & 0 & 0 & 0 & 0 \\ DC_4 & 10 & 0 & 0 & 220 & 150 \end{array} \right) \quad (4.10)$$

(b) Stage 1 assignment (from plants to DCs)

Now we use the same greedy method as in (a) by considering plants as DCs, and DCs as customers. The demands of the DCs are the flows out of the DCs assigned in (a), which are (590, 400, 0, 370). The flows after applying the greedy method are given in (4.11) and the graph are shown in Figure D.2.

Figure 4.5: Flows from DCs to customers after Step 1

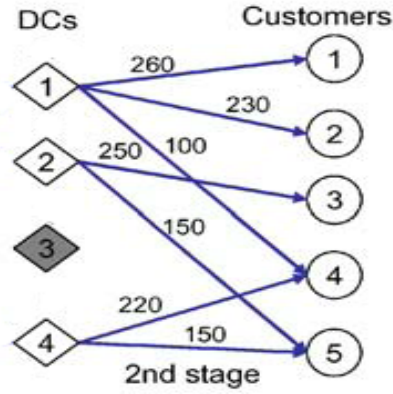
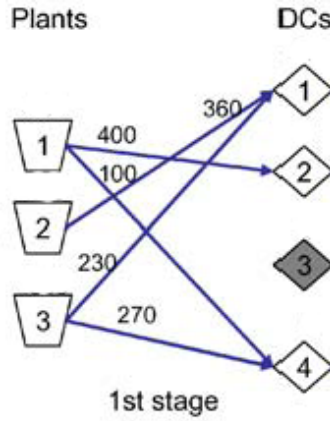


Figure 4.6: Flows from plants to DCs after Step 2



$$\left(\begin{array}{c|cccc} & DC_1 & DC_2 & DC_3 & DC_4 \\ \hline P_1 & 0 & 400 & 0 & 100 \\ P_2 & 360 & 0 & 0 & 0 \\ P_3 & 230 & 0 & 0 & 270 \end{array} \right) \quad (4.11)$$

In order to apply a genetic algorithm, we encode the feasible solution into a chromosome. Let P_2 be a 4×5 binary matrix, where $P_2^{jk} = 1$ if the flow from DC_j to customer k is positive, and 0 otherwise. Let P_1 be a 3×4 binary matrix, where $P_1^{ij} = 1$ if there exists a positive flow from plant i to customer k , and 0 otherwise.

Then $P = (P_1; P_2')$ is the corresponding chromosome, which consists of P_1 and the transpose of P_2 .

After initialization, we have the shipments from plants to depots as well as the shipments from depots to customers, and the chromosomes are binary matrices. All initial chromosomes can produce feasible solutions in our LP, given by

$$\text{Minimize } Z = \sum_{i=1}^I \sum_{j=1}^J t_{ij} x_{ij} + \sum_{j=1}^J \sum_{k=1}^K c_{jk} y_{jk} \quad (4.12)$$

s.t.

$$\sum_{j=1}^J x_{ij} \leq a_i, \forall i \quad (4.13)$$

$$\sum_{k=1}^K y_{jk} \leq b_j, \forall j \quad (4.14)$$

$$\sum_{j=1}^J y_{jk} \geq d_k, \forall k \quad (4.15)$$

$$\sum_{i=1}^I \sum_{j=1}^J x_{ij} = \sum_{j=1}^J \sum_{k=1}^K y_{jk} \quad (4.16)$$

$$0 \leq x_{ij} \leq a_i \cdot P_1^{ij}, \quad \forall i, j \quad (4.17)$$

$$0 \leq y_{jk} \leq b_j \cdot P_2^{jk}, \quad \forall j, k \quad (4.18)$$

Constraints (D.5) and (4.18) restrict the flows on the edges determined by the chromosome.

Step 1. Evolution.

Step 1.1. Queen-Bee Crossover. We use the current best chromosome as the queen and select 10 chromosomes in proportion to their fitness to mate with the queen, where fitness is the reciprocal of cost. We take the union of the two chromosomes as a base for the new chromosome. We open the route only if it is

either open in the queen or in the (regular) bee, and close the route otherwise. We use the same greedy method as in the initialization phase on these open routes to select a subset of these routes. The chromosome that corresponds to this subset of routes is the new chromosome after crossover.

Suppose we have the queen given in (4.19) and the bee given in (4.20).

$$\text{Queen} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \quad (4.19)$$

$$\text{Bee} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \quad (4.20)$$

We close all routes that are not in the queen or the bee. This produces the following intermediate matrix.

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \quad (4.21)$$

After applying the greedy method using only the routes or edges that correspond to the nonzero entries in (4.21), we generate the chromosome in (4.22). Note that (4.22) has fewer nonzero entries than (4.21). For example, of the two possible routes from plant 1 in (4.21), only one is open in the greedy solution (4.22).

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \quad (4.22)$$

The assignments given by the LP solution are shown in (4.23).

$$\left(\begin{array}{c|cccc} & DC_1 & DC_2 & DC_3 & DC_4 \\ \hline P_1 & 500 & 0 & 0 & 0 \\ P_2 & 70 & 0 & 0 & 420 \\ P_3 & 0 & 330 & 0 & 40 \end{array} \right) \left(\begin{array}{c|ccccc} & C_1 & C_2 & C_3 & C_4 & C_5 \\ \hline DC_1 & 0 & 0 & 250 & 320 & 0 \\ DC_2 & 0 & 230 & 0 & 0 & 100 \\ DC_3 & 0 & 0 & 0 & 0 & 0 \\ DC_4 & 260 & 0 & 0 & 0 & 200 \end{array} \right) \quad (4.23)$$

Step 1.1. Mutation. We apply point-wise mutation with probability $PM = 0.5$ to the 10 new chromosomes after crossover. For each chromosome, we flip every gene, that is, $P_{ij} = 1 - P_{ij}$. In general, feasibility will be maintained by mutation. This is due to the fact that there are fewer nonzero entries than zeros in every chromosome, and by flipping, we generate more nonzero entries. Therefore, more edges are open. Then we assign the quantity by only considering the open edges. We use the greedy method to reassign the providers and only consider the open edges. Suppose we apply mutation to the chromosome in (4.24), and by flipping every entry of (4.24), we produce the matrix in (4.25).

$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \quad (4.24)$$

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad (4.25)$$

After applying the greedy method using only the routes or edges that correspond to the nonzero entries in (4.25), we generate the following matrix as a chromosome.

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad (4.26)$$

Using our LP, we obtain the new solution given in (4.27). Notice that DC_4 is closed.

$$\left(\begin{array}{c|cccc} & DC_1 & DC_2 & DC_3 & DC_4 \\ \hline P_1 & 60 & 310 & 0 & 0 \\ P_2 & 0 & 10 & 480 & 0 \\ P_3 & 500 & 0 & 0 & 0 \end{array} \right) \left(\begin{array}{c|ccccc} & C_1 & C_2 & C_3 & C_4 & C_5 \\ \hline DC_1 & 260 & 0 & 0 & 0 & 300 \\ DC_2 & 0 & 0 & 0 & 320 & 0 \\ DC_3 & 0 & 230 & 250 & 0 & 0 \\ DC_4 & 0 & 0 & 0 & 0 & 0 \end{array} \right) \quad (4.27)$$

We now have 10 new chromosomes after applying queen-bee crossover and 10 new chromosomes after point-wise mutation.

Step 1.3. Selection. To generate the next population, 20 chromosomes are selected from 20 children and 20 parents using proportional selection. Since we use an elite genetic algorithm, we keep the current best chromosome from crossover and mutation. The current best chromosome is updated if a new best chromosome is produced after crossover and mutation.

We stop the algorithm if the maximum step size is reached or the current best solution is the same for five generations.

4.3 Computational Results

We compare the performance of TsTP-EGA to the performance of pb-GA from Gen et al. [29]. We coded TsTP-EGA and our version of pb-GA in MATLAB and used a 3.06 GHz Pentium IV machine with 1GB RAM.

We use the four test problems given in [29]. The input data for the test problems were generated randomly. The cost coefficients for the first and second stages of the transportation problem were generated from a uniform distribution between 1 and 20. The demand requirements of customers were drawn from a uniform distribution between 5 and 60. The capacities of plants and DCs were also randomly generated considering constraints in the model. The optimal solutions were provided in [29].

In Table 4.3, we give the best, average, and worst solution values and average

Table 4.2: Data for test problems

Problem	Number of plants	Number of DCs	Number of customers	Optimal solution
$3 \times 4 \times 5$	3	4	5	1089
$4 \times 5 \times 10$	4	5	10	2283
$4 \times 5 \times 15$	4	5	15	2527
$8 \times 10 \times 20$	8	10	20	2886

Table 4.3: Performance of TsTP-EGA and pb-GA

Problem	POP	OPT	Method	Best	GAP1	Mean	GAP2	Worst	GAP3	Avg. Time (s)
$3 \times 4 \times 5$	10	1089	pb-GA	1089	0.00	1090.8	0.17	1095	0.55	0.48
			TsTP-EGA	1089	0.00	1089.0	0.00	1089	0.00	0.36
$4 \times 5 \times 10$	20	2283	pb-GA	2283	0.00	2292.5	0.42	2317	1.49	1.03
			TsTP-EGA	2298	0.66	2318.1	1.54	2342	2.58	1.43
$4 \times 5 \times 15$	30	2527	pb-GA	2527	0.00	2535.0	0.32	2543	0.63	3.32
			TsTP-EGA	2529	0.08	2550.1	0.91	2577	1.98	4.47
$8 \times 10 \times 20$	80	2886	pb-GA	2923	1.28	2936.5	1.75	2968	2.84	19.06
			TsTP-EGA	2932	1.59	3021.2	4.68	3065	6.20	17.10

$$\text{GAP1} = 100 \frac{\text{Best} - \text{OPT}}{\text{OPT}} \%$$

$$\text{GAP2} = 100 \frac{\text{Average} - \text{OPT}}{\text{OPT}} \%$$

$$\text{GAP3} = 100 \frac{\text{Worst} - \text{OPT}}{\text{OPT}} \%$$

computation times of 10 runs for both TsTP-GA and pb-GA. We also give the gap between the solution value and the optimal solution. The algorithm terminated when the number of generations reached 300 or the current best solution was the same for five generations.

The average running times of pb-GA and TsTP-EGA on each problem are comparable. TsTP-EGA found the optimal solution to problem $3 \times 4 \times 5$ in all 10 runs, while pb-GA did not. However, as the problem sizes increase, the performance of TsTP-EGA with respect to solution quality is worse than the performance of pb-GA.

For pb-GA, the gap between the best solution value and the optimal solution

is between 0.00% and 1.28%. The gap between the average solution value and the optimal solution is between 0.17% and 1.75%. The gap between the worst solution value and the optimal solution is between 0.55% and 2.84%.

For TsTP-EGA, the gap between the best solution value and the optimal solution is between 0.00% and 1.59%. The gap between the average solution value and the optimal solution is between 0.00% and 4.68%. The gap between the worst solution value and the optimal solution is between 0.00% and 6.20%.

The structure of TsTP-GA is much simpler than the structure of pb-GA. The best solution values of TsTP-GA and pb-GA are very close. Intelligent operators could be used to improve the performance of TsTP-GA.

4.4 Conclusions

We developed a TsTP-EGA to solve the two-stage transportation problem. TsTP-EGA has two stages. In the first stage, we determine the shipments from plants to distribution centers (DCs). In the second stage, we determine the shipments from DCs to customers. The algorithm starts with the shipment assignments in stage 2. We use a genetic algorithm to determine the routes or edges from DCs to customers and assign quantities using an LP model. After stage 1, we apply the same routine, by considering DCs as the customers and plants as the DCs. The demands of the DCs are the shipments from DCs determined in the stage 2 assignment. TsTP-EGA is modified from the EGA given in Chapter 2, which is designed to solve the concave cost supply scheduling problem. We used the same queen-

bee selection crossover and point-wise mutation in genetic algorithm, and therefore the algorithm is very easy to implement. Our computational results show that the TsTP-EGA is simple in structure, quick in convergence, and yields good solutions. More importantly, our work demonstrates that GAs can be applied successfully to a variety of supply scheduling problems.

Chapter 5

Generalized Orienteering Problem

5.1 Overview

5.1.1 Literature review

The orienteering problem has been studied extensively in the operations research literature since the early 1980s. In the orienteering problem (OP), we are given a network in which a start point and an end point are specified, and points have associated scores. Given a fixed amount of time, the goal is to determine a path from the start point to the end point through a subset of points in order to maximize the total score of the path. In the generalized orienteering problem (GOP), each point has a score with respect to several attributes and the overall objective function is nonlinear. The GOP is more difficult to solve than the OP, which is NP-hard [50].

Researchers have proposed a wide variety of heuristics for solving these two problems. Wang et al. [50, 51] constructed artificial neural networks and obtained high-quality solutions to the OP and GOP in a reasonable amount of computing time. Wang et al. [50] were among the first to propose a neural network algorithm for solving the orienteering problem.

Wang et al. [50] applied a modified, continuous Hopfield neural network to

solve the OP. They examined four sets of test problems. These sets involve 32, 21, 33, and 32 points and are derived from the problems studied first by Tsiligirides [49]. They tested their neural network approach on a total of 67 problems taken from the literature and compared their results to the results produced by Tsiligirides's stochastic algorithm [49] and Chao's heuristic [6]. The neural network approach was faster than Chao's heuristic and produced better results than Tsiligirides's stochastic algorithm. The neural network generated the same solution values as Chao's heuristic in 62 of 67 cases. In the five cases where it is outperformed by Chao's heuristic, the difference is small.

Wang et al. [51] used a Hopfield neural network (modified from the network in [50]) to solve the GOP. They generated a problem with 27 Chinese cities and found that a neural network was able to solve the GOP effectively.

Chao et al. [7] used a fast and effective heuristic to solve the OP. They initialized the procedure by constructing an ellipse over the entire set of points where the start and end points are the two foci of the ellipse and the maximum time limit is the length of the major axis. In generating a path, they considered only the points that are within the ellipse, since any path that contains a point outside the ellipse will violate the maximum time constraint. Chao et al. generated several paths over the points and, in the improvement step, they allowed the total score to decrease in the hope of ultimately finding a path with a large total score that is nearly optimal.

Chao et al. applied their heuristic (OH) to 67 problems taken from Tsiligirides [49] and compared their solutions with the published results. OH produced better

solutions on 45 of 67 problems when compared to the published results, and produced the same solutions on 22 of 67 problems. OH solved all 67 problems in less than 30 seconds. The authors generated 40 new test problems with geometric structures (e.g., points in a square shape) and applied OH to these new problems. Their heuristic was shown to be computationally efficient and it performed well on the new test problems. Chao et al. coded their version of Tsiligirides's stochastic algorithm (TA). For 26 square-shaped test problems, OH produced scores for seven problems that were better than the scores produced by TA, while TA produced scores for three problems that were better than the scores produced by OH. For 14 diamond-shaped problems, OH produced scores on all 14 problems that were better than the scores produced by TA. For both sets of problems, the CPU time for OH was much shorter than the time required by TA.

Tasgetiren [47] presented a genetic algorithm (GA) to solve the orienteering problem. The performance of the genetic algorithm is evaluated against heuristics specifically designed to solve the orienteering problem and an artificial neural network. A GA with an adaptive penalty function was developed to solve the orienteering problem. First, the initial population of size λ is constructed probabilistically. A list of the n points to be visited is generated in a random order. Then, each point is assigned a random number $[0,1]$. If the number is less than an insertion probability, the point is inserted into the tour; otherwise it is not included in the tour. At each generation, two parents are determined by tournament selection and random selection, to produce offspring through crossover. This process continues until μ offspring are produced. Hence, the size of the population is increased to $\lambda + \mu$

at the end of each generation. Then, local search with add, omit, replace, and swap operators is applied to individuals selected randomly from the offspring generated by the crossover operator. For the population of the next generation, tournament selection is used to establish the population, again among $\lambda + \mu$ individuals, thus maintaining a population of size λ . This procedure is repeated until a stopping criterion is met.

Tasgetiren developed an adaptive function to penalize the infeasible solutions. He used a Near-Feasibility Threshold for the constraints, that is,

$$F_p = F - (F_{all} - F_{feas}) \left(\frac{TD - D_{lim}}{NFT} \right)^\alpha,$$

where F_p is the penalized fitness value of the objective function, F is the unpenalized value of the objective function, F_{all} represents the unpenalized value of the best solution found, and F_{feas} denotes the value of the best feasible solution found. TD is the total distance of the tour and α is a predefined severity parameter (Tasgetiren set $\alpha = 4$).

NFT is the Near-Feasible Threshold for the constraint and is defined by

$$NFT = \frac{NFT_0}{1 + \lambda \times N_g},$$

where NFT_0 is the initial value of the NFT , which is set to 10% of the constraint's value. N_g represents the number of generations. Thus, solutions are adaptively penalized according to their distance from feasibility.

Tasgetiren applied his GA to benchmark test problems from [49]. He compared the results produced by GA to the results produced by Tsiligirides's stochastic algorithm (T), Chao's heuristic (C) [6, 7], and an artificial neural network (NN) [50]. GA outperformed Tsiligirides's stochastic algorithm. For 67 problems, GA

produced better scores than T in 46 of 67 problems, and produced the same score in 21 problems. GA produced better scores than NN in eight of 67 problems while NN was better on one problem, and the results were the same on 58 problems. GA produced the same results as C in 64 of 67 problems. In three problems, GA produced better results than C. GA solved all 67 test problems in less than one minute.

Tasgetiren claimed that the GA was able to find new best-known solutions to three problems. Based on the computational results, the GA performed very well across a wide variety of problem instances.

Geem et al. [24] applied Harmony Search (HS) to solve the GOP. Harmony search is based on an analogy to music improvisation where musicians improvise the pitches of their instruments to obtain better harmony. The algorithm of Geem et al. mimics three major behaviors of musicians: 1) memory consideration; 2) pitch adjustment; and 3) random choice. In memory consideration, HS chooses any one city from the cities stored in Harmony Memory. In pitch adjustment, HS chooses the nearest city as the next city to visit. In random choice, HS chooses any one city from all possible cities. The authors compared the results produced by their HS algorithm to the results produced by the NN of Wang et al. [51] on a set of 27 Chinese cities. Geem et al. claimed that HS found equal or better solutions when compared to the solutions of NN. HS found better solutions to three of five scenarios when compared to NN, and produced two solutions that were the same as the results of NN.

Liang et al. [35] developed an ant colony optimization approach to the OP.

They combined an unusual sequenced local search and a distance-based penalty function with mainstream ant colony ideas. Results on 67 test problems [49] showed that the ant colony method performed as well or better than Tsiligirides’s algorithm [49] in all cases and does so with very modest computational effort. Furthermore, the ant colony method is insensitive to starting point, problem instance, and problem size.

5.1.2 Description of the GOP

We consider the following hypothetical problem. A traveler goes to China and has time to visit several cities. The traveler has multiple goals in mind (e.g., enjoy the natural beauty of the country, visit sites of historical interest, attend major cultural events, and identify promising business opportunities). Since travel between Chinese cities is expensive and our traveler must adhere to a budget restriction, we impose a limit on the overall distance that the journey can take. Given this distance constraint, the traveler seeks to do well (in terms of score) with respect to each goal. This problem can be modeled as the generalized orienteering problem.

Potential applications of the GOP involve military scenarios. For example, consider a submarine or an unmanned aircraft that is involved in surveillance activities. The craft must visit and photograph a subset of points (each with multi-dimensional benefits) and return to its base subject to a fuel or time constraint.

The main difference between the GOP and the OP lies in the objective function. In the OP, the total score is obtained by summing the scores associated with

each point along the path from start to end. In the GOP, the total score is a more complicated (nonlinear) function of the points visited.

The motivation for the objective function is based on the following observation for our traveler in China. The Tai Mountain is quite beautiful and its score with respect to natural beauty might be eight. The Huang Mountain, however, is the most beautiful in all of China; its score might be ten. If one visits the Tai Mountain alone, a score of eight is achieved. Visiting both mountains, however, might lead to a score of less than eighteen. The GOP is a combinatorial optimization problem with multiple goals and a nonlinear objective function whose value we want to maximize.

We formulate the GOP and describe a genetic algorithm approach for solving the GOP in Section 5.2. In Section 5.3, we apply our genetic algorithm to solve a problem with 27 Chinese cities taken from [51]. We conduct an extensive computational comparison with the neural network of [51]. We present our conclusions in Section 5.4.

5.2 Problem Description

Let $G(V, E)$ be a complete graph with n points (or cities), where V is the set of points and E is the set of edges in the graph. The cost $d(i, j)$ is the Euclidean distance between points i and j . We assume that distances satisfy the triangle inequality. Both the start and end points are the same (point 1), although this need not be the case. Each point in V has a score vector $S(i) = (S_1(i), S_2(i), \dots, S_m(i))$, where m is the number of independent goals, and $S_g(i)$ is the score of point i with

respect to goal g .

Now we present the objective function. Given the dominant role played by the point with the largest score with respect to each goal, we define the total score of a path P starting at point 1 and ending at point 1 as

$$Z = \sum_{g=1}^m W_g \left\{ \max_{i \in P} (S_g(i)) \right\}, \quad (5.1)$$

where W_g is the weight that the traveler attaches to goal g .

Sine the objective function given by (5.1) is not continuously differentiable and therefore difficult to handle, Wang et al. [51] approximated Z by \bar{Z} where

$$\bar{Z} = \sum_{g=1}^m W_g \left[\left\{ \sum_{i \in P} [S_g(i)]^k \right\}^{1/k} \right]. \quad (5.2)$$

It can be shown that as k grows very large, \bar{Z} approaches Z . If $k = 1$ and $m = 1$, (5.2) is the objective function of the OP. Therefore, the OP is a special case of the GOP. Since the OP is an NP-hard problem, the GOP is at least as hard. The key constraint is given by

$$\text{Total path length} \leq D_{\text{lim}}, \quad (5.3)$$

where D_{lim} is the pre-defined distance limit. We want to maximize (5.1) subject to (5.3). In other words, the total distance of the path cannot exceed the maximum distance D_{lim} .

5.2.1 Structure of our Genetic Algorithm

Genetic algorithms make use of a vocabulary borrowed from the theory of evolution and natural selection. There is a population of individuals. These individuals are often referred to as chromosomes. Chromosomes typically contain a linear sequence of genes. Each gene represents a piece or part of a solution. An evolutionary process run on a population of chromosomes corresponds to a search over the space of potential solutions. GAs have been applied to a variety of combinatorial optimization problems including the vehicle routing problem and the traveling salesman problem [38, 52]. We now describe a GA for solving the GOP.

Input: Distance matrix, Point locations, Score vector (S), Weight vector (W), Distance limit (D_{lim}), Max step, Population size (POP), Probability of crossover (PC), Probability of mutation (PM).

Step 0: Initialize a population of paths, where each path starts and ends at point 1. Set the initial generation $i = 1$.

Step 1: For generation i , perform queen-bee crossover and two-opt mutation on all paths regardless of feasibility.

Step 2: Given the distance limit, transform each path into a feasible path and select the next generation based on proportional fitness. Increase i by one.

Step 3: Go to Step 1 and perform another iteration. Terminate the algorithm if the number of generations reaches Max step, or the current best solution stays the same for up to five iterations. Output the best solution.

Output: Best \bar{Z} -score, Best path, Total distance of the best path.

5.2.2 Algorithm Details

Now we explain the steps of our GA in detail.

Step 0. Initialization

Each chromosome is a path (tour) starting from city 1 and ending at city 1. The initial population size is POP ($POP = 51$). Each path contains all 27 cities and each city (except the first) is visited exactly once. Start with city 1. Choose the closest city with probability 0.4, the second closest with probability 0.3, and the third closest with probability 0.3. When there are two cities remaining to visit, we select the closest city with probability 0.6 and the second closest with probability 0.4. We select the last city with probability 1 and return to city 1. We do this 50 times to generate 50 initial paths. Each time, we save the (forward) tour and the reverse tour, which yields 100 tours. From each tour, we extract a feasible path by removing the fewest consecutive cities at the end of the tour (other than city 1) such that the remaining path is still feasible. (As a result, the forward and reverse tours will typically have different objective function values.) We then select the 51 feasible paths with the largest objective function values to serve as the initial generation.

Step 1. Evolution

Continue until the number of generations reaches Max step (Max step = 50), or the current best solution stays the same for up to five iterations.

Crossover

We use queen-bee selection and edge recombination crossover (ERC) [52]. The current best path is called the Queen and we select 25 paths by proportional fitness

to mate with the Queen. Suppose path I is the Queen and path J is one of the 25 paths. We now define the adjacency matrix A . The dimension of the matrix A is 27×4 . There are 27 rows that correspond to the 27 cities in China. There are four columns in A , since each city has a predecessor city and a successor city on a path, and we are counting two pairs of predecessors and successors on path I and path J . For example, in row i , the first entry is the successor city of city i on path I , the second entry is the predecessor city of city i on path I ; the third entry is the successor city of city i on path J , and the fourth entry is the predecessor city of city i on path J . With $PC = 0.95$, we generate new paths I' and J' from A in the following way.

For simplicity, consider an example with eight cities. Suppose the parent paths are $I = [1, 3, 4, 6, 7, 2, 5, 8, 1]$ and $J = [1, 4, 7, 6, 2, 3, 5, 8, 1]$. In matrix A , we have a_{i1} = successor of city i on path I , a_{i2} = predecessor of city i on path I , a_{i3} = successor of city i on path J , and a_{i4} = predecessor of city i on path J . Because a city has at most four different neighbors on the two paths, the resulting adjacency matrix A is 8×4 and is given by

$$A = \begin{pmatrix} 3 & 8 & 4 & 0 \\ 5 & 7 & 3 & 6 \\ 4 & 0 & 5 & 2 \\ 6 & 3 & 7 & 0 \\ 8 & 2 & 0 & 3 \\ 7 & 4 & 2 & 0 \\ 2 & 6 & 0 & 4 \\ 0 & 5 & 0 & 0 \end{pmatrix}. \quad (5.4)$$

The first row of A is $(3, 8, 4, 0)$, since 3 follows 1 and 8 precedes 1 on path I and 4 follows 1 and 8 precedes 1 on path J . Since 8 appears twice, we set $a_{i4} = 0$

instead of $a_{i4} = 8$. Hence, the “close” neighbors of city 1 on path I and J are $(3, 8, 4, 0)$, the “close” neighbors of city 2 are $(5, 7, 3, 6)$, and so on. The matrix A does not include city 1, since city 1 is the starting point. We begin by selecting a city to follow city 1.

To generate path I' , we start from city 1. We select a successor from the list of, at most, four close neighbors. Since the edge distances are known, we select the closest neighbor with probability one half. If this city is not selected, we select one of the remaining close neighbors with equal probability. (If there is only one neighbor, we always select it.) Suppose city 4 is selected in row one of A .

We now repeat the process from city 4. Suppose city 3 is selected as the next city to visit, and then city 5. From city 5, city 8 is chosen. Since city 8 has only city 5 as a close neighbor and it is already on our path, we randomly select a remaining city. Suppose this is city 6. From city 6, we select city 7 and then city 2. This yields $I' = [1, 4, 3, 5, 8, 6, 7, 2, 1]$. Path J' is generated in a similar way.

To avoid selecting the same city more than once, we delete a city from matrix A when it is selected. An advantage of ERC is that some of the path segments of parents are inherited by their offspring.

Each crossover operation begins with two parents and leads to two offspring. The total number of offspring computed from crossover per generation is $25 \times 2 \times 2 = 100$, since each offspring is stored in forward order and reverse order.

Two-Opt Mutation

We apply two-point mutation on every path with the probability of mutation $PM = 0.5$. We search between two cities other than city 1, and apply two-opt to

obtain an improved sequence of these cities. That is, we select two cities α and β and reverse the sequence of cities on the path from α to β in the tour. We accept the new path if the score is improved (increased). We continue two-opt until we can no longer improve the score of the path [38].

Step 2. Selection

The paths generated by crossover and mutation may not be feasible GOP paths. In particular, the total distance may exceed the maximum distance limit. In this step, we determine the feasibility of a path. If a path is infeasible, we repair it so that it becomes feasible. Selection has two parts. Given the upper bound on total distance, we first test the feasibility of each path and delete some cities if including them results in the violation of the distance constraint. Then, we select the next generation of paths based on proportional fitness.

Part I. Feasibility Check

If a path is infeasible, then we modify it to restore feasibility. Specifically, given a path of cities, we accumulate the distances between consecutive cities until the total distance exceeds the distance limit. We remove the last city (other than city 1) from the end of the path so that the remaining path is feasible. Finally, we calculate the objective function \bar{Z} of each path.

Part II. Proportional Fitness Selection

We select the next generation by proportional fitness. Each path has a fitness equal to its \bar{Z} -score (since we seek to maximize \bar{Z} -score). Therefore, the larger the \bar{Z} -score, the more likely the corresponding path is to survive to the next generation. Specifically, we have $\text{Prob}\{\text{select path } i\} = \bar{Z}_i / (\sum_{i=1}^{POP} \bar{Z}_i)$. In addition, we always

retain the best path found from one generation to the next. This makes the process an elitist genetic algorithm (EGA) [20], which ensures that the quality of the population will not degenerate from one generation to the next. In each generation, we check forward and reverse paths. Each path is available for selection according to its fitness. Since there are 100 intermediate paths from crossover, 100 intermediate paths from mutation, and 102 parent paths, we actually select 51 paths from 302 candidate paths. In particular, we retain the best path and select 50 others based on proportional fitness. We note that feasibility with respect to distance is ignored during crossover and mutation in order to promote diversity. Only during the selection step do we focus on feasibility.

In terms of parameter values, we started with $POP = 51$. In preliminary experiments in which we tried larger values, we observed that running times increased significantly, but the performance of EGA did not. PC and PM were set to common values (0.95 and 0.5) reflecting the relative importance of crossover and mutation within EGA.

5.3 Computational Experiments

We use the example of 27 cities in China for our computational experiments. In Table 5.1, we give the longitudes, latitudes, and scores for each of these cities. Each city has four scores (S_1 , S_2 , S_3 , and S_4) which represent natural beauty, historical significance, cultural and educational attractions, and business opportunities, respectively. These scores are scaled from 1 to 10. The higher the score, the more

attractive the city. The distance between cities i and j is calculated over a spherical surface and the average radius of the earth (6371 kilometers) is used in these calculations (see the Appendix for details). The distance matrix is available at http://www.rhsmith.umd.edu/faculty/bgolden/vrp_data.htm.

In this section, we compare our GA with a sophisticated artificial neural network model (ANN) [51]. The ANN is a modified, continuous Hopfield neural network that uses discrete-time dynamics and gradient descent to update intermediate solutions.

Our GA was coded in MATLAB. The ANN was coded in C. Both codes were run on a Windows XP, Pentium-III PC with 1.0 GHz speed and 384MB of memory.

We used five different values of k in our experiments. In particular,

$$\text{if } k = 1, \quad \bar{Z} = \sum_{g=1}^m W_g \left[\sum_{i \in P} [S_g(i)] \right]; \quad (5.5)$$

$$\text{if } k = 3, \quad \bar{Z} = \sum_{g=1}^m W_g \left[\left\{ \sum_{i \in P} [S_g(i)]^3 \right\}^{1/3} \right]; \quad (5.6)$$

$$\text{if } k = 4, \quad \bar{Z} = \sum_{g=1}^m W_g \left[\left\{ \sum_{i \in P} [S_g(i)]^4 \right\}^{1/4} \right]; \quad (5.7)$$

$$\text{if } k = 5, \quad \bar{Z} = \sum_{g=1}^m W_g \left[\left\{ \sum_{i \in P} [S_g(i)]^5 \right\}^{1/5} \right]; \quad (5.8)$$

$$\text{if } k = 10, \quad \bar{Z} = \sum_{g=1}^m W_g \left[\left\{ \sum_{i \in P} [S_g(i)]^{10} \right\}^{1/10} \right]. \quad (5.9)$$

Each comparison involves a combination of weight vectors and a distance limit. In each case, Beijing is both the origin and destination. The weight vectors used in our experiments are $W_0 = (0.25, 0.25, 0.25, 0.25)$, $W_1 = (1, 0, 0, 0)$, $W_2 = (0, 1, 0, 0)$, $W_3 = (0, 0, 1, 0)$, and $W_4 = (0, 0, 0, 1)$. W_0 gives equal weight to each of the four

Table 5.1: Locations and Scores of 27 Cities in China

	City	Lon.	Lat.	S_1	S_2	S_3	S_4		City	Lon.	Lat.	S_1	S_2	S_3	S_4
1	Beijing	116.40	39.91	8	10	10	7	15	Wuhan	114.30	30.55	6	6	8	6
2	Tianjin	117.18	39.16	6	5	8	8	16	Changsha	113.00	28.20	6	6	6	5
3	Jinan	117.00	36.67	7	7	5	6	17	Guangzhou	113.15	23.15	6	6	5	10
4	Qingdao	120.33	36.06	7	4	5	7	18	Haikou	110.35	20.02	7	3	4	8
5	Shijiazhuang	114.50	38.05	5	4	5	5	19	Guilin	110.29	25.28	10	4	4	4
6	Taiyuan	112.58	37.87	5	6	5	5	20	Xi'an	108.92	34.28	5	9	8	6
7	Huhehaote	111.70	40.87	6	6	5	5	21	Yinchuan	106.27	38.48	5	7	5	5
8	Zhengzhou	113.60	34.75	5	6	5	5	22	Lanzhou	103.80	36.03	7	6	5	6
9	Huangshan	118.29	29.73	9	3	2	2	23	Chengdu	104.07	30.66	6	7	6	5
10	Nanjing	118.75	32.04	7	8	8	6	24	Guiyang	106.70	26.59	8	5	4	5
11	Shanghai	121.45	31.22	5	4	9	9	25	Kunming	102.80	25.05	9	7	7	6
12	Hangzhou	120.15	30.25	9	8	7	6	26	Shenyang	123.40	41.80	5	8	5	6
13	Nanchang	115.88	28.35	7	6	5	5	27	Dalian	121.60	38.92	7	5	6	7
14	Fuzhou	119.30	26.10	6	5	5	7								

Table 5.2: EGA vs. ANN ($k = 1$)

W	D_{lim}	Algorithm	Total Distance	Score	Path	Time (sec)
W_0	5000	EGA	4987.5	99.50	1-2-5-6-8-15-13-9-11-12-10-3-4-27-26-1	32.5
		ANN	4987.5	99.50	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	61.2
W_1	5000	EGA	4962.7	105.00	1-2-27-4-10-11-12-9-14-13-16-15-8-6-5-3-1	37.7
		ANN	4811.7	105.00	1-7-6-5-3-8-15-16-13-9-12-11-10-4-27-2-1	36.0
W_2	5000	EGA	4987.5	97.00	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1	24.8
		ANN	4987.5	97.00	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	34.8
W_3	5000	EGA	4987.5	102.00	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1	34.2
		ANN	4987.5	102.00	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	40.8
W_4	5000	EGA	4953.4	96.00	1-2-3-5-6-8-15-16-13-14-9-12-11-10-4-27-1	36.9
		ANN	4953.4	96.00	1-27-4-10-11-12-9-14-13-16-15-8-6-5-3-2-1	100.2
W_0	11000	EGA	10960.5	164.75	1-2-27-26-4-3-5-6-7-21-22-20-23-25-24-19-18-17-16-13-14-9-12-11-10-15-8-1	38.1
		ANN	10516.5	164.75	1-26-27-4-10-11-12-9-14-13-15-16-17-18-19-24-25-23-20-22-21-7-6-5-8-3-2-1	70.8

goals. Weight vectors W_1 , W_2 , W_3 , and W_4 emphasize a single goal, respectively. The two distance limits are 5000 km and 11,000 km. The computational results are presented in Tables 5.2 to 5.6. Geem et al. [24] applied HS to cases where $k = 5$ and the weight vectors were W_0 , W_1 , W_2 , W_3 , and W_4 . They set the distance limit $D_{\text{lim}} = 5000$.

In Table 5.2, we provide the results for EGA and ANN on six problems with $k = 1$. The objective function is $\bar{Z} = \sum_{g=1}^m W_g [\sum_{i \in P} [S_g(i)]]$. We list the total distance, score, the best path, and the running time produced by EGA and ANN. EGA and ANN generate the same solutions on all six problems. The average running time of EGA is 34.03 seconds and the average running time of ANN is 57.3 seconds.

Table 5.3: EGA vs. ANN ($k = 3$)

W	D_{lim}	Algorithm	Total Distance	Score	Path	Time (sec)
W_0	5000	EGA	4976.7	16.58	1-2-3-5-8-20-15-16-13-9-10-12-11-4-27-1	21.2
		ANN	4987.5	16.76	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	100.8
W_1	5000	EGA	4987.7	17.95	1-2-3-4-10-11-12-9-13-16-19-24-20-6-5-1	38.2
		ANN	4987.7	17.95	1-2-3-4-10-11-12-9-13-16-19-24-20-6-5-1	51.0
W_2	5000	EGA	4987.9	17.04	1-2-3-10-12-9-13-15-8-20-22-21-7-6-5-1	24.1
		ANN	4862.7	16.87	1-2-3-10-11-12-9-13-15-8-20-22-21-6-5-1	51.0
W_3	5000	EGA	4987.5	17.45	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1	32.8
		ANN	4987.5	17.45	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	30.0
W_4	5000	EGA	4999.0	16.67	1-2-27-4-10-11-12-14-17-15-8-6-5-1	21.2
		ANN	4941.3	16.67	1-27-4-11-12-14-17-16-15-8-3-5-1	81.0
W_0	11000	EGA	10505.7	19.55	1-5-6-7-21-22-20-23-25-24-19-18-17-16-15-13-14-9-12-11-10-8-3-4-27-26-2-1	23.8
		ANN	10428.3	19.55	1-2-26-27-4-10-11-12-9-14-13-15-16-17-18-19-24-25-23-22-21-20-8-3-5-6-7-1	60.0

Table 5.4: EGA vs. ANN ($k = 4$)

W	D_{lim}	Algorithm	Total Distance	Score	Path	Time (sec)
W_0	5000	EGA	4972.8	13.66	1-2-3-5-6-20-15-13-9-12-11-10-4-27-26-1	23.4
		ANN	4901.4	13.71	1-2-3-4-10-11-12-9-13-17-19-16-15-8-5-1	70.2
W_1	5000	EGA	4999.6	14.60	1-2-3-4-10-12-9-13-16-19-24-23-6-1	24.1
		ANN	4946.0	14.69	1-2-7-4-10-12-9-13-16-19-24-8-3-1	51.0
W_2	5000	EGA	4974.6	13.96	1-2-26-27-4-10-12-9-13-16-15-20-8-3-1	24.5
		ANN	4983.5	13.87	1-2-3-8-20-15-13-9-12-11-10-4-27-26-1	34.8
W_3	5000	EGA	4996.8	14.29	1-2-27-4-10-11-12-9-13-16-15-8-20-6-5-3-1	20.7
		ANN	4987.5	14.29	1-27-4-10-11-12-9-13-16-15-8-20-6-5-3-2-1	34.8
W_4	5000	EGA	4845.2	13.78	1-2-27-4-10-11-12-14-17-16-15-3-1	24.4
		ANN	4970.8	13.78	1-27-4-10-11-12-9-14-17-16-15-3-2-1	70.8
W_0	11000	EGA	10985.8	15.31	1-2-26-27-4-9-11-12-10-15-16-13-14-17-18-19-24-25-23-22-21-20-8-3-5-6-7-1	29.1
		ANN	10428.3	15.31	1-2-26-27-4-3-8-10-11-12-9-14-13-15-16-17-18-19-24-25-23-20-22-21-7-6-5-1	60.0

In Table 5.3, we provide the results for EGA and ANN on six problems with $k = 3$. The objective function is $\bar{Z} = \sum_{g=1}^m W_g [\{\sum_{i \in P} [S_g(i)]^3\}^{1/3}]$. We list the total distance, score, the best path, and the running time produced by EGA and ANN. EGA beats ANN on one problem, ANN beats EGA on one problem, and there are four ties. The average running time of EGA is 26.88 seconds and the average running time of ANN is 62.3 seconds.

In Table 5.4, we provide the results for EGA and ANN on six problems with $k = 4$. The objective function is $\bar{Z} = \sum_{g=1}^m W_g [\{\sum_{i \in P} [S_g(i)]^4\}^{1/4}]$. We list the total distance, score, the best path, and the running time produced by EGA and ANN. EGA beats ANN on one problem, ANN beats EGA on two problems, and there

Table 5.5: EGA vs. ANN ($k = 5$)

W	D_{lim}	Algorithm	Total Distance	Score	Path	Time (sec)
W_0	5000	EGA	4833.5	12.28	1-2-3-10-15-16-19-17-13-9-12-11-4-1	32.4
		ANN	4993.4	12.38	1-2-3-10-11-12-9-13-17-19-16-20-6-5-1	61.2
		HS	4993.4	12.38	1-2-3-10-11-12-9-13-17-19-16-20-6-5-1	NA
W_1	5000	EGA	4943.9	13.08	1-2-27-4-10-12-9-13-16-19-24-3-1	21.9
		ANN	4987.7	13.05	1-2-3-4-10-11-12-9-13-16-19-24-20-6-5-1	46.2
		HS	4985.4	13.08	1-2-3-15-24-19-13-9-12-10-4-27-1	NA
W_2	5000	EGA	4875.6	12.51	1-2-3-5-6-8-20-15-9-12-10-4-27-1	22.1
		ANN	4875.1	12.51	1-2-26-27-3-10-11-12-9-13-15-20-6-5-1	40.2
		HS	4910.6	12.56	1-26-27-4-10-12-9-13-16-15-20-8-3-2-1	NA
W_3	5000	EGA	4996.8	12.78	1-2-27-4-10-11-12-9-13-16-15-8-20-6-5-3-1	29.8
		ANN	4987.5	12.78	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1	46.2
		HS	4987.5	12.78	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1	NA
W_4	5000	EGA	4954.0	12.40	1-2-27-4-10-11-12-14-17-15-8-3-1	21.1
		ANN	4989.8	12.36	1-2-3-10-9-13-16-17-14-12-11-4-27-1	90.0
		HS	4845.2	12.40	1-2-27-4-10-11-12-14-17-16-15-3-1	NA
W_0	11000	EGA	10825.2	13.35	1-2-26-27-4-10-11-12-9-14-13-16-17-18-19-24-25-23-22-21-20-15-8-3-5-6-7-1	20.9
		ANN	10428.3	13.35	1-7-6-5-3-8-20-21-22-23-25-24-19-18-17-16-15-13-14-9-12-11-10-4-27-26-2-1	100.2

are three ties. The average running time of EGA is 24.37 seconds and the average running time of ANN is 53.6 seconds.

In Table 5.5, we provide the results for EGA and ANN on six problems with $k = 5$. The objective function is $\bar{Z} = \sum_{g=1}^m W_g [\{\sum_{i \in P} [S_g(i)]^5\}^{1/5}]$. We list the total distance, score, the best path, and the running time produced by EGA and ANN. We also include the published results from Geem et al. [24]. Geem et al. did not run the problem with weight W_0 and distance limit equal to 11000, and they did not publish the running time. EGA beats ANN on two problems, ANN beats EGA on one problem, and there are three ties. HS beats EGA on two problems, and there are three ties. The average running time of EGA is 24.7 seconds and the average running time of ANN is 64.0 seconds.

In Table 5.6, we provide the results for EGA and ANN on six problems with $k = 10$. The objective function is $\bar{Z} = \sum_{g=1}^m W_g [\{\sum_{i \in P} [S_g(i)]^{10}\}^{1/10}]$. We list the total distance, score, the best path, and the running time produced by EGA and ANN. EGA beats ANN on four problems, and there are two ties. The average

Table 5.6: EGA vs. ANN ($k = 10$)

W	D_{lim}	Algorithm	Total Distance	Score	Path	Time (sec)
W_0	5000	EGA	4999.2	10.54	1-2-3-4-10-11-12-9-13-17-19-20-1	24.2
		ANN	4993.4	10.53	1-2-3-10-11-12-9-13-17-19-16-20-6-5-1	100.2
W_1	5000	EGA	4891.0	10.75	1-3-24-19-13-9-12-10-4-27-2-1	24.0
		ANN	4893.3	10.73	1-2-3-10-11-12-9-13-16-19-24-15-8-5-1	49.8
W_2	5000	EGA	4945.3	10.57	1-5-6-21-20-15-9-12-10-3-26-1	23.8
		ANN	4951.1	10.56	1-2-26-27-3-10-11-12-9-15-8-20-6-5-1	49.8
W_3	5000	EGA	4728.0	10.62	1-2-5-6-8-20-15-16-13-9-12-11-10-4-27-1	23.8
		ANN	4762.3	10.62	1-2-27-4-10-11-12-13-16-15-20-8-3-5-1	36.0
W_4	5000	EGA	4980.1	10.48	1-2-27-4-10-11-12-9-14-17-16-15-3-1	25.2
		ANN	4785.1	10.47	1-27-4-11-12-14-17-13-9-10-3-2-1	79.8
W_0	11000	EGA	10847.6	10.70	1-3-8-21-22-20-23-24-25-18-17-19-13-16-15-14-11-12-9-10-4-26-27-2-1	26.4
		ANN	10751.8	10.70	1-2-26-27-4-3-8-20-15-10-11-12-9-14-13-16-19-17-18-24-25-23-22-21-7-6-5-1	100.2

running time of EGA is 24.57 seconds and the average running time of ANN is 69.3 seconds.

In each of the five tables of computational results, there are six problems. On the 30 problems, EGA beats ANN (with respect to score) on eight problems, ANN beats EGA on four problems, and ANN and EGA are tied on 18 problems. Despite the fact that artificial neural networks are specifically designed to handle highly nonlinear data (which is what we have here) and genetic algorithms are not, EGA slightly outperformed the ANN based on score. In addition, EGA is faster than ANN. On average, EGA took 26.91 seconds and ANN took 61.3 seconds.

The impact of k on the results is clear. If $k = 1$, then the objective function is the sum of the relevant individual scores. The larger the value of k , the larger the effect of high individual scores on the overall objective function. The value of the objective function decreases to the highest score of the city on the path as k goes to infinity. Setting $k = 5$ is a reasonable compromise, in that the total score is still heavily influenced by the city with the highest score on a path, but not exclusively so.

5.4 Conclusions

In comparing EGA against ANN, we observe the following key points. First, the running times of EGA are generally smaller than those of ANN. This is despite the fact that ANN is coded in C and EGA is coded in (the much slower) MATLAB. Secondly, ANN is a code specifically designed to handle nonlinear problems (such as the GOP), whereas EGA is not. EGA is comparable to ANN with respect to solution quality and it is faster. Our results demonstrate that genetic algorithms can perform well on the GOP and should perform well on other nonlinear combinatorial optimization problems.

Chapter 6

The Shortest Pickup Planning Tour Problem with Time Windows

6.1 Overview

6.1.1 Introduction

A package delivery company needs to pick up packages from many commercial customers by the end of each day. The typical pickup starts when a driver finishes making deliveries. Drivers visit these customers everyday and, on some days, a customer has no packages for the driver. In Figure 6.1, we give an example of a pickup route. In this example, node 1 is the start depot and node 6 is the end depot. In practice, the driver usually returns to the start depot after he visits all customers.

The pickup time windows (TWs) are pre-defined in the company's contracts with the customers. Some TWs are wide, since a company does not care about the arrival times of drivers, as long as the packages are picked up. Some TWs are narrow. For example, a medical center opens its loading dock from 2:55pm to 3:05pm each day and will allow a visit only during this time window. A driver arriving before the earliest time of the TW must wait. A driver arriving after the latest time of the TW violates the contract and a penalty occurs, although he still picks up the package. In this paper, we consider soft TWs. If the driver picks up the package later than

Figure 6.1: An example of a pickup route with six customers



the time window, the company pays a certain amount of money as a penalty.

We wish to sequence the pickups in order to minimize the total cost incurred while meeting each customer's time window.

In practice, a route with only pickups has approximately 30 stops per driver. In contrast, a package delivery route has 100 to 150 stops.

There are many pickup routes each day, and there is only one driver on each route. We assume that the capacity of the truck is unlimited. We can regard each pickup route as a traveling salesman problem with time windows (TSPTW), since different pickup routes are unrelated. After an optimal or near-optimal solution is found for a route, the driver always follows the same order for the customers on the route each day.

We call this the shortest pickup tour problem with time windows and denote it by SPTP-TW. The SPTP-TW is the problem of finding a minimum cost path

that visits each city exactly once, where each city must be visited within a specific time window.

The TSPTW can be solved to optimality only if the time windows are relatively narrow [21]. Dynamic programming can solve the TSPTW with up to 50 stops to optimality. Since time windows are often wide in practice (e.g., typically half a day in grocery distribution) and finding an optimal solution with an exact procedure is nearly impossible, heuristic methods are needed.

In Section 6.2, we review the literature on the TSPTW. In Section 6.3, we describe a solution method based on dynamic programming. We provide computational results with our heuristic in Section 6.4. We present conclusions in Section 6.5.

6.2 Literature Review

Consider a network $G = (N, A)$ where $N = \{1, \dots, n\}$ is the set of nodes and A is the set of arcs. Each node $i \in N$ has been assigned a time window $[a_i, b_i]$ and a service time s_i . An arc a_{ij} joins node i and node j . Each arc $a_{ij} \in A$ has a time duration of t_{ij} and a cost c_{ij} . An arc a_{ij} is feasible if $a_i + s_i + t_{ij} \leq b_j$. In other words, a driver can visit node j after node i , if he can arrive at node j after he picks up the package at node i , and drives to node j before the latest time b_j . For example, if $a_i = 1550$, $s_i = 3$, $t_{ij} = 25$, and $b_j = 1600$, then arc a_{ij} is feasible, because $a_i + s_i + t_{ij} = 1550 + 3 + 25 = 1578$, which is less than $b_j = 1600$. A path in the network G is defined as a sequence of nodes $\{i_1, i_2, \dots, i_k\}$ such that each arc

$a_{i_j, i_{j+1}}$ belongs to A and the time service begins at node i_j , $j = 1, \dots, k$, is within the time window of that node. Let the time service begins at node i be t_i . If a driver goes from node i to node j , the time service begins at node j is given by $t_j = \max\{t_i + s_i + t_{ij}, a_j\}$ if $t_i + s_i + t_{ij} \leq b_j$, and $t_j = \infty$, otherwise. If $t_i = 1550$, $s_i = 3$, $t_{ij} = 25$, $[a_j, b_j] = [1450, 1600]$, then $t_i + s_i + t_{ij} = 1550 + 3 + 25 = 1578$, which is less than $b_j = 1600$, therefore $t_j = \max\{t_i + s_i + t_{ij}, a_j\} = \max\{1578, 1450\} = 1578$. If we change t_i to 1580 instead of 1550 and keep the other values the same, then $t_i + s_i + t_{ij} = 1580 + 3 + 25 = 1608$, which is later than b_j , then $t_j = \infty$, and arc a_{ij} is infeasible. This definition implies waiting at node j if the driver arrives there too early, and avoiding the arc a_{ij} if the driver cannot make it on time. For the optimal TSPTW solution, we require a least-cost feasible path starting at node 1 and ending at node n that visits every node in G exactly once.

