MINIMUM-RISK ROUTING THROUGH A MAPPED MINEFIELD

by

Christopher D. Odom

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Mineral and Energy Economics).

Golden, Colorado
Date 4/11/2011

Signed: Christopher D. Odom
Christopher D. Odom

Signed: Alexandra M. Newman
Dr. Alexandra Newman
Thesis Advisor

WOOD.ROGER.KEVIN.1231652287
Digitally signed by WOOD.ROGER.KEVIN.1231652287
DN: c=US, o=U.S. Government, ou=DoD, ou=PKI, ou=USN, cn=WOOD.ROGER.KEVIN.1231652287
Date: 2011.04.10 12:08:05 -07'00'

Signed: _____
Dr. Kevin Wood
Thesis Advisor

Golden, Colorado
Date 4/11/2011

Signed: R G Eggert
Dr. Rod Eggert
Professor and Division Director
Division of Economics and Business

ABSTRACT

The ability to autonomously map sea minefields presents navies with the opportunity to route a ship more safely through a minefield. We embed a directed graph in a representation of a minefield area. Vertices represent waypoints and edges denote possible segments for ship transit. We present a model to identify an *s-t* path that minimizes the probability that a ship incurrs unacceptable damage. Standard shortest-path based methods employ an "edge-additive model" that may over-accumulate the total risk associated with a path and yield poor solutions. Our "threat-additive model" avoids these problems. The solution methods include (a) two integer programs (IPs) implemented in commercial optimization software, and (b) a novel path-enumeration algorithm, **MRPA** (Minimum-Risk Path Algorithm), implemented in C++. **MRPA** identifies an initial solution and bounds the optimal solution before implicitly enumerating paths. Computational results indicate that **MRPA** solves problems faster than standard IP software in 80% of the 35 test problems we consider. These problems contain 963 vertices, 2792 edges, and 30-35 threats.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Area of Operations . . . . . AO

Branch and Bound . . . . . BB

Closest Point of Appraoch . . . . . CPA

Implicit Path-Enumeration Algorithm . . . . . IPA

Integer Program . . . . . IP

Integrated Development Environment . . . . . IDE

Mine Countermeasures . . . . . MCM

Minimum-Risk Path Algorithm . . . . . MRPA

Near-Minimum Risk Path . . . . . NMRP

Near-Shortest Simple Paths . . . . . NSSP

Unmanned Aerial Vehicle . . . . . UAV

Unmanned Underwater Vehicle . . . . . UUV

Unrestricted Near-Shortest Paths . . . . . UNSP

ACKNOWLEDGEMENTS

Foremost, I would like to thank my wife for her adoring devotion, for her caring support, and for the countless hours she sacrificed so I could work on this thesis. I would like to thank my advisors, Professors Kevin Wood and Alexandra Newman. A special thanks goes to Professor Wood for his suggestion that I undertake this research topic and for his idea to modify an implicit path enumeration algorithm for my specific application. And to Professor Newman, I am grateful for all of her helpful comments on my thesis and for all of the concepts I learned in her six classes.

CHAPTER 1

INTRODUCTION

Unmanned underwater vehicles (UUVs) can map minefields. This provides navies with the opportunity to use a model whose solution would allow a ship to transit a mapped minefield more safely. We embed a directed graph in a representation of a minefield area. Vertices represent waypoints and edges denote possible segments for ship transit. Using a "threat-additive" model that is more accurate than a standard "edge-additive" model, we identify a minimum-risk route, i.e., an s-t path that minimizes the probability of a ship incurring unacceptable damage. The solution methods include (a) two integer programs implemented in commercial optimization software, and (b) an implicit path-enumeration algorithm implemented in C++.

1.1 Problem Statement

Minimizing the risk of threat detection is a recurring objective when optimizing routes for military vehicles (Murphey et al., 2003). We define the minimum-risk routing problem as follows: A vehicle must transit from point s to point t in a convex, two-dimensional region, i.e., the area of operations (AO). We embed a directed graph $G(V, E)$ in a representation of this region where vertices V represent waypoints and edges E denote possible segments for ship transit. We refer to this graph as the *AO network*. A two-dimensional AO suffices because a surface ship is restricted to maneuvering on a planar surface. A set of threats exists in the plane, or may be viewed as existing in the plane, and we wish to identify an s-t path that minimizes risk, such a route is *optimal*. Hereafter, we refer to a vehicle as a ship and a threat as a mine.

Risk from a mine to a ship is represented mathematically by a risk function. The risk function we use computes risk from the probability of mine actuation. An actua-

tion curve, which transforms the *closest point of approach* (CPA) into the probability of mine actuation, calculates this probability. The closest point of approach is the minimum distance between a mine and each edge that falls within the *damage radius* R_m of mine m. The damage radius, R_m, is the distance beyond which a ship no longer incurs damage from a mine even if it explodes. A mine *impinges* upon an edge in the AO network if the CPA between the edge and the mine is less than R_m. We refer to a circle defined by the damage radius as a *threat circle.*

For initial explanatory purposes, we assume a Cookie-Cutter risk function (Eagle, 2008). If a ship transits mine m's threat circle, the ship incurs risk r_m, and 0 otherwise. This function causes the risk associated with mine m to be homogeneous on every edge in m's threat circle. In Chapter 3, we use a different risk function, which allows positive risk from a mine to differ by edge.

An *edge-additive (risk) model* accumulates risk as follows: Let M_e be the set of mines that impinge upon edge e. Risk associated with an edge e is $\sum_{m \in M_e} r_m$. The total risk associated with transiting an s-t path is $\sum_{e \in E_{s\text{-}t}} \sum_{m \in M_e} r_m$, where $E_{s\text{-}t}$ is the set of edges on the s-t path. A shortest-path algorithm, which is computationally inexpensive, can identify a solution to an edge additive-model. However, an edge-additive model may over-accumulate risk when a mine impinges upon two or more edges on a path. (See Section 3.2.1 for details on the logarithmic transformation that linearizes the product of the non-actuation probabilities to derive the additive risk measures we use here.)

A *threat-additive (risk) model* accumulates risk from threats instead of edges. Let $M_{s\text{-}t}$ be the set of mines on an s-t path. Total risk is computed by summing the risk from each mine that impinges upon the s-t path; thus, $r^T(E_{s\text{-}t}) = \sum_{m \in M_{s\text{-}t}} r_m$ is the "true", i.e., threat-additive, risk associated with path $E_{s\text{-}t}$. An optimal path from the threat-additive model minimizes the probability of a ship incurring unacceptable damage, assuming independence among actuation events, while the path obtained

from the edge-additive model will not, except under special circumstances.

1.2 Motivation

Mines can cause damage to U.S. naval assets, especially in littoral, i.e., shallow-water, regions that must be traversed to accomplish a mission such as an amphibious landing. In Desert Storm, the Iraqi military laid over 1150 mines which prohibited any amphibious landing on Kuwait's coast (Cornish, 2003). This minefield also damaged the *USS Tripoli* LPH-10 and the *USS Princeton* CG-59, causing $27.5 million of damage (CNO, 1991; Cornish, 2003). The shallow water in potential conflict areas such as the Persian Gulf, the Strait of Hormuz, the Red Sea, the Yellow Sea, and the Korean Strait makes these locations excellent for mine laying: mines could prevent an amphibious landing in a future conflict (Cornish, 2003).

In order for the U.S. Navy to maintain its current ability to successfully navigate minefields, new methods of mine warfare must compensate for the aging and down-sizing of the United States mine warfare fleet. Polmar et al. (2005) show that beginning in the 1990s, there have been attempts to discard most Mine Countermeasures (MCM) ships and the MH-53E Sea Dragon MCM helicopters. No new MCM ships have been built since 1991. The lack of replacements for the aging MCM ships implies that frigates, cruisers, destroyers, and amphibious ships must perform more MCM missions.

The U.S. Navy's ability to traverse minefields safely must increasingly rely on mine avoidance, through minefield mapping, instead of *minesweeping*, i.e., the clearance of mines. Although currently ill-equipped for mine warfare missions such as minesweeping, frigates, cruisers, destroyers, and amphibious ships can carry unmanned underwater vehicles that are *mine-hunting* capable, i.e., capable of locating and classifying mines (Morris, 1997). Using UUVs and other mine warfare equipment carried by surface ships and submarines is referred to as *organic MCM*. Organic MCM give existing

ships the ability to conduct their own MCM missions through minefield mapping and optimal routing. Optimal routing through a mapped minefield minimizes the risk to a ship, that is, provides a route with the highest probability of survival.

This thesis limits itself to the study of routing to minimize risk, and does not consider the potential for reducing risk that mine clearance could add. Button et al. (2009) show that UUVs have already demonstrated mapping capabilities for mine detection. However, clearance technology is still under development. Muljowidodo et al. (2009) propose expending a remotely operated UUV for mine disposal by ramming into a mine. This technology has been installed in minesweepers in Europe. Muljowidodo et al. also suggest using UUVs to deliver a payload of explosives to the mine, allowing for repeated use of the UUV in mine clearance. Mine clearance technology needs further development to reach the same technological level as minefield mapping. Given that the current UUV technology is best suited for minefield mapping and not clearance, this thesis concentrates on constructing a model whose solution would allow a ship to transit a mapped minefield more safely without the opportunity to clear any mines.

1.3 Thesis Contributions

This thesis develops a model, and two accompanying solution methods, to identify a minimum-risk route for a ship traversing a mapped minefield. Using a threat-additive model that is more accurate than an edge-additive model, we identify a minimum-risk route, i.e., an s-t path that minimizes the probability of a ship incurring unacceptable damage. We find solutions through the implementation of integer programming formulations in commercial optimization software and a novel implicit path enumeration algorithm, **MRPA** (Minimum-Risk Path Algorithm), implemented in C++. **MRPA** solves problems faster than standard IP software in 80% of the 35 test problems we consider and is useful for practical applications in which near-real-

time use is important.

1.4 Thesis Outline

The structure of this thesis is as follows: Section 2 describes previous work on minimum-risk routing for ships traversing a mapped minefield, and related topics. Section 3 describes the model we use to identify minimum-risk routes. Section 4 presents a novel solution method for a threat-additive model, an implicit path-enumeration algorithm, denoted **MRPA**. Section 5 provides computational results. Section 6 concludes.

CHAPTER 2

MINIMUM-RISK ROUTING FOR SHIPS AND AIRCRAFT: REVIEW

This chapter reviews the literature on identifying a minimum-risk route for a ship to transit through a mapped minefield and for an aircraft to traverse a threat environment. Aircraft routing under certain threat scenarios and ship routing to minimize risk from sea mines have many similarities. The modeling approaches employed to solve these two problems frequently overlap and can be classified as either continuous or discrete.

A number of continuous models have been proposed for minimum-risk routing, e.g., Inanc et al. (2008), Karelahti et al. (2008), Mercer and Sidhu (2007), and Zabarankin et al. (2002). These models have theoretical and computational drawbacks (see Inanc et al., 2008; Ruz et al., 2006; Zabarankin et al., 2006, 2002). Because discrete models avoid these difficulties and their solutions can be made arbitrarily close to the solutions from continuous models (Kim and Hespanha, 2003; Muhandiramge et al., 2009), the literature indicates that discrete models are better suited for practical use. Consequently, this thesis uses a discrete model.

Bekker and Schmid (2006), Boerman (1994), and Li (2009) use a discrete approach, embedding a directed graph in a representation of the minefield area, and accumulate risk according to an edge-additive model. Li (2009) attempts to mitigate risk over-accumulation by using short and long edges. While these edges may ameliorate risk over-accumulation because they provide greater flexibility in the routing structure, Li demonstrates that the use of short and long edges does not completely prohibit risk over-accumulation. This fact motivates the threat-additive model that this thesis develops.

