PROBABILISTIC ANALYSIS AND RESULTS
OF COMBINATORIAL PROBLEMS
WITH MILITARY APPLICATIONS

By

DON A. GRUNDEL

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2004
Copyright 2004
by
Don A. Grundel
I dedicate this work to Bonnie, Andrew and Erin.
ACKNOWLEDGMENTS

I wish to express my heartfelt thanks to Professor Panos Pardalos for his guidance and support. His extraordinary energetic personality inspires all those around him. What I appreciate most about Professor Pardalos is he sets high goals for himself and his students and then tirelessly strives to reach those goals.

I am grateful to the United States Air Force for its financial support and for allowing me to pursue my lifelong goal. Within the Air Force, I owe a debt of gratitude to Dr. David Jeffcoat for his counsel and assistance throughout my PhD efforts.

My appreciation also goes to my committee members Stan Uryasev, Joseph Geunes, and William Hager for their time and thoughtful guidance. I would like to thank my collaborators Anthony Okafor, Carlos Oliveira, Pavlo Krakhmal, and Lewis Pasiliao.

Finally, to my family, Bonnie, Andrew and Erin, who have been extremely supportive – I could not have completed this work without their love and understanding.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>viii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>xii</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Probabilistic Analysis of Combinatorial Problems</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Main Contributions and Organization of the Dissertation</td>
<td>3</td>
</tr>
<tr>
<td>2 SURVEY OF THE MULTIDIMENSIONAL ASSIGNMENT PROBLEM</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Formulations</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Complexity</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Applications</td>
<td>8</td>
</tr>
<tr>
<td>2.3.1 Weapon Target Assignment Problem</td>
<td>8</td>
</tr>
<tr>
<td>2.3.2 Considering Weapon Costs in the Weapon Target Assignment Problem</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Summary</td>
<td>12</td>
</tr>
<tr>
<td>3 CHARACTERISTICS OF THE MEAN OPTIMAL SOLUTION TO THE MAP</td>
<td>13</td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>13</td>
</tr>
<tr>
<td>3.1.1 Basic Definitions and Results</td>
<td>14</td>
</tr>
<tr>
<td>3.1.2 Motivation</td>
<td>15</td>
</tr>
<tr>
<td>3.1.3 Asymptotic Studies and Results</td>
<td>16</td>
</tr>
<tr>
<td>3.1.4 Chapter Organization</td>
<td>19</td>
</tr>
<tr>
<td>3.2 Mean Optimal Costs for a Special Case of the MAP</td>
<td>20</td>
</tr>
<tr>
<td>3.3 Branch and Bound Algorithm</td>
<td>23</td>
</tr>
<tr>
<td>3.3.1 Procedure</td>
<td>24</td>
</tr>
<tr>
<td>3.3.2 Sorting</td>
<td>27</td>
</tr>
<tr>
<td>3.3.3 Local Search</td>
<td>27</td>
</tr>
<tr>
<td>3.4 Computational Experiments</td>
<td>28</td>
</tr>
<tr>
<td>3.4.1 Experimental Procedures</td>
<td>28</td>
</tr>
<tr>
<td>3.4.2 Mean Optimal Solution Costs</td>
<td>29</td>
</tr>
<tr>
<td>3.4.3 Curve Fitting</td>
<td>33</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>3.5</td>
<td>Algorithm Improvement Using Numerical Models</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Improvement of B&amp;B</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Comparison of B&amp;B Implementations</td>
</tr>
<tr>
<td>3.6</td>
<td>Remarks</td>
</tr>
<tr>
<td>4</td>
<td>PROOFS OF ASYMPTOTIC CHARACTERISTICS OF THE MAP</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>4.2</td>
<td>Greedy Algorithms</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Greedy Algorithm 1</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Greedy Algorithm 2</td>
</tr>
<tr>
<td>4.3</td>
<td>Mean Optimal Costs of Exponentially and Uniformly Distributed Random MAPs</td>
</tr>
<tr>
<td>4.4</td>
<td>Mean Optimal Costs of Normal-Distributed Random MAPs</td>
</tr>
<tr>
<td>4.5</td>
<td>Remarks on Further Research</td>
</tr>
<tr>
<td>5</td>
<td>PROBABILISTIC APPROACH TO SOLVING THE MULTISENSOR MULTITARGET TRACKING PROBLEM</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>5.2</td>
<td>Data Association Formulated as an MAP</td>
</tr>
<tr>
<td>5.3</td>
<td>Minimum Subset of Cost Coefficients</td>
</tr>
<tr>
<td>5.4</td>
<td>GRASP for a Sparse MAP</td>
</tr>
<tr>
<td>5.4.1</td>
<td>GRASP Complexity</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Search Tree Data Structure</td>
</tr>
<tr>
<td>5.4.3</td>
<td>GRASP vs Sparse GRASP</td>
</tr>
<tr>
<td>5.5</td>
<td>Conclusion</td>
</tr>
<tr>
<td>6</td>
<td>EXPECTED NUMBER OF LOCAL MINIMA FOR THE MAP</td>
</tr>
<tr>
<td>6.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>6.2</td>
<td>Some Characteristics of Local Neighborhoods</td>
</tr>
<tr>
<td>6.3</td>
<td>Experimentally Determined Number of Local Minima</td>
</tr>
<tr>
<td>6.4</td>
<td>Expected Number of Local Minima for n = 2</td>
</tr>
<tr>
<td>6.5</td>
<td>Expected Number of Local Minima for n ≥ 3</td>
</tr>
<tr>
<td>6.6</td>
<td>Number of Local Minima Effects on Solution Algorithms</td>
</tr>
<tr>
<td>6.6.1</td>
<td>Random Local Search</td>
</tr>
<tr>
<td>6.6.2</td>
<td>GRASP</td>
</tr>
<tr>
<td>6.6.3</td>
<td>Simulated Annealing</td>
</tr>
<tr>
<td>6.6.4</td>
<td>Results</td>
</tr>
<tr>
<td>6.7</td>
<td>Conclusions</td>
</tr>
<tr>
<td>7</td>
<td>MAP TEST PROBLEM GENERATOR</td>
</tr>
<tr>
<td>7.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>7.1.1</td>
<td>Test Problem Generators</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Test Problem Libraries</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3–1</td>
<td>Mean optimal solution costs obtained from the closed form equation for MAPs of sizes $n = 2, 3 \leq d \leq 10$ and with cost coefficients that are independent exponentially distributed with mean one.</td>
<td>23</td>
</tr>
<tr>
<td>3–2</td>
<td>Number of runs for each experiment with uniform or exponential assignment costs.</td>
<td>29</td>
</tr>
<tr>
<td>3–3</td>
<td>Number of runs for each experiment standard normal assignment costs.</td>
<td>30</td>
</tr>
<tr>
<td>3–4</td>
<td>Mean optimal costs for different sizes of MAPs with independent assignment costs that are uniform in $[0, 1]$.</td>
<td>31</td>
</tr>
<tr>
<td>3–5</td>
<td>Mean optimal costs for different sizes of MAPs with independent assignment costs that are exponential with mean 1.</td>
<td>31</td>
</tr>
<tr>
<td>3–6</td>
<td>Mean optimal costs for different sizes of MAPs with independent assignment costs that are standard normal.</td>
<td>31</td>
</tr>
<tr>
<td>3–7</td>
<td>Curve fitting results for fitting the form $(An + B)^C$ to the mean optimal costs for MAPs with uniform assignment costs.</td>
<td>35</td>
</tr>
<tr>
<td>3–8</td>
<td>Curve fitting results for fitting the form $(An + B)^C$ to the mean optimal costs for MAPs with exponential assignment costs.</td>
<td>35</td>
</tr>
<tr>
<td>3–9</td>
<td>Curve fitting results for fitting the form $A(n + B)^C$ to the mean optimal costs for MAPs with standard normal assignment costs.</td>
<td>36</td>
</tr>
<tr>
<td>3–10</td>
<td>Estimated and actual mean optimal costs from ten runs for variously sized MAPs developed from different distributions. Included are the average difference and largest difference between estimated mean optimal cost and optimal cost.</td>
<td>37</td>
</tr>
<tr>
<td>3–11</td>
<td>Results showing comparisons between three primal heuristics and the numerical estimate of optimal cost for several problem sizes and types. Shown are the average feasible solution costs from 50 runs of each primal heuristic on random instances.</td>
<td>40</td>
</tr>
<tr>
<td>3–12</td>
<td>Average time to solution in seconds of solving each of five randomly generated problems of various sizes and types. The experiment involved using the B&amp;B solution algorithm with different starting upper bounds developed in three different ways.</td>
<td>43</td>
</tr>
</tbody>
</table>
5–1 Comparisons of the number of cost coefficients of original MAP to that in $\mathcal{A}$. ................................................................. 63

5–2 Table of experimental results of comparing solution quality and time-to-solution for GRASP in solving fully dense and reduced simulated MSMTT problems. Five runs of each algorithm were conducted against each problem. ................................................... 68

6–1 Average number of local minima (2-exchange neighborhood) for different sizes of MAPs with independent assignment costs. .............. 75

6–2 Average number of local minima (3-exchange neighborhood) for different sizes of MAPs with i.i.d. standard normal assignment costs. . . 76

6–3 Proportion of local minima to total number of feasible solutions for different sizes of MAPs with i.i.d. standard normal costs. ........ 76

7–1 Timed results of producing test problems of various sizes. ............ 105

7–2 Chi-square goodness-of-fit test for normal distribution of assignment costs for six randomly selected 5x5x5 test problems. .............. 106

7–3 Number of discrete local minima per $10^6$ feasible solutions. The range is a 95-percent confidence interval based on proportionate sampling. 108

7–4 Comparison of solution times in seconds using an exact solution algorithm of the branch-and-bound variety. ................................. 109

7–5 Comparison of solution results using a GRASP algorithm. ............ 109
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3–1</td>
<td>Branch and Bound on the Index Tree.</td>
<td>24</td>
</tr>
<tr>
<td>3–2</td>
<td>Plots of mean optimal costs for four different sized MAPs with exponential assignment costs.</td>
<td>30</td>
</tr>
<tr>
<td>3–3</td>
<td>Surface plots of mean optimal costs for $3 \leq d \leq 10$ and $2 \leq n \leq 10$ sized MAPs with exponential assignment costs.</td>
<td>32</td>
</tr>
<tr>
<td>3–4</td>
<td>Plots of mean optimal costs for four different sized MAPs with standard normal assignment costs.</td>
<td>32</td>
</tr>
<tr>
<td>3–5</td>
<td>Plots of standard deviation of mean optimal costs for four different sized MAPs with exponential assignment costs.</td>
<td>33</td>
</tr>
<tr>
<td>3–6</td>
<td>Plots of standard deviation of mean optimal costs for four different sized MAPs with standard normal assignment costs.</td>
<td>34</td>
</tr>
<tr>
<td>3–7</td>
<td>Three dimensional MAP with exponential assignment costs. Plot includes both observed mean optimal cost values and fitted values. The two lines are nearly indistinguishable.</td>
<td>36</td>
</tr>
<tr>
<td>3–8</td>
<td>Plots of fitted and mean optimal costs from ten runs of variously sized MAPs developed from the uniform distribution on $[10, 20]$. Note that the observed data and fitted data are nearly indistinguishable.</td>
<td>38</td>
</tr>
<tr>
<td>3–9</td>
<td>Plots of fitted and mean optimal costs from ten runs of variously sized MAPs developed from the exponential distribution with mean three.</td>
<td>38</td>
</tr>
<tr>
<td>3–10</td>
<td>Plots of fitted and mean optimal costs from ten runs of variously sized MAPs developed from a normal distribution, $N(\mu = 5, \sigma = 2)$.</td>
<td>39</td>
</tr>
<tr>
<td>3–11</td>
<td>Branch and bound on the index tree.</td>
<td>41</td>
</tr>
<tr>
<td>5–1</td>
<td>Example of noisy sensor measurements of target locations.</td>
<td>57</td>
</tr>
<tr>
<td>5–2</td>
<td>Example of noisy sensor measurements of close targets. In this case there is false detection and missed targets.</td>
<td>57</td>
</tr>
<tr>
<td>5–3</td>
<td>Search tree data structure used to find a cost coefficient or determine a cost coefficient does not exist.</td>
<td>66</td>
</tr>
<tr>
<td>5–4</td>
<td>Search tree example of a sparse MAP.</td>
<td>67</td>
</tr>
</tbody>
</table>
6–1 Proportion of feasible solutions that are local minima when considering the 2-exchange neighborhood and where costs are i.i.d. standard normal. ................................................................. 76

6–2 Plots of solution quality versus number of local minima when using the 2-exchange neighborhood. The MAP has a size of $d = 4, \ n = 5$ with cost coefficients that are i.i.d. standard normal. ................................. 89

6–3 Plots of solution quality versus number of local minima when using a 3-exchange neighborhood. The MAP has a size of $d = 4, \ n = 5$ with cost coefficients that are i.i.d. standard normal. ................................. 90

7–1 Tree graph for 3x4x4 MAP. .................................................. 98

7–2 Initial tree graph with assignment costs and lower bound path costs. ................................................................. 101

7–3 Tree graph with optimal path and costs. ........................................ 102

7–4 Tree graph used to consider all feasible nodes at level 3 from the first node in level 2. ................................................................. 103

7–5 Final tree graph for a 3x4x4 MAP. ........................................... 104

7–6 Typical normal probability plot for a 5x5x5 test problem. .......... 106

7–7 Typical histogram of 20x30x40 test problem. .......................... 107
The work in this dissertation examines combinatorial problems from a probabilistic approach in an effort to improve existing solution methods or find new algorithms that perform better. Applications addressed here are focused on military uses such as weapon-target assignment, path planning and multisensor multitarget tracking; however, these may be easily extended to the civilian environment.

A probabilistic analysis of combinatorial problems is a very broad subject; however, the context here is the study of input data and solution values.

We investigate characteristics of the mean optimal solution values for random multidimensional assignment problems (MAPs) with axial constraints. Cost coefficients are taken from three different random distributions: uniform, exponential and standard normal. In the cases where cost coefficients are independent uniform or exponential random variables, experimental data indicate that the average optimal value of the MAP converges to zero as the MAP size increases. We give a short proof of this result for the case of exponentially distributed costs when the number of elements in each dimension is restricted to two. In the case of standard normal
costs, experimental data indicate the average optimal value of the MAP goes to negative infinity as the MAP size increases. Using curve fitting techniques, we develop numerical estimates of the mean optimal value for various sized problems. The experiments indicate that numerical estimates are quite accurate in predicting the optimal solution value of a random instance of the MAP.

Using a novel probabilistic approach, we provide generalized proofs of the asymptotic characteristics of the mean optimal costs of MAPs. The probabilistic approach is then used to improve the efficiency of the popular greedy randomized adaptive search procedure.

As many solution approaches to combinatorial problems rely, at least partly, on local neighborhood searches, it is widely assumed the number of local minima has implications on solution difficulty. We investigate the expected number of local minima for random instances of the MAP. We report on empirical findings that the expected number of local minima does impact the effectiveness of three different solution algorithms that rely on local neighborhood searches.

A probabilistic approach is used to develop an MAP test problem generator that creates difficult problems with known unique solutions.
CHAPTER 1
INTRODUCTION

Combinatorial optimization problems are found in everyday life. They are particularly important in military applications as they most often concern management and efficient use of scarce resources. Applications of combinatorial problems are in a period of rapid development which follows from the widespread use of computers and the data available from information systems. Although computers have allowed expanded combinatorial applications, most of these problems remain very hard to solve. The purpose of the work in this dissertation is to examine combinatorial problems from a probabilistic approach in an effort to improve existing solution methods or find new algorithms that perform better. Most applications addressed here are focused on military applications; however, most may be easily extended to the civilian environment.

1.1 Probabilistic Analysis of Combinatorial Problems

In general, probabilistic analysis of combinatorial problems is a very broad subject; however, the context being used here is the study of problem input values and solution values of combinatorial problems. An obvious goal is to determine if parameters (e.g., mean, standard deviation, etc.) of these values can be used to improve the efficiency of a solution algorithm. Alternatively, parameters of these values may be useful in selecting an appropriate solution algorithm. Although problem instance size is directly correlated with the difficulty of determining a solution, we often face problems of similar size that have far different computing times. One can conclude from this that characteristics of the problem data are significant factors.

An example of the study of solution values is by Barvinok and Stephen [13], where the authors obtain a number of results regarding the distribution of solution values.
of the quadratic assignment problem. In the paper, the authors consider questions such as, how well does the optimum of a sample of random permutations approximate the true optimum? They explore an interesting approach in which they consider the “k-th sphere” around the true optimum. The k-th sphere, in simple terms, quantifies the nearness of permutations to the optimum permutation. By allowing the true optimum to represent a bullseye, the authors observe as the k-th sphere contracts to the optimal permutation, the average solution value of a sample of permutations steadily improves.

A study of the quadratic assignment problem (QAP) is found work by Abreu et al. [1] where the authors consider using average and variance of solution costs to establish the difficulty of a particular instance.

Sanchis and Schnabl [103] study the “landscape” of the traveling salesman problem. Considered are number of local minima and autocorrelation functions. The concept of landscape was introduced by Wright [111] and can be thought of as a map of solution values such that there are peaks and valleys. Landscape roughness can give an indication of problem difficulty.

In a study of cost inputs, Reilly [94] suggests that the degree of correlation among input data may influence the difficulty of finding a solution. It is suggested that an extreme level of correlation can produce very challenging problems.

In this dissertation, we use a probabilistic approach to consider how input costs affect solution values in an important class of problems called the multidimensional assignment problem. We also consider the mean optimal costs of various problem instances to include some asymptotic characteristics. We include another interesting probabilistic analysis which is our study of local minima and how the number of local minima affects solution methods. Finally, we use a probabilistic approach to design and analyze a test problem generator.
1.2 Main Contributions and Organization of the Dissertation

The main contributions and organization of this dissertation are briefly discussed in the following paragraphs.

**Survey of the multidimensional assignment problem.** A brief survey of the multidimensional assignment problem (MAP) is provided in Chapter 2. In this chapter, we provide alternative formulations and applications for this important and difficult problem.

**Mean optimal solution values of the MAP.** In Chapter 3 we report experimentally determined values of the mean optimal solution costs of MAPs with cost coefficients that are independent random variables that are uniformly, exponentially or normally distributed. Using the experimental data, we then find curve fitting models that can be used to accurately determine their mean optimal solution costs. Finally, we show how the numerical estimates can be used to improve at least two solution methods of the MAP.

**Proof of asymptotic characteristics of the MAP.** In Chapter 4 we prove some asymptotic characteristics of the mean optimal costs using a novel probabilistic approach.

**Probabilistic approach to solving the data association problem.** Using the probabilistic approach introduced in Chapter 4, we extend the approach in Chapter 5 to more efficiently solve the data association problem that results from the multisensor multitarget tracking problem. In the multisensor multitarget problem noisy measurements are made with an arbitrary number of spatially diverse sensors regarding an arbitrary number of targets with the goal of estimating the trajectories of all the targets present. Furthermore, the number of targets may change by moving into and out of detection range. The problem involves a data association of sensor measurements to targets and estimates the current state of each target. The combinatorial nature of the problem results from the data association problem; that is how
do we optimally partition the entire set of measurements so that each measurement is attributed to no more than one target and each sensor detects a target no more than once?

Expected number of local minima for the MAP. The number of local minima in a problem may provide insight to more appropriate solution methods. Chapter 6 explores the number of local minima in the MAP and then considers the impact of the number of local minima on three solution methods.

MAP test problem generator. As examined in the first five chapters, a probabilistic analysis can be used to develop a priori knowledge of problem instance hardness. In Chapter 7 we develop an MAP test problem generator and use some probabilistic analyses to determine the generator’s effectiveness in creating quality test problems with known unique optimal solutions. Also included is a brief survey of sources of combinatorial test problems.
CHAPTER 2
SURVEY OF THE MULTIDIMENSIONAL ASSIGNMENT PROBLEM

The MAP is a higher dimensional version of the standard (two-dimensional, or linear) assignment problem. The MAP is stated as follows: given $d$, $n$-sets $A_1, A_2, \ldots, A_d$, there is a cost for each $d$-tuple\(^1\) in $A_1 \times A_2 \times \cdots \times A_d$. The problem is to minimize the cost of $n$ tuples such that each element in $A_1 \cup A_2 \cup \cdots \cup A_d$ is in exactly one tuple. The problem was first introduced by Pierskalla \([86]\). Solution methods have included branch and bound \([87, 10, 84]\), Greedy Randomized Adaptive Search Procedure (GRASP) \([4, 74]\), Lagrangian relaxation \([90, 85]\), a genetic algorithm based heuristic \([25]\), and simulated annealing \([27]\).

2.1 Formulations

A well-known instance of the MAP is the three-dimensional assignment problem (3DAP). An example of the 3DAP consists of minimizing the total cost of assigning $n_i$ items to $n_j$ locations at $n_k$ points in time. The three-dimensional MAP can be

\(^1\) Tuple an abstraction of the sequence: single, double, triple,\ldots, $d$-tuple. Tuple is used to denote a point in a multidimensional coordinate system.
formulated as

\[
\min \sum_{i=1}^{n_i} \sum_{j=1}^{n_j} \sum_{k=1}^{n_k} c_{ijk} x_{ijk}
\]
\[
\text{s.t. } \sum_{j=1}^{n_j} x_{ijk} = 1 \quad \text{for all } i = 1, 2, \ldots, n_i,
\]
\[
\sum_{i=1}^{n_i} x_{ijk} \leq 1 \quad \text{for all } j = 1, 2, \ldots, n_j,
\]
\[
\sum_{i=1}^{n_i} x_{ijk} \leq 1 \quad \text{for all } k = 1, 2, \ldots, n_k,
\]
\[
x_{ijk} \in \{0, 1\} \quad \text{for all } i, j, k \in \{1, \ldots, n\},
\]
\[
n_i \leq n_j \leq n_k,
\]

where \(c_{ijk}\) is the cost of assigning item \(i\) to location \(j\) at time \(k\). In this formulation, the variable \(x_{ijk}\) is equal to 1 if and only if the \(i\)-th item is assigned to the \(j\)-th location at time \(k\) and zero otherwise. If we consider additional dimensions for this problem, the formulation can be similarly extended in the following way:

\[
\min \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} c_{i_1\ldots i_d} x_{i_1\ldots i_d}
\]
\[
\text{s.t. } \sum_{i_{d-1}=1}^{n_{d-1}} \cdots \sum_{i_2=1}^{n_2} x_{i_1\ldots i_d} = 1 \quad \text{for all } i_1 = 1, 2, \ldots, n_1,
\]
\[
\sum_{i_{d-1}=1}^{n_{d-1}} \cdots \sum_{i_k=1}^{n_k+1} \sum_{i_{k+1}=1}^{n_{k+1}} \cdots \sum_{i_d=1}^{n_d} x_{i_1\ldots i_d} \leq 1
\]
\[
\text{for all } k = 2, \ldots, d - 1, \text{ and } i_k = 1, 2, \ldots, n_k,
\]
\[
\sum_{i_{d-1}=1}^{n_{d-1}} \cdots \sum_{i_{d-2}=1}^{n_{d-2}} x_{i_1\ldots i_d} \leq 1 \quad \text{for all } i_d = 1, 2, \ldots, n_d,
\]
\[
x_{i_1\ldots i_d} \in \{0, 1\} \quad \text{for all } i_1, i_2, \ldots, i_d \in \{1, \ldots, n\},
\]
\[
n_1 \leq n_2 \leq \cdots n_d,
\]

where \(d\) is the dimension of the MAP.
If we allow \( n_1 = n_2 = \cdots = n_d = n \), an equivalent formulation states the MAP in terms of permutations \( \delta_1, \ldots, \delta_{d-1} \) of numbers 1 to \( n \). Using this notation, the MAP is equivalent to

\[
\min_{\delta_1, \ldots, \delta_{d-1} \in \Pi^n} \sum_{i=1}^{n} c_{i, \delta_1(i), \ldots, \delta_{d-1}(i)},
\]

where \( \Pi^n \) is the set of all permutations of \( \{1, \ldots, n\} \).

### 2.2 Complexity

Solving even moderate sized instances of the MAP is a difficult task. A linear increase in the number of dimensions brings an exponential increase in the number of cost coefficients in the problem and the number of feasible solutions, \( N \), is given by the relation

\[
N = \prod_{i=2}^{d} \frac{n_i!}{(n_i - n_1)!}.
\]

In general, the MAP is known to be \( NP \)-hard, a fact which follows from results work by Garey and Johnson [44]. Even in the case when costs take on a special structure of triangle inequalities, Crama and Spieksma [31] prove the three-dimensional problem remains \( NP \)-hard. However, special cases that are not \( NP \)-hard do exist.

Burkard, Rüdolf, and Woeginger [23] investigate the three-dimensional problems with decomposable cost coefficients. Given three \( n \)-element sequences \( a_i, b_i \) and \( c_i \), \( i = 1, \ldots, n \), a cost coefficient \( d_{ijk} \) is decomposable when \( d_{ijk} = a_ibjc_k \). Burkard [23] finds the minimization and maximization of the three-dimensional assignment problem have different complexities. While the maximization problem is solvable in polynomial time, the minimization problem remains \( NP \)-hard. On the other hand, Burkard [23] identifies several structures where the minimization problem is polynomially solvable.