Dumas et al.[21] developed three tests to eliminate some infeasible paths which greatly enhanced the performance of a dynamic programming approach to minimizing the total cost for the TSPTW. These three tests took advantage of the time window constraints to significantly reduce the state space and the number of state transitions in the dynamic programming model. These reductions were performed both prior to and during the execution of the algorithm. The authors claimed that the approach did not experience problems stemming from increasing problem size, wider or overlapping time windows, or an increasing number of states nearly as rapidly as the method of Baker [3]. (Baker used a longest path algorithm to obtain bounding information for subproblems in a branch-and-bound solution procedure. This algorithm was shown to be effective on several small size to moderate size ve-

hicle scheduling problems where a large percentage of the demand points possessed time windows. However, this algorithm did not offer improvements over existing algorithms for the general TSP.) The computational results indicated that the algorithm was successful in solving problems with up to 200 nodes and fairly wide TWs as long as three hours. When the density of the nodes in a geographical region was kept constant as the problem size increased, the algorithm was capable of solving realistic problems with up to 800 nodes. For these problems, the CPU time increased linearly with problem size.

Let $F(S, j, t)$ be the least cost of a path starting at the depot, passing through every node of S exactly once, ending at $i \in S$, and ready to serve i at time t or later. The recursive formula from Dumas et al. is given by

$$F(S, j, t) = \min_{(i,j)} \{F(S - \{j\}, i, t') + c_{ij} | t \geq t' + s_i + t_{ij}, a_j \leq b_j\}, \quad (6.1)$$

where c_{ij} is the shipping cost from node i to node j , s_i is the service time at node i , and t_{ij} is the travel time from i to j . $Esst_j$ and b_j are the earliest service start time and latest service start time, respectively, at node j .

The prior test, known as the dominance test, was used to eliminate the state (S, i, t^2) , if $t^1 \leq t^2$ and $F(S, i, t^1) \leq F(S, i, t^2)$. In other words, given the same S and ending node i , the state $(t, F(S, i, t))$ with a higher cost and a later time would be eliminated. Here the TWs are integers, and therefore the number of states are countable. After the dominance test, the two-dimensional labels $(t, F(S, i, t))$ of (S, i) can be ordered by increasing time and decreasing cost value. Let $FIRST(S, i)$ be the time value of the first label of that list, which is the earliest arrival time at i

with the highest cost associated with (S, i) . There were three post-feasibility tests.

The first test was global. It eliminated the states (S, i, t) for all t by only examining the earliest time to begin service at i . If the earliest arrival time at i is later than the latest arrival time of any other node $j \notin S$, that is, $t_i > b_j$, for any $j \notin S$, where t_i is the earliest arrival time at node i , and therefore a feasible arrival at any other node $j \notin S$ is impossible, then all the states (S, i, t) are eliminated.

The second test stated that all predecessors of node j must be visited before j . If any predecessor of node j is not in S , then (S, j, t) is eliminated.

The third test eliminated the state (S, i, t) , if (S, i, t) can be feasibly extended to $(S \cup \{i\}, j, t')$ but cannot be extended further to any other node $k \notin S$.

Computational results showed that the algorithm was successful in solving problems from [33] with up to 200 nodes and fairly wide TWs. The behavior of the algorithm is exponential with respect to the width of the time window. Nevertheless, for narrow widths, its behavior was less than exponential allowing larger size problems to be solved. For example, a 250-node problem with an average width of 20 was easy to solve in less than 10 seconds.

Based on the computational results, the authors found that the second test was the most beneficial. The first test was the least powerful because much of its work was performed by reducing the width of the time windows in the preprocessing phase.

Gendreau et al. [27] gradually built a route by inserting at each step a vertex in its neighborhood on the current route, and performing a local reoptimization. This was done when checking the feasibility of the remaining part of the route. Once

a feasible route was determined, an attempt was made to improve it by applying a post-optimization phase based on the successive removal and reinsertion of all vertices.

The authors first constructed a heuristic called GENIUS for the symmetric TSP, which consisted of a tour construction phase (GENI, for generalized insertion), followed by a post-optimization phase (US, for unstringing and stringing).

Vertices were ranked according to a criterion that measured the difficulty of insertion. Vertices were then inserted in nonincreasing order of difficulty. To make an insertion, unrouted vertices of the ordered list were considered in turn. The first vertex that could be feasibly routed was inserted in its best possible position in the path. When no unrouted vertex could be feasibly inserted in the path, a backtracking procedure was applied.

They initialized a partial path consisting of a start node and end node. The set S contained all the other nodes in nonincreasing order of the time window width. They selected one node in S at a time, in order, and inserted the node between two of its p closest neighbors using GENI, where p is an input parameter. They implemented the best insertion among all p possible insertions, and removed the node from S . Nodes were inserted until $S = \emptyset$. The backtracking procedure started when no feasible insertion could be identified. The node right after the start node was removed from the path and placed back at the end of S . If the number of times this node is removed from the path is equal to an acceptable upper limit, the algorithm terminates. The algorithm could not identify a feasible solution.

The authors applied a post-optimizer similar to US. Starting with the imme-

mediate successor of the start node, they removed, in turn, each node of the path using the reverse GENI procedure and reinserted it in the path using GENI. If this solution improved the current solution, it was accepted. Otherwise, if this solution was infeasible, it was discarded.

The algorithm was run on 375 test problems divided into four sets: (1) the individual vehicle routes produced by the VRPTW tabu search heuristic of Potvin et al. [40] on the RC2 problems proposed by Solomon [45], (2) the TSPTW instances solved optimally by Langevin et al. [33], (3) the TSPTW instances solved optimally by Dumas et al. [21], and (4) instances generated as in Dumas et al. [21], with wider time windows.

Computational results showed that on instances with narrow TWs, their heuristic was not as quick as the two exact approaches proposed in the literature [33, 21]. However, when the time windows increased, it was the only method capable of consistently producing feasible solutions with reasonable computing times. The authors attributed the success to: (1) using GENIUS to perform insertions at each step and to postprocess the solution, (2) neighborhoods based on distance and time criteria, and (3) the backtracking procedure used when no feasible insertions were possible.

Baker et al.[4] proposed a dynamic programming algorithm that solved TSPTW in time that was linear in the number of cities. This algorithm is based on the method that he proposed in 1996. A network $G^* = (V^*, A^*)$ has $n + 1$ layers of nodes, one layer for each position in the tour, with the home city (city 1) appearing at both the beginning and the end of the tour. The home city is a source node s (the only node in layer 1) and a sink node t (the only node in layer $n + 1$) of the network. The

structure of G^* creates a one-to-one correspondence between tours in G satisfying the time windows condition and $s - t$ paths in G^* . Furthermore, optimal tours in G correspond to the shortest $s - t$ paths in G^* . Every node in layer i of G^* corresponds to a state specifying which city is in position i , and which cities are visited in positions 1 through $i - 1$. The computational effort involved in solving a problem breaks up into (a) constructing G^* , and (b) finding a shortest source-sink path in G^* . Constructing G^* requires three steps: (1) identify the nodes, (2) identify the arcs, and (3) assign costs to the arcs. Here the bulk of the effort goes into (1) and (2), since the arc costs of G^* are obtained trivially from those of the original graph G .

Focacci et al. [23] proposed a hybrid approach for solving the TSPTW by first finding a feasible path and then finding the best path. The algorithm was based on a constraint programming framework, and used problem-oriented propagation algorithms and a filtering algorithm based on the calculation of bounds on the optimal solution value. This algorithm was effective for the TSPTW, and the overall approach is also effective when compared to state-of-the-art methods, such as branch-and-bound and branch-and-cut.

Mingozi et al. [39] used an exact algorithm based on dynamic programming and made use of bounding functions to reduce the state space graph. They used an algorithm for the TSP problem with time windows and precedence constraints: Before visiting vertex i , the salesman must visit every vertex of a given set of vertices. A new bounding procedure is described where forward and backward dynamic programming recursions of the reduced state space are combined in order to derive

a better reduced state space by eliminating those states that cannot lead to the optimal solution. The resulting lower bound is further improved by using Lagrangian ascent and state space decompression. The lower bounds are embedded into a dynamic programming algorithm for finding the optimal solution.

6.3 Description of Dynamic Programming

6.3.1 Introduction to Dynamic Programming

Dynamic programming (DP) is a way of solving decision problems by finding an optimal strategy. The types of problems that can be solved by DP are problems that can be broken down into a sequence of single decisions made one after the other and problems where several decisions can be separated to give this same structure (here the word “after” is being used in its sense of both time and space) [44].

The optimal solutions of subproblems can be used to find the optimal solutions of the overall problem. For example, the shortest path to a goal from a vertex in a graph can be found by first computing the shortest path to the goal from all adjacent vertices, and then using this to select the best overall path, as shown in Figure 6.2. If the shortest path from s to t passes through node p , the subpaths (s, p) and (p, t) are the shortest paths from s to p , and p to t , respectively.

Let $d(v)$ denote the length of a shortest path from s to v . If we know the lengths of the shortest paths from s to every neighbor (predecessor) of v , then we can find the length of the shortest path from s to v . The recursive formula for calculating the length of the shortest path from s to v is given by

6.3.2 Dynamic Programming in Pickup Planning Tour

Suppose we have a set of stops, denoted by V , to be visited, where the start depot and the end depot are fixed, and are not necessarily the same. The driver begins at the start depot, visits all stops, and finishes at the end depot. We assume that we have a complete graph, namely, any two stops are connected, regardless of the violation in TWs. We include a penalty function if the time window is violated. The penalty function is a linear function $p \times \max(t_i - b_i, 0)$, where p is a parameter whose value needs to be set.

We define $f(S, i)$ to be the minimum cost of beginning at the start depot, visiting all stops in S , and ending at stop i , which does not belong to S . It can be recursively defined as

$$f(S, i) = \min_{j \in S} \{f(S \setminus \{j\}, j) + c_{ji}\}, \quad (6.3)$$

where c_{ji} is the cost of traveling from stop j to stop i . This cost is a linear combination of time and mileage.

The optimal cost is then

$$f(V \setminus \{n\}, n) = \min_{k \in V} \{f(V \setminus \{n, k\}, k) + c_{kn}\}. \quad (6.4)$$

We start by calculating $f(\emptyset, i)$, the cost beginning at the start depot, going through the empty set, and terminating at stop i . We then increase the cardinality of set S by one, update the real arrival time at stop i , until we find the minimum cost, and trace back the best route.

6.3.3 Outline of Dynamic Programming

We introduce the following notation before describing the dynamic programming model.

Parameters

dp_D	Dollar per mile
dp_T	Dollar per minute
p	Penalty for violating a TW (dollars per minute)

Inputs

N	Number of stops
T_{ij}	Travel time between i and j
D_{ij}	Travel distance between i and j
a_i	Earliest service start time of i
b_i	Latest service start time of i
s_i	Service time at i

Initialization.

We calculate the initial service start time ($Resst_i$) of each stop i , if the driver visits stop i immediately after the driver departs the start depot.

$$Resst_i = \max\{a_0 + T_{0i} + s_0, a_i\}, \quad (6.5)$$

where a_0 is the earliest service start time at the start depot, T_{0i} is the travel time from the start depot to node i , and s_0 is the service time at the start depot. If the driver visits stop i immediately after he departs the start depot, then the real earliest service start time of stop i is a_0 plus the travel time from the start depot to stop i and the service time at the start depot, or the earliest service start time of stop i , whichever is the latest. In other words, the real earliest service start time of stop i is the earliest time that the driver can arrive, after he finishes the service at the start depot, but cannot be earlier than a_i , the predefined earliest service start

time at stop i . If he arrives earlier, then he has to wait until a_i . No penalty is imposed on an early arrival.

Since set S is empty, the cost of beginning at the start depot, going through empty set S , and terminating at stop j is

$$f(\emptyset, j) = dp_T \cdot (Resst_j - a_0) + dp_D \cdot D_{0j} + p \cdot \max(Resst_j - b_j, 0) \quad (6.6)$$

Iteration 1.

For $|S| = 1$, the real earliest service start time of terminal stop j is

$$Resst_j = \max\{a_j, Resst_i + T_{ij} + s_i\}, \quad (6.7)$$

where $Resst_i$ is the real earliest service start time of terminal stop i if S is empty. The terminal stop is the first stop visited after S . $Resst_i$ is from initialization, which is given in (6.5). Since stop $i \in V - \{Depot, j\}$, the number of $Resst_j$ is $N - 2$, depending on different i .

Then the cost $f(\{i\}, j)$ is given by

$$f(\{i\}, j) = f(\emptyset, i) + dp_T \cdot (Resst_j - Resst_i) + dp_D \cdot D_{ij} + p \cdot \max(Resst_j - b_j, 0). \quad (6.8)$$

For each stop j , we calculate all possible costs by considering the predecessors of j and taking the minimum cost. Suppose $f(i^*, j)$ is the minimum cost and i^* is the immediate predecessor of the route with minimum cost. We update $Resst_j$ as

$$Resst_j = \max\{a_j, Resst_{i^*} + T_{i^*j} + s_{i^*}\}. \quad (6.9)$$

Iteration k .

For $|S| = k$, calculate the real earliest service start time of terminal stop j

$$Resst_j = \max\{a_j, Resst_i + T_{ij} + s_i\}, \quad (6.10)$$

where $Resst_i$ is the real earliest service start time of terminal stop i from iteration $k - 1$.

Then $f(S, i)$, the cost of beginning at the start depot, going through the stops in S , and terminating at stop j is given by

$$f(S, i) = \min_{j \in S} \{f((S - \{j\}), j) + dp_D(Resst_i - Resst_j) + dp_T \cdot D_{ji} + p \cdot \max(Resst_i - b_i, 0)\}. \quad (6.11)$$

We take the minimum of $f(S, i)$, considering all orderings of stops in S that end at stop i . Suppose j^* is the immediate predecessor of the route with minimum cost, that is,

$$j^* = \arg \min_{j \in S} \{f(S \setminus \{j\}), i) + dp_T(Resst_i - Resst_j) + dp_D \cdot D_{ji} + p \cdot \max(Resst_i - b_i, 0)\}. \quad (6.12)$$

We update $Resst_i$ by

$$Resst_i = \max\{a_i, Resst_{j^*} + T_{j^*i} + s_{j^*}\}. \quad (6.13)$$

We stop when all nodes are visited and output the minimum cost of the best path, mileage of the best path, ordered stops of the best path, and $Resst$ of each stop on the best path.

In Appendix D, we apply the dynamic programming model to a small example to illustrate how it works.

Table 6.1: Summary statistics for 20 routes

Route	Stops	Original TW			TW1			TW2		
		Mean width	Max width	Min width	Mean width	Max width	Min width	Mean width	Max width	Min width
1	18	122.22	300	100	115.70	300	53	108.50	300	50
2	15	143.33	550	100	137.53	550	50	96.67	550	50
3	20	337.50	650	250	193.90	550	73	148.60	550	48
4	15	147.33	300	100	137.93	300	34	111.73	300	34
5	14	367.86	650	150	193.00	650	80	162.93	650	74
6	17	150.00	650	150	142.80	650	41	124.60	650	41
7	20	252.50	450	150	220.00	350	100	177.40	350	65
8	13	261.54	450	200	117.38	450	9	107.38	450	9
9	17	259.80	450	200	216.60	266	113	171.30	250	82
10	15	230.00	350	150	167.10	350	26	103.70	350	26
11	15	342.20	650	150	191.47	350	57	121.30	350	52
12	17	286.47	650	150	197.82	650	73	170.76	650	50
13	17	356.35	850	200	270.94	550	71	235.24	450	50
14	21	347.62	650	100	226.00	450	50	219.24	450	50
15	18	345.33	450	250	212.33	350	56	184.56	350	56
16	20	407.75	650	150	237.80	450	73	117.85	350	67
17	17	330.88	700	200	252.94	650	100	182.65	650	50
18	19	392.11	650	250	299.47	450	59	196.37	450	50
19	15	262.20	350	200	224.13	350	56	168.60	250	56
20	20	245.00	325	50	184.00	325	39	166.80	325	39
Avg	17.15	279.40	536.25	162.50	196.94	449.55	60.65	153.81	433.75	49.95

6.4 Computational Experiments

We used 20 routes from the company’s database. The number of nodes on each route ranges between 13 and 21. In Table 6.1, we list the summary statistics of time windows (TWs), tightened TWs (referred as TW1), and further tightened TWs (referred as TW2).

In Tables 6.2, 6.3, and 6.4, we compare the results produced by the company’s current algorithm (APS) and our dynamic programming based heuristic (HDP). Each table contains the cost of the best path, the distance of the best path, and the running times of APS and HDP. We also calculate the percentage improvement of HDP over APS.

In Table 6.2, we compare the results produced by HDP and the company’s current algorithm (APS) on 20 routes. The average width of the original TWs is

Table 6.2: Performance of APS and HDP on 20 routes with original TW

Route	Stops	Cost(\$)		% ¹	Mileage (mile)		% ¹	Time (s)	
		APS	HDP		APS	HDP		APS	HDP
1	18	42.49	42.49	0.00	2.89	2.89	0.00	10.5	174.19
2	15	43.38	42.72	1.52	9.20	7.88	14.35	12.4	18.00
3	20	28.27	28.27	0.00	4.36	4.36	0.00	15.6	937.08
4	15	56.03	55.65	0.68	7.81	7.05	9.73	10.5	18.53
5	14	13.48	13.40	0.59	2.85	2.78	2.46	12.4	7.67
6	17	32.68	32.68	0.00	4.97	4.97	0.00	15.6	75.64
7	20	25.05	25.05	0.00	3.28	3.28	0.00	10.5	995.12
8	13	13.72	13.72	0.00	2.76	2.76	0.00	12.4	5.73
9	17	23.52	23.09	1.83	4.52	4.13	8.63	15.6	69.88
10	15	15.65	15.65	0.00	3.00	3.00	0.00	10.5	19.26
11	15	38.23	38.12	0.29	14.22	14.03	1.34	12.4	18.13
12	17	47.30	47.11	0.40	4.15	3.76	9.40	15.6	75.44
13	17	36.51	36.51	0.00	12.61	12.61	0.00	16.5	74.30
14	21	38.34	37.99	0.91	5.16	4.88	5.43	18.4	2084.11
15	18	22.46	22.46	0.00	3.91	3.91	0.00	17.4	204.47
16	20	77.39	77.39	0.00	59.39	59.39	0.00	19.5	961.76
17	17	20.77	20.57	0.96	4.62	4.48	3.03	13.8	74.04
18	19	39.32	39.28	0.10	16.85	16.63	1.31	15.6	384.47
19	15	40.70	39.43	3.12	15.29	14.14	7.52	19.5	17.61
20	20	32.79	32.79	0.00	9.78	9.78	0.00	17.3	921.63
Avg	17.15	34.40	34.22	0.54	9.58	9.34	2.56	14.6	356.85

%¹: Percent improvement ($100 \frac{APS-HDP}{APS} \%$)

280 clicks (100 clicks equals one hour), which is about two hours and fifty minutes. The maximum width is 536 clicks (about five and a half hours) and the minimum width is 163 clicks (about an hour and a half). HDP produces better solutions on 10 of 20 problems than APS, with respect to cost and mileage, and ties APS on 10 problems. The average percentage improvement for HDP on cost is 0.54%, and the average percentage improvement on mileage is 2.56%. The average computational time of HDP is six minutes which is much longer than APS (average of 14.6 seconds).

In Table 6.3, we compare the results produced by HDP and to APS on 20 routes with tightened TWs. The average width of the tightened TWs is 197 clicks (100 clicks equals one hour), which is about two hours. The maximum width is 450 clicks (about four and a half hours), and the minimum width is 60 clicks (50 minutes). HDP produces better solutions on 17 of 20 problems with respect to cost and 15

Table 6.3: Performance of APS and HDP on 20 routes with TW1

Route	Stops	Cost(\$)		% ¹	Mileage (mile)		% ¹	Time (s)	
		APS	HDP		APS	HDP		APS	HDP
1	18	41.69	40.48	2.90	4.11	4.31	-4.87	16.5	157.11
2	15	42.72	42.72	0.00	7.88	7.88	0.00	18.4	18.34
3	20	29.08	28.68	1.38	4.98	4.67	6.22	17.4	1,079.00
4	15	42.05	41.82	0.55	8.09	7.64	5.56	19.5	18.00
5	14	13.40	13.40	0.00	2.78	2.78	0.00	13.8	6.99
6	17	35.03	32.68	6.71	4.68	4.97	-6.20	15.6	77.00
7	20	52.30	52.30	0.00	4.36	4.36	0.00	19.5	996.56
8	13	14.30	14.03	1.89	3.12	2.94	5.77	17.3	5.29
9	17	24.02	23.94	0.33	4.75	4.70	1.05	16.5	72.99
10	15	17.25	16.43	4.75	4.19	3.61	13.84	18.4	16.02
11	15	41.00	40.19	1.98	16.25	15.62	3.88	17.4	40.19
12	17	47.70	47.44	0.55	4.94	4.42	10.53	19.5	74.58
13	17	64.13	61.26	4.48	25.78	20.03	22.30	13.8	74.90
14	21	40.68	38.11	6.32	6.66	4.99	25.08	15.6	2772.41
15	18	24.35	23.69	2.71	5.21	4.66	10.56	19.5	191.72
16	20	82.83	80.54	2.76	59.83	59.62	0.35	17.3	1,027.25
17	17	55.55	55.35	0.36	7.82	7.41	5.24	16.5	74.52
18	19	42.64	40.10	5.96	19.72	17.46	11.46	18.4	363.47
19	15	49.80	41.97	15.72	21.56	15.43	28.43	17.4	16.64
20	20	46.60	36.75	21.14	18.15	12.07	33.50	19.5	932.89
Avg	17.15	40.36	38.59	4.37	11.74	10.48	10.77	17.39	400.79

¹%: Percent improvement ($100 \frac{APS-HDP}{APS} \%$)

problems with respect to mileage. HDP ties APS on three problems respect to cost and three problems with respect to mileage. APS produces better solutions than HDP on two problems with respect to mileage. The average percentage improvement for HDP on cost is 4.37%, and the average percentage improvement on mileage is 10.77%. The average computational time of HDP is 6.6 minutes which is much longer than APS (average of 17.34 seconds).

In Table 6.4, we compare the results produced by HDP to APS on 20 routes with further tightened TWs (TW2). The average width of the further tightened TWs is 154 clicks, which is about an hour and a half. The maximum width is 450 clicks (about four and a half hours), and the minimum width is 50 clicks (half an hour). HDP produces better solutions than APS on 18 of 20 problems with respect to cost and 19 problems with respect to mileage. HDP produces the same

Table 6.4: Performance of APS and HDP on 20 routes with TW2

Route	Stops	Cost(\$)		% ¹	Mileage (mile)		% ¹	Time (s)	
		APS	HDP		APS	HDP		APS	HDP
1	18	41.09	40.48	1.48	4.75	4.31	9.26	13.8	156.89
2	15	55.38	54.72	1.19	9.20	7.88	14.35	15.6	161.57
3	20	29.30	29.23	0.24	5.16	5.11	0.97	19.5	1,080.70
4	15	43.68	43.65	0.07	11.36	11.29	0.62	17.3	18.06
5	14	13.48	13.40	0.59	2.85	2.78	2.46	16.5	6.95
6	17	33.41	32.68	2.18	5.53	4.97	10.13	18.4	78.05
7	20	52.77	52.77	0.00	5.31	5.17	2.64	17.4	907.43
8	13	14.26	14.03	1.61	3.09	2.94	4.85	19.5	3.39
9	17	24.55	23.94	2.48	5.26	4.70	10.65	13.8	73.10
10	15	20.20	17.35	14.11	6.51	4.31	33.79	15.6	16.02
11	15	41.62	40.19	3.44	16.62	15.62	6.02	19.5	17.24
12	17	47.67	47.38	0.61	4.88	4.30	11.89	17.3	74.65
13	17	64.07	62.41	2.59	25.65	22.34	12.90	16.5	74.58
14	21	38.99	38.53	1.18	5.63	5.23	7.10	18.4	2180.30
15	18	26.30	25.76	2.05	6.86	6.45	5.98	19.5	191.72
16	20	83.97	84.16	-0.23	59.88	60.25	-0.62	13.8	941.95
17	17	56.82	55.36	2.57	10.35	7.44	28.12	15.6	79.04
18	19	52.07	51.05	1.96	21.11	19.07	9.66	19.5	361.29
19	15	51.25	44.71	12.76	22.66	18.02	20.48	17.3	16.39
20	20	54.67	36.75	32.78	24.42	12.07	50.57	15.3	887.50
Avg	17.15	42.28	40.43	4.38	12.85	11.21	12.77	17.00	366.34

%¹: Percent improvement ($100 \frac{APS-HDP}{APS} \%$)

solutions as APS on one problem with respect to cost. APS produces one solution that is better than HDP with respect to cost and mileage. The average percentage improvement for HDP on cost is 4.38%, and the average percentage improvement on mileage is 12.77%. The average computational time of HDP is 6 minutes which is much longer than APS (average of 17 seconds).

6.4.1 Exact Dynamic Programming

The current dynamic programming model does not include the time variable in the state (S, i) , although we did include *Resst* in each generation. The exact dynamic programming (EDP) includes the time variable t in the state. For example, a state in the exact dynamic programming is (S, i, t) , where S is the set of stops to be visited after the driver departs from the start depot, stop i is to be visited after

S , and t is the service start time at stop i . We want to compare the performance of HDP to the performance of EDP.

Let $F(S, j, t)$ be the least cost of a path starting at start depot, passing through every node of S , a subset of all nodes other than the depot exactly once, ending at $i \notin S$, and ready to serve i at time t . The recursive formula is given by

$$F(S, j, t) = \min_{(i,j)} \{F(S - \{j\}), i, t'\} + c_{ij} | t \geq t' + s_i + t_{i,j}, a_j \leq b_j \}. \quad (6.14)$$

Note that there are several paths that visit set S and end at i . Among them, we only keep the Pareto optimal element. In other words, given (S, i, t_1) and (S, i, t_2) , the second state can be eliminated if $t_1 \leq t_2$ and $F(S, i, t_1) \leq F(S, i, t_2)$.

6.4.2 Computational Results

In Tables 6.5, 6.6 and 6.7, we compare the results produced by HDP and EDP. Each table contains the cost, the distance of the best path, and the running time of each procedure. We also calculate the percentage improvement of HDP and EDP over APS.

In Table 6.5, we compare the results produced by EDP to those produced by APS and HDP on 20 routes. The average width of the original TWs is 280 clicks (100 clicks equals one hour), which is about two hours and fifty minutes. The maximum width is 536 clicks (about five and a half hours) and the minimum width is 163 clicks (about an hour and a half). EDP produces the same solutions on all routes as HDP. EDP produces better solutions on 10 of 20 problems than APS, with respect to cost and mileage, and ties APS on 10 problems. The average percentage

Table 6.5: Performance of APS, HDP, and EDP on 20 routes with original TW

Route	Stops	Cost(\$)			% ¹	Mileage (mile)			% ¹	Time (s)		
		APS	HDP	EDP		APS	HDP	EDP		APS	HDP	EDP
1	18	42.49	42.49	42.49	0.00	2.89	2.89	2.89	0.00	10.5	174.19	281.11
2	15	43.38	42.72	42.72	1.52	9.20	7.88	7.88	14.35	12.4	18.00	24.11
3	20	28.27	28.27	28.27	0.00	4.36	4.36	4.36	0.00	15.6	937.08	1125.65
4	15	56.03	55.65	55.65	0.68	7.81	7.05	7.05	9.73	10.5	18.53	46.28
5	14	13.48	13.40	13.40	0.59	2.85	2.78	2.78	2.46	12.4	7.67	43.85
6	17	32.68	32.68	32.68	0.00	4.97	4.97	4.97	0.00	15.6	75.64	91.11
7	20	25.05	25.05	25.05	0.00	3.28	3.28	3.28	0.00	10.5	995.12	1478.54
8	13	13.72	13.72	13.72	0.00	2.76	2.76	2.76	0.00	12.4	5.73	7.08
9	17	23.52	23.09	23.09	1.83	4.52	4.13	4.13	8.63	15.6	69.88	78.21
10	15	15.65	15.65	15.65	0.00	3.00	3.00	3.00	0.00	10.5	19.26	35.82
11	15	38.23	38.12	38.12	0.29	14.22	14.03	14.03	1.34	12.4	18.13	30.64
12	17	47.30	47.11	47.11	0.40	4.15	3.76	3.76	9.40	15.6	75.44	103.58
13	17	36.51	36.51	36.51	0.00	12.61	12.61	12.61	0.00	16.5	74.30	90.64
14	21	38.34	37.99	37.99	0.91	5.16	4.88	4.88	5.43	18.4	2084.11	4182.51
15	18	22.46	22.46	22.46	0.00	3.91	3.91	3.91	0.00	17.4	204.47	275.68
16	20	77.39	77.39	77.39	0.00	59.39	59.39	59.39	0.00	19.5	961.76	1435.66
17	17	20.77	20.57	20.57	0.96	4.62	4.48	4.48	3.03	13.8	74.04	106.97
18	19	39.32	39.28	39.28	0.10	16.85	16.63	16.63	1.31	15.6	384.47	660.10
19	15	40.70	39.43	39.43	3.12	15.29	14.14	14.14	7.52	19.5	17.61	31.35
20	20	32.79	32.79	32.79	0.00	9.78	9.78	9.78	0.00	17.3	921.63	1122.59
Avg	17.15	34.40	34.22	34.22	0.54	9.58	9.34	9.34	2.56	14.6	356.85	562.57

¹: Percent improvement $(100 \frac{APS-EDP}{APS} \%)$

improvement for EDP on cost is 0.54%, and the average percentage improvement on mileage is 2.56%. The average computational time of EDP is nine minutes which is longer than HDP (average of six minutes).

In Table 6.6, we compare the results produced by EDP to those produced by APS and HDP on 20 routes with tightened TWs. The average width of the tightened TWs is 197 clicks (100 clicks equals one hour), which is about two hours. The maximum width is 450 clicks (about four and a half hours), and the minimum width is 60 clicks (50 minutes). EDP produces the same solutions on all routes as HDP. EDP produces better solutions than APS on 17 of 20 problems with respect to cost and 15 problems with respect to mileage. EDP ties APS on three problems respect to cost and three problems with respect to mileage. APS produces better solutions than EDP on two problems with respect to mileage. The average percentage improvement

Table 6.6: Performance of APS, HDP, and EDP on 20 routes with TW1

Route	Stops	Cost(\$)			% ¹	Mileage (mile)			% ¹	Time (s)		
		APS	HDP	EDP		APS	HDP	EDP		APS	HDP	EDP
1	18	41.69	40.48	40.48	2.90	4.11	4.31	4.31	-4.87	16.5	157.11	304.95
2	15	42.72	42.72	42.72	0.00	7.88	7.88	7.88	0.00	18.4	18.34	27.95
3	20	29.08	28.68	28.68	1.38	4.98	4.67	4.67	6.22	17.4	1,079.00	1090.8
4	15	42.05	41.82	41.82	0.55	8.09	7.64	7.64	5.56	19.5	18.00	40.85
5	14	13.40	13.40	13.40	0.00	2.78	2.78	2.78	0.00	13.8	6.99	28.79
6	17	35.03	32.68	32.68	6.71	4.68	4.97	4.97	-6.20	15.6	77.00	89.32
7	20	52.30	52.30	52.30	0.00	4.36	4.36	4.36	0.00	19.5	996.56	1109.94
8	13	14.30	14.03	14.03	1.89	3.12	2.94	2.94	5.77	17.3	3.51	5.56
9	17	24.02	23.94	23.94	0.33	4.75	4.70	4.70	1.05	16.5	72.99	95.5
10	15	17.25	16.43	16.43	4.75	4.19	3.61	3.61	13.84	18.4	16.02	29.03
11	15	41.00	40.19	40.19	1.98	16.25	15.62	15.62	3.88	17.4	18.42	31.68
12	17	47.70	47.44	47.44	0.55	4.94	4.42	4.42	10.53	19.5	74.58	103.14
13	17	64.13	61.26	61.26	4.48	25.78	20.03	20.03	22.30	13.8	74.90	98.38
14	21	40.68	38.11	38.11	6.32	6.66	4.99	4.99	25.0	15.6	2772.41	3546.8
15	18	24.35	23.69	23.69	2.71	5.21	4.66	4.66	10.56	19.5	191.72	255.94
16	20	82.83	80.54	80.54	2.76	59.83	59.62	59.62	0.35	17.3	1,027.25	1542.35
17	17	55.55	55.35	55.35	0.36	7.82	7.41	7.41	5.24	16.5	74.52	104.45
18	19	42.64	40.10	40.10	5.96	19.72	17.46	17.46	11.46	18.4	363.47	1045.1
19	15	49.80	41.97	41.97	15.72	21.56	15.43	15.43	28.43	17.4	16.64	38.14
20	20	46.60	36.75	36.75	21.14	18.15	12.07	12.07	33.50	19.5	932.89	1261.19
Avg	17.15	40.36	38.59	38.59	4.37	11.74	10.48	10.48	10.77	17.39	399.62	542.49

%¹: Percent improvement ($100 \frac{APS - EDP}{APS} \%$)

for EDP on cost is 4.37%, and the average percentage improvement on mileage is 10.77%. The average computational time of EDP is 9 minutes which is longer than HDP (average of 6.6 minutes).

In Table 6.7, we compare the results produced by EDP to those produced by APS and HDP on 20 routes with further tightened TWs (TW2). The average width of the further tightened TWs is 154 clicks, which is about an hour and a half. The maximum width is 450 clicks (about four and a half hours), and the minimum width is 50 clicks (half an hour). EDP produces the same solutions on all routes as HDP. EDP produces better solutions than APS on 18 of 20 problems with respect to cost and 19 problems with respect to mileage. EDP produces the same solutions as APS on one problem with respect to cost. APS produces one solution that is better than EDP with respect to cost and mileage. The average percentage improvement

Table 6.7: Performance of APS, HDP, and EDP on 20 routes with TW2

Route	Stops	Cost(\$)			% ¹	Mileage (mile)			% ¹	Time (s)		
		APS	HDP	EDP		APS	HDP	EDP		APS	HDP	EDP
1	18	41.09	40.48	40.48	1.48	4.75	4.31	4.31	9.26	13.8	156.89	303.37
2	15	55.38	54.72	54.72	1.19	9.20	7.88	7.88	14.35	15.6	16.15	26.46
3	20	29.30	29.23	29.23	0.24	5.16	5.11	5.11	0.97	19.5	1,080.70	1171.51
4	15	43.68	43.65	43.65	0.07	11.36	11.29	11.29	0.62	17.3	18.06	25.73
5	14	13.48	13.40	13.40	0.59	2.85	2.78	2.78	2.46	16.5	6.95	35.65
6	17	33.41	32.68	32.68	2.18	5.53	4.97	4.97	10.13	18.4	78.05	72.11
7	20	52.77	52.70	52.70	0.13	5.31	5.17	5.17	2.64	17.4	907.43	1029.05
8	13	14.26	14.03	14.03	1.61	3.09	2.94	2.94	4.85	19.5	3.39	5.11
9	17	24.74	24.46	24.46	1.13	5.40	5.17	5.17	4.26	13.8	73.1	76.04
10	15	20.20	17.35	17.35	14.11	6.51	4.31	4.31	33.79	15.6	16.02	28.15
11	15	41.62	40.19	40.19	3.80	16.62	15.62	15.62	6.38	19.5	17.24	26.14
12	17	47.67	47.38	47.38	0.61	4.88	4.30	4.30	11.89	17.3	74.65	105.5
13	17	64.07	62.41	62.41	2.59	25.65	22.34	22.34	12.90	16.5	74.58	110.64
14	21	38.99	38.53	38.53	1.18	5.63	5.23	5.23	7.10	18.4	2180.3	12638.4
15	18	26.30	24.59	24.59	6.50	6.86	5.55	5.55	19.10	19.5	191.72	258.82
16	20	83.97	84.16	84.16	-0.23	59.88	60.25	60.25	-0.62	13.8	941.95	1765.45
17	17	56.82	55.36	55.36	2.57	10.35	7.44	7.44	28.12	15.6	79.04	83.62
18	19	52.07	51.05	51.05	1.96	21.11	19.07	19.07	9.66	19.5	361.29	701.34
19	15	51.25	44.71	44.71	12.76	22.66	18.02	18.02	20.48	17.3	16.39	36.54
20	20	54.67	36.75	36.75	32.78	24.42	12.07	12.07	50.57	15.3	887.5	1426.75
Avg	17.15	42.28	40.43	40.43	4.38	12.85	11.21	11.21	12.77	17.00	359.07	996.32

%¹: Percent improvement ($100 \frac{APS-EDP}{APS} \%$)

on cost is 4.38%, and the average percentage improvement on mileage is 12.77%.

The average computational time of EDP is 16.6 minutes which is longer than HDP (average of six minutes).

6.5 Conclusions

The dynamic programming-based heuristic produced better routes in most cases than the company's current algorithm. It solved problems with 13 to 20 stops in reasonable amount of time and produced solutions that were about 10% better on average than the company's algorithm.

Some intelligent techniques are needed to speed up dynamic programming to solve larger problems. The possible options are geographic clustering and space reduction by the time windows.

Chapter 7

Conclusions

In this dissertation, we discussed five important problems in operations research: the concave cost supply scheduling problem, the controlled tabular adjustment problem, the two-stage transportation problem, the generalized orienteering problem, and the shortest pickup planning tour problem with time windows.

We applied an elite genetic algorithm to solve the GOP and compared it with an artificial neural network, a code that is specifically designed to handle nonlinear problems (such as the GOP). The results produced by EGA were comparable to the results produced by the artificial neural network with respect to solution quality. EGA was much faster. Our results demonstrated that genetic algorithms perform well on the GOP and should perform well on other nonlinear combinatorial optimization problems.

We combined genetic algorithms with linear programming to solve SSP, CTA, and TsTP. This is due to the fact that all the three problems can be modeled as mixed integer programs. However, it is not practical or sometimes impossible to find optimal solutions to MIPs in a reasonable amount of computing time. Linear programs are easy to formulate and solve. We used genetic algorithms to determine the binary decision variables, and then solved the resulting LP model.

To check the performance of our heuristics, we found the optimal solutions of

small-size to medium-size problems using enumeration or mixed integer programming.

We developed two genetic algorithms (GAs) to solve two versions of the concave cost supply scheduling problem. The first case had n providers and a single manufacturing unit. We used a genetic algorithm to select the providers and assign quantities using the greedy method. The first GA produced optimal solutions to all benchmark problem instances. The second case had n providers and m manufacturing units. We used a genetic algorithm to determine the open edges from providers to the manufacturing units, then formulated a linear program to distribute the quantities to be delivered. Our GA was simple in structure, quick in convergence, and produced nearly optimal solutions.

We applied EGA-CTA to solve the controlled tabular adjustment problem. EGA-CTA had two phases. In phase 1, we used a genetic algorithm to perturb the sensitive cells, and in phase 2, we solved the remaining LP model to perturb the nonsensitive cells. EGA-CTA was simple in structure and quick in convergence. It produced optimal or near-optimal solutions on medium-size problems in a reasonable amount of computing time.

We modified the genetic algorithm in the concave cost supply scheduling problem to solve the two-stage transportation problem. Our computational results showed that the genetic algorithm was simple in structure, quick in convergence, and produced good solutions. More importantly, our work demonstrated that GAs can be applied successfully to a variety of supply scheduling problems.

Based on extensive computational experiments, our genetic algorithms per-

formed well with respect to solution quality and running time. In addition, they were relatively straightforward with a small number of parameters. Overall, our work shows that genetic algorithms can be very effective in handling difficult combinatorial optimization problems. We expect to see many more successful applications of genetic algorithms to NP-hard problems.

We developed a dynamic programming-based heuristic to solve the SPTP-TW. Our heuristic produced better routes in most cases than the current algorithm used by a package delivery company. Our heuristic solved problems with 13 to 20 stops in a reasonable amount of computing time and produced solutions that were about 10% better on average than the company's algorithm.

Appendix A

MATLAB Code for Computing Distances

Input

$a1$ = Longitude of node 1,

$b1$ = Latitude of node 1,

$a2$ = Longitude of node 2,

$b2$ = Latitude of node 2,

The function returns the distance (in km) between node 1 and node 2.

```
function d=dist(a1,b1,a2,b2)
```

```
    R = 6371;
```

```
    a1 = a1 * 3.1416/180;
```

```
    b1 = (90 - b1) * 3.1416/180;
```

```
    a2 = a2 * 3.1416/180;
```

```
    b2 = (90 - b2) * 3.1416/180;
```

```
    c = a1 - a2;
```

```
    d = b1 - b2;
```

```
    e = b1 + b2;
```

```
    x1 = cos(d * 0.5)/cos(e * 0.5)/tan(0.5 * c);
```

```
    x2 = sin(d * 0.5)/sin(e * 0.5)/tan(0.5 * c);
```

```
    f = atan(x1) + atan(x2);
```

```
    g = asin(sin(c) * sin(b1)/sin(f));
```

```
    % Output the distance
```

```
    d = R * g;
```

Appendix B

Data for Ten New Concave Cost Supply Scheduling Problems

Table B.1: Ten Providers and Twelve Manufacturing Units(10P12U)

Fixed Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12
P1	17	12	10	6	11	7	5	18	12	8	17	7
P2	15	5	11	9	8	14	18	15	17	10	16	9
P3	9	14	8	20	14	11	12	13	14	9	18	15
P4	9	17	18	8	14	6	5	9	20	13	7	19
P5	15	7	14	15	15	18	6	14	17	9	16	8
P6	10	9	20	9	6	16	17	14	10	10	9	11
P7	18	8	15	15	14	12	12	16	11	7	15	20
P8	12	14	13	6	18	14	16	16	10	8	15	10
P9	5	7	7	12	14	20	10	19	8	16	13	17
P10	19	9	11	5	18	12	12	11	11	19	6	9

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12
P1	0.8	0.5	0.2	0.7	0.6	0.2	0.6	0.7	0.7	0.6	0.2	0.3
P2	1	0.8	0.4	0.1	0.4	1	0.9	0.7	0.2	0.2	0.7	0.6
P3	0.7	0.3	0.6	0.5	0.9	0.8	0.6	0.6	0.6	0.3	0.2	0.3
P4	0.4	0.2	1	1	0.4	0.2	0.7	0.4	0.2	0.5	0.4	0.2
P5	0.1	0.4	0.9	0.6	0.1	0.4	0.4	0.6	1	0.4	0.7	0.2
P6	0.9	0.1	1	0.9	0.6	0.8	0.9	0.1	0.5	0.9	0.1	0.7
P7	0.4	0.7	0.8	0.5	0.7	0.8	0.8	0.1	0.2	0.4	0.5	0.3
P8	0.1	0.1	0.9	0.9	0.7	0.8	0.2	0.2	0.6	0.6	0.4	0.7
P9	0.4	0.1	0.7	0.3	0.9	0.1	0.6	0.6	0.3	0.1	0.5	0.6
P10	0.9	0.6	0.4	0.3	0.7	0.5	1	0.2	0.4	0.8	0.3	0.9

Demand

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12
D	110	90	30	90	130	90	50	80	120	70	110	60

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10		
m	10	10	10	10	10	10	10	10	10	10		
M	70	70	150	160	140	150	180	80	90	110		

Table B.2: 12P15U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	12	12	8	15	5	19	14	19	16	15	8	10	8	20	13
P2	8	12	15	5	19	18	18	19	19	16	19	20	16	19	17
P3	12	15	8	20	20	13	8	20	6	13	7	17	8	14	11
P4	15	18	5	15	5	10	17	9	6	10	15	19	8	19	16
P5	15	8	10	20	14	17	14	10	8	6	8	11	10	12	12
P6	6	20	18	20	8	8	17	19	15	7	19	10	7	11	10
P7	12	14	7	19	5	11	5	9	5	18	15	15	10	16	11
P8	19	18	8	10	18	10	13	12	9	7	18	7	14	11	6
P9	10	14	7	13	5	11	10	13	10	9	11	15	10	14	14
P10	8	8	14	19	7	13	7	13	11	19	15	14	14	9	17
P11	8	8	16	18	7	5	9	8	11	12	19	19	18	5	13
P12	5	17	6	8	10	13	19	19	16	7	20	18	19	18	19

Variable Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.5	0.5	0.6	0.5	1	0.9	0.9	0.1	0.1	0.8	1	0.8	0.5	0.4	0.9
P2	0.1	0.5	1	0.7	0.1	0.1	0.7	0.1	0.1	0.4	0.9	0.5	0.9	0.9	0.9
P3	0.7	0.5	0.4	0.1	0.1	0.7	0.7	0.9	0.5	0.3	0.9	0.1	1	0.1	1
P4	0.1	0.1	0.8	0.8	0.9	0.4	0.9	0.9	0.9	1	0.3	1	0.5	0.6	0.5
P5	0.6	0.6	0.3	0.2	1	0.5	0.9	0.7	0.6	0.8	0.6	0.6	0.9	0.7	0.6
P6	0.3	0.1	0.1	0.9	0.7	0.2	0.8	0.3	1	0.8	0.1	0.3	0.5	0.2	0.9
P7	0.8	0.6	1	0.6	0.8	0.9	0.1	0.7	1	0.5	0.4	0.2	0.4	0.9	0.7
P8	0.2	0.3	0.7	0.3	0.1	0.7	0.8	0.5	0.6	0.1	0.7	0.8	0.6	0.3	0.2
P9	0.3	0.4	0.6	0.7	0.8	0.4	0.7	0.1	0.6	0.5	1	0.7	0.3	0.2	0.9
P10	0.5	0.7	1	0.7	0.9	0.6	0.4	0.1	0.6	0.4	0.6	0.8	0.7	0.4	1
P11	0.2	0.6	0.7	1	0.7	1	0.1	0.2	0.2	0.7	0.4	0.7	0.6	0.4	0.8
P12	0.9	0.2	1	0.8	0.1	0.9	0.5	0.8	0.3	1	0.9	0.1	0.9	0.3	0.3

Demand															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
D	20	130	120	90	50	60	110	20	140	30	110	70	80	90	80

Minimum Quantity and Capacity												
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
m	10	10	10	10	10	10	10	10	10	10	10	10
M	150	60	150	100	110	200	180	150	60	200	170	180

Table B.3: 15P18U

Fixed Cost																
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	
P1	18	8	16	7	20	13	7	12	10	9	19	14	15	11	6	
P2	15	9	10	8	16	12	15	20	17	10	5	13	20	19	10	
P3	9	14	5	17	17	16	7	10	18	11	6	19	6	14	17	
P4	20	11	8	17	12	8	8	11	13	18	12	15	6	13	13	
P5	18	12	11	20	17	13	6	14	17	15	9	5	6	14	7	
P6	9	6	7	13	20	10	20	17	17	14	18	13	20	12	12	
P7	13	10	13	11	5	16	9	12	7	11	6	12	20	19	9	
P8	13	9	11	16	17	5	14	16	5	10	15	14	10	10	19	
P9	17	14	20	14	14	9	10	8	18	19	5	17	14	15	16	
P10	5	17	15	11	20	19	20	14	14	7	13	20	9	17	20	
P11	15	13	7	10	13	18	15	19	10	20	17	6	15	15	7	
P12	19	13	20	18	8	9	19	7	6	19	14	15	11	6	15	
P13	13	7	5	6	13	20	16	13	16	11	19	18	11	15	15	
P14	15	11	20	5	18	15	19	9	11	10	9	20	18	11	14	
P15	7	9	18	6	9	5	9	16	10	17	15	16	8	16	15	
	U16	U17	U18													
P1	10	9	17													
P2	18	10	19													
P3	18	7	20													
P4	20	16	7													
P5	12	8	8													
P6	18	7	10													
P7	15	6	13													
P8	10	11	16													
P9	16	7	11													
P10	16	17	6													
P11	16	8	7													
P12	17	20	8													
P13	11	15	14													
P14	17	13	10													
P15	14	6	19													

Variable Cost																
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	
P1	1	0.3	0.7	0.2	0.2	0.9	0.9	1	0.4	0.5	0.8	0.2	1	0.1	0.8	
P2	0.8	0.4	0.3	0.2	0.7	0.5	0.5	0.9	0.2	0.6	0.6	0.9	0.9	0.3	0.9	
P3	0.6	0.4	0.9	0.8	0.7	0.8	1	0.9	0.3	1	0.7	0.9	0.6	0.8	0.7	
P4	0.1	0.3	0.6	0.7	0.9	0.1	0.4	0.3	0.5	0.6	0.4	0.6	0.2	0.9	0.6	
P5	1	0.9	0.6	0.8	0.8	0.3	0.5	0.9	0.2	0.9	0.4	0.3	0.2	0.2	0.7	
P6	0.6	0.7	0.4	1	0.8	0.4	0.4	0.9	0.6	0.5	0.5	1	0.2	0.7	0.2	
P7	0.9	0.5	1	0.5	0.3	0.2	1	0.9	0.7	0.2	0.5	1	0.1	0.5	0.3	
P8	0.5	0.1	0.8	0.2	0.8	0.8	0.8	0.4	0.1	0.4	0.5	0.5	0.2	0.7	0.5	
P9	0.7	1	0.2	0.4	0.7	0.1	0.3	0.2	0.3	0.9	0.8	0.7	0.9	0.5	0.4	
P10	0.6	0.7	0.9	1	0.2	0.3	0.2	0.5	0.2	0.8	0.6	0.8	0.4	0.8	0.7	
P11	0.4	0.4	0.7	0.3	0.1	0.5	0.9	0.4	0.8	0.6	0.1	0.8	0.4	0.5	0.5	
P12	0.2	0.7	0.1	0.5	0.1	0.1	0.8	0.5	0.9	1	0.5	0.6	0.2	0.7	1	
P13	0.8	0.3	0.6	0.2	0.1	0.1	0.3	0.1	0.5	0.3	1	0.1	0.4	0.2	0.7	
P14	1	0.7	0.8	0.3	1	0.3	0.8	0.7	0.7	0.3	0.4	0.6	0.3	0.6	1	
P15	0.8	1	0.9	0.1	0.1	0.6	0.8	0.1	0.5	0.4	0.1	0.3	0.7	0.8	0.3	
	U16	U17	U18													
P1	0.6	0.2	0.3													
P2	0.3	0.1	0.2													
P3	0.6	0.8	0.9													
P4	0.8	1	0.7													
P5	0.5	0.4	1													