Another method that mitigates risk over-accumulation only applies to a specific case. Carlyle et al. (2009) explain that the severity of risk over-accumulation is modest

when the risk is based on dependent, yet small, probabilities. Although the military expects to encounter situations in which risk is based on a small probability, these situations cannot be guaranteed. A more accurate risk model is necessary to account for this contingency.

We address the modeling omissions described above by developing a threat-additive model that accurately accumulates risk to identify a minimum-risk route for a ship traversing a mapped minefield. This improvement provides mission commanders with superior information about the true risk on a path and may identify lower-risk paths. This thesis develops and tests two solution approaches, an integer program and an implicit path enumeration algorithm, for the threat-additive model. The path-enumeration algorithm may be important for practical applications in which near-real-time use and independence of licensed optimization software are necessary.

CHAPTER 3

MODELING A MINIMUM-RISK ROUTE IN A MAPPED MINEFIELD

This chapter describes the modeling approach we use to plan a route for a ship to transit through a mapped minefield. Section 3.1 describes the AO network structure and assumptions. Section 3.2 presents the probabilistic objective function and an edge-additive, shortest-path formulation. The probabilistic objective function maximizes the probability of survival. This maximization is equivalent to minimizing the probability a ship incurs unacceptable damage. We use this shortest-path formulation to explain a shortcoming of edge-additive models: risk over-accumulation when a mine impinges upon two or more edges on a path. Section 3.3 describes several threat-additive integer programs, which better approximate the risk on a path. Initially, we represent risk from a mine with a Cookie-Cutter risk function. We replace this risk function with a Triangle actuation curve, which causes positive risk from a mine to differ by edge.

3.1 Network Structure and Assumptions

We use the same basic set of problem specifications and assumptions as Li (2009) to model risk on an s-t path. We assume that there are no ship navigational errors, that the exact location and characteristics of each mine are known, that no modifications are made to the minefield once it is mapped, and that the minefield area represents an entrance to a harbor. The entrance to this harbor is typically rectangular with a length and width of several thousand yards (Li, 2009).

We embed a directed graph, $G = (V, E)$, in a representation of the minefield area to construct the AO network. Vertices, $v \in V$, represent waypoints at which the ship has the opportunity to change its direction, and edges, $e \in E$, denote possible segments for ship transit. The area of operations contains the portion of the minefield

that is traversable by a ship. The remainder of the minefield is not traversable due to shallow water. A ship traverses a simple s-t path, which is specified as a subset of the edges in E. A ship travels from left to right and can enter and exit the minefield at any vertex in the respective, leftmost and rightmost column of vertices. Any vertex spacing less than the minimum turn radius is impractical due to the lack of maneuverability for a large ship. Vertex spacing that is too large fails to account for all possible routes. To guarantee that the solution we find accurately approximates the solution to a continuous model, the angles between neighboring vertices must approach zero (Kim and Hespanha, 2003; Muhandiramge et al., 2009). Decreasing the vertex spacing is one approach that reduces these angles. However, the vertex spacing in the network we construct is limited to the minimum turn radius of the ship transiting the route. We use the same vertex spacing as Li (2009), which corresponds to the minimum turn radius of an 800 ton ship.

Although the magnitude of ship damage can vary with distance, ship type, or any other factor, we assume that if a mine actuates, a ship incurs unacceptable damage and the mission is a failure. Consequently, the probability of mine actuation models the likelihood a ship incurs damage that prevents mission accomplishment. We assume that only one ship transits the specified minefield, that each mine has at most one chance to actuate because of the network structure, and that previously encountered mines do not influence the actuation probability of a mine. The assumption that the sensors within different mines do not communicate leads to mine-to-mine independence; that is, each mine actuation is an independent event. We wish to maximize the probability that no mines actuate for a given path. Independent actuation events permit us to take the product of the probabilities of the non-actuation events.

3.2 Edge-Additive Model: Shortest-Path Formulation

This section describes an edge-additive model, which is formulated with a shortest-path approach, similar to the approaches that Bekker and Schmid (2006) and Li (2009) use. We assume a Cookie-Cutter risk function. Section 3.2.1 explains how we linearize the probabilistic objective function that maximizes the probability of survival, potentially over-accumulating it. Section 3.2.2 presents the formulation, SP. We use SP to demonstrate risk over-accumulation and to compare its solutions with those obtained from the threat-additive models, which accurately accumulate risk, in Section 5.

3.2.1 Probabilistic Objective Function: Logarithmic Transformation

This sections applies the logarithmic transformation to the objective function we use in the edge-additive model in Section 3.2.2. We assume a Cookie-Cutter risk function, which represents risk from a mine as a fixed cost (Eagle, 2008). Let p_m represent the probability of mine actuation if a ship transits the threat circle. To avoid trivial solutions, p_m is greater than 0 and less than 1. The objective function given below does not accumulate the fixed charge p_m for each mine a ship encounters. Rather, the objective accumulates p_m on each edge a ship traverses in mine m's threat circle, and the independence assumption among actuation events, fails when a mine impinges upon multiple edges on a path.

DEFINITIONS

- M_e the set of mines that impinges upon edge e
- $E_{s\text{-}t}$ the set of edges on an s-t path
- p_m the probability of actuation of mine m (Cookie-Cutter risk function)
- x_e a binary variable: 1 if edge e is on the optimal s-t path, 0 otherwise

We wish to determine a simple s-t path through a mapped minefield that maximizes a ship's probability of survival. To determine this probability, we take the product of the non-actuation probabilities of the mines a ship encounters on an s-t path, as shown by Objective (3.1).

$$\max_{E_{s\text{-}t} \in E} \prod_{e \in E_{s\text{-}t}} \left(\prod_{m \in M_e} (1 - p_m) \right) \tag{3.1}$$

Objective (3.1) can be incorporated into a formulation that maximizes this expression over the variables x_e subject to flow balance Constraints (3.2). Let A be an $n \times m$ matrix where n is the number of vertices and m is the number of edges. The values in each row are 1 if an edge is incident to the vertex, -1 if an edge emanates from the vertex, and 0 otherwise. Let b be an $n \times 1$ vector that equals -1 at vertex s, 1 at vertex t, and zero otherwise.

$$A\mathbf{x} = b \tag{3.2}$$

The use of Constraints (3.2) permits the maximization of Objective (3.1) over $\mathbf{x}$, the vector of decision variables, with the variables x_e appearing in the exponent. This expression is Objective (3.3).

$$\max_{\mathbf{x}} \prod_{e \in E} \left(\prod_{m \in M_e} (1 - p_m) \right)^{x_e} \tag{3.3}$$

Taking the product of the non-actuation probabilities in Objective (3.3) results in a nonlinear objective function, which we linearize by applying a logarithmic transformation (Shorack, 1964) to derive Objective (3.4).

$$\max_{\mathbf{x}} \sum_{e \in E} \left(\sum_{m \in M_e} \ln(1 - p_m) \right) x_e \tag{3.4}$$

Because p_m is a probability, the expression $\ln(1 - p_m)$ is always non-positive. For ease of implementation in a standard shortest-path algorithm, we transform the objective from maximization to minimization by multiplying Objective (3.4) by a

negative sign. For simplification, let $r_m = -\ln(1 - p_m)$ be the risk associated with the impingement of mine m on an edge.

$$\min_{\mathbf{x}} \sum_{e \in E} \left(\sum_{m \in M_e} r_m \right) x_e \tag{3.5}$$

The antilog of a negated objective function value is the probability of survival for a ship transiting a path, $P(Survival) = e^{(-Objective\ Value)}$.

Without a penalty for distance, the optimal route may be irregularly shaped, and consequently, the route is incompatible with the maneuverability of a ship. To find routes conducive for ship transit, we apply a small distance penalty to the objective (Li, 2009).

3.2.2 Shortest-Path Formulation (SP)

SP is a "shortest-path" formulation. The probability of mine actuation for each edge a mine impinges upon is p_m. (Note: For computational testing in Section 5, we use p_{em} instead of p_m to account for positive risk that differs by edge.) SP is an edge-additive model that over-accumulates risk if a ship traverses two or more edges in a threat circle. The formulation is given below.

Constraints (3.2) maintain the continuity of a path. The "lengths" on each edge are non-negative: r_m is greater than or equal to zero because $p_m \in (1, 0]$. We do not define p_m at 1 because the natural log of 1 is negative infinity; instead, we use an approximation of $1 - \epsilon$ for 1, where ϵ is the same parameter we use for the distance penalty in the objective. Non-negative edge "lengths" permit this formulation to be solved by a simple shortest-path algorithm such as Dijkstra's algorithm.

PARAMETERS

d_e length of edge e

$r_m = -\ln(1 - p_m)$ risk associated with the impingement of mine m on an edge

ϵ edge length penalty ($\epsilon = 10^{-7}$ in this thesis)

OBJECTIVE FUNCTION

$$\min_{\mathbf{x}} \quad \sum_{e \in E} \left(\sum_{m \in M_e} r_m \right) x_e + \epsilon \sum_{e \in E} d_e x_e \tag{3.6}$$

CONSTRAINTS

$$\text{s.t. (3.2)}$$
$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{3.7}$$

This shortest path formulation has no structure that precludes the same mine from impinging upon multiple edges in the optimal path, $E^*_{s\text{-}t}$. If a mine impinges upon two or more edges in $E^*_{s\text{-}t}$, then this formulation over-accumuates risk. Let $m \in M_e$ and $m' \in M_{e'}$, where $e, e' \in E^*_{s\text{-}t}$ such that $e \neq e'$. If $m = m'$, the formulation accumulates risk as follows: $r_m + r_{m'} = 2r_m$; thus, this sum double counts the risk from a mine on a path.

3.3 Threat-Additive Models: Integer Programming Formulations

This section describes the threat-additive models that are integer programs, which better approximate the true risk on a path. Section 3.3.1 describes a formulation, TMR0, that uses a Cookie-Cutter risk function. Typically, risk from a mine is not the same on every edge it impinges upon. We redefine p_m as p_{em}, the probability of mine actuation m on edge e. Section 3.3.2 presents the Triangle actuation curve that computes this parameter by transforming the CPA into the probability of mine actuation on edge e. Subsequently, this new parameter appears in two integer programming formulations, TMR1 and TMR2, in which positive risk from a mine differs by edge. TMR1 and TMR2 are the two integer programs we use for computational testing in Chapter 5. Section 3.3.3 develops an integer program TMR1, a min-max formulation. Section 3.3.4 presents a formulation with a potentially tighter LP relaxation than TMR1. The probabilistic objective functions presented in this section are

specified in their linear form, after we apply the logarithmic transformation as given in Section 3.2.1.

3.3.1 Threat-Additive Minimum-Risk Formulation 0 (TMR0)

The Threat-Additive Minimum-Risk Formulation (TMR0) uses a Cookie-Cutter risk function. The variables y_m ensure that if a ship transits mine m's threat circle, the objective incurs the penalty r_m. We do not conduct any computational testing on TMR0. Constraints (3.2) maintain the continuity of a path. Constraints (3.9) ensure that the objective accumulates risk r_m from each mine a ship encounters.

VARIABLES

y_m a binary variable: 1 if a ship traverses mine m's threat circle, 0 otherwise.

OBJECTIVE FUNCTION

$$\min_{\mathbf{x},\mathbf{y}} \sum_{m \in M} r_m y_m + \epsilon \sum_{e \in E} d_e x_e \tag{3.8}$$

CONSTRAINTS

s.t. (3.2), (3.7)

$$x_e \leq y_m \qquad \forall e \in E, m \in M_e \tag{3.9}$$

$$y_m \in \{0,1\} \qquad \forall m \in M \tag{3.10}$$

3.3.2 Heterogeneous Probabilities of Mine Actuation: Triangle Actuation Curve Risk Function

We use a Cookie-Cutter risk function in formulations SP and TMR0 to explain the differences between edge-additive and threat-additive models. However, a Cookie-Cutter risk function is an over-simplication of reality. Positive risk from a mine is not the same on every edge. To account for this, we add an index e to the actuation probability parameter p_m to define the actuation probability of mine m on edge e, p_{em}. A Triangle actuation curve transforms the CPA into the probability of mine

actuation on edge e. (Note: We must also add the index e to r_m, which results in $r_{em} = -\ln(1 - p_{em})$.)