A polynomially solvable case of the MAP occurs when the cost coefficients are taken from a Monge matrix [22]. An \( m \times n \) matrix \( C \) is called a Monge matrix if \( c_{ij} + c_{rs} \leq c_{is} + c_{rj} \) for all \( 1 \leq i < r \leq m \), \( 1 \leq j < s \leq n \). Another way to describe
the Monge array is to again consider the matrix $C$. Any two rows and two columns must intersect at exactly four elements. The rows and columns satisfy the Monge property if the sum of the upper-left and lower-right elements is at most the sum of the upper-right and lower-left elements. This can easily be extended to higher dimensions. Because of the special structure of the Monge matrix, the MAP becomes polynomially solvable with a lexicographical greedy algorithm and the identity permutation is an optimal solution.

2.3 Applications

The MAP has applications in numerous areas such as, data association [8], scheduling teaching practices [42], production of printed circuit boards [30], placement of distribution warehouses [87], multisensor multitarget problems [74, 91], tracking elementary particles [92] and multiagent path planning [84]. More examples and an extensive discussions of the subject can be found in two extensive surveys [81, 19]. A particular military application of the MAP is the Weapon Target Assignment problem which is discussed in the following subsection.

2.3.1 Weapon Target Assignment Problem

The target-based Weapon Target Assignment (WTA) problem [81] considers optimally assigning $W$ weapons to $T$ targets so that the total expected damage to the targets is maximized. The term target-based is used to distinguish these problems from the asset-based or defense-based problems where the goal of these problems is to assign weapons to incoming missiles to maximize the surviving assets. The target-based problems primarily apply to offensive strategies.

Assume at a particular instant in time the number and location of weapons and targets are known with certainty. Then a single assignment may be made at that instant. Consider $W$ weapons and $T$ targets and define $x_{ij}, i = 1, 2, \ldots, W, j =$
1, 2, \ldots, T as:

\[ x_{ij} = \begin{cases} 
1 & \text{if weapon } i \text{ assigned to target } j, \\
0 & \text{otherwise.} 
\end{cases} \]

Given that weapon \( i \) engages target \( j \), the outcome is random.

\[ P(\text{target } j \text{ is destroyed by weapon } i) = P_{ij} \]
\[ P(\text{target } j \text{ is not destroyed by weapon } i) = 1 - P_{ij} \]

If one assumes that each weapon engagement is independent of every other engagement, then the outcomes of the engagements are independent and Bernoulli distributed. Note that we let \( q_{ij} = (1 - P_{ij}) \) which is the probability that target \( j \) survives an encounter with weapon \( i \).

Now assign \( V_j \) to indicate a value for each target \( j \). The objective is to maximize the damage to targets or minimize the value of the targets which may be formulated

\[
\min \sum_{j=1}^{T} V_j \prod_{i=1}^{W} q_{ij}^{x_{ij}} \quad (2.1)
\]

subject to

\[
\sum_{j=1}^{T} x_{ij} = 1, \quad i = 1, 2, \ldots, W \\
x_{ij} \in \{0, 1\}.
\]

This is a nonlinear assignment problem and is known to be NP-complete. Notice a few characteristics of the above problem.

- Since there is no cost for employing a weapon, all weapons will be used.
- The solution may result in some targets not being targeted because they are relatively worthless and/or because they are very difficult to defeat.

A transformation of this formulation to an MAP may be accomplished. Using a two weapon, two target example, the transformation follows. First observe that the objective function of (2.1) may be written as

\[
\min V_1[q_{11}^{x_{11}}q_{21}^{x_{21}}] + V_2[q_{12}^{x_{12}}q_{22}^{x_{22}}]. \quad (2.2)
\]
Obviously, the individual probabilities of survival, $q_{ij}$, go to one if weapon $i$ does not engage target $j$. Therefore, using the first term of the objective function in equation (2.2) as an example, the first term becomes

\[ V_1[q_{11}q_{21}] \text{ if } x_{11} = 1 \text{ and } x_{21} = 1, \text{ or} \]
\[ V_1[q_{11}] \text{ if } x_{11} = 1 \text{ and } x_{21} = 0, \text{ or} \]
\[ V_1[q_{21}] \text{ if } x_{11} = 0 \text{ and } x_{21} = 1, \text{ or} \]
\[ V_1 \text{ if } x_{11} = 0 \text{ and } x_{21} = 0. \]

Notice these terms are now constant cost values. A different decision variable, $\rho_{\alpha\beta j}$, may be introduced that represents the status of engaging the different weapons on target $j$. $\alpha = \{1, 2\}$ represents weapon 1’s status of engagement on target $j$, where $\alpha = 1$ means weapon 1 engages target $j$ and $\alpha = 2$ otherwise. Similarly, $\beta = \{1, 2\}$ represents weapon 2’s status of engagement of target $j$. For example,

\[ \rho_{11j} = \begin{cases} 
1 & \text{both the first and second weapon engage target } j, \\
0 & \text{else,}
\end{cases} \]

and,

\[ \rho_{12j} = \begin{cases} 
1 & \text{the first but not the second weapon engages target } j, \\
0 & \text{else.}
\end{cases} \]

The cost values may now be represented by $c_{\alpha\beta j}$. For example, $c_{11j} = V_1[q_{11}q_{21}]$ and $c_{12j} = V_1[q_{11}]$. Using these representations, the first term of objective function (2.2) becomes

\[ c_{111}\rho_{111} + c_{121}\rho_{121} + c_{211}\rho_{211} + c_{221}\rho_{221}. \]
For the two weapon, two target scenario, (2.1) may reformulated to a three dimensional MAP as follows.

\[
\begin{align*}
\min & \quad \sum_{\alpha=1}^{2} \sum_{\beta=1}^{2} \sum_{j=1}^{2} c_{\alpha \beta j} \rho_{\alpha \beta j} \\
\text{s.t.} & \quad \sum_{j=1}^{2} \sum_{\beta=1}^{2} \rho_{\alpha \beta j} = 1 \quad \forall \, \alpha = 1, 2 \\
& \quad \sum_{\alpha=1}^{2} \sum_{j=1}^{2} \rho_{\alpha \beta j} = 1 \quad \forall \, j = 1, 2 \\
& \quad \sum_{\alpha=1}^{2} \sum_{j=1}^{2} \rho_{\alpha \beta j} = 1 \quad \forall \, \beta = 1, 2 \\
& \quad \rho_{\alpha \beta j} \in \{0, 1\} \quad \forall \, \alpha, \beta, j.
\end{align*}
\]

In general, reformulation of (2.1) will result in a \(W + 1\) dimensional MAP. The number of indices will be \(T\). As mentioned above, weapon costs are not considered in this formulation which results in all weapons being assigned. A more realistic formulation that considers weapon costs is developed in the next subsection.

### 2.3.2 Considering Weapon Costs in the Weapon Target Assignment Problem

The formulation in the previous subsection excludes weapon costs which can result in overkill or poor use of expensive weapons on low valued targets. A more realistic formulation includes weapon costs. Let \(C_i\) be the cost of the \(i\)-th weapon and let \(j = T + 1\) be a dummy target. We may now reformulate (2.1) as

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{W} \sum_{j=1}^{T} C_i x_{ij} - \sum_{i=1}^{W} C_i x_{ij} + \sum_{j=1}^{T} V_j \prod_{i=1}^{W} d_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{j=1}^{T+1} x_{ij} = 1, \quad i = 1, 2, \ldots, W \\
& \quad x_{ij} = \{0, 1\}.
\end{align*}
\]
The first summation term considers the costs of weapons assigned to actual targets. The second summation term considers the savings by applying weapons to the dummy target.

Following a similar development as in the previous subsection, we obtain a generalized MAP formulation that incorporates weapon costs.

\[
\begin{align*}
\min & \quad \sum_{w_1=1}^{T+1} \sum_{w_2=1}^{T+1} \cdots \sum_{j=1}^{T+1} c_{w_1w_2\ldots j}\rho_{w_1w_2\ldots j} \\
\text{s.t.} & \quad \sum_{w_2=1}^{T+1} \cdots \sum_{j=1}^{T+1} \rho_{w_1w_2\ldots j} = 1 \quad \forall \ w_1 = 1, 2, \ldots, T + 1 \\
& \quad \sum_{w_1=1}^{T+1} \cdots \sum_{w_{k-1}=1}^{T+1} \sum_{w_{k+1}=1}^{T+1} \cdots \sum_{j=1}^{T+1} \rho_{w_1w_2\ldots j} = 1 \\
& \quad \forall \ k = 1, \ldots, W - 1, \text{ and } w_k = 1, 2, \ldots, T + 1 \\
& \quad \sum_{w_1=1}^{T+1} \cdots \sum_{w_{W-1}=1}^{T+1} \rho_{w_1w_2\ldots j} = 1 \quad \forall \ j = 1, 2, \ldots, T + 1 \\
& \quad \rho_{w_1w_2\ldots j} \in \{0, 1\} \quad \forall \ w_1, w_2, \ldots, j.
\end{align*}
\]

This formulation results in a \( W + 1 \) dimensional MAP with \( T + 1 \) elements in each dimension.

### 2.4 Summary

The MAP has been studied extensively in the last couple of decades and its applications in both military and civilian arenas has been rapidly expanding. The difficult nature of the problem requires researchers to continuously consider novel solution methods and a probabilistic approach provides some needed insight in developing these solution methods.
CHAPTER 3
CHARACTERISTICS OF THE MEAN OPTIMAL SOLUTION TO THE MAP

In this chapter, we investigate characteristics of the mean optimal solution values for random MAPs with axial constraints. Throughout the study, we consider cost coefficients taken from three different random distributions: uniform, exponential and standard normal. In the cases of uniform and exponential costs, experimental data indicate that the mean optimal value converges to zero when the problem size increases. We give a short proof of this result for the case of exponentially distributed costs when the number of elements in each dimension is restricted to two. In the case of standard normal costs, experimental data indicate the mean optimal value goes to negative infinity with increasing problem size. Using curve fitting techniques, we develop numerical estimates of the mean optimal value for various sized problems. The experiments indicate that numerical estimates are quite accurate in predicting the optimal solution value of a random instance of the MAP.

3.1 Introduction

$NP$-hard problems present important challenges to the experimental researcher in the field of algorithms. That is because, being difficult to solve in general, careful restrictions must be applied to a combinatorial optimization problem in order to solve some of its instances. However, it is also difficult to create instances that are representative of the problem, suitable for the technique or algorithm being used, and at the same time interesting from the practical point of view.

One of the simplest and, in some cases, most useful ways of creating problem instances consists of drawing values from a random distribution. Using this procedure, one wishes to create a problem that is difficult “on average,” but that can also appear as the outcome of some natural process.
Thus, one of the questions that arises is how a random problem will behave in terms of solution value, given some distribution function and parameters from which values are taken. This question turns out to be very difficult to solve in general. As an example, for the Linear Assignment Problem (LAP), results have not been easy to prove, despite intense research in this field [5, 28, 29, 55, 82].

In this chapter we perform a computational study of the asymptotic behavior for instances of the MAP.

3.1.1 Basic Definitions and Results

The MAP is an \( NP \)-hard combinatorial optimization problem, which extends the Linear Assignment Problem (LAP) by adding more sets to be matched. The number \( d \) of sets corresponds to the dimension of the MAP. In the special case of the LAP, we have \( d = 2 \). Chapter 2 provides an overview of the MAP to include formulations and applications.

Let \( z(I) \) be the value of the optimum solution for an instance \( I \) of the MAP. We denote by \( \bar{z}^* \) the expected value of \( z(I) \), over all instances \( I \) constructed from a random distribution (the context will make clear what specific distribution we are talking about). In the problem instances considered in this chapter, we have \( n_1 = n_2 = \cdots n_d = n \).

Our main contribution in this chapter is the development of numerical estimates of the mean optimal costs for randomly generated instances of the MAP. The experiments performed show that for uniform \([0, 1]\) and exponentially distributed costs, the optimum value converges to zero as the problem size increases. These results are not surprising for an increase in \( d \) since the number of cost coefficients increases exponentially with \( d \). However, convergence to zero for increasing \( n \) is not as obvious since the objective function is the sum of \( n \) cost coefficients. Experiments with standard normally distributed costs show that the optimum value goes to \(-\infty\) as the problem
size increases. More interestingly, the experiments show convergence even for small values of $n$ and $d$.

The three distributions (exponential, uniform and normal) were chosen for analysis as they are very familiar to most practitioners. Although we would not expect real-world problems to have cost coefficients that follow exactly these distributions, we believe that our results may be extended to other cost coefficient distributions.

### 3.1.2 Motivation

The study of asymptotic values for MAPs has important motivations arising from theory and from practical applications. First, there are few theoretical results on this subject, and therefore, practical experiments are a good method for determining how MAPs behave for instances with random values. Determining asymptotic values for such problems is a major open question in combinatorics, which can be made clear by careful experimentation.

Another motivation for this work has been the possible use of asymptotic results in the practical setting of heuristic algorithms. When working with MAPs, one of the greatest difficulties is the need to cope with a large number of entries in the multidimensional vector of costs. For example, in an instance with $d$ dimensions and minimum dimension size $n$, there are $n^d$ cost elements that must be considered for the optimum assignment. Solving an MAP can become very hard when all elements of the cost vector must be read and considered during the algorithm execution. This happens because the time needed to read $n^d$ values makes the algorithm exponential on $d$. A possible use of the results shown in this chapter allows one, having good estimates of the expected value of an optimal solution and the distribution of costs, to discard a large number of entries in the cost vector, which have low probability of being part of the solution. By doing this, we can improve the running time of most algorithms for the MAP.
Finally, while some computational studies have been performed for the random LAP, such as by Pardalos and Ramakrishnan [82], there are limited practical and theoretical results for the random MAP. In this chapter we try to improve in this respect by presenting extensive results of computational experiments for the MAP.

3.1.3 Asymptotic Studies and Results

Asymptotical studies of random combinatorial problems can be traced back to the work of Beardwood, Halton and Hammersley [14] on the traveling salesman problem (TSP). Other work includes studies of the minimum spanning tree [41, 105], Quadratic Assignment Problem (QAP) [21] and, most notably, studies of the Linear Assignment Problem (LAP) [5, 28, 55, 64, 83, 76, 82, 109]. A more general analysis was made on random graphs by Lueker[69].

In the case of the TSP, the problem is to let $X_i, X_i = 1, \ldots, n$, be independent random variables uniformly distributed on the unit square $[0, 1]^2$, and let $L_n$ denote the length of the shortest closed path (usual Euclidian distance) which connects each element of $\{X_1, X_2, \ldots, X_n\}$. The classic result proved by Beardwood et al. [14] is

$$\lim_{n \to \infty} \frac{L_n}{\sqrt{n}} = \beta$$

with probability one for a finite constant $\beta$. This becomes significant, as addressed by Steele [104], because it is key to Karp’s algorithm [54] for solving the TSP. Karp uses a cellular dissection algorithm for the approximate solution. The above result may be summarized as implying that the optimal tour through $n$ points is sharply predictable when $n$ is large and the dissection method tends to give near-optimal solutions when $n$ is large. This points to an idea of using asymptotic results to develop effective solution algorithms.

In the minimum spanning tree problem, consider an undirected graph $G = (N, A)$ defined by the set $N$ of $n$ nodes and a set $A$ of $m$ arcs, with a length $c_{ij}$ associated with each arc $(i, j) \in A$. The problem is to find a spanning tree of $G$, called a minimum
spanning tree (MST), that has the smallest total length, $L_{MST}$, of its constituent arcs [3]. If we let each arc length $c_{ij}$ be an independent random variable drawn from the uniform distribution on $[0, 1]$, Frieze [41] showed that

$$E[L_{MST}] \rightarrow \zeta(3) = \sum_{j=1}^{\infty} \frac{1}{j^3} = 1.202 \ldots \text{ as } n \rightarrow \infty.$$ 

This was followed by Steele [105], where the Tutte polynomial for a connected graph is used to develop an exact formula for the expected value of $L_{MST}$ for a finite graph with uniformly distributed arc costs. Additional work concerning the directed minimum spanning tree is also available [17].

For the Steiner tree problem which is an $NP$-hard variant of the MST, Bollobás, et al. [18] proved that with high probability the weight of the Steiner tree is $(1 + O(1))(k - 1)(\log n - \log k)/n$ when $k = O(n)$ and $n \rightarrow \infty$ and where $n$ is the number of vertices in a complete graph with edge weights chosen as i.i.d. random variables distributed as exponential with mean one. In the problem, $k$ is the number of vertices contained in the Steiner tree.

A famous result that some call the Burkard-Fincke condition relates to the QAP. The QAP was introduced by Koopmans and Beckmann [60] in 1957 as a model for the location of a set of indivisible economical activities. QAP applications, extensions and solution methods are well covered in work by Horst et al. [51]. The Burkard-Fincke condition [21] is that the ratio between the best and worst solution values approaches one as the size of the problem increases.

Another way to think of this is for a large problem any permutation is close to optimal. According to Burkard and Fincke [21] this condition applies to all problems in the class of combinatorial optimization problems with sum- and bottleneck objective functions. The Linear Ordering Problem (LOP) [26] falls into this category as well. Burkard and Fincke suggest that this result means that very simple heuristic algorithms can yield good solutions for very large problems.
Recent work by Aldous and Steele [6] provides part survey, part tutorial on the objective method in understanding asymptotic characteristics of combinatorial problems. They provide some concrete examples of the approach and point out some unavoidable limitations.

In terms of the asymptotic nature of combinatorial problems, the most explored problem has been the LAP. In the LAP we are given a matrix $C^{n \times n}$ with coefficients $c_{ij}$. The objective is to find a minimum cost assignment; i.e., $n$ elements $c_{1j_1}, \ldots, c_{nj_n}$, such that $j_p \neq j_q$ for all $p \neq q$, with $j_i \in \{1, \ldots, n\}$, and $\sum_{i=1}^{n} c_{ij_i}$ is minimum.

A well known conjecture by Mézard and Parisi [71, 72] states that the optimal solution for instances where costs $c_{ij}$ are drawn from an exponential or uniform distribution, approaches $\pi^2/6$ when $n$ (the size of the instance) approaches infinity. Pardalos and Ramakrishnan [82] provide additional empirical evidence that the conjecture is indeed valid. The conjecture was expanded by Parisi [83], where in the case of costs drawn from an exponential distribution the expected value of the optimal solution of an instance of size $n$ is given by

$$\sum_{i=1}^{n} \frac{1}{i^2}. \quad (3.1)$$

Moreover,

$$\sum_{i=1}^{n} \frac{1}{i^2} \rightarrow \frac{\pi^2}{6} \quad \text{as } n \rightarrow \infty.$$

This conjecture has been further strengthened by Coppersmith and Sorkin [28]. The authors conjecture that the expected value of the optimum $k$-assignment, for a fixed matrix of size $n \times m$, is given by

$$\sum_{i,j \geq 0, i+j<k} \frac{1}{(m-i)(n-j)}.$$
They also presented proofs of this conjecture for small values of \( n, m \) and \( k \). The conjecture is consistent with previous work [71, 83], since it can be proved that for \( m = n = k \) this is simply the expression in (3.1)

Although until recently the proofs of these conjectures have eluded many researchers, there has been progress in the determination of upper and lower bounds. Walkup [109] proved an upper bound of 3 on the asymptotic value of the objective function, when the problem size increases. This was improved by Karp [55], who showed that the limit is at most 2. On the other hand, Lazarus [64] proved a lower bound of \( 1 + 1/e \approx 1.3679 \). More recently this result was improved by Olin [76] to the tighter lower bound value of 1.51.

Finally, recent papers by Linusson and Wästlund [67] and Nair et al. [75] have solved the conjectures of Mézard and Parisi, and Coppersmith and Sorkin.

Concerning the MAP, not many results are known about the asymptotic behavior of the optimum solution for random instances. However, one example of recent work is that by Huang et. al. [52]. In this work the authors consider the complete \( d \)-partite graph with \( n \) vertices in each of \( d \) sets. If all edges in this graph are assigned independent weights that are uniformly distributed on \([0,1]\), then the expected minimum weight perfect \( d \)-dimensional matching is at least \( \frac{3}{16}n^{1-2/d} \). They also describe a randomized algorithm to solve this problem where the expected solution has weight at most \( 5d^3n^{1-2/d} + d^{15} \) for all \( d \geq 3 \). However, note that for even a moderate size for \( d \), this upper bound is not tight.

### 3.1.4 Chapter Organization

This chapter is organized as follows. In the next section, we give a closed form result on the mean optimal costs for a special case of the MAP when the number of elements in each dimension is equal to 2. The method used to solve the MAP employs a branch-and-bound algorithm, described in Section 3.3, to find exact solutions to the problem. Then, in Section 3.4 we present the computational results and
curve fitting models to estimate the mean optimal costs. Following this, we provide some methods to use the numerical models to improve the efficiency of two solution algorithms. Finally, concluding remarks and future research directions are presented in Section 3.6.

3.2 Mean Optimal Costs for a Special Case of the MAP

In this section we present a result regarding the asymptotical behavior of $\bar{z}^*$ in the special case of the MAP where $n = 2$, $d \geq 3$, and cost elements are independent exponentially distributed with mean one. This is done to give a flavor of how these results can be obtained. For proofs of a generalization of this theorem, including normal distributed costs, refer to Chapter 4. Initially, we employ the property stated in the following proposition.

**Proposition 3.1** In an instance of the MAP with $n = 2$ and i.i.d. exponential cost coefficients with mean 1, the cost of each feasible solution is an independent gamma distributed random variable with parameters $\alpha = 2$, and $\lambda = 1$.

**Proof:** Let $I$ be an instance of MAP with $n = 2$. Each feasible solution for $I$ is an assignment $a_1 = c_1, \delta_1(1), ..., \delta_{d-1}(1)$, $a_2 = c_2, \delta_1(2), ..., \delta_{d-1}(2)$, with cost $z = a_1 + a_2$. The important feature of such assignments is that for each fixed entry $c_1, \delta_1(1), ..., \delta_{d-1}(1)$, there is just one remaining possibility, namely $c_2, \delta_1(2), ..., \delta_{d-1}(2)$, since each dimension has only two elements. This implies that different assignments cannot share elements in the cost vector, and therefore different assignments have independent costs $z$. Now, $a_1$ and $a_2$ are independent exponential random variables with parameter 1. Thus $z = a_1 + a_2$ is a Gamma($\alpha, \lambda$) random variable, with parameters $\alpha = 2$ and $\lambda = 1$.

According to the proof above, it is clear why instances with $n \geq 3$ do not have the same property. Different feasible solutions share elements of the cost vector, and therefore the feasible solutions are not independent of each other. For example, consider a problem of size $d = 3$, $n = 3$. A feasible solution to this problem is
Another feasible solution is $c_{111}$, $c_{223}$, and $c_{332}$. Note that both solutions share the cost coefficient $c_{111}$ and are not independent.

Suppose that $X_1, X_2, \ldots, X_k$ are $k$ independent gamma distributed variables. Let $X_{(i)}$ be the $i$th smallest of these. Applying order statistics [33], we have the following expression for the expected minimum value of $k$ independent identically distributed random variables

$$E[X_{(1)}] = \int_0^\infty kxf(x)(1 - F(x))^{k-1} dx,$$

where $f(x)$ and $F(x)$ are, respectively, the density and distribution functions of the gamma random variable.

The problem of finding $\overline{z^*}$ for the special case when $n = 2$ and $d \geq 3$ corresponds to finding the expected minimum cost $E[X_{(1)}]$, for $k = 2^{d-1}$ independent gamma distributed feasible solution costs, with parameters $\alpha = 2$, and $\lambda = 1$ (note that $k$ is the number of feasible solutions). Through some routine calculus, and noting a resulting pattern as $k$ is increased, we find the following relationship

$$\overline{z^*} = \sum_{j=0}^{k-1} \binom{k-1}{j} \prod_{i=1}^{j+2} \frac{i}{k}.$$

The above equation can be used to prove the asymptotic characteristics of the mean optimal cost of the MAP as $d$ increases. We also note that this special result for the MAP follows directly from Lemma2(ii) by Szpankowski [106]. As an alternative approach, we use the above equation to prove the following theorem.

**Theorem 3.2** For the MAP with $n = 2$, and i.i.d. exponential cost coefficients with mean one, $\overline{z^*} \rightarrow 0$ as $d \rightarrow \infty$. 


**Proof:** When \( d \to \infty \), then \( 2^{d-1} = k \to \infty \) as well. So we prove the result when \( k \to \infty \). We have

\[
\bar{z}^* = \sum_{j=0}^{k-1} \frac{(k - 1)^j}{j!} \frac{2\pi(k - 1)^{(k - j)(k - j + 1)}}{k^{k-j+1}}
\]

Equality (3.4) is found by a change of variable. Using Stirling’s approximation \( n! \approx (n/e)^n \sqrt{2\pi n} \), we have

\[
\bar{z}^* \approx \sum_{j=0}^{k-1} \left(\frac{k - 1}{e}\right)^j \frac{\sqrt{2\pi(k - 1)^{(k - j)(k - j + 1)}}}{j!}
\]

Note that the summation in Formula (3.7) is exactly \( E[(k-j)(k-j+1)] \) for a Poisson distribution with parameter \( k \), which therefore has value \( k \). Thus,

\[
\bar{z}^* \leq e\sqrt{2\pi\frac{(k - 1)^{k-1/2}}{k^k}}
\]

and as

\[
\frac{(k - 1)^{k-1/2}}{k^k} \to 0 \quad \text{when} \quad k \to \infty,
\]

the theorem is proved.

As will be shown in Section 3.4, experimental results support these conclusions, even for relatively small values of \( d \). Table 3–1 provides the value of \( \bar{z}^* \) for MAPs of sizes \( n = 2, 3 \leq d \leq 10 \). We note that a similar approach and results may be obtained for other distributions of cost coefficients. For example, we have similar results if the
cost coefficients are independent gamma distributed random variables, since the sum of gamma random variables is again a gamma random variable.