P6	0.1	0.9	0.5
P7	1	1	0.2
P8	0.4	0.1	0.2
P9	0.4	0.9	0.9
P10	0.8	0.8	0.7
P11	0.9	0.7	0.2
P12	0.8	0.6	0.2
P13	0.3	0.9	0.3
P14	0.8	0.6	0.1
P15	0.1	0.8	0.3

Demand

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
D	40	40	30	30	20	120	110	150	130	80	50	80	130	60	130
	U16	U17	U18												
	150	130	80												

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	90	130	130	60	90	100	190	70	70	100	140	160	90	200	170

Table B.4: 18P20U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	10	14	11	5	13	14	8	13	14	18	18	11	12	10	11
P2	17	7	12	16	11	13	17	14	9	20	18	5	5	13	20
P3	13	8	16	7	13	8	16	7	9	17	18	11	10	7	14
P4	10	6	6	13	19	18	6	19	7	18	19	7	8	6	14
P5	16	7	15	16	15	7	15	8	13	16	15	19	10	7	6
P6	15	18	12	5	19	10	18	20	14	9	13	6	17	6	8
P7	15	10	20	8	6	18	10	9	6	8	14	20	7	17	8
P8	19	18	18	8	17	16	16	14	8	17	5	17	12	14	6
P9	19	18	13	7	20	13	8	5	10	13	9	6	14	7	14
P10	19	14	11	19	10	14	5	13	16	15	8	6	7	11	10
P11	16	7	6	19	14	6	19	9	16	20	9	13	19	7	17
P12	12	12	8	7	14	7	17	8	19	17	6	16	10	17	15
P13	11	11	9	20	12	18	10	16	15	20	20	16	8	18	6
P14	6	6	17	9	9	19	14	13	7	20	17	5	8	17	18
P15	20	10	15	14	11	20	12	14	13	7	16	12	16	20	17
P16	14	12	8	20	6	9	6	18	10	12	20	8	13	13	17
P17	13	17	19	12	19	9	7	10	20	12	7	14	7	18	20
P18	19	14	5	20	17	14	7	18	17	15	16	11	10	9	14
	U16	U17	U18	U19	U20										
P1	7	17	5	14	15										
P2	19	10	18	20	14										
P3	17	9	12	16	8										
P4	18	20	14	16	10										
P5	14	14	10	19	13										
P6	7	17	14	14	16										
P7	12	13	14	8	20										
P8	5	9	10	15	16										
P9	5	15	20	12	18										
P10	8	14	8	17	14										
P11	13	16	17	19	9										
P12	8	6	15	18	11										
P13	12	12	17	20	9										
P14	14	18	14	11	17										
P15	9	18	15	10	9										
P16	8	15	9	20	18										
P17	7	19	16	5	9										
P18	7	9	16	16	9										

Variable Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.3	0.1	0.5	0.8	0.8	0.5	0.4	1	0.6	0.4	0.9	1	0.8	0.9	0.9
P2	0.4	1	0.8	0.2	0.4	0.4	0.4	0.4	0.8	0.8	0.1	0.5	0.9	0.4	0.1
P3	0.9	0.9	0.2	0.6	0.6	0.9	0.6	0.1	0.9	0.8	0.7	0.4	0.9	0.8	0.8
P4	0.9	0.9	0.4	0.9	0.1	0.6	0.9	0.8	0.4	0.4	1	0.4	0.6	1	0.3
P5	0.6	0.8	0.2	0.1	0.7	0.8	0.1	0.3	0.3	0.8	1	0.9	0.1	0.8	0.4
P6	0.4	0.4	0.1	0.6	0.1	0.2	0.3	1	1	0.4	0.7	0.9	0.7	0.4	0.9
P7	0.7	0.1	1	0.5	0.2	0.6	0.9	0.1	0.7	0.5	0.7	0.9	0.7	1	1
P8	0.5	1	0.5	0.3	0.4	0.1	0.3	1	0.9	0.3	0.4	0.1	0.9	0.6	0.2
P9	0.1	0.5	0.7	0.4	0.4	0.9	0.9	0.3	0.5	0.2	0.1	0.7	1	0.3	0.8
P10	0.4	0.6	0.4	0.1	0.8	1	0.5	1	0.7	0.9	0.4	0.4	0.5	0.7	0.1
P11	0.6	0.3	0.6	0.2	0.6	0.3	0.4	0.2	0.8	0.4	0.4	0.1	0.7	0.6	0.3
P12	1	0.4	0.8	0.9	0.3	0.4	0.7	1	0.6	0.7	0.9	0.6	0.5	0.6	0.5
P13	0.9	0.7	0.5	0.9	0.9	0.9	0.9	0.3	0.4	0.3	0.6	0.8	0.4	0.4	0.4
P14	0.7	0.8	0.8	0.2	0.5	0.9	0.5	0.2	0.1	0.7	0.1	0.4	0.6	0.2	0.3
P15	0.6	0.2	0.9	0.3	0.7	0.4	0.5	0.2	0.1	0.2	0.3	0.5	0.7	0.7	0.7

P16	0.7	0.8	0.9	0.6	0.7	0.2	0.7	0.6	0.9	0.4	1	0.8	0.4	0.5	1
P17	1	0.8	0.3	0.1	0.1	0.4	0.5	0.9	0.6	0.1	1	0.1	1	0.7	0.3
P18	0.4	0.4	0.3	0.5	0.7	0.4	0.7	0.9	0.7	0.9	0.5	0.3	0.2	0.1	0.8
	U16	U17	U18	U19	U20										
P1	0.7	0.6	0.8	0.8	0.3										
P2	0.3	0.6	0.8	0.6	1										
P3	0.7	0.4	0.5	0.5	0.3										
P4	1	0.3	0.2	0.4	0.1										
P5	0.9	0.2	0.8	0.7	0.2										
P6	0.3	0.4	0.5	0.4	0.7										
P7	0.9	0.9	0.2	1	0.5										
P8	0.6	0.6	0.3	0.4	0.4										
P9	1	0.2	0.8	0.7	1										
P10	0.9	0.3	0.7	0.4	0.64										
P11	0.5	0.1	0.6	0.1	0.9										
P12	0.1	0.8	0.2	0.5	0.1										
P13	0.2	0.7	0.2	1	0.1										
P14	0.6	0.8	1	0.9	0.8										
P15	0.3	0.6	0.4	0.2	0.4										
P16	0.6	0.7	1	0.3	0.6										
P17	0.5	0.6	0.9	0.3	0.5										
P18	0.5	1	0.9	0.8	0.7										

Demand

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
D	30	100	70	80	150	80	90	30	70	150	60	60	60	100	110
	U16	U17	U18	U19	U20										
	30	150	140	110	20										

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	170	70	150	140	90	130	90	160	200	70	80	60	140	150	100
	P16	P17	P18												
	10	10	10												
	140	190	90												

Table B.5: 20P25U

Fixed Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	9	11	19	6	13	16	14	5	17	10	9	10	5	11	19
P2	15	20	12	8	17	18	5	9	16	7	17	20	11	10	12
P3	15	5	8	9	5	8	10	14	13	18	16	15	7	5	16
P4	12	15	10	20	8	16	13	15	11	18	14	5	13	16	8
P5	18	10	7	15	12	6	11	8	19	14	15	9	16	11	11
P6	5	5	6	14	8	9	12	20	14	14	12	11	11	18	9
P7	18	8	9	15	19	11	10	20	17	19	10	19	7	15	8
P8	9	15	16	6	20	16	10	7	14	20	7	6	7	15	9
P9	19	13	19	14	19	8	18	15	6	18	8	10	10	14	18
P10	15	17	15	11	17	18	18	18	16	20	18	9	10	10	6
P11	10	10	6	10	15	9	7	20	7	8	9	5	15	5	13
P12	6	15	11	15	17	13	9	5	20	7	15	14	18	15	19
P13	13	5	10	7	11	6	15	7	12	8	14	13	8	10	13
P14	13	16	16	11	12	20	6	19	5	11	10	13	11	11	12
P15	17	10	8	17	16	7	18	17	5	15	20	9	10	19	19
P16	14	18	20	14	18	7	16	7	5	6	7	7	8	17	13
P17	20	19	7	8	15	8	8	16	12	11	9	9	9	13	6
P18	7	14	15	13	20	17	10	18	9	19	10	7	19	7	15
P19	14	15	18	8	8	19	8	20	11	8	13	17	12	19	6
P20	19	15	12	14	17	11	12	18	14	18	14	6	11	8	6
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25					
P1	18	16	10	6	15	10	12	8	9	11					
P2	17	5	9	9	5	18	17	9	13	6					
P3	8	18	10	12	19	8	20	14	10	19					
P4	17	11	20	14	11	5	12	15	7	7					
P5	17	6	12	17	14	16	7	15	6	8					
P6	12	16	17	8	12	5	5	12	7	15					
P7	8	7	13	11	5	18	6	8	13	20					
P8	11	10	10	8	8	12	19	14	5	15					
P9	7	7	17	9	17	16	11	13	14	13					
P10	18	6	12	19	5	10	8	14	17	13					
P11	17	14	13	6	12	9	8	18	20	15					
P12	9	20	18	17	10	15	16	11	12	13					
P13	9	9	16	18	11	13	19	15	17	15					
P14	5	14	7	18	5	7	6	12	13	16					
P15	18	10	6	5	9	12	16	16	7	18					
P16	7	7	14	7	6	15	19	20	12	16					
P17	17	8	15	16	6	7	11	10	18	20					
P18	13	20	7	7	15	10	15	6	15	9					
P19	15	13	16	17	11	7	8	8	5	7					
P20	7	14	10	18	13	11	6	13	17	6					

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.2	0.6	0.4	0.7	0.1	1	0.2	0.6	0.8	0.1	0.8	0.9	0.4	0.6	0.1
P2	0.7	1	0.9	0.2	0.5	0.6	0.5	0.5	0.1	0.7	0.9	0.9	0.7	0.3	0.2
P3	0.7	0.4	0.4	0.3	0.3	0.5	0.2	0.9	0.6	0.3	1	1	0.5	0.9	0.3
P4	0.9	0.6	0.8	0.6	0.7	0.4	0.7	0.6	0.5	0.2	0.2	0.9	0.5	0.1	0.9
P5	0.4	0.7	0.2	0.4	0.4	0.6	1	0.2	0.3	0.6	1	0.9	0.1	0.4	0.7
P6	0.5	0.7	0.8	0.7	0.7	0.2	0.3	1	0.4	0.7	0.5	0.5	0.8	1	0.8
P7	0.8	0.6	0.2	0.6	0.1	0.7	0.8	0.7	0.1	0.3	0.1	0.9	1	1	0.5
P8	0.4	0.9	0.4	0.2	0.9	0.1	1	0.1	1	0.8	0.6	0.8	0.3	0.8	0.9
P9	0.9	0.3	0.4	1	0.9	0.3	0.4	0.3	0.2	0.7	0.9	0.4	0.5	0.9	0.2
P10	0.2	0.1	0.5	0.3	0.4	0.4	0.1	0.4	0.4	0.1	0.5	0.4	0.3	1	0.1
P11	0.1	1	1	0.2	0.7	0.7	0.5	0.7	0.4	0.8	0.4	0.9	1	0.9	0.6
P12	0.1	0.8	0.4	0.6	0.1	0.6	1	0.7	0.8	0.4	1	0.3	0.6	0.4	0.9
P13	0.2	0.9	0.5	0.6	0.2	0.1	0.6	0.5	0.6	0.9	0.1	0.5	0.5	0.4	0.4

P14	0.6	0.8	0.2	0.6	1	0.9	0.8	0.8	0.8	0.2	0.7	0.5	0.8	0.4	1
P15	0.8	1	0.6	0.4	0.2	0.7	0.9	0.7	0.5	0.1	0.6	0.8	0.3	0.2	0.8
P16	0.5	0.6	0.1	0.6	0.9	0.1	0.5	0.5	1	1	0.5	0.5	0.5	0.9	0.7
P17	0.1	0.9	0.9	0.1	0.1	0.2	0.4	0.1	0.9	0.2	0.6	0.3	1	0.4	0.8
P18	0.2	0.5	0.2	0.3	0.6	0.9	0.6	0.7	0.8	0.6	0.2	0.5	0.5	0.2	0.5
P19	0.1	1	0.1	0.7	0.3	0.3	0.5	0.4	0.1	0.3	0.9	0.1	0.7	1	1
P20	0.4	0.1	0.6	1	0.7	0.5	1	0.2	0.8	0.3	0.3	0.7	0.7	0.8	0.8
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25					
P1	0.9	0.8	0.9	0.4	0.9	0.3	0.3	0.9	0.4	0.6					
P2	0.5	0.3	1	0.8	0.8	0.9	0.1	0.9	0.8	0.4					
P3	0.9	0.1	0.2	0.9	0.1	0.1	0.6	0.2	0.9	0.8					
P4	0.2	0.5	0.3	0.6	0.8	0.9	0.2	0.3	0.6	0.3					
P5	0.2	0.7	0.7	0.4	0.9	0.6	0.1	0.1	0.7	0.8					
P6	0.6	0.9	0.8	0.3	0.3	0.9	0.3	0.5	0.5	0.8					
P7	0.2	0.2	0.9	0.1	0.6	0.6	0.3	0.4	0.9	0.3					
P8	0.9	0.3	0.8	0.6	0.3	0.6	0.9	0.1	0.8	0.1					
P9	0.2	1	0.8	0.2	0.6	1	0.7	0.4	1	0.2					
P10	0.9	0.2	0.2	0.1	0.9	0.1	0.5	0.8	1	1					
P11	0.3	0.7	0.9	0.1	1	0.3	0.7	0.2	0.2	0.9					
P12	0.4	0.8	0.4	0.1	0.8	0.5	0.6	0.3	1	0.9					
P13	0.5	0.4	0.6	0.4	1	0.7	1	0.6	0.9	0.2					
P14	0.3	0.2	0.4	0.5	0.1	0.8	0.2	0.8	0.6	0.1					
P15	0.6	0.3	0.3	0.5	0.5	1	0.1	0.3	0.6	0.5					
P16	0.3	0.6	0.2	0.9	0.1	0.1	0.2	0.4	0.5	0.1					
P17	0.2	0.3	0.7	0.9	0.9	0.1	0.1	0.7	0.1	0.4					
P18	0.2	0.4	0.3	1	0.3	0.8	0.7	0.2	0.5	0.9					
P19	0.8	0.6	0.6	0.1	0.9	0.3	0.1	0.6	0.5	0.6					
P20	0.5	0.2	1	0.9	0.5	0.9	0.8	0.9	0.8	0.1					

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
D	50	80	50	20	70	140	20	130	80	60	40	90	140	20	20
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25					
	20	100	110	70	70	120	50	120	40	100					

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	190	200	160	60	50	180	110	90	50	140	140	140	80	100	60
	P16	P17	P18	P19	P20										
	10	10	10	10	10										
	80	100	90	90	190										

Table B.6: 25P30U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	13	9	15	8	20	12	6	8	7	9	17	12	18	14	13
P2	11	13	10	20	13	13	12	15	17	13	15	7	7	9	5
P3	18	7	20	16	11	18	20	9	5	16	8	16	12	19	5
P4	19	17	8	6	9	8	15	5	18	12	17	20	11	8	11
P5	10	5	12	10	11	11	9	5	18	6	9	19	6	14	5
P6	13	8	8	8	16	10	12	8	5	8	7	6	8	5	15
P7	13	6	9	6	10	8	14	13	18	16	14	13	5	6	17
P8	7	13	6	13	6	20	12	9	6	8	18	8	6	19	15
P9	13	14	14	15	19	6	15	15	16	9	13	16	15	10	10
P10	19	12	12	16	16	11	20	16	7	19	7	18	20	7	6
P11	19	20	14	17	9	17	5	8	17	19	13	11	16	8	5
P12	5	20	5	20	16	14	18	19	6	13	12	14	16	11	12
P13	12	9	12	20	19	5	12	12	7	11	13	11	7	9	9
P14	19	18	6	16	17	10	19	10	5	7	12	20	6	14	11
P15	6	16	5	7	15	6	11	9	15	6	14	16	6	19	13
P16	8	9	9	15	20	14	13	12	5	7	20	19	18	6	20
P17	10	11	19	9	12	5	12	19	8	19	20	11	16	14	16
P18	17	9	6	10	19	8	17	16	6	20	18	20	19	6	16
P19	19	16	13	12	12	12	19	12	10	9	12	6	14	18	16
P20	6	17	12	12	16	19	17	14	17	16	17	9	14	15	16
P21	18	17	8	20	14	7	12	13	9	19	9	11	18	14	20
P22	20	10	12	7	11	7	14	10	19	14	13	17	5	7	19
P23	14	7	8	9	12	11	15	7	16	10	12	8	9	6	15
P24	9	13	6	17	16	17	16	5	8	14	18	7	20	18	5
P25	5	13	8	12	7	16	20	10	6	16	20	20	6	15	7
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	8	12	12	20	12	18	20	18	20	15	17	17	9	19	15
P2	9	10	13	10	20	19	7	10	7	17	12	8	13	17	20
P3	10	19	18	17	7	14	5	14	12	10	7	11	14	11	17
P4	9	6	12	17	12	19	17	10	11	15	13	16	17	9	8
P5	15	6	17	12	15	14	9	20	8	17	13	16	8	6	6
P6	14	15	16	5	20	15	20	19	7	14	13	14	14	17	18
P7	5	14	12	12	6	10	14	18	9	5	9	5	7	19	13
P8	14	5	14	16	19	17	6	9	8	20	7	20	20	11	9
P9	12	7	9	15	11	9	18	11	14	9	8	16	13	15	9
P10	6	9	17	12	9	8	9	6	17	17	10	14	7	19	15
P11	8	8	20	15	6	20	16	8	15	11	10	17	14	13	19
P12	6	18	5	5	9	9	20	6	6	9	16	16	6	12	20
P13	13	8	17	20	11	8	5	17	16	7	8	17	12	13	7
P14	5	8	7	19	12	11	5	7	7	18	14	10	8	12	6
P15	12	10	16	11	7	9	9	6	12	15	7	13	8	14	15
P16	20	19	9	12	7	19	8	14	6	16	14	5	9	6	5
P17	20	10	12	9	18	7	20	12	11	9	16	12	20	17	11
P18	8	13	18	6	10	15	11	7	11	12	7	9	10	20	18
P19	7	16	15	12	6	19	5	8	18	10	7	20	19	9	5
P20	19	20	20	15	15	14	16	14	17	10	11	19	19	6	7
P21	13	7	11	19	8	15	19	14	19	16	13	11	9	11	7
P22	10	6	18	6	12	12	8	13	12	16	18	14	12	16	7
P23	9	20	5	15	11	13	19	18	12	13	12	12	11	14	18
P24	15	12	13	18	12	19	13	8	13	16	5	11	19	9	14
P25	20	14	14	15	7	12	6	14	16	9	6	11	18	5	18

Variable Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.2	0.9	0.7	0.9	0.2	0.5	0.9	0.8	0.9	0.4	0.6	0.5	0.7	0.1	0.7
P2	0.2	0.8	0.9	0.2	1	0.3	0.9	0.5	0.9	0.4	0.1	0.5	0.1	0.8	0.4
P3	0.5	0.2	0.5	0.8	0.6	0.1	1	1	0.7	0.4	0.8	0.4	0.6	0.5	0.1

P4	0.8	0.2	1	0.3	0.5	0.9	0.2	0.5	0.9	0.1	0.9	0.6	1	0.7	0.3
P5	0.7	0.8	0.1	0.8	0.1	0.4	0.9	0.2	0.2	0.4	0.1	1	0.2	0.3	0.2
P6	0.5	0.8	0.2	0.2	0.9	0.1	0.8	1	0.3	0.7	0.9	0.9	0.7	0.6	0.4
P7	0.9	0.3	0.9	0.8	0.3	0.4	0.8	0.7	0.9	0.6	0.2	0.9	0.6	0.5	0.9
P8	0.7	0.1	0.7	0.1	1	0.6	0.2	1	0.9	0.6	0.2	0.9	0.5	0.4	0.4
P9	0.5	0.4	0.7	0.6	0.1	0.8	0.1	0.9	0.1	0.7	0.4	0.4	0.6	0.4	0.5
P10	0.4	0.9	0.2	0.7	0.3	0.3	0.6	0.8	0.5	0.5	0.3	0.3	0.3	1	0.9
P11	0.5	0.3	0.5	1	0.7	0.2	0.8	1	1	0.6	1	0.6	0.6	1	0.7
P12	1	0.8	0.3	0.7	0.9	1	0.9	0.9	0.2	0.4	0.8	0.8	0.2	0.3	0.2
P13	0.9	0.3	1	0.8	0.6	1	0.8	0.5	0.5	0.9	0.3	0.6	0.2	0.8	1
P14	0.3	0.5	0.6	0.2	1	0.6	0.9	0.1	0.4	0.8	1	1	0.6	0.9	0.7
P15	0.2	0.9	0.8	0.2	0.2	0.5	0.3	1	1	0.8	0.1	0.2	1	0.9	1
P16	0.1	0.3	0.7	0.4	0.5	0.9	0.2	0.9	0.9	0.8	0.2	0.5	0.1	0.9	0.7
P17	0.9	0.7	0.8	0.6	0.2	0.3	0.5	1	1	0.1	0.1	0.3	0.9	0.8	0.1
P18	0.7	0.2	0.6	0.7	1	0.9	0.4	0.9	0.9	0.7	0.8	0.9	0.8	0.1	0.4
P19	0.6	0.8	0.7	0.2	0.6	0.6	1	0.8	0.3	0.6	0.4	0.3	0.6	0.4	0.2
P20	0.5	0.4	0.6	0.9	0.6	0.5	0.6	0.6	0.1	0.5	0.1	0.8	0.9	0.2	0.8
P21	0.6	0.3	0.3	0.5	0.3	0.5	0.4	0.8	0.9	0.4	0.1	0.3	0.8	0.8	0.4
P22	0.9	0.8	0.9	0.8	0.5	0.2	0.2	0.3	0.6	0.1	0.5	0.3	0.3	0.7	1
P23	0.9	0.6	0.9	0.3	0.8	0.9	0.4	0.8	0.8	0.2	0.1	0.3	0.4	0.1	0.1
P24	0.4	1	0.3	0.4	1	0.2	0.6	0.4	0.4	0.3	0.5	0.8	0.2	0.9	0.4
P25	0.9	0.6	0.2	0.6	0.6	0.9	0.4	0.8	0.6	0.8	0.8	0.4	0.1	0.4	0.1
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	0.1	0.1	0.1	1	0.1	0.1	0.1	0.4	0.1	0.1	0.6	0.4	0.7	0.2	0.5
P2	0.8	0.8	0.5	1	0.1	0.9	0.3	0.1	0.1	0.2	0.9	0.1	0.5	0.6	0.6
P3	0.2	0.6	0.2	1	0.9	0.7	0.1	0.2	0.4	0.3	0.4	0.5	1	0.4	0.8
P4	0.6	0.8	0.7	0.1	0.4	1	1	0.7	0.5	0.7	0.3	1	0.8	1	0.2
P5	1	0.6	0.8	1	0.9	0.7	0.8	1	0.3	0.2	0.6	0.1	0.9	0.7	0.7
P6	0.3	1	0.5	0.1	0.8	0.1	0.5	0.1	1	0.3	0.2	0.3	0.7	0.2	0.2
P7	0.3	0.6	0.5	0.5	0.8	0.3	0.6	0.1	0.4	0.6	0.2	0.6	0.3	0.5	0.3
P8	0.9	0.1	0.4	0.4	0.9	0.4	0.6	0.9	0.4	0.9	0.1	0.7	0.1	0.9	0.7
P9	0.3	0.3	0.4	0.3	0.1	0.8	0.7	0.3	0.5	0.9	0.3	0.8	1	1	0.7
P10	0.8	0.4	0.6	0.2	0.6	1	0.7	0.6	0.4	0.2	0.2	0.3	0.1	0.4	0.6
P11	0.8	0.5	0.2	0.5	0.7	0.7	0.8	0.9	1	1	0.2	0.2	0.3	0.5	0.7
P12	0.4	0.2	0.2	0.5	0.2	0.8	0.9	0.5	0.7	0.8	0.6	0.3	1	0.6	0.6
P13	0.4	0.6	0.2	1	0.1	0.6	0.9	0.5	0.1	0.5	0.4	0.7	1	0.5	0.3
P14	0.3	0.7	0.7	0.3	1	0.4	0.2	0.3	0.9	0.1	0.2	0.5	0.9	1	0.1
P15	1	1	0.1	0.4	0.4	0.2	0.6	0.8	0.6	0.2	0.1	1	0.7	0.8	0.4
P16	0.3	1	1	0.5	0.1	0.8	1	0.3	0.8	0.2	0.7	0.4	0.2	0.9	0.1
P17	0.5	0.3	0.4	0.3	0.4	0.2	1	0.5	0.7	0.4	0.1	0.1	0.6	0.4	0.9
P18	0.3	0.5	0.8	0.4	0.9	0.6	0.6	1	0.6	0.6	0.7	1	0.4	0.5	0.6
P19	0.9	0.3	0.1	0.5	0.3	0.8	1	1	0.9	0.5	0.2	0.5	0.9	0.5	0.1
P20	0.5	0.6	0.9	1	0.1	0.6	0.9	1	0.4	0.8	0.6	0.9	0.1	0.5	0.2
P21	0.9	0.1	0.8	0.5	0.9	0.2	0.9	0.2	0.6	0.7	0.3	0.5	0.9	0.9	0.6
P22	0.4	0.7	0.9	0.2	0.8	0.2	0.8	0.2	1	0.4	0.9	0.2	0.5	0.8	0.3
P23	0.2	0.7	0.6	0.6	0.3	0.2	0.9	0.2	0.2	0.6	0.4	0.8	0.8	0.5	0.8
P24	0.3	0.2	0.1	0.8	0.2	0.3	0.4	0.1	1	0.7	0.2	0.9	0.2	0.8	0.7
P25	0.6	0.9	0.3	0.2	0.8	0.9	0.2	0.7	0.6	0.1	0.9	0.5	0.3	0.1	0.1

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
	50	40	30	20	70	40	60	50	90	100	120	80	110	50	60
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
	130	30	110	30	120	130	120	140	60	90	150	90	60	110	130

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	100	50	90	140	180	180	70	130	150	60	170	170	140	90	120
	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25					
	10	10	10	10	10	10	10	10	10	10					
	180	60	200	180	70	70	60	180	160	170					

Table B.7: 30P35U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	8	9	18	12	11	9	17	16	9	13	17	6	14	7	8
P2	20	18	20	15	8	11	13	5	9	8	15	16	13	13	17
P3	20	5	5	5	12	18	10	7	16	14	7	7	17	10	14
P4	8	19	11	20	20	14	19	7	7	9	18	17	17	18	17
P5	8	15	16	8	14	6	9	19	7	8	5	17	6	17	11
P6	15	15	20	9	13	13	7	8	15	16	13	11	16	12	11
P7	11	15	13	8	20	7	8	17	17	15	17	14	20	13	11
P8	7	7	9	8	5	6	14	7	7	9	8	19	15	12	8
P9	7	7	9	10	6	15	9	14	6	5	13	12	6	16	15
P10	17	5	20	17	16	16	11	12	5	13	12	17	20	20	15
P11	9	18	7	5	15	19	6	10	13	19	8	14	17	19	7
P12	14	14	18	14	17	15	16	9	11	11	5	7	16	12	12
P13	6	19	6	10	9	7	16	9	10	8	11	19	8	12	6
P14	15	18	9	16	10	8	13	11	16	13	18	20	18	15	11
P15	8	11	10	18	8	13	19	13	18	19	17	13	17	19	18
P16	19	7	17	5	16	8	20	20	8	11	12	7	10	11	9
P17	15	13	15	16	17	9	12	10	17	17	15	19	11	5	5
P18	13	10	8	7	10	15	13	14	6	19	11	16	17	14	13
P19	11	17	8	9	6	5	14	16	7	12	15	17	17	6	5
P20	19	20	16	18	15	8	7	7	11	19	11	6	20	18	8
P21	15	10	7	5	10	14	7	18	10	17	13	18	18	18	20
P22	13	20	13	7	5	6	6	12	9	19	15	10	5	6	16
P23	10	12	6	5	19	10	10	9	10	13	11	16	18	14	14
P24	13	9	13	15	8	18	6	14	15	15	6	18	15	18	13
P25	10	12	14	11	20	8	14	19	13	15	11	8	8	11	9
P26	20	15	19	15	15	18	14	5	13	13	20	12	5	12	16
P27	6	20	16	16	13	6	18	5	5	5	14	11	5	18	17
P28	6	19	10	17	13	18	18	6	8	7	13	8	15	7	6
P29	16	18	18	15	7	11	20	13	13	14	11	11	15	7	15
P30	18	5	13	15	11	9	18	15	9	12	8	7	19	6	5
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	5	13	10	12	7	5	17	19	11	11	9	14	13	9	17
P2	7	14	14	13	7	14	17	10	14	9	14	20	9	10	7
P3	7	8	12	11	13	18	15	17	7	8	11	8	13	7	7
P4	18	15	8	12	17	5	8	10	11	5	20	20	17	6	15
P5	17	17	9	9	12	6	10	14	18	17	5	6	13	6	17
P6	19	16	20	6	10	18	5	9	5	20	11	16	6	11	14
P7	8	5	16	13	16	12	14	8	5	13	6	18	13	5	6
P8	6	9	16	14	9	6	9	17	11	15	13	13	13	6	16
P9	16	17	8	18	13	19	11	6	14	6	9	17	16	10	14
P10	14	10	8	18	20	15	6	5	11	16	14	19	19	11	14
P11	16	7	8	6	16	9	9	7	16	18	14	17	11	9	15
P12	5	6	19	7	6	16	10	5	14	6	8	14	5	11	12
P13	17	11	20	12	10	14	13	16	6	13	17	10	14	17	17
P14	15	9	10	10	17	7	17	18	13	7	17	14	6	15	5
P15	5	6	19	18	11	5	16	12	11	19	10	19	15	15	15
P16	13	20	14	8	13	10	13	13	15	7	12	16	7	18	13
P17	18	10	15	18	9	7	14	16	16	5	15	14	16	19	5
P18	17	12	19	12	20	11	20	8	11	16	17	9	19	14	12
P19	5	12	12	14	16	10	14	20	5	13	10	10	18	12	7
P20	16	5	20	18	12	7	19	8	9	9	17	8	17	13	12
P21	13	6	12	5	19	13	14	7	9	20	17	13	16	15	6
P22	18	7	6	12	16	19	20	19	14	6	18	11	13	10	6
P23	7	14	16	20	10	13	10	7	8	17	7	10	18	12	19
P24	12	11	18	15	6	7	8	7	5	10	18	20	13	18	19
P25	12	10	19	18	5	16	11	7	11	18	8	6	5	16	18
P26	20	8	12	9	5	6	18	12	6	13	18	7	18	9	15

P27	20	8	11	17	7	17	9	9	7	16	10	9	20	12	12
P28	20	5	12	14	12	16	15	16	19	14	19	18	13	11	16
P29	14	11	20	5	18	10	13	18	15	13	7	6	7	12	19
P30	19	17	5	20	20	5	15	16	15	11	14	17	6	14	7
	U31	U32	U33	U34	U35										
P1	6	18	13	20	5										
P2	11	15	13	5	12										
P3	9	9	14	9	18										
P4	14	18	10	15	11										
P5	6	11	13	13	20										
P6	20	14	5	18	9										
P7	10	8	7	19	12										
P8	13	11	5	16	17										
P9	12	18	7	15	13										
P10	5	7	8	18	14										
P11	5	15	17	7	15										
P12	12	11	7	17	18										
P13	5	5	20	18	12										
P14	12	7	6	5	7										
P15	18	11	8	6	17										
P16	7	14	11	5	10										
P17	7	19	14	7	7										
P18	6	15	14	20	18										
P19	7	13	9	5	19										
P20	14	10	16	18	6										
P21	20	16	16	17	12										
P22	7	16	18	17	8										
P23	13	12	15	6	19										
P24	7	16	12	6	9										
P25	11	12	10	16	12										
P26	19	7	10	9	10										
P27	15	9	17	12	11										
P28	13	14	5	14	14										
P29	12	18	5	7	7										
P30	5	7	18	8	5										

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.2	0.9	0.7	0.6	0.9	0.3	0.3	0.1	0.9	0.6	1	0.5	0.2	0.2	0.1
P2	1	0.7	0.4	0.9	0.9	0.6	0.6	0.9	0.5	0.3	0.1	0.4	0.4	0.3	0.6
P3	0.8	0.4	0.8	0.1	0.5	0.4	0.3	0.9	0.9	0.5	0.2	0.1	0.4	0.7	0.5
P4	1	0.6	1	0.6	0.7	0.9	0.2	0.9	0.9	0.3	0.4	0.2	0.6	0.9	0.9
P5	0.7	0.6	0.2	0.3	0.2	1	0.5	0.3	0.5	0.3	0.2	0.9	0.1	0.2	0.7
P6	0.2	0.8	0.8	0.9	0.1	0.2	0.3	0.5	0.5	0.2	0.9	0.2	0.5	0.9	0.4
P7	0.4	0.2	0.7	0.4	0.3	0.4	0.5	0.7	0.5	0.8	0.3	0.5	0.5	0.7	0.1
P8	1	0.7	0.4	0.3	0.8	0.2	0.8	0.9	0.6	0.2	0.4	0.5	0.5	0.6	0.6
P9	1	0.3	0.3	0.1	0.3	0.7	0.5	0.6	0.1	0.9	0.1	0.4	0.2	0.9	0.4
P10	1	0.7	0.5	0.6	0.5	1	0.6	0.5	0.2	0.6	0.7	0.6	0.4	0.7	0.3
P11	0.3	0.3	1	0.9	0.7	0.7	0.7	0.8	0.6	0.7	0.2	1	0.9	0.7	0.2
P12	0.4	0.1	1	0.5	0.4	1	0.6	0.1	0.3	0.4	1	0.7	0.4	0.4	0.1
P13	0.1	0.3	0.6	1	0.4	0.3	0.3	0.2	0.4	0.2	0.1	0.6	0.7	0.7	0.4
P14	0.9	0.4	0.6	0.3	0.5	0.6	0.8	0.4	0.3	0.7	0.1	0.2	0.6	0.8	0.1
P15	0.8	0.8	0.4	0.5	0.8	0.5	0.4	0.2	0.5	0.5	0.6	1	0.4	0.4	0.6
P16	0.6	0.4	0.7	0.7	0.7	0.2	0.9	0.8	0.4	0.9	0.2	0.9	0.5	0.4	0.3
P17	0.9	0.8	0.7	0.7	0.6	0.5	0.2	0.6	0.9	0.2	0.1	0.5	0.1	0.8	0.8
P18	0.2	0.4	1	1	1	0.5	0.9	0.7	0.4	0.7	0.1	0.4	0.4	0.6	0.9
P19	0.5	0.2	0.3	0.5	0.5	0.1	0.8	0.5	1	0.6	0.3	0.3	0.2	1	0.7
P20	0.4	0.6	0.4	0.5	0.9	0.9	1	0.3	1	0.5	0.4	0.3	0.3	0.3	0.4
P21	1	0.4	0.9	0.5	0.3	0.7	0.8	0.9	0.6	1	1	0.4	0.5	0.5	1
P22	0.4	1	0.7	0.8	0.5	0.1	0.4	0.6	0.2	0.8	0.4	0.3	0.9	0.4	0.5
P23	0.1	0.6	0.7	0.8	0.8	0.8	0.1	0.1	0.3	0.3	0.8	1	0.4	0.6	0.3
P24	0.7	0.5	0.9	0.8	0.2	1	0.2	0.8	0.3	0.1	0.4	0.4	0.3	0.9	1
P25	0.7	0.4	0.3	0.5	0.4	0.2	1	0.9	0.6	0.5	0.1	0.4	0.7	0.9	0.5
P26	0.2	1	0.1	0.7	0.8	0.8	0.1	0.4	0.8	0.1	0.8	0.8	0.4	0.4	1

P27	1	0.6	0.1	0.2	0.3	0.6	0.2	0.7	0.1	0.3	0.7	0.9	0.8	0.3	0.1
P28	0.1	1	0.9	0.8	0.2	0.5	0.9	0.2	0.3	0.1	1	0.7	0.2	0.3	0.3
P29	0.4	0.8	0.2	0.3	0.3	0.9	0.9	0.4	0.8	0.3	0.7	0.5	0.8	0.6	0.5
P30	0.9	0.9	0.3	0.2	0.9	0.4	0.8	0.5	0.7	0.7	0.2	0.8	0.2	0.9	0.7
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	0.1	0.2	0.4	0.4	0.8	0.2	0.5	0.7	0.2	0.5	0.1	0.1	0.8	0.2	0.8
P2	0.3	0.5	0.8	0.6	0.8	0.2	0.9	0.4	0.1	0.9	0.9	0.9	0.7	0.6	0.3
P3	0.8	0.6	0.3	0.8	0.1	0.7	0.7	0.1	0.2	0.2	1	0	0.2	0.3	0.6
P4	0.4	1	0.7	0.3	0.6	0.6	0.9	0.5	0.8	0.1	1	0.5	0.7	1	0.1
P5	0.6	0.2	0.1	0.1	1	0.9	0.2	0.2	1	0.2	0.6	0.2	0.1	0.3	1
P6	0.7	0.6	0.4	0.7	0.5	0.1	0.3	1	0.6	0.2	0.8	0.9	0.9	0.1	0.7
P7	0.8	0.1	0.8	0.8	0.6	0.4	0.8	1	0.1	0.4	0.9	0.7	0.4	0.9	0.9
P8	0.3	1	0.6	0.1	0.6	0.6	0.8	0.9	1	0.5	1	0.3	0.8	0.8	0.1
P9	0.9	0.9	0.4	0.1	0.1	0.3	0.8	0.4	0.6	0.6	0.6	0.2	0.7	0.5	0.2
P10	0.3	0.6	0.5	0.6	1	1	0.8	0.6	0.9	0.5	0.5	0.4	0.4	0.6	1
P11	0.3	0.9	0.6	0.4	0.3	0.7	0.6	0.4	0.3	0.9	1	0.8	0.7	0.4	1
P12	0.2	1	0.1	1	0.3	0.1	0.8	0.4	0.4	0.9	0.4	0.4	0.8	1	0.7
P13	0.3	0.9	0.9	0.2	0.4	0.7	0.8	0.6	0.9	0.6	1	0.2	0.2	0.9	0.3
P14	0.4	0.7	0.3	0.9	0.3	0.8	0.3	0.5	0.2	0.4	0.5	0.6	0.6	0.3	0.5
P15	0.4	0.4	1	0.7	0.4	0.2	0.1	0.8	0.1	0.6	0.8	0.6	0.4	0.7	0.9
P16	0.6	0.2	0.4	0.8	1	0.7	0.2	0.8	0.3	0.6	0.7	0.8	0.4	1	0.9
P17	0.8	0.2	0.8	0.5	0.4	0.8	0.6	0.3	0.9	0.6	0.1	0.7	0.8	0.1	0.2
P18	0.9	0.2	0.5	0.7	1	0.5	0.5	0.8	0.9	0.8	0.4	0.8	0.7	0.8	0.5
P19	0.5	0.7	1	0.1	0.6	0.9	0.3	0.1	0.2	0.3	1	0.9	0.6	0.5	0.4
P20	0.2	0.7	0.7	0.4	0.2	0.5	0.9	0.9	0.2	0.7	0.7	0.6	0.8	0.6	0.3
P21	0.2	0.7	1	0.4	0.3	0.8	0.4	0.5	0.2	1	0.3	0.1	0.8	0.3	0.2
P22	0.8	0.6	1	0.4	0.1	0.6	0.5	0.4	0.5	0.5	0.8	0.6	0.2	1	0.7
P23	0.4	0.2	0.9	0.2	0.9	0.3	0.7	0.2	0.7	1	0.9	0.4	0.5	0.2	0.5
P24	0.3	0.6	0.1	0.7	0.7	0.1	1	0.9	0.9	0.3	0.9	0.6	0.6	0.1	0.2
P25	0.4	0.6	0.9	0.1	0.8	0.6	0.4	0.5	0.2	0.2	0.1	0.5	1	0.6	0.3
P26	0.7	0.8	0.9	0.7	0.9	0.8	0.6	0.7	0.4	0.6	0.9	0.6	1	0.2	0.8
P27	0.9	0.3	0.1	0.2	0.1	0.7	0.4	0.5	0.1	0.7	0.9	0.4	0.1	0.7	0.1
P28	0.9	0.3	1	0.6	0.2	1	0.1	1	0.9	0.1	0.2	0.7	0.7	1	0.2
P29	0.4	0.3	0.4	0.3	0.7	0.8	0.2	0.6	0.4	0.4	0.6	0.6	0.3	0.3	0.3
P30	0.7	0.3	1	0.3	0.4	0.5	1	0.2	0.4	0.6	1	0.5	0.8	0.3	0.2
	U31	U32	U33	U34	U35										
P1	0.8	0.5	1	0.3	0.4										
P2	0.1	1	0.5	0.6	0.7										
P3	1	0.9	0.7	0.4	0.5										
P4	0.6	0.1	1	1	0.5										
P5	0.6	0.3	0.3	0.7	0.9										
P6	0.5	0.4	1	0.2	0.7										
P7	0.6	0.9	0.7	0.3	0.2										
P8	1	0.6	0.3	0.1	0.1										
P9	1	0.6	0.2	0.8	0.6										
P10	0.8	0.5	0.9	0.7	0.4										
P11	0.4	0.3	1	0.7	1										
P12	0.5	1	0.3	0.8	0.2										
P13	0.7	0.1	1	0.3	1										
P14	0.5	0.7	0.7	0.7	0.1										
P15	0.5	1	1	0.6	0.4										
P16	1	0.9	0.6	1	0.8										
P17	0.5	0.6	0.5	0.5	0.9										
P18	0.3	0.1	0.9	0.9	0.1										
P19	1	0.2	0.8	0.5	0.6										
P20	0.1	0.3	0.9	0.9	0.9										
P21	0.9	0.9	0.2	1	0.3										
P22	0.9	0.6	1	0.1	0.6										
P23	0.4	0.5	1	0.4	0.1										
P24	0.9	0.5	0.6	1	1										
P25	0.6	0.3	0.6	1	0.7										
P26	1	0.6	0.1	1	0.8										
P27	0.1	0.6	0.1	0.1	1										
P28	1	0.1	0.4	0.1	0.7										

P29	1	0.2	0.3	0.3	0.5
P30	0.1	0.9	1	1	0.9

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
	90	110	130	140	20	120	100	30	50	80	60	30	130	130	20
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
	140	120	100	50	100	140	40	140	60	30	30	140	80	100	100
	U31	U32	U33	U34	U35										
	90	120	110	60	120										

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	130	60	120	170	190	110	140	100	190	90	140	100	80	100	90
	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	60	160	130	80	170	100	100	90	160	70	180	170	120	160	120

Table B.8: 40P50U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	12	5	14	4	6	17	19	3	3	12	5	18	16	18	19
P2	10	12	18	18	10	19	6	15	11	7	8	19	16	16	7
P3	16	1	7	17	1	5	9	6	11	7	12	2	14	13	13
P4	12	13	17	16	1	3	10	19	2	3	2	20	3	3	12
P5	8	4	10	16	16	1	8	6	19	16	17	9	2	19	4
P6	13	12	15	12	14	15	1	3	17	1	3	4	2	18	12
P7	12	8	8	19	18	1	8	13	17	12	6	2	3	5	20
P8	14	1	17	4	5	20	2	7	14	12	7	8	6	20	18
P9	2	17	7	17	20	19	17	5	6	15	20	16	2	16	6
P10	14	15	13	14	7	9	14	15	5	16	6	10	19	9	7
P11	4	16	11	1	14	19	10	17	2	9	14	3	8	17	7
P12	15	19	2	16	9	3	4	2	9	18	6	9	7	6	14
P13	4	16	3	7	8	12	1	14	19	19	5	12	8	13	20
P14	13	2	19	8	7	16	17	1	6	19	20	17	1	13	10
P15	15	18	1	11	20	16	15	14	6	20	11	8	14	11	12
P16	14	1	7	5	2	12	19	9	19	2	5	9	13	5	16
P17	6	13	11	12	7	20	1	6	17	8	13	7	18	10	8
P18	7	18	19	10	10	7	3	6	11	13	12	20	4	11	10
P19	13	19	10	14	11	2	11	3	2	3	12	19	17	2	12
P20	10	7	14	8	3	2	2	14	18	19	14	5	19	4	7
P21	10	18	5	8	12	16	11	15	5	19	2	6	20	11	19
P22	4	8	11	19	1	12	14	16	20	19	7	7	20	2	14
P23	9	14	16	13	5	6	16	20	19	15	19	14	13	17	6
P24	5	20	19	1	3	10	7	6	3	18	3	8	20	16	14
P25	8	8	9	3	8	12	15	12	4	18	15	6	7	16	7
P26	11	13	3	18	4	5	12	9	4	12	18	15	15	12	17
P27	20	7	5	8	6	11	14	9	15	4	16	6	3	10	10
P28	15	4	8	6	14	13	10	19	16	2	4	11	10	20	1
P29	4	19	17	12	5	20	13	9	8	7	13	16	20	13	20
P30	8	20	14	20	19	14	6	11	18	4	2	19	5	19	20
P31	11	13	5	9	5	8	7	10	1	11	5	19	8	16	3
P32	11	18	17	20	9	3	5	2	19	17	17	19	8	15	18
P33	20	8	15	10	4	8	18	8	9	9	8	19	16	10	5
P34	14	2	1	6	16	9	20	10	7	4	6	2	8	4	13
P35	15	7	7	5	11	18	9	14	5	12	10	5	14	17	15
P36	11	19	3	12	1	10	12	2	6	19	13	14	18	16	17
P37	8	11	9	8	19	18	20	12	20	9	20	8	9	5	18
P38	8	1	8	2	9	18	9	18	6	9	6	6	18	9	13
P39	1	6	1	18	20	1	1	15	1	16	11	13	11	3	8
P40	16	18	4	14	18	1	19	6	1	20	20	14	18	13	8
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	3	3	1	7	9	8	19	8	18	3	7	12	16	13	3
P2	3	3	19	20	18	2	12	15	18	19	2	8	14	3	4
P3	11	3	11	9	7	4	16	8	1	4	20	20	1	4	2
P4	14	20	19	4	12	15	17	19	6	14	6	15	7	14	19
P5	12	8	5	7	12	12	12	20	19	14	16	19	7	4	15
P6	8	10	1	11	2	19	7	7	6	5	12	19	6	17	2
P7	5	12	1	2	7	14	5	14	19	2	8	12	5	5	17
P8	16	1	16	10	14	2	15	8	15	13	5	9	1	1	2
P9	16	20	3	5	14	16	13	4	6	3	6	15	19	17	15
P10	12	14	2	5	17	15	4	2	5	19	20	13	1	19	3
P11	12	2	8	16	10	9	15	9	19	7	18	17	9	8	16
P12	18	10	7	18	4	12	1	14	7	12	19	1	12	14	3
P13	14	12	7	16	7	6	9	10	5	2	17	9	8	4	3
P14	10	2	9	4	7	12	15	7	9	16	6	9	17	16	6
P15	17	18	6	20	3	6	8	3	10	9	17	18	7	17	9
P16	15	17	12	11	18	20	1	13	4	6	14	18	1	18	9