Actuation curves assume that the CPA is measured from a mine to a ship. A limitation of discrete models is that the CPA, d_{em}, is measured from mine m to an edge e. Accumulating the actuation probabilities on each edge a ship traverses in a mine's threat circle does not accurately capture the true CPA, i.e., the CPA from a mine and to a ship. A threat-additive model accurately approximates the true risk on a path by accumulating the maximum risk, which corresponds to the CPA between a mine and a ship.

To determine p_{em}, we use a Triangle actuation curve for the risk function (Eagle, 2008). This actuation curve, $A_m(d_{em})$, renders the probability of mine m actuating for a given CPA, denoted by d_{em}. This actuation curve, Figure 3.1, identifies the new parameters p_{em} $\forall e \in E, m \in M_e$.

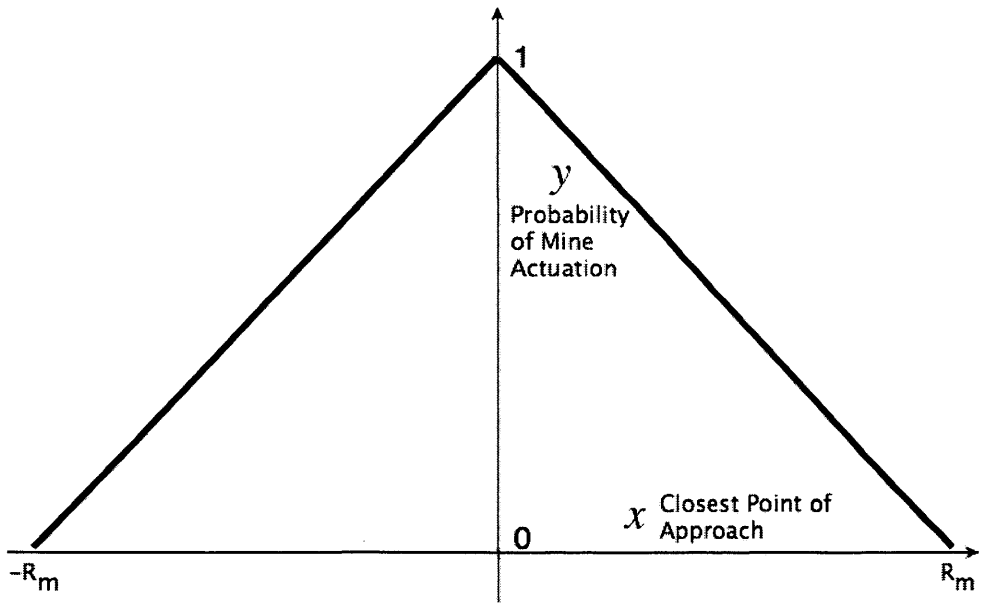

Figure 3.1: Triangle actuation curve. The x-axis is the CPA (d_{em}), and the y-axis is the probability of mine actuation (Eagle, 2008).

An actuation curve provides the aggregate probability instead of a cumulative, or an incremental, probability of actuation. This curve approximates the probability

that the sensors in a mine detect a ship and cause mine detonation over a path. Detection and detonation do not necessarily occur at the CPA. Actuation curves assume an infinitely-long, straight-line approach of a ship to a mine, and are thus symmetric about the y-axis. Approaches from any direction that result in identical CPA values yield the same probability of actuation. The solutions we find may violate the straight-line assumption since an optimal path, comprised of short-line segments, may deviate from an infinitely-long, straight line.

$$A_m(d_{em}) = \begin{cases} \frac{R_m - d_{em}}{R_m} & \text{if } d_{em} < R_m \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

3.3.3 Threat-Additive Minimum-Risk Formulation 1 (TMR1)

The Threat-Additive Minimum-Risk Formulation (TMR1) is a min-max formulation that minimizes the maximum risk a mine imposes across the edges on a path. This formulation introduces a new variable, w_m, which represents the maximum risk from mine m. The objective accumulates the variables w_m. Constraints (3.2) maintain the continuity of a path. Constraints (3.13) ensure the correct accumulation of risk by using a min-max construct. The formulation is given below.

ADDITIONAL SETS

$m \in M$ set of all mines m

ADDITIONAL VARIABLES

w_m maximum risk from mine m on an s-t path

OBJECTIVE FUNCTION

$$\min_{\mathbf{w},\mathbf{x}} \quad \sum_{m \in M} w_m + \epsilon \sum_{e \in E} d_e x_e \tag{3.12}$$

CONSTRAINTS

s.t. (3.2), (3.7)

$$w_m \geq r_{em} x_e \qquad \forall e \in E, m \in M_e \tag{3.13}$$

$$w_m \geq 0 \qquad \forall m \in M \tag{3.14}$$

3.3.4 Threat-Additive Minimum-Risk Formulation 2 (TMR2)

Threat-Additive Minimum-Risk Formulation (TMR2) is a formulation with a potentially stronger LP relaxation than the relaxation of TMR1. This formulation accumulates the maximum risk from a mine on a set of edges. TMR2 is related to the model given in Nehme (2009).

E_m is the set of all edges e within mine m's threat circle. We order the set E_m from the largest risk value, $j = 1$, to smallest risk value, $j = |E_m|$. The risk on edge e from mine m, r_{em}, is now represented by r_{k_j}, where $k = em$ and j denotes a location in the ordered set.

$\overline{E}_m$ is the set of all edges e in mine m's threat circle such that $k = em$, $r_{k_j} \geq r_{k_{j+1}}$, $j = 1, ..., |E_m| + 1$, and $r_{k_{|E_m|+1}} = 0$. We define a set $\overline{E}_m$ for each mine. The addition of a second index e to the risk values and the ordering of the edges in a threat circle by risk value result in a new definition for the y_m variables that TMR0 uses.

ADDITIONAL SETS

$e \in E_m$ set of all edges e within mine m's threat circle.

$k_j \in \overline{E}_m$ set of all edges e in mine m's threat circle such that $k = em$ and $r_{k_j} \geq r_{k_{j+1}}$.

ADDITIONAL VARIABLES

y_{k_j} a binary variable: 1 if a ship traverses edge e in m's threat circle ($k = em$) corresponding to index j, 0 otherwise.

TMR2 forms tighter constraints. From the min-max construct of TMR1, recall Constraints (3.13).

$$w_m \geq r_{em} x_e \quad \forall e \in E, m. \in M_e$$

Since indexing over $\{E \cup M_e\}$ is equivalent to indexing over $\{M \cup E_m\}$, these constraints can be written as follows:

$$w_m \geq r_{em} x_e \quad \forall m \in M, e \in E_m \tag{3.15}$$

We incorporate the variables y_{k_j} into the formulation in Constraints (3.16) and (3.17). All the y_{k_j} variables with a larger j index than the index associated with the maximum risk edge are forced to 1. The manner in which Constraints (3.16) and (3.17) assign values to the variables y_{k_j} ensures that the objective function correctly accumulates the maximum risk.

$$x_e \leq y_{k_j} \quad \forall m \in M, k_j \in \overline{E}_m, j = 1, ..., |E_m| \tag{3.16}$$

$$y_{k_j} \leq y_{k_{j+1}} \quad \forall m \in M, k_j \in \overline{E}_m, j = 1, ..., |E_m| \tag{3.17}$$

Each constraint of (3.18) is separable by mine, implying that these constraints can be written with equalities because we minimize the objective function. To derive Objective (3.19), we move Constraints (3.18) into the objective of TMR1, substituting w_m with its equation.

$$w_m \geq \sum_{k_j \in \overline{E}_m, j=1,...,|E_m|} \left(r_{k_j} - r_{k_{j+1}}\right) y_{k_j} \ \forall m \in M \tag{3.18}$$

Objective (3.19) removes multiple counting. For example, a ship traverses edges 1, 2, and 3, and the largest risk is on edge 1. The ordering of the risk values is $r_{k_1} \geq r_{k_2} \geq r_{k_3} \geq 0$. The objective is:

$$(r_{k_1} - r_{k_2})y_{k_1} + (r_{k_2} - r_{k_3})y_{k_2} + (r_{k_3} - 0)y_{k_3}$$

The variables y_{k_1}, y_{k_2}, and y_{k_3} equal 1, resulting in the correctly accumulated objective value of r_{k_1}, i.e., the maximum risk value from all the edges a ship traverses in mine m's threat circle.

OBJECTIVE FUNCTION

$$\min_{\mathbf{x},\mathbf{y}} \quad \sum_{m \in M} \sum_{k_j \in \overline{E}_m, j=1,...,|E_m|} \left(r_{k_j} - r_{k_{j+1}}\right) y_{k_j} + \epsilon \sum_{e \in E} d_e x_e \tag{3.19}$$

Constraints (3.2) maintain the continuity of a path. Constraints (3.16) and (3.17) force the objective function to correctly accumulate risk. Constraints (3.16) ensure that if a ship travels on an edge in a threat circle, the objective counts the risk.

Constraints (3.7) and (3.20) restrict the variables to be binary.

CONSTRAINTS

s.t. (3.2),(3.7), (3.16), (3.17)

$$y_{k_j} \in \{0,1\} \qquad \forall m \in M, k_j \in \overline{E}_m, j = 1, ..., |E_m| \qquad (3.20)$$

CHAPTER 4

IMPLICIT PATH ENUMERATION FOR MINIMUM-RISK ROUTING

A modeling language such as AMPL and an integer programming solver such as CPLEX provide an easy method to model and solve our formulations in Section 3. However, software security accreditation, licensing issues, and memory limitations make this type of solution method undesirable for certain military applications. Accreditation is often at the cost of a company and requires disclosure of its proprietary code, which most companies are reluctant to do. Commercial license managers have difficulty maintaining licenses on multiple machines on different ships at different locations with limited access and connectivity. Although capable of solving a variety of problem types, commercial software is not the most memory efficient software for certain military applications.

We present a novel algorithm that provides an alternative solution method to using the Branch and Bound algorithm that commercial integer programming software implements, and is, therefore, better suited for military applications. We develop an implicit path-enumeration algorithm that determines a minimum-risk path associated with formulations TMR1 and TMR2. We compare our novel algorithm against the Branch and Bound algorithm that CPLEX implements in Chapter 5.

We modify the Byers and Waterman (1984) algorithm, which enumerates all unrestricted near-shortest paths (UNSP). UNSP are paths with "loops" that are within a factor of $1 + \delta$ of the shortest path length, where δ is a small positive number. The algorithm we present implicitly enumerates all simple paths to find a minimum-risk path while ensuring the correct accumulation of risk. We provide the algorithm's pseudocode in Section 4.2, which is modified from Carlyle and Wood (2005). The Carlyle and Wood (2005) algorithm enumerates all near-shortest simple paths (NSSP). For simplicity, we omit distance penalties in the description of our algorithm.