Table 3–1: Mean optimal solution costs obtained from the closed form equation for MAPs of sizes \( n = 2, 3 \leq d \leq 10 \) and with cost coefficients that are independent exponentially distributed with mean one.

<table>
<thead>
<tr>
<th>( d ) ( n )</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.804</td>
</tr>
<tr>
<td>4</td>
<td>0.530</td>
</tr>
<tr>
<td>5</td>
<td>0.356</td>
</tr>
<tr>
<td>6</td>
<td>0.242</td>
</tr>
<tr>
<td>7</td>
<td>0.167</td>
</tr>
<tr>
<td>8</td>
<td>0.116</td>
</tr>
<tr>
<td>9</td>
<td>0.080</td>
</tr>
<tr>
<td>10</td>
<td>0.056</td>
</tr>
</tbody>
</table>

### 3.3 Branch and Bound Algorithm

This section describes the Branch and bound (B&B) algorithm used in the experiments to optimally solve the MAPs. Branch and bound is essentially an implicit enumeration algorithm. The worst-case scenario for the algorithm is to have to calculate every single feasible solution. However, by using a bounding technique, the algorithm is typically able to find an optimal solution by only searching a limited number of solutions. The index-based B&B is an extension of the three dimensional B&B proposed by Pierskalla [87] where an index tree data structure is used to represent the cost coefficients. There are \( n \) levels in the index tree with \( n^{d-1} \) nodes on each level for a total \( n^d \) nodes. Each level of the index tree has the same value in the first index. A feasible solution can be constructed by first starting at the top level of the tree. The partial solution is developed by moving down the tree one level at a time and adding a node that is feasible with the partial solution. The number of nodes that are feasible to a partial solution developed at level \( i \), for \( i = 1, 2, \ldots, n \) is \((n-i)^{d-1}\). A complete feasible solution is obtained upon reaching the bottom or \( n^{th}\)-level of the tree. Deeper MAP tree representations provide more opportunities for
B&B algorithms to eliminate branches. Therefore, we would expect the index-based B&B to be more effective for a larger number of elements in each dimension.

### 3.3.1 Procedure

The B&B approach proposed here finds the optimal solution by moving through the index-based tree representation of the MAP. The algorithm avoids having to check every feasible solution by eliminating branches with lower bounds that are greater than the best-known solution. The approach is presented as a pseudo-code in Figure 3–1.

```plaintext
procedure IndexBB(L)
  1  for i = 1,...,n do k_i ← 0
  2  S ← ∅
  3  i ← 1
  4  while i > 0 do
  5    if k_i = |L_i| then
  6      S ← S\{s_i
  7      k_i ← 0
  8      i ← i − 1
  9    else
 10      k_i = k_i + 1
 11    if Feasible(S, L_{i,k_i}) then
 12      S ← S ∪ L_{i,k_i}
 13    if LB(S) < \bar{z}^∗ then
 14      if i = n then
 15        \bar{S} ← S
 16        \bar{z} ← Objective(S)
 17      else
 18        i ← i + 1
 19    else
 20      S ← S\{s_i
 21 end
 22 return(\bar{S}, \bar{z})
end IndexBB
```

Figure 3–1: Branch and Bound on the Index Tree.

The algorithm initializes the tree level markers k_i, the solution set S, and the current tree level i in Steps 1–3. The value of the best-known solution set \bar{S} is denoted as \bar{z}. Level markers are used to track the location of cost coefficients on
the tree levels and $L_i$ is the set of coefficients at each level $i$. The solution set $S$ contains the cost coefficients taken from the different tree levels. Steps 4–21 perform an implicit enumeration of every feasible path in the index-based tree. The procedure investigates every possible path below a given node before moving on to the next node in the same tree level. Once all the nodes in a given level are searched or eliminated from consideration through the use of upper and lower bounds, the algorithm moves up to the previous level and moves to the next node in the new level. Step 11 checks if a given cost coefficient $L_{i,k_i}$, which is the $k_i$-th node on level $i$, is feasible to the partial solution set. If the cost coefficient is feasible and if its inclusion does not cause the lower bound of the objective function to surpass the best-known solution, then the coefficient is kept in the solution set. Otherwise, it is removed from $S$ in Step 20.

A lower bound that may be implemented to try to remove some of the tree branches is given by:

$$
LB(S) = \sum_{i=1}^{r} S_i + \sum_{i=r+1}^{n} \min_{i,j_m} c_{i,j_2,\ldots,j_d},
$$

where $r = |S|$ is the size of the partial solution and $S_i$ is the cost coefficient selected from level $i$ of the index-based MAP representation. This expression finds a lower bound by summing the values of all the cost coefficients that are already in the partial solution and the minimum cost coefficient at each of the tree levels underneath the last level searched. The lower bound consists of $n$ elements, one from each level. If a cost coefficient from a given level is in the partial solution, then that coefficient is used in the calculation of the lower bound. If none of the coefficients from a given level is found in the partial solution, then the smallest coefficient from that level is used.

Before starting the algorithm, an initial feasible solution is needed for an upper bound. A natural selection would be

$$
\bar{S} = \{ c_{i,j_2,j_3,\ldots,j_d} \mid i = j_m \text{ for } m = 2, 3, \ldots, d; \ i = 1, 2, \ldots, n \}.
$$
The algorithm initially partitions the cost array into \( n \) groups or tree levels with respect to the value of their first index. The first coefficient to be analyzed is the node furthest to the left at level \( i = 1 \). If the lower bound of the partial solution that includes that node is lower than the initial solution, the partial solution is kept. It then moves to the next level with \( i = 2 \) and again analyzes the node furthest to the left. The algorithm keeps moving down the tree until it either reaches the bottom or finds a node that results in a partial solution having a lower bound value higher than the initial solution. If it does reach the bottom, a feasible solution has been found. If the new solution has a lower objective value than the initial solution, the latest solution is kept as the current best-known solution. On the other hand if the algorithm does encounter a node which has a lower bound greater than the best-known solution, then that node and all the nodes underneath it are eliminated from the search. The algorithm then analyzes the next node to the right of the node that did not meet the lower bound criteria. Once all nodes at a given level have been analyzed, the algorithm moves up to the previous level and begins searching on the next node to the right of the last node analyzed on that level.

We discuss different modifications that may be implemented on the original B&B algorithm to help increase the rate of convergence. The B&B algorithm’s performance is directly related to the tightness of the upper and lower bounds. The rest of this section addresses the problem of obtaining a tighter upper bound. The objective is to obtain a good solution as early as possible. By having a low upper bound early in the procedure, we are able to eliminate more branches and guarantee an optimal solution in a shorter amount of time. The modifications that we introduce are sorting the nodes in all the tree levels and performing a local search algorithm that guarantees local optimality.
3.3.2 Sorting

There are two ways to sort the index-based tree. The first is to sort every level of the tree once before the branch and bound algorithm begins. By using this implementation, the sorting complexity is minimized. However, the drawback is that infeasible cost coefficients are mixed in with the feasible ones. The algorithm would have to perform a large number of feasibility checks whenever a new coefficient is needed from each level.

The second way to sort the tree is to perform a sort procedure every time a cost coefficient is chosen. At a given tree level, a set of coefficients that are still feasible to the partial solution is created and sorted. Finding coefficients that are feasible is computationally much less demanding than checking if a particular coefficient is still feasible. The drawback with the second method is the high number of sorting procedures that need to be performed. For our test problems, we have chosen to implement the first approach, which is to perform a single initial sorting of the coefficients for each tree level. This choice was made because the first method performed best in practice for the instances we tested.

3.3.3 Local Search

The local search procedure improves upon the best-known solution by searching within a predefined neighborhood of the current solution to see if a better solution can be found. If an improvement is found, this solution is then stored as the current solution and a new neighborhood is searched. When no better solution can be found, the search is terminated and a local minimum is returned.

Because an optimal solution in one neighborhood definition is not usually optimal in other neighborhoods, we implement a variable neighborhood approach. A description of this metaheuristic and its applications to different combinatorial optimization problems is given by Hansen and Mladenović [47]. Variable neighborhood works by exploring multiple neighborhoods one at a time. For our branch and bound
algorithm, we implement the intrapermutation 2- and $n$-exchanges and the interpermutation 2-exchange presented by Pasiliao [84]. Starting from an initial solution, we define and search the first neighborhood to find a local minimum. From that local minimum, we redefine and search a new neighborhood to find an even better solution. The metaheuristic continues until all neighborhoods have been explored.

3.4 Computational Experiments

In this section, the computational experiments performed are explained. In the first subsection, we describe the experimental procedures employed. Then, in latter subsections, the results from the experiments are presented and discussed. The results include mean optimal costs and their standard deviation, for each type of problem and size. In the last subsection we present some interesting results, based on curve fitting models.

3.4.1 Experimental Procedures

The experimental procedures involved creating and exactly solving MAPs using the B&B algorithm described in the preceding section. There were at least 15 runs for each experiment where the number of runs was selected based on the practical amount of time to complete the experiment. Generally, as the size of the problem increased, the number of runs in the experiment had to be decreased. Also, as the dimension, $d$, of the MAP increased, the maximum number elements, $n$, decreased. Tables 3–2 and 3–3 provide a summary of the size of each experiment for the various types of problems.

The time taken by an experiment ranged from as low as a few seconds to as high as 20 hours on a 2.2 GHz Pentium 4 processor. We observed that problem instances with standard normal assignment costs took considerably longer time to solve; therefore, problem sizes and number of runs per experiment are smaller. The assignment costs $c_{i_1 \ldots i_d}$ for each problem instance were drawn from one of three distributions. The
Table 3–2: Number of runs for each experiment with uniform or exponential assignment costs.

<table>
<thead>
<tr>
<th>n \ d</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>2000</td>
</tr>
<tr>
<td>3</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>500</td>
<td>200</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>500</td>
<td>200</td>
<td>50</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>200</td>
<td>50</td>
<td>20</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>1000</td>
<td>1000</td>
<td>200</td>
<td>50</td>
<td>20</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>500</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1000</td>
<td>500</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>500</td>
<td>500</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The first distribution of assignment costs used was the uniform $U[0, 1]$. The next distribution used was the exponential with mean one, being determined by $c_{i_1 \cdots i_d} = -\ln U$. Finally, the third distribution used was the standard normal, $N(0, 1)$, with values determined by the polar method [63] as follows:

1. Generate $U_1$ and $U_2$, for $U_1, U_2 \sim U[0, 1]$.
2. Let $V_1 = 2U_1 - 1$, $V_2 = 2U_2 - 1$, and $W = V_1^2 + V_2^2$.
3. If $W > 1$, go back to 1, else $c_{i_1 \cdots i_d} = V_1 \sqrt{-\frac{2\ln W}{W}}$.

### 3.4.2 Mean Optimal Solution Costs

A summary of results for MAPs is provided in Tables 3–4, 3–5, and 3–6. We observe that in all cases the mean optimal cost gets smaller as the size of the MAP increases. Figure 3–2 shows the plots for problems with dimension $d = 3$, $d = 5$, $d = 7$ and $d = 10$, as examples for the exponential case (plots for the uniform case are similar). We observe that plots for higher dimensional problems converge to zero for smaller values of $n$. This is emphasized in the surface plot, Figure 3–3, of a subset of the data. Figure 3–4 shows plots for problems with the same number of dimensions
Table 3–3: Number of runs for each experiment standard normal assignment costs.

<table>
<thead>
<tr>
<th>n \ d</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
</tr>
<tr>
<td>3</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>50</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>200</td>
<td>20</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1000</td>
<td>1000</td>
<td>500</td>
<td>100</td>
<td>50</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1000</td>
<td>1000</td>
<td>50</td>
<td>20</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>1000</td>
<td>20</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1000</td>
<td>1000</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1000</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>500</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

as the problems in Figure 3–2, but for the standard normal case. Different from the uniform and exponential cases, the mean optimal solution costs appear to approach $-\infty$ with increasing $n$.

![Figure 3–2: Plots of mean optimal costs for four different sized MAPs with exponential assignment costs.](image)

We observe that in the uniform and exponential cases the standard deviation of optimal costs converges to zero as the size of the MAP gets larger. Clearly, this just confirms the asymptotic characteristic of the results. However, a trend is difficult to
Table 3–4: Mean optimal costs for different sizes of MAPs with independent assignment costs that are uniform in $[0,1]$.

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.584509</td>
<td>0.41955</td>
<td>0.308078</td>
<td>0.21852</td>
<td>0.15988</td>
<td>0.112458</td>
</tr>
<tr>
<td>3</td>
<td>0.54078</td>
<td>0.295575</td>
<td>0.155189</td>
<td>0.085318</td>
<td>0.0455428</td>
<td>0.024862</td>
</tr>
<tr>
<td>4</td>
<td>0.480825</td>
<td>0.209272</td>
<td>0.0884739</td>
<td>0.0386061</td>
<td>0.0171049</td>
<td>0.0075508</td>
</tr>
<tr>
<td>5</td>
<td>0.41716</td>
<td>0.151543</td>
<td>0.0549055</td>
<td>0.019867</td>
<td>0.00736</td>
<td>0.002596</td>
</tr>
<tr>
<td>6</td>
<td>0.374046</td>
<td>0.114551</td>
<td>0.0357428</td>
<td>0.011516</td>
<td>0.003542</td>
<td>0.000994</td>
</tr>
<tr>
<td>7</td>
<td>0.334805</td>
<td>0.0897928</td>
<td>0.02465</td>
<td>0.0067875</td>
<td>0.001724</td>
<td>0.00002</td>
</tr>
<tr>
<td>8</td>
<td>0.30229</td>
<td>0.0724017</td>
<td>0.0175195</td>
<td>0.004348</td>
<td>0.000815</td>
<td>0.000013</td>
</tr>
<tr>
<td>9</td>
<td>0.277139</td>
<td>0.0587219</td>
<td>0.012982</td>
<td>0.00253</td>
<td>0.0002733</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.252156</td>
<td>0.0486118</td>
<td>0.00961</td>
<td>0.00168</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.15988</td>
<td>0.112458</td>
</tr>
<tr>
<td>3</td>
<td>0.0455428</td>
<td>0.024862</td>
</tr>
<tr>
<td>4</td>
<td>0.0171049</td>
<td>0.0075508</td>
</tr>
<tr>
<td>5</td>
<td>0.00736</td>
<td>0.002596</td>
</tr>
<tr>
<td>6</td>
<td>0.003542</td>
<td>0.000994</td>
</tr>
<tr>
<td>7</td>
<td>0.001724</td>
<td>0.00002</td>
</tr>
<tr>
<td>8</td>
<td>0.000815</td>
<td>0.000013</td>
</tr>
<tr>
<td>9</td>
<td>0.0002733</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00002</td>
<td></td>
</tr>
</tbody>
</table>

Table 3–5: Mean optimal costs for different sizes of MAPs with independent assignment costs that are exponential with mean 1.

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.8046875</td>
<td>0.532907</td>
<td>0.353981</td>
<td>0.241978</td>
<td>0.167296</td>
<td>0.100279</td>
</tr>
<tr>
<td>3</td>
<td>0.63959</td>
<td>0.319188</td>
<td>0.165122</td>
<td>0.0829287</td>
<td>0.046346</td>
<td>0.0251884</td>
</tr>
<tr>
<td>4</td>
<td>0.531126</td>
<td>0.212984</td>
<td>0.0903552</td>
<td>0.0391171</td>
<td>0.0170434</td>
<td>0.0077126</td>
</tr>
<tr>
<td>5</td>
<td>0.45308</td>
<td>0.155833</td>
<td>0.0548337</td>
<td>0.020256</td>
<td>0.007272</td>
<td>0.0026945</td>
</tr>
<tr>
<td>6</td>
<td>0.39676</td>
<td>0.116469</td>
<td>0.0355034</td>
<td>0.0116512</td>
<td>0.003491</td>
<td>0.000944</td>
</tr>
<tr>
<td>7</td>
<td>0.34943</td>
<td>0.0909123</td>
<td>0.0251856</td>
<td>0.0068995</td>
<td>0.001482</td>
<td>0.000325</td>
</tr>
<tr>
<td>8</td>
<td>0.310489</td>
<td>0.0723551</td>
<td>0.0175745</td>
<td>0.0044</td>
<td>0.00066</td>
<td>0.00013</td>
</tr>
<tr>
<td>9</td>
<td>0.28393</td>
<td>0.0595148</td>
<td>0.01233</td>
<td>0.002665</td>
<td>0.0001667</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.263487</td>
<td>0.049335</td>
<td>0.009165</td>
<td>0.0019</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.167296</td>
<td>0.100279</td>
</tr>
<tr>
<td>3</td>
<td>0.046346</td>
<td>0.0251884</td>
</tr>
<tr>
<td>4</td>
<td>0.0170434</td>
<td>0.0077126</td>
</tr>
<tr>
<td>5</td>
<td>0.007272</td>
<td>0.0026945</td>
</tr>
<tr>
<td>6</td>
<td>0.003491</td>
<td>0.000944</td>
</tr>
<tr>
<td>7</td>
<td>0.001482</td>
<td>0.000325</td>
</tr>
<tr>
<td>8</td>
<td>0.00066</td>
<td>0.00013</td>
</tr>
<tr>
<td>9</td>
<td>0.0001667</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00002</td>
<td></td>
</tr>
</tbody>
</table>

Table 3–6: Mean optimal costs for different sizes of MAPs with independent assignment costs that are standard normal.

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-1.52566</td>
<td>-2.04115</td>
<td>-2.46001</td>
<td>-2.91444</td>
<td>-3.29715</td>
<td>-3.68093</td>
</tr>
<tr>
<td>3</td>
<td>-3.41537</td>
<td>-4.59134</td>
<td>-5.57906</td>
<td>-6.44952</td>
<td>-7.28343</td>
<td>-7.91587</td>
</tr>
<tr>
<td>4</td>
<td>-5.6486</td>
<td>-7.52175</td>
<td>-9.05299</td>
<td>-10.3701</td>
<td>-11.5257</td>
<td>-12.5916</td>
</tr>
<tr>
<td>5</td>
<td>-8.00522</td>
<td>-10.6145</td>
<td>-12.6924</td>
<td>-14.5221</td>
<td>-16.128</td>
<td>-17.4676</td>
</tr>
<tr>
<td>7</td>
<td>-13.2918</td>
<td>-17.2931</td>
<td>-20.6462</td>
<td>-23.4246</td>
<td>-25.8241</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-3.29715</td>
<td>-3.68093</td>
</tr>
<tr>
<td>3</td>
<td>-7.22834</td>
<td>-7.91587</td>
</tr>
<tr>
<td>4</td>
<td>-11.5257</td>
<td>-12.5916</td>
</tr>
<tr>
<td>5</td>
<td>-16.128</td>
<td>-17.4676</td>
</tr>
<tr>
<td>6</td>
<td>-20.9121</td>
<td>-22.7178</td>
</tr>
<tr>
<td>7</td>
<td>-25.8241</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-28.1166</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-28.7188</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00003</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-4.04804</td>
<td>-4.29477</td>
</tr>
<tr>
<td>3</td>
<td>-8.57385</td>
<td>-9.14707</td>
</tr>
<tr>
<td>4</td>
<td>-13.5045</td>
<td>-14.4328</td>
</tr>
<tr>
<td>5</td>
<td>-18.8873</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.00003</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.00002</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.00013</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.0001667</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00002</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3-3: Surface plots of mean optimal costs for $3 \leq d \leq 10$ and $2 \leq n \leq 10$ sized MAPs with exponential assignment costs.

Figure 3-4: Plots of mean optimal costs for four different sized MAPs with standard normal assignment costs.
detect for standard deviation of optimal costs in the standard normal case. Figure 3–5 shows the plots for the three, five, seven and ten dimensional problems, as examples, for the exponential case (plots for the uniform case are similar).

![Figure 3–5: Plots of standard deviation of mean optimal costs for four different sized MAPs with exponential assignment costs.](image)

Figure 3–5: Plots of standard deviation of mean optimal costs for four different sized MAPs with exponential assignment costs.

No clear trend is given in Figure 3–6 which shows the plots for the same dimensional problems but for the standard normal case.

### 3.4.3 Curve Fitting

Curve fits for the mean optimal solution costs were performed for the three types of problems using a least squares approach. The solver tool in Microsoft’s Excel was used to minimize the sum of squares. Several nonlinear models were tested for the purpose of developing a model to estimate the mean optimal cost, $\zeta^*_e$. The tested models include the following:

- **Power Fit**, $\zeta^*_e = An^B$
- **Shifted Power Fit**, $\zeta^*_e = A(n + B)^C$
- **Scaled Power Fit**, $\zeta^*_e = (An + B)^C$
Figure 3–6: Plots of standard deviation of mean optimal costs for four different sized MAPs with standard normal assignment costs.

- Exponential, $z_e^* = Ae^{Bn}$
- Reciprocal Quadratic, $z_e^* = A + Bn + Cn^2$

In each case the fit was calculated by fixing $d$ and varying $n$. For the uniform and exponential problems the Scaled Power Fit was found to be the best model. For the standard normal problems the Shifted Power Fit was used. The results of curve fitting are shown in Tables 3–7, 3–8, and 3–9. We observe that curves fit surprisingly well to the collected data. Figure 3–7 is a plot of the curve fitting model and observed data for the exponential case where $d = 3$. Note that the curves are nearly indistinguishable. This is typical for most problem sizes. A closer analysis of the curve fitting parameters for both uniform and exponential type problems indicates that as the dimension of the MAP increases, the curve fitting parameter $C$ approaches $(d - 2)$. A heuristic argument of why this is so is given in the following.

Consider the case of uniformly distributed cost coefficients. For each level of the index tree representation of the MAP, the expected value of the minimum order
Table 3–7: Curve fitting results for fitting the form \((An + B)^C\) to the mean optimal costs for MAPs with uniform assignment costs.

<table>
<thead>
<tr>
<th>(d)</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>Sum of Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.102</td>
<td>1.133</td>
<td>-1.764</td>
<td>8.80E-04</td>
</tr>
<tr>
<td>4</td>
<td>0.183</td>
<td>0.977</td>
<td>-2.932</td>
<td>7.74E-05</td>
</tr>
<tr>
<td>5</td>
<td>0.319</td>
<td>0.782</td>
<td>-3.359</td>
<td>8.28E-07</td>
</tr>
<tr>
<td>6</td>
<td>0.300</td>
<td>0.776</td>
<td>-4.773</td>
<td>5.77E-07</td>
</tr>
<tr>
<td>7</td>
<td>0.408</td>
<td>0.627</td>
<td>-4.997</td>
<td>6.28E-07</td>
</tr>
<tr>
<td>8</td>
<td>0.408</td>
<td>0.621</td>
<td>-6.000</td>
<td>7.91E-07</td>
</tr>
<tr>
<td>9</td>
<td>0.408</td>
<td>0.621</td>
<td>-7.000</td>
<td>3.44E-07</td>
</tr>
<tr>
<td>10</td>
<td>0.408</td>
<td>0.621</td>
<td>-8.000</td>
<td>9.50E-07</td>
</tr>
</tbody>
</table>

Table 3–8: Curve fitting results for fitting the form \((An + B)^C\) to the mean optimal costs for MAPs with exponential assignment costs.

<table>
<thead>
<tr>
<th>(d)</th>
<th>(A)</th>
<th>(B)</th>
<th>(C)</th>
<th>Sum of Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.300</td>
<td>0.631</td>
<td>-1.045</td>
<td>5.26E-05</td>
</tr>
<tr>
<td>4</td>
<td>0.418</td>
<td>0.550</td>
<td>-1.930</td>
<td>1.07E-05</td>
</tr>
<tr>
<td>5</td>
<td>0.406</td>
<td>0.601</td>
<td>-3.009</td>
<td>2.40E-06</td>
</tr>
<tr>
<td>6</td>
<td>0.420</td>
<td>0.594</td>
<td>-3.942</td>
<td>8.39E-08</td>
</tr>
<tr>
<td>7</td>
<td>0.414</td>
<td>0.601</td>
<td>-5.001</td>
<td>9.42E-07</td>
</tr>
<tr>
<td>8</td>
<td>0.413</td>
<td>0.617</td>
<td>-5.999</td>
<td>9.45E-07</td>
</tr>
<tr>
<td>9</td>
<td>0.418</td>
<td>0.600</td>
<td>-7.000</td>
<td>1.94E-07</td>
</tr>
<tr>
<td>10</td>
<td>0.414</td>
<td>0.607</td>
<td>-8.000</td>
<td>6.68E-07</td>
</tr>
</tbody>
</table>

The statistic is given by \(E[X^{(1)}] = 1/(nd-1 + 1)\) as there are \(nd-1\) coefficients on each level of the tree. And as there is one coefficient from each of the \(n\) levels in a feasible solution we may expect \(\bar{z}^* = O(n \cdot n^{-(d-1)}) = O(n^{-d-2})\). The same argument can be made for the exponential case where \(E[X^{(1)}] = 1/n^{d-1}\).

Again using a least squares approach, if we rebuild the curve fitting models for the uniform and exponential cases by fixing \(C = 2 - d\), we find, as expected, the lower dimension models result in higher sum of squares. The worst fitting model is that of the uniform case with \(d = 3\). In this case the sum of squares increases from \(8.80E-04\) to \(3.32E-03\) and the difference in the model estimate and actual results for \(n = 3\) increases from 2.3% to 5%. Although we believe fixing \(C = 2 - d\) can provide adequate fitting models, in the remainder of this chapter we continue to use the more accurate models (where \(C\) is not fixed to \(C = 2 - d\)); however, it is obvious the higher dimension problems are unaffected.
Table 3–9: Curve fitting results for fitting the form $A(n + B)^C$ to the mean optimal costs for MAPs with standard normal assignment costs.