P17	7	6	9	10	19	11	4	1	15	7	4	16	18	15	9
P18	5	8	3	3	9	17	2	14	12	13	7	1	17	18	6
P19	17	6	9	16	19	9	1	3	16	3	6	13	18	7	16
P20	6	13	11	15	14	10	19	1	18	16	5	4	3	14	17
P21	9	9	12	3	17	12	6	2	15	3	20	3	4	15	7
P22	13	7	12	4	18	7	18	18	16	2	20	3	11	18	1
P23	11	10	6	14	5	7	10	9	5	3	1	14	20	6	7
P24	18	1	14	8	17	10	1	16	4	17	13	12	10	3	11
P25	9	10	3	11	1	15	9	16	9	12	5	14	1	1	16
P26	14	16	17	8	3	13	11	11	16	11	15	10	19	14	19
P27	12	10	20	4	9	8	6	2	14	8	8	16	14	11	16
P28	4	12	1	7	6	20	2	12	2	10	3	18	14	19	11
P29	11	16	11	11	16	12	4	4	17	5	6	19	12	13	4
P30	17	18	10	14	6	6	20	10	14	7	5	6	7	4	17
P31	14	2	2	16	17	6	12	19	4	15	2	18	1	10	12
P32	2	14	10	15	16	9	13	9	1	12	10	14	8	17	20
P33	9	5	7	15	18	1	18	4	5	8	9	1	8	11	19
P34	5	8	1	8	10	16	20	9	4	20	16	15	5	7	1
P35	7	20	9	5	20	5	7	19	12	18	18	14	11	11	5
P36	13	7	20	3	10	7	11	13	7	9	3	9	16	6	11
P37	19	14	7	11	20	11	1	11	4	12	10	18	10	11	16
P38	8	8	13	10	5	3	1	19	11	14	14	3	2	10	11
P39	13	6	2	5	4	16	1	11	11	7	18	18	12	20	1
P40	2	13	2	12	19	1	1	1	9	17	1	16	3	16	19
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	3	18	13	1	9	7	7	6	14	2	1	20	12	12	18
P2	12	19	9	3	12	13	9	12	18	17	14	12	18	9	9
P3	16	6	1	20	15	10	16	4	12	2	14	4	11	5	8
P4	17	12	14	6	6	8	12	8	12	4	13	4	19	13	5
P5	16	1	17	3	4	13	16	16	9	5	11	7	1	17	18
P6	16	9	6	7	5	13	3	20	8	6	12	5	8	11	9
P7	12	7	18	20	18	8	11	19	15	4	1	18	8	18	16
P8	4	17	6	15	1	1	4	18	3	13	13	8	14	16	18
P9	3	3	20	9	15	3	9	13	4	18	8	12	5	7	11
P10	8	8	15	7	5	16	1	15	6	6	11	14	15	19	13
P11	12	19	7	5	10	20	5	5	11	3	17	6	14	16	17
P12	9	18	11	12	14	14	1	16	14	15	4	7	9	7	10
P13	6	16	17	4	13	14	19	6	12	4	6	16	11	13	3
P14	18	19	19	18	8	12	1	8	16	17	7	14	9	16	7
P15	4	17	19	8	10	1	11	14	5	16	17	19	10	18	1
P16	6	17	5	8	8	15	7	7	16	3	9	12	10	4	6
P17	8	7	15	3	19	4	19	6	10	13	3	3	4	13	17
P18	14	18	8	20	9	3	13	17	6	20	15	17	17	15	2
P19	19	20	7	10	2	18	1	6	1	6	5	4	3	6	20
P20	7	7	10	12	4	13	20	13	19	8	8	14	17	19	14
P21	11	17	20	17	8	10	6	20	18	15	13	19	5	18	10
P22	19	18	16	13	18	11	4	8	17	16	20	1	13	4	13
P23	18	2	5	18	6	20	3	3	7	2	19	11	20	18	14
P24	10	19	14	16	13	1	18	15	17	18	8	17	12	19	10
P25	11	17	1	20	12	8	13	3	6	12	7	9	3	10	12
P26	14	18	18	4	19	12	1	13	20	2	17	18	1	8	12
P27	18	18	19	12	17	5	4	7	10	14	13	1	5	16	16
P28	11	14	5	18	8	6	10	14	3	10	11	16	15	15	4
P29	2	9	13	14	11	10	14	20	1	20	6	8	6	20	13
P30	4	16	13	6	17	17	5	1	10	2	10	18	10	15	3
P31	8	14	1	7	7	5	1	17	20	16	20	11	10	18	18
P32	12	5	5	14	9	1	16	16	1	11	5	1	14	8	19
P33	20	14	11	18	18	19	15	5	4	6	8	10	3	7	10
P34	3	8	7	5	11	20	8	11	14	17	4	5	17	2	8
P35	14	12	7	2	18	8	2	5	8	17	13	10	17	12	16
P36	7	8	2	3	13	18	19	20	15	5	11	11	14	4	6
P37	10	1	11	10	8	18	20	14	13	18	16	2	18	9	7
P38	7	16	11	15	20	6	8	15	17	10	18	14	20	8	18
P39	10	9	20	13	15	14	12	13	17	5	15	13	13	15	12

P40	1	1	12	9	16	9	2	5	5	3	2	15	19	8	20
	U46	U47	U48	U49	U50										
P1	20	20	2	9	12										
P2	10	2	13	18	2										
P3	9	8	6	14	11										
P4	3	15	11	20	14										
P5	12	15	14	5	6										
P6	20	2	18	18	4										
P7	20	6	6	10	5										
P8	10	2	12	6	19										
P9	5	9	12	16	8										
P10	4	9	20	20	9										
P11	18	2	10	17	15										
P12	14	6	5	1	2										
P13	7	7	6	3	2										
P14	15	5	10	10	13										
P15	16	5	15	15	12										
P16	15	6	17	6	16										
P17	11	19	15	15	6										
P18	20	2	10	12	1										
P19	20	12	4	1	10										
P20	16	20	10	14	14										
P21	11	12	11	3	17										
P22	15	8	6	12	17										
P23	5	5	3	15	18										
P24	3	15	14	20	18										
P25	15	19	12	7	19										
P26	17	17	18	2	13										
P27	9	19	18	14	7										
P28	7	2	14	3	2										
P29	11	10	13	17	13										
P30	10	12	7	19	11										
P31	11	13	19	7	5										
P32	3	12	19	15	3										
P33	17	20	19	5	15										
P34	19	12	2	7	5										
P35	15	6	5	20	14										
P36	18	2	10	13	17										
P37	8	7	2	15	14										
P38	5	13	4	9	15										
P39	15	7	6	18	8										
P40	16	15	7	3	4										

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.6	0.7	0.9	0.6	0.8	0.9	0.7	0.4	0.7	0.9	0.1	0.6	0.1	0.2	1
P2	0.4	0.3	0.2	0.7	0.3	0.4	0.4	0.1	0.9	0.4	0.5	0.4	0.5	1	1
P3	0.7	0.7	0.4	0.8	0.6	1	0.2	0.1	0.8	0.4	1	0.4	0.8	0.6	0.8
P4	0.1	0.9	0.7	1	0.5	0.3	0.1	0.8	0.9	0.9	0.5	1	1	1	0.8
P5	0.9	0.4	0.5	0.8	0.2	0.1	0.4	0.3	0.8	1	0.4	0.2	0.5	0.7	0.7
P6	0.5	0.7	0.2	1	0.1	0.2	0.4	0.4	0.6	1	0.3	0.8	1	0.7	0.9
P7	0.1	0.8	0.1	0.1	0.5	0.2	0.1	0.3	0.1	0.3	1	0.1	0.7	0.2	0.4
P8	0.4	0.9	0.7	0.6	0.5	0.8	0.4	0.1	0.5	0.5	0.6	0.2	0.4	0.1	0.2
P9	0.1	0.8	0.3	0.9	0.5	0.9	0.7	0.2	0.6	0.4	0.8	0.9	0.8	0.2	0.5
P10	0.3	0.4	0.8	0.7	0.3	0.7	0.6	0.2	1	0.2	0.6	0.2	0.6	0.8	0.7
P11	0.3	0.3	0.5	0.8	0.8	0.6	0.3	0.1	0.6	0.2	0.6	0.5	0.4	0.2	0.3
P12	0.3	0.7	0.9	0.6	1	0.9	0.6	0.6	0.6	0.5	1	0.7	0.4	1	0.2
P13	0.4	0.2	0.8	0.4	0.1	0.5	0.1	0.8	0.9	0.8	0.1	0.6	0.4	0.3	1
P14	0.4	0.7	0.4	0.3	0.6	0.2	1	1	0.2	0.3	0.3	0.8	0.1	0.4	1
P15	0.3	0.5	0.3	0.3	0.1	0.4	0.1	0.9	0.4	0.4	0.2	0.8	0.4	0.4	0.2
P16	0.8	0.1	1	0.8	0.9	0.3	0.3	0.8	0.4	0.3	0.6	0.4	0.2	0.6	0.1
P17	0.2	0.6	0.5	1	0.8	0.6	0.6	0.2	0.7	0.7	0.9	0.1	0.9	1	0.5
P18	0.5	0.5	0.4	0.5	0.7	0.5	0.7	1	0.3	0.7	0.2	0.3	0.5	0.2	0.6
P19	0.1	0.2	0.5	0.2	0.3	0.7	0.8	0.4	0.6	0.5	0.6	0.6	0.6	0.8	0.9

P20	0.7	1	0.4	0.2	0.8	1	0.5	0.4	0.8	0.4	0.8	0.7	0.6	0.6	0.9
P21	0.1	0.8	0.1	0.5	0.7	0.1	0.2	1	0.2	0.5	0.3	1	0.5	0.6	0.9
P22	0.1	1	0.7	0.7	1	0.5	0.8	1	0.6	0.5	0.3	0.8	0.3	0.7	0.3
P23	0.3	0.8	0.6	0.5	0.7	0.1	0.7	0.9	0.8	0.7	0.3	0.7	0.5	0.8	0.1
P24	1	0.9	0.9	0.7	0.6	0.7	1	0.5	0.1	0.5	0.3	0.1	0.9	0.2	0.1
P25	0.6	0.1	1	1	0.8	0.3	0.4	0.4	0.8	0.3	0.7	0.5	1	0.2	0.7
P26	0.5	0.5	0.9	0.2	0.4	0.8	0.4	0.4	0.7	0.3	1	0.2	0.6	0.9	0.1
P27	0.1	0.6	0.5	0.1	0.1	0.8	0.2	0.3	0.2	1	0.8	0.5	0.3	0.1	0.5
P28	0.3	0.6	0.6	0.9	0.3	0.7	0.1	0.3	0.8	0.6	0.8	0.3	1	0.9	0.8
P29	0.3	0.8	0.9	0.3	0.2	0.7	1	0.1	0.2	0.6	0.5	0.4	0.6	0.4	0.9
P30	0.5	0.8	1	0.5	0.1	0.9	1	0.3	0.7	0.9	0.1	0.3	0.1	0.3	0.6
P31	0.5	0.7	0.9	0.2	0.9	0.7	0.5	0.2	0.1	0.8	0.2	0.6	0.3	0.4	0.5
P32	0.3	0.3	1	0.3	0.8	0.1	0.9	0.3	0.6	0.1	0.8	0.6	0.5	0.8	0.8
P33	0.3	0.5	0.1	0.4	0.1	0.6	0.4	0.5	0.1	1	0.5	0.5	0.7	0.9	0.9
P34	1	0.9	0.3	0.1	0.1	0.2	0.2	1	0.8	0.4	0.4	0.8	0.8	1	0.6
P35	0.5	0.9	0.9	0.3	0.3	1	0.8	0.2	0.3	0.6	0.7	0.8	0.2	0.4	0.3
P36	0.5	0.1	0.5	0.8	0.1	0.1	0.4	0.9	0.7	0.7	0.2	0.3	0.8	1	0.6
P37	0.8	0.2	0.1	0.7	0.4	0.8	0.6	0.8	0.3	0.5	1	0.7	0.3	0.9	0.2
P38	0.8	0.7	0.4	0.9	0.5	0.6	1	0.5	0.8	0.2	0.2	0.3	0.4	0.6	0.6
P39	0.5	1	0.8	0.6	0.1	0.9	0.6	0.5	0.2	0.9	0.1	0.1	0.4	0.2	0.7
P40	0.8	0.8	0.1	0.6	1	0.9	0.3	1	0.5	0.9	0.6	0.5	0.9	1	0.5
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	0.5	0.5	0.1	1	0.6	0.9	0.8	0.2	0.8	0.1	0.5	0.8	0.4	0.7	0.4
P2	1	0.7	0.9	0.9	0.5	1	0.5	0.5	1	0.5	0.4	1	0.4	1	0.3
P3	0.3	0.1	0.9	1	0.9	0.3	0.6	0.3	1	0.2	0.3	0.1	0.3	0.4	0.5
P4	0.3	1	0.6	0.5	0.1	0.8	0.1	0.8	0.2	0.8	0.8	0.9	0.2	0.9	0.2
P5	0.7	0.6	0.6	0.2	0.6	0.1	0.3	0.9	0.7	0.1	0.8	0.1	0.3	0.6	0.7
P6	0.7	0.5	0.8	0.9	0.1	0.2	0.6	0.5	0.4	0.2	1	0.4	0.3	0.5	0.4
P7	0.7	1	0.4	0.6	0.9	0.3	0.1	0.4	0.6	0.2	0.5	0.5	0.8	0.1	0.5
P8	0.4	0.4	0.9	0.4	0.2	0.6	0.8	0.9	0.6	0.8	0.4	0.6	0.1	0.7	0.8
P9	0.3	0.8	0.1	0.4	0.9	0.6	0.6	0.2	0.2	0.1	0.1	0.5	0.2	1	0.9
P10	0.5	0.9	0.2	0.3	0.9	1	0.4	0.4	1	0.3	0.1	0.5	0.3	0.5	0.4
P11	0.4	0.4	0.2	0.5	0.1	0.1	0.4	1	1	0.2	0.4	0.7	0.6	0.4	0.6
P12	0.4	0.3	0.3	0.3	0.7	0.8	0.3	0.2	0.1	0.5	0.1	0.3	0.1	0.7	0.7
P13	0.7	0.5	0.3	0.3	0.7	0.8	0.2	0.7	0.1	0.3	1	0.2	0.4	0.3	0.4
P14	0.1	0.8	0.5	0.2	0.5	0.8	0.5	0.7	0.8	0.2	0.5	0.1	0.7	0.2	0.2
P15	0.1	0.5	1	0.6	0.9	0.8	0.1	0.9	0.7	0.5	1	0.8	0.9	0.1	0.4
P16	1	0.5	0.9	0.5	1	0.6	0.7	1	1	0.2	0.9	0.4	0.9	0.3	0.1
P17	0.3	0.2	0.5	0.6	0.5	0.7	0.1	0.2	0.6	0.3	0.6	0.2	0.4	0.6	0.1
P18	0.4	1	0.6	0.6	0.9	0.2	0.2	0.8	0.6	0.4	1	0.4	0.6	0.5	0.2
P19	0.5	0.6	0.7	0.4	0.3	0.4	0.2	0.1	0.6	0.6	0.5	1	0.2	0.4	0.5
P20	0.4	0.4	0.9	0.6	0.9	0.4	1	0.9	0.4	0.2	0.7	0.5	0.6	0.3	0.8
P21	0.1	0.8	0.4	0.9	1	1	0.9	0.6	0.9	0.3	1	0.8	0.1	0.2	0.5
P22	0.6	0.2	0.1	0.6	0.8	0.8	0.3	0.4	0.6	0.5	0.1	0.6	0.5	0.8	0.3
P23	1	0.6	0.5	1	0.1	0.9	0.6	0.8	0.4	0.4	0.5	0.8	0.4	0.3	0.9
P24	0.6	0.8	1	0.5	0.3	0.6	0.2	0.5	0.9	0.4	0.3	0.8	0.3	0.1	1
P25	0.2	0.2	0.5	0.8	0.5	0.5	0.1	0.5	1	0.8	0.2	0.3	0.3	0.1	0.4
P26	0.1	0.1	0.7	0.6	0.9	0.2	0.4	0.6	0.6	0.2	1	1	0.6	0.9	0.9
P27	0.5	0.9	0.9	0.4	0.7	0.8	0.6	0.6	0.6	0.7	0.6	0.3	0.3	0.1	0.5
P28	0.6	0.4	1	0.1	0.3	0.1	0.5	0.7	0.7	0.1	0.1	0.5	0.5	0.6	0.3
P29	0.1	1	0.8	0.1	0.4	0.8	1	1	0.5	0.6	0.7	0.1	0.6	1	0.1
P30	0.9	0.6	0.1	1	0.6	0.1	0.9	0.8	0.7	0.5	0.3	0.5	0.4	0.4	0.3
P31	0.2	0.4	0.7	0.8	0.9	0.2	0.1	0.7	0.1	0.2	0.5	0.3	0.7	0.5	0.1
P32	0.1	0.2	0.8	0.2	0.6	0.4	0.2	0.1	0.1	0.9	0.2	0.1	0.9	0.6	0.1
P33	0.5	1	0.6	0.7	0.9	0.5	0.4	0.4	0.4	0.7	0.2	0.9	0.1	0.4	0.8
P34	0.7	0.9	0.9	0.5	1	0.4	1	0.3	0.2	1	0.9	0.2	1	0.2	0.1
P35	1	0.1	0.8	0.3	0.9	0.3	0.1	0.4	0.6	0.3	0.9	0.1	0.5	0.5	0.3
P36	0.8	0.7	0.9	0.2	0.5	0.2	0.3	0.2	0.1	0.3	0.2	0.1	0.2	0.7	0.4
P37	0.3	0.1	0.7	1	0.6	0.7	0.1	0.1	0.9	0.7	0.7	0.4	1	0.9	0.6
P38	0.7	0.7	0.3	0.3	0.4	0.4	0.5	0.8	0.6	0.9	0.4	0.6	0.6	0.4	0.8
P39	0.4	0.7	0.4	0.2	0.8	0.6	0.2	0.8	1	0.7	0.8	0.7	0.1	0.9	0.5
P40	0.1	0.2	1	0.9	0.9	0.8	0.9	0.3	0.4	0.1	0.6	0.5	0.3	0.3	0.8
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	0.1	0.5	0.8	0.1	0.9	0.8	0.4	0.5	0.1	0.41	0.8	1	0.9	0.1	0.8

P2	0.1	0.3	0.3	0.9	0.2	0.1	0.7	1	0.5	0.8	0.8	0.4	0.5	0.5	0.6
P3	0.2	0.9	0.3	0.7	0.4	0.1	0.7	0.7	0.8	0.8	0.6	0.7	0.7	0.9	0.6
P4	0.4	0.7	0.5	1	1	0.5	0.5	0.5	0.8	0.3	0.1	0.5	0.3	1	1
P5	0.5	0.6	0.7	0.8	0.7	0.8	0.2	0.3	1	0.6	0.6	0.7	0.5	0.7	0.1
P6	0.2	0.9	0.8	0.3	0.2	0.5	0.5	0.6	0.4	0.6	0.8	0.4	0.9	0.6	0.5
P7	0.8	0.4	0.3	0.1	0.2	1	0.8	0.9	0.5	1	0.6	0.7	0.9	0.9	0.7
P8	0.7	0.8	0.3	0.2	0.7	0.4	0.8	1	0.7	0.6	0.7	1	0.5	0.4	0.7
P9	0.5	0.3	0.8	0.6	0.6	0.5	0.8	0.2	0.4	0.4	0.9	0.5	0.4	0.9	0.6
P10	0.5	0.8	0.7	0.6	0.3	0.1	0.1	0.1	0.3	0.5	0.5	0.7	0.9	0.8	0.6
P11	1	0.5	0.3	1	0.9	0.6	0.8	0.7	0.3	0.9	0.2	0.5	1	0.7	0.4
P12	0.8	0.1	0.4	0.8	0.7	0.2	0.9	0.5	0.9	0.9	0.1	1	0.2	1	0.8
P13	0.1	0.6	0.5	0.1	0.8	0.5	0.3	0.5	0.5	0.9	0.7	0.6	0.3	0.3	0.2
P14	0.8	0.5	0.9	0.5	0.6	0.6	0.9	0.2	0.2	0.9	1	0.4	0.8	0.4	0.2
P15	1	0.9	0.5	0.1	0.3	0.8	0.2	0.9	0.7	0.5	0.7	0.3	0.6	0.4	0.8
P16	0.9	0.2	0.7	0.2	0.3	0.5	1	0.8	0.1	0.8	0.5	0.6	0.6	0.7	0.9
P17	0.5	0.3	0.2	0.3	0.1	0.6	0.5	1	0.2	0.9	0.7	0.7	0.8	0.2	0.7
P18	0.4	0.4	0.6	0.7	0.1	0.4	0.9	0.9	0.7	1	0.4	0.4	0.3	0.5	0.2
P19	0.3	1	0.4	0.5	0.9	0.1	0.3	1	0.6	0.4	0.7	0.8	1	0.2	0.9
P20	1	0.6	1	0.1	0.4	0.8	0.6	0.1	0.4	0.6	1	0.6	0.3	0.4	0.7
P21	0.2	0.3	1	0.8	0.9	0.6	0.7	0.3	0.9	0.7	0.5	0.5	0.1	0.4	0.4
P22	0.1	1	0.5	0.2	0.4	0.8	1	1	0.6	0.1	0.7	0.1	0.4	0.8	0.4
P23	0.3	0.2	0.9	0.6	0.2	0.7	0.3	0.9	0.8	0.8	0.3	0.4	0.7	0.3	0.5
P24	0.6	0.7	0.9	0.3	0.2	0.8	0.1	1	0.8	0.7	0.7	0.7	0.7	0.4	0.2
P25	0.6	1	0.3	0.9	0.1	0.6	0.8	0.9	0.1	0.5	0.7	0.6	1	0.8	0.8
P26	0.7	1	0.8	0.9	0.7	0.8	0.7	0.9	0.7	0.1	0.2	0.9	0.7	0.4	0.1
P27	0.3	0.6	0.8	0.8	0.2	0.4	0.7	1	1	0.3	0.1	0.1	0.9	1	0.3
P28	0.3	0.9	0.4	0.5	0.4	0.7	0.3	0.2	0.2	1	0.6	1	0.6	1	1
P29	0.7	0.1	1	0.2	1	0.2	0.2	0.3	0.9	0.3	0.8	0.4	1	0.4	0.1
P30	0.1	0.9	0.9	0.9	0.4	0.2	0.8	0.2	0.1	0.3	0.8	1	0.4	0.9	0.4
P31	1	0.6	1	0.8	0.9	0.9	0.8	0.5	0.4	0.6	0.3	1	0.8	0.7	0.6
P32	0.9	1	0.3	0.4	0.7	0.3	1	0.2	0.2	0.2	0.2	0.7	0.9	1	0.4
P33	0.4	0.4	0.5	0.1	0.8	0.3	0.7	1	0.2	0.4	1	0.6	0.3	1	0.9
P34	0.2	0.7	0.3	0.8	0.5	0.5	0.4	0.4	0.9	0.4	0.7	0.4	0.4	1	0.8
P35	0.1	0.8	1	0.5	0.3	1	0.4	0.7	0.4	0.4	0.2	0.1	0.5	0.7	1
P36	0.3	0.3	0.1	0.1	0.2	0.5	0.5	0.7	1	0.3	0.1	0.7	0.9	0.7	0.2
P37	0.1	1	0.8	0.5	0.5	0.3	0.7	0.8	0.8	0.5	0.2	0.9	0.8	0.4	0.7
P38	0.8	0.1	1	0.4	0.5	0.1	0.9	0.4	0.4	0.4	0.4	0.4	0.6	0.2	0.6
P39	0.4	0.8	0.2	0.9	0.3	0.7	0.3	0.7	0.6	0.6	0.9	0.8	0.7	0.6	0.7
P40	0.9	0.1	0.6	0.4	0.2	0.1	0.3	0.2	0.1	0.5	0.9	1	0.7	0.9	0.7
	U46	U47	U48	U49	U50										
P1	0.1	0.9	0.2	0.4	0.2										
P2	0.1	0.6	0.8	0.4	0.8										
P3	0.8	0.7	0.6	0.1	0.3										
P4	0.8	0.8	0.4	0.5	1										
P5	0.9	0.8	0.8	0.9	0.9										
P6	0.4	0.1	0.4	0.8	0.2										
P7	0.7	0.8	0.6	0.6	0.5										
P8	0.7	0.9	0.9	0.1	0.6										
P9	0.9	0.2	0.8	0.7	0.1										
P10	0.5	0.6	0.2	0.7	1										
P11	0.5	0.4	0.6	0.9	0.1										
P12	0.5	1	0.2	0.7	0.4										
P13	0.6	1	0.8	0.3	0.6										
P14	0.2	0.9	0.9	0.8	0.6										
P15	0.3	0.5	0.8	0.1	1										
P16	0.8	0.3	0.3	0.2	0.2										
P17	0.2	0.8	0.1	0.5	0.5										
P18	0.6	0.2	0.3	0.3	1										
P19	0.9	0.7	0.6	0.7	0.6										
P20	0.5	0.9	0.3	0.5	0.2										
P21	0.1	0.3	0.1	0.7	0.6										
P22	1	0.4	0.2	0.6	0.4										
P23	0.8	0.2	0.2	0.6	0.3										
P24	0.1	0.7	0.1	0.7	0.5										

P25	0.1	0.8	0.3	0.7	0.2
P26	0.1	0.8	0.6	0.8	0.6
P27	0.3	0.5	1	0.7	0.5
P28	0.4	0.8	0.8	1	1
P29	0.2	0.7	0.6	0.5	0.3
P30	0.4	0.9	0.1	0.7	0.4
P31	0.7	0.6	0.7	0.6	0.5
P32	0.2	0.1	0.1	0.9	0.3
P33	0.2	0.5	0.9	0.8	1
P34	0.4	0.9	0.6	0.9	0.3
P35	0.1	0.4	0.5	0.5	0.6
P36	0.7	0.8	0.3	0.7	0.9
P37	0.5	0.4	0.7	0.4	0.9
P38	0.2	0.1	0.7	0.4	1
P39	0.4	0.8	0.7	0.1	0.9
P40	1	0.3	0.3	0.4	0.9

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
	140	130	80	30	70	30	100	20	100	140	90	130	70	30	90
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
	90	150	60	30	50	40	30	80	140	100	150	140	30	30	40
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
	50	150	40	30	60	130	40	30	40	130	20	80	40	150	130
	U46	U47	U48	U49	U50										
	120	30	130	30	100										

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	150	120	50	140	130	190	170	160	140	100	200	100	170	100	170
	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	120	140	170	120	140	100	100	60	110	120	110	200	170	80	140
	P31	P32	P33	P34	P35	P36	P37	P38	P39	P40					
	10	10	10	10	10	10	10	10	10	10					
	130	160	140	110	150	110	60	150	70	140					

Table B.9: 45P50U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	12	15	12	6	5	12	16	2	6	5	5	19	20	7	3
P2	14	18	3	12	10	19	4	6	18	10	18	11	11	20	6
P3	12	9	17	5	19	19	7	19	20	11	10	20	16	16	17
P4	8	13	10	19	6	11	2	7	15	10	18	5	2	17	14
P5	14	5	18	8	17	9	10	3	4	8	6	17	7	17	7
P6	1	20	3	19	6	17	7	9	8	15	11	12	17	18	15
P7	12	6	1	19	6	3	11	18	1	4	18	6	11	19	17
P8	6	12	5	5	19	5	19	13	17	13	13	5	3	7	1
P9	2	14	20	12	2	7	18	7	4	9	18	12	2	16	2
P10	4	9	1	13	18	5	19	6	11	10	11	11	1	1	11
P11	12	14	5	20	7	2	1	8	20	17	1	13	7	13	8
P12	8	16	6	15	18	8	6	2	15	17	8	16	10	7	6
P13	18	11	10	4	6	18	10	5	14	9	19	18	4	14	5
P14	6	17	4	15	9	15	19	8	7	18	16	12	17	5	4
P15	18	8	16	15	11	4	9	14	7	5	13	1	17	13	18
P16	15	4	11	20	13	20	17	4	13	15	17	5	1	15	12
P17	18	19	5	7	20	16	17	1	2	17	16	9	5	1	16
P18	7	12	3	18	15	11	15	4	5	15	9	15	18	2	11
P19	3	10	1	4	11	13	17	5	17	5	18	12	5	18	20
P20	2	5	11	2	20	14	5	19	3	9	2	4	5	1	9
P21	8	4	1	17	4	5	1	4	8	18	20	19	1	16	12
P22	20	5	14	9	11	9	1	2	9	10	4	15	2	16	2
P23	17	2	5	10	10	20	17	15	12	5	15	6	16	9	12
P24	1	4	10	15	2	15	20	19	3	9	2	5	10	5	11
P25	2	16	5	19	2	8	8	18	16	3	10	10	5	10	5
P26	14	14	14	17	11	7	4	9	18	6	2	7	11	4	13
P27	6	9	20	6	9	5	6	5	5	3	8	15	12	14	14
P28	18	2	10	10	4	17	11	16	8	6	6	7	14	13	19
P29	13	10	20	6	19	15	13	11	6	14	8	1	6	12	19
P30	3	1	10	15	7	5	16	16	17	5	17	16	2	6	1
P31	6	1	7	10	15	2	11	19	3	9	14	19	4	8	17
P32	20	14	17	11	18	2	2	19	10	10	10	13	18	12	15
P33	5	3	9	12	15	13	16	13	12	3	4	17	15	4	4
P34	17	20	16	16	3	5	12	8	17	17	8	16	12	20	18
P35	6	6	13	7	3	19	18	14	3	2	2	19	10	17	5
P36	19	10	12	14	19	2	19	3	13	1	17	12	7	17	12
P37	8	1	18	11	12	18	11	7	19	9	6	20	18	5	17
P38	13	14	1	6	7	14	8	3	12	2	6	8	11	14	15
P39	10	15	20	12	7	12	13	16	17	14	20	3	20	2	15
P40	14	20	15	13	8	10	3	18	9	5	6	1	9	3	8
P41	10	1	5	18	11	20	7	5	12	17	12	1	10	10	13
P42	8	2	11	2	15	19	9	13	17	5	13	14	20	11	5
P43	7	13	7	3	5	7	18	10	17	19	19	2	16	20	2
P44	13	2	4	14	3	18	2	11	6	14	4	11	2	19	2
P45	13	19	16	5	4	14	4	14	18	7	5	9	12	16	1
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	15	5	8	7	19	5	5	18	11	16	1	16	19	10	12
P2	6	16	19	11	3	4	20	8	14	19	15	9	20	8	20
P3	15	18	1	12	7	11	6	13	2	19	2	7	13	7	5
P4	6	1	10	14	3	7	1	10	14	19	2	17	19	13	18
P5	15	11	6	3	13	8	13	9	14	18	15	6	11	12	9
P6	19	11	20	9	6	3	5	14	8	16	9	17	13	11	18
P7	16	5	11	19	19	15	15	9	6	9	18	3	12	13	11
P8	6	17	1	18	2	19	3	7	11	19	12	3	14	8	14
P9	12	6	9	2	10	17	15	13	8	13	13	7	8	20	6
P10	16	5	10	4	8	11	8	14	19	10	10	12	10	12	10
P11	9	15	9	11	13	5	8	8	16	19	11	9	11	3	4

P12	1	16	1	20	11	9	7	13	10	14	11	16	19	7	13
P13	19	8	12	8	2	9	4	8	4	16	20	16	15	13	19
P14	4	18	13	6	15	5	9	5	7	15	17	10	13	18	13
P15	11	14	13	19	10	13	8	14	1	19	10	20	10	7	13
P16	3	16	20	7	2	12	18	20	11	2	18	13	19	18	2
P17	18	1	2	7	18	20	20	1	17	5	7	16	1	19	19
P18	14	2	1	4	5	3	5	9	8	8	7	1	3	6	13
P19	16	3	8	14	10	18	8	5	19	10	5	8	16	12	6
P20	10	12	7	20	2	14	17	2	20	16	7	5	8	7	20
P21	17	4	12	12	20	18	5	17	5	6	10	15	19	19	18
P22	5	3	5	1	12	15	20	11	14	2	10	13	7	1	9
P23	12	2	16	18	5	13	14	2	10	3	4	19	7	14	8
P24	4	11	9	3	8	6	3	13	1	8	1	8	4	10	17
P25	12	4	7	3	6	1	16	4	15	14	10	2	15	12	2
P26	1	6	1	9	13	17	2	11	4	17	9	20	3	14	5
P27	15	1	13	19	14	8	13	15	8	6	10	14	7	11	14
P28	5	20	20	8	7	1	14	18	14	5	5	6	10	12	6
P29	6	4	1	3	11	19	10	14	7	9	15	15	19	19	10
P30	4	16	12	13	7	11	5	7	20	12	10	6	1	19	12
P31	19	11	20	9	4	1	20	2	9	9	20	15	18	18	10
P32	12	3	12	19	19	19	11	8	18	7	7	17	1	8	3
P33	17	10	4	10	16	18	3	14	20	2	4	20	13	3	16
P34	2	8	4	9	19	19	18	7	3	4	1	6	6	13	20
P35	20	7	16	17	11	15	14	18	7	18	3	2	7	17	1
P36	14	17	20	11	20	11	6	8	10	18	18	15	4	6	15
P37	15	16	17	20	8	12	18	5	6	17	18	12	3	19	15
P38	4	20	3	8	12	15	17	1	2	11	8	11	10	12	2
P39	10	13	13	6	13	11	17	16	1	9	4	20	1	4	3
P40	18	8	7	19	1	19	20	2	4	16	1	17	10	8	8
P41	3	14	16	8	4	14	18	11	8	6	16	20	20	11	15
P42	5	8	10	6	2	3	5	7	7	3	5	5	18	7	6
P43	13	6	18	6	10	14	19	10	10	7	16	5	9	13	7
P44	20	15	13	11	6	20	6	3	5	15	3	8	18	14	12
P45	7	3	1	15	18	7	7	11	8	20	6	8	13	3	4
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	17	13	7	12	14	3	16	16	4	6	3	16	5	6	13
P2	14	14	4	6	11	2	8	16	6	2	6	18	6	14	8
P3	14	19	8	17	18	8	14	16	15	5	6	4	4	3	6
P4	12	16	19	12	5	6	9	7	18	16	15	5	14	20	10
P5	14	1	2	8	2	6	6	4	13	9	19	15	5	5	17
P6	13	1	19	16	6	13	13	7	5	3	10	14	2	4	18
P7	3	8	19	15	19	10	2	4	15	13	5	5	6	1	19
P8	9	9	17	7	19	1	2	20	1	15	13	14	1	16	19
P9	8	6	8	15	6	4	8	3	6	10	17	9	11	18	18
P10	4	16	15	20	6	9	4	17	16	10	11	7	20	14	7
P11	16	9	2	10	18	1	5	20	10	13	8	15	9	9	20
P12	3	8	1	18	14	20	13	5	10	5	18	2	1	6	1
P13	4	12	8	11	4	12	20	1	12	12	18	7	6	6	10
P14	18	12	15	11	11	1	15	4	19	14	20	6	10	20	16
P15	1	15	7	2	11	13	9	13	16	16	11	7	3	14	11
P16	15	10	9	1	16	15	14	18	16	16	19	5	8	11	20
P17	11	1	15	7	1	17	12	15	6	18	10	9	12	1	9
P18	6	4	14	12	4	1	17	1	10	20	7	8	18	4	16
P19	9	14	17	13	2	11	4	10	11	9	4	6	12	8	11
P20	17	2	18	14	6	8	6	5	14	2	7	2	6	15	11
P21	2	12	12	4	9	17	18	20	3	10	4	17	4	9	18
P22	12	5	10	12	20	15	11	5	8	4	10	10	16	1	7
P23	6	11	12	20	7	11	1	14	6	7	20	7	7	20	2
P24	18	6	10	9	20	17	2	16	4	11	3	5	7	19	4
P25	4	7	17	11	10	1	19	1	11	6	10	14	10	1	7
P26	7	3	7	15	5	7	19	9	12	10	9	7	6	16	9
P27	6	10	4	7	15	2	9	14	3	9	11	3	19	10	8
P28	3	4	7	6	17	2	18	14	1	9	15	15	11	8	9
P29	15	9	2	9	13	3	19	10	11	10	10	18	8	3	14

P30	18	3	6	13	19	9	18	13	10	15	19	20	13	3	8
P31	14	13	19	1	3	4	5	20	2	10	9	6	1	13	8
P32	18	5	17	20	19	10	17	1	15	19	19	9	17	1	16
P33	19	20	15	9	9	4	11	2	17	18	15	13	11	10	13
P34	8	9	9	18	4	4	15	11	18	13	14	16	6	5	4
P35	8	6	10	5	14	8	11	2	1	5	3	11	6	1	20
P36	20	14	11	16	4	14	13	4	19	15	12	1	1	11	13
P37	4	14	3	1	14	14	5	6	3	19	15	3	12	14	9
P38	2	7	2	7	15	6	18	17	10	12	1	8	18	16	14
P39	17	19	4	13	16	10	19	20	16	5	1	7	8	20	7
P40	20	18	10	15	15	12	19	18	10	17	3	2	1	7	20
P41	20	10	18	17	10	14	6	15	20	10	7	2	18	19	18
P42	4	5	10	17	11	20	15	9	11	11	11	9	11	17	1
P43	2	4	5	7	18	15	10	1	3	14	2	4	16	19	9
P44	14	5	19	9	10	14	1	19	12	11	12	15	3	10	20
P45	7	8	6	3	18	14	16	6	1	12	7	5	17	5	6
	U46	U47	U48	U49	U50										
P1	9	5	11	9	17										
P2	14	19	13	17	14										
P3	2	6	17	14	17										
P4	2	3	11	9	20										
P5	3	6	17	8	20										
P6	9	1	13	15	17										
P7	18	5	8	14	13										
P8	20	12	3	18	11										
P9	11	19	13	14	6										
P10	13	17	8	12	5										
P11	16	5	15	13	20										
P12	5	13	8	10	11										
P13	8	2	9	8	16										
P14	4	1	15	8	20										
P15	5	19	15	3	2										
P16	12	7	18	14	18										
P17	1	7	10	20	16										
P18	13	13	16	7	15										
P19	19	19	8	16	9										
P20	16	7	13	6	2										
P21	4	17	1	8	3										
P22	19	13	13	3	5										
P23	10	2	15	3	8										
P24	7	9	12	17	5										
P25	11	2	16	15	16										
P26	17	9	19	7	14										
P27	10	17	13	18	3										
P28	13	13	2	16	14										
P29	7	11	5	5	11										
P30	14	4	20	17	4										
P31	11	14	5	7	18										
P32	8	17	20	1	11										
P33	20	18	17	13	3										
P34	7	17	15	13	2										
P35	9	5	8	15	7										
P36	8	6	3	10	11										
P37	7	7	4	6	2										
P38	12	2	16	19	19										
P39	7	15	7	4	11										
P40	5	18	2	15	14										
P41	9	3	12	8	1										
P42	15	11	8	15	14										
P43	1	13	2	12	11										
P44	13	15	1	16	12										
P45	11	6	7	4	16										

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	0.2	0.3	0.8	0.9	0.1	0.9	1	0.1	0.3	0.9	0.1	1	0.6	0.1	0.3
P2	0.9	1	1	0.6	0.3	0.6	0.5	0.5	0.6	0.2	0.9	1	0.3	0.2	0.1
P3	1	0.5	0.9	0.9	1	0.4	0.4	0.3	0.9	0.3	0.2	1	0.4	0.1	0.8
P4	0.6	0.1	0.7	0.1	0.3	0.2	0.7	0.5	0.1	0.8	1	0.8	0.4	0.9	1
P5	0.3	0.9	0.9	0.7	0.1	0.7	0.2	0.8	0.7	0.8	0.9	0.2	1	0.5	1
P6	0.3	0.3	0.8	0.2	0.2	0.3	0.6	0.2	0.2	0.3	0.5	0.8	0.2	0.6	1
P7	1	0.2	0.6	1	0.7	1	0.3	0.7	0.4	0.3	1	1	0.5	1	0.9
P8	0.9	0.2	0.9	0.4	0.4	0.6	0.4	0.5	0.8	0.1	1	0.3	0.9	1	0.4
P9	0.3	0.5	0.2	1	0.4	0.7	0.5	0.8	0.7	0.1	0.2	0.9	0.3	0.8	0.6
P10	0.2	1	0.4	0.3	1	0.1	0.8	0.8	1	0.5	0.8	0.7	0.2	0.8	0.2
P11	0.3	0.4	0.9	0.5	0.6	0.5	0.5	0.9	0.3	0.2	0.8	0.5	0.2	0.4	0.8
P12	0.1	0.7	0.7	0.9	0.1	0.7	0.1	0.4	0.4	0.2	0.9	0.8	0.4	1	0.8
P13	0.3	0.3	0.6	0.5	0.8	0.9	0.9	0.6	0.1	1	0.6	0.9	1	0.5	0.3
P14	0.8	0.9	0.1	0.6	0.8	0.1	0.5	0.9	1	0.7	0.2	0.2	0.5	0.7	0.1
P15	0.1	0.4	0.2	0.1	0.7	1	0.5	0.1	0.8	0.9	0.1	0.8	0.5	0.8	1
P16	0.9	0.2	0.4	0.8	0.8	0.7	0.3	0.8	0.3	1	0.9	0.5	0.4	0.2	0.8
P17	0.5	0.9	0.3	0.9	0.9	0.5	0.7	0.6	0.3	0.6	0.9	0.7	1	0.4	0.2
P18	0.6	0.2	0.3	0.7	0.5	0.4	0.6	0.4	0.8	0.4	0.8	0.3	0.5	0.8	0.3
P19	0.3	0.1	0.8	0.1	0.3	1	0.7	0.9	0.1	1	1	1	0.3	1	1
P20	0.5	0.8	0.1	0.3	0.3	0.4	0.3	0.9	0.6	0.2	0.1	0.9	0.3	0.9	0.5
P21	1	0.2	0.8	0.7	0.9	0.1	1	0.4	0.3	0.1	0.1	0.6	0.7	0.5	0.1
P22	1	0.5	0.1	0.3	0.5	0.8	0.4	1	0.8	0.7	0.3	0.4	0.9	0.8	0.9
P23	0.2	0.5	0.2	0.9	0.7	0.1	0.4	0.9	0.5	0.9	0.9	0.7	0.3	0.6	0.9
P24	0.4	0.5	1	0.1	0.3	1	0.6	0.5	0.9	0.4	0.3	0.9	0.5	0.5	0.3
P25	0.4	0.7	0.6	0.6	0.1	0.6	0.7	0.4	0.9	0.2	0.6	0.6	0.9	0.9	0.7
P26	0.2	0.3	0.9	0.2	0.8	0.5	0.9	0.3	0.2	0.5	1	0.5	1	1	0.6
P27	0.8	0.3	0.6	0.6	0.2	0.7	1	0.6	0.3	0.9	0.3	0.1	1	0.8	0.8
P28	0.6	0.9	0.2	0.6	0.6	0.4	1	0.4	0.1	0.6	0.5	0.2	0.1	0.1	0.1
P29	0.5	0.6	0.2	0.2	0.1	0.4	0.5	0.3	0.4	0.4	0.5	0.3	0.9	0.6	1
P30	0.3	0.1	0.5	0.8	0.6	0.9	0.9	0.1	0.2	0.6	0.5	0.3	0.2	1	0.4
P31	0.9	0.5	0.1	0.5	0.8	0.1	0.8	0.6	0.9	0.7	0.1	0.3	0.6	0.1	0.8
P32	1	0.4	0.7	0.3	0.9	0.6	0.7	0.3	0.4	1	0.1	0.5	0.9	0.1	0.7
P33	0.5	0.3	0.7	0.1	0.8	0.9	0.6	0.1	0.7	0.4	0.5	0.8	0.5	0.7	0.8
P34	0.7	0.4	0.1	0.3	0.4	0.6	0.7	0.3	0.8	0.6	1	0.2	0.5	0.4	0.6
P35	0.3	0.2	0.2	1	0.1	0.9	0.3	1	0.5	0.7	0.8	0.2	0.6	0.2	0.2
P36	1	0.4	0.9	0.9	0.2	1	0.1	0.2	0.4	0.8	0.7	0.8	0.3	0.2	0.7
P37	1	0.2	0.6	0.5	0.4	0.4	0.1	0.4	0.1	1	0.8	0.3	0.2	0.9	0.8
P38	0.3	1	0.1	1	0.5	0.7	0.1	0.8	0.9	0.6	0.7	0.5	0.9	0.1	0.4
P39	0.1	0.5	0.1	0.2	0.1	0.5	0.9	1	0.7	0.1	0.5	0.5	0.3	0.8	0.1
P40	0.5	0.2	0.5	0.1	1	0.5	0.9	0.4	0.1	0.3	0.7	0.8	0.8	0.5	0.6
P41	0.3	0.1	0.3	0.8	0.5	0.3	0.6	0.9	0.2	0.7	0.8	0.7	0.3	0.1	0.1
P42	1	0.4	0.2	1	0.5	0.5	1	0.7	0.6	1	0.5	1	1	1	0.2
P43	1	0.8	1	0.9	0.9	1	0.4	0.4	0.5	0.9	0.6	0.4	0.5	0.6	1
P44	0.4	0.7	0.5	0.9	0.7	0.2	0.3	0.6	0.8	0.5	0.4	0.2	0.3	1	0.9
P45	0.9	0.3	0.1	0.7	0.4	0.9	0.7	0.9	0.7	0.9	0.4	0.6	0.2	1	0.4
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	0.7	0.3	0.5	0.8	0.2	0.1	0.5	0.1	0.8	0.9	0.7	0.6	0.4	1	0.6
P2	0.8	0.6	0.2	1	0.7	0.7	0.2	0.4	1	0.5	0.6	0.3	0.6	0.8	0.4
P3	0.5	0.1	0.9	0.3	0.6	0.3	0.4	0.4	0.8	0.5	0.1	0.6	0.9	0.7	0.2
P4	0.5	0.6	0.5	0.3	1	0.2	0.9	0.2	0.9	0.3	0.2	0.3	1	1	1
P5	0.3	0.8	0.7	0.2	1	0.4	0.9	0.9	0.1	0.6	0.7	0.7	0.7	0.7	0.1
P6	0.8	0.8	0.9	0.1	0.1	0.8	0.3	0.6	0.8	0.4	0.7	0.7	1	0.9	0.3
P7	0.5	0.2	0.1	0.9	0.8	0.5	0.9	0.1	0.7	0.6	0.4	0.1	0.5	0.1	0.2
P8	0.4	0.3	0.9	0.9	0.5	1	0.3	0.9	0.9	0.7	0.5	0.6	0.9	0.1	0.9
P9	0.1	0.3	0.8	0.8	1	0.2	0.6	0.1	1	0.1	0.8	0.6	0.9	0.9	0.9
P10	0.1	0.2	0.1	0.8	0.8	0.6	0.8	0.1	0.6	0.4	0.3	1	0.8	0.9	0.8
P11	0.5	0.5	0.4	0.1	0.2	0.5	0.4	0.9	0.9	0.1	0.8	0.4	0.1	0.1	0.8
P12	0.9	0.2	0.7	0.3	0.3	0.5	0.5	0.7	0.1	0.1	0.4	0.3	0.7	0.4	0.1
P13	0.8	0.3	0.2	0.2	0.5	0.2	0.5	0.8	0.8	0.1	0.9	0.1	0.5	0.3	0.2
P14	0.4	1	0.1	0.6	1	0.4	0.6	0.3	0.8	0.8	0.7	0.6	0.8	0.9	0.9
P15	0.5	0.5	1	0.5	0.7	0.9	0.6	0.3	0.7	0.6	0.6	0.5	0.5	0.7	0.9
P16	0.3	0.4	0.9	0.3	0.6	0.9	1	0.6	0.7	1	0.7	0.6	0.4	0.5	0.4
P17	1	0.1	0.5	0.2	0.3	0.8	0.7	0.9	0.3	0.3	0.7	0.7	0.4	0.3	0.1

P18	0.1	0.7	0.7	0.7	1	0.5	0.8	0.1	0.9	0.8	0.1	0.2	0.3	0.6	0.7
P19	0.3	0.3	1	0.3	0.9	0.4	0.5	0.7	0.1	0.7	1	0.4	0.7	0.2	0.1
P20	0.8	0.7	0.6	1	1	0.2	0.7	0.8	0.8	0.8	0.9	0.4	0.3	0.8	1
P21	0.5	1	0.2	0.5	0.3	1	0.6	0.1	0.3	0.6	0.5	0.1	0.7	0.7	0.3
P22	0.1	0.5	0.2	0.1	0.5	0.3	0.2	0.9	1	0.4	1	1	0.4	0.7	0.2
P23	0.9	0.1	0.8	0.1	0.6	0.4	0.1	0.8	0.5	1	0.2	0.9	0.4	0.7	0.6
P24	0.3	0.1	0.5	0.2	1	1	0.8	0.9	0.8	0.8	0.4	0.9	1	1	1
P25	0.8	0.6	0.3	0.6	0.7	0.5	0.9	0.4	0.6	0.4	0.1	1	0.4	0.9	0.7
P26	0.2	0.5	0.4	0.7	0.1	0.2	0.2	0.6	0.7	1	0.4	0.3	0.1	0.2	0.8
P27	0.7	0.1	0.8	1	0.5	0.5	0.3	0.5	1	0.4	0.8	1	0.6	0.8	0.2
P28	0.5	1	0.8	0.8	0.7	0.9	0.9	1	0.7	0.7	0.7	0.3	0.7	0.1	0.5
P29	0.6	0.5	0.7	0.2	0.9	0.1	0.7	0.3	1	0.9	0.3	0.2	0.7	0.9	0.4
P30	0.6	0.6	0.7	0.9	0.7	1	0.5	0.2	0.3	0.1	0.9	0.3	1	0.7	0.3
P31	0.3	0.3	0.1	0.9	0.5	1	1	0.2	0.1	0.6	0.1	0.2	0.2	0.9	0.9
P32	0.4	1	0.6	0.4	0.2	0.2	1	0.5	0.4	0.3	0.8	0.2	0.3	0.3	1
P33	0.4	0.9	0.4	0.6	0.1	1	0.6	0.1	0.9	0.5	1	0.7	0.1	1	0.8
P34	0.3	0.6	0.2	1	0.1	0.5	0.6	0.7	0.2	1	0.7	0.5	0.4	0.4	0.1
P35	0.7	0.2	0.8	0.9	0.4	0.5	0.5	0.9	0.8	0.8	0.1	0.9	0.3	0.1	0.9
P36	0.1	0.9	0.1	0.8	0.8	0.3	1	1	1	0.1	0.2	0.3	0.4	0.4	0.4
P37	1	0.2	0.5	0.5	0.4	0.3	0.5	0.3	1	0.2	0.5	0.9	0.7	0.7	0.4
P38	0.2	0.8	0.6	0.2	0.3	0.6	0.2	0.2	0.7	0.3	0.7	0.1	0.1	1	0.9
P39	0.4	0.9	0.8	0.8	0.4	0.3	0.8	0.5	0.6	0.1	0.2	0.4	0.3	0.4	0.9
P40	0.3	0.9	0.6	0.9	0.1	0.5	0.6	0.6	0.3	0.4	0.7	0.1	1	0.5	0.5
P41	0.3	1	0.8	0.9	0.4	0.7	0.7	0.5	0.3	0.4	0.7	0.7	0.1	0.7	1
P42	0.9	0.4	0.5	0.4	0.5	0.7	0.3	0.8	0.1	0.5	0.5	0.9	0.1	0.8	0.3
P43	0.8	0.4	1	0.2	0.7	0.9	0.6	0.9	1	0.7	0.5	0.6	0.8	0.3	0.1
P44	0.9	0.3	0.4	0.9	1	0.3	0.2	0.2	0.3	0.9	0.1	0.2	0.1	0.7	1
P45	0.9	0.6	0.6	0.9	0.5	0.5	0.1	0.5	0.1	0.3	0.7	1	0.3	0.4	0.5
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	0.1	0.1	0.5	1	0.8	0.9	0.8	0.6	0.8	1	0.6	0.3	0.5	0.8	0.8
P2	0.3	0.6	0.1	0.6	0.8	0.2	0.2	0.3	0.8	1	0.9	0.4	0.7	0.1	0.9
P3	0.5	1	0.9	0.8	0.6	0.5	0.3	0.7	0.8	0.3	0.6	1	0.8	0.6	0.1
P4	0.3	0.5	0.1	0.1	0.8	0.8	0.7	0.2	0.4	0.5	0.7	0.7	1	0.7	0.8
P5	0.5	0.9	1	0.8	0.8	0.2	0.3	0.3	0.4	0.4	1	1	0.2	0.5	0.7
P6	1	0.6	0.5	1	0.7	0.5	0.2	0.5	0.3	0.5	0.8	0.7	0.2	0.2	0.8
P7	0.3	0.1	1	0.2	0.4	0.2	0.9	0.4	0.3	0.1	0.1	0.7	0.7	0.7	0.9
P8	0.3	0.1	0.1	0.8	0.3	1	1	0.4	0.6	0.3	1	1	0.3	0.9	0.3
P9	0.8	0.1	0.7	0.3	0.4	0.1	0.7	0.5	0.1	0.8	0.1	0.8	1	1	0.7
P10	0.8	0.1	0.8	0.8	0.3	0.1	0.1	0.9	0.1	0.7	0.9	0.5	0.6	0.8	1
P11	0.9	0.4	0.1	0.9	1	1	0.7	0.5	0.8	1	0.8	1	0.2	0.6	1
P12	0.9	0.1	0.8	0.3	0.5	0.7	0.9	0.9	0.3	0.3	0.4	0.4	0.2	0.7	1
P13	0.8	0.7	0.8	0.7	0.1	0.3	0.2	1	0.4	0.6	0.1	0.7	0.1	0.3	0.9
P14	0.8	0.2	0.5	0.4	0.6	0.6	0.4	0.6	0.7	0.4	0.1	1	0.3	0.7	0.5
P15	0.4	0.6	0.6	0.5	0.1	0.4	1	0.5	1	0.1	0.8	0.3	0.6	0.8	0.1
P16	0.5	0.2	0.9	0.2	0.3	0.5	0.5	0.1	0.5	0.8	0.3	0.9	0.2	1	0.1
P17	0.5	1	0.1	0.6	0.7	0.1	0.4	1	1	0.5	0.7	0.7	1	0.6	0.6
P18	0.7	0.8	0.8	0.8	0.6	1	1	0.4	1	0.1	1	0.3	0.8	0.9	0.2
P19	0.3	1	0.7	0.8	0.1	0.1	0.2	0.7	1	0.6	1	0.9	0.4	0.4	0.8
P20	0.6	0.2	0.7	0.1	0.1	0.1	0.7	0.6	0.4	0.8	0.7	0.3	0.1	1	0.2
P21	0.5	0.4	1	0.4	0.7	0.1	0.8	0.2	0.3	0.1	0.4	0.2	0.1	0.6	0.7
P22	0.7	0.3	0.6	0.8	0.7	0.6	0.8	0.3	0.3	1	0.2	0.4	0.8	0.6	1
P23	0.5	0.5	0.5	0.9	0.1	0.1	0.1	0.7	1	0.2	0.1	0.6	1	0.4	0.9
P24	0.1	0.3	0.8	0.7	0.6	0.7	0.4	1	1	1	0.2	0.4	1	0.3	0.7
P25	0.9	0.5	0.7	1	0.1	0.3	0.3	0.1	0.6	0.8	1	0.9	0.3	0.6	0.3
P26	0.5	0.2	0.9	0.9	0.5	0.6	0.4	0.6	0.7	0.3	0.1	0.9	0.2	0.7	0.4
P27	0.5	0.2	0.7	0.7	1	0.8	0.2	0.3	0.6	0.7	0.5	0.7	0.2	0.6	0.6
P28	0.6	0.3	0.5	0.9	0.5	0.8	0.1	0.4	0.3	0.9	0.6	0.4	0.7	0.4	0.8
P29	0.1	0.9	1	0.4	0.7	0.1	0.5	0.1	0.6	1	1	0.7	0.2	0.3	0.1
P30	0.2	0.3	0.4	0.8	0.3	0.1	0.3	0.8	0.3	0.9	0.1	1	0.5	1	0.6
P31	0.4	0.7	0.9	0.4	0.4	0.5	0.4	0.7	0.9	0.4	0.6	0.4	0.1	0.4	0.9
P32	0.3	0.5	0.6	1	0.3	0.6	1	0.7	0.8	0.6	0.3	0.1	0.7	0.9	0.6
P33	0.6	0.4	0.1	0.8	0.1	0.4	0.1	0.4	1	0.6	0.7	0.9	0.5	0.3	1
P34	0.8	0.9	0.5	0.3	0.4	0.4	0.1	0.1	0.3	1	0.1	0.3	0.3	0.3	0.5
P35	0.8	0.2	0.3	0.1	0.7	0.3	0.5	0.1	0.6	0.5	0.7	0.5	0.5	1	1