DEFINITIONS

- n_m maximum number of edges that mine m impinges upon in a simple s-t path
- $E_{s\text{-}u}$ the edges in sub-path s-u
- $E_{s\text{-}t}$ the edges in a simple s-t path
- M_{uv} the set of mines that impinge upon edge (u, v)
- M_{su} the set of mines that impinge upon sub-path s-u
- $M_{s\text{-}t} = \cup_{(u,v)\in E_{s\text{-}t}} M_{uv}$ the set of mines associated with path $E_{s\text{-}t}$
- $r_{em} = -\ln(1 - p_{em})$ risk associated with mine m on edge e
- $r_m^T = \max_{e_m \in E_m \cap E_{s\text{-}t}} \{r_{em}\}$ true risk from mine m on path $E_{s\text{-}t}$
- $r^T(E_{s\text{-}t}) = \sum_{m\in M_{s\text{-}t}} r_m$ the true (threat-additive) risk associated with path $E_{s\text{-}t}$
- $r^T(v)$ the true risk on the "current sub-path" $E_{s\text{-}v}$
- $\underline{r}_{uv}$ a lower bound on the true risk of traversing (u, v), subject to the requirement that for any simple s-t path $E_{s\text{-}t}$ in G, $\sum_{(u,v)\in E_{s\text{-}t}} \underline{r}_{uv} \leq r^T(E_{s\text{-}t})$

4.1 Minimum-Risk Path Algorithm (MRPA)

The **MRPA** developed here is a type of Branch and Bound (BB) algorithm or A* search algorithm. **MRPA** employs a heuristic **H** to identify an initial incumbent solution. The true risk associated with the s-t path from the incumbent solution serves as an upper bound, $\bar{r}$. After executing **H**, **MRPA** uses an implicit path-enumeration procedure to update the upper bound $\bar{r}$ while lower bounds $\underline{r}'(v)$ limit enumeration. When **MRPA** identifies a better solution, corresponding to higher probability of survival, the algorithm updates the upper bound and incumbent solution. Backtracking

occurs when the risk on a sub-path, i.e., a path from s to any vertex v such that $v \neq t$, exceeds the upper bound. (Note: An inverse relationship exists between the probability of survival on a path and the upper bound.)

The heuristic **H** works as follows: A backward shortest-path algorithm identifies a shortest path from t to s using edge "lengths" of $\sum_{m \in M_{uv}} \frac{r_{em}}{n_m}$, where n_m bounds the maximum number of edges that mine m impinges upon in an s-t path and M_{uv} is the set of mines that impinge upon edge $e = (u, v)$. Weighting the risk on an edge by the reciprocal of n_m ensures that the "shortest" path from v to t provides a lower bound on the true risk. **H** determines an upper bound, $\bar{r}$, by calculating the true (threat-additive) risk associated with the path $E_{s\text{-}t}$ identified by the shortest-path algorithm.

After employing **H**, **MRPA** begins implicitly enumerating paths. We use the lower bounds and the current upper bound to determine if a sub-path s-u should be extended along edge (u, v). The lower bounds must be recomputed to account for mines that impinge upon the sub-path s-u. M_{su} denotes the set of mines that impinge upon this sub-path. Recomputing the lower bounds requires the use of a backward shortest-path algorithm in which the only mines in the instance are $M \backslash M_{su}$ and all vertices on the current sub-path are forbidden. **MRPA** must recompute $\underline{r}'(v)$ each time M_{su} changes to ensure that $\underline{r}'(v)$ is a valid lower bound.

This procedure for updating that lower bound differs from the algorithms of Byers and Waterman (1984), Carlyle and Wood (2005), and Carlyle et al. (2008). These algorithms never recompute the lower bounds. The Byers and Waterman (1984) algorithm enumerates unrestricted paths, the Carlyle and Wood (2005) algorithm enumerates simple paths, and the Carlyle et al. (2008) algorithm solves constrained shortest-path problems using Lagrangian relaxation and enumeration. In **MRPA**, the work spent recomputing the lower bounds $\underline{r}'(v)$ may or may not be compensated by the work saved during enumeration.

Each time **MRPA** extends a sub-path along edge $e' = (u, v)$ and $M_{uv} \cap M_{su} \neq \emptyset$, we must update r_m^T, the true risk from mine m, if:

$$r_{e'm} > \max_{e \in E_m \cap E_{s\text{-}u}} \{r_{em}\}$$

MRPA replaces the value of $\max_{e \in E_m \cap E_{s\text{-}u}} \{r_{em}\}$ in $r(E_{s\text{-}u})$ with the value of $r_{e'm}$. This replacement procedure ensures that **MRPA** accumulates risk according to a threat-additive model.

4.2 MRPA Pseudocode

ASSUMPTIONS:

- An s-t path exists in G.
- Neither vertex s nor vertex t is threatened by mines.

DESCRIPTION: An algorithm to find a minimum-risk s-t path for a threat-additive model. The format and terminology of this algorithm follow Carlyle and Wood (2005).

INPUT: A directed graph $G = (V, E)$ in adjacency-list format;

$s, t \in V$ with $s \neq t$; list of mines $m \in M$;

list of edges E_m mine m impinges upon $\forall m \in M$;

risk values r_{em} $\forall m \in M, e \in E_m$;

in the adjacency-list input of G, "firstEdge(v)" points to the first edge

in a linked list of edges directed out of v.

OUTPUT: print("Min-risk path is ", $\widehat{E}_{s\text{-}t}$, " and min-risk value for that path is ", $e^{-\bar{r}}$); /* $\widehat{E}_{s\text{-}t}$ is the set of edges that comprises the incumbent solution */

{

$\underline{r}_{uv} \leftarrow \sum_{m \in M_{uv}} \frac{r_{em}}{n_m}$ $\forall (u, v) \in E$

/* Next three steps constitue **H** to find a "good" starting path */

Find a "shortest" s-t path $\widehat{E}_{s\text{-}t}$ with respect to edge lengths $\underline{r}_{uv}$;

$M_{s\text{-}t} \leftarrow \cup_{(u,v)\in\bar{E}_{s\text{-}t}} M_{uv}$;

$\bar{r} \leftarrow \sum_{m\in M_{s\text{-}t}} r_m^T$; /* true risk for heuristic path, i.e., an upper bound */

for(all $v \in V$){ $\underline{r}'(v) \leftarrow$ lower bound on risk from v to t; }

/* The above requires a backward shortest-path solution */

for(all $v \in V$) { nextEdge(v) $\leftarrow$ firstEdge(v); }

vStack $\leftarrow s$;

eStack $\leftarrow \emptyset$;

$r^T(s) \leftarrow 0$; /* $r^T(v)$ is the true risk on the "current sub-path" $E_{s\text{-}v}$*/

$M_{su} \leftarrow \emptyset$; /* These are the mines on the current path */

/* Note: M_{su} may not be a list, but implied by other data structures */

isOnPath(s) $\leftarrow$ TRUE; **for**(all $v \in V - s$){ isOnPath(v) $\leftarrow$ FALSE; }

while(vStack is not empty){

  $u \leftarrow$ vertex at the top of theStack;

  while(nextEdge(u) $\neq \emptyset$) {

    $(u, v) \leftarrow$ the edge pointed to by nextEdge(u);

    increment nextEdge(u);

    if(isOnPath(v) = FALSE and $r^T(u) + \underline{r}_{uv} + \underline{r}'(v) < \bar{r}$ {

      /* Add (u, v) to current path */

      isOnPath(v) $\leftarrow$ TRUE;

      push v onto vStack;

      push (u, v) onto eStack;

      update M_{su} if necessary;

      /* Compute true risk of current path $E_{s\text{-}v}$ */

      $r^T(v) \leftarrow \sum_{m\in M_{s\text{-}v}} r_m^T$;

      if($r^T(v) \geq \bar{r}$) { goto **Backtrack**; }

      if($v = t$){ $\bar{r} \leftarrow r^T(t)$; $\widehat{E}_{s\text{-}t} \leftarrow$ *eStack*; go to **Backtrack**; }

/* We have extended the path "successfully" */

RECOMPUTE $\underline{r}'(v)$ values assuming the only mines in the model are $M \backslash M_{su}$, and assuming that all vertices on the current path are forbidden;

}

}

Backtrack: Pop u from vStack;

isOnPath(u) $\leftarrow$ FALSE;

nextEdge(u) $\leftarrow$ firstEdge(u);

if(eStack is not empty){ pop (v, u) from eStack; }

update M_{su};

RECOMPUTE $\underline{r}'(v)$ values assuming the only mines in the model are $M \backslash M_{su}$, and assuming that all vertices on the current path are forbidden;

}

print("Min risk path is ", $\widehat{E}_{s\text{-}t}$, " and min risk value for that path is ", $e^{-\bar{r}}$);

/* Probability of Survival is $e^{-\bar{r}}$ */

}

4.3 Improvements in Efficiency

We add a preprocessing step **P** (Carlyle et al., 2008) immediately following the heuristic **H**. Preprocessing removes edges from the graph that cannot form part of an optimal solution. **P** identifies a "shortest-path" from s to t using a shortest-path algorithm in the forward direction with the same edge "lengths" as in **H**. The forward shortest-path algorithm calculates a lower bound $\underline{r}'_f(v)$ on the true risk of transiting from s to v $\forall v \in V$. If $\underline{r}'_f(v) + \underline{r}'(v)$ is greater than or equal to the initial solution's risk value, $\bar{r}$, **P** removes all incoming and outgoing edges at vertex v. **P** also removes edges

that have a risk value, $\sum_{m \in M_{uv}} r_{em}$, that is greater than or equal to $\bar{r}$. The pseudocode is as follows:

/* Begin Preprocessing **P** */

{

Identify a shortest path from s to t with a forward shortest-path algorithm using edge "lengths" $\underline{r}_{uv}$;

for(all $v \in V$){ $\underline{r}'_f(v) \leftarrow$ lower bound on risk from s to v; }

for(all $u \in V$){

if($\underline{r}'_f(u) + \underline{r}'(u) \geq \bar{r}$) {

$E_{u\text{-}v} \leftarrow \emptyset$; /* $E_{u\text{-}v}$ is the set of edges emanating from u */

} **else** {

nextEdge(u) $\leftarrow$ firstEdge(u);

while(nextEdge(u) $\neq \emptyset$) {

$(u, v) \leftarrow$ the edge pointed to by nextEdge(u);

increment nextEdge(u);

if($\sum_{m \in M_{uv}} r_{em} \geq \bar{r}$) { remove (u, v); }

}

}

}

}

We avoid recomputing the lower bound $\underline{r}'(v)$ each time M_{su} changes by removing the check $r^T(u) + \underline{r}_{uv} + \underline{r}'(v) > \bar{r}$. During computation, we find that the work spent recomputing the lower bounds is greater than the work these bounds save in enumeration. We propose two explanations for this occurrence: (1) the lower bounds are weak, and (2) the lower bounds in **P** limit the work spent in enumeration before it begins. The version of **MRPA** we use for computational testing in Section 5 never

recomputes the lower bounds. **MRPA** uses the lower bounds to extend a sub-path from v to t when $\underline{r}'(v)$ is 0, which indicates that the sub-path v-t is risk free, by concatenating the partial path s-v found during enumeration and the sub-path v-t found during the calculation of the initial lower bounds.

After **P**, we execute network reductions **R** (Carlyle et al., 2009). **R** eliminates vertices that are risk free, i.e., no mines impinge upon the inbound and outbound edges of the vertex. For example, we are at vertex u and no mines impinge upon any edges in the reverse and forward directions. Let i denote a tail vertex of an edge in the reverse direction, and j denote a head vertex of an edge in the forward direction. **R** eliminates edges (i, u) $\forall i$ and (u, j) $\forall j$ and constructs edges (i, j) $\forall i, j$ while ensuring that (i, j) does not already exist in the set of edges E.

CHAPTER 5

COMPUTATIONAL RESULTS

This chapter presents computational results from **MRPA** and the formulations SP, TMR1, and TMR2. A shortest path algorithm is able to solve SP and is considered the default solution method. TMR1 and TMR2 are solved with an integer program. We test our formulations using AMPL 12 as the modeling language and CPLEX 12.1 (IBM, 2010) as the solver on a Dell PowerEdge R410 machine with 12GB of RAM and a 2.26GHz processor. We use the default parameters in CPLEX. We code the **MRPA** using C++ for the language and Xcode 3.1.4 for the Integrated Development Environment (IDE). We test the **MRPA** using a MacBook Pro with a 2.26GHz processor and 2GB of RAM.