<table>
<thead>
<tr>
<th>$d$</th>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
<th>Sum of Squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>-1.453</td>
<td>-0.980</td>
<td>1.232</td>
<td>7.27E-02</td>
</tr>
<tr>
<td>4</td>
<td>-1.976</td>
<td>-0.986</td>
<td>1.211</td>
<td>1.54E-01</td>
</tr>
<tr>
<td>5</td>
<td>-2.580</td>
<td>-1.053</td>
<td>1.164</td>
<td>2.85E-02</td>
</tr>
<tr>
<td>6</td>
<td>-2.662</td>
<td>-0.915</td>
<td>1.204</td>
<td>1.68E-02</td>
</tr>
<tr>
<td>7</td>
<td>-3.124</td>
<td>-0.956</td>
<td>1.174</td>
<td>1.20E-03</td>
</tr>
<tr>
<td>8</td>
<td>-3.230</td>
<td>-0.882</td>
<td>1.194</td>
<td>3.13E-03</td>
</tr>
<tr>
<td>9</td>
<td>-3.307</td>
<td>-0.819</td>
<td>1.218</td>
<td>1.71E-03</td>
</tr>
<tr>
<td>10</td>
<td>-3.734</td>
<td>-0.874</td>
<td>1.187</td>
<td>1.52E-04</td>
</tr>
</tbody>
</table>

Figure 3–7: Three dimensional MAP with exponential assignment costs. Plot includes both observed mean optimal cost values and fitted values. The two lines are nearly indistinguishable.

An obvious question to ask is what happens with variations of the distribution parameters. For example, what is the numerical estimation of $\bar{z}^*$ when the cost coefficients are distributed as uniform on $[a, b]$ or exponential with mean $\lambda$? We propose without proof the following numerical models to estimate $\bar{z}^*$.

For cost coefficients that are uniform on $[a, b]$, the curve fit or numerical estimation is $\bar{z}^* \approx \bar{z}^*_c = a n + (b - a)(An + B)^C$, using the curve fit parameters for the uniform case on $[0, 1]$ found in Table 3–7. For cost coefficients that are exponential with mean $\lambda$, the curve fit is $\bar{z}^* \approx \bar{z}^*_c = \lambda(An + B)^C$ using the curve fit parameters for the exponential case with $\lambda = 1$ found in Table 3–8.
Table 3–10: Estimated and actual mean optimal costs from ten runs for variously sized MAPs developed from different distributions. Included are the average difference and largest difference between estimated mean optimal cost and optimal cost.

<table>
<thead>
<tr>
<th>d</th>
<th>n</th>
<th>Distribution with Parameters</th>
<th>$\bar{z}^*$</th>
<th>$\bar{z}$</th>
<th>Ave $\Delta$</th>
<th>Max $\Delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>12</td>
<td>Uniform on [5,10]</td>
<td>61.1</td>
<td>61.1</td>
<td>0.143</td>
<td>0.428</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>Expo, $\lambda = 3$</td>
<td>0.415</td>
<td>0.404</td>
<td>0.0618</td>
<td>0.154</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>$N(\mu = 0, \sigma = 3)$</td>
<td>-86.4</td>
<td>-86.5</td>
<td>1.62</td>
<td>3.48</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>Uniform on [-1,1]</td>
<td>-12</td>
<td>-12</td>
<td>1.65E-03</td>
<td>3.16E-03</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>$N(\mu = 5, \sigma = 2)$</td>
<td>-7.24</td>
<td>-7.27</td>
<td>0.448</td>
<td>0.73</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Expo, $\lambda = 10$</td>
<td>1.90E-02</td>
<td>1.95E-02</td>
<td>2.62E-03</td>
<td>5.47E-03</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Uniform on [10,20]</td>
<td>60</td>
<td>60</td>
<td>0.003</td>
<td>0.008</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>Expo, $\lambda = 1.5$</td>
<td>4.13E-04</td>
<td>3.07E-04</td>
<td>1.15E-04</td>
<td>2.30E-04</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>$N(\mu = -5, \sigma = 2)$</td>
<td>-62.8</td>
<td>-63.2</td>
<td>0.944</td>
<td>2.26</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>Uniform on [-10,-5]</td>
<td>-70</td>
<td>-70</td>
<td>3.60E-04</td>
<td>6.70E-04</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>$N(\mu = 1, \sigma = 4)$</td>
<td>-53.8</td>
<td>-53.3</td>
<td>0.831</td>
<td>2.12</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>Expo, $\lambda = 2$</td>
<td>7.57E-04</td>
<td>8.00E-04</td>
<td>1.10E-04</td>
<td>4.03E-04</td>
</tr>
</tbody>
</table>

The situation is just a bit more involved for the normal case. Consider when the mean of the standard normal is changed from 0 by an amount $\mu$ and the standard deviation is changed by a factor $\sigma$. That is, the cost coefficients have the distribution $N(\mu, \sigma)$. Then $\bar{z}^* \approx \bar{z}^*_e = n\mu + \sigma A(n + B)^C$ using the curve fit parameters found in Table 3–9.

To assist in validating the numerical estimation models discussed above, experiments were conducted to compare the numerical estimates of the mean optimal costs and results of solved problems. The experiments created ten instances of different problem sizes and of different distributions and solved them to optimality. A variety of parameters were used for each distribution in an effort to exercise the estimation models. In the first experiment, we report mean optimal solution, estimated mean optimal solution, the max $\Delta$, and mean $\Delta$ where $\Delta = |\bar{z}^*_e - z(I)|$. That is, $\Delta$ for a problem instance is the difference between the predicted or estimated mean optimal cost and the actual optimal cost. Results of these experiments are provided in Table 3–10. We observe that the numerical estimates of the mean optimal costs are quite close to actual results.
Similar to Figure 3–7, Figures 3–8, 3–9 and 3–10 have plotted results of $\bar{z}^*$ and $\bar{z}_e^*$ (fitted data) for random instances of different sized problems. As in the above experiments, the number of runs is limited to ten for each problem size. As the plots of $\bar{z}^*$ and $\bar{z}_e^*$ are close to each other, this further validates the numerical models for estimating $\bar{z}^*$.

Figure 3–8: Plots of fitted and mean optimal costs from ten runs of variously sized MAPs developed from the uniform distribution on [10, 20]. Note that the observed data and fitted data are nearly indistinguishable.

Figure 3–9: Plots of fitted and mean optimal costs from ten runs of variously sized MAPs developed from the exponential distribution with mean three.

3.5 Algorithm Improvement Using Numerical Models

The numerical estimates of the mean optimal cost can be used to accurately predict the optimal solution cost of a random instance of an MAP that is constructed from a uniform, exponential or normal distribution. However, we still lack a solution.
In this section, we investigate whether the numerical estimates can be used to improve a branch and bound (B&B) exact solution method.

### 3.5.1 Improvement of B&B

The B&B solution method under consideration is that described in this chapter, Section 3.3. Recall that the B&B performs best by establishment of a tight upper bound early in the process. A tight upper bound allows significant pruning of the branches of the search tree. We consider the use of the numerical estimates to set tighter upper bounds than would be available through other primal heuristics. An advantage of the primal heuristic is, of course, a solution is at hand; whereas, the numerical estimate is a bound only with no solution. The heuristic used in Section 3.3 randomly selects a starting solution and then performs a variable local neighborhood search to find a local minimum. Alternatively, we also consider the global greedy and a variation of the maximum regret approaches as suggested by Balas and Saltzman [10]. In the global greedy approach, a starting solution is constructed step-by-step by selecting the smallest feasible cost coefficient then a variable local neighborhood search is applied to find a local minimum. For maximum regret, a feasible solution is constructed as follows. The difference between the two smallest feasible costs associated with each level of the index tree is calculated. This difference is called the *regret* as it represents the penalty for not choosing the smallest cost in the row.
Table 3–11: Results showing comparisons between three primal heuristics and the numerical estimate of optimal cost for several problem sizes and types. Shown are the average feasible solution costs from 50 runs of each primal heuristic on random instances.

<table>
<thead>
<tr>
<th>d</th>
<th>n</th>
<th>Distribution with Parameters</th>
<th>Random</th>
<th>Greedy</th>
<th>Max Regret</th>
<th>Numerical Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>10</td>
<td>Uniform on [0,1]</td>
<td>0.320</td>
<td>0.218</td>
<td>0.214</td>
<td>0.00341</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Uniform on [0,1]</td>
<td>0.433</td>
<td>0.201</td>
<td>0.182</td>
<td>0.00195</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Uniform on [0,1]</td>
<td>0.429</td>
<td>0.186</td>
<td>0.168</td>
<td>0.0019</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>Uniform on [0,1]</td>
<td>0.320</td>
<td>0.218</td>
<td>0.214</td>
<td>0.00341</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>Uniform on [0,1]</td>
<td>0.283</td>
<td>0.219</td>
<td>0.216</td>
<td>0.00152</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>Expo, $\lambda = 1$</td>
<td>0.611</td>
<td>0.226</td>
<td>0.2426</td>
<td>0.00251</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Expo, $\lambda = 1$</td>
<td>0.490</td>
<td>0.244</td>
<td>0.216</td>
<td>0.00190</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Expo, $\lambda = 1$</td>
<td>0.430</td>
<td>0.217</td>
<td>0.175</td>
<td>0.00114</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>Expo, $\lambda = 1$</td>
<td>0.385</td>
<td>0.267</td>
<td>0.270</td>
<td>0.00318</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>Expo, $\lambda = 1$</td>
<td>0.320</td>
<td>0.224</td>
<td>0.215</td>
<td>0.00145</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>-12.91</td>
<td>-21.29</td>
<td>-21.57</td>
<td>-23.40</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>-12.91</td>
<td>-18.51</td>
<td>-18.97</td>
<td>-20.89</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>-8.99</td>
<td>-15.77</td>
<td>-16.08</td>
<td>-17.51</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>-6.99</td>
<td>-11.67</td>
<td>-11.883</td>
<td>-13.53</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>-7.00</td>
<td>-12.60</td>
<td>-12.67</td>
<td>-14.44</td>
</tr>
</tbody>
</table>

The smallest feasible cost in the row with the largest regret is selected. This differs from the approach by Balas and Saltzman [10] where they consider every row in the multi-dimensional cost matrix, whereas we consider only the $n$ rows in the index tree. We took this approach as a trade-off between complexity and quality of the starting solution. Table 3–11 provides a comparison of starting solution cost values for the three primal heuristics described above along with a comparison of the numerical estimate of the optimal cost for various problem sizes and distribution types. The table shows the results of the average of 50 random generated instances.

In terms of an upper bound, the results of Table 3–11 indicate that, generally, the greedy primal heuristic is better than the random heuristic and max regret is better than greedy. For the uniform and exponential cases, the numerical estimate of optimal costs is clearly smaller than any of the results of the heuristics. In the normal cases, the numerical estimate is not significantly smaller. For the uniform and exponential cases, it appears much is to be gained by somehow incorporating the numerical estimate into an upper bound.
We propose using a factor \( \tau > 1 \) of the numerical estimate as the upper bound. If a feasible solution is found, the new solution serves as the upper bound. If a feasible solution is not found, then the estimated upper bound is incremented upwards until a feasible solution is found. This process guarantees an optimal solution will be found.

Figure 3–11 is fundamentally the same as Figure 3–1 except for the outside loop which increments the estimated upper bound upward until a feasible solution is found.

```plaintext
procedure IndexBB(L)
  1  solution_found = false
  2  while solution_found = false do
  3    \( \bar{z}^* = \bar{z}^* \cdot \tau \)
  4    for i = 1,\ldots,n do \( k_i \leftarrow 0 \)
  5    S \leftarrow \emptyset
  6    i \leftarrow 1
  7    while i > 0 do
  8      if \( k_i = |L_i| \) then
  9        S \leftarrow S \setminus \{s_i\}
 10       \( k_i \leftarrow 0 \)
 11      i \leftarrow i - 1
 12    else
 13      \( k_i = k_i + 1 \)
 14      if Feasible(S, L_i,k_i) then
 15        S \leftarrow S \cup L_i,k_i
 16      if LB(S) < \( \bar{z}^* \) then
 17        if i = n then
 18          \( \bar{S} \leftarrow S \)
 19          \( \bar{z} \leftarrow \text{Objective}(S) \)
 20          solution_found = true
 21        else
 22          i \leftarrow i + 1
 23      else
 24        S \leftarrow S \setminus \{s_i\}
 25      end
 26    end
 27  end
 28  return(\( \bar{S}, \bar{z} \))
end IndexBB
```

Figure 3–11: Branch and bound on the index tree.

The trade-off which must be considered is if the upper bound is estimated too low and incremented upwards too slow, then it may take many iterations over the
index tree before a feasible solution is found. However, no benefit is gained by setting
the upper bound too high. We found through less-than-rigorous analysis that $\tau$ set to
a value such that the upper bound is incremented upward by one standard deviation
of the optimal cost (see Figures 3–5 and 3–6) is a nice compromise.

3.5.2 Comparison of B&B Implementations

Table 3–12 compares the performance of the B&B algorithm using the random
primal heuristic for a starting upper bound versus using the maximum regret heuris-
tic versus using a numerical estimate for the upper bound. The table shows the
average times to solution of five runs on random instances of various problem sizes
and distribution types. In the uniform and exponential cases, we observe that B&B
using maximum regret generally does slightly better than using a random starting
solution. We also observe the approach of using a numerically estimated upper bound
significantly outperforms the other approaches in solving problems with uniformly or
exponentially distributed costs. However, there is no clear difference between the
approaches when solving problems with normally distributed costs. This is explained
by the small differences in the starting upper bounds for each approach.

3.6 Remarks

In this chapter we presented experimental results for the asymptotic value of the
optimal solution for random instances of the MAP. The results lead to the following
conjectures which will be addressed in detail in Chapter 4.

Conjecture 3.3 Given a $d$-dimensional MAP with $n$ elements in each dimension,
if the $n^d$ cost coefficients are independent exponentially distributed random variables
with mean 1 or independent uniformly distributed random variables in $[0,1]$, $z^* \to 0$
as $n \to \infty$ or $d \to \infty$. 
Table 3–12: Average time to solution in seconds of solving each of five randomly generated problems of various sizes and types. The experiment involved using the B&B solution algorithm with different starting upper bounds developed in three different ways.

<table>
<thead>
<tr>
<th>$d$</th>
<th>$n$</th>
<th>Distribution with Parameters</th>
<th>Random</th>
<th>Max Regret</th>
<th>Numerical</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>10</td>
<td>Uniform on $[0,1]$</td>
<td>1305</td>
<td>1311</td>
<td>795</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Uniform on $[0,1]$</td>
<td>19.1</td>
<td>19.2</td>
<td>13.9</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Uniform on $[0,1]$</td>
<td>20.5</td>
<td>20.4</td>
<td>13.1</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>Uniform on $[0,1]$</td>
<td>0.3</td>
<td>0.29</td>
<td>0.13</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>Uniform on $[0,1]$</td>
<td>1.15</td>
<td>1.12</td>
<td>0.4</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>Expo, $\lambda = 1$</td>
<td>1279</td>
<td>1285</td>
<td>1201</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Expo, $\lambda = 1$</td>
<td>25.5</td>
<td>25.8</td>
<td>17.8</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>Expo, $\lambda = 1$</td>
<td>21.8</td>
<td>24.5</td>
<td>13.4</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>Expo, $\lambda = 1$</td>
<td>0.24</td>
<td>0.23</td>
<td>0.1</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>Expo, $\lambda = 1$</td>
<td>1.67</td>
<td>1.66</td>
<td>0.57</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>54.9</td>
<td>47.3</td>
<td>54.2</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>89.9</td>
<td>89.6</td>
<td>89.2</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>24.7</td>
<td>24.6</td>
<td>24.6</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>1.25</td>
<td>1.23</td>
<td>1.24</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>$N(\mu = 0, \sigma = 1)$</td>
<td>30.7</td>
<td>30.2</td>
<td>30.7</td>
</tr>
</tbody>
</table>

Conjecture 3.4 Given a $d$-dimensional MAP with $n$ elements in each dimension, if the $n^d$ cost coefficients are independent standard normal random variables, $z^* \to -\infty$ as $n \to \infty$ or $d \to \infty$.

We also presented in this chapter curve fitting results to accurately estimate the mean optimal costs of variously sized problems constructed with cost coefficients independently drawn from the uniform, exponential or normal distributions. Of high interest of course is how numerical estimates of mean optimal cost can be used to improve existing solution algorithms or is they can be used to find new solution algorithms. To this end, we have shown that using numerical estimates can significantly improve the performance of a B&B exact solution method.
CHAPTER 4
PROOFS OF ASYMPTOTIC CHARACTERISTICS OF THE MAP

4.1 Introduction

The experimental work detailed in Chapter 3 leads to conjectures concerning the asymptotic characteristics of the mean optimal costs of randomly generated instances of the MAP where costs are assigned independently to assignments. In this chapter, we provide proofs of more generalized instances of Conjecture 3.3 and prove Conjecture 3.4. The proofs are based on building an index tree to represent the cost coefficients of the MAP and then selecting a minimum subset of cost coefficients such that at least one feasible solution can be expected from this subset. Then an upper bound on the cost of this feasible solution is established and used to complete the proofs. Throughout this chapter we consider MAPs with \( n \) elements in each of the \( d \) dimensions.

Before presenting the theorems and their proofs concerning the asymptotic nature of these problems, we first consider a naive approach to establishing the asymptotic characteristics based on some greedy algorithms.

4.2 Greedy Algorithms

Consider the case of the MAP where cost coefficients are independent exponentially distributed random variables with mean 1. By Conjecture 3.3 the mean optimal costs are thought to go to zero with increasing problem size. Suppose we consider the solution from a greedy algorithm. As the solution serves as an upper bound to the optimal solution, we can try to prove the conjecture if we can show the mean of the sub-optimal solutions goes to zero with increasing problem size. However, as will be shown this is difficult with two common greedy algorithms.
4.2.1 Greedy Algorithm 1

The first algorithm that we consider uses the index tree data structure proposed by Pierskalla [87] to represent the cost coefficients of the MAP. There are $n$ levels in the index tree with $n^{d-1}$ nodes on each level for a total $n^d$ nodes. Each level of the index tree has the same value in the first index. A feasible solution can be constructed by first starting at the top level of the tree. The partial solution is developed by moving down the tree one level at a time and adding a node that is feasible with the partial solution. The number of nodes at level $i$ that are feasible to a partial solution developed from levels $1, 2, \ldots, i-1$ is $(n-i+1)^{d-1}$. A complete feasible solution is obtained upon reaching the bottom or $n^{th}$-level of the tree.

The proposed greedy algorithm is as follows:

**Input** MAP of dimension $d$ and $n$ elements in each dimension in the form of an index tree.

**Build** a partial solution, $S_i, i = 1$, by choosing the smallest cost coefficient from row 1 of the tree.

**For** $i = 2, \ldots, n$, continue to construct a solution by choosing the smallest cost coefficient in row $i$ of the tree that is feasible with $S_{i-1}$ constructed from rows $1, \ldots, i-1$.

We wish to calculate the expected solution cost from this algorithm for the MAP constructed from i.i.d. exponential random variables with mean 1. Let the mean solution cost resulting from the algorithm be represented by $\bar{z}^*_u$. Suppose that $X_1, X_2, \ldots, X_k$ are $k$ i.i.d. exponential random variables with mean 1. Let $X_{(i)}$ be the $i$th smallest of these. Applying order statistics [33], we have the following expression for the expected minimum value of $k$ independent identically distributed random variables: $E[X_{(1)}] = 1/k$.

We may now construct a feasible solution using the above greedy algorithm. We do so by recalling that the number of nodes that are feasible at level $i + 1$ to a partial
solution developed down to level $i$, for $i = 1, 2, \ldots, n$ is $(n-i)^{d-1}$. Considering this and the fact that cost coefficients are independent, the expected solution cost of $S_1$ is $\frac{1}{n^{d-1}}$, the expected solution cost of $S_2$ is $\frac{1}{n^{d-1}} + \frac{1}{(n-1)^{d-1}}$ and so forth. Therefore, we find

$$\bar{z}^* = \sum_{i=0}^{n-1} \frac{1}{(n-i)^{d-1}}$$

(4.1)

$$> 1,$$  

(4.2)

where equation (4.2) holds because the $n$-th term of equation (4.1) is one.

Since $\bar{z}^* > 0$, we conclude this greedy approach cannot be used to prove Conjecture 3.3. However, maybe a more global approach will work.

4.2.2 Greedy Algorithm 2

The following algorithm is described by Balas and Saltzman [10] as the GREEDY heuristic. The algorithm is as follows:

**Input** MAP of dimension $d$ and $n$ elements in each dimension as matrix $A$.

**For** $i = 1, \ldots, n$, construct the partial solution $S_i$ by choosing the smallest element in matrix $A$ and then exclude the $d$ rows covered by this element.

Using this covering approach, we see the number of nodes that are feasible to a partial solution developed up to iteration $i$, for $i = 1, 2, \ldots, n$ is $(n-i)^d$. For example, all $n^d$ cost coefficients are considered in the first iteration. The next iteration has $(n-1)^d$ nodes for consideration. The expected solution cost of $S_1$ is $1/n^d$. The expected solution cost of $S_2$ is $1/n^d + 1/n^d + 1/(n-1)^d$. The extra $1/n^d$ appears in the expression because, in general, the expected minimum value of the uncovered nodes is at least as much as the expected minimum value found in the previous iteration. We could now develop the expression for $\bar{z}^*$; however, we note that the algorithm’s last iteration will consider only one cost coefficient. Therefore, again, we have the result that $\bar{z}^* > 1$ when using this algorithm.
We conclude that these simple greedy approaches cannot be used to prove the conjectures concerning the asymptotic characteristics of the MAP. In the next sections, we resort to a novel probabilistic approach.

4.3 Mean Optimal Costs of Exponentially and Uniformly Distributed Random MAPs

To find the asymptotic cost when the costs are uniformly or exponentially distributed, we use an argument based on the probabilistic method \[7\]. Basically, we show that, for a subset of the index tree, the expected value of the number of feasible paths in this subset is at least one. Thus, such a set must contain a feasible path and this fact can be used to give an upper bound on the cost of the optimum solution. This is explained in the next proposition.

**Proposition 4.1** Using an index tree to represent the cost coefficients of the MAP, randomly select $\alpha$ different nodes from each level of the tree and combine these nodes from each level into set $\mathcal{A}$. $\mathcal{A}$ is expected to produce at least one feasible solution to the MAP when

$$\alpha = \left\lceil \frac{n^{d-1}}{(n!)^{d-1}} \right\rceil \quad \text{and} \quad |\mathcal{A}| = n\alpha. \quad (4.3)$$

**Proof:** Consider there are $n^{d-1}$ cost coefficients on each of the $n$ levels of the index tree representation of an MAP of dimension $d$ and with $n$ elements in each dimension. Now consider there are $(n^{d-1})^n$ paths (not necessarily feasible to the MAP) in the index tree from the top level to the bottom level. The number of feasible paths (or feasible solutions to the MAP) in the index tree is $(n!)^{d-1}$. Therefore, the proportion $\varrho$ of feasible paths to all paths in the entire index tree is

$$\varrho = \frac{(n!)^{d-1}}{(n^{d-1})^n}. \quad (4.4)$$

Create a set $\mathcal{A}$ of nodes to represent a reduced index tree by selecting $\alpha$ nodes randomly from each level of the overall index tree and placing them on a corresponding
level in the reduced index tree. The number of nodes in \( \mathcal{A} \) is obviously \( n\alpha \). For this reduced index tree of \( \mathcal{A} \), there are \( \alpha^n \) paths (not necessarily feasible to the MAP) from the top level to the bottom level. Since the set of nodes in \( \mathcal{A} \) were selected randomly, we may now use \( \varrho \) to determine the expected number of feasible paths in \( \mathcal{A} \) by simply multiplying \( \varrho \) by the number of all paths in the reduced tree of \( \mathcal{A} \). That is

\[
E[\text{number feasible paths in } \mathcal{A}] = \varrho \alpha^n.
\]

We wish to ensure that the expected number of feasible paths \( \mathcal{A} \) is at least one. Thus, over all possible choices of the \( n \) subsets of \( \alpha \) elements, there must be one choice such that there is one feasible path (in fact there may be many since the expected value gives only the average over all possible solutions). Therefore,

\[
\varrho \alpha^n \geq 1,
\]

which results

\[
\alpha \geq \left( \frac{1}{\varrho} \right)^\frac{1}{n}.
\]

Incorporating the value of \( \varrho \) from (4.4) we get

\[
\alpha \geq \frac{n^{d-1}}{(n!)^{\frac{d-1}{n}}}.
\]

Therefore, since \( \alpha \) must be an integer, we get (4.3).

We now take a moment to discuss the concept of order statistics. For more complete information, refer to statistics books such as by David [33]. Suppose that \( X_1, X_2, \ldots, X_k \) are \( k \) independent identically distributed variables. Let \( X_{(i)} \) be the \( i \)-th smallest of these. Then \( X_{(i)} \) is called the \( i \)-th order statistic for the set \( \{X_1, X_2, \ldots, X_k\} \).
In the rest of the section, we will consider bounds for the value of the $\alpha$-th order statistic of i.i.d. variables drawn from a random distribution. This value will be used to derive an upper bound on the cost of the optimal solution for random instances, when $n$ or $d$ increases. Note that, in some places (e.g., Equation (4.6)), we assume that $\alpha = n^{d-1}/n!^{d-1}/n!$. This is a good approximation in the following formulas because

(a) if $n$ is fixed and $d \to \infty$, then $\alpha \to \infty$, and therefore there is no difference between $\alpha$ and $n^{d-1}/n!^{d-1}/n!$;

(b) if $d$ is fixed and $n \to \infty$, then $\alpha \to e^{d-1}$. This is not difficult to derive, since

$$\frac{n}{n!^{\frac{1}{\pi}}} \approx \frac{n}{[(\frac{n}{e})^{n}(2\pi n)^{\frac{1}{2}}]^\frac{1}{\pi}} = \frac{e}{(2\pi n)^{\frac{1}{2\pi}}}.$$ 

But

$$(2\pi n)^{\frac{1}{2\pi}} = (2\pi e^{\log n})^{\frac{1}{2\pi}} = (2\pi)^{\frac{1}{2\pi}} \cdot e^{\frac{\log n}{2\pi}},$$

and both factors in the right have limit equal to 1. However, $e^{d-1}$ is a constant value, and will not change the limit of the whole formula, as $n \to \infty$.