P36	0.7	0.8	0.3	0.6	0.5	0.7	0.4	0.5	0.9	0.3	0.2	0.7	0.9	0.8	0.3
P37	0.1	0.9	0.8	0.4	0.1	1	0.8	0.6	0.6	0.9	0.4	1	0.4	0.7	0.5
P38	0.4	0.1	0.1	0.2	0.4	0.9	0.9	0.8	0.6	0.3	0.9	0.3	0.2	0.3	0.9
P39	0.8	0.4	0.1	0.8	0.8	0.8	1	0.9	0.6	0.3	0.7	0.4	0.3	0.9	0.3
P40	0.8	0.5	0.7	0.7	0.8	0.2	0.1	0.2	0.3	0.3	0.2	0.8	0.5	0.6	0.1
P41	0.9	0.8	0.5	0.5	0.8	0.1	1	0.1	0.9	0.9	0.9	0.9	0.5	0.1	0.9
P42	0.9	0.1	0.5	0.5	0.7	0.5	0.4	0.8	1	0.4	0.7	0.3	0.8	1	1
P43	0.3	1	0.1	0.2	0.4	0.3	0.3	0.6	0.2	0.4	1	0.5	0.7	0.8	0.4
P44	0.2	0.7	0.2	0.7	0.7	0.5	0.4	1	0.8	0.7	0.3	0.8	0.9	0.9	0.8
P45	0.2	0.9	0.7	0.2	0.5	0.2	0.4	0.4	0.7	0.8	0.4	0.2	0.3	0.2	0.9
	U46	U47	U48	U49	U50										
P1	0.5	0.9	0.7	0.3	1										
P2	0.7	0.2	0.3	0.8	1										
P3	0.8	0.1	0.7	0.7	1										
P4	0.9	0.2	0.7	1	0.1										
P5	0.7	0.1	0.7	0.4	1										
P6	0.7	0.4	1	0.6	0.8										
P7	0.4	0.9	0.7	0.2	0.5										
P8	0.2	0.6	0.2	0.2	0.7										
P9	0.8	0.6	0.6	0.3	0.2										
P10	0.7	0.5	0.2	0.4	0.8										
P11	1	1	0.4	0.7	0.4										
P12	0.3	0.3	0.5	0.5	0.1										
P13	0.7	0.7	0.4	0.6	0.4										
P14	1	0.7	0.5	0.3	0.3										
P15	0.6	0.3	0.5	0.4	0.3										
P16	0.1	0.7	0.8	0.8	0.2										
P17	0.7	0.4	0.5	0.7	0.2										
P18	0.5	0.9	1	0.6	0.5										
P19	0.2	0.3	0.6	0.6	0.1										
P20	0.1	0.6	0.9	0.1	0.9										
P21	0.7	0.5	0.4	0.7	0.3										
P22	0.8	1	0.3	0.4	1										
P23	0.9	0.1	0.7	0.1	0.6										
P24	0.6	0.6	0.2	0.8	0.3										
P25	0.3	0.8	0.1	0.7	0.6										
P26	0.2	0.3	0.8	0.3	0.4										
P27	0.9	0.4	0.5	0.3	0.7										
P28	0.5	0.7	0.7	0.5	1										
P29	0.3	0.6	0.1	1	0.6										
P30	0.9	0.3	0.1	0.7	0.5										
P31	0.6	0.1	0.5	0.3	0.9										
P32	0.2	0.9	0.6	0.1	0.1										
P33	0.1	0.1	1	0.5	0.2										
P34	0.5	0.3	0.1	0.1	0.3										
P35	0.3	0.7	0.1	1	0.4										
P36	0.2	1	0.1	0.5	1										
P37	0.8	0.6	0.3	0.2	0.5										
P38	0.7	0.8	0.2	0.5	0.9										
P39	0.9	0.4	0.2	0.5	0.5										
P40	0.3	0.4	1	0.4	0.3										
P41	0.8	0.8	0.4	0.6	0.8										
P42	0.7	1	0.9	0.3	0.2										
P43	0.1	0.9	0.5	0.3	0.9										
P44	0.7	1	0.1	0.3	1										
P45	0.2	0.3	0.7	0.5	0.2										

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
	120	110	30	80	40	20	120	110	50	70	110	120	120	130	40
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
	90	50	150	120	80	120	90	60	100	60	120	90	110	80	120
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
	60	40	120	30	70	50	130	20	20	130	110	110	110	20	130

	U46	U47	U48	U49	U50
	60	150	70	150	130

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	140	70	190	180	70	130	60	70	60	70	160	140	140	200	130
	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	170	70	140	190	120	200	170	120	80	120	110	140	80	60	170
	P31	P32	P33	P34	P35	P36	P37	P38	P39	P40	P41	P42	P43	P44	P45
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	70	60	50	180	60	80	70	180	100	110	120	80	90	200	120

Table B.10: 50P55U

Fixed Cost															
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	7	11	17	6	15	18	14	18	19	19	14	12	9	3	7
P2	9	16	4	7	13	12	2	18	15	1	15	6	16	3	5
P3	3	16	11	12	11	3	7	18	9	2	18	6	6	5	12
P4	3	15	7	2	4	4	14	8	2	14	18	2	8	8	11
P5	2	6	20	8	16	18	2	10	17	10	12	5	11	4	15
P6	1	16	10	1	2	20	13	5	19	9	1	19	20	14	15
P7	17	13	13	20	6	13	11	18	12	13	10	19	7	4	2
P8	17	4	9	16	13	6	8	13	7	13	11	11	13	19	5
P9	12	8	19	17	2	17	8	9	1	6	6	9	15	1	5
P10	16	9	6	13	20	13	5	16	1	16	5	4	20	13	2
P11	1	13	8	7	6	11	10	17	10	11	5	12	12	14	12
P12	20	3	19	3	5	4	16	12	14	18	13	19	6	16	3
P13	14	19	1	18	9	10	13	15	20	19	1	8	14	1	13
P14	17	5	10	1	13	9	16	8	18	9	8	2	20	19	10
P15	6	15	3	19	10	20	9	17	1	13	9	13	5	17	2
P16	13	6	20	9	5	1	16	9	2	1	5	16	17	3	13
P17	9	5	13	17	16	16	15	17	4	4	5	16	19	5	7
P18	15	8	1	4	18	12	1	4	4	2	16	5	16	11	19
P19	1	2	10	9	20	11	19	3	4	8	14	20	2	2	2
P20	12	4	11	2	9	7	16	16	12	15	2	7	1	19	6
P21	12	3	14	16	5	5	20	7	10	18	3	8	18	15	16
P22	18	18	12	10	1	6	12	17	15	7	9	11	4	9	9
P23	17	5	8	20	10	11	2	13	14	16	14	7	20	11	17
P24	10	15	19	12	9	16	14	13	3	8	14	17	2	6	1
P25	11	1	1	17	4	18	17	12	20	12	20	19	4	8	7
P26	4	14	12	11	13	12	2	20	1	11	18	5	1	9	5
P27	3	2	15	11	14	12	9	13	11	16	1	5	15	8	12
P28	4	10	12	10	3	12	16	11	17	2	17	13	11	2	19
P29	8	14	13	2	15	16	11	2	5	15	14	5	9	4	20
P30	2	2	10	8	11	9	17	9	19	10	15	7	11	1	5
P31	15	15	14	3	15	8	10	8	5	12	20	11	18	4	15
P32	3	8	9	16	15	20	19	10	7	8	3	3	19	13	10
P33	15	1	13	20	19	18	16	17	9	6	3	20	10	12	17
P34	12	1	14	17	18	7	12	18	4	8	10	7	2	7	4
P35	3	4	7	15	8	3	3	15	16	3	20	9	13	3	2
P36	5	6	4	7	17	12	14	15	5	17	7	10	7	14	20
P37	13	3	20	16	19	4	15	7	13	5	12	18	11	1	11
P38	9	19	14	16	6	7	18	16	3	13	9	19	20	6	16
P39	3	4	10	3	17	15	7	20	14	16	9	16	6	11	15
P40	19	10	17	4	3	1	12	17	15	3	13	3	20	12	3
P41	16	1	9	19	7	19	14	20	18	17	10	8	5	12	17
P42	9	4	17	18	13	3	13	9	9	13	12	7	5	9	6
P43	20	13	1	11	9	5	8	1	5	8	20	13	9	10	19
P44	10	11	18	8	9	9	11	1	18	8	4	13	6	19	12
P45	11	11	6	8	20	9	20	13	13	4	10	17	2	6	13
P46	15	8	1	19	3	3	1	1	6	8	4	9	10	7	15
P47	10	16	10	16	8	5	17	19	16	13	17	13	9	16	2
P48	6	5	17	9	13	11	2	12	13	15	7	13	13	4	8
P49	17	4	20	19	1	1	7	9	9	19	8	8	15	2	10
P50	1	20	17	15	12	18	3	6	20	17	17	10	1	2	3
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	10	11	16	19	13	19	10	6	2	4	7	19	1	20	4
P2	10	18	12	9	17	14	9	10	18	13	3	5	3	11	17
P3	12	18	5	7	12	19	11	14	14	20	13	10	13	20	9
P4	18	13	5	2	17	9	18	20	9	7	7	8	13	12	14
P5	10	13	1	7	9	2	14	1	7	2	8	8	7	18	5
P6	10	14	8	9	11	1	20	7	6	13	9	9	9	15	3

P7	14	13	17	10	2	3	18	2	17	10	5	20	12	5	18
P8	20	18	13	5	19	3	20	13	13	1	16	18	1	11	13
P9	9	2	18	9	7	17	14	11	4	18	17	8	10	13	14
P10	3	9	16	4	1	18	6	8	14	18	14	3	12	14	7
P11	16	4	17	20	14	3	12	3	19	15	18	4	10	7	6
P12	13	13	17	7	7	6	19	15	15	4	15	2	2	11	13
P13	11	13	3	10	9	19	9	4	17	9	8	15	13	2	3
P14	12	10	20	1	6	7	3	15	13	4	10	6	1	3	12
P15	5	10	19	2	9	5	17	11	16	5	16	1	15	20	7
P16	12	5	15	20	17	16	8	1	9	7	16	9	15	20	15
P17	14	1	12	13	10	1	20	1	1	3	7	15	3	11	2
P18	20	1	20	7	10	15	19	12	3	11	12	4	7	20	19
P19	14	19	2	12	5	20	9	11	11	12	8	3	13	11	11
P20	18	8	9	9	20	13	4	12	20	14	1	9	13	6	17
P21	5	16	5	10	1	11	8	18	4	1	12	12	12	7	12
P22	5	6	6	7	11	17	6	18	16	7	6	15	9	7	19
P23	8	10	15	12	5	5	8	13	13	2	20	8	6	5	2
P24	10	11	6	3	11	19	3	11	5	12	6	13	1	1	15
P25	17	16	13	15	18	18	19	14	14	2	12	15	11	18	18
P26	2	19	19	17	9	7	13	13	19	6	18	10	13	16	13
P27	18	16	1	11	15	6	15	16	18	2	4	20	17	14	3
P28	10	6	2	16	1	16	9	12	20	16	14	18	4	2	7
P29	11	6	18	16	15	1	19	15	10	11	8	8	4	17	19
P30	11	18	5	1	1	4	5	19	12	18	7	5	12	16	19
P31	16	4	13	1	18	3	14	6	4	8	12	3	7	7	19
P32	6	20	14	10	6	7	19	14	3	11	19	17	4	2	11
P33	15	8	10	19	8	11	19	18	6	10	3	17	17	9	9
P34	16	15	5	16	8	17	8	10	8	12	6	10	13	20	9
P35	17	16	18	2	9	16	20	3	8	15	8	12	14	17	15
P36	7	4	4	17	17	1	7	4	8	1	2	6	9	15	16
P37	10	15	10	2	10	11	18	4	11	12	9	9	7	8	11
P38	20	2	16	8	15	3	5	8	16	20	9	7	7	4	1
P39	1	4	12	17	6	14	13	13	3	16	2	7	17	19	5
P40	8	16	8	8	12	5	20	5	5	17	15	18	11	10	18
P41	14	14	15	2	2	19	16	3	3	5	15	7	5	14	6
P42	12	3	15	17	20	13	12	12	18	19	17	3	15	20	16
P43	11	10	17	11	14	20	16	10	7	17	2	17	16	7	16
P44	6	7	16	3	18	12	3	2	11	8	9	7	1	6	2
P45	4	18	3	18	20	2	3	10	16	13	3	7	9	13	17
P46	3	12	13	3	9	1	15	10	19	5	11	8	10	6	20
P47	5	3	10	13	17	19	16	7	5	4	4	3	13	19	4
P48	8	8	10	7	15	13	14	4	19	13	6	14	16	3	7
P49	14	9	1	15	6	12	1	14	3	3	11	16	1	2	1
P50	7	4	3	1	8	1	12	19	5	19	13	13	9	8	16
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	9	8	16	11	12	15	6	20	5	11	9	11	1	20	2
P2	5	1	3	10	18	18	20	16	18	8	15	6	16	5	9
P3	20	11	11	8	17	11	17	10	3	3	16	6	12	3	9
P4	20	6	4	19	6	8	13	7	4	5	5	2	3	11	14
P5	2	7	4	15	16	7	16	16	3	7	6	8	18	16	5
P6	8	16	3	14	5	15	6	10	18	17	10	7	11	12	19
P7	15	18	4	11	10	15	20	14	14	18	11	10	5	12	16
P8	6	8	20	7	12	5	3	20	5	3	2	6	16	9	11
P9	16	9	13	6	9	5	4	5	19	12	14	18	10	7	9
P10	11	9	11	8	14	18	18	11	20	12	1	4	10	13	3
P11	1	8	3	3	8	13	16	9	4	8	20	10	18	4	9
P12	5	3	5	18	18	14	18	20	19	18	11	17	5	8	14
P13	8	12	15	20	10	1	16	7	6	17	2	8	13	1	7
P14	18	19	14	5	11	17	14	1	6	4	5	2	1	8	5
P15	7	12	15	7	17	17	10	9	3	12	5	9	5	14	17
P16	3	7	16	6	16	5	8	19	20	16	13	2	7	5	11
P17	14	6	10	13	1	18	20	4	4	5	12	18	7	10	10
P18	5	17	6	17	9	13	11	11	1	9	1	1	10	11	20
P19	3	7	11	15	15	15	9	12	6	13	13	19	13	20	19

P20	6	5	5	6	17	12	11	13	14	12	19	10	17	19	13
P21	8	19	18	20	8	19	10	14	11	20	18	13	7	7	19
P22	14	16	6	13	4	4	14	5	16	2	5	8	13	9	13
P23	12	13	5	5	2	11	8	2	13	20	10	9	20	11	14
P24	3	16	15	3	1	2	8	9	1	17	16	18	15	12	16
P25	10	8	13	20	18	15	8	17	11	19	3	9	19	18	16
P26	15	13	14	16	20	19	18	13	3	18	9	16	19	16	15
P27	15	11	19	5	12	4	8	5	16	3	8	9	20	11	20
P28	4	13	10	15	2	3	11	18	11	10	1	14	13	6	9
P29	16	5	17	1	12	10	5	11	3	17	8	1	4	15	14
P30	15	3	20	17	6	15	5	14	13	18	11	20	2	9	16
P31	14	12	16	4	1	1	13	2	20	8	2	17	13	8	8
P32	15	12	7	16	11	15	10	9	6	5	17	5	10	19	14
P33	3	10	12	16	6	4	15	12	12	7	16	15	17	1	9
P34	18	7	20	12	8	3	6	16	11	14	4	2	5	17	8
P35	7	13	12	18	15	12	18	12	6	1	17	8	6	20	11
P36	4	14	9	15	15	5	4	18	20	19	2	5	20	11	4
P37	2	19	11	18	15	2	15	11	15	18	8	18	16	20	6
P38	18	9	17	8	9	12	4	12	15	8	17	7	13	4	8
P39	14	15	18	7	15	16	8	15	17	12	1	17	12	11	10
P40	6	18	3	8	9	11	16	2	8	12	20	7	14	1	17
P41	9	14	4	10	11	13	9	9	14	19	1	19	9	1	7
P42	5	14	15	3	7	4	1	8	15	8	20	10	10	12	19
P43	14	7	14	8	7	1	10	16	10	11	1	5	2	20	19
P44	19	12	14	9	2	4	4	5	15	17	16	11	20	9	12
P45	8	3	18	4	11	20	10	12	9	6	9	1	2	18	8
P46	9	7	16	13	6	9	18	7	14	16	14	15	4	14	3
P47	11	5	15	2	10	3	19	13	10	12	20	2	10	2	7
P48	18	2	15	6	10	15	13	14	19	7	15	2	18	15	18
P49	5	7	19	16	20	16	11	13	5	3	1	16	6	5	12
P50	9	16	2	18	19	13	10	18	8	6	10	7	1	10	16
	U46	U47	U48	U49	U50	U51	U52	U53	U54	U55					
P1	11	15	13	5	13	4	7	13	17	1					
P2	13	12	5	14	3	11	6	12	6	16					
P3	2	16	12	16	20	2	15	20	3	10					
P4	15	19	1	12	12	14	16	17	11	7					
P5	10	6	9	14	17	17	9	14	4	7					
P6	6	6	13	18	15	5	3	15	16	6					
P7	7	9	11	4	20	6	4	19	14	9					
P8	10	2	20	9	6	19	5	4	6	14					
P9	3	11	20	7	7	17	4	4	4	19					
P10	5	20	7	9	3	11	2	9	2	12					
P11	19	20	10	7	5	7	18	2	5	1					
P12	4	4	12	1	4	11	9	9	17	4					
P13	20	14	2	6	3	6	16	15	13	1					
P14	3	9	6	18	1	20	20	20	17	1					
P15	6	8	12	7	19	5	9	7	8	6					
P16	14	5	14	3	18	4	13	13	16	11					
P17	12	15	15	6	3	3	3	8	15	8					
P18	16	4	13	15	4	12	19	10	16	13					
P19	1	19	5	20	15	15	18	9	20	16					
P20	18	5	10	8	9	10	2	12	1	1					
P21	3	6	18	15	13	6	15	18	11	12					
P22	9	9	15	18	6	1	13	17	16	10					
P23	11	17	17	5	2	10	16	6	19	10					
P24	5	2	5	4	13	13	5	3	17	13					
P25	3	8	3	17	2	8	6	15	11	11					
P26	14	12	9	3	9	9	9	13	5	1					
P27	8	5	20	15	16	20	1	4	19	19					
P28	19	16	3	2	13	4	18	9	18	20					
P29	5	18	6	9	3	18	18	1	11	1					
P30	18	8	20	11	19	10	9	7	6	18					
P31	3	9	19	3	13	8	5	8	17	15					
P32	11	4	10	13	13	8	12	11	18	4					

P33	2	2	12	3	13	4	3	9	2	20									
P34	5	2	4	20	10	20	3	14	17	9									
P35	8	10	19	4	8	2	1	10	17	6									
P36	11	16	3	11	14	19	12	20	11	16									
P37	14	8	3	12	9	2	16	3	2	10									
P38	1	13	6	15	20	14	17	15	19	10									
P39	15	4	2	6	4	7	16	18	4	16									
P40	14	20	10	12	19	3	3	15	16	4									
P41	5	12	16	6	5	11	18	10	15	7									
P42	5	14	9	8	11	4	9	19	10	15									
P43	20	13	2	20	3	1	20	14	6	4									
P44	9	4	5	19	3	1	3	10	2	7									
P45	9	18	15	10	17	13	17	4	17	19									
P46	20	3	10	20	1	10	18	5	14	11									
P47	16	15	7	17	7	5	6	16	18	12									
P48	11	15	15	7	11	10	7	11	4	19									
P49	17	1	11	20	15	1	1	5	2	7									
P50	14	3	15	6	1	2	14	6	15	19									

Variable Cost

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
P1	1	0.3	0.7	0.5	0.9	0.8	0.5	0.1	0.9	0.5	0.7	0.8	1	0.8	0.2
P2	0.7	0.4	0.9	0.9	0.6	0.5	0.9	0.9	0.7	0.9	0.7	0.4	0.3	0.4	0.6
P3	0.3	0.4	0.8	0.7	0.5	0.6	0.8	0.1	0.7	0.1	0.5	0.4	0.9	0.1	0.8
P4	0.1	0.4	0.7	0.8	0.7	0.1	0.5	0.5	0.4	0.2	0.7	0.7	0.8	0.5	0.6
P5	0.2	0.9	0.5	0.9	0.8	0.7	0.4	0.2	0.2	0.2	0.5	0.9	0.5	0.9	0.5
P6	0.9	0.4	0.5	0.6	0.6	0.8	0.6	0.8	0.5	0.2	0.8	1	0.9	0.8	0.5
P7	0.4	0.5	0.5	0.6	0.7	0.2	0.9	0.8	0.8	0.9	0.7	0.3	0.3	0.7	0.6
P8	0.1	0.4	0.7	0.2	0.7	0.3	0.6	0.6	0.5	0.6	1	0.4	0.5	0.4	0.5
P9	0.3	0.8	0.6	1	0.8	0.3	0.5	0.2	1	0.4	0.1	0.8	0.9	0.3	0.3
P10	1	0.8	0.7	0.2	0.1	0.1	0.3	0.9	1	0.1	0.9	0.7	0.6	0.3	0.9
P11	0.9	1	0.2	0.9	1	0.2	0.2	0.8	0.8	0.7	0.8	0.7	0.9	0.3	0.5
P12	0.2	0.2	0.6	0.8	0.4	0.4	0.3	0.9	0.7	0.3	1	0.7	0.3	0.7	0.7
P13	0.2	0.2	0.5	0.5	0.1	1	0.2	0.4	0.4	0.2	0.9	0.4	0.8	0.5	0.4
P14	1	0.3	0.1	0.5	1	1	0.6	0.7	0.8	0.2	0.1	0.6	0.1	0.5	0.2
P15	0.1	0.6	0.6	0.7	0.4	0.3	0.5	0.3	0.7	1	0.4	0.6	0.2	0.2	0.3
P16	0.6	0.8	0.9	0.7	0.3	0.2	0.7	1	0.9	1	0.4	0.3	0.6	0.2	0.3
P17	1	0.7	0.1	0.6	0.4	0.3	0.2	0.1	0.1	0.8	0.1	0.9	0.4	0.8	1
P18	0.9	0.6	0.2	0.4	0.3	0.1	0.4	0.7	0.1	0.8	1	0.8	0.8	0.3	0.2
P19	0.8	1	0.7	0.8	0.4	0.9	0.4	0.1	0.8	1	0.2	0.5	0.1	0.5	0.9
P20	0.4	0.4	0.5	1	0.4	0.3	0.6	0.6	0.2	0.5	0.5	0.9	0.9	0.9	0.7
P21	0.9	0.2	0.2	0.1	0.9	0.2	0.9	0.6	1	0.2	0.1	0.4	0.5	0.5	0.5
P22	0.7	0.5	0.5	0.8	0.2	0.1	0.4	0.7	1	0.5	0.2	0.5	1	0.4	0.4
P23	0.9	0.5	0.8	1	0.4	0.8	0.7	0.7	0.1	0.7	1	0.7	0.8	0.1	0.8
P24	0.4	0.9	1	0.4	1	0.5	0.1	0.3	0.7	0.5	0.1	0.2	1	0.4	0.2
P25	0.6	0.9	0.7	0.1	1	0.2	0.9	0.7	0.9	0.2	0.7	0.2	0.5	0.7	0.2
P26	0.1	0.4	0.3	1	1	0.4	0.3	0.1	0.3	1	0.4	0.6	0.2	0.8	0.2
P27	0.5	0.5	1	0.7	1	1	0.9	0.7	0.4	0.6	0.7	0.8	0.1	0.2	0.9
P28	0.9	0.2	0.9	0.3	0.1	0.4	0.6	0.2	0.9	0.4	0.5	0.5	0.3	1	1
P29	0.5	0.2	0.7	0.6	0.4	0.8	0.9	0.4	0.6	0.8	0.1	0.9	0.7	0.9	1
P30	0.8	0.4	0.4	0.5	0.2	0.9	0.1	1	0.7	0.1	0.7	0.8	0.8	0.9	0.6
P31	0.2	0.6	0.6	0.5	0.4	0.6	0.9	0.3	0.4	0.1	0.4	0.9	0.7	0.9	0.1
P32	0.1	0.4	1	0.7	0.7	0.5	0.9	0.9	0.6	0.1	0.3	0.7	0.2	0.8	0.6
P33	0.4	0.3	0.5	0.1	0.3	0.3	0.6	0.9	0.3	0.6	0.6	0.2	0.1	0.4	0.4
P34	0.2	1	0.6	0.3	0.5	0.9	0.2	0.4	0.9	0.3	0.1	1	0.2	0.4	0.5
P35	0.2	0.5	0.7	0.4	1	0.8	0.5	0.2	0.5	0.2	0.1	0.2	0.1	0.9	0.5
P36	0.9	0.3	0.3	0.3	0.8	0.5	0.1	0.2	0.8	1	0.8	0.9	0.3	0.8	0.9
P37	0.8	0.9	0.6	0.6	0.8	0.3	1	0.1	0.6	1	0.9	0.7	0.3	0.4	0.4
P38	0.6	0.2	0.2	0.7	1	0.4	0.2	0.1	0.4	0.4	1	0.8	0.8	0.7	0.7
P39	1	0.8	0.5	1	0.4	0.1	1	0.7	0.7	0.9	0.1	0.4	1	0.2	0.6
P40	0.8	0.5	0.1	1	0.6	0.2	0.2	0.9	0.8	1	0.9	0.8	0.8	1	0.2
P41	0.4	0.5	0.9	0.4	0.8	0.1	0.8	0.5	1	0.1	0.1	0.8	0.7	1	0.5
P42	1	0.9	0.1	0.3	0.2	0.7	0.3	1	0.1	0.5	0.2	0.7	0.8	0.3	0.6
P43	0.1	0.8	0.6	0.1	0.5	0.9	0.5	0.8	0.5	0.7	1	0.4	0.7	0.6	0.9

P44	0.1	0.8	0.2	0.7	0.5	0.6	0.4	0.1	0.3	0.6	0.2	0.2	0.6	0.3	0.2
P45	1	0.8	0.9	0.7	0.3	0.3	0.5	0.6	0.5	0.8	0.1	0.7	0.8	1	0.4
P46	0.1	0.4	0.1	0.1	0.8	0.7	0.6	1	0.4	0.2	0.6	0.5	0.4	0.1	0.4
P47	0.3	0.5	0.1	0.8	0.9	0.6	0.2	0.8	0.5	0.2	0.3	0.8	0.1	0.2	0.9
P48	0.3	0.3	0.9	0.8	0.2	0.6	0.1	0.2	0.1	0.9	0.7	0.4	0.7	0.3	0.3
P49	0.7	0.1	0.1	0.8	0.9	1	1	0.8	0.4	0.9	0.7	0.6	0.1	0.6	0.7
P50	0.1	0.9	0.1	1	0.7	0.8	0.7	0.6	0.3	0.1	0.8	0.9	0.9	0.6	0.4
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
P1	0.5	1	1	0.5	0.9	0.1	0.4	0.9	0.1	0.2	0.3	0.2	0.7	0.3	0.2
P2	0.8	0.4	0.9	0.6	0.4	0.8	0.6	0.5	0.7	0.7	0.8	1	0.6	0.9	0.2
P3	1	1	0.8	0.5	0.5	0.3	0.7	0.4	1	0.8	0.5	0.8	0.3	0.5	1
P4	0.2	0.5	0.8	0.9	0.3	0.3	0.9	0.3	0.9	1	0.3	0.3	0.1	0.1	0.7
P5	0.5	0.5	0.5	1	0.1	0.3	0.1	0.7	0.7	1	0.6	0.5	0.2	0.7	0.8
P6	0.2	0.7	0.4	0.2	0.6	0.9	0.7	1	1	0.1	0.4	0.6	0.3	0.6	0.1
P7	0.1	0.1	0.3	0.5	0.5	1	0.6	0.4	0.5	0.6	0.2	0.9	1	0.6	0.1
P8	0.3	0.4	0.6	0.8	0.4	0.2	0.5	0.5	0.1	0.7	0.8	0.3	0.3	0.8	0.8
P9	1	0.2	1	0.8	0.9	0.3	0.5	0.1	0.9	0.6	0.4	0.4	0.9	0.4	0.1
P10	0.3	0.8	0.7	0.3	0.7	0.7	0.7	0.2	0.7	0.2	0.6	0.7	0.7	0.7	0.9
P11	0.3	0.1	0.1	0.9	0.4	0.5	1	0.3	0.9	0.7	0.9	0.5	1	0.4	0.6
P12	0.2	0.1	0.3	0.2	0.1	0.9	0.2	0.1	0.8	0.6	0.3	0.4	0.1	0.9	0.9
P13	0.6	0.2	0.6	0.7	0.1	0.9	0.9	0.7	0.5	0.1	0.9	0.2	0.5	0.1	0.4
P14	0.8	0.7	0.1	0.9	0.8	1	0.8	0.4	0.4	0.6	0.6	0.8	0.8	0.5	0.9
P15	0.6	0.5	1	0.3	0.5	0.6	0.8	0.2	1	1	0.1	0.8	1	0.9	1
P16	0.3	0.7	0.1	0.3	0.7	0.4	0.4	0.8	1	0.2	0.1	0.2	0.2	1	0.2
P17	0.2	0.2	0.4	0.1	0.4	0.1	0.8	1	0.2	0.7	0.2	0.4	0.2	0.3	0.4
P18	0.3	0.8	1	0.7	0.6	0.9	0.9	0.7	0.7	0.9	1	0.5	0.9	0.7	0.4
P19	0.4	0.3	0.4	0.8	0.7	1	0.9	0.5	0.7	0.1	0.6	0.5	0.9	0.9	0.5
P20	0.6	0.7	0.2	0.2	0.5	0.6	0.3	0.8	0.4	0.1	0.6	0.6	0.3	0.1	1
P21	0.4	1	0.7	1	0.2	0.8	0.4	0.5	0.1	0.6	0.4	0.8	0.8	0.4	0.3
P22	0.8	0.3	0.2	0.7	0.3	0.5	0.9	0.9	0.8	0.8	1	0.5	0.2	0.8	0.9
P23	0.7	0.2	0.6	0.3	0.6	0.1	0.6	0.5	0.2	0.1	0.1	0.4	1	1	0.6
P24	0.1	0.7	0.8	0.1	0.2	0.9	0.8	0.6	0.7	1	0.6	0.8	0.7	1	0.1
P25	0.5	0.1	0.6	0.6	0.3	0.3	0.1	0.4	0.8	0.6	0.5	0.2	0.6	0.8	0.9
P26	0.4	0.9	0.9	0.9	0.3	0.9	0.7	0.2	0.2	0.9	1	0.7	0.4	0.3	0.5
P27	0.2	0.1	0.4	0.9	0.3	0.6	1	0.4	0.3	0.4	1	0.8	0.7	0.8	0.5
P28	1	0.5	0.4	0.1	0.2	0.9	0.7	0.4	0.1	0.4	0.8	0.3	0.2	0.5	0.9
P29	0.4	0.5	0.1	1	0.7	0.7	1	0.5	0.5	0.9	0.2	0.6	0.3	0.4	0.4
P30	0.5	0.5	0.7	0.1	0.6	0.3	0.3	0.3	0.9	0.5	0.6	1	0.4	0.7	0.2
P31	0.5	0.5	0.4	1	0.2	0.4	1	0.1	0.4	0.2	0.2	0.2	0.1	0.6	0.8
P32	0.2	1	0.3	0.1	0.7	0.5	0.4	1	1	0.5	0.1	0.9	0.1	0.1	0.7
P33	1	1	0.5	1	0.5	1	1	0.8	0.7	0.9	0.1	0.4	0.1	0.4	0.1
P34	0.9	0.5	0.8	0.1	1	0.9	1	0.3	0.5	0.4	0.1	0.3	0.7	1	0.1
P35	0.8	0.9	0.4	0.9	0.4	0.2	0.6	0.3	0.6	1	0.5	0.9	0.5	0.3	0.7
P36	0.9	0.8	1	0.5	0.7	0.1	1	0.5	0.4	0.7	0.5	0.9	0.6	0.3	0.2
P37	0.4	1	0.6	0.8	0.9	1	0.9	0.5	0.7	0.4	0.9	0.6	0.9	0.4	1
P38	0.7	0.8	0.2	0.6	0.9	0.2	0.9	0.1	1	0.4	0.1	0.8	0.4	0.8	0.1
P39	0.5	0.8	0.9	0.2	0.5	1	0.1	0.8	0.8	0.4	0.3	0.5	0.7	1	0.5
P40	0.3	0.9	0.8	0.5	0.4	0.5	0.2	0.5	0.6	0.6	0.3	1	0.6	0.5	0.8
P41	0.6	0.5	0.5	0.4	0.9	0.9	1	0.9	0.9	0.8	0.5	0.5	0.3	0.1	0.4
P42	0.4	0.2	0.8	0.9	0.8	0.1	0.7	0.4	0.6	1	0.6	0.5	0.8	0.9	0.6
P43	0.5	0.8	0.8	0.9	0.1	0.1	0.6	0.3	0.1	0.8	0.1	0.4	0.4	0.9	0.1
P44	0.6	0.4	0.3	1	0.5	0.3	0.9	0.4	0.1	1	0.5	0.7	0.6	1	0.3
P45	0.1	0.4	0.7	0.8	0.1	1	0.6	1	0.4	0.8	0.4	0.9	0.1	0.3	0.7
P46	1	0.6	0.8	0.7	0.8	0.2	0.5	0.1	0.9	0.9	0.4	0.9	0.8	0.2	0.8
P47	0.5	1	0.9	0.3	0.9	0.3	0.7	1	0.5	0.2	0.4	0.7	0.5	0.4	0.6
P48	0.5	0.4	0.3	0.8	0.5	1	0.9	0.9	0.3	0.6	0.4	0.7	0.4	0.2	0.5
P49	0.5	0.2	0.7	0.6	0.9	0.8	1	0.8	0.1	0.5	1	1	0.7	0.1	0.8
P50	0.6	0.4	0.3	0.7	0.1	0.8	0.5	1	1	0.2	1	0.1	0.8	0.8	0.1
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
P1	0.1	0.8	0.5	1	0.5	0.5	0.9	0.6	0.3	0.7	0.9	0.1	0.7	0.4	0.9
P2	1	0.3	0.3	0.9	0.8	0.2	0.1	0.9	0.2	0.3	0.7	0.3	0.5	0.1	1
P3	0.7	0.3	0.9	0.7	0.2	0.3	0.7	0.7	0.4	0.6	0.5	0.1	0.1	0.4	0.1
P4	0.2	0.9	0.2	0.2	1	0.5	0.4	0.4	0.4	0.4	0.6	0.2	0.1	0.5	0.9
P5	0.4	0.1	0.5	0.8	0.8	1	0.9	0.4	0.7	0.8	0.2	1	0.6	0.7	0.3

P6	0.6	0.8	1	0.8	0.8	0.5	0.7	0.9	0.1	1	1	0.7	0.3	0.9	0.6
P7	0.9	0.7	0.8	0.1	0.5	0.4	0.2	0.9	0.9	0.5	1	0.2	0.9	0.8	0.5
P8	1	0.5	1	0.5	0.9	0.9	0.2	0.4	0.6	0.8	0.6	0.5	0.5	0.1	0.5
P9	0.2	0.1	0.6	0.2	0.6	0.2	0.8	0.4	0.9	0.1	0.6	1	0.3	0.9	0.2
P10	0.1	0.4	0.8	0.4	1	0.7	0.1	0.1	0.1	0.3	0.6	0.5	0.8	0.6	0.6
P11	0.2	0.6	0.5	0.7	0.7	1	0.2	0.2	0.6	1	0.1	0.9	0.1	0.6	0.2
P12	0.3	0.6	0.2	0.6	0.4	0.7	0.9	0.6	1	0.8	0.2	0.9	0.2	0.7	0.7
P13	0.9	0.9	0.4	0.9	0.5	0.6	0.7	0.7	0.7	0.7	0.6	0.8	0.6	0.7	1
P14	0.5	0.3	0.6	0.7	0.7	1	0.8	0.8	0.9	1	0.6	0.7	0.8	0.5	0.6
P15	0.2	0.7	1	0.4	0.6	0.7	0.1	1	0.9	0.7	0.5	0.8	0.7	0.3	0.9
P16	0.7	0.3	0.3	0.8	0.6	1	0.4	0.8	1	0.6	0.9	0.8	0.4	0.8	0.2
P17	0.2	0.3	1	0.4	0.6	0.1	0.1	0.2	0.9	0.9	0.2	1	0.8	0.9	0.5
P18	0.7	0.4	1	0.5	0.1	0.7	0.6	0.3	0.8	0.1	1	0.3	0.2	0.7	0.1
P19	0.8	0.7	0.1	0.2	0.5	0.1	0.3	0.4	0.9	0.4	0.3	0.5	0.9	0.2	0.6
P20	0.3	0.3	0.1	0.7	0.6	0.1	0.7	0.5	0.1	0.4	0.3	1	0.7	0.9	0.7
P21	0.4	1	0.5	0.4	0.8	0.4	1	0.2	0.4	1	0.6	0.1	0.8	0.1	0.7
P22	0.2	0.1	0.5	0.1	0.7	1	0.5	0.6	0.1	0.5	0.5	0.3	0.3	0.8	0.1
P23	1	0.7	0.3	0.9	1	0.2	0.7	0.3	0.7	0.5	1	0.3	0.7	0.4	0.5
P24	1	0.3	1	0.7	0.5	0.1	0.5	0.8	0.3	0.7	0.3	0.8	0.8	0.8	0.7
P25	0.8	1	0.9	0.5	1	0.1	0.7	1	0.1	0.3	0.7	0.6	0.1	0.9	0.5
P26	0.7	0.3	0.5	0.5	0.1	0.8	0.1	0.2	0.8	0.3	0.3	0.4	0.5	0.6	0.7
P27	0.3	0.4	0.8	0.7	0.7	0.6	1	0.8	0.4	0.8	0.5	0.6	0.2	0.4	0.7
P28	0.7	0.1	0.5	0.4	0.2	0.4	0.1	0.5	0.7	0.9	0.8	0.2	0.6	0.6	1
P29	0.7	1	0.1	0.9	0.7	1	0.1	0.1	0.8	0.9	0.3	0.8	0.3	0.1	0.9
P30	0.8	0.4	0.8	0.4	0.8	0.2	0.2	0.7	0.1	1	0.5	0.2	0.3	0.2	0.1
P31	0.5	0.9	0.9	0.1	0.7	0.2	0.3	1	0.2	0.3	0.9	0.8	0.1	0.9	0.4
P32	0.2	0.1	0.1	0.8	0.1	0.7	0.3	0.7	0.6	0.6	0.3	0.9	0.1	0.3	0.9
P33	0.9	0.1	0.5	0.1	0.7	0.4	0.6	0.5	0.2	0.1	0.7	0.7	0.2	0.5	0.8
P34	0.9	0.3	1	0.3	0.3	0.6	0.6	0.9	0.3	0.3	0.3	0.6	0.3	0.7	0.4
P35	0.4	0.5	0.4	0.5	0.7	0.6	1	0.2	0.6	0.5	0.4	0.5	1	0.9	0.2
P36	0.6	1	1	0.8	0.3	0.5	0.1	0.4	0.8	0.8	0.5	0.4	0.3	0.1	0.7
P37	0.1	1	0.8	0.7	0.5	0.2	0.9	1	0.2	0.3	1	0.7	0.7	0.6	0.1
P38	0.9	0.8	0.9	0.9	0.8	0.7	1	0.9	0.9	0.7	0.2	0.8	0.8	0.6	0.8
P39	0.5	0.1	0.6	0.5	0.2	0.2	1	0.2	1	0.7	1	0.8	0.1	0.4	0.6
P40	0.8	0.7	0.1	0.1	0.7	0.9	0.5	0.8	0.1	0.1	0.6	0.9	0.8	0.8	0.2
P41	0.9	0.3	0.3	0.4	0.5	0.3	1	0.4	0.8	0.9	0.3	0.2	0.4	0.8	0.1
P42	0.4	1	0.3	0.8	0.8	1	0.2	0.9	0.6	0.1	0.3	0.2	0.9	0.9	0.8
P43	0.5	0.7	0.4	0.2	0.9	1	1	0.4	0.7	0.6	0.3	0.4	0.1	0.1	1
P44	0.7	0.2	0.5	0.2	0.4	0.6	0.1	0.3	0.4	0.2	0.5	0.1	1	0.7	0.7
P45	0.7	0.8	1	0.7	0.8	0.7	0.2	0.3	0.9	0.5	0.7	0.3	1	0.6	1
P46	0.9	0.8	1	0.6	0.4	0.4	1	0.9	0.2	0.3	0.3	0.2	0.4	0.5	0.7
P47	0.5	0.2	0.7	0.8	0.1	1	0.7	0.1	0.5	0.2	0.5	0.9	0.9	1	0.9
P48	0.6	0.4	0.7	0.6	0.4	0.5	0.2	0.8	0.2	0.5	0.9	0.5	0.3	0.1	0.7
P49	0.5	0.3	0.3	0.2	0.1	0.8	0.7	0.5	0.6	0.3	0.5	0.2	0.8	0.2	0.5
P50	0.8	0.5	0.3	0.8	0.6	0.8	0.1	0.4	0.3	0.7	0.2	0.6	0.2	0.1	0.1
	U46	U47	U48	U49	U50	U51	U52	U53	U54	U55					
P1	0.6	0.8	0.5	0.4	0.2	0.2	0.7	0.4	0.6	0.2					
P2	0.6	0.5	0.6	0.4	0.5	0.3	0.6	0.8	0.6	0.7					
P3	0.4	0.7	0.1	0.1	0.7	0.7	0.1	0.1	0.2	0.6					
P4	1	0.3	0.2	0.9	0.3	0.7	1	0.7	0.9	0.1					
P5	0.6	1	0.4	0.7	0.4	0.7	0.7	0.4	0.5	0.7					
P6	0.8	0.5	1	0.1	0.6	0.3	0.9	0.4	0.7	0.1					
P7	0.7	1	0.7	0.3	0.4	0.2	0.5	0.5	0.5	0.7					
P8	0.4	0.4	0.9	0.8	1	0.6	0.1	0.6	0.9	1					
P9	0.1	0.3	1	0.1	0.3	1	0.3	0.5	0.3	0.7					
P10	0.1	0.2	0.4	0.3	0.8	0.4	0.7	1	0.6	1					
P11	0.8	0.3	1	0.2	0.6	0.9	1	0.4	0.5	0.5					
P12	0.8	0.4	0.5	0.5	0.7	0.2	0.1	0.8	0.6	0.3					
P13	0.9	0.4	0.6	0.2	0.3	0.9	0.7	0.1	0.6	0.5					
P14	0.2	0.2	0.3	0.2	0.2	0.5	0.8	0.3	0.3	1					
P15	0.2	0.3	0.1	0.4	0.4	1	0.9	0.3	0.1	0.1					
P16	1	0.2	0.8	0.7	0.2	0.1	0.3	1	1	0.3					
P17	0.5	0.3	0.1	0.3	0.2	1	0.2	0.1	0.4	0.7					
P18	0.4	0.3	0.1	0.1	0.5	0.9	0.9	0.1	0.8	0.9					

P19	0.5	0.4	0.4	0.9	0.8	0.9	0.5	0.8	0.2	0.1
P20	0.8	0.5	1	0.1	0.6	0.6	0.3	0.4	0.1	0.6
P21	1	0.4	0.8	0.7	1	0.3	0.6	1	0.3	0.6
P22	0.1	0.8	0.8	0.7	0.9	0.4	0.4	0.4	0.8	0.2
P23	0.5	0.8	0.3	0.8	0.9	0.5	0.5	1	0.9	0.7
P24	1	0.2	0.3	0.9	0.1	0.6	0.2	0.6	0.3	0.1
P25	0.9	0.8	0.7	1	0.2	0.2	0.8	0.4	0.5	0.4
P26	0.7	1	0.4	0.3	0.7	0.6	0.3	0.6	0.9	1
P27	0.9	1	0.1	0.2	0.5	0.1	0.3	0.7	1	0.4
P28	0.5	0.1	1	0.1	0.6	1	0.9	0.7	0.7	1
P29	0.1	0.8	0.2	0.5	0.6	0.7	0.6	0.3	0.7	0.3
P30	0.3	0.9	0.8	0.7	0.4	0.5	1	0.8	0.5	1
P31	0.2	0.6	1	0.5	0.6	0.7	0.1	0.9	0.9	0.1
P32	1	0.1	1	1	1	0.6	1	0.4	0.3	0.1
P33	0.5	0.3	0.6	0.9	0.9	0.9	1	0.8	0.4	0.1
P34	0.5	0.6	0.7	0.2	0.3	1	0.3	0.2	0.5	0.7
P35	0.9	0.8	0.4	0.8	0.1	0.6	0.8	0.2	0.6	0.3
P36	0.9	0.3	0.5	0.1	0.1	0.6	0.8	0.1	1	0.5
P37	0.3	1	0.9	0.1	0.5	0.5	0.8	0.5	0.6	0.8
P38	0.3	0.5	0.5	0.8	0.8	0.3	0.7	0.3	0.7	0.3
P39	0.4	0.7	0.4	1	0.4	0.1	0.9	1	0.4	0.4
P40	0.8	0.3	0.8	0.2	0.3	0.5	0.1	0.6	0.2	1
P41	0.6	0.6	0.7	0.6	0.7	0.8	0.9	1	1	0.2
P42	0.9	0.7	0.7	0.4	0.3	0.3	0.5	0.5	0.8	0.4
P43	0.9	0.8	1	0.2	0.1	0.6	0.1	0.2	0.1	0.8
P44	0.7	0.9	1	0.5	0.9	0.4	0.5	0.9	0.4	0.9
P45	0.6	0.9	0.3	0.3	0.9	0.8	1	0.5	0.8	0.7
P46	0.9	0.7	0.2	0.5	1	0.5	0.4	1	0.8	1
P47	0.3	0.3	1	0.4	0.1	0.1	0.1	0.9	1	0.8
P48	0.7	0.8	1	0.4	0.1	0.9	0.3	0.6	0.8	0.1
P49	0.4	0.2	1	0.8	0.6	0.9	0.8	0.4	0.7	0.4
P50	0.2	0.6	0.7	0.6	0.6	0.4	0.8	0.2	0.6	1

Demand

D	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
	150	150	30	20	60	60	100	100	80	50	20	90	50	90	90
	U16	U17	U18	U19	U20	U21	U22	U23	U24	U25	U26	U27	U28	U29	U30
	140	20	30	150	60	150	90	140	130	70	130	60	120	50	20
	U31	U32	U33	U34	U35	U36	U37	U38	U39	U40	U41	U42	U43	U44	U45
	90	140	120	70	130	30	100	120	70	110	130	70	40	100	110
	U46	U47	U48	U49	U50	U51	U52	U53	U54	U55					
	100	50	50	40	50	130	30	130	20	70					

Minimum Quantity and Capacity

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
m	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
M	50	100	100	140	130	90	120	110	80	90	180	70	200	200	90
	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	70	180	180	120	130	160	120	100	160	90	60	200	80	80	70
	P31	P32	P33	P34	P35	P36	P37	P38	P39	P40	P41	P42	P43	P44	P45
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
	200	50	120	200	60	130	70	200	180	100	70	200	130	120	130
	P46	P47	P48	P49	P50										
	10	10	10	10	10										
	190	100	140	140	60										

Appendix C

Data for Controlled Tabular Adjustment

Sensitive cells are bold and marked with *. Sensitive entries must be assigned a value 20% greater than the original value or 20% smaller than the original value (flooring to the nearest integer). For example, if the value of a sensitive cell is 943, then it will be perturbed to a value that is either 188 (flooring the value to the nearest integer) greater or 188 less than 943, which is either 1131 or 755.