AMPL scripts generate vertices and edges for a test network and mine locations. Node spacing is 100 yards. The length (x-direction) and width (y-direction) of the network is 3000 yards by 3000 yards. Edges exist between immediate neighboring vertices in the forward and diagonal directions. Sets of edges, those that connect a source vertex to a sink vertex, represent possible paths for ship transit. AMPL scripts uniformly distribute the x and y coordinates of mines in the minefield. This network topology and mine location distribution are representative of that found in Bekker and Schmid (2006) and Li (2009). The number of edges and vertices in the test network is 2792 and 963, respectively. Threat locations and damage radii in each test problem vary. Figure 5.1 shows a minefield constructed by the above procedure.

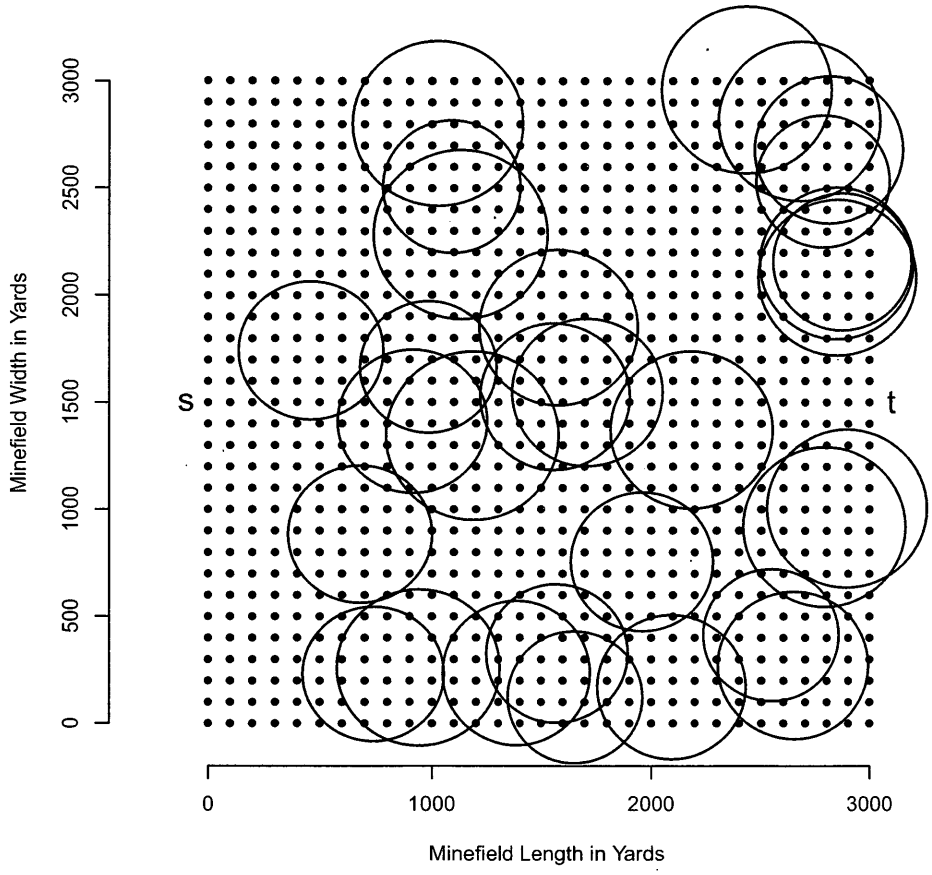

Figure 5.1: A minefield representative of those we construct for testing. Circles denote threat circles. At the center of each circle is a mine. Dots denote vertices. Edges exist between immediate neighboring vertices in the forward and diagonal directions and are omitted from this figure for clarity.

We construct 35 test problems. The results from instances 1-5 are included in this chapter for explanatory purposes. Appendix B presents results from instances 6-35, which increases the sample size of our results through testing on 30 minefields with the same number of mines and minimum and maximum damage radii. The number of mines in each problem is given in Table 5.1. Each mine has a uniformly distributed damage radius R_m, which depends upon the instance. The minimum and maximum damage radii for each instance are given in Table 5.1. By using mines with various

damage radii, we can "seed" the minefield with different types of mines, extending the previous assumption in the literature that all mines are identical. The damage radii given below are similar to the radii given in Bekker and Schmid (2006) and Li (2009).

Table 5.1: Problem Instance Mine Specifications. The minimum and maximum damage radii ("Range") are the parameters we use in the uniform distribution that generates the damage radius R_m of each mine m in a problem instance. Results for Problems 6-35 appear in Appendix B.

Problem Instance	Number of Mines	Minimum Range (yd)	Maximum Range (yd)
1	30	300	400
2	30	300	400
3	30	300	400
4	35	300	425
5	30	300	450
6-35	32	325	425

5.1 Solution Quality

We compare the optimal probability of survival for the paths identified by the threat-additive model, solved with integer programming and **MRPA**, with the probability of survival for the path identified by the edge-additive model, solved with a shortest-path algorithm. "Optimal" only refers to the solutions obtained from solving a threat-additive model. Table 5.2 reports the probability of survival values. The optimal probability of survival in Problems 3, 4, and 5 is small, i.e., less than 0.7. A small probability of survival is undesirable to most decision makers and ship drivers. Yet, a situation may arise in which a harbor is mined, and a ship must transit the minefield quickly or encounter certain destruction by other threats, such as missiles. In this situation, the ship is better off attempting to transit the minimum-risk route that is provided rather than arbitrarily navigating the minefield. For this reason, we

conduct testing over instances in which the probability of survival spans the complete probability range of zero to one, inclusive.

Table 5.2: Optimal probability of survival computed through **MRPA** and through the integer programs TMR1 and TMR2. "SP True" is the probability of survival of the SP solution evaluated under the threat-additive model. "SP Gap (%)" is the percentage difference between "Optimal $P(Survival)$" and "SP True." Results for Problems 6-35 appear in Table B.2.

Problem Instance	Optimal $P(Survival)$	SP True	SP Gap (%)
1	0.777	0.777	0.00%
2	0.864	0.864	0.00%
3	0.684	0.618	9.65%
4	0.615	0.597	2.93%
5	0.584	0.584	0.00%

Although an edge-additive model over-accumulates risk, the path identified by the shortest-path algorithm is often optimal. The optimal routes from solving an edge-additive and a threat-additive model correspond to nearly-identical solutions in 21 of 35 test problems. We define nearly-identical solutions as those in which the objective values, evaluated under a threat-additive model, are equal; but, the sets of edges comprising the routes may differ. (Note: When the shortest-path is optimal but differs from the optimal paths identified by other means, the optimization problem has multiple optimal solutions.)

The solutions to threat-additive and edge-additive models appear to be identical when the set of edges that accumulates risk in an edge-additive model and the edges corresponding to the maximum probability of actuation in a threat-additive model intersect for each threat on an optimal route. We observe identical solutions in Problems 1, 2, and 5. Threat-additive and edge-additive solutions appear to diverge when the optimization procedure must choose between penetrating more deeply inside a threat circle and penetrating more threat circles. In Problems 3 and 4, this situation

arises when a ship must transit the intersection of three threat circles. Graphical depictions of these two cases are given in Figure 5.6 and Figure 5.7.

A numerical example below explains how solutions to edge-additive and threat-additive models diverge. A ship must transit the intersection of three threat circles on one of two paths, $\mathcal{A}$ or $\mathcal{B}$. Each of the three edges in Path $\mathcal{A}$ exists in a distinct threat circle; each edge possesses a corresponding probability of survival of 0.9. Two mines impinge upon Path $\mathcal{B}$, which consists of an edge, e_1, in one threat circle and two edges, e_2 and e_3, in another threat circle. The probability of survival on edges e_1, e_2, and e_3 is 0.9, 0.9, and 0.85, respectively. The optimization procedure of an edge-additive model produces a solution corresponding to transiting Path $\mathcal{A}$ with a probability of survival of $0.9^3 = 0.729$. The corresponding threat-additive solution results from minimizing the maximum risk; in this case, a ship transits Path $\mathcal{B}$ with a probability of survival of $(0.9) \cdot (0.85) = 0.765$. Path $\mathcal{B}$ is sub-optimal in an edge-additive model because the probability of survival on this path is $(0.9) \cdot (0.9) \cdot (0.85) = 0.689$, which is over-accumualted by 0.076.

We report the upper and lower bounds on the initial s-t path that heuristic **H** identifies in Table 5.3 and Table B.1. Frequently, the initial upper bound and incumbent

Table 5.3: We report the upper and lower bounds on the initial s-t path that heuristic **H** identifies. Optimal objective function values obtained by **MRPA** are given in the third column with the corresponding probability of survival given in the second column. The optimality gap is given in the fifth column. Results for problems 6-35 appear in Table B.1.

Problem Instance	Optimal *P(Survival)*	Optimal Objective	Lower Bound	Upper Bound	Optimality Gap (%)
1	0.777	0.252	0.114	0.252	54.8%
2	0.864	0.146	0.048	0.146	67.1%
3	0.684	0.380	0.178	0.481	63.0%
4	0.615	0.486	0.209	0.516	59.5%
5	0.584	0.538	0.245	0.538	54.5%

solution are, respectively, the optimal objective function value and solution to the threat-additive model. One explanation of this occurrence is that SP often identifies a path, in terms of true risk, that is equivalent to a solution from the threat-additive model. **H** identifies the optimal solution in 21 of the 35 test problems, as shown by Table 5.3 and Table B.1 when the values in the third and fifth columns equal each other. In the other 14 problems, **MRPA** identifies the optimal solution in all but two instances after 600 seconds of solution time. We define the optimality gap as the absolute percent deviation between the initial solution's objective function value and the lower bound. This gap serves as an upper bound on the potential improvement that the objective function value may realize during the remainder of the algorithm.

5.2 Run Time

The threat-additive model produces a minimum-risk route (if it can be solved optimally), but at a substantial computational cost compared to the shortest-path heuristic. The default solution method is a shortest path algorithm, which has a negligible run time, implying that identifying solutions to edge-additive models requires little work. We find that the increased computational cost of identifying solutions to threat-additive models is minimal; **MRPA** typically solves problems in fewer than one second and solves them faster than CPLEX in 80% of the problems considered.

Table 5.4 and Table B.3 compare run times of the different solution methods for Problems 1-5 and 6-35, respectively. Run times from the integer programs depend upon the specific formulation. Using CPLEX with default parameters, TMR1 solves problems faster than TMR2 86% of the time, although the LP relaxation of TMR2 is stronger than the relaxation of TMR1 every time (see Table B.4 for LP relaxation values). SP performs better than any other method because shortest path models can be solved in polynomial time.

MRPA's run time varies depending upon the number of edges and vertices that

Table 5.4: The solution time ("Run Time") of solution methods **MRPA** and integer programs TMR1 and TMR2 on Problems 1-5. No corresponding SP solution time exceeds 0.01 seconds. Results for problems 6-35 appear in Table B.3.

Problem Instance	Run Time (sec)		
	TMR1	TMR2	**MRPA**
1	1.27	2.93	0.12
2	.86	2.02	<.01
3	1.48	9.42	0.09
4	2.55	14.56	0.50
5	2.96	45.14	2.28

P and **R** eliminate and the amount of enumeration that the upper bound limits. Table B.3 displays the number of edges and vertices remaining in Problems 6-35. In two of the instances, **MRPA** fails to prove optimality after a reasonable length of run time, i.e., 600 seconds, but does provide solutions within 0.3% of optimality. The number of edges and vertices remaining after preprocessing and reductions in these two instances is larger than the number in most other instances.