**Proposition 4.2** Let $\bar{z}^*_u = nE[X_{(\alpha)}]$, where $E[X_{(\alpha)}]$ is the expected value of the $\alpha$-th order statistic for each level of the index tree representation of the MAP. Then, $\bar{z}^*_u$ is an upper bound to the mean optimal solution cost of an instance of an MAP with independent identically distributed cost coefficients.

**Proof:** Consider any level $j$ of the index tree and select the $\alpha$ elements with lowest cost on that level. Let $A_j$ be the set composed by the selected elements. Since the cost coefficients are independent and identically distributed, the nodes in $A_j$ are randomly distributed across the level $j$. Now, pick the maximum node $v \in A_j$, i.e.,

$$v = \max\{w \mid w \in A_j\}.$$ 

The expected value of $v$ is the same as the expected value of the $\alpha$-th order statistic among $n^{d-1}$ cost values for this level of the tree. Since each level of the index tree
has the same number of independent and identically distributed cost values, we may
conclude that $E[X_{(\alpha)}]$ is the same for each level in the index tree. By randomly
selecting $\alpha$ cost values from each of the $n$ levels of the index tree, we expect to have
at least one feasible solution to the MAP by Proposition 4.1. Thus, it is clear that
an upper bound cost for the expected feasible solution is $\bar{z}_u^* = nE[X_{(\alpha)}]$.

**Theorem 4.3** Given a $d$-dimensional MAP with $n$ elements in each dimension, if
the $n^d$ cost coefficients are independent exponentially distributed random variables
with mean $\lambda > 0$, then $\bar{z}^* \to 0$ as $n \to \infty$ or $d \to \infty$.

**Proof:** We first note that for independent exponentially distributed variables the
expected value of the $\alpha^{th}$ order statistic for $k$ i.i.d. variables is given by

$$E[X_{(\alpha)}] = \sum_{j=0}^{\alpha-1} \frac{\lambda}{k-j}.$$  \hspace{1cm} (4.5)

Note that (4.5) has $\alpha$ terms and the term of largest magnitude is the last term. Using
the last term, an upper bound on (4.5) is developed as

$$E[X_{(\alpha)}] \leq \sum_{j=0}^{\alpha-1} \frac{\lambda}{k-(\alpha-1)} = \frac{\alpha\lambda}{k-\alpha+1}.$$  \hspace{1cm} (4.6)

Now, using Propositions 4.1 and 4.2, the upper bound for the mean optimal solution
to the MAP with exponential costs may be developed as

$$\bar{z}_u^* = n\frac{\alpha\lambda}{k-\alpha+1} \leq n\frac{\alpha\lambda}{k-\alpha} = \frac{n\lambda}{\frac{k}{\alpha-1}},$$

where $k = n^{d-1}$ is the number of cost elements on each level of the index tree. To
prove $\bar{z}_u^* \to 0$, we must first substitute the values of $k$ and $\alpha$ into (4.6), which gives

$$\bar{z}_u^* \leq \frac{n\lambda}{(n!)^{\frac{d-1}{n}} - 1}. \hspace{1cm} (4.6)$$
Let \( n = \gamma \) and \( n! = \delta \), where \( \gamma \) and \( \delta \) are some fixed numbers. Then (4.6) becomes

\[
\bar{z}_u^* \leq \frac{\gamma \lambda}{\delta^{\frac{d-1}{d}}} - 1 \approx \frac{\gamma \lambda}{\delta^{\frac{d-1}{d}}},
\]

as \( d \) gets large. Therefore,

\[
\lim_{d \to \infty} \bar{z}_u^* \leq \lim_{d \to \infty} \frac{\gamma \lambda}{\delta^{\frac{d-1}{d}}} = 0.
\]

Now, let \( d - 1 = \gamma \), where \( \gamma \) is some fixed number. Then (4.6) becomes

\[
\bar{z}_u^* = \frac{n \lambda}{(n!)^{\frac{1}{n}}} - 1 \approx \frac{n \lambda}{(n!)^{\frac{1}{n}}},
\]

as \( n \) gets large. Using Stirling’s approximation \( n! \approx (n/e)^n \sqrt{2\pi n} \),

\[
\frac{n \lambda}{(n!)^{\frac{1}{n}}} \approx \frac{n \lambda}{((n/e)^n \sqrt{2\pi n})^{\frac{1}{n}}}
= \frac{n \lambda}{((n/e)^\gamma (2\pi)^{\frac{1}{2\pi}})}
= \frac{n \lambda}{n^{(\gamma + \frac{1}{2\pi})} ((\frac{1}{e})^\gamma)}
\leq \frac{n \lambda}{n^{(\frac{2n\gamma + 1}{2\pi})} ((\frac{1}{e})^\gamma)} \quad (4.7)
\]

\[
= \frac{\lambda}{n^{(\frac{2n\gamma + 1}{2\pi})} ((\frac{1}{e})^\gamma)} \quad (4.8)
\]

where (4.7) holds because \( (2\pi)^{\frac{\gamma}{2\pi}} \) approaches one from the right as \( n \to \infty \). Considering that \( (\frac{1}{e})^\gamma \) is a constant and that the exponent to \( n \) is greater than one for any \( \gamma \geq 2 \), which holds because \( d \geq 3 \), then (4.8) will approach zero as \( n \to \infty \).

Therefore, for the exponential case

\[
\lim_{n \to \infty} \bar{z}_u^* = 0 \quad \text{and} \quad \lim_{d \to \infty} \bar{z}_u^* = 0 \quad \text{from above.}
\]

Note that \( \bar{z}^* \) is bounded from below by zero because the lower bound of any cost coefficient is zero (a characteristic of the exponential random variable with \( \lambda > 0 \)). Since \( 0 \leq \bar{z}^* \leq \bar{z}_u^* \), the proof is complete.
**Theorem 4.4** Given a $d$-dimensional MAP with $n$ elements in each dimension, if the $n^d$ cost coefficients are independent uniformly distributed random variables in $[0, 1]$, then $\bar{z}^* \to 0$ as $n \to \infty$ or $d \to \infty$.

**Proof:** For the case of the uniform variable in $[0, 1]$, the expected value of the $\alpha^{th}$ order statistic for $k$ i.i.d. variables is given by

$$E[X_{(\alpha)}] = \frac{\alpha}{k+1}.$$

Therefore, using Propositions 4.1 and 4.2, the upper bound on the mean optimal solution for an MAP with uniform costs in $[0, 1]$ is

$$\bar{z}_u^* = \frac{n\alpha}{k+1} \leq \frac{n\alpha}{k}, \quad (4.9)$$

where $k = n^{d-1}$ is the number of cost elements on each level of the index tree. We must now substitute the values of $k$ and $\alpha$ into (4.9), which becomes

$$\bar{z}_u^* \leq \frac{n}{(n!)^{\frac{d-1}{n}}} \quad (4.10)$$

Applying to (4.10) Stirling’s approximation, in the same way as used in Theorem 4.3, we see that $\bar{z}_u^* \to 0$ as $n \to \infty$ or $d \to \infty$. Note again that $\bar{z}^*$ is bounded from below by zero because the lower bound of any cost coefficient is zero (a characteristic of the uniform random variable in $[0, 1]$). Since $0 \leq \bar{z} \leq \bar{z}_u^*$, this completes the proof.

**Theorem 4.5** Given a $d$-dimensional MAP with $n$ elements in each dimension, for some fixed $n$, if the $n^d$ cost coefficients are independent, uniformly distributed random variables in $[a, b]$, then $\bar{z}^* \to na$ as $d \to \infty$.

**Proof:** For the case of the uniform variable in $[a, b]$, the expected value of the $\alpha^{th}$ order statistic for $k$ i.i.d. variables is given by David [33]

$$E[X_{(\alpha)}] = a + \frac{(b-a)\alpha}{k+1}.$$
Therefore, using Propositions 4.1 and 4.2, the upper bound on the mean optimal solution for an MAP with uniform costs in \([a, b]\) is

\[
\bar{z}_u^* = n\left(a + \frac{(b-a)\alpha}{k+1}\right) \leq n\left(a + \frac{(b-a)\alpha}{k}\right)
\]

\[
= na + \frac{(b-a)n\alpha}{k}, \quad (4.11)
\]

where \(k = n^{d-1}\) is the number of cost elements on each level of the index tree. We must now substitute values of \(k\) and \(\alpha\) into (4.11), which results

\[
\bar{z}_u^* \leq na + \frac{(b-a)n}{(n!)^{\frac{d-1}{n}}}. \quad (4.12)
\]

It becomes immediately obvious from (4.12) that for a fixed \(n\) and as \(d \to \infty\), \(\bar{z}_u^* \to n\alpha\). As \(\bar{z}^* \leq \bar{z}_u^*\) and \(na\) is an obvious lower bound for this instance of the MAP we conclude that, for fixed \(n\), \(\bar{z}^* \to na\) as \(d \to \infty\).

### 4.4 Mean Optimal Costs of Normal-Distributed Random MAPs

We want to now prove results similar to the theorems above, for the case where cost values are taken from a normal distribution. This will allow us to prove Conjecture 3.4. A bound on the cost of the optimal solution for normal distributed random MAPs can be found, using a technique similar to the one used in the previous section. However, in this case a reasonable bound is given by the median order statistics, as described in the proof of the following theorem.

**Theorem 4.6** Given a \(d\)-dimensional MAP, for a fixed \(d\), with \(n\) elements in each dimension, if the \(n^d\) cost coefficients are independent standard normal random variables, then \(\bar{z}^* \to -\infty\) as \(n \to \infty\).

**Proof:** First note that for odd \(k = 2r + 1\), \(X_{(r+1)}\) is the median order statistic and for even \(k = 2r\), we define the median as \(\frac{1}{2}(X_r + X_{(r+1)})\). Obviously, the expected value of the median in both cases is zero. Let \(k = n^{d-1}\) and note that, as \(n\) or \(d\) get large, \(\alpha \ll r\) for either odd or even case. Therefore we may immediately conclude
$E[X_{(\alpha)}] < 0$. Using Propositions 4.1 and 4.2, we see that $\bar{z}^{*} \leq \bar{z}_n^{*} = nE[X_{(\alpha)}]$ and $\bar{z}^{*} \to -\infty$ as $n \to \infty$.

**Theorem 4.7** Given a $d$-dimensional MAP with $n$ elements in each dimension, for a fixed $n$, if the $n^d$ cost coefficients are independent standard normal random variables, then $\bar{z}^{*} \to -\infty$ as $d \to \infty$.

**Proof:** We use the results from Cramér [32] to establish the expected value of the $i$th order statistic of $k$ independent standard normal variables. With $i \leq k/2$ we have

$$E[X_{(i)}] = -\sqrt{2 \log k} + \frac{\log(\log k)}{2\sqrt{2 \log k}} + \frac{\log(4\pi)}{2\sqrt{2 \log k}} + \frac{(S_1 - C)}{\sqrt{2 \log k}} - O\left(\frac{1}{\log k}\right), \quad (4.13)$$

where $S_1 = \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{i-1}$ and $C$ denotes Euler’s constant, $C \approx 0.57722$. As $d \to \infty, k \to \infty$ and the last term of (4.13) may be dropped. In addition, a slight rearrangement of (4.13) is useful:

$$E[X_{(i)}] \approx -\sqrt{2 \log k} + \frac{\log(\log k)}{2\sqrt{2 \log k}} + \frac{\log(4\pi)}{2\sqrt{2 \log k}} + \frac{(S_1 - C)}{\sqrt{2 \log k}}. \quad (4.14)$$

It is not difficult to see that as $k \to \infty$, the sum of the first three terms of (4.14) goes to $-\infty$. Therefore, we consider the last term of (4.14) as $k \to \infty$.

$$\frac{(S_1 - C)}{\sqrt{2 \log k}} = \frac{-C + \sum_{j=1}^{i-1} \frac{1}{j}}{\sqrt{2 \log k}} \approx \frac{-C + \int_1^{i-1} \frac{1}{x} \, dx}{\sqrt{2 \log k}} = \frac{\log(i-1) - C}{\sqrt{2 \log k}} = \frac{\log(i-1) - C}{\sqrt{2 \log k}}. \quad (4.15)$$
Noting that the second term of (4.15) goes to zero as $k \to \infty$, and also making the substitutions $i = \alpha = n^{d-1}/n!\frac{d-1}{n}$ and $k = n^{d-1}$, we have

$$\frac{(S_1 - C)}{\sqrt{2 \log k}} \leq \frac{\log \left( \frac{n^{d-1}}{(n!\frac{d-1}{n})} - 1 \right)}{\sqrt{2 \log n^{d-1}}} \leq \frac{\log \left( \frac{n^{d-1}}{(n!\frac{d-1}{n})} \right)}{\sqrt{2 \log n^{d-1}}}$$

$$= \frac{\log(n^{d-1}) - \log((n!\frac{d-1}{n}))}{\sqrt{2 \log n^{d-1}}}$$

$$= \frac{(d-1) \log(n) - (d-1) \log(n!\frac{1}{n})}{\sqrt{2 \log n^{d-1}}}.$$  \hspace{1cm} (4.16)

It is clear that for a fixed $n$, and as $d \to \infty$, the right hand side of (4.16) approaches zero. Therefore, using Propositions 4.1 and 4.2 we have $\bar{z}^* \leq \bar{z}_u^* = nE[X_{(\alpha)}]$ and $E[X_{(\alpha)}] \to -\infty$ for a fixed $n$ and $d \to \infty$. The proof is complete.

4.5 Remarks on Further Research

In this chapter, we proved some asymptotic characteristics of random instances of the MAP. This was accomplished using a probabilistic approach. An interesting direction of research is how the probabilistic approach can be used to improve the performance of existing solution algorithms. Chapter 5 applies the probabilistic approach to reduce the cardinality of the MAP which, in turn, is then solved by GRASP. We show this process can result in better solutions in less time for the data association problem in the multisensor multitarget tracking problem.
CHAPTER 5
PROBABILISTIC APPROACH TO SOLVING THE MULTISENSOR MULTITARGET TRACKING PROBLEM

5.1 Introduction

The data association problem arising from multisensor multitarget tracking (MSMTT) can be formulated as an MAP. Although the MAP is considered a hard problem, a probabilistic approach to reducing problem cardinality may be used to accelerate the convergence rate. With the use of MSMTT simulated data sets, we show that the data association problem can be solved faster and with higher quality solutions due to these exploitations.

The MSMTT problem is a generalization of the single sensor single target tracking problem. In the MSMTT problem noisy measurements are made from an arbitrary number of spatially diverse sensors (for example cooperating remote agents) regarding an arbitrary number of targets with the goal of estimating the state of all the targets present. See Figure 5–1 for visual representation of the problem. Because of noise, measurements are imperfect. The problem is exacerbated with many close targets and noisy measurements. Furthermore, the number of targets may change by moving into and out of detection range and there are instances of false detections as shown in Figure 5–2. The MSMTT solves a data association problem on the sensor measurements and estimates the current state of each target based on the data association problem for each sensor scan.

The combinatorial nature of the MSMTT problem results from the data association problem; that is, given $d$ sensors with $n$ target measurements each, how do we optimally partition the entire set of measurements so that each measurement is attributed to no more than one target and each sensor detects a target no more than
Once? The data association problem maximizes the likelihood that each measurement is assigned to the proper target. In MSMTT, a scan is made at discrete, periodic moments in time. In practical instances, the data association problem should be solved in real time - particularly in the case of cooperating agents searching for and identifying targets. Combining data from more than one sensor with the goal of improving decision-making is termed sensor fusion.

Solving even moderate-sized instances of the MAP has been a difficult task, since a linear increase in the number of dimensions (in this case, sensors) brings an
exponential increase in the size of the problem. As such, several heuristic algorithms [74, 90] have been applied to this problem. However, due to the size and complexity of the problem, even the heuristics struggle to achieve solutions in realtime. In this chapter we propose a systematic approach to reduce the size and complexity of the data association problem, yet achieve higher quality solutions in faster times.

This chapter is organized as follows. We first give some background on data association for the MSMTT problem. We then introduce a technique that may be used to reduce the size of the problem. Following that, we discuss the heuristic, Greedy Randomized Adaptive Search Procedure (GRASP), and how GRASP can be modified to work effectively on a sparse problem. Finally, we provide some comparative results of these solution methods.

5.2 Data Association Formulated as an MAP

Data association is formulated as an MAP where the cost coefficients are derived from a computationally expensive negative log-likelihood function. The data association problem for the MSMTT problem is to match sensor measurements in such a way that no measurement is matched more than once and overall matching is the most likely association of measurements to targets. In the MAP, elements from $d$ disjoint sets are matched in such a way that the total cost associated with all matchings is minimized. It is an extension of the two-dimensional assignment problem where there are only two disjoint sets. For sets of size $n$, the two-dimensional assignment problem has been demonstrated to be solvable in $O(n^3)$ arithmetic operations using the Hungarian method [62], for example. However, the three-dimensional assignment problem is a generalization of the three dimensional matching problem which is shown by Garey and Johnson [44] to be $NP$-hard.

A review of the multtarget multisensor problem formulation and algorithms is provided by Poore [89]. Bar-Shalom, Pattipati, and Yeddanapudi [11] also present a combined likelihood function in multisensor air traffic surveillance.
Suppose that we have $S$ sensors observing an unknown number of targets $T$. The Cartesian coordinates of sensor $s$ is known to be $\omega_s = [x_s, y_s, z_s]'$, while the unknown position of target $t$ is given by $\omega_t = [x_t, y_t, z_t]'$. Sensor $s$ takes $n_s$ measurements, $z_{s,i}$. Since the measurements of target locations are noisy, we have the following expression for measurement $i_s$ from sensor $s$:

$$z_{s,i_s} = \begin{cases} h_s(\omega_t, \omega_s) + \omega_{s,i_s} & \text{if measurement } i_s \text{ is produced by target } t \\ u_{s,i_s} & \text{if measurement } i_s \text{ is a false alarm} \end{cases}$$

The measurement noise, $\omega_{s,i_s}$, is assumed to be normally distributed with zero mean and covariance matrix $R_s$. The nonlinear transformation of measurements from the spherical to Cartesian frame is given by $h_s(\omega_t, \omega_s)$.

Consider the $S$-tuple of measurements $Z_{i_1,i_2,...,i_S}$, each element is produced by a different sensor. Using dummy measurements $z_{s,0}$ to make a complete assignment, the likelihood that each measurement originates from the same target $t$ located at $\omega_t$ is given.

$$\Lambda(Z_{i_1,i_2,...,i_S}|\omega_t) = \prod_{s=1}^{S} [P_{D_s} \cdot p(z_{s,i_s}|\omega_t)]^{\delta_{s,i_s}} [1 - P_{D_s}]^{1-\delta_{s,i_s}}, \quad (5.1)$$

where

$$\delta_{s,i_s} = \begin{cases} 0 & \text{if } i_s = 0 \text{ (dummy measurement)} \\ 1 & \text{if } i_s > 0 \text{ (actual measurement)} \end{cases}$$

and $P_{D_s} \leq 1$ is the the detection probability for sensor $m$. The likelihood that the set of measurements $Z_{i_1,i_2,...,i_S}$ corresponds to a false alarm is as follows.

$$\Lambda(Z_{i_1,i_2,...,i_S}|t = 0) = \prod_{s=1}^{S} [P_{F_s}]^{\delta_{s,i_s}}, \quad (5.2)$$

where $P_{F_s} \geq 0$ is the probability of false alarm for sensor $s$. 
The cost of associating a set of measurements $Z_{i_1, i_2, ..., i_S}$ to a target $t$ is given by

$$
c^t_{i_1, i_2, ..., i_S} = \frac{\Lambda(Z_{i_1, i_2, ..., i_S} | \omega_t)}{\Lambda(Z_{i_1, i_2, ..., i_S} | t = 0)}
= \prod_{s=1}^{S} \left[ \frac{P_{D_s} \cdot p(z_{s, i_s} | \omega_t)}{P_{F_s}} \right]^{\delta_{s, i_s}} [1 - P_{D_s}]^{1 - \delta_{s, i_s}}.
$$

(5.3)

This is the likelihood that $Z_{i_1, i_2, ..., i_S}$ corresponds to an actual target and not a false alarm.

Multiplying a large set of small numbers leads to round off errors as the product approaches zero. To avoid this problem, we apply the logarithm function on both sides. The cost of assigning a set of measurements $Z_{i_1, i_2, ..., i_S}$ to a target $t$ is given by the negative logarithms of the likelihood ratio.

$$
c''_{i_1, i_2, ..., i_S} = -\ln \left( \frac{\Lambda(Z_{i_1, i_2, ..., i_S} | \omega_t)}{\Lambda(Z_{i_1, i_2, ..., i_S} | t = 0)} \right)
$$

(5.4)

Instead of maximizing the likelihood function, we now try to minimize the negative log-likelihood ratio. A good association would, therefore, have a large negative cost.

In practice, the actual location of target $t$ is not known. If it were, then obtaining measurements would be useless. We define an estimate of the target position as

$$
\hat{\omega}_t = \arg \max_{\omega_t} \Lambda(Z_{i_1, i_2, ..., i_S} | \omega_t).
$$

The estimated target position maximizes the likelihood of a given set of measurements.
The generalized likelihood ratio utilizes an estimated target position. Our negative log-likelihood ratio takes the following form

\[
c_{i_1,i_2,\ldots,i_S} = - \sum_{s=1}^{S} \delta_{s,i_s} \cdot \left[ \ln \frac{P_{D_s}}{2\pi P_F \sqrt{|R_s|}} - \left[ z_{i,s} - h(\hat{\omega}_t, \omega_s) \right]' R_s^{-1} \left[ z_{i,s} - h(\hat{\omega}_t, \omega_s) \right] \right] - \sum_{s=1}^{S} [1-\delta_{s,i_s}] \cdot \ln (1 - P_{D_s}). \tag{5.5}
\]

We can do a type of gating\(^1\) by simply dropping any association with \(c_{i_1,i_2,\ldots,i_S} > 0\). A feasible solution of the MTMST problem assigns each measurement to no more than one \(S\)-tuple or association \(Z_{i_1,i_2,\ldots,i_S}\). In other words, each measurement may not be associated with more than one target. The result is a multidimensional assignment problem that chooses tuples of measurements minimizing the negative log likelihood. This is formally given as a 0-1 integer program.

\[
\begin{align*}
\text{min} & \quad \sum_{Z_{i_1,i_2,\ldots,i_S}} c_{i_1,i_2,\ldots,i_S} \cdot \rho_{i_1,i_2,\ldots,i_S} \\
\text{s.t.} & \quad \sum_{i_2,i_3,\ldots,i_S} \rho_{i_1,i_2,\ldots,i_S} = 1 \quad \forall \ i_1 = 1, 2, \ldots, n_1 \\
& \quad \sum_{i_1,\ldots,i_{s-1},i_{s+1},\ldots,i_S} \rho_{i_1,i_2,\ldots,i_S} \leq 1 \quad \forall \ i_s = 1, 2, \ldots, n_s; \quad \forall \ s = 2, 3, \ldots, S-1 \\
& \quad \sum_{i_1,i_2,\ldots,i_{S-1}} \rho_{i_1,i_2,\ldots,i_S} \leq 1 \quad \forall \ i_S = 1, 2, \ldots, n_S, \\
\end{align*}
\tag{5.6}
\]

\(^1\) Gating is a process of initially excluding some measurement-target assignments because of an arbitrarily large distance between the measurement and target.
where \( \rho_{i_1,i_2,\ldots,i_S} = \begin{cases} 1 & \text{if the tuple } Z_{i_1,i_2,\ldots,i_S} \text{ is assigned to the same target} \\ 0 & \text{otherwise} \end{cases} \)

\[ Z_{i_1,i_2,\ldots,i_S} = \{ z_{1,i_1}, z_{2,i_2}, \ldots, z_{S,i_S} \} \]

\[ n_1 = \min_s n_s \forall s = 1, 2, \ldots, S \]

\[ z_{s,i_s} \in \mathbb{R}^3 \]

The objective is to find \( n_1 \) measurement associations so that the sum of all the negative log-likelihood costs are minimized. Measurements are assigned to a maximum of one association or \( S \)-tuple. We define the Boolean decision variable \( \rho_{i_1,i_2,\ldots,i_S} \) to be zero when not all measurements \( \{ z_{1,i_1}, z_{2,i_2}, \ldots, z_{S,i_S} \} \) are assigned to the same target.

The total number of possible partitions of \( \sum_{s=1}^{S} n_s \) measurements into \( T \) targets is given by

\[ \Psi_M = \begin{cases} \left[ \sum_{i=0}^{n_S} \binom{T}{n_S-i} \cdot \frac{n_s!}{i!} \right]^S & \text{for } n_S \leq T \\ \left[ \sum_{i=0}^{T} \binom{T}{n_S-i} \cdot \frac{n_M!}{(n-S+i)!} \right] & \text{for } n_S > T \end{cases} \]  

(5.7)

where \( n_S \geq \max \{ n_1, n_2, \ldots, n_{S-1} \} \).