All nonsensitive cells can be modified to values within 20% of their original values. The last column and the last row are the column total and row total respectively.

Table C.1: 10by10 table with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	142	322	818	293	80	928*	167	387	486*	3623
R2	811	406	795	19	957	855	986	251	483*	5563
R3	325	57	291	676	952	40	822	224	1000	4387
R4	261	230	525*	561	826	288	253	240*	551	3735
R5	215	665	309	470	506	688*	193	274	926	4246
R6	235	128	413	635	182	403	689	326	923	3934
R7	757	401	196	400	154	65	120	29	865*	2987
R8	414	792	816	395	826*	338	197	510	210	4498
R9	580	979	467	802*	433	241*	635	367	431	4935
CT	3740	3980	4630	4251	4916	3846	4062	2608	5875	37908

Table C.2: 15by15 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	278	288	735	497	55	191	725	974	704	802
R2	757	905	421	658	1	982	412	348	668	444
R3	558	961	951	735	223	497	7	316	738*	790
R4	602	832	293	466	46	86	513	715	35*	980
R5	503	755	238	943*	89	543	903	861	631	262
R6	535	702	875	986	885	405	627	385*	848	527
R7	954*	714	647	447	175	835	970	135	251	910
R8	227	978	861*	14	486	416	773	488	523	782
R9	997	346	176	68	309	335	376	952	719	779
R10	599	899	172	819	69	956	317	5	760	309*
R11	870	872	762	669	902*	821	833	640	156	783
R12	490*	424	796	763	239	635	232	616	269*	991
R13	84*	949	691	616	895	13	301	965	102	596
R14	462	903	22	760	796	314	236	494*	685	945
CT	7916	10528	7640	8441	5170	7029	7225	7894	7089	9900
	C11	C12	C13	C14	RT					
R1	206	773*	873	807	7908					
R2	840	530*	257	156*	7379					
R3	400	400	252*	92	6920					
R4	497	701	803	517	7086					
R5	379	337*	315	947	7706					

R6	807	393	962	30	8967					
R7	676	623*	512*	3	7852					
R8	591	126*	110	663	7038					
R9	618	649	756	148	7228					
R10	715	81	846	718	7265					
R11	731	565	723	240*	9567					
R12	760	482*	945	361	8003					
R13	469	717	859	186	7443					
R14	439	497	99*	554	7206					
CT	8128	6874	8312	5422	107568					

Table C.3: 20by20 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	897	810	389*	479*	607	652	763	448	643	281
R2	343*	419*	119	519	972	329	339	802	160	159
R3	784	3	797	208	200	978	623	55	142	408
R4	139	577	684	316	512	446*	520	366*	652	872
R5	212	990*	330*	426*	146	546	92	435	84	555*
R6	519*	203	581*	598	657	581	764*	164	840	996
R7	347	298	40	382	132	143	382	637	780	428
R8	464	344	209	316	898	71	248*	20*	997	100
R9	939	286*	765	149*	619	623*	808	318	349*	254
R10	233	140	238	874	861	652	874	964*	947	886
R11	161	535*	428	407	832	193	680	918	46	881
R12	28	612	524	623	139	451	409*	909	401	191
R13	740	982	486	0*	782	347	296	832	98	374
R14	776	708*	603	39	846	230	495*	30	648	400
R15	561	493	994	26	260	327	123	746*	629	197
R16	530	805	700	0	285	171	203	95	39	393
R17	819	43	292	722	104	935	459	987	414	810
R18	950	68	444	209	808	504	768	222	800	38
R19	486	432	140	575	203	418	204	686	831*	284
CT	9928	8748	8763	6868	9863	8597	9050	9634	9500	8507
	C11	C12	C13	C14	15	16	17	18	19	RT
R1	443	248	152	207	914	583	693	856	486*	10551
R2	164	914	775*	430	259	622*	985	637	233	9180
R3	778*	163*	26	851	253	395	687	871	681	8903
R4	868	530*	959	591	23	182	467	613	140	9457
R5	18	644	392	327	265	510	456	810	467	7705
R6	660*	123	587	146	950	742	873*	165	474	10623
R7	171	367	978	459	874	152	442	921	980	8913
R8	919	134*	205	164	777	560	665	874	610	8575
R9	921	703	706	151	704	74	53	524	650	9596
R10	538	813*	243	364	678	197	195	550	18	10265
R11	422	456	617	743	996*	928	19	381	657	10300
R12	817	423	386	716	605	82	130	368	560	8374
R13	640	34	319	962	666	86*	256	143	277	8320
R14	768	384	891	408	377	88	983	922	462	10058
R15	464	266	4	116	559	892	901	700	234	8492
R16	233	113	433*	282*	855	35	982	955	984	8093
R17	50	616	403	602	231*	830	69	233	487	9106
R18	732	955	333	327	559*	729	305	818	811	10380

R19	372	557*	338	903	234	404	326	395	280	8068
CT	9978	8443	8747	8749	10779	8091	9487	11736	9491	174959

Table C.4: 25by25 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	22	870	614*	339*	686	217	231	442	334	267
R2	631	524*	309	463	133*	761	135	431	859	467
R3	41	24	722	886	937	925	756	377	834	687
R4	977	969	645	93	772	404*	280	226	145	93
R5	519	898	707	395	642*	304	85	885	94	951*
R6	605	47	737	488	986	978	142	929	16	728
R7	106	29	51	183*	510	648	510	599	336*	723
R8	598	405	134	302	460	503	763	944	650	946
R9	701	285	987	224	501	430	516	47	965	66
R10	631	591	54	869	728*	17	744	292	756	104
R11	536	840	240	22	148	476*	878	863	996	757
R12	587	268*	784	149	773*	520	587	519*	141	316
R13	524	669	677	953*	624	458	648	979	421	976*
R14	58	797	560	771	995	90	721	997	108	457
R15	516	894	642	203	69	800	423	892	992	401
R16	192	233	220	79	124	538	689	720	939	8
R17	484	845	466*	599	208	517	158	744	489	650
R18	301	180	716	790	331	782	777	105	329	261
R19	19	768	458	44	100	957	609	247	930	430
R20	610	447	418	412	714	103	380	752	149	661
R21	891	714	202	108	738	762	279	845	624	622*
R22	811	462	94	103	875	61	847	454	248	989
R23	468	245	566	761	345	448	516	157	981*	299
R24	517	520*	626	949*	223	995	877	626	212	916
CT	11345	12524	11629	10185	12622	12694	12551	14072	12548	12775
	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
R1	755	416	950	837	896	255	521	337	656	395
R2	286	65*	657	630	732	955	369	69	868	295
R3	555	86	555	679	428	163	693	519*	801	778
R4	703*	681	458	532	298	403	152	790*	199	494
R5	618	761	621	579	281	100	722	841	821	544
R6	734	68	797	453	211*	735	577*	789*	83	376
R7	118	520	995	313	506	820	299	709	869	656
R8	464	271	69	187	349	914	14	988	174	403
R9	510	705	625	227	631	153	311	649	836	18
R10	227*	488	873*	579	421	813	695	441	471	881
R11	886	369	716	286	789	205	430	629*	559	246
R12	861	433	59	886	256	288*	224	591	564*	866
R13	519	543	379	849	546	815	486	116	143	785
R14	133	704	306*	64	174	193	670	387	70	681
R15	341	317	364	616	597	685	974	280	174	922
R16	582	556	689	761	824	528*	342	68	264	994
R17	42*	335	433	99	681	27	321	52	370	384
R18	388*	897	767*	244	455	272	423*	699	817	825*
R19	49*	905	978	658	727	489*	686	363	401	514
R20	237	728	259*	470*	858*	833	476	340	542*	562

R21	955	295	632	531	871	584	603	488	943	347
R22	564	221	878*	728*	853	178	540	472	732	981*
R23	155	936	674	428	833	171	441	752	615	254
R24	134	968	620	641	14*	790	900	2*	516	624
CT	10816	12268	14354	12277	13231	11369	11869	11371	12488	13825
	C21	C22	C23	C24	RT					
R1	101*	462	556	302	11461					
R2	553	782	70	633	11677					
R3	928	726	95	449	13644					
R4	303	148	964*	737*	11466					
R5	364	501*	330*	231*	12794					
R6	201	646	124	508	11958					
R7	862	927	939	666	12894					
R8	399	786	506	562	11791					
R9	404	800	160	312	11063					
R10	645	146	212	126	11804					
R11	657*	132	513	274	12447					
R12	220	301	587	902*	11682					
R13	269	557	136	425	13497					
R14	856	97	392	376	10657					
R15	260	375	645	316	12698					
R16	385	786*	870	85	11476					
R17	336	891	238	81	9450					
R18	403	652	152	340	11906					
R19	46	991*	310	256	11935					
R20	925	274	296	613	12059					
R21	200	98	652	606*	13590					
R22	667	683	875	508*	13824					
R23	350	613	338	636	11982					
R24	265	399	978	304	13616					
CT	10599	12773	10938	10248	291371					

Table C.5: 30by30 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	980	423	508	214	332	473	156	849	411	485
R2	813	696	899	128	360*	547	62	277	560	153
R3	960	388	715	715	795	162	287	576	465*	834
R4	881	362*	741	612	102	538	508	646	900	754
R5	1108	643	432	555	363	493	186	274	89	553
R6	539	539	953	281	739*	859*	125	693*	834	176
R7	140*	675*	278	688	521	443	600	531*	924	799
R8	485*	830	536	445	770	80	199	390	886	365
R9	531	557	72	793	594	91	475	500	601	61
R10	63	915	956	93	470	362	185	886*	551	520
R11	823	591	747	546	567	814*	909	416	984	832
R12	18*	425	5	582	191	920	543*	469	957*	283
R13	161	996*	94	747	749	614	355*	473	399	436
R14	455	786	303	567	765	483	917*	518	861	115*
R15	564	800*	928	250	199	486*	362	828	522	305
R16	679	331	687	556*	999	688	618	156	115	934
R17	475	761	83	286	105	532	993	661	275	873
R18	529	52	470	117	700	754	390	73	350	686
R19	703	430	341*	742	611	130	618	93	778	281
R20	879	403	990	206	501	301	6	773	565	957
R21	987	514	442	592	293	709	761	508	732	244
R22	84	623	235	566	738	895	302	94	781	29

R23	249*	622	655	900	385	11	78	680*	151	331
R24	18	334	707	639	614*	620*	444	853	907	363
R25	374	254	38	367	303	119	323	549*	268	567
R26	264	766*	752	524	782	288	950	930	217*	833
R27	649*	161	700	412	522	276	651	251	141	915
R28	456	323*	876	981	894	769	172	2	592	863
R29	724	106	159	448	458	634	409	157	608	222
CT	14591	15306	15302	14552	15422	14091	12584	14106	16424	14769
	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
R1	201	755	431	152	35	409	806	117*	17	455
R2	444	198	370	368	232	34	487	747	444	858*
R3	25	79	334	350	345*	28	263	213	549	709*
R4	961	931	242*	431	318	457	66	811*	227	296
R5	697	335	697	112	176	788*	316*	621	573	115
R6	236	795	397	546	469	892*	536	538	408	999
R7	682	198*	737	277	951	698	637	881	267	995
R8	832	283	777	85	240	746	343	495	444*	572
R9	821	484	270	177	789	366	990	158*	452	509
R10	388	489	440	164	187	574	798	614*	49	715
R11	262	201	560	120	786*	319	662	729	416	580
R12	388	568	285	248	385*	122	883	148	421	366
R13	541*	46	730	822	357	998	87	391	597	37
R14	819	857	59	93	346	185	550	417	316	845
R15	659*	185	579	664	640	576*	947	774	594	703*
R16	140	925	582	683*	944*	845	162	51	124	836
R17	181	361	751	562	659	394	604	547*	400	538
R18	365	889	535	894	251	374	71	885	825	635
R19	498	454	29	890	167	809*	104	324	139	178
R20	301	934	999	811	308	555	813	382	862	153
R21	224	596	182	545	508	349	655	608	605	396
R22	323*	286	145	397	812	326	355*	351	532	405*
R23	83	194	871	471	193	285	623	933	276	511
R24	673	385	552	35	863	466	419	111	349*	950
R25	505	973	630	51	923*	483	760	263	34	543
R26	410	324	921	426*	523	327	588	750	630	291*
R27	338	935	240	538	887	818	79	643	694	774
R28	633	553	541*	968	873	747	897	848	536	776
R29	918	657	118	216	485	349	302	21	231	540
CT	13548	14870	14004	12096	14652	14319	14803	14371	12011	16280
	C21	C22	C23	C24	C25	C26	C27	C28	C29	RT
R1	590	792	460	457	514	560	953	902	463	13900
R2	348	937	992	215	783	652	547	588	380	14119
R3	946	875	158	466	697	647	856	310	55	13802
R4	811	299	323	938*	316	27*	492	652	575	15217
R5	805	955	475	327	217	279	941	795	904	13824
R6	292	638	769	423	531*	55	836	933	18	16049
R7	920	990	926*	65	335	342*	455	735	599	17289
R8	797	242	918	154	460	312	706	752	911	15055
R9	129*	623*	430	793	925	944	216	585*	690	14626
R10	480	596	888	642	128	407	487	880	448	14375
R11	369	180	247	840	298	292	254	456	870	15670
R12	927	431	96	294	916	586	821	86	178*	12542

R13	388	61	823	198	786	595	119	827	386	13813
R14	961	894	993	215	877*	837	100	716	249	16099
R15	945	248	604	648	873*	530	546	560	240	16759
R16	270	518	754*	547	114	782	694	33	828	15595
R17	958*	226	103	909	33	888	923	612	329	15022
R18	316*	947	178	20	799	932	85	966	98	14186
R19	496	404	967	605	695	118	186	392	508	12690
R20	240*	294	482	683	904	367*	516	656*	136	15977
R21	715	712	605	497*	267	720	981	532	4	15483
R22	561	946	140*	813	752	602*	545	293	656	13587
R23	568	529	782	650	626	56*	649	70	681*	13113
R24	310	512	941	259	162	672	220	303	29	13710
R25	868	454	453	20	485	893	651	898	401	13450
R26	654	767	920	572	97	390	357	60	202*	15515
R27	220	182	309	312	512*	541	698	968	530*	14896
R28	248	597	664	518	618	685	49	901	381	17961
R29	170*	152	701	29	105	717	842	422	375	11275
CT	16302	16001	17101	13109	14825	15428	15725	16883	12124	425599

Table C.6: 40by40 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	113	781	61	38	82	701	342	711	959	452
R2	724*	437	133*	867*	566	404*	827	2	916*	689
R3	896	900	855	220	556	531	259	979	873	736
R4	826	180	311	512	907	869	942	451*	670	571*
R5	815	194	340	347	903	723	725	682	544	48
R6	210	398	162	208	748	426	279	371	715	945
R7	644	741	257	587*	148	584	878	948	172	192
R8	34	965*	882*	631	815	308	948	611	98	448
R9	333	367	281	84	737	782	332*	307	112*	534
R10	612	175	342*	30	747	239	216	935	508	533
R11	489	998	812	561	896	579*	76	548	24	450
R12	405	387	415	284	822	998	763	594*	192	245
R13	667	478	433	166	829	921	869	332	228*	207
R14	252	953*	383	863	434	507	395	8	314	646
R15	121*	627	755	829	431	966	408*	776	710*	22
R16	574	653	109	320	391	976	966	755	319	368
R17	976	550	613	208	491	637	830	771	267	858
R18	218	354	98	266*	728	185	857	389	916	71
R19	840*	846	86	353	66	254	143	949	532	677
R20	213	645*	441	735	563	556	955	814	690	466
R21	521	284	473	895	111	901	301	830	4	648
R22	926	194	822	782	113	5	503*	131	525*	706
R23	277	15	288	103	396	342*	794	840	238	661
R24	558	935	478	613	5	492	148	98	831	466
R25	233	150	634	184	60	442*	888	978	778	492
R26	382	249	216	389	922	796	518	310	369	288
R27	36	418	820	295*	487*	571*	278	432	495	849
R28	898	392	270	682	926	821	93*	981	265	478
R29	864	47	439	607	83	686*	529	207*	509	527
R30	880	256	273	874	289	380	861	340	681	988
R31	87	162	926	222	865	238	516	988	237	508
R32	246	286	847	30	756	435	324	635	266	842
R33	563	665	301	142	522	484	559	434	775	532*

R34	324	553	313	980	882	291	359	873	690	42
R35	587	925	654	90	173	375	27	634	501	752
R36	242	581	717	395	619	344	986	835	434	74
R37	450	548*	119	411	706	187	888	76	560*	627
R38	951	700	735	165	5	931	938	886	490	170*
R39	428	75	712	517	592*	555	603	807	803*	706
CT	19415	19064	17806	16485	20372	21422	22123	23248	19210	19514
	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
R1	345*	628	167	94	441	504	62	890	618	740
R2	605	591	627	293	963	950*	226	590	26	360
R3	274	924*	272	726*	130	532	863	464*	32	70
R4	333	673	40	76	343	772	798	211	104	695
R5	3	248	79	780	211*	297	348	663	544	821
R6	435*	31	140	843	331	742*	818	887	586	118
R7	311	348	471	982	513	791	535	116	356	950
R8	326	266	509	620	446	172	832	809	622	557
R9	347	186	676	321	526	572	821*	760	15	389*
R10	66	286	386	92	992	651	916	755	249	390
R11	351	280	155	138	59	620	210	192	925	508
R12	37	498*	585	399*	592	703	263	415	618	969
R13	712	741	139	719	618	305	508	708	882	209
R14	620	347	911	289	941	329	931	543	983	813
R15	315	952	842	337	985	860	774	695*	819*	487
R16	224	721	784*	66	817	872	482	573	851	71
R17	798	397	223	994	30	59	502	10	607	721
R18	605	712	765	741	517*	81	840	191	27	629
R19	55*	949	306	312	788	497	688*	249	504	990
R20	920	481	264*	785	760	162	803	719	901	640*
R21	41	357	564	115	113	948	861	412	403	944*
R22	333	314	669	561	634*	740	691	889	156	433
R23	514	187	29	957	744	91	809	428	92	216
R24	291	467	260	106	216*	234	394	383	171	252*
R25	373	112	433	304	588	968	828*	668	295	280
R26	820	915	367	779	464	818	101	17	327	87
R27	791	387	492	811*	14*	81	551*	7	490	528
R28	405	512	299	111	333	790	859	215	364	360
R29	580	432*	402*	946	980	590	981	605	20	711*
R30	721	805	897	588	828	506	845	626	259	550
R31	440	371	569	98	861	85	971	809	158	378
R32	727	764	35	793	324	41	8	483	429*	140*
R33	209	368	123	678	183	381	92	813	407	922
R34	918	168	310	200*	766	462*	577	608	245	677
R35	119	433	298*	863*	48	170	200	806	959*	73
R36	579	240	8	86	556	731	297	570*	306	993
R37	759	192*	949	977	858	834	63	489	702	244
R38	291*	849	897	550	609	647	10	215	615	606
R39	698	301	281	648	310	944	889	66	131*	926
CT	17291	18433	16223	19778	20432	20532	22847	19549	16798	20447
	C21	C22	C23	C24	C25	C26	C27	C28	C29	C30
R1	652	630	532	176	705	4	614	575	390	725
R2	855	68*	400	347	418	698	654	399	970	989
R3	22	337	182	281	512	78	15	216	69	897

R4	777*	840	94	247	876	435	148	17	417*	740
R5	58	38	238	571	97	866	899	364	119	455
R6	403	712	347	492	63	42	884	642	533	607
R7	323	513	369	951	257	171	497*	312	964	779
R8	780	506	848	500	567	867	296	188*	889	792
R9	998	458*	808*	290	881	153	676*	644	955	402
R10	903	430	859	42	301	474	899	774	946	392
R11	896	951	417	778*	331	510	596	438	855	482
R12	66	403	223	286	670	578	876	910	844	227
R13	985	971	341	842	646*	605*	732	895	738	167
R14	115*	799	287	702	456	476	592	511	539	346
R15	95	567	10	953	241	607	452	993	837	629
R16	44	202	966	514	315	733	639	208	926	644
R17	802*	395	114	31	155	934	89	420	75*	81
R18	998	560	764	988	765*	605	141	929*	283	57
R19	227	839	745	456	903	637	348	561*	961	199*
R20	985	6	609	868	159	419*	248	7	359	365
R21	664	806	67	261	761	308	326	177	338	39
R22	423	768	147	28	295	199	835	740	845	15
R23	902	48	710	603	329	659	472	501	540	521
R24	113	68	69	618	804	653	66	476	381	889
R25	30	356	267	208	944	581	747	510	709	714
R26	775	373	992	487	529	276	413	441	812	596
R27	863	757*	138	499*	184	322	205*	43	488	895
R28	720	777*	735*	702	233	649	370	90	234	708
R29	603	196	936	116	890	300	554*	15	757	264
R30	339*	788	11	216	317	143	555	54	183	335
R31	187	676	865	944	120	522	793	752	882	743
R32	207	194	84	184*	909	160	14	909	571	545
R33	549	121	572	434*	59	461	987	381	363	537
R34	583	482	431	314	181	571	569	907	2	422
R35	699	364	676	940	246	286	323*	952	451	954
R36	97	700	164	229	387*	944	573	254	974	516
R37	150	573	19	318	510	96	176	926*	585	306
R38	314	504	386	403	576	633*	778	134	26*	338
R39	59	436	17	325	955	498	842	283	521	352
CT	19261	19212	16439	18144	18547	18153	19893	18548	22331	19664
	C31	C32	C33	C34	C35	C36	C37	C38	C39	RT
R1	530	915*	858	692	767	383	774	911	628	20190
R2	162	992	773	286	269	846	675	932	102	21631
R3	791	380	104	731	309	875	246	523	216	17876
R4	103	239	941	17	685*	512	765	97	447	18681
R5	630	226	270	93	248	876	17	930	601	16911
R6	783*	196	817	972	788*	905	99	658	542	19878
R7	597	449*	426	602	959	392	110	515*	31	19741
R8	162	502	427	168	671	959	200	378	114	20713
R9	637	741	14	190	498	446*	683	704*	767	19427
R10	211	853	331	444*	922	749	727	382	252	20011
R11	463	22	88	806	672	694	205	950	240	19265
R12	184	119	204	490	131	415	852	836	150	18648
R13	345	356	140	597*	233*	397*	479	902	819	21861
R14	838	287	281	310	271	763	592	767	252	20646

R15	115	912	144	178	477	348	211	194	982*	21656
R16	174	680	331	643	841*	348	12	840*	868	20820
R17	924*	643	246	401	403	551*	290	577*	626*	18299
R18	927	784	412	166	324	241	644	510	978	20266
R19	214	967	893	295	399*	762	68	573	527	20658
R20	24	101	134	187	835	710	510	881	592	20512
R21	797	518	404	122	230	289	469	493	537	17332
R22	791	268	318	839	992	491	586	670	333	19710
R23	17	780	723	354*	497	549	397	163	64	16850
R24	395	436	530	803	10	596	261	734	668	15968
R25	80	326	632*	886	546	348*	913	223	725	19433
R26	247	951	736	904	252	135	287	581	916	19837
R27	526	311*	773	334	718	556	429	617	651	18142
R28	273	865	652*	800	960	438	540	232	510*	20542
R29	365	336	236	766	527*	378*	883	774	987	20628
R30	904	321	24*	320	938	791	598	388	437*	20109
R31	114	348	46*	639	657	419	462*	996	371*	20025
R32	157	541	661	673	793*	818	45	266	499*	16641
R33	74*	986	387*	525	323	536	162	243	637*	17490
R34	493	735	243	155	693*	398	158	318	859	18752
R35	291	651*	367	363	915	713	821	313	7	19019
R36	31	583	949	524	803	783*	287	717*	492	19600
R37	161	890	28	269	868	341	5	318*	117	17895
R38	261	986	878*	160	728	40	877*	224	863*	20369
R39	910	834	772	963	264	857	11	521	342	20754
CT	15701	22030	17193	18667	22416	21648	16350	21851	19749	756786

Table C.7: 50by50 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	384	769	246	33*	749	845	121	79	984	74
R2	961	569	402	214	661	309	713	87	480	478
R3	892	576	205	329	594	252	840	663	648	102
R4	140	80	944*	541	483	149	95	0	274	121
R5	633	249*	281	112	87	542	788	299	508	364
R6	182	830	413	241	458	927	183	129	38	893
R7	172	374	926	621	943*	252	489	673	819	593
R8	572	517	949*	422	932	864	431	503	866*	479*
R9	137	734	839	634	314	814	553	633	569	196
R10	187	871	517	786	574	661	893	617	391	360
R11	216	227	325	559*	246	532	651	145	112	411*
R12	87	809	94	882	967	478	51	419	332*	568*
R13	728	903	821	646	169	482	678*	887	188	134
R14	603	697	849	955	337	461	682	825	195	75*
R15	141	839	365	357	861	361	388	905	28	771
R16	29	171	245	969	442	141	268	3*	398	147
R17	142	921	357	829	525	636	3*	502	399	247
R18	351	774	813	69	531	63	885	124	853	569
R19	1*	481	605	163	558	715	906	949	657	689
R20	475*	595	8	129	409	657	191	570	633	327
R21	843	607	859	806	184	619	477*	875	541	43
R22	155	528	179	427	769	119	221	104	687	356
R23	513	959	726	887	762	34	947	947*	61	190
R24	944*	450	581	982*	102	174	851	512	257	69
R25	834	174	650	137	171	777	130	293	386	277
R26	70	609	254*	467	847	605	255	961	868	381*

R27	836	785	487	604	906	517	259	517	418	123
R28	999	574	384	785	911	322	519	406	183	654
R29	589	330	116	596*	477	963	312	625	500*	40
R30	339	734	6	475*	14	936*	161	48	726	37
R31	674	697	843*	188	799	438*	378*	212	507	199
R32	271	360	295	707	399	380	1	67	144	936
R33	933	532	305	162	29	551	546	183	387	309
R34	956	283	441*	397*	889	950	419	493	851	447
R35	904	868	190	732	344	663	38*	721	387	21
R36	55	395	376	679	610	851*	841	667	750	547
R37	560	871	900	102	410*	769	872	487	383	78
R38	744	450	260	775	271	650*	283	134	385	592
R39	990	177	484	34*	87	129	975	540	358	370
R40	186	651	418	287	188	858*	563	258	2	891
R41	729	930	273	361	230	657	766*	552	4	399*
R42	88	871	514	366	736	844	563	274	130	708
R43	650	200	149	367	764	263	448	831	485	629
R44	89	894	86	111	953	34	87	655	255	907
R45	586	660	905	878	147	345	751	389	384	349
R46	236	226	539	216	894	304	438	224	670	59
R47	874	781	50	257	312	3*	40	309	215	935
R48	915	361	705	387*	407	52	558	30	358*	57
R49	802	131*	170	608	751	625	374	974	580	805
CT	24397	28074	22349	23271	25203	24573	22882	22300	21234	19006
	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
R1	986	593	661	275	56	910	418	986*	262	415
R2	218	479	499	158	187	120	995	858	681	536
R3	826	556	547	874	1*	999	811*	285	434	329
R4	943	317	4	779	74	968	606	757	999	750
R5	464	152	187*	264	154	522*	905	45	661	415
R6	757	498	963	939	171	514	473*	18	189	887
R7	771*	400	260	798	275*	617	317	147	293	498
R8	618	176	99	703	875	473	347	796	85	619
R9	734	704	902	400	89	438	647	242	105	468
R10	443	672*	584	255	75	443	39	153*	103*	377
R11	59	712	248	333	329	906	969	412	511	693*
R12	375	88	95	738*	859	472	969	874	678	668
R13	680	34*	620	447	589	634	480	68	290	673
R14	630	347	681	365	374	503	932	932	698	657
R15	444	166*	177	465	714*	765	209	461	319	289
R16	252	769	952*	666*	84	528*	927*	79	480	923*
R17	573	871	850	783	849	611	217	701	225	953
R18	861	657	637	969	710	422	589	247	665	143
R19	811*	400	753	726	534	769	786	223	957	341
R20	69*	873	48	284	719	76	672	952	36	243
R21	719	486	914	145	60	419	643	269	742	940
R22	369	254	37	271	345	261	801	181	32	529
R23	295	272*	131	637*	989	95	640	627	147	169*
R24	102	937*	210	866*	4*	399	83	144	955	149
R25	631	303*	695*	559*	600	431	710	271	212	229
R26	888*	499	487	379	339	85*	799	25	285	538
R27	63*	250*	49	908	617	172	545	325	81	181

R28	947	750	645	690	391*	46	210	528	278	705*
R29	37	974	398	85	557	795	607	556	750	237
R30	581*	528	364*	317	41	162	318	844	891	394
R31	58	353	558	356	123	533	230	534*	772	355
R32	457	210	311	118	86	446	20*	257	584	928*
R33	156	332	726	959	49	73	179	87	211	219
R34	770	445	933*	545	223	624	950	603	205	764
R35	965	550*	19	586	324	279	43	135	264	768
R36	462	944*	377	660	381	465	379	251*	251*	924
R37	696	21	868	790	672	276	3	776	570	730
R38	869*	548*	996	847	240	486	45	523	189	568
R39	451	169	386	449	840	228	546	195	743	948
R40	873	27	74	527	830	152	992	878	483	354
R41	437	181	999	830	781	241	621	745	592	108*
R42	483	744	141*	292	933	569	622	284	898	995
R43	236	465	446	228	538	363	917	746	933	987
R44	312	563	226	277	888	243	760	570	610	99
R45	980	704	622	887	331	884	321	807	571	366
R46	839	147	578*	268	345*	187	116*	242	414	554
R47	442	312	113	504	948	796	760	703	324	123
R48	763	694	654	805	367	5	61	2	999	145
R49	497	202	619	373	344	827*	830	452	493	77
CT	26892	22328	23343	26379	20904	22232	26059	21796	23150	24960
	C21	C22	C23	C24	C25	C26	C27	C28	C29	C30
R1	177	884	289	168	381	655	694	458	280	878
R2	530*	218	888	458	577	579	254	273	142	358
R3	10	464	831	236*	774	797	856	876	812	150
R4	246	41	865*	203	491	206	599	871	997	0
R5	998	648	86	98*	410	612*	122	857	401	886
R6	370	63*	534	776	340	723	544	636	877*	827
R7	617	737	466	550	136	7	719	950	693	372
R8	227	100	152	12	507	594	991	901	914	747*
R9	332	592	450	177	963	466*	721	365	511	725
R10	780*	915	644	167	157	418	481*	70	421	608
R11	182*	580	203	134	367	275	595	842	941	708
R12	231	262	475	853	598	879	560	453*	239	881*
R13	856	90	661	993	786	64	492	744	440	570
R14	155	530	569*	998	826	307	941	611*	161	982
R15	972	480	775	263	879	227	968	271	477	796
R16	639	996	643	448	679	228	131	565	540	540
R17	591*	771	429	536	58	364*	781	754*	305	970
R18	78	775*	137	630	676	65	685	956	696	321
R19	684	883	654	935	43	515	168	643	814	392
R20	182	541*	570	380	58	927	963	633	602*	701
R21	817	516	551	766	914	694*	120	306	132	509*
R22	410	913	632	712	262	59	657	232	177	74
R23	737	7	366	564	87*	410	299	166	592	164
R24	27*	383	449	19	675	704*	112	929	748	238
R25	582	369	89	76	523	1000	278	672	594	906
R26	137	114	144	526	244*	381	993	40	652*	526
R27	432	873	742	618	210*	656	837	866	121	958
R28	177*	775	779	376	884	701	370*	350	542	126

R29	64	993	938*	163	21	162	760	416	892	683*
R30	937	185	584	753*	835	162	61	533	642	541
R31	796	171	124	115	346	930	553	560	962*	140
R32	89	28	196	203*	55	305	3	807	917	89
R33	905	772	838	53*	570	932	582	167	851	578
R34	883	986	448	940*	480	509	631	390	331	58
R35	334	897	161	788	13	677	132	308	322	56
R36	334	359	737	953	286	960	83	290	918*	743
R37	178	300	623	504	268*	356*	467	504	175	437
R38	94	54	376	348	455*	559	182	532	821	389
R39	695	33	27	509	804*	604	573	845	270	818*
R40	416	520	322	470	767	657	401	62	542	265*
R41	383	760	341	232*	717*	960	39	397	4*	359
R42	925	584	608	69	332	432	118*	807	140*	655
R43	348	371	772	848	403	80*	231	770*	299	393
R44	751	331	979	403	187*	123	835	79	58	689
R45	756	535	754	460*	629	989	318*	523	820	550
R46	950	210	724	572*	275	142	752	84	578*	864
R47	88	690	317	701	340	910	920	207	749	935
R48	184	703	633*	177*	899	129	506	95	985	166
R49	814*	728*	103	972	44	578	65	158	727	90
CT	23100	24730	24678	22905	22231	24669	24143	24824	26824	25411
	C31	C32	C33	C34	C35	C36	C37	C38	C39	C40
R1	353	291*	284	389	262	960	173	71	796	809
R2	570	594	922	424	356	927	197	445	111	813
R3	20*	761	708	887	168	401	759	931	398	410
R4	200	877	105	686	874	990	188*	427	332	893*
R5	371	317	659	773	651	174	209	294	952	980
R6	581	349	947	737	583	877	285*	133	324	133
R7	380	750*	984	288	829*	795	478	61	811	153*
R8	662	817	136	649*	240	193	551	400	834	736
R9	251	28	821	780	763	421	306	690	774	314
R10	143	467	109	397	0	790	894*	0	426	905
R11	543	416	457	92	13	217	182	478	666	775
R12	152	816	543	458	25	460	845	187	982	359
R13	897	896	127	133	907	254	746	449	379	849
R14	620	462	963	659	53	329	510	767	337	161
R15	63*	655	248	589*	287*	432*	110	776	889	154
R16	964	454*	924	456	749	82	572	989	404	35
R17	954	219	785	659	504	249	765	988	468	978*
R18	706	332	730	320	79	646	783	242	651	403
R19	883	742	266	856	305	319	490	141	912	33
R20	953	771	701	635	140	766	513	794	195	751
R21	738	316*	778	317	138	289*	610	649	895	589
R22	856	14	177	947	965	832	71	568	305	179
R23	432	430	535	35	522	545	641	240	27	790
R24	662	45	864	603	937	706	971	891	212	300
R25	40	93	22	207	845*	981	88	210	62	531
R26	18	612	379	747*	341	953	972	746	119	739
R27	171	285	866	440	141	893	37	685	490	507
R28	931	327	956	469	170	157	97	19	983	752
R29	713	889*	781	495	562	200	607	135	577	17

R30	420	291	661	32	822	341	736	634	716	300
R31	590	596	201	995*	639	187	341	261	296	750
R32	893	517*	711	25	118	847	334*	65	790	946
R33	8	121	215	23	914	796	325	484	408	427
R34	710	948	246*	263	535*	638	735	255	173	255*
R35	680	338	940	717	814*	427	931*	170	222	383
R36	536	196	637	251	191	578	772	503	375	698
R37	505	704	411*	266	247	368	253*	370	278*	511
R38	178	464	470	523*	237	572	648*	802	258	514
R39	840	791	258	729	86	254	766	214	997	148
R40	587	183	947	369	619	475	908	219*	165	736*
R41	816	751	856	924	918	211	333	778	856	307
R42	91	280	541	117	35	824	831	53	229*	780
R43	172	90	842	97	704*	443	358	503*	990	329
R44	671	899	997	138	120	807	373	994	733	662*
R45	325	134	291	213	887	562	210	598*	920	281
R46	404	0	79*	520	503	603*	466	697	587	449*
R47	44	797*	65	841	7	966	421	217	343*	38
R48	685	546	199	70	607	862	570	521	265	161
R49	904	199	430	602	139	386	164	877	768	299
CT	24886	22870	26774	22842	21551	26985	24125	22621	25680	24022
	C41	C42	C43	C44	C45	C46	C47	C48	C49	RT
R1	602	359	797	186	581	947	878	410	822	24680
R2	64	171	883	609	957	291	946	186	54	23402
R3	166	236	112	61	150	295	729	431	469*	24661
R4	162	761*	350	693	422	108	593	668	81	22953
R5	84	467	842	951	523	354	211	838*	255	22655
R6	501	423	559	859	724	986	157*	642	462	25655
R7	290	501	606	316	298	207	245	18	607	24102
R8	888*	310	368	734	278	556	187*	43	362	25415
R9	334	856	820	332	460	691	375	866	613*	25949
R10	164	876	786	324	937	367	902	81*	822	23052
R11	116	430	138	979	178	349	395	646	191	20684
R12	830	568	927	93	938	471	182	471	313	25554
R13	93	611	801*	499	109	863	523	709	868	26560
R14	981	453	251	849	303	533	334	179	882	27504
R15	623	166	667	792	723	327	686	933	779	25032
R16	591	846	745	618	285	534	858	260	578	24826
R17	758	957	964	641	598*	428	29	797	386	28880
R18	140	251	768	881*	806	342	872*	994	669	26566
R19	96	717	911	755	201	389	5 91	8	551	27245
R20	612	9	892	135	70	322	533	481	840	23636
R21	113	927*	886*	491	646	859	968*	472	797	27994
R22	708*	603	298	163	868	672	177	258*	231	19645
R23	57*	783*	368	513	894	812	280	842	876	23042
R24	32	135	739	957	975	437	487	515	314	23837
R25	190	201	981	173	524	926	326	583	759	21301
R26	256	464	587	174*	362	327*	576	472	847	23089
R27	238	433	999	931*	238	455	755	231	92	23843
R28	896	841	748	91	186	741	382	841	456*	26050
R29	497	870	94	949*	628*	547	852	155	6	24210
R30	269	523	495	9	959	399	460	545	377	22138

R31	434	108	160	684	450	672	374	873	80	22195
R32	669	24	525	874	393	390*	94	941*	438	19263
R33	59	727	559*	586	248*	928	586	592	418*	21600
R34	628	692	774	316	718	94	24	784	238	26870
R35	967	628	134	443	680	136	624	84	447	22254
R36	585*	287	169	964	373	181	371	911	454	25560
R37	397	577	480	627	854	395	180	197	13	22279
R38	643	492*	740	132	428	491	896	589	521*	23263
R39	74	113	585	826	239	954	274	820	659	23904
R40	30	369	763	916	563	1	272	654	823	23513
R41	257	742	434	698	652	785	550	762	821	27079
R42	743	763	191*	458	331*	780	194	98	838	23902
R43	184	726	683	601	356*	943	504*	138	737	24560
R44	948	807	817	593	842	356	12	133	940	24896
R45	736	271	257	550	815	327	213	122	467	26380
R46	829	207	65	628	808	316	980	320	210	21318
R47	627	826	174	335	945	418*	660*	877*	367	23626
R48	514	619	938	975	221	244	784	342	178	22103
R49	78	960	702	293	211	169	18	484	268	22764
CT	20753	25686	28532	27257	25948	24115	22513	25206	24276	1181489

Table C.8: 4by4by4 table with 10% of sensitive cells

Level 1					Level 2				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	3	819	583*	1405	R1	214*	148*	206	568
R2	997	389	328	1714	R2	824	4	785	1613
R3	710	138	288	1136	R3	951	324	798	2073
CT	1710	1346	1199*	4255	CT	1989	476	1789	4254

Level 3					Level 4				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	788*	636	706	2130	R1	1005	1603	1495	4103
R2	768*	429	435	1632	R2	2589	822	1548	4959
R3	707	789	215	1711	R3	2368	1251	1301	4920
CT	2263	1854	1356	5473	CT	5962	3676	4344	13982

Table C.9: 5by5by3 table with 10% of sensitive cells

Level 1						Level 2					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	220	119	168	637	1144	R1	663	854	614	769*	2900
R2	546	212	832	763	2353	R2	408	628*	763	701	2500
R3	818	197	766*	231	2012	R3	188*	212	914*	347	1661
R4	699*	559	450*	989*	2697	R4	17	606	30	41	694
CT	2283	1087	2216	2620	8206	CT	1276	2300	2321	1858	7755

Level 3					
	C1	C2	C3	C4	RT
R1	883	973	782	1406	4044
R2	954	840	1595	1464	4853
R3	1006	409	1680	578	3673
R4	716	1165	480	1030	3391
CT	3559	3387	4537	4478	15961

Table C.10: 4by4by7 table with 10% of sensitive cells

Level 1					Level 2				
	C1	C2	C3	RT		C1	C2	C3	RT

R1	917	10	199*	1126	R1	214	148	206	568
R2	419	20	429*	868	R2	824	4	785	1613
R3	542	594	660	1796	R3	951*	324	798*	2073
CT	1878	624	1288	3790	CT	1989	476	1789	4254
Level 3					Level 4				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	894*	203	747	1844	R1	58	199	445	702
R2	525	379	190	1094	R2	203	832	193*	1228
R3	698	900*	290	1888	R3	378	822*	341	1541
CT	2117	1482	1227	4826	CT	639	1853	979	3471
Level 5					Level 6				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	353	604*	932	1889	R1	813	272	466	1551
R2	672	503	682	1857	R2	838	709	303	1850
R3	860*	645	534	2039	R3	854	818	727*	2399
CT	1885	1752	2148	5785	CT	2505	1799	1496	5800
Level 7									
	C1	C2	C3	RT					
R1	3445	1427	2804	7676					
R2	3503	3124	2102	8729					
R3	3483	4276	2894	10653					
CT	10431	8827	7800	27058					

Table C.11: 5by5by5 table with 10% of sensitive cells

Level 1					Level 2						
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	876	894	284	583	2637	RT	737	199	469	423	1828
R2	433	530	783	794	2540	RT	226	641	681	59*	1607
R3	415	768*	439	320*	1942	R3	305	971*	498	960*	2734
R4	745	683*	134	370	1932	R4	268*	213	207	575	1263
CT	2469	2875	1640	2067	9051	CT	1536	2024	1855	2017	7432
Level 3					Level 4						
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	137	299	65*	516	1017	R1	12	661	988	334	1995
R2	580	209	461	603	1853	R2	760*	380	568	50	1758
R3	874*	990	214	727	2805	R3	15	789	643	412	1859
R4	440	839	607*	451*	2337	R4	933	629	630	44	2236
CT	2031	2337	1347	2297	8012	CT	1720	2459	2829	840	7848
Level 5											
	C1	C2	C3	C4	RT						
R1	1762	2053	1806	1856	7477						
R2	1999	1760	2493	1506	7758						
R3	1609	3518	1794	2419	9340						
R4	2386	2364	1578	1440	7768						
CT	7756	9695	7671	7221	32343						

Table C.12: 5by5by7 table with 10% of sensitive cells

Level 1					Level 2						
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	203	747	525	379*	1854	R1	199*	445	203	832	1679
R2	190	698	900	290*	2078	R2	193	378	822	341*	1734
R3	568	621*	980	12	2181	R3	370	795	271	894	2330
R4	469	334	641	568	2012	R4	65*	433	209	794	1501
CT	1430	2400	3046	1249	8125	CT	827	2051	1505	2861	7244
Level 3					Level 4						

	C1	C2	C3	C4	RT			C1	C2	C3	C4	RT
R1	604	932	672	503	2711	R1		272	466	838*	709	2285
R2	682	860	645	534*	2721	R2		303	854	818*	727	2702
R3	703	957*	252	199	2111	R3		547	523	876	299*	2245
R4	988*	226	380	59	1653	R4		583	580	783	603	2549
CT	2977	2975	1949	1295	9196	CT		1705	2423	3315	2338	9781

Level 5

Level 6

	C1	C2	C3	C4	RT			C1	C2	C3	C4	RT
R1	199*	419	20	429	1067	R1		15*	846	681*	305	1847
R2	542	594	660*	309	2105	R2		151	497	342	838	1828
R3	445	880	737	661	2723	R3		695	173	137	284	1289
R4	423	760*	681*	50	1914	R4		516	530*	461	415	1922
CT	1609	2653	2098	1449	7809	CT		1377	2046	1621	1842	6886

Level 7

	C1	C2	C3	C4	RT							
R1	1492	3855	2939	3157	11443							
R2	2061	3881	4187	3039	13168							
R3	3328	3949	3253	2349	12879							
R4	3044	2863	3155	2489	11551							
CT	9925	14548	13534	11034	49041							

Table C.13: 10by10by3 table with 10% of sensitive cells

Level 1

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	191	856	816	457*	12	6	49	650	553	3990
R2	199*	733	10	754*	920*	368	731	905	632	5252
R3	549	335	392	699*	414	838	425	566*	511	4729
R4	489*	701	807*	485	665*	140	823	999	59*	5168
R5	549	597*	571	962	740	634	84	916	254	5307
R6	513	422	72*	292	336	53	498	562	113	2861
R7	755	815	201	626	60	271	474	596	478	4276
R8	161	956	29	610	92	376	833	452*	147	3656
R9	769	621	640	353*	491	464	71	608	621*	4638
CT	4175	6036	3538	5238	4130	3150	3988	6254	3368	39877

Level 2

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	422*	490	461	451	902	297	693	983	400	5099
R2	625	376	420*	794*	845*	621	194	569	234	4678
R3	932	656*	627	397	655	372	595	717	776	5727
R4	186	983*	704	115	365	567	674	962*	360	4916
R5	262	49	701	751	432	803	945	602	873	5418
R6	733	961	553	858	680	357	434	617*	898*	6091
R7	791	670*	273	537	89	409*	909	329	597	4604
R8	829	596	812	701*	425	166	839	957*	870	6195
R9	444*	952	247	188	409	611	314*	175	246	3586
CT	5224	5733	4798	4792	4802	4203	5597	5911	5254	46314

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	613	1346	1277	908	1314	303	742	1633	953	9089
R2	824	1109	430	1548	1765	989	925	1474	866	9930
R3	1481	991	1019	1096	1069	1210	1020	1283	1287	10456
R4	675	1684	1511	600	1030	707	1497	1961	419	10084
R5	811	646	1272	1713	1172	1437	1029	1518	1127	10725

R6	1246	1383	625	1150	1016	410	932	1179	1011	8952
R7	1546	1485	474	1163	149	680	1383	925	1075	8880
R8	990	1552	841	1311	517	542	1672	1409	1017	9851
R9	1213	1573	887	541	900	1075	385	783	867	8224
CT	9399	11769	8336	10030	8932	7353	9585	12165	8622	86191

Table C.14: 10by10by4 table with 10% of sensitive cells

Level 1

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	596	222*	933	450	356	895	933	702	455	5542
R2	562*	868	200	122	770	47	289	65	63	2986
R3	212	673	666	15	985	621*	822*	660	605	5259
R4	637*	623	877	779	679*	12*	458*	509	380*	4954
R5	314	503*	918	908	763*	754	272*	36	340	4808
R6	229	891	373	502	674	111	24	520	251	3575
R7	522	335	149	726	285*	241	230	135	117	2740
R8	180	537	13	254	594	864	792	192	772	4198
R9	483	2	213	408	942*	311	323	400	525	3607
CT	3735	4654	4342	4164	6048	3856	4143	3219	3508	37669

Level 2

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	816	704	713	172	49	286	131	848	81	3800
R2	319	372	49	522	375*	598	889	234	264	3622
R3	498	958	131	288*	17*	560	263	214	660	3589
R4	170	686	13	307	74*	227	703	74	276	2530
R5	638*	948	113*	156	722	663	419*	81	466	4206
R6	862	488	531	422	957	565	870	192	934	5821
R7	895	437	136	399*	869*	978	681	22	69	4486
R8	32*	276	889*	569	331*	568	153	639	380	3837
R9	608	790*	103	408	150	169	734	506*	641	4109
CT	4838	5659	2678	3243	3544	4614	4843	2810	3771	36000

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	977	522*	228	969	755	251	941	209*	851	5703
R2	375	74	567	117	823	949	102	933	1000	4940
R3	290	767	95	817	819	244	754*	602	183	4571
R4	540	677	310*	927	71	516	582	193	771*	4587
R5	987*	828*	812	122	652	883	213	851	914	6262
R6	657*	993*	181	660	192	969	27	716	137	4532
R7	942	471	532	358	626	641	666	262*	853	5351
R8	734	368	866	159	659	980*	833	669	442	5710
R9	176	514	157	53	384*	897	411	169	16	2777
CT	5678	5214	3748	4182	4981	6330	4529	4604	5167	44433

Level 4

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	2389	1448	1874	1591	1160	1432	2005	1759	1387	15045
R2	1256	1314	816	761	1968	1594	1280	1232	1327	11548
R3	1000	2398	892	1120	1821	1425	1839	1476	1448	13419
R4	1347	1986	1200	2013	824	755	1743	776	1427	12071
R5	1939	2279	1843	1186	2137	2300	904	968	1720	15276