Thus far, the integer programs and the **MRPA** determine solutions using a 0% optimality tolerance. We specify a 5% optimality tolerance to analyze its impact on the objective value and the solution time. Specifying a 5% optimality tolerance does not improve the solution times of the integer programs. However, we find a statistically significant reduction, at the 0.01 level, in solution times for **MRPA**. The differences in objective function value, $P(Survival)$, and solution time between identifying a near-minimum risk path (NMRP) and the optimal path are given in Table B.5 for Problems 1-35. The average reduction in solution time, excluding the two problems that did not solve, is 0.3 seconds.

Figure 5.2 shows that a 5% optimality tolerance reduces the probability of survival by no more than 0.02. This 0.02 reduction occurs when the objective value is approximately 1, which corresponds to a probability of survival of 0.4. The reduction in the probability of survival that a 5% tolerance causes may be less than the

upper bound of 0.02. Most scenarios that the military encounters have an optimal probability of survival that is much greater than 0.4. A higher optimal probability of survival results in a smaller gap between the NMRP and the optimal path probability of survival. Figure 5.2 and Figure 5.3 show the nonlinear relationship between the objective function values and the probability of survival. This nonlinear relationship, coupled with the probability of survival that the military usually encounters, may permit the use of a much larger optimality tolerance in near-minimum risk routing than the tolerances used in other applications.

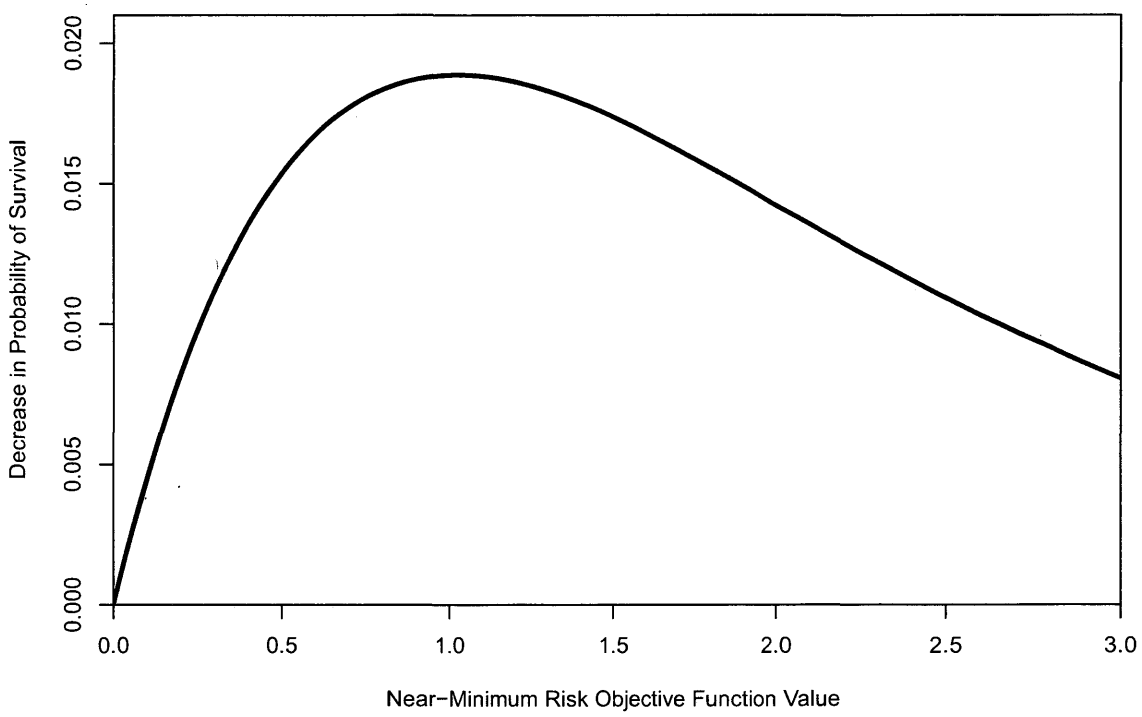

Figure 5.2: The x-axis is the objective function value. The y-axis shows the gap between the optimal $P(Survival)$ and the near-minimum risk path, NMRP, $P(Survival)$. The NMRP objective function value is (in the worst-case) 5% more than the optimal value. However, the gap between the NMRP $P(Survival)$ and the optimal $P(Survival)$ is less than 5% because the function relating the objective function value to the $P(Survival)$ is nonlinear.

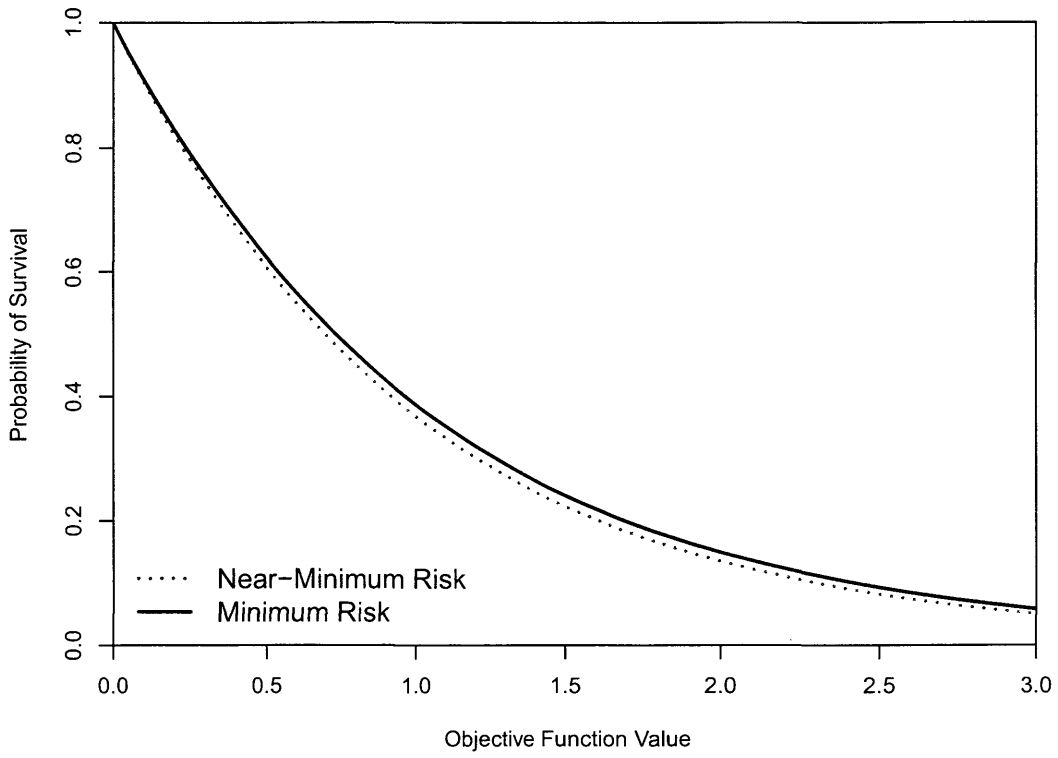

Figure 5.3: The x-axis is the objective function value, and the y-axis is the probability of survival. The solid line shows the $P(Survival)$ for optimal objective function values. The dotted line shows the $P(Survival)$ for the objective function values that are (in the worst-case) 5% more than the optimal value.

5.3 Graphical Depiction of Solutions

The minimum-risk routes from solution methods TMR1 and **MRPA**, in addition to the route from SP, are shown in Figure 5.4 - Figure 5.8 for Problems 1-5. Threat circles for mines that impinge upon the optimal route are shown in bold black; all others are in grey. Diamonds denote mine locations. The deviations in optimal routes between TMR1 and **MRPA** are due to multiple optimal solutions.

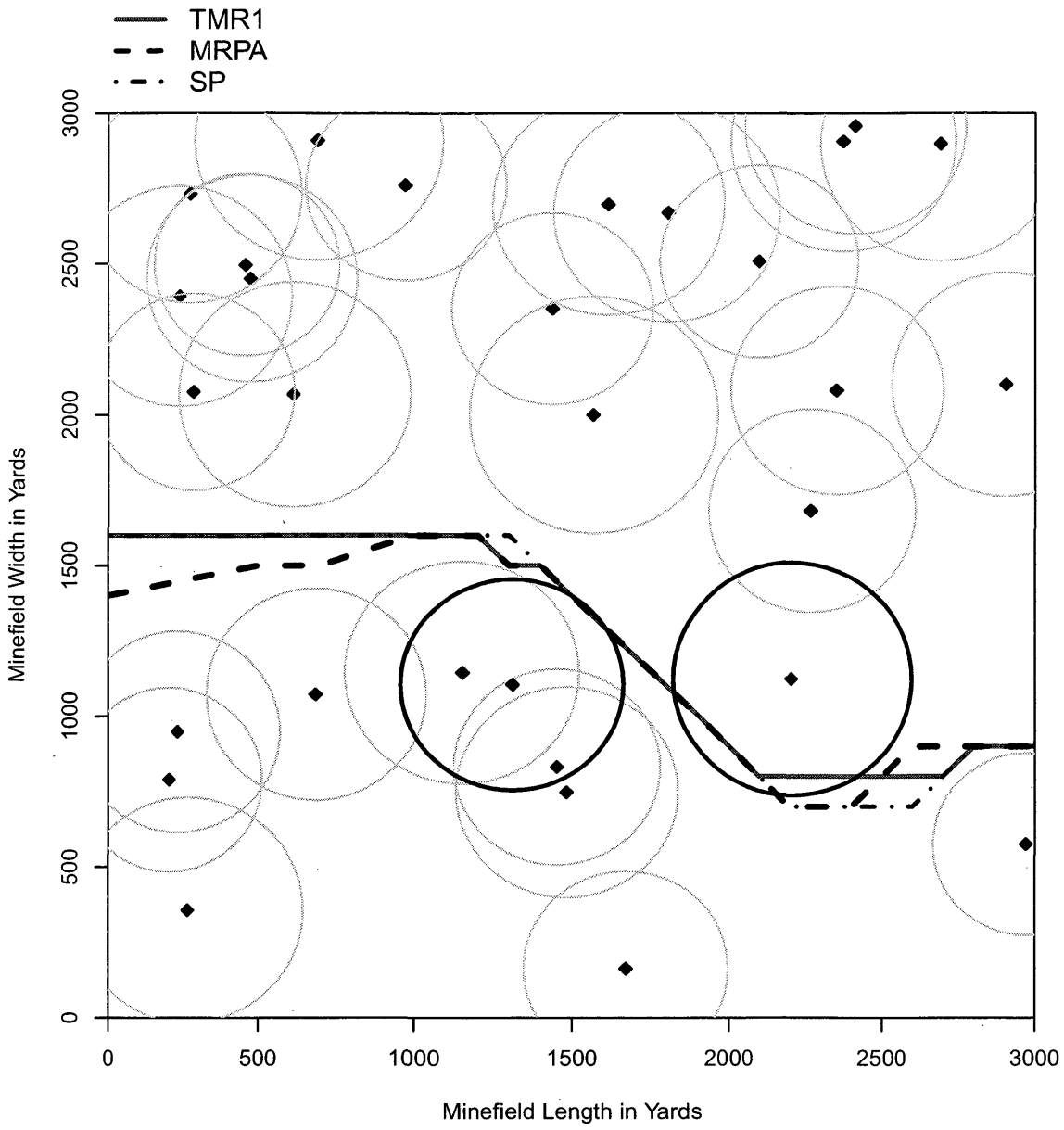

Figure 5.4: Problem Instance 1 - A ship travels from left to right. SP's route and the minimum-risk routes for TMR1 and **MRPA** are shown. Threat circles for mines that impinge upon the routes are bold black; all others are in grey. Black diamonds denote mine locations.