### 5.3 Minimum Subset of Cost Coefficients

Our objective is to preprocess the fully-dense data association problem by reducing the size of the problem to a smaller subset. We would expect advantages such as reduced storage requirements and less complexity for some algorithms. The development of a minimum subset of cost coefficients is based on the work in Chapter 4 (specifically Proposition 4.1) where we use the index tree representation of the MAP and randomly select \( \alpha \) nodes from each level of tree where

\[ \alpha = \left\lfloor \frac{n^{d-1}}{(n!)^{(d-1)/n}} \right\rfloor. \]  

(5.8)
When these $\alpha$ nodes from each level are combined into set $A$, we can expect this set to contain at least one feasible solution to the MAP. For the generalized MAP with dimension $d$ and $n_i$ elements in each dimension $i$, $i = 1, 2, \ldots, d$, and $n_1 \leq n_2 \leq \cdots \leq n_d$, we can easily extend equation (5.8) by noting the number of feasible solutions is $\prod_{i=2}^{d} \frac{n_i!}{(n_i-n_1)!}$. Using this we find

$$\alpha = \left\lceil \frac{\prod_{i=2}^{d} n_i}{\left( \prod_{i=2}^{d} \frac{n_i!}{(n_i-n_1)!} \right)^{\frac{1}{n_1}}} \right\rceil.$$  \hspace{1cm} (5.9)

Consider an MAP where the cost coefficients of the index tree are sorted in non-decreasing order for each level of the tree. If the cost coefficients are independent identically distributed random variables then the first $\alpha$ cost coefficients are from random locations at each level. Therefore, we may use Proposition 4.1 and conclude we can expect at least one feasible solution in $A$. The cardinality of this set $A$ is substantially smaller than the original MAP which may result in faster solution times. Table 5–1 shows a comparison of the size of $A$ to the size of the three original problems. Since the reduced set is made up of the smallest cost coefficients we expect good solution values.

Table 5–1: Comparisons of the number of cost coefficients of original MAP to that in $A$.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Number of Cost Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>5x5x5</td>
<td>125</td>
</tr>
<tr>
<td>10x10x10x10</td>
<td>10000</td>
</tr>
<tr>
<td>8x9x10x10x10</td>
<td>72000</td>
</tr>
</tbody>
</table>

|                 | 20                          |
|                 | 110                         |
|                 | 72                          |

Now consider an MAP where cost coefficients are not independent and identically distributed. In real world applications, cost coefficients will most likely be dependent. Consider, for example, a multisensor multitarget tracking situation where a small target is tracked among other large targets. We can expect a higher noise/signal ratio for the smaller target. Thus, cost coefficients associated with measurements
of the smaller target in the data association will be correlated to each other. In the case of dependent cost coefficients, Proposition 4.1 cannot be directly applied because the $\alpha$ smallest cost coefficients will not be randomly distributed across each level of the index tree. However, using Proposition 4.1 as a starting point, consider selecting some multiple, $\tau > 1$, of $\alpha$ cost coefficients from each level of a sorted index tree. For example, select the first $\tau\alpha$ cost coefficients from each of the sorted levels of the index tree to form a smaller index tree $A$. As $\tau$ is increased, the cardinality of $A$ obviously increases but so does the opportunity that a feasible solution exists in $A$. The best value of $\tau$ depends upon the particular MAP instance, but we can empirically determine a suitable estimate. In this chapter, we use a consistent value of $\tau = 10$ wherever the probabilistic approach is used.

### 5.4 GRASP for a Sparse MAP

A greedy randomized adaptive search procedure (GRASP) \cite{36, 37, 38, 4} is a multi-start or iterative process in which each GRASP iteration consists of two phases. In a construction phase, a random adaptive rule is used to build a feasible solution one component at a time. In a local search phase, a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result.

#### 5.4.1 GRASP Complexity

It is easy to see that GRASP can benefit in terms of solution times for the MAP by reducing the size of the problem. This can be seen by noting there are $N$ cost coefficients in the complete MAP where $N = \prod_{i=1}^{d} n_i$. As the complexity of the construction phase can be shown to be $O(N)$ \cite{4}, a smaller $N$ will directly reduce the time it takes for each construction phase. As it is easy to see that reducing the problem size to something less than $N$ helps in the construction phase, it remains to be seen how the local search phase is effected.

The local search phase of GRASP for the MAP often relies on the 2-exchange neighborhood \cite{74, 4}. A thorough examination of other neighborhoods for the MAP
is provided in the work by Pasiliao [84]. The local search procedure is as follows. Start from a current feasible solution, examine one neighbor at a time. If a lower cost is found adopt the neighbor as the current solution and start the local search procedure again. Continue the process until no better solution is found. The size of the 2-exchange neighborhood is \( d \binom{n_1}{2} \). As the size of the neighborhood is not directly dependent upon \( N \) there appears, at first, to be no advantage or disadvantage of reducing the number of cost coefficients in the problem. However, an obstacle surfaces in the local search procedure because, as the construction phase produces a feasible solution, we have no guarantee a neighbor of this solution even exists in the sparse problem. A feasible solution consists of \( n_1 \) cost coefficients. A neighbor in the 2-exchange neighborhood has the same \( n_1 \) cost coefficients except for two. In a sparse MAP, most cost coefficients are totally removed from the problem. Therefore, the local search phase first generates a potential neighbor and then must determine whether the neighbor exists. In a complete MAP, the procedure may access the cost matrix directly; however, the sparse problem cannot be accessed directly in the same way. A simple procedure is to simply scan all cost coefficients in the sparse problem to find the two new cost coefficients or to determine that one does not exist. This is an expensive procedure. We propose a data structure which provides a convenient, inexpensive way of evaluating existing cost coefficients or determining that they do not exist.

5.4.2 Search Tree Data Structure

We propose to use a search tree data structure to find a particular cost coefficient or determine that one does not exist in the sparse problem. The search tree has \( d + 1 \) levels. The tree is constructed such that there are \( n_i \) branches extending from each node at level \( i, i = 1, 2, \ldots, d \). The bottom level, \( i = d + 1 \), (leaves of the tree) contains each of the cost coefficients (if they exist). The maximum number of nodes in the tree including the leaves is equal to \( 1 + \sum_{i=1}^{d} \prod_{j=1}^{i} n_j \) and therefore, the time
to construct the tree is $O(N)$. An example of this search tree is given in Figure 5–3 for a complete 3x3x3 MAP. When searching for a particular cost coefficient, start at level $i = 1$ and traverse down branch $y, y = 0, \ldots, n$, where $y$ is the element of the $i^{th}$ dimension for the cost coefficient. Continue this process until either level $i = d + 1$ is reached, in which case the cost coefficient exists, or a null pointer is reached, in which case we may conclude the cost coefficient does not exist. It is obvious the search time is $O(d)$.

![Search tree data structure used to find a cost coefficient or determine a cost coefficient does not exist.](image)

Figure 5–3: Search tree data structure used to find a cost coefficient or determine a cost coefficient does not exist.

A search tree built from sparse data is shown in Figure 5–4. As an example of searching for cost coefficient (001), start at level 1 and traverse down branch labelled “0” to the node at level 2. From level 2, traverse again down branch labelled “0” to the node at level 3. From level 3, traverse down branch labelled “1” to the cost coefficient. Another example is searching for cost coefficient (222). Start at level 1 and traverse down branch labelled “2” to the node at level 2. From level 2, traverse again down branch labelled “2” to find it is a null pointer. The null pointer indicates the cost coefficient does not exist in the sparse MAP.
The GRASP algorithm can benefit from this search tree data structure if the problem is sparse. In a dense problem, it would be best to put cost coefficients in a matrix which can be directly accessed – this would benefit the local search phase. However, in the sparse problem, completely eliminating cost coefficients reduces storage and benefits the construction phase. It remains a matter of experimentation and closer examination to find the level of sparseness where the search tree data structure becomes more beneficial.

5.4.3 GRASP vs Sparse GRASP

To compare the performance of GRASP to solve a fully dense problem against the performance of GRASP to solve a sparse problem, we used simulated data from a multisensor multitarget tracking problem [74]. The problems ranged in size from five to seven sensors. Those with five sensors had five to nine targets. Problems with six and seven sensors had just five targets. Two problems of each size were tested. The problem size is indicated by the problem title. For example, “s5t6rm1” means problem one with five sensors and six targets. The experiment conducted five runs of each solution algorithm and reports the average time-to-solution, the average solution value and the best solution value from the five runs. The solution
times can be easily adjusted for each algorithm by simply adjusting the number of iterations. An obvious consequence is that as the number of iterations goes down, the solution quality generally gets worse. To create sparse instances of each problem, the probabilistic approach described above in Section 5.3 was used where $\tau = 10$. Table 5–2 shows the results of the experiment. Except for problems s5t8rm1 and s5t8rm2, reducing the problem size increased solution quality with less time-to-solution.

Table 5–2: Table of experimental results of comparing solution quality and time-to-solution for GRASP in solving fully dense and reduced simulated MSMTT problems. Five runs of each algorithm were conducted against each problem.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Opt Sol</th>
<th>Ave Sol</th>
<th>Ord Grasp Ave Sol</th>
<th>Best Sol Ave Sol</th>
<th>Time (sec)</th>
<th>Sparse Grasp Ave Sol</th>
<th>Best Sol Ave Sol</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s5t5rm1</td>
<td>-50</td>
<td>-49.2</td>
<td>-50</td>
<td>0.026</td>
<td>-50</td>
<td>-50</td>
<td>0.022</td>
<td></td>
</tr>
<tr>
<td>s5t5rm2</td>
<td>-44</td>
<td>-38</td>
<td>-41</td>
<td>0.024</td>
<td>-43.8</td>
<td>-44</td>
<td>0.024</td>
<td></td>
</tr>
<tr>
<td>s5t6rm1</td>
<td>-57</td>
<td>-54</td>
<td>-51.4</td>
<td>0.044</td>
<td>-49.4</td>
<td>-52</td>
<td>0.044</td>
<td></td>
</tr>
<tr>
<td>s5t6rm2</td>
<td>-45</td>
<td>-38.6</td>
<td>-41</td>
<td>0.0462</td>
<td>-45</td>
<td>-45</td>
<td>0.04</td>
<td></td>
</tr>
<tr>
<td>s5t7rm1</td>
<td>-63</td>
<td>-52.6</td>
<td>-59</td>
<td>0.0902</td>
<td>-61.2</td>
<td>-62</td>
<td>0.0962</td>
<td></td>
</tr>
<tr>
<td>s5t7rm2</td>
<td>-66</td>
<td>-59.2</td>
<td>-62</td>
<td>0.0862</td>
<td>-61.8</td>
<td>-62</td>
<td>0.0822</td>
<td></td>
</tr>
<tr>
<td>s5t8rm1</td>
<td>-74</td>
<td>-64.8</td>
<td>-67</td>
<td>0.1322</td>
<td>-71.2</td>
<td>-72</td>
<td>0.1262</td>
<td></td>
</tr>
<tr>
<td>s5t8rm2</td>
<td>-33</td>
<td>-20.6</td>
<td>-32</td>
<td>0.1402</td>
<td>-17</td>
<td>-25</td>
<td>0.1542</td>
<td></td>
</tr>
<tr>
<td>s5t9rm1</td>
<td>-84</td>
<td>-74.6</td>
<td>-78</td>
<td>1.7044</td>
<td>-74.4</td>
<td>-77</td>
<td>1.8326</td>
<td></td>
</tr>
<tr>
<td>s5t9rm2</td>
<td>-65</td>
<td>-59</td>
<td>-61</td>
<td>1.6664</td>
<td>-60.6</td>
<td>-63</td>
<td>1.5702</td>
<td></td>
</tr>
<tr>
<td>s6t5rm1</td>
<td>-48</td>
<td>-44.4</td>
<td>-48</td>
<td>0.9676</td>
<td>-48</td>
<td>-48</td>
<td>0.9194</td>
<td></td>
</tr>
<tr>
<td>s6t5rm2</td>
<td>-45</td>
<td>-42</td>
<td>-42</td>
<td>0.9754</td>
<td>-45</td>
<td>-45</td>
<td>0.8392</td>
<td></td>
</tr>
<tr>
<td>s7t5rm1</td>
<td>-51</td>
<td>-41.6</td>
<td>-44</td>
<td>1.378</td>
<td>-50.4</td>
<td>-51</td>
<td>1.0556</td>
<td></td>
</tr>
<tr>
<td>s7t5rm2</td>
<td>-52</td>
<td>-44.8</td>
<td>-47</td>
<td>1.4804</td>
<td>-52</td>
<td>-52</td>
<td>1.0916</td>
<td></td>
</tr>
</tbody>
</table>

5.5 Conclusion

In this chapter, we implemented techniques to reduce the size of the data association problem that is linked to the MSMTT problem. Empirical results indicate that probabilistically reducing the cardinality generally increases the solution quality and decreases the time-to-solution for heuristics such as GRASP. We suggest that further research is needed to study this approach on problems that are initially sparse in the first place which is a common occurrence in real-world problems. Additionally, we believe the probabilistic approach to reducing MAP size could be extended to other solution algorithms such as simulated annealing.
CHAPTER 6
EXPECTED NUMBER OF LOCAL MINIMA FOR THE MAP

As discussed in previous chapters, the MAP is an \( NP \)-hard combinatorial optimization problem occurring in many applications, such as data association. As many solution approaches to this problem rely, at least partly, on local neighborhood searches, it is widely assumed the number of local minima has implications on solution difficulty. In this chapter, we investigate the expected number of local minima for random instances of this problem. Both 2-exchange and 3-exchange neighborhoods are considered. We report on experimental findings that expected number of local minima does impact effectiveness of three different solution algorithms that rely on local neighborhood searches.

6.1 Introduction

In this chapter we develop relationships for the expected number of local minima. The 2-exchange local neighborhood appears as the most commonly used neighborhood in meta-heuristics such as GRASP that are applied to the MAP as evidenced in several different works [4, 74, 27]. Although the 2-exchange is most common in the literature, we include in this chapter some analysis of the 3-exchange neighborhood for comparison purposes.

The motivation of this chapter is that the number of distinct local minima of an MAP may have implications for heuristics that rely, at least partly, on repeated local searches in neighborhoods of feasible solutions [112]. In general, if the number of local minima is small then we may expect better performance from meta-heuristic algorithms that rely on local neighborhood searches. A solution landscape is considered to be rugged if the number of local minima is exponential with respect to the size of the problem [78]. Evidence by Angel and Zissimopoulos [9] showed that ruggedness
of the solution landscape has a direct impact on the effectiveness of the simulated annealing heuristic in solving at least one other hard problem, the quadratic assignment problem.

The concept of solution landscapes was first introduced by Wright [111] as a non-mathematical way to describe the action during evolution of selection and variation [102]. The idea is to imagine the space in which evolution occurs as a landscape. In one dimension there is the genotype and in another dimension there is a height or fitness. Evolution can be viewed as the movement of the population, represented as a set of points (genotypes), towards higher (fitter) areas of the landscape. In an analogous way, a solution process for a combinatorial problem can be viewed as the movement from some feasible solution with its associated cost (fitness) towards better cost (fitter) areas within the solution landscape. As pointed out by Smith et al. [102], the difficulty of searching in a given problem is related to the structure of the landscape, however, the exact relationship between different landscape features and the time taken to find good solutions is not clear. To name a couple of the landscape features of interest are number local optima and basins of attraction.

Reidys and Stadler [93] describe some characteristics of landscapes and express that local optima play an important role since they might be obstacles on the way to the optimal solution. From a minimization perspective, if $\hat{x}$ is a feasible solution of some optimization problem and $f(\hat{x})$ is the solution cost, then $\hat{x}$ is a local minima iff $f(\hat{x}) \leq f(\hat{y})$ for all $\hat{y}$ in the neighborhood of $\hat{x}$. Obviously the size of the neighborhood depends upon the definition of the neighborhood. According to Reidys and Stadler [93] there is no simple way of computing the number of local minima without exhaustively generating the solution landscape. However, the number can be estimated as done in some recent works [43, 45].

Rummukainen [98] describes some aspects of landscape theory which have been used to prove convergence of simulated annealing. Of particular interest are some
results on the behavior of local optimization on a few different random landscape classes. For example, the expected number of local minima is given for the $N - k$ landscape.

Associated with local minima is a basin $B(\hat{x})$ defined by means of a steepest descent algorithm $[93]$. Let $f(x_i)$ be the cost of some feasible solution $x_i$. Starting from $x_i, i = 0$, record for all $y$-neighbors the solution cost $f(y)$. Let $x_i + 1 = y$ for neighbor $y$ where $f(y)$ is the smallest for all neighbors and $f(y) < f(x_i)$. Stop when $x_i$ is a local minima. It becomes apparent; however, that a basin may have more than one local minima because of the definition of local minima is not strict. The basin sizes becomes important for simple meta-heuristics. For example, consider selecting a set of feasible solutions that are uniformly distributed in the solution landscape and performing a steepest descent. A question is what is the probability of starting in the basin with the global minima? This question is partially addressed by Garnier and Kaller $[45]$.

Long and Williams $[68]$ mention that problems are generally easier to solve when the number of local optima is small, but the difficulty can increase significantly when the number of local optima is large. The authors consider the quadratic 0-1 problem where instances are randomly generated over integers symmetric about 0. For such problems, the authors show the expected number of local maxima increases exponentially with respect to $n$, the size of the problem. They also reconcile this result with Pardalos and Jha $[80]$ who showed when test data are generated from a normal distribution, the expected number of local maxima approaches 1 as $n$ gets large.

Angel and Zissimopoulos $[9]$ introduce a ruggedness coefficient which measures the ruggedness of the QAP solution landscape. They conclude that because the QAP landscape is rather flat, this gives theoretical justification for the effectiveness of local search algorithms. The ruggedness coefficient is an extension of the autocorrelation coefficient introduced by Weinberger $[110]$. The larger the autocorrelation coefficient
the more flat is the landscape – and so, as postulated by the authors, the more suited is the problem for any local-search-based heuristic. Angel and Zissimopoulos [9] calculate the autocorrelation coefficient for the QAP as being no smaller than $n/4$ and no larger than $n/2$ which is considered relatively large. They develop the parameter, ruggedness coefficient, $\zeta$, which is independent of problem size and lies between 0 to 100. Close to 100 means the landscape is very steep. They go on to show experimentally that increasing $\zeta$ for the same problem size results in higher relative solution error and higher number of steps when using a simulated annealing algorithm by Johnson et al. [53]. The conclusions Angel and Zissimopoulos [9] are a relatively low ruggedness coefficient for the QAP gives theoretical justification of the effectiveness of local-search-based heuristics for the QAP.

This chapter will further investigate the assumption that number of local minima impacts the effectiveness of algorithms such as simulated annealing in solving the MAP.

The next section provides some additional background on the 2-exchange and 3-exchange local search neighborhoods. Then in Section 6.3, we provide experimental results on the average number of local minima for variously sized problems with assignment costs independently drawn from different distributions. Section 6.4 describes the expected number of local minima for MAPs of size of $n = 2$ and $d \geq 3$ where the cost elements are independent identically distributed random variables from any probability distribution. Section 6.5 describes lower and upper bounds for the expected number of local minima for all sizes of MAPs where assignment costs are independent standard normal random variables. Then in Section 6.6, we investigate whether the expected number of local minima impacts the performance of various algorithms that rely on local searches. Some concluding remarks are given in the last section.
6.2 Some Characteristics of Local Neighborhoods

A first step is to establish the definition of a neighborhood of a feasible solution. Let $\mathcal{N}_p(k)$ be the $p$-exchange neighborhood of the $k$-th feasible solution, $k = 1, \ldots, N$, where $N$ is the number of feasible solutions to the MAP. The $p$-exchange neighborhood is all $p$- or less element exchange permutations in each dimension of the feasible solution. The neighborhood is developed from the work by Lin and Kernighan [66].

If $z_k$ is the solution cost of the $k$-th solution, then $z_k$ is a discrete local minimum iff $z_k \leq z_j$ for all $j \in \mathcal{N}_p(k)$. As an example of a 2-exchange neighbor, consider the following feasible solution to an MAP with $d = 3, n = 3$: $\{111, 222, 333\}$. A neighbor is $\{111, 322, 233\}$. The solution $\{111, 222, 333\}$ is a local minimum if its solution cost is less than or equal to all of its neighbor’s solution costs.

The formula for the number of neighbors, $J$, of a feasible solution in the 2-exchange neighborhood of an MAP with dimension $d$ and $n$ elements in each dimension is as follows

$$J = |\mathcal{N}_2(k)| = d \binom{n}{2}.$$

(6.1)

It is obvious that for a fixed $n$, $J$ is linear in $d$. On the other hand for a fixed $d$, $J$ is quadratic in $n$. If we define a flat MAP as one with relatively small $n$ and define a deep MAP as one with relatively large $n$, then we expect larger neighborhoods in deep problems.

Similarly, for $n > 2$ the size of the 3-exchange neighborhood is as follows

$$J = |\mathcal{N}_3(k)| = d \left[ \binom{n}{2} + 2 \binom{n}{3} \right].$$

(6.2)

Similar to above for the 2-exchange, it becomes clear $J$ is linear with respect to $d$ and cubic with respect to $n$.

The minimum number of local minima for any instance is one - the global minimum. At the other extreme, the maximum number of local minima is $(n!)^{d-1}$ which
is the number of feasible solutions of an MAP. This occurs if all cost coefficients are equal.

6.3 Experimentally Determined Number of Local Minima

Studies were made of randomly produced instances of MAPs to empirically determine $E[|M|]$. The assignment costs $c_{i_1 \cdots i_d}$ for each problem instance were drawn from one of three distributions. The first distribution of assignment costs used is the uniform, $U[0, 1]$. The next distribution used is the exponential with mean one, being determined by $c_{i_1 \cdots i_d} = -\ln U$. Finally, the third distribution used was the standard normal, $N(0, 1)$, with values determined by the polar method [63].

Table 6–1 shows the average number of local minima for randomly generated instances of the MAP when considering a 2-exchange neighborhood. For small sized problems, the study was conducted by generating an instance of an MAP and counting number of local minima through complete enumeration of the feasible solutions. The values in the tables are the average number of local minima from 100 problem instances. For larger problems (indicated by * in the table), the average number of local minima was found by examining a large number of generated problem instances. For each instance of a problem we randomly selected a feasible solution and determined whether it was a local minimum. This technique gives an estimate of the probability that any feasible solution is a local minima. This relationship was then used to estimate the average number of local minima by multiplying the probability by the number of feasible solutions. This technique showed to have results consistent with the complete enumeration method mentioned above for small problems. Regardless of the distribution that cost coefficients were drawn, a standard deviation of 40-percent and 5-percent were observed for problems of sizes $d = 3, n = 3$ and

---

1 The number examined depends on problem size. The number ranged from $10^6$ to $10^7$. 
$d = 5, n = 5$, respectively. It is clear from the tables that $E[M]$ is effected by the distribution from which assignment costs are drawn. For example, problems generated from the exponential distribution have more local minima than problems generated from the normal distribution.

Table 6–2 shows similar results for the 3-exchange neighborhood and when cost coefficients are i.i.d. standard normal. We note, as expected, evidence indicates $E[M]$ is smaller in the 3-exchange case versus the 2-exchange case for the same sized problems.

Table 6–1: Average number of local minima (2-exchange neighborhood) for different sizes of MAPs with independent assignment costs.

<table>
<thead>
<tr>
<th>$n \setminus d$</th>
<th>Uniform on $[0,1]$</th>
<th>Exponential $\lambda = 1$</th>
<th>Standard Normal Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5.60</td>
<td>5.47</td>
<td>4.54</td>
</tr>
<tr>
<td>5</td>
<td>21.0</td>
<td>22.7</td>
<td>16.3</td>
</tr>
<tr>
<td>6</td>
<td>116</td>
<td>122</td>
<td>75.6</td>
</tr>
</tbody>
</table>

Table 6–3 shows the average proportion of feasible solutions that are local minima for both the 2-exchange and 3-exchange neighborhoods where costs are i.i.d. standard normal random variables. The table is followed by Figure 6–1 which includes plots of the proportion of local minima to number of feasible solutions. We observe that for
Table 6–2: Average number of local minima (3-exchange neighborhood) for different sizes of MAPs with i.i.d. standard normal assignment costs.

<table>
<thead>
<tr>
<th>$n \backslash d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.55</td>
<td>5.98</td>
<td>26.0</td>
<td>124</td>
</tr>
<tr>
<td>4</td>
<td>3.27</td>
<td>43.1</td>
<td>670</td>
<td>1.11E+4</td>
</tr>
<tr>
<td>5</td>
<td>8.27</td>
<td>516</td>
<td>3.48E+4*</td>
<td>2.65E+06*</td>
</tr>
<tr>
<td>6</td>
<td>28.8</td>
<td>8710*</td>
<td>3.06E+06*</td>
<td>1.22E+09*</td>
</tr>
</tbody>
</table>

either fixed dimension and increasing number of elements or visa versa, the proportion of local minima approaches zero.

Table 6–3: Proportion of local minima to total number of feasible solutions for different sizes of MAPs with i.i.d. standard normal costs.