R6	1748	2372	1085	1584	1823	1645	921	1428	1322	13928
R7	2359	1243	817	1483	1780	1860	1577	419	1039	12577
R8	946	1181	1768	982	1584	2412	1778	1500	1594	13745
R9	1267	1306	473	869	1476	1377	1468	1075	1182	10493
CT	14251	15527	10768	11589	14573	14800	13515	10633	12446	118102

Table C.15: 10by10by4 table with 10% of sensitive cells

Level 1

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	954	829	441	493	366	738	764	550	48	5183
R2	666	800	713	990	58	292	616	827	649	5611
R3	133	566	385	196	742	903	41	65	762	3793
R4	136	339	994	93	573	384	246	161	362	3288
R5	245	767	208	509	359	277	444	944	309	4062
R6	54	37	587	364	783	789	919	438	622	4593
R7	111	679	132	331	329	983	332	643	744	4284
R8	867	302	437	163	442	552	700	617	828	4908
R9	34	918	925	500	599	634	691	740	296	5337
CT	3200	5237	4822	3639	4251	5552	4753	4985	4620	41059

Level 2

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	881	971	6	83	139	27	583	777	985	4452
R2	462	289	720	152	539	91	46	611	115	3025
R3	464	255	757	512	995	20	286	879	655	4823
R4	755	10	906	241	185	862	898	883	231	4971
R5	468	45	252	625	276	579	423	912	558	4138
R6	115	313	924	976	466	634	781	592	895	5696
R7	403	980	845	575	488	966	8	322	299	4886
R8	625	152	676	552	45	934	939	518	318	4759
R9	857	388	678	430	221	945	981	819	82	5401
CT	5030	3403	5764	4146	3354	5058	4945	6313	4138	42151

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	699	300	292	196	15	102	685	610	205	3104
R2	48	695	733	203	190	507	952	847	473	4648
R3	71	238	575	713	867	637	677	488	971	5237
R4	313	334	350	902	738	857	604	979	450	5527
R5	497	165	397	626	678	823	996	815	637	5634
R6	846	817	330	717	232	660	327	115	964	5008
R7	97	399	233	137	432	949	166	26	181	2620
R8	55	309	823	525	965	899	452	364	596	4988
R9	683	472	641	829	399	6	70	261	875	4236
CT	3309	3729	4374	4848	4516	5440	4929	4505	5352	41002

Level 4

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	305	998	692	978	641	893	967	989	912	7375
R2	460	259	622	819	599	884	169	114	683	4609
R3	581	16	408	867	486	944	586	891	171	4950
R4	463	247	278	324	112	819	140	601	643	3627
R5	75	777	481	991	509	941	614	690	769	5847

R6	172	235	206	643	318	538	739	319	20	3190
R7	293	821	67	868	261	452	828	355	415	4360
R8	404	3	756	919	13	618	177	773	782	4445
R9	802	88	996	117	301	817	45	704	77	3947
CT	3555	3444	4506	6526	3240	6906	4265	5436	4472	42350

Level 5

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	2839	3098	1431	1750	1161	1760	2999	2926	2150	20114
R2	1636	2043	2788	2164	1386	1774	1783	2399	1920	17893
R3	1249	1075	2125	2288	3090	2504	1590	2323	2559	18803
R4	1667	930	2528	1560	1608	2922	1888	2624	1686	17413
R5	1285	1754	1338	2751	1822	2620	2477	3361	2273	19681
R6	1187	1402	2047	2700	1799	2621	2766	1464	2501	18487
R7	904	2879	1277	1911	1510	3350	1334	1346	1639	16150
R8	1951	766	2692	2159	1465	3003	2268	2272	2524	19100
R9	2376	1866	3240	1876	1520	2402	1787	2524	1330	18921
CT	15094	15813	19466	19159	15361	22956	18892	21239	18582	166562

Table C.16: 11by11 table with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	RT
R1	798	436	552	522	248	665	608	531	447	141	4948
R2	139	944	822	605	833	876	978	572	683	594	7046
R3	46	177	626	890	46	878	507	144	114	856	4284
R4	951	716	49	187	966	271	661	673	447	443	5364
R5	410	964	181	173	697	256	820	321	526	35	4383
R6	924	337	310	717	759	109	814	418	82	439	4909
R7	479	403	186	423	733	569	821	750	65	313	4742
R8	898	608	71	363	827	528	651	53	167	331	4497
R9	158	281	682	548	828	332	738	274	338	831	5010
R10	866	558	765	984	14	939	323	394	882	136	5861
CT	5669	5424	4244	5412	5951	5423	6921	4130	3751	4119	51044

Table C.17: 16by16 with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	944	104	875	503	153	598	326	546	262	301
R2	360	685	13	497	28	511	784	509	5	222
R3	938	639	955	806	449	987	796	777	465	460
R4	193	460	881	636	682	389	465	816	365	304
R5	92	829	615	639	678	112	603	604	357	889
R6	560	904	835	978	565	684	313	734	674	569
R7	901	96	514	998	9	464	518	784	74	861
R8	160	86	617	267	8	318	448	48	37	436
R9	800	920	769	875	472	832	788	627	310	13
R10	264	497	987	166	972	386	981	32	670	319
R11	636	495	280	85	196	629	462	79	646	185
R12	602	267	468	876	43	824	758	372	822	883
R13	521	964	82	979	417	497	317	268	632	75
R14	590	487	649	432	958	867	590	21	161	316
R15	720	543	270	160	761	929	155	749	429	600
CT	8281	7976	8810	8897	6391	9027	8304	6966	5909	6433
	C11	C12	C13	C14	C15	RT				
R1	189	791	701	795	627	715				
R2	33	518	275	655	187	282				
R3	842	415	919	230	699	0377				
R4	793	969	394	393	248	988				

R5	148	390	836	330	818	940				
R6	514	311	676	466	152	935				
R7	678	909	222	242	944	214				
R8	876	562	173	607	814	457				
R9	770	708	309	825	389	407				
R10	788	276	637	432	840	247				
R11	377	445	407	126	791	839				
R12	914	47	374	931	8	189				
R13	627	144	973	451	750	697				
R14	418	1000	445	419	422	775				
R15	36	613	11	279	636	891				
CT	8003	8098	7352	7181	8325	15953				

Table C.18: 21by21 with 10% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
R1	904	578	542	255	921	897	392	755	129	980	485
R2	684	153	539	819	39	50	524	473	175	745	129
R3	441	759	167	636	883	222	5	877	130	571	512
R4	469	774	84	269	921	376	119	331	426	908	860
R5	327	29	119	223	24	174	595	906	185	607	187
R6	347	434	968	990	48	754	528	824	196	261	57
R7	153	406	228	683	977	376	405	25	27	441	622
R8	423	225	137	308	347	288	761	98	150	85	162
R9	558	277	966	554	558	99	251	889	636	648	985
R10	945	343	737	40	453	454	320	189	417	80	978
R11	959	433	44	211	564	245	13	434	305	531	289
R12	425	53	1	532	583	627	612	214	430	452	971
R13	353	496	887	684	596	880	220	675	75	414	420
R14	40	688	125	737	408	385	754	945	220	252	735
R15	623	996	869	921	880	611	986	978	485	839	129
R16	974	724	544	279	955	122	707	462	556	460	233
R17	560	341	316	25	121	785	380	116	145	355	762
R18	197	458	573	470	927	213	638	344	342	389	865
R19	670	234	934	91	593	22	824	18	224	254	111
R20	348	894	950	13	217	589	868	8	62	737	685
CT	10400	9295	9730	8740	11015	8169	9902	9561	5315	10009	10177
	C12	C13	C14	C15	C16	C17	C18	C19	C20	RT	
R1	263	655	577	656	1	540	489	884	112	11015	
R2	358	270	904	115	626	421	616	896	934	9470	
R3	621	438	63	618	673	444	61	922	461	9504	
R4	687	102	645	410	165	520	672	268	62	9068	
R5	372	534	627	595	795	207	337	163	953	7959	
R6	526	477	674	127	833	391	460	492	655	10042	
R7	946	709	511	992	331	704	356	529	505	9926	
R8	744	89	627	185	290	680	728	143	475	6945	
R9	19	208	619	887	692	539	728	830	1000	11943	
R10	754	756	800	908	434	997	253	471	41	10370	
R11	582	295	589	787	849	891	476	43	279	8819	
R12	893	282	36	882	294	129	384	192	966	8958	
R13	720	943	746	807	369	923	155	520	985	11868	
R14	851	102	314	237	852	92	100	670	920	9427	
R15	663	11	920	587	548	604	888	141	272	12951	
R16	853	58	28	680	2	546	927	413	377	9900	

R17	364	721	579	98	195	581	510	810	21	7785	
R18	572	487	574	515	178	759	280	251	451	9483	
R19	989	892	266	280	918	475	164	889	966	9814	
R20	234	205	606	990	622	784	622	286	519	10239	
CT	12011	8234	10705	11356	9667	11227	9206	9813	10954	195486	

Table C.19: 26by26 with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
R1	407	405	888	1	433	519	959	479	168	199
R2	896	713	946	165	545	358	633	308	648	821
R3	355	834	741	7	975	13	283	693	953	248
R4	587	870	464	14	674	485	512	532	761	27
R5	633	178	865	513	257	212	241	542	889	479
R6	600	921	428	80	853	599	635	461	648	970
R7	788	494	556	952	256	261	235	374	475	880
R8	282	115	467	857	54	523	87	752	636	865
R9	208	989	461	259	644	864	936	972	147	670
R10	120	283	814	992	390	827	293	529	940	860
R11	873	922	965	475	497	741	229	855	138	69
R12	758	940	979	104	920	574	211	335	397	529
R13	349	265	869	606	699	226	847	724	805	154
R14	868	418	68	280	12	373	606	372	738	940
R15	239	923	280	298	564	866	578	370	102	989
R16	584	761	674	486	339	939	384	61	670	638
R17	421	587	72	19	300	114	592	872	134	588
R18	859	367	697	438	898	4	471	556	947	786
R19	269	499	444	684	464	542	950	367	390	465
R20	683	796	951	843	475	174	231	724	76	90
R21	901	814	874	357	86	194	434	187	314	145
R22	203	632	550	676	644	821	76	308	207	147
R23	264	44	789	856	617	765	641	889	682	285
R24	637	177	621	686	93	81	113	427	975	572
R25	406	331	310	737	565	750	828	586	668	483
CT	13190	14278	15773	11385	12254	11825	12005	13275	13508	12899
	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
R1	392	624	622	837	540	24	692	369	409	88
R2	838	226	412	450	24	435	902	490	226	88
R3	66	252	326	447	297	572	857	770	361	507
R4	555	553	602	664	504	641	829	720	188	219
R5	579	597	973	677	356	249	338	948	998	790
R6	312	787	112	875	282	888	323	503	442	947
R7	860	475	155	925	946	792	18	881	928	954
R8	95	739	935	895	55	798	925	662	116	727
R9	336	466	56	35	653	668	998	686	449	41
R10	684	171	727	589	532	753	981	115	630	588
R11	930	679	725	357	809	929	579	193	312	800
R12	780	236	284	475	953	677	787	481	336	78
R13	478	666	374	485	443	455	796	649	7	835
R14	713	844	67	883	726	58	736	711	183	578
R15	606	517	343	455	428	873	147	866	2	481
R16	767	263	51	31	95	808	544	649	417	256

R17	315	865	538	873	114	839	276	83	25	810
R18	23	41	124	314	160	993	606	28	240	720
R19	56	263	441	262	140	182	165	514	990	334
R20	76	951	748	429	696	652	456	746	533	992
R21	827	118	915	897	107	934	290	873	836	204
R22	30	687	490	651	78	966	569	762	689	700
R23	232	884	269	272	460	934	763	606	781	882
R24	82	803	622	220	145	425	956	137	417	894
R25	156	219	215	964	907	430	469	416	726	798
CT	10788	12926	11126	13962	10450	15975	15002	13858	11241	14311
	C21	C22	C23	C24	C25	RT				
R1	207	770	72	677	411	11192				
R2	192	532	801	817	442	12908				
R3	793	80	840	627	944	12841				
R4	237	770	31	516	695	12650				
R5	579	209	969	180	759	14010				
R6	338	26	537	680	407	13654				
R7	993	739	486	440	439	15302				
R8	704	323	134	790	445	12981				
R9	185	928	748	78	560	13037				
R10	912	990	576	644	186	15126				
R11	278	264	246	25	361	13251				
R12	679	559	597	554	97	13320				
R13	720	147	956	619	533	13707				
R14	928	72	205	308	311	11998				
R15	659	471	213	621	244	12135				
R16	791	975	666	46	886	12781				
R17	552	412	791	135	882	11209				
R18	348	585	239	824	379	11647				
R19	338	572	837	668	36	10872				
R20	357	826	714	579	148	13946				
R21	477	451	266	388	15	11904				
R22	209	582	937	301	89	12004				
R23	126	973	42	992	551	14599				
R24	847	573	866	946	959	13274				
R25	199	984	447	845	424	13863				
CT	12648	13813	13216	13300	11203	324211				

Table C.20: 31by31 with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
R1	679	289	245	902	588	918	129	701	213	408	948
R2	280	333	831	988	677	52	984	387	938	262	543
R3	816	759	533	934	43	369	343	454	9	158	107
R4	186	973	188	177	287	999	237	927	215	99	463
R5	868	390	391	364	860	106	658	512	68	110	526
R6	344	522	569	836	913	836	223	317	323	974	537
R7	600	39	407	60	126	561	416	629	405	542	791
R8	122	629	171	997	40	991	243	103	332	919	646
R9	763	130	855	992	48	335	425	856	253	774	413
R10	34	373	288	9	489	334	176	875	418	420	163
R11	523	412	258	428	21	881	762	635	944	750	967
R12	867	612	884	784	575	520	356	485	932	888	68
R13	344	210	423	925	571	840	874	871	61	497	62
R14	396	359	656	130	53	33	143	333	245	429	128
R15	559	258	107	838	250	514	888	520	491	66	910
R16	462	234	852	404	941	936	262	878	616	643	713
R17	671	394	675	950	98	788	732	670	311	221	592

R18	315	765	698	426	875	947	887	1000	647	678	675
R19	106	778	271	983	343	437	93	187	360	191	48
R20	30	807	999	109	665	500	882	931	885	230	648
R21	755	593	924	462	416	520	512	550	107	421	457
R22	605	437	509	192	477	858	379	529	306	51	550
R23	977	159	312	579	264	527	721	182	129	586	972
R24	50	95	272	738	656	919	722	842	609	956	379
R25	419	969	156	22	792	550	11	646	329	874	979
R26	424	20	389	155	303	63	531	359	748	482	982
R27	368	505	175	65	110	89	199	906	620	858	860
R28	454	198	703	249	401	705	553	3	824	132	603
R29	208	337	936	849	686	909	751	603	439	653	837
R30	46	786	394	206	830	779	86	760	737	31	953
CT	13271	13365	15071	15753	13398	17816	14178	17651	13514	14303	17520
	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	
R1	625	901	445	318	800	433	636	269	191	543	
R2	8	625	242	365	483	847	129	687	723	549	
R3	996	475	80	458	226	354	907	831	20	871	
R4	264	20	969	923	272	354	264	743	912	466	
R5	927	424	870	110	530	52	629	763	358	966	
R6	140	523	313	614	97	356	236	655	768	480	
R7	640	609	712	422	459	391	434	119	100	107	
R8	876	546	782	963	275	906	985	395	463	362	
R9	922	217	287	243	475	728	679	759	722	703	
R10	961	684	858	833	265	641	904	547	999	534	
R11	554	367	883	539	626	208	55	175	839	346	
R12	671	194	751	590	478	963	291	113	862	242	
R13	56	74	353	987	695	760	846	394	922	558	
R14	481	373	906	985	615	53	874	111	787	162	
R15	942	909	328	534	894	569	980	908	450	467	
R16	874	818	594	879	356	461	258	406	176	890	
R17	868	525	569	333	532	879	999	134	904	191	
R18	786	681	952	763	165	170	836	989	339	476	
R19	822	942	277	848	208	860	60	842	42	208	
R20	673	773	998	625	388	969	802	735	783	644	
R21	84	167	19	172	799	789	390	374	672	370	
R22	448	205	998	161	716	818	382	530	942	360	
R23	983	995	399	65	539	550	251	645	366	143	
R24	806	42	406	190	772	469	836	720	505	948	
R25	819	253	569	624	118	220	849	421	903	55	
R26	753	461	991	946	772	829	120	788	274	307	
R27	937	746	356	123	347	920	719	268	251	690	
R28	992	593	497	798	925	240	377	383	519	620	
R29	33	396	335	694	611	180	782	39	975	168	
R30	37	289	664	759	182	899	628	197	389	507	
CT	18978	14827	17403	16864	14620	16868	17138	14940	17156	13933	
	C22	C23	C24	C25	C26	C27	C28	C29	C30	RT	
R1	70	689	225	119	210	757	61	665	316	14293	
R2	251	366	402	583	127	30	774	421	441	14328	
R3	281	625	983	411	880	173	410	430	771	14707	
R4	706	509	284	695	195	542	108	711	111	13799	
R5	339	907	14	444	279	5	998	690	332	14490	

R6	183	522	156	901	865	833	757	827	446	16066	
R7	555	754	994	34	883	906	939	322	890	14846	
R8	874	86	812	184	284	757	329	963	406	16441	
R9	386	36	501	569	480	599	888	798	89	15925	
R10	705	536	117	164	191	315	375	362	448	14018	
R11	484	585	339	46	185	397	732	935	779	15655	
R12	437	746	994	315	70	753	778	617	694	17530	
R13	45	804	98	562	685	338	381	701	356	15293	
R14	648	884	7	40	139	836	781	907	743	13237	
R15	37	501	149	452	616	831	630	241	866	16705	
R16	349	937	481	129	425	295	696	277	536	16778	
R17	740	405	393	587	936	563	730	730	238	17358	
R18	427	232	362	61	548	462	963	976	194	18295	
R19	226	68	147	844	834	351	73	322	888	12659	
R20	170	41	539	728	306	74	314	115	577	16940	
R21	769	977	593	805	787	498	323	649	785	15739	
R22	233	995	17	910	884	834	884	823	710	16743	
R23	965	5	749	45	887	303	372	712	521	14903	
R24	810	964	288	826	802	485	909	400	8	17424	
R25	143	43	803	676	392	554	758	292	304	14543	
R26	675	467	534	633	832	395	157	685	531	15606	
R27	444	178	863	632	630	495	963	831	680	15828	
R28	722	638	201	742	382	729	833	854	706	16576	
R29	534	233	53	645	333	671	308	659	592	15449	
R30	196	257	130	880	988	1	99	49	194	12953	
CT	13404	14990	12228	14662	16055	14782	17323	17964	15152	465127	

Table C.21: 41by41 with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
R1	812	660	228	355	509	735	766	354	476	412	177
R2	208	539	618	898	984	666	162	10	353	836	705
R3	509	251	44	246	387	875	818	641	743	98	7
R4	561	357	586	965	562	462	841	184	839	950	404
R5	886	816	644	782	380	157	261	47	295	307	854
R6	57	931	106	287	579	659	378	362	706	18	544
R7	832	819	93	345	138	655	151	676	521	188	707
R8	617	485	151	897	430	878	362	879	517	939	125
R9	774	999	422	963	714	486	902	175	807	612	998
R10	689	65	315	361	405	178	228	132	836	662	255
R11	69	962	304	825	420	726	497	246	464	890	754
R12	696	714	825	735	29	237	573	528	10	46	529
R13	78	689	413	280	534	729	377	64	951	367	661
R14	498	281	194	384	439	280	162	771	174	352	988
R15	296	924	251	66	321	38	327	201	127	642	139
R16	151	599	692	42	659	760	349	997	569	713	622
R17	158	141	496	740	304	672	709	773	217	626	772
R18	284	640	469	667	879	701	397	625	545	971	945
R19	931	45	318	933	664	347	320	18	772	748	245
R20	268	25	789	9	961	548	829	712	736	465	730
R21	178	817	416	616	551	297	425	78	951	182	835
R22	890	564	373	501	630	815	394	828	24	683	318
R23	615	316	959	515	825	462	354	685	364	930	746
R24	849	921	264	877	914	616	999	889	513	592	216
R25	132	256	569	488	683	225	59	207	672	910	676
R26	93	396	418	993	322	845	662	238	712	900	806

R27	883	890	139	768	490	584	599	259	47	728	671
R28	133	225	393	716	466	187	874	130	280	823	495
R29	473	290	920	851	677	569	34	123	349	447	956
R30	354	566	221	699	105	698	278	966	771	69	280
R31	654	679	190	68	95	92	629	866	315	502	757
R32	267	60	648	390	218	854	971	739	737	322	639
R33	449	576	54	13	578	730	739	863	606	880	674
R34	295	427	802	144	800	273	779	901	371	651	703
R35	669	550	502	880	104	575	693	316	83	782	450
R36	401	573	195	286	84	440	849	197	734	219	55
R37	136	795	635	616	756	182	717	520	298	760	112
R38	162	792	68	383	217	166	664	593	986	922	587
R39	322	750	835	671	402	112	850	168	866	323	586
R40	177	381	462	140	53	401	234	739	78	841	78
CT	17506	21766	17021	21395	19268	19912	21212	18700	20415	23308	21801
	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	
R1	935	586	82	812	1	6	83	378	366	937	
R2	955	451	284	950	647	13	681	423	56	177	
R3	226	958	279	602	28	275	195	633	295	752	
R4	550	854	452	132	105	945	820	962	278	319	
R5	250	237	241	48	524	183	397	509	571	419	
R6	880	527	767	445	130	863	20	531	155	47	
R7	763	716	999	642	203	311	281	794	390	286	
R8	579	157	581	669	33	881	839	175	899	71	
R9	449	855	971	483	241	163	954	163	102	480	
R10	163	565	684	603	819	848	621	845	41	361	
R11	244	496	816	192	29	976	435	649	727	113	
R12	928	937	169	909	314	120	410	438	639	249	
R13	231	92	143	275	111	66	74	551	180	420	
R14	793	10	710	243	655	687	200	788	50	925	
R15	424	873	317	893	658	470	874	838	909	435	
R16	709	499	807	279	442	25	179	857	211	490	
R17	335	220	392	445	328	918	586	473	880	848	
R18	403	828	983	396	802	175	835	681	390	803	
R19	475	196	881	573	397	142	388	571	829	569	
R20	472	439	533	194	638	866	976	255	767	234	
R21	282	419	822	827	151	143	82	489	729	277	
R22	110	425	929	399	42	315	62	568	968	50	
R23	431	684	894	818	143	678	347	361	137	269	
R24	223	325	663	122	637	273	768	385	520	874	
R25	873	128	111	940	731	601	831	427	236	469	
R26	752	199	294	608	656	338	573	871	711	834	
R27	168	174	75	904	575	722	83	310	241	378	
R28	597	776	838	546	884	403	677	40	865	521	
R29	59	595	909	178	883	746	758	644	350	625	
R30	537	763	654	677	995	91	194	724	199	245	
R31	742	865	890	635	904	247	120	882	275	499	
R32	980	426	817	717	80	410	127	326	370	702	
R33	122	856	44	338	907	270	623	491	942	540	
R34	1	437	215	327	681	434	977	747	30	254	
R35	975	521	562	914	678	849	129	789	372	181	
R36	956	91	64	251	773	589	447	600	793	579	

R37	703	772	601	391	578	746	714	959	469	261	
R38	734	791	621	680	357	791	431	511	827	902	
R39	669	522	116	641	704	21	790	745	270	585	
R40	55	541	408	74	647	8	465	851	712	583	
CT	20733	20806	21618	20772	19111	17608	19046	23234	18751	18563	
	C22	C23	C24	C25	C26	C27	C28	C29	C30	C31	
R1	919	836	896	972	592	356	410	409	899	898	
R2	401	2	469	630	845	127	346	206	36	943	
R3	77	14	433	797	822	733	277	515	94	868	
R4	550	962	71	539	538	823	518	481	246	901	
R5	949	377	154	639	412	176	587	382	526	790	
R6	940	870	226	487	551	805	245	133	514	281	
R7	880	267	718	705	929	938	758	614	379	411	
R8	604	172	182	439	68	852	119	940	448	476	
R9	64	368	889	964	686	331	260	473	445	258	
R10	298	610	386	260	704	54	301	896	990	404	
R11	472	45	392	0	599	884	786	159	841	612	
R12	199	688	156	607	799	266	805	265	989	944	
R13	810	314	416	771	324	481	3	800	911	372	
R14	524	803	239	429	801	433	895	167	906	167	
R15	671	87	136	53	540	41	293	636	945	772	
R16	607	890	481	928	83	567	227	86	266	893	
R17	645	646	407	827	315	342	128	557	452	842	
R18	614	497	887	188	889	466	407	211	335	195	
R19	566	306	57	126	605	414	841	225	645	231	
R20	147	629	513	960	507	752	420	597	812	437	
R21	498	494	144	541	478	650	6	522	595	422	
R22	308	206	754	704	165	901	190	608	541	980	
R23	392	594	449	885	212	802	391	483	355	821	
R24	455	211	205	898	569	333	49	758	259	350	
R25	581	417	773	418	588	998	250	187	87	916	
R26	974	699	93	131	480	862	513	7	986	864	
R27	146	306	965	497	392	354	290	191	656	425	
R28	815	872	185	745	936	640	406	262	427	239	
R29	910	684	701	179	678	442	453	119	5	989	
R30	264	929	764	642	163	232	253	133	697	3	
R31	867	583	389	507	58	147	175	225	659	896	
R32	53	754	92	204	886	941	690	249	383	152	
R33	821	689	242	586	87	232	353	294	939	268	
R34	329	875	36	78	217	298	368	724	185	516	
R35	530	781	480	384	144	815	529	561	192	704	
R36	965	570	107	295	722	17	293	418	171	322	
R37	281	657	692	20	706	196	655	23	307	595	
R38	929	273	692	35	434	823	215	38	329	241	
R39	615	712	322	364	314	597	270	448	35	247	
R40	177	455	6	185	25	577	773	851	897	531	
CT	21847	21144	16199	19619	19863	20698	15748	15853	20384	22176	
	C32	C33	C34	C35	C36	C37	C38	C39	C40	RT	
R1	753	766	208	962	723	421	876	861	389	22816	
R2	151	269	91	998	766	868	122	306	532	18724	
R3	230	903	165	142	782	713	673	902	653	18655	
R4	995	72	602	871	303	876	369	895	847	23587	

R5	861	336	912	406	744	429	354	644	545	19031	
R6	456	139	415	273	944	362	950	92	383	18058	
R7	106	850	548	542	876	904	713	630	647	22925	
R8	861	196	506	642	279	833	14	685	866	20346	
R9	879	132	491	190	912	589	239	980	823	22686	
R10	891	182	366	439	555	766	379	649	871	19677	
R11	829	252	953	542	38	348	112	430	819	19947	
R12	614	555	180	992	236	767	168	689	635	20589	
R13	810	560	215	352	651	975	596	270	562	17479	
R14	945	452	528	351	75	124	981	354	561	19319	
R15	465	378	340	935	880	197	645	853	941	19831	
R16	750	896	480	52	963	906	609	66	638	21039	
R17	772	138	982	329	279	949	754	39	25	20461	
R18	824	427	226	999	790	321	889	944	957	24485	
R19	206	99	336	326	462	338	958	994	584	18681	
R20	576	491	519	923	312	240	536	288	43	21148	
R21	749	720	672	231	136	265	694	802	520	18706	
R22	3	576	993	492	953	919	592	842	907	21522	
R23	536	78	9	176	31	722	409	802	848	20528	
R24	504	735	273	581	582	349	720	143	816	21230	
R25	374	429	502	133	772	989	644	68	346	19696	
R26	479	178	613	700	188	579	821	878	429	22695	
R27	792	470	290	890	819	514	634	112	222	18653	
R28	405	640	516	661	707	107	274	926	856	21488	
R29	355	554	840	728	313	898	974	373	753	22384	
R30	231	172	597	743	521	612	524	838	930	19334	
R31	427	223	153	140	585	681	147	231	676	18675	
R32	409	18	365	246	243	728	701	604	474	18992	
R33	808	995	839	117	422	632	150	61	964	20794	
R34	42	31	886	659	614	700	437	349	123	17716	
R35	245	378	95	337	431	587	635	591	375	20368	
R36	786	353	822	367	332	232	663	133	214	16958	
R37	328	131	891	140	449	189	668	301	787	19737	
R38	937	530	819	324	35	284	560	812	736	21231	
R39	151	533	438	520	785	23	687	449	452	18910	
R40	960	410	16	507	876	306	533	739	775	17527	
CT	22495	16247	19692	19958	21364	22242	22404	21625	24524	806628	

Table C.22: 51by51 with 30% of sensitive cells

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
R1	797	388	496	675	213	755	870	206	970	134	556
R2	176	271	322	17	627	761	627	134	848	960	968
R3	899	595	501	756	582	9	591	796	636	87	0
R4	874	471	827	716	438	797	601	4	799	680	953
R5	764	843	193	979	192	88	377	345	259	498	996
R6	816	379	477	958	303	836	359	788	724	89	999
R7	238	569	505	525	659	540	194	50	103	862	93
R8	759	337	833	380	796	249	509	967	875	934	975
R9	149	501	864	655	85	966	450	141	147	399	185
R10	886	762	186	201	570	107	717	101	11	100	627
R11	575	384	641	165	329	996	721	924	759	522	230
R12	404	381	299	665	378	352	581	387	394	98	602
R13	513	377	821	72	948	16	326	442	868	459	26
R14	366	825	827	712	655	435	615	72	270	109	876
R15	431	208	469	177	248	614	395	49	781	108	585
R16	739	81	787	250	899	23	422	242	874	938	685
R17	898	454	299	499	825	125	578	822	433	821	364

R18	362	243	361	395	804	120	390	156	260	272	362
R19	303	154	928	781	57	401	677	904	574	259	251
R20	349	339	13	717	478	660	691	767	1	947	108
R21	196	232	536	722	777	218	462	859	68	552	490
R22	240	392	979	937	798	732	828	987	561	32	370
R23	655	307	112	565	658	347	386	816	4	900	685
R24	876	845	563	677	849	138	138	762	94	897	726
R25	418	577	250	982	128	325	303	26	77	707	788
R26	328	252	177	653	648	367	522	239	58	776	258
R27	529	892	665	430	409	496	964	195	601	45	326
R28	615	692	837	629	639	41	458	980	220	484	926
R29	629	21	3	74	323	385	676	373	89	847	489
R30	993	713	183	266	59	865	142	945	291	528	794
R31	659	465	508	754	867	152	487	11	885	417	737
R32	703	218	659	333	115	949	350	564	816	648	824
R33	637	562	460	657	926	801	149	239	238	571	278
R34	21	173	329	146	495	478	803	172	478	630	28
R35	19	421	755	194	250	678	975	324	205	596	614
R36	71	253	559	311	719	919	987	932	649	112	527
R37	370	442	937	56	126	526	273	351	671	751	218
R38	847	868	411	443	245	861	114	36	335	411	394
R39	810	262	439	371	636	432	246	619	289	169	765
R40	621	445	321	655	554	225	61	817	859	250	865
R41	11	765	536	95	110	976	433	231	212	517	479
R42	87	173	954	394	556	686	175	325	706	991	609
R43	942	14	884	376	39	117	603	730	780	296	928
R44	687	750	777	290	402	315	284	26	270	906	132
R45	239	737	933	599	347	287	686	444	561	234	200
R46	855	606	675	664	181	883	873	539	420	705	587
R47	74	372	453	350	303	750	71	226	213	407	759
R48	857	113	827	350	745	89	852	112	338	677	725
R49	280	892	36	146	256	798	972	746	45	612	35
R50	828	437	810	791	805	412	553	7	886	436	5
CT	26395	22453	27217	24205	24051	24098	25517	21930	22505	25380	26002
	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	
R1	781	921	47	875	469	641	125	992	878	290	
R2	237	170	60	173	634	551	210	937	616	18	
R3	578	301	358	122	379	166	681	16	405	520	
R4	362	6	526	463	434	163	728	672	331	87	
R5	583	741	982	177	329	794	71	680	688	333	
R6	873	595	603	586	919	294	790	642	483	764	
R7	118	960	464	462	224	784	420	998	205	314	
R8	198	512	77	39	617	166	262	848	88	774	
R9	826	41	176	476	661	933	784	554	460	669	
R10	234	440	717	970	320	275	180	706	642	880	
R11	946	707	209	959	421	731	237	886	310	332	
R12	293	204	184	920	842	982	738	100	234	399	
R13	38	749	702	616	570	524	682	410	654	812	
R14	900	3	902	877	980	994	322	180	147	672	
R15	561	99	84	229	457	698	154	370	368	703	
R16	470	423	573	426	634	226	727	911	347	63	
R17	406	956	569	871	868	840	930	271	175	216	

R18	819	869	279	939	162	991	496	363	684	965	
R19	517	616	951	241	495	335	386	571	231	138	
R20	139	966	700	49	941	38	612	160	443	83	
R21	499	67	263	808	814	538	229	698	372	236	
R22	797	853	635	230	354	120	320	965	841	705	
R23	436	856	402	396	333	318	468	199	938	757	
R24	9	802	918	437	867	824	910	324	370	659	
R25	842	957	728	293	992	7	207	915	859	6	
R26	565	541	939	682	73	25	46	258	884	420	
R27	898	724	420	207	552	161	895	730	15	562	
R28	612	59	169	261	212	686	656	857	5	827	
R29	457	867	495	545	281	444	463	153	737	539	
R30	584	467	529	502	958	285	894	707	195	444	
R31	801	255	249	276	779	854	241	988	444	775	
R32	713	147	923	178	861	468	743	340	921	664	
R33	827	565	738	75	394	849	877	308	963	106	
R34	789	228	514	300	81	512	212	607	112	139	
R35	114	36	377	371	789	285	857	326	121	430	
R36	498	174	18	248	366	51	559	522	149	271	
R37	327	974	602	534	207	213	166	136	700	420	
R38	88	439	755	741	731	145	953	867	14	523	
R39	207	557	295	608	984	958	650	301	155	770	
R40	924	449	358	692	909	299	588	244	794	43	
R41	433	829	5	323	887	716	244	910	463	794	
R42	447	121	716	262	813	203	732	38	838	414	
R43	683	331	794	347	460	401	541	589	9	801	
R44	605	929	967	890	50	107	520	87	36	938	
R45	575	9	446	427	321	27	28	813	862	459	
R46	80	479	427	207	78	3	21	768	490	175	
R47	196	297	774	476	615	753	521	236	399	61	
R48	120	524	2	911	874	335	303	683	187	42	
R49	785	610	942	465	630	716	475	222	571	222	
R50	235	893	959	374	670	444	392	754	884	307	
CT	25025	25318	25522	23536	28291	22873	24246	26812	22717	22541	
	C22	C23	C24	C25	C26	C27	C28	C29	C30	C31	
R1	510	837	13	528	542	35	893	378	515	699	
R2	143	845	617	492	438	534	941	284	622	564	
R3	591	660	965	548	214	46	264	161	2	86	
R4	805	684	678	692	809	906	456	641	139	759	
R5	680	348	320	842	890	763	876	529	652	646	
R6	718	120	474	432	158	711	764	11	178	879	
R7	343	460	70	981	672	669	60	242	641	622	
R8	198	396	112	216	122	783	540	977	548	442	
R9	294	653	346	422	801	965	267	140	456	1000	
R10	602	586	900	224	2	704	525	600	583	538	
R11	679	26	544	586	996	98	725	756	302	414	
R12	570	243	23	930	96	982	199	192	834	465	
R13	381	274	329	381	216	611	857	58	92	252	
R14	287	321	510	346	770	869	841	50	968	791	
R15	131	193	708	537	384	199	727	329	548	221	
R16	784	185	633	393	113	766	197	47	726	843	
R17	767	667	837	812	235	603	906	866	82	245	

R18	896	124	931	950	72	565	893	343	67	602	
R19	287	378	504	558	350	518	127	64	742	860	
R20	513	421	33	736	221	311	866	783	994	315	
R21	966	310	448	228	123	564	503	431	255	897	
R22	149	718	130	680	779	764	232	937	334	118	
R23	42	162	246	137	772	132	803	841	188	805	
R24	184	206	293	332	832	707	704	686	662	650	
R25	681	7	12	886	356	139	465	986	264	526	266
R26	265	902	126	381	13	606	763	541	537	331	
R27	499	28	902	624	183	554	56	340	913	145	
R28	414	224	801	105	879	768	206	152	788	775	
R29	845	206	928	967	640	67	585	357	307	21	
R30	279	525	760	203	92	758	906	850	605	346	
R31	343	420	74	748	445	557	115	193	277	953	
R32	6	225	708	712	721	19	703	234	509	112	
R33	472	225	42	71	778	757	556	930	105	668	
R34	983	350	410	629	349	320	90	697	880	341	
R35	135	241	123	379	150	200	746	560	815	137	
R36	361	85	615	825	915	129	223	893	275	601	
R37	506	368	108	364	841	112	642	720	279	142	
R38	881	171	996	281	113	934	20	291	757	366	
R39	348	495	61	28	2	201	437	770	603	890	
R40	829	995	158	355	843	675	651	849	735	391	
R41	866	531	218	126	522	730	901	528	710	255	
R42	662	158	329	737	457	490	585	15	111	786	
R43	557	104	768	297	888	668	614	253	558	467	
R44	614	965	405	806	888	88	474	131	654	719	
R45	560	45	238	940	831	826	149	318	202	831	
R46	770	227	477	765	827	619	268	284	66	738	
R47	301	973	573	625	537	983	750	198	326	980	
R48	101	808	762	588	183	280	328	294	847	722	
R49	245	724	195	157	423	347	290	927	381	114	
R50	424	841	504	637	478	695	34	187	650	289	
CT	24467	21365	22831	25689	23744	26653	26244	22122	24546	26099	
	C32	C33	C34	C35	C36	C37	C38	C39	C40	C41	
R1	952	883	809	464	44	363	227	130	10	330	
R2	741	786	325	662	529	715	782	829	376	180	
R3	992	511	115	124	856	923	226	584	93	625	
R4	52	700	155	527	960	963	58	267	307	19	
R5	596	480	130	19	769	430	703	148	169	149	
R6	906	9	383	424	946	458	934	360	91	75	
R7	202	585	117	275	562	499	563	966	259	762	
R8	686	365	460	703	190	656	88	406	98	843	
R9	135	152	571	980	88	235	493	925	363	804	
R10	650	887	25	991	949	72	131	947	352	907	
R11	515	522	230	787	74	117	563	306	895	270	
R12	611	784	706	787	950	381	148	627	48	17	
R13	44	836	843	978	391	39	139	762	247	40	
R14	322	399	590	143	200	125	162	553	967	361	
R15	206	408	741	63	174	516	565	297	722	411	
R16	456	74	331	659	102	265	401	825	568	546	
R17	551	122	597	85	551	777	583	141	28	518	

R18	134	819	282	911	344	444	774	110	883	992	
R19	207	902	444	327	952	627	975	670	429	325	
R20	839	308	8	466	51	797	125	207	103	106	
R21	989	766	812	941	144	827	569	361	298	34	
R22	138	421	917	784	487	513	931	93	393	35	
R23	496	790	633	382	150	240	992	970	9	558	
R24	841	956	119	396	433	984	86	565	152	124	
R25	537	738	478	502	479	546	648	730	612	471	
R26	22	618	503	776	79	373	929	670	73	689	
R27	618	528	370	911	873	534	531	595	701	166	
R28	827	876	24	748	665	503	17	974	150	951	
R29	445	215	562	949	813	394	473	628	883	27	
R30	77	904	108	504	155	148	871	171	294	785	
R31	288	873	198	474	443	441	705	717	657	36	
R32	590	559	681	344	93	223	260	294	974	379	
R33	46	179	193	75	828	505	136	53	322	83	
R34	938	956	335	70	569	5	404	400	741	196	
R35	512	339	333	831	933	346	280	224	783	568	
R36	461	661	757	279	998	725	706	591	10	631	
R37	907	936	437	73	461	794	552	605	74	665	
R38	969	857	884	193	718	106	891	454	599	403	
R39	740	482	58	683	602	937	326	4	414	270	
R40	14	528	714	959	356	511	386	173	352	760	
R41	382	817	228	729	14	501	813	284	254	812	
R42	294	681	768	949	434	151	751	6	630	274	
R43	763	671	188	140	811	21	631	294	352	319	
R44	947	328	407	857	165	239	470	652	330	364	
R45	307	708	50	669	234	433	588	586	135	850	
R46	85	222	244	869	872	809	662	344	747	691	
R47	881	492	142	459	751	92	876	281	629	850	
R48	238	992	122	596	649	382	873	390	529	299	
R49	614	889	479	237	204	780	365	882	644	97	
R50	689	41	685	596	327	27	397	845	891	355	
CT	25452	29555	20291	27350	24422	22492	25759	23896	20640	21022	
	C42	C43	C44	C45	C46	C47	C48	C49	C50	RT	
R1	38	290	604	878	47	740	501	697	715	25751	
R2	730	89	475	295	479	743	847	309	18	24707	
R3	637	720	849	910	81	874	432	826	759	23652	
R4	540	954	647	680	326	259	632	6	436	25989	
R5	504	251	619	446	491	326	406	842	614	25550	
R6	550	905	529	317	540	6	579	402	356	26492	
R7	156	575	274	423	532	137	809	51	838	22632	
R8	454	636	453	570	537	679	875	392	689	25309	
R9	685	436	188	246	108	196	420	901	636	24028	
R10	82	203	339	773	158	814	245	181	380	23982	
R11	917	39	733	349	763	172	712	211	856	26141	
R12	648	697	300	13	620	935	499	636	605	23983	
R13	467	897	524	782	343	275	914	754	588	23939	
R14	689	898	602	617	589	528	483	837	506	27063	
R15	902	650	286	161	442	938	189	258	361	20055	
R16	492	981	179	439	351	961	16	665	345	24083	
R17	76	256	861	959	373	962	460	365	860	27365	

R18	385	783	963	25	940	38	941	831	505	26839	
R19	480	336	974	466	855	148	655	309	613	24852	
R20	384	268	581	870	889	100	779	460	977	22712	
R21	748	101	712	528	919	357	513	823	316	25119	
R22	458	930	71	725	981	206	434	967	119	27120	
R23	813	202	135	251	423	904	237	116	451	23418	
R24	272	51	79	64	862	813	292	604	552	26186	
R25	380	324	936	356	276	778	403	910	970	26742	
R26	233	160	0	823	482	664	811	179	686	21946	
R27	619	900	403	31	644	653	567	429	586	25619	
R28	681	921	547	517	269	818	175	269	994	26903	
R29	406	564	73	809	990	853	156	567	245	23865	
R30	947	828	564	955	385	998	858	939	989	28148	
R31	834	211	951	624	581	52	393	656	166	25029	
R32	857	633	173	246	139	598	395	160	191	23875	
R33	59	110	222	920	661	331	851	213	793	22404	
R34	782	491	844	925	850	619	913	224	130	22688	
R35	572	456	802	926	688	949	989	217	84	23055	
R36	161	482	470	508	816	285	435	78	535	23406	
R37	271	31	984	277	178	158	633	891	242	22251	
R38	942	968	971	697	82	243	823	356	270	26457	
R39	637	94	617	746	604	484	773	541	82	23452	
R40	830	777	625	667	766	129	476	68	307	26852	
R41	147	426	969	618	823	138	358	94	570	24333	
R42	832	990	83	856	674	21	722	441	358	24485	
R43	390	531	531	587	398	203	149	460	850	24128	
R44	698	881	866	820	127	579	712	650	841	26645	
R45	957	491	270	824	414	684	790	543	540	24247	
R46	820	543	122	206	244	200	903	680	942	24962	
R47	37	668	276	483	809	99	935	622	349	24283	
R48	892	822	589	406	224	425	944	171	22	24144	
R49	665	269	248	199	113	362	1	468	402	22177	
R50	489	669	445	738	821	155	637	582	510	26520	
CT	27245	26388	25558	27551	25707	23589	28672	23851	25749	1235583	

Table C.23: 4by4by4 table with 30% of sensitive cells

Level 1					Level 2				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	948	308	516	1772	R1	509	884	83	1476
R2	421	840	610	1871	R2	2	14	603	619
R3	352	742	580	1674	R3	788	242	641	1671
CT	1721	1890	1706	5317	CT	1299	1140	1327	3766

Level 3					Level 4				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	673	953	54	1680	R1	2130	2145	653	4928
R2	824	340	291	1455	R2	1247	1194	1504	3945
R3	834	466	566	1866	R3	1974	1450	1787	5211
CT	2331	1759	911	5001	CT	5351	4789	3944	14084

Table C.24: 5by5by3 table with 30% of sensitive cells

Level 1						Level 2					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	760	922	443	995	3120	R1	684	968	764	110	2526
R2	61	114	190	148	513	R2	700	80	659	665	2104
R3	715	859	92	319	1985	R3	777	732	348	802	2659
R4	678	700	735	543	2656	R4	179	110	238	405	932

CT	2214	2595	1460	2005	8274	CT	2340	1890	2009	1982	8221
Level 3											
	C1	C2	C3	C4	RT						
R1	1444	1890	1207	1105	5646						
R2	761	194	849	813	2617						
R3	1492	1591	440	1121	4644						
R4	857	810	973	948	3588						
CT	4554	4485	3469	3987	16495						

Table C.25: 4by4by7 table with 30% of sensitive cells

Level 1					Level 2				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	951	396	358	1705	R1	180	721	26	927
R2	465	90	29	584	R2	253	726	253	1232
R3	5	906	6	917	R3	250	187	767	1204
CT	1421	1392	393	3206	CT	683	1634	1046	3363
Level 3					Level 4				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	866	871	50	1787	R1	474	51	61	586
R2	407	163	21	591	R2	3	209	546	758
R3	342	657	745	1744	R3	814	134	556	1504
CT	1615	1691	816	4122	CT	1291	394	1163	2848
Level 5					Level 6				
	C1	C2	C3	RT		C1	C2	C3	RT
R1	945	375	241	1561	R1	466	896	336	1698
R2	599	60	236	895	R2	927	257	696	1880
R3	7	808	247	1062	R3	484	330	178	992
CT	1551	1243	724	3518	CT	1877	1483	1210	4570
Level 7									
	C1	C2	C3	RT					
R1	3882	3310	1072	8264					
R2	2654	1505	1781	5940					
R3	1902	3022	2499	7423					
CT	8438	7837	5352	21627					

Table C.26: 5by5by7 table with 30% of sensitive cells

Level 1						Level 2					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	234	656	397	372	1659	R1	549	392	414	425	1780
R2	717	186	704	365	1972	R2	511	701	485	140	1837
R3	674	360	49	751	1834	R3	999	549	571	740	2859
R4	803	602	733	553	2691	R4	84	254	422	292	1052
CT	2428	1804	1883	2041	8156	CT	2143	1896	1892	1597	7528
Level 3						Level 4					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	932	627	655	595	2809	R1	335	699	838	566	2438
R2	776	983	115	567	2441	R2	489	807	665	823	2784
R3	962	262	701	432	2357	R3	59	597	962	634	2252
R4	945	873	961	858	3637	R4	916	513	72	336	1837
CT	3615	2745	2432	2452	11244	CT	1799	2616	2537	2359	9311
Level 5											
	C1	C2	C3	C4	RT						
R1	2050	2374	2304	1958	8686	R1					

R2	2493	2677	1969	1895	9034	R2					
R3	2694	1768	2283	2557	9302	R3					
R4	2748	2242	2188	2039	9217	R4					
CT	9985	9061	8744	8449	36239	CT					

Table C.27: 5by5by7 table with 30% of sensitive cells

Level 1						Level 2					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	959	733	931	368	2991	R1	485	357	685	950	2477
R2	875	313	686	864	2738	R2	820	720	256	969	2765
R3	7	691	812	596	2106	R3	170	21	480	998	1669
R4	422	571	218	53	1264	R4	242	972	28	237	1479
CT	2263	2308	2647	1881	9099	CT	1717	2070	1449	3154	8390

Level 3						Level 4					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	642	357	868	107	1974	R1	150	954	982	556	2642
R2	413	910	335	183	1841	R2	801	54	70	928	1853
R3	3	272	186	871	1332	R3	302	219	797	430	1748
R4	491	134	382	75	1082	R4	843	404	512	832	2591
CT	1549	1673	1771	1236	6229	CT	2096	1631	2361	2746	8834

Level 5						Level 6					
	C1	C2	C3	C4	RT		C1	C2	C3	C4	RT
R1	960	681	196	206	2043	R1	288	238	837	350	1713
R2	655	891	111	194	1851	R2	784	503	629	527	2443
R3	84	167	555	380	1186	R3	399	33	85	908	1425
R4	368	341	69	286	1064	R4	71	602	683	39	1395
CT	2067	2080	931	1066	6144	CT	1542	1376	2234	1824	6976

Level 7					
	C1	C2	C3	C4	RT
R1	3484	3320	4499	2537	13840
R2	4348	3391	2087	3665	13491
R3	965	1403	2915	4183	9466
R4	2437	3024	1892	1522	8875
CT	11234	11138	11393	11907	45672

Table C.28: 10by10by3 table with 30% of sensitive cells

Level 1										
	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	301	102	469	859	462	22	796	236	685	3932
R2	439	99	436	154	136	691	776	325	307	3363
R3	346	354	813	312	748	100	536	131	708	4048
R4	816	764	103	851	405	474	494	510	935	5352
R5	459	734	57	636	427	70	508	799	196	3886
R6	215	172	464	597	935	462	300	75	469	3689
R7	14	738	931	513	416	856	914	868	432	5682
R8	224	376	286	327	226	306	895	146	733	3519
R9	544	670	686	898	906	978	29	188	209	5108
CT	3358	4009	4245	5147	4661	3959	5248	3278	4674	38579

Level 2										
	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	965	596	717	186	903	760	314	494	945	5880
R2	497	554	181	934	999	961	1	544	673	5344
R3	981	20	983	379	443	993	135	864	998	5796

R4	782	905	212	743	305	145	530	227	518	4367
R5	777	602	72	876	708	19	869	799	714	5436
R6	787	206	75	191	308	830	710	750	945	4802
R7	54	636	461	424	388	823	836	776	436	4834
R8	13	839	284	476	962	779	254	341	34	3982
R9	558	584	766	967	179	898	282	609	998	5841
CT	5414	4942	3751	5176	5195	6208	3931	5404	6261	46282

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	1266	698	1186	1045	1365	782	1110	730	1630	9812
R2	936	653	617	1088	1135	1652	777	869	980	8707
R3	1327	374	1796	691	1191	1093	671	995	1706	9844
R4	1598	1669	315	1594	710	619	1024	737	1453	9719
R5	1236	1336	129	1512	1135	89	1377	1598	910	9322
R6	1002	378	539	788	1243	1292	1010	825	1414	8491
R7	68	1374	1392	937	804	1679	1750	1644	868	10516
R8	237	1215	570	803	1188	1085	1149	487	767	7501
R9	1102	1254	1452	1865	1085	1876	311	797	1207	10949
CT	8772	8951	7996	10323	9856	10167	9179	8682	10935	84861

Table C.29: 10by10by4 table with 30% of sensitive cells

Level 1

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	696	513	893	822	973	563	266	292	287	5305
R2	870	523	326	940	497	721	31	335	25	4268
R3	677	425	616	408	211	888	770	109	211	4315
R4	142	293	167	811	19	986	325	676	822	4241
R5	261	561	253	215	470	193	235	635	689	3512
R6	757	400	120	414	395	197	580	802	635	4300
R7	302	512	559	493	128	345	559	558	257	3713
R8	735	191	704	773	757	658	412	444	257	4931
R9	961	223	316	400	92	293	86	35	701	3107
CT	5401	3641	3954	5276	3542	4844	3264	3886	3884	37692

Level 2

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	221	576	93	994	923	951	61	402	220	4441
R2	255	129	361	434	971	648	366	7	424	3595
R3	328	150	629	706	591	886	368	937	421	5016
R4	322	80	387	406	957	251	57	952	224	3636
R5	230	826	240	665	506	274	128	182	326	3377
R6	401	154	29	792	826	510	979	433	367	4491
R7	432	991	487	969	842	738	783	205	278	5725
R8	497	725	802	873	905	1	348	840	156	5147
R9	951	497	738	400	602	466	513	980	803	5950
CT	3637	4128	3766	6239	7123	4725	3603	4938	3219	41378

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	625	287	292	68	959	318	333	34	566	3482
R2	592	4	994	936	958	863	94	816	34	5291
R3	560	64	177	430	837	963	68	511	830	4440
R4	818	928	486	795	855	483	291	40	1000	5696
R5	525	288	551	309	688	926	413	403	923	5026
R6	196	65	865	816	338	210	467	241	431	3629

R7	691	184	397	827	789	60	141	860	288	4237
R8	55	974	206	807	421	982	668	530	558	5201
R9	735	7	790	252	832	46	715	497	517	4391
CT	4797	2801	4758	5240	6677	4851	3190	3932	5147	41393

Level 4

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	1542	1376	1278	1884	2855	1832	660	728	1073	13228
R2	1717	656	1681	2310	2426	2232	491	1158	483	13154
R3	1565	639	1422	1544	1639	2737	1206	1557	1462	13771
R4	1282	1301	1040	2012	1831	1720	673	1668	2046	13573
R5	1016	1675	1044	1189	1664	1393	776	1220	1938	11915
R6	1354	619	1014	2022	1559	917	2026	1476	1433	12420
R7	1425	1687	1443	2289	1759	1143	1483	1623	823	13675
R8	1287	1890	1712	2453	2083	1641	1428	1814	971	15279
R9	2647	727	1844	1052	1526	805	1314	1512	2021	13448
CT	13835	10570	12478	16755	17342	14420	10057	12756	12250	120463

Table C.30: 10by10by4 table with 30% of sensitive cells

Level 1

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	954	829	441	493	366	738	764	550	48	5183
R2	666	800	713	990	58	292	616	827	649	5611
R3	133	566	385	196	742	903	41	65	762	3793
R4	136	339	994	93	573	384	246	161	362	3288
R5	245	767	208	509	359	277	444	944	309	4062
R6	54	37	587	364	783	789	919	438	622	4593
R7	111	679	132	331	329	983	332	643	744	4284
R8	867	302	437	163	442	552	700	617	828	4908
R9	34	918	925	500	599	634	691	740	296	5337
CT	3200	5237	4822	3639	4251	5552	4753	4985	4620	41059

Level 2

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	881	971	6	83	139	27	583	777	985	4452
R2	462	289	720	152	539	91	46	611	115	3025
R3	464	255	757	512	995	20	286	879	655	4823
R4	755	10	906	241	185	862	898	883	231	4971
R5	468	45	252	625	276	579	423	912	558	4138
R6	115	313	924	976	466	634	781	592	895	5696
R7	403	980	845	575	488	966	8	322	299	4886
R8	625	152	676	552	45	934	939	518	318	4759
R9	857	388	678	430	221	945	981	819	82	5401
CT	5030	3403	5764	4146	3354	5058	4945	6313	4138	42151

Level 3

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	699	300	292	196	15	102	685	610	205	3104
R2	48	695	733	203	190	507	952	847	473	4648
R3	71	238	575	713	867	637	677	488	971	5237
R4	313	334	350	902	738	857	604	979	450	5527
R5	497	165	397	626	678	823	996	815	637	5634
R6	846	817	330	717	232	660	327	115	964	5008
R7	97	399	233	137	432	949	166	26	181	2620
R8	55	309	823	525	965	899	452	364	596	4988

R9	683	472	641	829	399	6	70	261	875	4236
CT	3309	3729	4374	4848	4516	5440	4929	4505	5352	41002

Level 4

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	305	998	692	978	641	893	967	989	912	7375
R2	460	259	622	819	599	884	169	114	683	4609
R3	581	16	408	867	486	944	586	891	171	4950
R4	463	247	278	324	112	819	140	601	643	3627
R5	75	777	481	991	509	941	614	690	769	5847
R6	172	235	206	643	318	538	739	319	20	3190
R7	293	821	67	868	261	452	828	355	415	4360
R8	404	3	756	919	13	618	177	773	782	4445
R9	802	88	996	117	301	817	45	704	77	3947
CT	3555	3444	4506	6526	3240	6906	4265	5436	4472	42350

Level 5

	C1	C2	C3	C4	C5	C6	C7	C8	C9	RT
R1	2839	3098	1431	1750	1161	1760	2999	2926	2150	20114
R2	1636	2043	2788	2164	1386	1774	1783	2399	1920	17893
R3	1249	1075	2125	2288	3090	2504	1590	2323	2559	18803
R4	1667	930	2528	1560	1608	2922	1888	2624	1686	17413
R5	1285	1754	1338	2751	1822	2620	2477	3361	2273	19681
R6	1187	1402	2047	2700	1799	2621	2766	1464	2501	18487
R7	904	2879	1277	1911	1510	3350	1334	1346	1639	16150
R8	1951	766	2692	2159	1465	3003	2268	2272	2524	19100
R9	2376	1866	3240	1876	1520	2402	1787	2524	1330	18921
CT	15094	15813	19466	19159	15361	22956	18892	21239	18582	166562

Appendix D

An Example to Illustrate the Dynamic Program

We have six nodes where the start depot is node 0, the end depot is node 5, and the other four nodes are indexed from 1 to 4. The graph is given in Fig D.1.