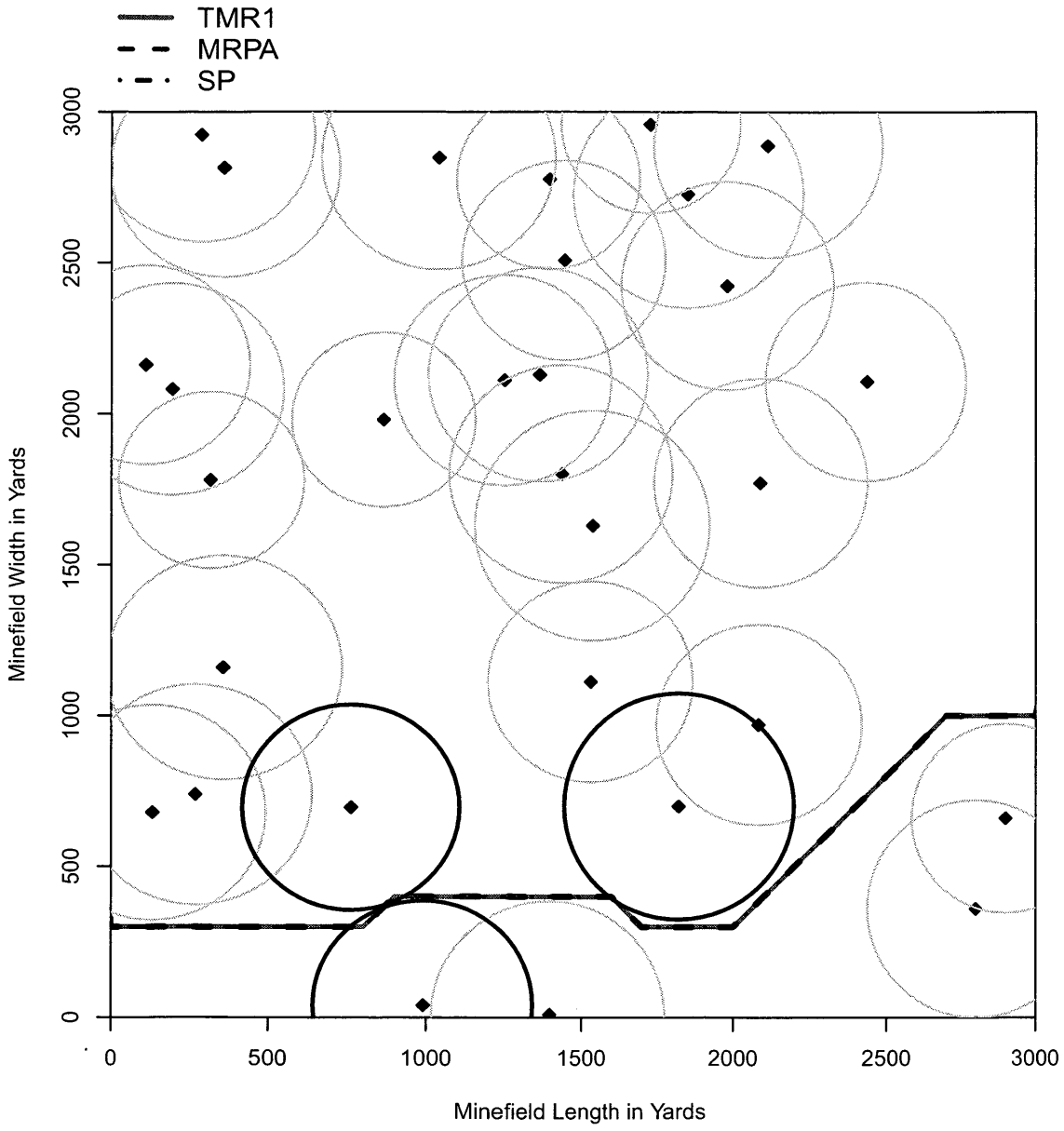

Figure 5.5: Problem Instance 2 - A ship travels from left to right. SP's route and the minimum-risk routes for TMR1 and **MRPA** are shown. Threat circles for mines that impinge upon the routes are bold black; all others are in grey. Black diamonds denote mine locations. All optimal routes are the same.

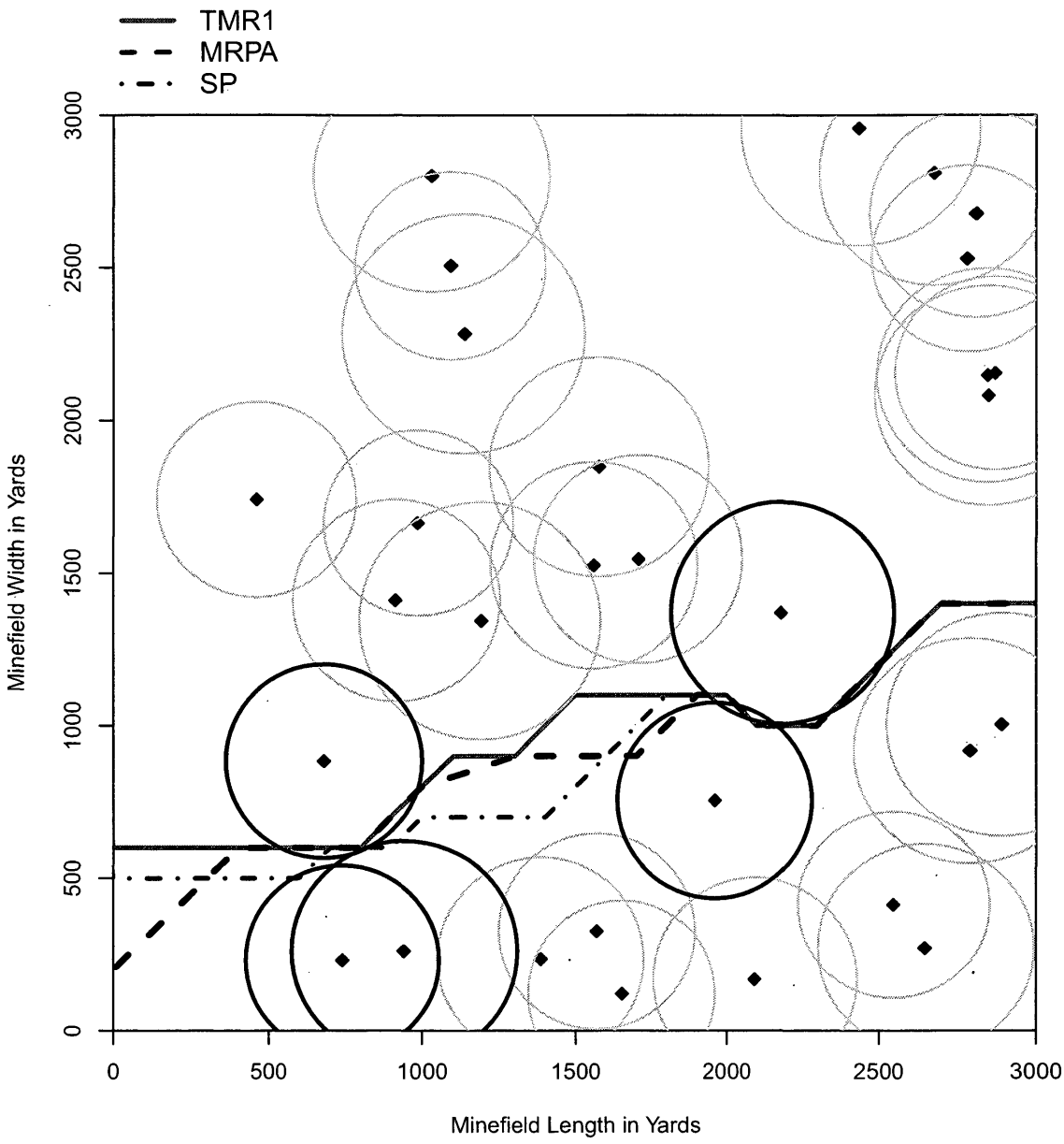

Figure 5.6: Problem Instance 3 - A ship travels from left to right. SP's route and the minimum-risk routes for TMR1 and **MRPA** are shown. Threat circles for mines that impinge upon the routes are bold black; all others are in grey. Black diamonds denote mine locations.

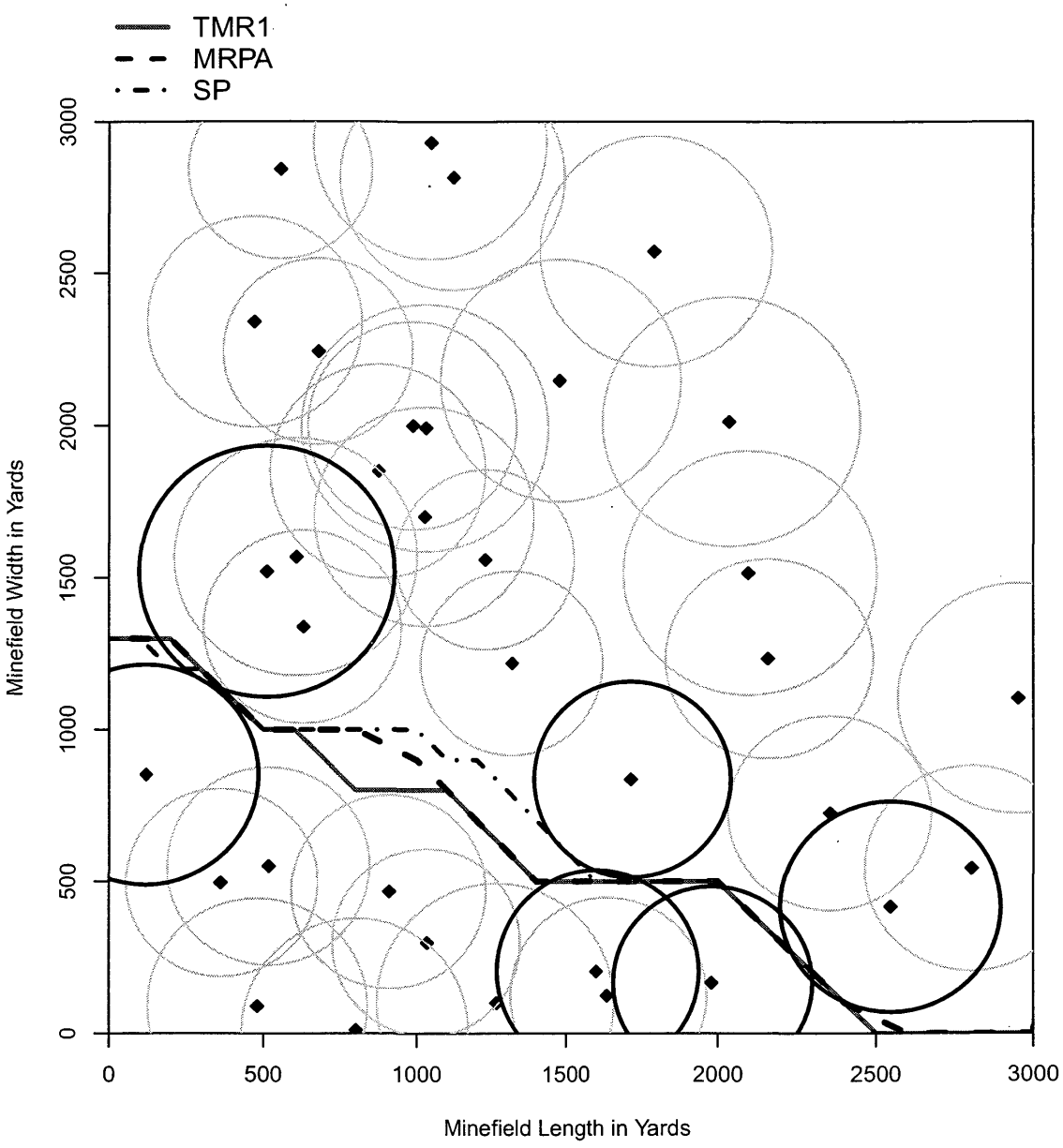

Figure 5.7: Problem Instance 4 - A ship travels from left to right. SP's route and the minimum-risk routes for TMR1 and **MRPA** are shown. Threat circles for mines that impinge upon the routes are bold black; all others are in grey. Black diamonds denote mine locations.