### Proportion of local minima to feasible solutions using standard normal costs and 2-exchange

<table>
<thead>
<tr>
<th>$n \backslash d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.50E-01</td>
<td>2.00E-01</td>
<td>1.67E-01</td>
<td>1.43E-01</td>
</tr>
<tr>
<td>3</td>
<td>4.88E-02</td>
<td>3.27E-02</td>
<td>2.37E-02</td>
<td>2.11E-02</td>
</tr>
<tr>
<td>4</td>
<td>8.02E-03</td>
<td>4.50E-03</td>
<td>2.87E-03</td>
<td>2.81E-03</td>
</tr>
<tr>
<td>5</td>
<td>1.13E-03</td>
<td>5.43E-04</td>
<td>3.14E-04</td>
<td>1.91E-04</td>
</tr>
<tr>
<td>6</td>
<td>1.50E-04</td>
<td>6.32E-05</td>
<td>2.94E-05</td>
<td>1.80E-05</td>
</tr>
</tbody>
</table>

### Proportion of local minima to feasible solutions using standard normal costs and 3-exchange

<table>
<thead>
<tr>
<th>$n \backslash d$</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4.23E-02</td>
<td>2.87E-02</td>
<td>2.05E-02</td>
<td>1.58E-02</td>
</tr>
<tr>
<td>4</td>
<td>5.60E-03</td>
<td>3.13E-03</td>
<td>2.03E-03</td>
<td>1.40E-03</td>
</tr>
<tr>
<td>5</td>
<td>6.40E-04</td>
<td>2.99E-04</td>
<td>1.68E-04</td>
<td>1.06E-04</td>
</tr>
<tr>
<td>6</td>
<td>5.94E-05</td>
<td>2.33E-05</td>
<td>1.14E-05</td>
<td>6.31E-06</td>
</tr>
</tbody>
</table>

Figure 6–1: Proportion of feasible solutions that are local minima when considering the 2-exchange neighborhood and where costs are i.i.d. standard normal.
6.4 Expected Number of Local Minima for $n = 2$

In the special case of an MAP where $n = 2$, $d \geq 3$, and cost elements are independent identically distributed random variables from some continuous distribution with c.d.f $F(\cdot)$, one can establish a closed form expression for the expected number of local minima. To show this, we recall that distribution $F_{X+Y}$ of the sum of two independent random variables $X$ and $Y$ is determined by the convolution of the respective distribution functions, $F_{X+Y} = F_X * F_Y$.

We now borrow from Proposition 3.1 to construct the following proposition.

**Proposition 6.1** In an instance of the MAP with $n=2$ and with cost coefficients that are i.i.d. random variables with continuous distribution $F$, the costs of all feasible solutions are independent distributed random variables with distribution $F * F$.

**Proof:** Let $I$ be an instance of MAP with $n = 2$. Each feasible solution for $I$ is an assignment $a_1 = c_1, \delta_1(1), ..., \delta_{d-1}(1)$, $a_2 = c_2, \delta_1(2), ..., \delta_{d-1}(2)$, with cost $z = a_1 + a_2$. The important feature of such assignments is that for each fixed entry of $c_1, \delta_1(1), ..., \delta_{d-1}(1)$, there is just one remaining possibility, namely $c_2, \delta_1(2), ..., \delta_{d-1}(2)$, since each dimension has only two elements. This implies that different assignments cannot share elements in the cost vector, and therefore different assignments have independent costs $z$. Now, $a_1$ and $a_2$ are independent variables from $F$. Thus $z = a_1 + a_2$ is a random variable with distribution $F * F$.

One other proposition is important to this development.

**Proposition 6.2** Given $j$ i.i.d. random variables with continuous distributions, the probability that the $r^{th}$, $r = 1, \ldots, j$, variable is the minimum value is $1/j$.

**Proof:** Consider $j$ i.i.d. random variables, $X_i, i = 1, \ldots, j$, with c.d.f. $F(\cdot)$ and p.d.f. $f(\cdot)$. Let $X_{(j-1)}$ be the minimum of $j - 1$ of these variables,

$$X_{(j-1)} = \min\{X_i \mid i = 1, \ldots, j, \ i \neq r\},$$
whose c.d.f. and p.d.f. are computed trivially as

\[ F_{(j-1)}(x) = P[X_{(j-1)} \leq x] = 1 - (1 - F(x))^{j-1}, \]

\[ f_{(j-1)}(x) = \frac{d}{dx} F_{(j-1)}(x) = (j - 1)(1 - F(x))^{j-2} f(x). \]

Then, the probability that the \( r \)-th random variable is minimal among \( j \) i.i.d. continuous variables, is

\[ P[r\text{-th r.v. is minimal}] = P[X_r \leq X_{(j-1)}] = P[Y \leq 0] = F_Y(0). \quad (6.3) \]

Here \( F_Y(\cdot) \) is the c.d.f. of random variable \( Y = X_r - X_{(j-1)} \), and, by convolution rule, it equals to

\[ F_Y(x) = \int_{-\infty}^{+\infty} F(x - y)(j - 1)(1 - F(-y))^{j-2} f(-y) dy. \]

Hence, the probability (6.3) can immediately be calculated as

\[ P[X_r \leq X_{(j-1)}] = \int_{-\infty}^{+\infty} F(-y)(j - 1)(1 - F(-y))^{j-2} f(-y) dy \]

\[ = \frac{1}{j} \int_{-\infty}^{+\infty} j(1 - F(-y))^{j-1} f(-y) dy = \frac{1}{j}, \]

since \( j(1 - F(-y))^{j-1} f(-y) \) is the p.d.f. of \( -X_{(j)} = -\min\{X_1, \ldots, X_j\} \). The last equality yields the statement of the proposition.

The obvious consequence of Proposition 6.2 is that given a sequence of independent random variables from a continuous distribution, position of the minimum value is uniformly located within the sequence regardless of the parent distribution.

We are now ready to prove the main result of this section.

**Theorem 6.3** In an MAP with cost coefficients that are i.i.d. continuous random variables where \( n = 2 \) and \( d \geq 3 \), the expected number of local minima is given by

\[ E[M] = \frac{2^{d-1}}{d+1}. \]  

(6.4)
**Proof:** Let \( N \) be the number of feasible solutions to an \( n = 2 \) MAP, \( N = 2^{d-1} \).

Introducing indicator variables

\[
Y_k = \begin{cases} 
1, & k^{th} \text{ solution, } k = 1, \ldots, N, \text{ is a local minimum;} \\
0, & \text{otherwise,}
\end{cases}
\]  

(6.5)

one can write \( M \) as the sum of indicator variables:

\[
M = \sum_{k=1}^{N} Y_k.
\]

From the elementary properties of expectation it follows that

\[
E[M] = \sum_{k=1}^{N} E[Y_k] = \sum_{k=1}^{N} P[Y_k = 1],
\]

(6.6)

where \( P[Y_k = 1] \) is the probability that the cost of \( k \)-th feasible solution does not exceed the cost of any of its neighbors. Any feasible solution has \( J = d(n - 2) = d \) neighbors whose costs, by virtue of Proposition 6.1, are i.i.d. continuous random variables. Then, Proposition 6.2 implies that the probability of the cost of \( k \)-th feasible solution being minimal among its neighbors is equal to

\[
P[Y_k = 1] = \frac{1}{d + 1},
\]

which, upon substitution into (6.6), yields the statement of the theorem (6.4).

**Remark 6.3.1** Equality (6.4) implies that the number of local minima in an \( n = 2, \ d \geq 3 \) MAP is exponential in \( d \) when the cost coefficients are independently drawn from *any* continuous distribution.

**Corollary 6.4** The proved relation (6.4) can be used to derive the expected ratio of the number of local minima \( M \) to the total number of feasible solutions \( N \) in an \( n = 2, \ d \geq 3 \) MAP:

\[
E[M/N] = \frac{E[M]}{N} = \frac{1}{d + 1}.
\]
This shows that the number of local minima in an $n = 2$ MAP becomes infinitely small comparing to the number of feasible solutions, when dimension $d$ of the problem increases. This asymptotic characteristic is reflected in the numerical data above and may be useful for the development of novel solution methods.

### 6.5 Expected Number of Local Minima for $n \geq 3$

Our ability to derive a closed-form expression (6.4) for the expected number of local minima $E[M]$ in the previous section has relied on the independence of costs of feasible solutions in an $n = 2$ MAP. As it is easy to verify directly, in case of $n \geq 3$ the costs coefficients are generally not independent. This complicates significantly the analysis if an arbitrary continuous distribution for assignment costs $c_{i_1 \ldots i_d}$ is assumed. However, as we show below, one can derive upper and lower bounds for $E[M]$ in the case when the costs coefficients of (2.1) are normally distributed random variables.

First, we introduce a proposition, which follows a similar development by Beck [16].

**Proposition 6.5** Consider an $n \geq 3$, $d \geq 3$ MAP whose costs are i.i.d. continuous random variables. Then the expected number of local minima can be represented as

$$E[M] = \sum_{k=1}^{N} P \left[ \bigcap_{j \in \mathcal{N}_2(k)} z_k - z_j \leq 0 \right]$$

(6.7)

where $\mathcal{N}_2(k)$ is the 2-exchange neighborhood of the $k$-th feasible solution, and $z_i$ is the cost of the $i$-th feasible solution.

**Proof:** As before, $M$ can be written as the sum of indicator variables $Y_k$ (6.5), which consequently leads to

$$E[M] = \sum_{k=1}^{N} E[Y_k] = \sum_{k=1}^{N} P[Y_k = 1].$$

(6.8)

As $Y_k = 1$ means $z_k \leq z_j$ for all $j \in \mathcal{N}_2(k)$, it is obvious that $P[Y_k = 1] = P[z_k - z_j \leq 0, \forall j \in \mathcal{N}_2(k)]$, which proves the proposition.
If we allow the \( n^d \) cost coefficients \( c_{i_1\ldots i_d} \sim N(0, 1) \) of the MAP to be independent standard normal \( N(0, 1) \) random variables, then for any two feasible solutions the difference of their costs \( Z_{ij} = z_i - z_j \) is a normal variable with mean zero.

Without loss of generality, consider the \( k = 1 \) feasible solution to (2.1) whose cost is

\[
z_1 = c_{1\ldots 1} + c_{2\ldots 2} + \ldots + c_{n\ldots n}. \tag{6.9}
\]

In the 2-exchange neighborhood \( \mathcal{N}_2(1) \), the cost of a feasible solution differs from (6.9) by two cost coefficients, e.g.,

\[
z_2 = c_{21\ldots 1} + c_{12\ldots 2} + c_{3\ldots 3} + \ldots + c_{n\ldots n}.
\]

Generally, the difference \( z_1 - z_l \) of costs of (6.9) and that of any neighbor \( l \in \mathcal{N}_2(1) \) has the form

\[
Z_{rsq} = c_{r\ldots r} + c_{s\ldots s} - c_{r\ldots rsr\ldots r} - c_{s\ldots srs\ldots s}, \tag{6.10}
\]

where the last two coefficients have “switched” indices in the \( q \)-th position, \( q = 1, \ldots, d \). Observing that

\[
Z_{rsq} = Z_{srq} \quad \text{for} \quad r, s = 1, \ldots, n; \quad q = 1, \ldots, d,
\]

consider the \( J \)-dimensional random vector

\[
Z = (Z_{111}, \ldots, Z_{11d}, Z_{121}, \ldots, Z_{12d}, \ldots, Z_{rs1}, \ldots, Z_{rsd}, \ldots, Z_{n11}, \ldots, Z_{nnd}), \quad r \leq s. \tag{6.11}
\]
Vector $\mathbf{Z}$ has normal distribution $N(0, \Sigma)$, with covariance matrix $\Sigma$ defined as

$$
\text{Cov}(Z_{rsq}, Z_{ijk}) = \begin{cases} 
4, & \text{if } i = r, j = s, q = k, \\
2, & \text{if } i = r, j = s, q \neq k, \\
1, & \text{if } (i = r, j \neq s) \text{ or } (i \neq r, j = s), \\
0, & \text{if } i \neq r, j \neq s.
\end{cases} \quad (6.12)
$$

For example, in case $n = 3, d = 3$ the covariance matrix $\Sigma$ has the form

$$
\Sigma = \begin{pmatrix}
4 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 4 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 2 & 4 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 4 & 2 & 2 & 1 & 1 & 1 \\
1 & 1 & 1 & 2 & 4 & 2 & 1 & 1 & 1 \\
1 & 1 & 1 & 2 & 2 & 4 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 4 & 2 & 2 \\
1 & 1 & 1 & 1 & 1 & 2 & 4 & 2 & 2 \\
1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 4
\end{pmatrix}
$$

Now, the probability in (6.7) can be expressed as

$$
P\left[ \bigcap_{j \in \mathcal{N}_k} z_k - z_j \leq 0 \right] = F_{\Sigma}(0), \quad (6.13)
$$

where $F_{\Sigma}$ is the c.d.f. of the $J$-dimensional multivariate normal distribution $N(0, \Sigma)$. While the value of $F_{\Sigma}(0)$ in (6.13) is difficult to compute exactly for large $d$ and $n$, lower and upper bounds can be constructed using Slepian inequality [107]. To this
end, let us introduce covariance matrices \( \Sigma = (\sigma_{ij}) \) and \( \bar{\Sigma} = (\bar{\sigma}_{ij}) \)

\[
\sigma_{ij} = \begin{cases} 
4, & \text{if } i = j, \\
2, & \text{if } i \neq j \text{ and } (i - 1) \text{ div } d = (j - 1) \text{ div } d, \\
0, & \text{otherwise},
\end{cases}
(6.14a)
\]

\[
\bar{\sigma}_{ij} = \begin{cases} 
4, & \text{if } i = j, \\
2, & \text{otherwise},
\end{cases}
(6.14b)
\]

so that \( \sigma_{ij} \leq \sigma_{ij} \leq \bar{\sigma}_{ij} \) holds for all \( 1 \leq i, j \leq J \), with \( \sigma_{ij} \) being the components of the covariance matrix \( \Sigma \) (6.12). Then, the Slepian inequality claims that

\[
F_{\Sigma}(0) \leq F_{\bar{\Sigma}}(0) \leq F_{\bar{\Sigma}}(0),
(6.15)
\]

where \( F_{\Sigma}(0) \) and \( F_{\bar{\Sigma}}(0) \) are c.d.f.’s of random variables \( X_{\Sigma} \sim N(0, \Sigma) \) and \( X_{\bar{\Sigma}} \sim N(0, \bar{\Sigma}) \), respectively.

As the variable \( X_{\Sigma} \) is equicorrelated, the upper bound in (6.15) can be expressed by the one-dimensional integral (see, e.g., [107])

\[
\int_{-\infty}^{+\infty} \Phi(a z)^J \, d\Phi(z), \quad a = \sqrt{\frac{\rho}{1 - \rho}},
(6.16)
\]

where \( \Phi(\cdot) \) is the c.d.f. of standard normal distribution:

\[
\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} \, dt,
\]

and \( \rho = \sigma_{ij}/\sqrt{\sigma_{ii} \sigma_{jj}} \) is the correlation coefficient of distinct components of the corresponding random vector. The correlation coefficient of the components of vector \( X_{\bar{\Sigma}} \) is \( \rho = \frac{1}{2} \), which allows for a simple closed-form expression for the upper bound in (6.15)

\[
F_{\bar{\Sigma}}(0) = \int_{-\infty}^{+\infty} \Phi(z)^J \, d\Phi(z) = \frac{1}{J + 1}.
(6.17)
This immediately yields the value of the upper bound \( E[M] \) for the expected number of local minima \( E[M] \):

\[
E[M] = \sum_{k=1}^{N} F_{\Sigma}(0) = \frac{2 (n!)^{d-1}}{n(n-1)d + 2}.
\]

Let us now calculate the lower bound for \( E[M] \) using (6.15). According to the covariance matrix \( \Sigma \) (6.14a), the vector \( X_\Sigma \) is comprised of \( \frac{n(n-1)}{2} \) groups of variables, each consisting from \( d \) elements,

\[
X_\Sigma = \left( X_1, \ldots, X_d, \ldots, X_{(i-1)d+1}, \ldots, X_{id}, \ldots, X_{(n(n-1)/2-1)d}, \ldots, X_{(n(n-1)/2)d} \right),
\]

such that the variables from different groups are independent, whereas variables within a group have \( d \times d \) covariance matrix defined as in (6.14b). Thus, one can express the lower bound \( F_{\Sigma}(0) \) in (6.15) as a product

\[
F_{\Sigma}(0) = \prod_{i=1}^{\frac{n(n-1)}{2}} P \left[ X_{(i-1)d+1} \leq 0, \ldots, X_{id} \leq 0 \right].
\]

Since variables \( X_{(i-1)d+1}, \ldots, X_{id} \), are equicorrelated with correlation coefficient \( \rho = \frac{1}{2} \), each probability term in the last equality can be computed similarly to evaluation of the lower-bound probability (6.17), i.e.,

\[
F_{\Sigma}(0) = \prod_{i=1}^{\frac{n(n-1)}{2}} \int_{-\infty}^{+\infty} \left[ \Phi(z) \right]^d d\Phi(z) = \left( \frac{1}{d + 1} \right)^{\frac{n(n-1)}{2}}.
\]

This finally leads to a lower bound for the number of expected minima:

\[
E[M] = \sum_{k=1}^{N} F_{\Sigma}(0) = \frac{(n!)^{d-1}}{(d + 1)^{n(n-1)/2}}.
\]

In such a way, we have proved the following
Theorem 6.6. In an \( n \geq 3, d \geq 3 \) MAP with i.i.d. standard normal cost coefficients, the expected number of local minima is bounded as

\[
\frac{(n!)^{d-1}}{(d+1)^n(n-1)/2} \leq E[M] \leq \frac{2(n!)^{d-1}}{n(n-1)d+2}. \tag{6.18}
\]

Remark 6.6.1 From (6.18) it follows that for fixed \( n \geq 3 \), the expected number of local minima is exponential in \( d \).

Corollary 6.7 Similarly to the case \( n = 2 \), the developed lower and upper bounds can (6.18) can be used to estimate the expected ratio of the number of local minima \( M \) to the total number of feasible solutions \( N \) in an \( n \geq 3, d \geq 3 \) MAP:

\[
\frac{1}{(d+1)^{n(n-1)/2}} \leq E[M/N] \leq \frac{2}{n(n-1)d+2}.
\]

6.6 Number of Local Minima Effects on Solution Algorithms

In this section, we examine the question of whether number of local minima in the MAP has an impact on heuristics that rely, at least in part, on local neighborhood searches. We consider three heuristics

- Random Local Search
- Greedy Randomized Adaptive Search (GRASP)
- Simulated Annealing

The heuristics described in the following three subsections were exercised against various sized problems that were randomly generated from the standard normal distribution.

6.6.1 Random Local Search

The random local search procedure simply steps through a given number of iterations. Each iteration begins by selecting a random starting solution. The algorithm then conducts a local search until no better solution can be found. The algorithm
captures the overall best solution and reports it after executing the maximum number of iterations. The following is a more detailed description of the steps involved.

1. Set iteration number to zero, $\text{Iter} = 0$.
2. Randomly select a current solution, $x_{current}$.
3. While not all neighbors of $x_{current}$ examined, select a neighbor, $x_{new}$, of the current solution.
   
   If $z_{x_{new}} < z_{x_{current}}$, then $x_{current} \leftarrow x_{new}$.
4. If $z_{x_{current}} < z_{x_{best}}$, then $x_{best} \leftarrow x_{current}$.
5. If $\text{Iter} < \text{Iter}_{\text{max}}$, increment $\text{Iter}$ by one and go to Step 2. Otherwise, end.

6.6.2 GRASP

A greedy randomized adaptive search procedure (GRASP) [36, 37, 38] is a multi-start or iterative process, in which each GRASP iteration consists of two phases. In a construction phase, a random adaptive rule is used to build a feasible solution one component at a time. In a local search phase, a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result. The neighborhood search is conducted similar to that in the random local search algorithm above. That is neighbors of a current solution are examined one at a time and if an improving solution is found, it is adopted as the current solution and local search starts again. Local search ends when no improving solution is found.

GRASP has been used in many applications and specifically in solving the MAP [4, 74].

6.6.3 Simulated Annealing

Simulated Annealing is a popular heuristic used in solving a variety of problems [57, 70]. Simulated annealing uses a local search procedure, but the process allows *uphill* moves. The probability of moving uphill is higher at high *temperatures*, but as the process *cools*, there is less probability of moving uphill. The specific steps
for simulated annealing used in this chapter are taken from work by Gosavi [46].

Simulated annealing was recently applied to the MAP by Clemmons et al. [27].

6.6.4 Results

The heuristic solution quality, $Q$, which is the relative difference from the optimal value, is reported and compared for the same sized problems with assignment costs independently drawn from the standard normal distribution. The purpose of our analysis is not to compare the efficiency of the heuristic algorithms, but to determine the extent to which the number of local minima affects the performance of these algorithms. Each run of an experiment involved the following general steps:

1. Generate random MAP instance with cost coefficients that are i.i.d. standard normal random variables.
2. Obtain $M$ by checking each feasible solution for being a local minimum.
3. Solve the generated MAP instance using each of the above heuristics 100 times and return the average solution quality, $\overline{Q}$, for each heuristic method.

Problem sizes were chosen based on the desire to test a variety of sizes and the practical amount of time to determine $M$ (as counting $M$ is the obvious bottleneck in the experiment). Four problem sizes chosen were $d = 3$, $n = 6$; $d = 4$, $n = 5$; $d = 4$, $n = 6$; and $d = 6$, $n = 4$. For problem size $d = 4$, $n = 6$ which has the largest $N$, a single run took approximately four hours on a 2.2 GHz Pentium 4 machine. The number of runs of each experiment varied for each problem size with fewer runs for larger problems. The number of runs were 100, 100, 30, and 50, respectively, for the problem sizes listed above.

Figure 6–2 displays plots of the average solution quality for each of the three heuristics versus the number of local minima. The plots are the results from problem size $d = 4$, $n = 5$ and are typical for the other problem sizes. Included in each plot is a best-fit linear least-squares line that provides some insight on the effect of $M$ on solution quality. A close examination of the figures indicates that the solution
quality improves with smaller $M$ for each heuristic solution method. This conclusion was verified using a regression analysis to determine that the effect of $M$ on average solution quality is statistically significant ($p$-value averaged approximately 0.01).

We have also investigated the 3-exchange neighborhood. Figure 6–3 displays plots of the average solution quality of the three heuristics versus the number of local minima when using the larger neighborhood. The parameters in each heuristic such as number of iterations for random local search and GRASP were kept the same for each heuristic. The only change made was the neighborhood definition. The plots for random local search and GRASP indicate that $M$ affects solution quality (regression analysis shows average $p$-value of 0.05). However, the effect of $M$ is not statistically significant in the case of simulated annealing when using the 3-exchange neighborhood ($p$-value of 0.4).

We note a couple interesting aspects when comparing Figures 6–2 and 6–3. The solution quality when using random local search or GRASP improves when using the larger neighborhood. This is to be expected, but at the expense of longer solution times. We found on average the random local search took approximately 30% longer and GRASP took about 15% longer to complete the same number of iterations. Simulated annealing’s performance in terms of solution quality dropped when the neighborhood size was increased. This is not surprising as the optimal starting temperature and cooling rate are functions of the problem-instance characteristics such as size, neighborhood definition, cost coefficient values, etc. Our experimental results for simulated annealing reiterate the necessity for properly tuning heuristic parameters when the neighborhood definition is changed.

6.7 Conclusions

In this chapter we experimentally determined the average number of local minima for various sized problems generated from different distributions. Evidence suggests the distribution of cost coefficients has an impact on the number of local minima
Figure 6-2: Plots of solution quality versus number of local minima when using the 2-exchange neighborhood. The MAP has a size of $d = 4$, $n = 5$ with cost coefficients that are i.i.d. standard normal.
Figure 6-3: Plots of solution quality versus number of local minima when using a 3-exchange neighborhood. The MAP has a size of $d = 4$, $n = 5$ with cost coefficients that are i.i.d. standard normal.
when $n > 2$. We proved a closed form expression for $E[M]$ when $n = 2$, $d \geq 3$. The expression holds for any case when the cost coefficients are i.i.d. from a continuous distribution. Upper and lower bounds to $E[M]$ are given in the case of $n > 2$, $d \geq 3$ and cost coefficients are i.i.d. standard normal variables. The bounds show that $E[M]$ is exponential with respect to $d$. Finally, experimental results are provided that show at least three heuristics that rely, at least partly, on local search are adversely impacted by increasing number of local minima.
CHAPTER 7
MAP TEST PROBLEM GENERATOR

Test problems are often used in computational studies to establish the efficiency of solution methods, or, as pointed out by Yong and Pardalos [112], test problems are important for comparing new solution methods against existing algorithms. Along with a collection of combinatorial test problems, the book by Floudas et al. [39] emphasizes the importance of well designed problems. As Barr et al. [12] point out, there is a definite need for a variety of test problems to check the robustness and accuracy of proposed algorithms. A probabilistic approach in the development and study of test problems may result in higher quality test instances.

In this chapter we develop a test problem generator for the MAP and use a probabilistic analysis to determine its effectiveness in generating hard problems.

7.1 Introduction

There are at least four basic sources of test problems:

1. Real world problems (e.g. [95]).
2. Libraries of standard test problems [20, 59, 95].
3. Randomly generated problems such as those with cost coefficients drawn independently from some distribution such as uniform on [0,1].
4. Problems generated from an algorithm such as the quadratic test problem generator [112].