The time matrix, distance matrix, time windows, and service times are given in Tables D.1a, D.1b, and D.1c.

We start with $S = \emptyset$ and generate the states (S, i) . The states at different iterations are shown in Table D.2(a) to D.2(d).

If more than one node is in the set S , the nodes are stored in an increasing order. The cost associated with each state is the minimum of the costs considering all orderings of the nodes in S . For example, in Table D.1d, the first state $(1, 2, 3, 4)$ has $S = \{1, 2, 3\}$ and the terminal stop 4 after S , and it is associated with six different partial paths: $(0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4)$, $(0 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 4)$, $(0 \rightarrow 2 \rightarrow 1 \rightarrow 3 \rightarrow 4)$, $(0 \rightarrow 2 \rightarrow 3 \rightarrow 1 \rightarrow 4)$, $(0 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 4)$, and $(0 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 4)$. Notice that the last stop is fixed, which is node 4. The cost of $(1, 2, 3, 4)$ is the minimum of the six partial paths.

Initialization.

For $i = 1, 2, 3, 4$, we calculate the initial service start time, denoted by $Resst_i$, if the driver visits stop i immediately after the driver departs the start depot.

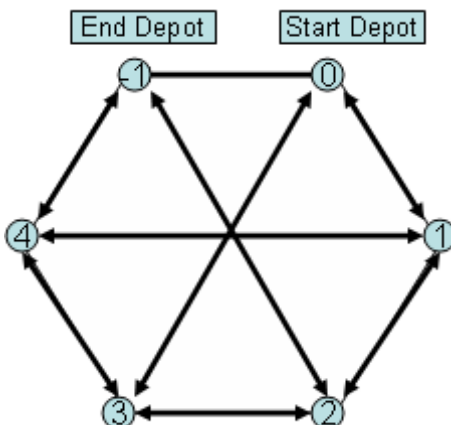
$$Resst_1 = \max\{a_0 + T_{01} + s_0, a_1\} = \max\{1450 + 13.90 + 3.36, 1450\} = 1467.26 \quad (D.1)$$

$$Resst_2 = \max\{a_0 + T_{02} + s_0, a_2\} = \max\{1450 + 34.59 + 3.36, 1450\} = 1487.95 \quad (D.2)$$

$$Resst_3 = \max\{a_0 + T_{03} + s_0, a_3\} = \max\{1450 + 33.25 + 3.36, 1450\} = 1486.61 \quad (D.3)$$

$$Resst_4 = \max\{a_0 + T_{04} + s_0, a_4\} = \max\{1450 + 31.76 + 3.36, 1450\} = 1485.12 \quad (D.4)$$

Figure D.1: An example with six nodes



The values of the parameters are $dp_D = \$0.5$, $dp_T = \$0.24$, and $p = \$100$.

We then calculate the cost function $f(\emptyset, j) = dp_D \cdot (Resst_j - a_0) + dp_T \cdot D_{0j} + p \cdot$

$\max(Resst_j - b_j, 0)$, for $j = 1, 2, 3, 4$.

$$f(\emptyset, 1) = 0.24(1467.26 - 1450) + 0.5(2.78) + 100 \cdot \max(1467.26 - 1750, 0) = 5.532$$

$$f(\emptyset, 2) = 0.24(1487.95 - 1450) + 0.5(6.92) + 100 \cdot \max(1487.95 - 1750, 0) = 12.57$$

$$f(\emptyset, 3) = 0.24(1486.61 - 1450) + 0.5(7.24) + 100 \cdot \max(1486.61 - 1750, 0) = 12.41$$

$$f(\emptyset, 4) = 0.24(1485.12 - 1450) + 0.5(5.45) + 100 \cdot \max(1485.12 - 1750, 0) = 11.83$$

Iteration 1.

If $S = \{1\}$, there are three possible terminating nodes in the partial path, (nodes 2, 3, and 4). The real earliest service start times of terminal stops are given by the following.

$$Resst_{12} = \max\{Resst_1 + T_{12} + s_1, a_2\} = \max\{1467.26 + 25.78 + 4.06, 1450\} = 1497.10$$

$$Resst_{13} = \max\{Resst_1 + T_{13} + s_1, a_3\} = \max\{1467.26 + 24.43 + 4.06, 1450\} = 1495.75$$

$$Resst_{14} = \max\{Resst_1 + T_{14} + s_1, a_4\} = \max\{1467.26 + 22.95 + 4.06, 1450\} = 1494.26$$

$Resst_1$ is given in (D.1). We use $Resst_{12}$ to denote the real service start time of node 2, if its predecessor is node 1. The real service start time of node j depends on its predecessor, if the set S is nonempty. For example, the real service start time of

Table D.1: Inputs for example with six nodes

Node	0	1	2	3	4	5
0	0	13.90	34.59	33.25	31.76	29.77
1	13.90	0	25.78	24.43	22.95	26.19
2	34.59	25.78	0	12.95	13.57	18.94
3	33.25	24.43	12.95	0	1.49	15.14
4	31.76	22.95	13.57	1.49	0	13.65
5	29.77	26.19	18.94	15.14	13.65	0

(a) Time matrix

Node	0	1	2	3	4	5
0	0	2.78	6.92	7.24	6.80	7.87
1	2.78	0	5.16	5.48	5.03	6.78
2	6.92	5.16	0	2.63	2.82	4.64
3	7.24	5.48	2.63	0	0.45	4.47
4	6.71	5.03	2.82	0.45	0	4.03
5	7.87	6.78	4.64	4.47	4.03	0

(b) Distance matrix

Node	a_i	b_i	s_i
0	1450	1700	3.36
1	1450	1750	4.06
2	1450	1825	3.36
3	1450	1750	4.06
4	1450	1750	3.36
5	1450	1716	3.56

(c) Time widows and service times

node 2, given $S = \{1\}$, is different from the real service start time of node 2, given $S = \{3\}$. It is confusing if we use $Resst_2$ to denote the two different times. We use $Resst_{12}$ and $Resst_{32}$ to denote the two different times. We use $Resst_{ij}$ to denote the real service start time of node j if its predecessor is node i to avoid any confusion. This notation is not necessary in the initialization step (if $S = \emptyset$, the predecessor of any node is the start depot).

Since $|S| = 1$, there is only one cost associated with the state $(\{1\}, j)$. Therefore $f(\{1\}, j) = f(\emptyset, 1) + dp_D \cdot (Resst_{1j} - Resst_1) + dp_T \cdot D_{1j} + p \cdot \max(Resst_{1j} - b_j, 0)$.

For $j = 2, 3, 4$, no time window is violated. We have

Table D.2: State set

State	S	Terminating Node
$K_0(1)$	\emptyset	1
$K_0(2)$	\emptyset	2
$K_0(3)$	\emptyset	3
$K_0(4)$	\emptyset	4

(a) Initial state set

State	Set S	Terminating Node
$K_1(1)$	1	2
$K_1(2)$	1	3
$K_1(3)$	1	4
$K_1(4)$	2	1
$K_1(5)$	2	3
$K_1(6)$	2	4
$K_1(7)$	3	1
$K_1(8)$	3	2
$K_1(9)$	3	4
$K_1(10)$	4	1
$K_1(11)$	4	2
$K_1(12)$	4	3

(b) State set at iteration 1

State	Set S	Terminating Node
$K_2(1)$	1 2	3
$K_2(2)$	1 2	4
$K_2(3)$	1 3	2
$K_2(4)$	1 3	4
$K_2(5)$	1 4	2
$K_2(6)$	1 4	3
$K_2(7)$	2 3	1
$K_2(8)$	2 3	4
$K_2(9)$	2 4	1
$K_2(10)$	2 4	3
$K_2(11)$	3 4	1
$K_2(12)$	3 4	2

(c) State set at iteration 2

State	Set S	Terminating Node
$K_3(1)$	1 2 3	4
$K_3(2)$	1 2 4	3
$K_3(3)$	1 3 4	2
$K_3(4)$	2 3 4	1

(d) State set at iteration 3

$$\begin{aligned}
f(\{1\}, 2) &= 5.53 + 0.24(1497.10 - 1467.26) + 0.5(5.16) = 15.27 \\
f(\{1\}, 3) &= 5.53 + 0.24(1495.75 - 1467.26) + 0.5(5.48) = 15.10 \\
f(\{1\}, 4) &= 5.53 + 0.24(1494.26 - 1467.26) + 0.5(5.04) = 14.53
\end{aligned}$$

If $S = \{2\}$, we have

$$\begin{aligned}
Resst_{21} &= \max\{Resst_2 + T_{21} + s_2, a_1\} = \max\{1487.95 + 25.78 + 3.36, 1450\} = 1517.09 \\
Resst_{23} &= \max\{Resst_2 + T_{23} + s_2, a_3\} = \max\{1487.95 + 12.59 + 3.36, 1450\} = 1504.27 \\
Resst_{24} &= \max\{Resst_2 + T_{24} + s_2, a_4\} = \max\{1487.95 + 13.57 + 3.36, 1450\} = 1504.89
\end{aligned}$$

$$f(\{2\}, j) = f(\emptyset, 2) + dp_D \cdot (Resst_{2j} - Resst_2) + dp_T \cdot D_{2j} + p \cdot \max(Resst_j - b_j, 0).$$

For $j = 1, 3, 4$, we have

$$\begin{aligned}
f(\{2\}, 1) &= 12.57 + 0.24(1517.09 - 1487.95) + 0.5(5.16) + 0 = 22.14 \\
f(\{2\}, 3) &= 12.57 + 0.24(1504.27 - 1487.95) + 0.5(2.63) + 0 = 17.80 \\
f(\{2\}, 4) &= 12.57 + 0.24(1504.89 - 1487.95) + 0.5(2.82) + 0 = 18.04
\end{aligned}$$

If $S = \{3\}$, we have

$$\begin{aligned}
Resst_{31} &= \max\{Resst_3 + T_{31} + s_3, a_1\} = \max\{1486.61 + 24.43 + 4.06, 1450\} = 1515.10 \\
Resst_{32} &= \max\{Resst_3 + T_{32} + s_3, a_2\} = \max\{1486.61 + 12.59 + 4.06, 1450\} = 1503.62 \\
Resst_{34} &= \max\{Resst_3 + T_{34} + s_3, a_4\} = \max\{1486.61 + 1.490 + 4.06, 1450\} = 1492.16
\end{aligned}$$

$$f(\{3\}, j) = f(\emptyset, 3) + dp_D \cdot (Resst_{3j} - Resst_3) + dp_T \cdot D_{3j} + p \cdot \max(Resst_j - b_j, 0).$$

For $j = 1, 2, 4$, we have

$$\begin{aligned}
f(\{3\}, 1) &= 12.41 + 0.24(1515.10 - 1487.95) + 0.5(5.16) + 0 = 21.99 \\
f(\{3\}, 2) &= 12.41 + 0.24(1503.62 - 1487.95) + 0.5(2.63) + 0 = 17.81 \\
f(\{3\}, 4) &= 12.41 + 0.24(1492.16 - 1487.95) + 0.5(2.82) + 0 = 13.96
\end{aligned}$$

If $S = \{4\}$, we have

$$\begin{aligned}
Resst_{41} &= \max\{Resst_4 + T_{41} + s_4, a_1\} = \max\{1485.12 + 22.95 + 3.36, 1450\} = 1511.43 \\
Resst_{42} &= \max\{Resst_4 + T_{42} + s_4, a_2\} = \max\{1485.12 + 13.57 + 3.36, 1450\} = 1502.05 \\
Resst_{43} &= \max\{Resst_4 + T_{43} + s_4, a_3\} = \max\{1485.12 + 1.490 + 3.36, 1450\} = 1489.97
\end{aligned}$$

$$f(\{4\}, j) = f(\emptyset, 4) + dp_D \cdot (Resst_{4j} - Resst_4) + dp_T \cdot D_{4j} + p \cdot \max(Resst_j - b_j, 0)$$

For $j = 1, 2, 3$, we have

$$\begin{aligned}
f(\{4\}, 1) &= 11.83 + 0.24(1511.43 - 1487.95) + 0.5(5.16) + 0 = 20.66 \\
f(\{4\}, 2) &= 11.83 + 0.24(1502.05 - 1487.95) + 0.5(2.63) + 0 = 17.30 \\
f(\{4\}, 3) &= 11.83 + 0.24(1489.97 - 1487.95) + 0.5(2.82) + 0 = 13.21
\end{aligned}$$

Iteration 2.

For $|S| = 2$, calculate the real earliest service start time of terminal stop j .

If $S = \{1, 2\}$ and the state is $(\{1, 2\}, 3)$, then the initial service start time at node 3 depends on the order of the nodes on this partial path. If the path is $(0 \rightarrow 1 \rightarrow 2 \rightarrow 3)$, then the initial service start time at node 3 is

$$Resst_{23} = \max\{Resst_{12} + T_{23} + s_2, a_3\} = \max\{1497.10 + 12.95 + 3.36, 1450\} = 1513.41.$$

$Resst_{12}$ is the real service start time of node 2 if the predecessor is node 1, and

$Resst_{12}$ is already calculated in iteration 1. Similarly, if the path is $(0 \rightarrow 2 \rightarrow 1 \rightarrow 3)$, then the initial service start time at node 3 is

$$Resst_{13} = \max\{Resst_{21} + T_{13} + s_1, a_3\} = \max\{1517.10 + 24.43 + 4.06, 1450\} = 1545.59.$$

The cost function $f(\{1, 2\}, 3)$ is

$$\begin{aligned}
& \min \begin{cases} f(\{1\}, 2) + dp_T(Resst_{23} - Resst_{12}) + dp_D(D_{23}) + p \cdot \max(Resst_{23} - b_3, 0) \\ f(\{2\}, 1) + dp_T(Resst_{13} - Resst_{21}) + dp_D(D_{13}) + p \cdot \max(Resst_{13} - b_3, 0) \end{cases} \\
&= \min \begin{cases} 15.27 + 0.24(1513.41 - 1497.10) + 0.5(2.63) + 100 \cdot \max(1513.41 - 1750, 0) \\ 22.14 + 0.24(1545.59 - 1517.10) + 0.5(5.48) + 100 \cdot \max(1545.59 - 1750, 0) \end{cases} \\
&= \min\{20.50, 31.72\} = 20.05
\end{aligned}$$

The cost associated with the state $(\{1, 2\}, 3)$ is $f(\{1, 2\}, 3) = 20.05$. The order is $(0 \rightarrow 1 \rightarrow 2 \rightarrow 3)$, and the initial service start time of node 3 is $Resst_{23} = 1513.41$.

Similarly, we can calculate the cost for state $(\{1, 2\}, 4)$. The cost associated with the state $(\{1, 2\}, 4)$ is $f(\{1, 2\}, 4) = 20.74$. The order is $(0 \rightarrow 1 \rightarrow 2 \rightarrow 4)$, and the initial service start time of node 4 is $Resst_{24} = 1514.03$.

If $S = \{1, 3\}$ and the state is $(\{1, 3\}, 2)$, then the associated cost is $f(\{1, 3\}, 2) = 20.51$. The order is $(0 \rightarrow 1 \rightarrow 3 \rightarrow 2)$. We keep the initial service start time of node 2 which is $Resst_{32} = 1512.76$.

If $S = \{1, 3\}$ and the state is $(\{1, 3\}, 4)$, then the associated cost is $f(\{1, 3\}, 4) = 16.67$. The order is $(0 \rightarrow 1 \rightarrow 3 \rightarrow 4)$. We keep the initial service start time of node 4 which is $Resst_{34} = 1510.30$.

If $S = \{1, 4\}$ and the state is $(\{1, 4\}, 2)$, then the associated cost is $f(\{1, 4\}, 2) = 20.00$. The order is $(0 \rightarrow 1 \rightarrow 4 \rightarrow 2)$. We keep the initial service start time of node 2 which is $Resst_{42} = 1511.19$.

If $S = \{1, 4\}$ and the state is $(\{1, 4\}, 3)$, then the associated cost is $f(\{1, 4\}, 3) = 16.67$. The order is $(0 \rightarrow 1 \rightarrow 4 \rightarrow 3)$. We keep the initial service start time of node 3 which is $Resst_{43} = 1499.11$.

If $S = \{2, 3\}$ and the state is $(\{2, 3\}, 1)$, then the associated cost is $f(\{2, 3\}, 1) = 27.40$. Both the order $(0 \rightarrow 2 \rightarrow 3 \rightarrow 1)$ and the order $(0 \rightarrow 3 \rightarrow 2 \rightarrow 1)$ have the same cost. We keep the initial service start time of node 1 for both orders, which are $Resst_{31} = 1532.76$ and $Resst_{21} = 1532.76$.

If $S = \{2, 3\}$ and the state is $(\{2, 3\}, 4)$, then the associated cost is $f(\{2, 3\}, 4) = 19.36$. The order is $(0 \rightarrow 2 \rightarrow 3 \rightarrow 4)$. We keep the initial service start time of node 4 which is $Resst_{34} = 1509.82$.

If $S = \{2, 4\}$ and the state is $(\{2, 4\}, 1)$, then the associated cost is $f(\{2, 4\}, 1) = 26.87$. Both the order $(0 \rightarrow 2 \rightarrow 4 \rightarrow 1)$ and the order $(0 \rightarrow 4 \rightarrow 2 \rightarrow 1)$ have the same cost. We keep the initial service start time of node 1 for both orders which are $Resst_{41} = 1531.19$ and $Resst_{21} = 1531.19$.

Table D.3: Initialization

State	S	Terminating Node	$Resst$ of Terminating node	Cost	Order
$K_0(1)$	\emptyset	1	$Resst_1 = 1467.26$	5.53	$0 \rightarrow 1$
$K_0(2)$	\emptyset	2	$Resst_2 = 1487.96$	12.57	$0 \rightarrow 2$
$K_0(3)$	\emptyset	3	$Resst_3 = 1486.61$	12.41	$0 \rightarrow 3$
$K_0(4)$	\emptyset	4	$Resst_4 = 1485.12$	11.83	$0 \rightarrow 4$

Table D.4: Iteration 1

State	S	Terminating Node	$Resst$ of Terminating node	Cost	Order
$K_1(1)$	1	2	$Resst_{12} = 1497.10$	15.27	$0 \rightarrow 1 \rightarrow 2$
$K_1(2)$	1	3	$Resst_{13} = 1495.75$	15.11	$0 \rightarrow 1 \rightarrow 3$
$K_1(3)$	1	4	$Resst_{14} = 1494.26$	14.53	$0 \rightarrow 1 \rightarrow 4$
$K_1(4)$	2	1	$Resst_{21} = 1517.10$	22.14	$0 \rightarrow 2 \rightarrow 1$
$K_1(5)$	2	3	$Resst_{23} = 1504.27$	17.80	$0 \rightarrow 2 \rightarrow 3$
$K_1(6)$	2	4	$Resst_{24} = 1504.89$	18.04	$0 \rightarrow 2 \rightarrow 4$
$K_1(7)$	3	1	$Resst_{31} = 1515.10$	21.99	$0 \rightarrow 3 \rightarrow 1$
$K_1(8)$	3	2	$Resst_{32} = 1503.62$	17.81	$0 \rightarrow 3 \rightarrow 2$
$K_1(9)$	3	4	$Resst_{34} = 1492.16$	13.96	$0 \rightarrow 3 \rightarrow 4$
$K_1(10)$	4	1	$Resst_{41} = 1511.43$	20.66	$0 \rightarrow 4 \rightarrow 1$
$K_1(11)$	4	2	$Resst_{42} = 1502.05$	17.30	$0 \rightarrow 4 \rightarrow 2$
$K_1(12)$	4	3	$Resst_{43} = 1489.97$	13.21	$0 \rightarrow 4 \rightarrow 3$

If $S = \{2, 4\}$ and the state is $(\{2, 4\}, 3)$, then the associated cost is $f(\{2, 4\}, 3) = 19.43$. The order is $(0 \rightarrow 2 \rightarrow 4 \rightarrow 3)$. We keep the initial service start time of node 3 which is $Resst_{43} = 1509.73$.

If $S = \{3, 4\}$ and the state is $(\{3, 4\}, 1)$, then the associated cost is $f(\{3, 4\}, 1) = 22.79$. Both the order $(0 \rightarrow 3 \rightarrow 4 \rightarrow 1)$ and the order $(0 \rightarrow 4 \rightarrow 3 \rightarrow 1)$ have the same cost. We keep the initial service start time of node 1 for both orders which are $Resst_{41} = 1518.46$ and $Resst_{31} = 1518.46$.

If $S = \{3, 4\}$ and the state is $(\{3, 4\}, 2)$, then the associated cost is $f(\{3, 4\}, 2) = 18.62$. The order is $(0 \rightarrow 4 \rightarrow 3 \rightarrow 2)$. We keep the initial service start time of node 2 which is $Resst_{43} = 1506.98$.

In Tables D.3, D.4, D.5, and D.6, we give the results of each iteration including the state, associated cost, $Resst$, and the order of the partial path.

From Table D.6, the cost associated with the state $(\{1, 2, 3\}, 4)$ is 22.06, and

Table D.5: Iteration 2

State	S	Terminating Node	$Resst$ of Terminating node	Cost	Order
$K_2(1)$	1 2	3	$Resst_{23} = 1513.41$	20.50	$0 \rightarrow 1 \rightarrow 2 \rightarrow 3$
$K_2(2)$	1 2	4	$Resst_{24} = 1514.03$	20.74	$0 \rightarrow 1 \rightarrow 2 \rightarrow 4$
$K_2(3)$	1 3	2	$Resst_{32} = 1512.76$	20.51	$0 \rightarrow 1 \rightarrow 3 \rightarrow 2$
$K_2(4)$	1 3	4	$Resst_{35} = 1513.41$	16.67	$0 \rightarrow 1 \rightarrow 3 \rightarrow 5$
$K_2(5)$	1 4	2	$Resst_{42} = 1511.19$	20.00	$0 \rightarrow 1 \rightarrow 4 \rightarrow 2$
$K_2(6)$	1 4	3	$Resst_{43} = 1499.11$	15.92	$0 \rightarrow 1 \rightarrow 4 \rightarrow 3$
$K_2(7)$	2 3	1	$Resst_{31} = 1532.76$ $Resst_{21} = 1532.76$	27.38 27.38	$0 \rightarrow 2 \rightarrow 3 \rightarrow 1$ $0 \rightarrow 3 \rightarrow 2 \rightarrow 1$
$K_2(8)$	2 3	4	$Resst_{34} = 1509.82$	19.36	$0 \rightarrow 2 \rightarrow 3 \rightarrow 4$
$K_2(9)$	2 4	1	$Resst_{41} = 1531.19$ $Resst_{21} = 1531.19$	26.87 26.87	$0 \rightarrow 2 \rightarrow 4 \rightarrow 1$ $0 \rightarrow 4 \rightarrow 2 \rightarrow 1$
$K_2(10)$	2 4	3	$Resst_{43} = 1509.73$	26.87	$0 \rightarrow 2 \rightarrow 4 \rightarrow 3$
$K_2(11)$	3 4	1	$Resst_{41} = 1518.46$ $Resst_{31} = 1518.46$	22.79 22.79	$0 \rightarrow 3 \rightarrow 4 \rightarrow 1$ $0 \rightarrow 4 \rightarrow 3 \rightarrow 1$
$K_2(12)$	3 4	2	$Resst_{32} = 1506.98$	18.62	$0 \rightarrow 4 \rightarrow 3 \rightarrow 2$

Table D.6: Iteration 3

State	Set S	Terminating Node	$Resst$ of Terminating node	Cost	Order
$K_3(1)$	1 2 3	4	$Resst_{34} = 1518.96$	22.06	$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$
$K_3(2)$	1 2 4	3	$Resst_{43} = 1518.87$	22.13	$0 \rightarrow 1 \rightarrow 2 \rightarrow 4 \rightarrow 3$
$K_3(3)$	1 3 4	2	$Resst_{32} = 1516.12$	21.32	$0 \rightarrow 1 \rightarrow 4 \rightarrow 3 \rightarrow 2$
$K_3(4)$	2 3 4	1	$Resst_{21} = 1536.12$	28.19	$0 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$

the best order for the state $(\{1, 2, 3\}, 4)$ is $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$. Therefore the $Resst$ of terminating node 4 is $Resst_{34} = 1518.96$. The immediate predecessor of node 4 is node 3.

The best order for the state $(\{1, 2, 4\}, 3)$ is $0 \rightarrow 1 \rightarrow 2 \rightarrow 4 \rightarrow 3$, and therefore the $Resst$ of terminating node 3 is $Resst_{43} = 1518.87$. The immediate predecessor of node 3 is node 4.

The best order for the state $(\{1, 3, 4\}, 2)$ is $0 \rightarrow 1 \rightarrow 4 \rightarrow 3 \rightarrow 2$, and therefore the $Resst$ of terminating node 2 is $Resst_{32} = 1516.12$. The immediate predecessor of node 2 is node 3.

The best order for the state $(\{2, 3, 4\}, 1)$ is $0 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$, and therefore the $Resst$ of terminating node 1 is $Resst_{41} = 1536.12$. The immediate predecessor of node 3 is node 4.

In the final iteration, we first calculate the *Resst* of ending depot 5, considering all four possible orderings in Iteration 3 given in Table D.6.

For $S = \{1, 2, 3, 4\}$, we have four possible orderings.

(1) If node 4 is the immediate predecessor of end depot, then from $K_3(1)$ in Table D.6, $Resst_{34} = 1518.96$. Therefore, we have

$$Resst_{4,5} = \max\{Resst_{34} + T_{4,5} + s_4, a_5\} = \max\{1518.96 + 13.65 + 3.36, 1450\} = 1535.97.$$

(2) If node 3 is the immediate predecessor of end depot, then from $K_3(2)$ in Table D.6, $Resst_{43} = 1518.87$. Therefore, we have

$$Resst_{3,5} = \max\{Resst_{43} + T_{3,5} + s_3, a_5\} = \max\{1518.87 + 15.14 + 4.06, 1450\} = 1538.07.$$

(3) If node 2 is the immediate predecessor of end depot, then from $K_3(3)$ in Table D.6, $Resst_{32} = 1518.87$. Therefore, we have

$$Resst_{2,5} = \max\{Resst_{32} + T_{2,5} + s_2, a_5\} = \max\{1516.12 + 18.94 + 3.36, 1450\} = 1538.43.$$

(4) If node 1 is the immediate predecessor of end depot, then from $K_3(4)$ in Table D.6, $Resst_{32} = 1518.87$. Therefore, we have

$$Resst_{1,5} = \max\{Resst_{41} + T_{1,5} + s_1, a_5\} = \max\{1536.12 + 26.19 + 4.06, 1450\} = 1566.37.$$

For $S_1 = \{1, 2, 3\}$, $S_2 = \{1, 2, 4\}$, $S_3 = \{1, 3, 4\}$, and $S_4 = \{2, 3, 4\}$, the final cost is then

$$\begin{aligned} & \min \begin{cases} f(S_1, 4) + dp_T(Resst_{4,5} - Resst_{34}) + dp_D(D_{4,5}) + p \cdot \max(Resst_{4,5} - b_5, 0) \\ f(S_2, 3) + dp_T(Resst_{3,5} - Resst_{43}) + dp_D(D_{4,5}) + p \cdot \max(Resst_{3,5} - b_5, 0) \\ f(S_3, 2) + dp_T(Resst_{2,5} - Resst_{32}) + dp_D(D_{4,5}) + p \cdot \max(Resst_{2,5} - b_5, 0) \\ f(S_4, 1) + dp_T(Resst_{1,5} - Resst_{41}) + dp_D(D_{4,5}) + p \cdot \max(Resst_{1,5} - b_5, 0) \end{cases} \\ &= \min \begin{cases} 22.06 + 0.24(1535.97 - 1518.96) + 0.5(4.03) + 100 \cdot \max(1535.97 - 1750, 0) \\ 22.13 + 0.24(1538.07 - 1518.87) + 0.5(4.47) + 100 \cdot \max(1538.07 - 1750, 0) \\ 21.32 + 0.24(1538.43 - 1516.12) + 0.5(4.64) + 100 \cdot \max(1538.43 - 1750, 0) \\ 28.19 + 0.24(1566.37 - 1536.12) + 0.5(6.78) + 100 \cdot \max(1566.37 - 1750, 0) \end{cases} \\ &= \min\{29.01, 29.83, 29.84, 39.69\} = 29.01 \end{aligned}$$

We then trace back the best order with cost equal to 29.01, which is given in

Figure D.2: Best Path

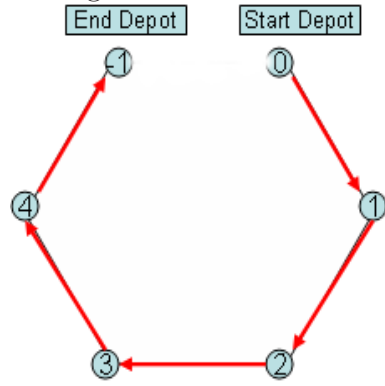


Fig D.2. The order is $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$.

Appendix E

Data for Four Two-stage Transportation Problems

Table E.1: Three plants, four DCs, and five customers ($3 \times 4 \times 5$)

T_{ij}	1	2	3	4
1	11	8	18	9
2	13	11	16	2
3	3	6	14	18

C_{ij}	1	2	3	4	5
1	12	5	6	2	8
2	9	15	13	8	1
3	14	3	11	13	9
4	4	9	13	15	4

P_1	P_2	P_3
27	47	26

(a) Distance matrix T (from plants to DCs) (b) Distance matrix C (from DCs to customers) (c) Plant capacity

DC_1	DC_2	DC_3	DC_4
36	23	22	19

C_1	C_2	C_3	C_4	C_5
17	21	18	18	26

(d) DC capacity (e) Customer demand

Table E.2: $4 \times 5 \times 10$

T_{ij}	1	2	3	4	5
1	18	6	19	4	13
2	9	6	15	10	14
3	10	14	18	2	11
4	16	15	19	19	13

C_{ij}	1	2	3	4	5	6	7	8	9	10
1	8	10	10	2	8	7	14	13	9	3
2	11	14	5	6	2	6	6	3	6	10
3	8	15	3	8	10	5	10	7	12	6
4	3	3	1	11	12	3	1	8	3	11
5	14	9	13	15	11	8	11	3	1	4

(a) Distance matrix T (from plants to DCs) (b) Distance matrix C (from DCs to customers)

P_1	P_2	P_3	P_4
48	79	37	36

DC_1	DC_2	DC_3	DC_4	DC_5
25	40	25	61	49

(c) Plant capacity (d) DC capacity

C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
23	24	20	14	13	22	26	22	11	25

(e) Customer demand

Table E.3: $4 \times 5 \times 15$

T_{ij}	1	2	3	4	5
1	3	5	2	2	13
2	11	5	3	18	12
3	10	13	11	10	14
4	19	16	18	5	4

(a) Distance matrix T (from plants to DCs)

C_{ij}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	4	2	15	4	10	11	6	5	8	6	1	4	6	8	14
2	12	14	6	14	10	13	15	7	2	10	8	13	10	4	5
3	15	1	13	12	7	3	1	2	2	14	4	14	6	9	12
4	14	8	6	13	3	2	9	5	7	5	13	14	3	5	14
5	5	2	1	13	15	2	2	14	13	6	13	3	5	7	2

(b) Distance matrix C (from DCs to customers)

P_1	P_2	P_3	P_4
83	45	77	95

(c) Plant capacity

DC_1	DC_2	DC_3	DC_4	DC_5
46	66	84	32	72

(d) DC capacity

C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}
13	25	13	13	22	19	24	27	11	17	13	19	28	24	32

(e) Customer demand

Table E.4: $8 \times 10 \times 20$

T_{ij}	1	2	3	4	5	6	7	8	9	10
1	15	4	6	4	18	14	3	13	7	3
2	7	8	11	16	14	8	4	3	8	15
3	3	2	9	4	11	19	10	9	6	9
4	18	6	15	17	5	4	3	14	16	17
5	8	19	16	18	2	11	2	3	3	9
6	10	6	16	2	9	6	3	17	16	3
7	14	13	8	4	9	8	13	16	5	4
8	9	19	13	8	7	19	10	8	7	19

(a) Distance matrix T (from plants to DCs)

C_{ij}	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	6	14	6	4	11	6	9	8	15	6	13	11	3	2	2	2	12	2	2	12
2	7	4	13	4	10	8	14	3	4	3	7	12	7	5	5	9	10	5	1	14
3	7	9	9	6	15	7	10	4	7	13	13	3	6	14	7	13	13	7	9	15
4	2	8	5	11	3	8	11	9	13	7	13	10	11	9	2	8	8	15	9	9
5	10	15	1	4	5	11	13	3	3	5	11	14	10	8	4	9	6	9	5	3
6	10	15	6	2	8	11	15	1	5	3	15	7	6	14	9	6	4	5	14	12
7	11	4	4	11	6	6	8	9	8	5	3	2	8	4	4	14	4	3	15	7
8	9	10	12	9	8	14	5	13	6	9	11	14	7	3	9	15	10	13	3	6
9	2	2	2	8	15	5	5	10	13	11	8	15	14	14	8	15	3	14	7	9
10	13	2	2	11	13	6	4	3	4	5	3	14	15	5	8	13	7	10	2	4

(b) Distance matrix C (from DCs to customers)

P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
52	63	57	58	25	44	68	33

(c) Plant capacity

DC_1	DC_2	DC_3	DC_4	DC_5	DC_6	DC_7	DC_8	DC_9	DC_{10}
57	21	52	45	23	48	44	22	57	31

(d) DC capacity

C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}	C_{20}
16	26	11	14	12	22	30	15	27	25	22	18	24	13	19	18	21	12	21	34

(e) Customer demand

Bibliography

- [1] M. Almeida and F. Carvalho. Exact Disclosure Prevention in Two-dimensional Statistical Tables. *Computers and Operations Research*, Vol.32, Issue 11, 2919-2936, 2005.
- [2] A. Amiri, Designing a Distribution Network in a Supply Chain System: Formulation and Efficient Solution Procedure. *European Journal of Operational Research*, Vol.171, 567C576, 2006.
- [3] E. Baker. An Exam Algorithm for the Time-Constrained Traveling Salesman Problem. *Operations Research*, Vol.31, No.5:938-945, 1983.
- [4] E. Balas and N. Simonetti. Linear Time Dynamic-Programming Algorithms for New Classes of Restricted TSPs: A Computational Study. *INFORMS journal on Computing*, Vol.13, No.1:56-75, 2001.
- [5] J. Castro. A Fast Network Flows Heuristic for Cell Suppression in Positive Tables. *Privacy in Statistical Databases*, Vol.3050, 136-148, 2004.
- [6] I.M. Chao. Algorithms and Solutions to Multi-level Vehicle Routing Problems, Ph.D. Dissertation, Applied Mathematics Program, University of Maryland, College Park, MD, 1993.
- [7] I.M. Chao, B.L. Golden, and E.A. Wasil. A Fast and Effective Heuristic for the Orienteering Problem. *European Journal of Operational Research*, Vol.88, Issue 3, 475-489, 1996.
- [8] S. Chauhan and J. Proth. The Concave Cost Supply Problem in Production, Manufacturing and Logistics. *European Journal of Operational Research*, Vol.148, 374-383, 2003.
- [9] S. Chauhan, A. Ereemeev, A. Romanova, V. Servakh, and G. Woeginger. Approximation of the Supply Scheduling Problem. *Operations Research Letters*, Vol.33, 249-254, 2005.
- [10] L. Cox. Suppression Methodology and Statistical Disclosure Control. *Journal of the American Statistical Association* 75, 377-385, 1980.
- [11] L. Cox. Linear Sensitivity Measures in Statistical Disclosure Control. *Journal of Statistical Planning and Inference*, Vol.5, 153-164, 1981.

- [12] L. Cox. Network Models for Complementary Cell Suppression. *Journal of the American Statistical Association* 90, 1453-1462, 1995.
- [13] L. Cox. Disclosure Risk for Tabular Economic Data. Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies (P. Doyle, J. Lane, J. Theeuwes and L. Zayatz, eds.), Chapter 8, New York: Elsevier, 167-183, 2001.
- [14] L. Cox and J. Kelly. Controlled Tabular Adjustment: An Empirical Study of Heuristic and Optimal Methods, 2004(submitted).
- [15] L. Cox. Confidentiality in Tables Viewed from An Algebraic Perspective, DIMASC/PPORTIA Workshop on Privacy-Preserving Data Mining, 2004.
- [16] L. Cox, J. Kelly and R. Patil. Balancing Quality and Confidentiality for Multivariate Tabular Data. *Privacy in Statistical Databases*, Vol. 050, 87-98, 2004.
- [17] L. Cox, J. Kelly and R. Patil. Computational Aspects of Controlled Tabular Adjustment: Algorithm and Analysis, *The Next Wave in Computing, Optimization, and Decision Technologies*, Vol. 9, 45-59, Springer US, 2006.
- [18] R. Dandekar and L. Cox. Synthetic Tabular Data-An Alternative to Complementary Cell Suppression, unpublished manuscript, 2002.
- [19] Dash Optimization, <http://www.dashoptimization.com>, version 2005A 16.01.
- [20] K.A. De Jong. An Analysis of the Behavior of a Class of Genetic Adaptive Systems. Doctoral dissertation, University of Michigan, Ann Arbor, 1975.
- [21] Y. Dumas, J. Desrosiers, E. Gelinas, and M. Solomon. An Optimal Algorithm for the Traveling Salesman Problem with Time Windows. *Operations Research*, Vol.43, No.2:367-371, 1995.
- [22] M. Fischetti, and J. Salazar-Gonzalez. Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints. *Journal of the American Statistical Association*, 95:916-928, 2000.
- [23] F. Focacci, A. Lodi, and M. Milano. A Hybrid Exam Algorithm for the TSPTW. *INFORMS journal on Computing*, Vol.14, No.4:403-417, 2002.
- [24] Z. Geem, C. Tseng, and Y. Park. Harmony Search for Generalized Orienteering Problem: Best Touring in China. *Lecture Notes in Computer Science*, Vol.3612, 741-750, 2005.

- [25] M. Gendreau, A. Hertz, and G. Laporte. New Insertion and Postoptimization Procedures for the Traveling Salesman Problem. *Operations Research*, Vol.40, No.6:1086-1094, 1992.
- [26] A. Geoffrion, G. Graves. Multicommodity distribution system design by benders decomposition. *Management Science*, Vol.20:822C844, 1974.
- [27] M. Gendreau, A. Hertz, G. Laporte, and M. Stan. A Generalized Insertion Heuristic for the Traveling Salesman Problem with Time Windows. *Operations Research*, Vol.43, No.3:367-371, 1998.
- [28] M. Gendreau, G. Laporte, and F. Semet. A Tabu Search Heuristic for the Undirected Selective Traveling Salesman Problem. *European Journal of Operational Research*, 106:539-545, 1998.
- [29] M. Gen, F. Altıparmak, and L. Lin. A Genetic Algorithm For Two-stage Transportation Problem Using Priority-based Encoding. *OR Spectrum*, Vol.28, No.3, 337-354, 2006.
- [30] K. Hindi, T. Basta, K. Pienkosz. Efficient Solution of a Multi-commodity, Two-stage Distribution Problem with Constraints on Assignment of Customers to Distribution Centers. *International Transactions in Operational Research*, 5(6):519C527, 1998.
- [31] F. Hitchcock. The Distribution of a Product from Several Sources to Numerous Localities. *Journal of Mathematical Physics*, Vol 20:24C230.
- [32] J. Kelly, B. Golden, and A. Assad. Cell Suppression: Disclosure Protection for Sensitive Tabular Data. *Networks* Vol.22, 397-417, 1992.
- [33] A. Langevin, M. Desrochers, J. Desrosiersand, S. G elinas, and F. Soumis. A Two-Commodity Flow Formulation for the Traveling Salesman and the Makespan Problems With Time Windows. *Networks*, 23, 631-640, 1993.
- [34] G. Laporte and I. Rodr iguez-Mart ın. Locating a Cycle in a Transportation or a Telecommunications Network. *Networks*, 50:92-108, 2007.
- [35] Y.C. Liang and A.E. Smith. An Ant Colony Approach to the Orienteering Problem. *Journal of the Chinese Institute of Industrial Engineers*, Vol.23, 403-414, 2006.
- [36] Y. Li, M. Gen, and K. Ida. Improved Genetic Algorithm for Solving Multi-objective Solid Transportation Problem with Fuzzy Number. *Computers and Industrial Engineering*, Vol. 33(3), 589-592(4), 1997.

- [37] Z. Michalewicz, G. Vignaux, and M. Hobbs. A Non-standard Genetic Algorithm for the Nonlinear Transportation Problem. *ORSA Journal on Computing*, 3(4):307C316, 1991.
- [38] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, 1996.
- [39] A. Mingozzi, L. Bianco, and S. Ricciardelli. Dynamic Programming Strategies for the Traveling Salesman Problem with Time Window and Precedence Constraints. *Operations Research*, Vol.45, No.3:365-377, 1997.
- [40] J. Potvin, T. Kervahut, B. Garciaa, and J. Rousseau. The Vehicle Routing Problem with Time Windows-Part I: Tabu Search. *INFORMS J. Computing* 8, 158-164, 1996.
- [41] H. Psaraftis. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, Vol.14, 130-154.
- [42] R. Ramesh and K. Brown. An efficient four-phase heuristic for the generalized orienteering problem *Computers and Operations Research*, Vol.18, Issue 2, 151-165, 1991.
- [43] E. Silver, D. Pyke, and R. Peterson. *Inventory Management and Production Planning and Scheduling*. Jonh Wiley & Sons, New York, 1998.
- [44] D. Smith. *Dynamic Programming: A Practical Introduction*, Ellis Horwood, Chichester, 1991.
- [45] M. Solomon. Algorithms for the Vehicle Routing and Scheduling Problem with Time Windows Constraints. *Operations Research* Vol.35, 254-265, 1987.
- [46] A. Syarif and M. Gen. Double Spanning Tree-based Genetic Algorithm for Two Stage Transportation Problem. *Int J Knowl-Based Intell Eng Syst* 7(4), 2003.
- [47] M. Tasgetiren. A Genetic Algorithm with an Adaptive Penalty Function for the Orienteering Problem. *Journal of Economic & Social Research*, Vol.4:2, 1-26, 2002.
- [48] B. Tilanus. Introduction to information system in logistics and transportation. *Information systems in logistics and transportation*. Elsevier, Amsterdam, pp 7C16.

- [49] T. Tsiligides. Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, Vol.35, 797-809, 1984.
- [50] Q. Wang, X. Sun, B.L. Golden, and J. Jia. Using Artificial Neural Networks to Solve the Orienteering Problem. *Annals of Operations Research*, Vol.61, Issue1-4, 111-120, 1995.
- [51] Q. Wang, X. Sun, and B.L. Golden. Using Artificial Neural Networks to Solve Generalized Orienteering Problems. in *Intelligent Engineering Systems Through Artificial Neural Networks: Volume 6* (C. Dagli, M. Akay, C. Chen, B. Fernández, and J. Ghosh, eds.). ASME Press, New York, 1063-1068, 1996.
- [52] D. Whitley, T. Starkweather, and D. Fuquay. Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator. in *Proceedings of the Third International Conference on Genetic Algorithms and their Applications*. (J.D. Shaeffer, ed.). Morgan Kaufmann Publishers, Palo Alto, CA, 133-140, 1989.
- [53] L. Willenborg and T. de Waal. Statistical Disclosure Control in Practice (Lecture Notes in Statistics 11), New York: Springer, 1996.
- [54] G. Zhou, M. Gen. An Effective Genetic Algorithm Approach to the Quadratic Minimum Spanning Tree Problem. *Computers and Operations Research*, Vol.25, No.3, 229-247, 1998.