As noted by Reilly [94], real world problems have the advantage of providing results consistent with at least some problems encountered in practice. However, in most cases there is not a sizeable set of real world problems to constitute a satisfactory experiment. Libraries of standard test problems serve as a good source of problems; however, again there may not be enough of the right-sized problems.
Randomly generated problems provide virtually an infinite supply of test problems; however, the optimal solution to large problems may remain unknown. An additional hazard with randomly generated problems is they are often “easy” to solve [21, 101]. These may be significant issues when evaluating the performance of a new algorithm. Generated test problems with known solutions can also be in virtually infinite supply and, importantly, a unique known solution can be very useful in fully evaluating a solution algorithm’s performance. Careful study of generated problems is necessary to determine the relative usefulness of the problems in terms of difficulty, realism, etc. Sanchis [99] mentions generated problems should have the following properties:

- polynomial-time generability with known solution
- hardness
- diversity

Sanchis goes on to say that meeting all three requirements can be difficult. For example, meeting the first requirement can be quite simple by creating a trivial instance; however, a trivial instance would most likely violate the second property. Also, it is recognized by many researchers that there is a need for standardized representations of problem instances [40]. A popular technique in designing nonlinear programming test problems is the use of Karush-Kuhn-Tucker optimality conditions as proposed by Rosen and Suzuki [97]. Test problem generators for integer-programming problems are difficult to construct and require a deep insight into the problem structure [56].

An interesting approach to test problem generation is that of the Discrete Mathematics and Theoretical Computer Science (DIMACS) challenges [34]. Over the past decade, the challenges have had the goal of encouraging the experimental evaluation of algorithms. It is recognized that comparisons must be made on standard test problems that are included as part of the challenges. Challenges have been held for TSP, cliques, coloring, and satisfiability.
To summarize sources of test problems, the following two sections describe available test problem generators and libraries that include ready-made test problems.

7.1.1 Test Problem Generators

Steiner problem in graphs. Khoury et al. [56] present a binary-programming formulation for the Steiner problem in graphs which is well known to be \( NP \)-hard. They use this formulation to generate test problems with known optimal solutions. The technique uses the KKT optimality conditions on the corresponding quadratically constrained optimization problem.

Maximum clique problem. In the comprehensive work by Hasselberg et al. [48], the authors consider several interesting problems. They introduce different test problem generators that arise from a variety of practical applications as well as the problem of maximum clique. Applications include coding theory problems, problems from Keller’s Conjecture, and problems in fault diagnosis. Work by Sanchis and Jagota [100] discusses a test problem generator that builds the complementary minimum vertex cover problem. The hardness of their generated problems relies on construction parameters. Sanchis [99] provides an algorithm to generate minimum vertex cover problem that is diverse, hard and of known solution.


FOR. Calamai et al. [24] describe a technique for generating convex, strictly concave and indefinite (bilinear or not) quadratic programming problems. Their approach
involves combining $m$ two-variable problems to construct a separable quadratic problem. Palubeckis [77] provides a method for generating hard rectilinear QAPs with known optimal solutions.

**Graph colorability.** The graph colorability problem is the difficult problem of finding the least number of colors that colors a graph where no two adjacent nodes are of the same color. An algorithm for generating a test problem with known solution can be found in work by Sanchis [99].

**Minimum dominating set.** A minimum dominating set problem generator is provided in work by Sanchis [99].

**Satisfiability.** Achlioptas et al. [2] propose a generator that only outputs satisfiable problem instances. They show how to finely control the hardness of the instances by establishing a connection between problem hardness and a kind of phase transition phenomenon in the space of problem instances. Uchida et al. [108] provide a web page dedicated to two methods of generating instances of the 3-satisfiability problem.

**Traveling salesman problem (TSP).** A web site dedicated to generating the Euclidean TSP is maintained by Moscato [73]. The site provides information concerning research in generation of instances of TSPs with known optimal solution. An approach to generating discrete problems with known optima based on a partial description of the solution polytope is provided by Pilcher and Rardin [88]. The approach is used to generate instances of the symmetric traveling salesman problem.

**Minimum cut-set.** The minimum cut-set is the problem of partitioning a graph in two parts such that the edges between the two partitions are the minimal separating set of edges. For applications involving circuit designs, Krishnamurthy [61] provides an appropriate test problem generator.
7.1.2 Test Problem Libraries

**Handbook of test Problems.** Floudas et al. [39] present a collection of test problems arising in literature studies and a wide spectrum of applications. Applications include: pooling/blending operations, heat exchanger network synthesis, phase and chemical reactor network synthesis, parameter estimation and data reconciliation, clusters of atoms and molecules, pump network synthesis, trim loss minimization, homogeneous azeotropic separation, dynamic optimization and optimal control problems.

**Miscellaneous.** The OR-Library maintained by Beasley [15] is an extensive collection of test instances.

**Quadratic assignment problem.** Not only are a host of QAP test instances available, but QAPLIB [20] also provides other useful information concerning this difficult problem.

**Satisfiability.** The SATLIB - The Satisfiability Library is available on the web [50]. It has a collection of benchmark problems, solvers, and tools related to satisfiability research.

**Vehicle routing problems with time windows.** A large set of instances with up to 1000 customers is available [49].

**Traveling salesman problem (TSP).** The TSP library [96] contains several types of TSP instances.

**Linear ordering problem.** Instances of the linear ordering problem are available on the web [95].
Various. Test instances of several types can also be found on the web [59]. Types of problems found there are assignment, min-cut clustering, linear programming, integer and mixed-integer programming, matrix decomposition, matching, maximum flow in directed and undirected graphs, minimum cost network flow and transportation, set partitioning, Steiner tree, traveling salesman, and capacitated vehicle routing.

Steiner. Concerning Steiner tree problems, a library is available [58] that collects available instances of Steiner tree problems in graphs and provides some information about their origins, solvability and characteristics.

Frequency assignment problems. A library for frequency assignment problems is available through the internet [35]. Along with test problems, the library provides a vast bibliography of work concerning this important class of problem.

Assuming a variety of hard and diverse problems exist, careful use of the problems is of course necessary. A nice treatment concerning experimenting/reporting on solution algorithm performance is provided by Barr et al. [12]. The authors describe conditions of designing computational experiments to carefully examine heuristics. They also, give reporting guidelines.

Several previous works on solution methods for the MAP used test problems that were generated using some random method. These test problems may be classified into three categories. The first category of problems as used by Balas and Saltzman [10] and Pierskalla [87] draws integer costs from a uniform distribution. A second category of test problems as by Frieze [42] used cost coefficients $c_{ijk} = a_{ij} + a_{ik} + a_{jk}$ where $a_{ij}$, $a_{ik}$, and $a_{jk}$ are randomly generated integers from a uniform distribution. The last category of test problems includes problems generated for a specific MAP application such as for multitarget multisensor data association [74] or for some special cost structure such as decomposable costs as examined by Burkard et al. [23].
7.2 Test Problem Generator

We propose an algorithm that produces a quality MAP test problem with a known unique solution. The three-dimensional MAP is used for illustration.

7.2.1 Proposed Algorithm

The algorithm first constructs a tree graph \([87]\) based on the desired size of the problem. Figure 7–1 shows a tree graph for a three-dimensional problem where \(n_1 = 3, n_2 = 4\) and \(n_3 = 4\) (or 3x4x4 as a convenient notation which may be extended to other dimensions).

Each node of the tree represents an assignment \(x_{ijk}\) and has an assignment cost \(c_{ijk}\). In general, there are \(n_1\) levels in the tree graph, where the root is at level 0. The number of nodes on any level other than the root is \(\prod_{i=2}^{d} n_i\). Branches in the tree graph are feasible paths from a triple at one level to a triple at the next level down. A path from level 0 to level \(n_1\) is a feasible solution to the MAP. Using Figure 7–1 as an example, if 121 is the assignment chosen from the 16 available assignments at level
1, then the feasible assignments at the next level are 212, 213, 214, 232, 233, 234, 242, 243, 244. If 232 is then chosen at level 2, then 313, 314, 343, 344 are available at level 3. Using this procedure, a feasible path is 121, 232, 313.

In addition to the assignment cost, $c_{ijk}$, associated with each node, there is another cost called Lower Bound Path Cost, $lb_{ijk}$. This is the cost of the assignment plus the minimum lower bound path cost of any feasible node (assignment) at the next level down. A lower bound path cost for any node provides a lower bound for the additional cost of going through that particular node on a path to the lowest level. The algorithm identifies a feasible solution that will remain the unique optimal solution. Once this optimal set of assignments is identified, random costs are applied to each of the assignments in the optimal set. Then working from level $n_1$ to the highest level of the tree, apply assignment costs, $c_{ijk}$, and lower bound path costs, $lb_{ijk}$, to each node (except for those on the optimal path that already have these costs assigned).

The following are procedural steps for generating an MAP of controllable size with a known unique optimal solution.

Step 1: Based on the dimension and number of elements for each alternative, build a tree graph of all possible assignments such that

$$c_{i_1\cdots i_d} = \infty,$$

$$lb_{i_1\cdots i_d} = \infty,$$

$$\forall \ i_1, i_2, \ldots, i_d.$$ 

Step 2: Select an optimal path from the root node to a leaf node.

Step 3: Apply random assignment costs that are uniform between some lower and upper values to each node on the optimal path and update their lower bound.
path costs such that

\[
\begin{align*}
    lb_{n_1\ldots i_d} &= c_{n_1\ldots i_d}, \\
    lb_{k\ldots i_d} &= c_{k\ldots i_d} + \min \{ \text{feasible } lb_{k+1\ldots i_d} \}, \quad \forall \ k = 1, \ldots, n_1 - 1.
\end{align*}
\]

Step 4: For each node at level \( n_1 \) that is feasible from the optimal path node at level \( n_1 - 1 \), apply a cost that is at least greater than the cost of the optimal path node at level \( n_1 \).

Step 5: Apply random costs that are uniform between some lower and upper values for the rest of nodes at level \( n_1 \).

Step 6: Work up the tree graph from level \( n_1 - 1 \). For each node at level \( k \), consider all feasible nodes at level \( k + 1 \). Set node’s cost such that its lower bound path cost is at least greater than the lower bound path cost of the optimal path node at level \( k \). That is, set its lower bound path cost such that

\[
    lb_{k\ldots i_d} = c_{k\ldots i_d} + \min \{ \text{feasible } lb_{k+1\ldots i_d} \} > \min \{ lb_{k\ldots i_d} \}, \quad \forall \ k = 1, \ldots, n_1 - 1.
\]

An example of this procedure for a three-dimensional problem follows. Consider the MAP above such that \( n_1 = 3, n_2 = 4 \) and \( n_3 = 4 \).

Step 1: Tree graph is shown in Figure 7–2.

Step 2. An optimal solution set of 141, 222, 334 is chosen by randomly choosing a node at each level.

Step 3. Integer costs uniform in [1,10] are applied to the optimal solution set such that

\[
\begin{align*}
    c_{141} &= 2, \\
    c_{222} &= 1, \\
    c_{334} &= 4,
\end{align*}
\]

thus giving an optimal cost of 7. The updated tree is shown in Figure 7–3.
Figure 7–2: Initial tree graph with assignment costs and lower bound path costs.

Steps 4 and 5. Integer costs uniform in [1,10] are applied to nodes at level 3 such that the lower bound path cost of the optimal path node at level 2 is not reduced.

Step 6. Starting at level 2, apply random costs to each node (other than nodes on the optimal path) such that its lower bound path cost is at least greater than the lower bound path cost of the optimal path node at the same level. Consider Figure 7–4 and the following calculation.

\[ lb_{211} = c_{211} + \min \{ lb_{322}, lb_{323}, lb_{324}, lb_{332}, lb_{333}, lb_{334}, lb_{342}, lb_{343}, lb_{344} \} > lb_{222}, \]
\[ = c_{211} + lb_{323} > lb_{222}, \]
\[ 8 = 7 + 1 > 5. \]

In this case, \( c_{211} \) was randomly determined yet ensuring \( lb_{211} \) exceeded \( lb_{222} \).
Continuing step 6 until all assignment costs are identified results in the MAP shown in Figure 7–5.

7.2.2 Proof of Unique Optimum

The proposed algorithm for generating an MAP results in a known unique optimal solution. Consider an optimal assignment set for an MAP. The path in the tree graph from the root node to the leaf node at level \( n_1 \) provides assignment costs and lower bound path costs at each level in the tree. Note the lower bound path cost of the optimal path node at level 1 is the optimal cost. A proof for maintaining a known unique optimal solution is given by induction as follows. First, the algorithm provides a unique optimal solution for \( n_1 = 1 \) and \( n_1 = 2 \). For \( n_1 = 1 \), it is obvious the algorithm (step 4) results in a unique optimal solution because all assignment costs are greater than the assignment cost for the proposed solution. For \( n_1 = 2 \), the algorithm (step 6) ensures the minimum lower bound path cost at level 1 is maintained at the
Figure 7.4: Tree graph used to consider all feasible nodes at level 3 from the first node in level 2.

node in the optimal assignment set. Since no other feasible path through a node at level 1 to level 2 can have a cost less than the optimal lower bound path cost, optimality is maintained. Now assume the algorithm results in a known unique optimal solution for \( n_1 = k \). By step 6 of the algorithm, the minimum lower bound path cost at any level is maintained at the node in the optimal assignment set. Therefore, the additional level will still maintain the minimum lower bound path cost at a node in the optimal assignment set and a known unique optimal solution results.

### 7.2.3 Complexity

Each node in the tree graph is assigned a cost. In the worst case, assigning a cost to a node means that all nodes on the next level down must be scanned to find feasible nodes and from these feasible nodes the minimum lower bound path cost is determined. Let \( n_1, n_2, \ldots, n_d \) be the number of elements up to dimension \( d \) such that
\[ n_1 \leq n_2 \leq \cdots \leq n_d. \] The total number of possible assignments (nodes in the tree graph excluding the root) is then \( \prod_{i=1}^{d} n_i \), which is \( O(n_d^d) \). The time to determine whether a node at the next level down is feasible is \( O(d) \). Therefore, complexity is \( O(dn_d^d) \).

Using a 2.2 GHz, Pentium 4 machine, the results of timed experimental runs producing different sized MAP test problems using the proposed algorithm are provided in Table 7–1. The results show that large problems with known solutions can be generated in a reasonable amount of time.

### 7.3 MAP Test Problem Quality

Although the proposed algorithm provides an MAP with a known unique optimal solution in manageable time for fairly large problems, questions remain with respect to the quality of the generated problems. For example, is the generated problem realistic, what are the characteristics of the assignments costs and are these problems...
useful in exercising solution methods? Although the definition of problem quality is somewhat subjective, we analyze several important quality characteristics in the following paragraphs.

7.3.1 Distribution of Assignment Costs

One measure of whether the generated MAP is realistic is the distribution of assignment costs. For several applications of the MAP, such as facility location assignments, one may reason that assignment costs are normally distributed. A normal probability plot is shown in Figure 7–6. This plot is that of a typical set of assignment costs from a generated test problem using the proposed algorithm. The plot indicates the costs may be normally distributed. Using chi-square goodness-of-fit, an analysis of six randomly selected 5x5x5 test problems that were generated using the proposed algorithm yielded the results shown in Table 7–2. Using a p-value statistic of 15-percent, we conclude assignment costs are normally distributed for five of the six cases. The goodness-of-fit test for run four does not indicate the assignment costs are normally distributed.

For larger MAP test problems, Chi-square goodness-of-fit tests failed to confirm that the assignment costs are normally distributed. However, as shown in Figure 7–7, a typical histogram of assignment costs for a 20x30x40 test problem shows the costs appear to be well behaved.

Table 7–1: Timed results of producing test problems of various sizes.

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Total Number of Nodes</th>
<th>Approximate Number of Feasible Solutions</th>
<th>Machine Run Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20x30x40</td>
<td>$2.4 \times 10^4$</td>
<td>$2.45 \times 10^{65}$</td>
<td>2</td>
</tr>
<tr>
<td>50x50x50</td>
<td>$1.25 \times 10^5$</td>
<td>$9.25 \times 10^{128}$</td>
<td>67</td>
</tr>
<tr>
<td>60x70x70</td>
<td>$2.94 \times 10^4$</td>
<td>$2.45 \times 10^{187}$</td>
<td>309</td>
</tr>
<tr>
<td>6x7x8x9x9</td>
<td>$2.7 \times 10^4$</td>
<td>$3.72 \times 10^{17}$</td>
<td>52</td>
</tr>
<tr>
<td>9x9x9x9x9</td>
<td>$5.9 \times 10^4$</td>
<td>$1.73 \times 10^{22}$</td>
<td>170</td>
</tr>
<tr>
<td>10x10x11x11x12</td>
<td>$1.33 \times 10^5$</td>
<td>$2.31 \times 10^{29}$</td>
<td>809</td>
</tr>
</tbody>
</table>
Figure 7–6: Typical normal probability plot for a 5x5x5 test problem.

Table 7–2: Chi-square goodness-of-fit test for normal distribution of assignment costs for six randomly selected 5x5x5 test problems.

<table>
<thead>
<tr>
<th>Run</th>
<th>Assignment Costs</th>
<th>Goodness-of-Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Standard Deviation</td>
</tr>
<tr>
<td>1</td>
<td>13.18</td>
<td>5.95</td>
</tr>
<tr>
<td>2</td>
<td>13.13</td>
<td>5.53</td>
</tr>
<tr>
<td>3</td>
<td>11.53</td>
<td>5.60</td>
</tr>
<tr>
<td>4</td>
<td>12.21</td>
<td>4.93</td>
</tr>
<tr>
<td>5</td>
<td>10.10</td>
<td>5.30</td>
</tr>
<tr>
<td>6</td>
<td>13.86</td>
<td>6.10</td>
</tr>
</tbody>
</table>

7.3.2 Relative Difficultly of Solving Test Problems

To answer the question of whether the algorithm produces test problems that are useful for exercising heuristic and exact solution methods, one should consider characteristics of the proposed test problem such as number of local minima and relative difficulty in solving the MAP.

The number of local minima of an MAP has implications for heuristics that rely, at least partly, on repeated local searches in neighborhoods of feasible solutions [112]. In our analysis the neighborhood of a feasible solution is defined as all 2-element exchange permutations [74]. Using this definition, the size of the neighborhood is
If \( \hat{x} \) is a feasible assignment set of the MAP and \( f(\hat{x}) \) is the solution, then \( \hat{x} \) is a local minimum iff \( f(\hat{x}) \leq f(\hat{y}) \) for all \( \hat{y} \) in the neighborhood of \( \hat{x} \). Complete enumeration of local minima is difficult for large problems; however, using a proportionate sampling approach [65] categories of the MAP may be compared. The proposed sampling approach is to randomly select a sample of feasible solutions and determine the fraction of the sample that are local minima. A 95-percent confidence interval on the fraction of local minima to total number of solutions may be calculated. Table 7–3 compares the number of local minima per \( 10^6 \) feasible solutions for various problem sizes and categories. The first set contains problems generated from the proposed algorithm. The next set, Category I, contains problems generated with integer random assignment costs that are uniform on [1,25]. Category II, using a three-dimensional problem as an example, are problems generated with \( c_{ijk} = a_{ij} + a_{ik} + a_{jk} \), where \( a_{ij}, a_{ik}, \text{ and } a_{jk} \) are randomly generated integers from a uniform distribution on [1,7]. These parameters were chosen based on approximate spread of minimum and maximum assignment costs for test problems generated by the proposed algorithm.
Analysis of results in Table 7–3 suggests the number of local minima for the generated MAP is comparable to other MAPs with different assignment costs structures. Another interesting aspect is it appears problems with relatively small $d$ and large $n_1$ have a smaller fraction of local minima. This suggests that algorithms using local search techniques compared to those that do not may converge to a global minimum faster for these type problems. However, additional research is needed concerning the number of local minima for MAPs and its impacts on different solution methods.

Table 7–3: Number of discrete local minima per $10^6$ feasible solutions. The range is a 95-percent confidence interval based on proportionate sampling.

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Test Problem</th>
<th>Category I</th>
<th>Category II</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3^9$</td>
<td>89,800 to 250,120</td>
<td>82,230 to 237,770</td>
<td>52,510 to 187,490</td>
</tr>
<tr>
<td>5x6x7x8</td>
<td>761 to 11519</td>
<td>9036 to 13164</td>
<td>7501 to 11299</td>
</tr>
<tr>
<td>6x7x8x9x10</td>
<td>501 to 819</td>
<td>518 to 842</td>
<td>422 to 718</td>
</tr>
<tr>
<td>9x10x10</td>
<td>0.4 to 0.7</td>
<td>0.2 to 1.2</td>
<td>0.0 to 0.8</td>
</tr>
</tbody>
</table>

Considering the relative difficulty in solving the proposed test problems, two different experiments were run. The first experiment measured the time to solve the different problem categories using a branch-and-bound exact solution algorithm as suggested by Pierskalla [87]. The experiment was run on a 2.2 GHz, Pentium IV machine. Five runs were conducted on each of the MAP sizes 9x10x10 and 5x6x7x8 for the categories described above for a total of 15 runs. The results as shown in Table 7–4 indicate the proposed test problems take longer to solve than the randomly generated problems. The second experiment used a version of GRASP to solve 20x30x40 and 6x7x8x9x10 MAPs as described above. Unlike in the first experiment where time-to-solve was used as a measure of difficulty, this experiment fixed the time the algorithm was allowed to run and then compared the resulting solution with the optimal solution. Five experiments were run on each problem category and size. The results in Table 7–5 show that, for this solution method, the proposed test
problems are more difficult to solve. These experiments indicate the test problems would be useful in exercising at least some exact and non-exact solution methods.

Table 7–4: Comparison of solution times in seconds using an exact solution algorithm of the branch-and-bound variety.

<table>
<thead>
<tr>
<th>Run</th>
<th>9x10x10</th>
<th>5x6x7x8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test Problem</td>
<td>Cat I</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>4</td>
<td>44</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>Mean</td>
<td>33</td>
<td>&lt; 1</td>
</tr>
</tbody>
</table>

Table 7–5: Comparison of solution results using a GRASP algorithm.

<table>
<thead>
<tr>
<th>Average percentage difference from optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>20x30x40</td>
</tr>
<tr>
<td>Test Problem</td>
</tr>
<tr>
<td>Category I</td>
</tr>
<tr>
<td>Category II</td>
</tr>
<tr>
<td>6x7x8x9x10</td>
</tr>
<tr>
<td>Test Problem</td>
</tr>
<tr>
<td>Category I</td>
</tr>
<tr>
<td>Category II</td>
</tr>
</tbody>
</table>

7.4 Test Problem Library

Approximately thirty MAP test problems are available for download at http://www.math.ufl.edu/coap/.

Also available is a C++ code of the MAP test problem generator.

7.5 Remarks

Generating MAPs of controllable size with a known unique solution is important for testing exact and non-exact solution methods. This chapter describes a method to develop test problems with a known unique solution. Developing the technique to generate these test problems is important, but the conditions of a quality MAP remain subjective. This chapter examined a few characteristics of quality such as
distribution of assignment costs, number of local minima and difficulty to solve. Based on these few characteristics, the generated MAP test problems appear to be realistic and challenging for exercising exact and heuristic solution methods.
CHAPTER 8
CONCLUSIONS

The work in this dissertation examined combinatorial problems from a probabilistic approach in an effort to improve existing solution methods or find new algorithms that perform better. A probabilistic analysis of combinatorial problems is a very broad subject; however, the context here is the study of input data and solution values.

We investigated characteristics of the mean optimal solution values for random multidimensional assignment problems (MAPs) with axial constraints. In the cases of uniform and exponential costs, experimental data indicates that the mean optimal value converges to zero when the problem size increases. In the case of standard normal costs, experimental data indicates the mean optimal value goes to negative infinity with increasing problem size. Using curve fitting techniques, we developed numerical estimates of the mean optimal value for various sized problems. The experiments indicate that numerical estimates are quite accurate in predicting the optimal solution value of a random instance of the MAP.

Our experimental approach to the MAP can be easily extended to other hard problems. For example, solution approaches to the QAP may benefit from numerical estimates of the optimal values. Additionally, future work is needed using real-world data. Other interesting work includes closer study of the numerical models. It is clear the parameters asymptotically approach particular values. Questions remain on what these values are and why they exist.

Further research and thought are needed to see how the numerical estimates of the mean optimal values can be used to improve existing solution algorithms or developing new algorithms.
Using a novel probabilistic approach, we proved the asymptotic characteristics of the mean optimal costs of MAPs. Further work is needed to develop and prove more global theorems on the asymptotic characteristics of combinatorial problems. In the example of the MAP, it appears the lower support of the parent distribution has some bearing on the mean optimal costs.

We investigated the expected number of local minima for random instances of the MAP and reported on their impact on three different solution algorithms that rely on local neighborhood searches. We also provided a closed form relationship for the average number of local minima in a special case of the MAP. We provided bounds on the average in more general cases of the MAP. More work in needed in this area. For example, an interesting study is to consider the distribution of local minima across the solution landscape and the distance between these local minima. An answer to this question may lead to novel solution approaches.

A probabilistic approach was used to develop an MAP test problem generator that creates difficult problems with known unique solutions. Test problem generators are often very useful to researchers. Additional work is necessary to create other generators and to use a probabilistic approach to ensure the generators produce hard problems that are useful in exercising solution algorithms.

Finally, continued exploitation of dual-use applications (military and civilian) is of great interest. Cross-fertilization of ideas benefits practitioners in all areas of research.
REFERENCES


A. Beck, “On the number of local maxima for the max-cut and bisection problems,” School of Mathematical Sciences, Tel-Aviv University, June 2003.


BIOGRAPHICAL SKETCH

The author, Don A. Grundel, was born in Fresno, California, in 1963. He grew up in Ocala, Florida, where he met his lovely wife Bonnie. He and his wife have two wonderful children, Andrew and Erin. He graduated from the University of Florida in 1986 with a Bachelor of Mechanical Engineering and went to work at Eglin AFB, Florida, as a design and construction engineer for the base’s civil works department. In 1994, he obtained an MBA from the University of West Florida. He went back to school at the University of Florida, Graduate Engineering and Research Center and obtained a master’s in industrial and systems engineering in 2001. He earned his PhD in August 2004.