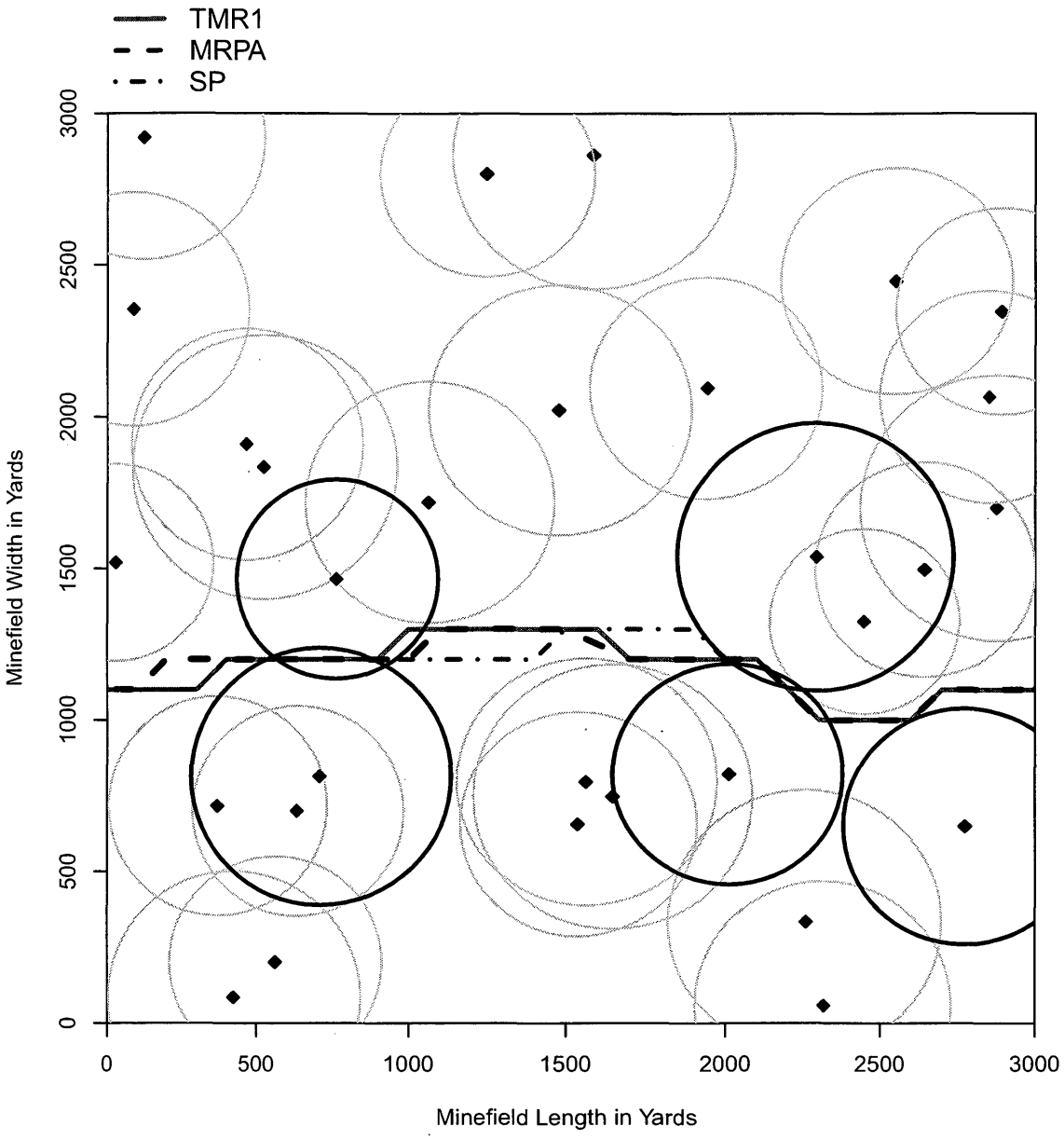

Figure 5.8: Problem Instance 5 - A ship travels from left to right. SP's route and the minimum-risk routes for TMR1 and **MRPA** are shown. Threat circles for mines that impinge upon the routes are bold black; all others are in grey. Black diamonds denote mine locations.

CHAPTER 6

SUMMARY AND CONCLUSIONS

This thesis has developed a model, and two accompanying solution methods, to identify a minimum-risk route for a ship traversing a mapped minefield. Standard shortest-path based methods employ an edge-additive model that may over-accumulate the total risk associated with a path and yield poor solutions. Our threat-additive model avoids these problems. The solution methods include (a) two integer programs (IPs) implemented in CPLEX, and (b) a novel path-enumeration algorithm, **MRPA** (Minimum-Risk Path Algorithm). **MRPA** may be useful for practical applications in which near-real-time use and independence of licensed optimization software are necessary.

MRPA solves problems faster than CPLEX in 80% of our test problems. For the instances in which **MRPA** fails to prove or reach optimality after 600 seconds, the probability of survival on the incumbent path is small, less than 0.5. For problem instances that CPLEX solves faster than **MRPA**, **MRPA** solves them in 35 seconds or fewer, with one exception of 131 seconds of solution time. If near-real-time performance is required of the algorithm, an implementer may wish to impose a time limit and use the incumbent solution if run time exceeds this limit.

Future work can concentrate on improving the efficiency of **MRPA** and testing it on minefields representing different general shapes. For instance, a long and narrow area of operations might realistically represent a harbor entrance, and it would be interesting to understand how such a shape affects solution times. The strength of the lower bounds influences **MRPA**'s solution time. We use an upper bound, n_m, on the maximum number of times that mine m can impinge upon a simple s-t path in the calculation of the lower bounds to ensure that they are valid. Strengthening these lower bounds by deriving a tighter bound for n_m may decrease solution time.

We focus on routing a ship in a mapped minefield, but **MRPA** can be applied to other applications such as routing military aircraft or minefield optimal clearance.

REFERENCES CITED

Bekker, J. and Schmid, J., 2006. "Planning the Safe Transit of a Ship through a Mapped Minefield." *ORION*, **22**(1): 1–18.

Boerman, D., 1994. *Finding an Optimal Path Through a Mapped Minefield.* Master's thesis, Department of Operations Research, Naval Postgraduate School, Monterey, California.

Button, R., Kamp, J., Curtin, T. and Dryden, J., 2009. "A Survey of Missions for Unmanned Undersea Vehicles." Tech. Rep. MG808, RAND Corporation.

Byers, T. and Waterman, 1984. "Determining all Optimal and Near-Optimal Solutions when Solving Shortest Path Problems by Dynamic Programming." *Operations Research*, **32**(6): 1381–1384.

Carlyle, M., Royset, J. and Wood, R., 2009. "Routing Military Aircraft With a Constrained Shortest-Path Algorithm." *Military Operations Research*, **14**(3): 31–52.

Carlyle, M. and Wood, R., 2005. "Near-Shortest and K-Shortest Simple Paths." *Networks*, **46**(2): 98–109.

Carlyle, W., Royset, J. and Wood, R., 2008. "Lagrangian Relaxation and Enumeration for Solving Constrained Shortest-Path Problems." *Networks*, **52**(4): 256–270.

CNO, 1991. *The United States Navy in Desert Shield/Desert Storm.* Naval Historical Center, Washington DC, Appendix A: Chronology ed., http://www.history.navy.mil/wars/dstorm/index.html.

Cornish, G., 2003. *U.S. Naval Mine Warfare Strategy: Analysis of the Way Ahead.* U.S. Army War College, Carlisle, Pennsylvania.

Eagle, J., 2008. *Lateral Range Curves/Sweepwidth.* Naval Postgraduate School, Monterey, California, 4 ed., Lecture Notes.

IBM, 2010. *IBM ILOG AMPL Version 12.1 Users guide: Standard (Command-line) Version Including CPLEX Directives.* Incline Village, Nevada.

Inanc, T., Muezzinoglu, M., Misovec, K. and Murray, R., 2008. "Framework for Low-Observable Trajectory Generation in Presence of Multiple Radars." *Journal of Guidance, Control, and Dynamics*, **31**(6): 1740–1749.

Karelahti, J., Virtanen, K. and Ostrom, J., 2008. "Automated Generation of Realistic Near-Optimal Aircraft Trajectories." *Journal of Guidance, Control, and Dynamics*, **31**(3): 674–688.

Kim, J. and Hespanha, J., 2003. "Discrete Approximations to Continuous Shortest Path: Application to Minimum-Risk Path Planning for Groups of UAVs." In *42nd IEEE Conference on Decision and Control*, Maui, Hawaii, 1734–1740.

Li, P.-C., 2009. *Planning the Optimal Transit For A Ship Through A Mapped Minefield.* Master's thesis, Department of Operations Research, Naval Postgraduate School, Monterey, California.

Mercer, G. N. and Sidhu, H. S., 2007. "Two Continuous Methods for Determining a Minimal Risk Path through a Minefield." In *Proceedings of the 13th Biennial Computational Techniques and Applications Conference*, W. Read and A. J. Roberts, eds., James Cook University Townsville, North Queensland, Australia, vol. 48, C293–C306.

Morris, D., 1997. *The Mine Warfare Cycle: History, Indications, and Future.* Master's thesis, Command and Staff College, Marine Corps University, Quantico, Virginia.

Muhandiramge, R., Boland, N. and Wang, S., 2009. "Convergent Network Approximations for the Continuous Euclidean Length Constrained Minimum Cost Path Problem." *SIAM Journal*, **20**(1): 55–77.

Muljowidodo, K., Sapto Adi, N., Budiyono, A. and Prayogo, N., 2009. "Design of SHRIMP ROV for Surveillance and Mine Sweeper." *Indian Journal of Marine Sciences*, **38**(3): 332–337.

Murphey, B., Uryasev, S. and Zabarankin, M., 2003. "Trajectory Optimization in a Threat Environment." Research Report 9, Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida.

Nehme, M., 2009. *Two-Person Games for Stochastic Network Interdiction: Models, Methods, and Complexities.* Ph.D. thesis, University of Texas at Austin, Austin, Texas.

Polmar, N., Morison, S. L., Burgess, R. R. and Olver, J., 2005. *The Naval Institute Guide to the Ships and Aircraft of the U.S. Fleet.* Naval Institute Press, Annapolis, Maryland, 18 ed.

Ruz, J., Arevalo, O., De La Cruz, J. and Pajares, G., 2006. "Using MILP for UAVs Trajectory Optimization under Radar Detection Risk." In *11th IEEE International Conference on Emerging Technologies and Factory Automation*, Prague, Czech Republic, 957–960.

Shorack, G., 1964. "Algorithms and Analog Computers for the Most Reliable Route Through a Network." *Operations Research*, **12**(4): 632–633.

Zabarankin, M., Uryasev, S. and Murphey, R., 2006. "Aircraft Routing under the Risk of Detection." *Naval Research Logistics*, **53**(8): 728–747.

Zabarankin, M., Uryasev, S. and Pardalos, P., 2002. "Optimal Risk Path Algorithms." In *Cooperative Control and Optimization*, P. Pardalos, D. Hearn and R. Murphey, eds., Springer US, vol. 66 of *Applied Optimization*, 273–298.

APPENDIX A - PSEUDOCODE FOR THE BYERS AND WATERMAN NEAR-SHORTEST-PATH ENUMERATION ALGORITHM

This pseudocode is from Carlyle and Wood (2005), which modifies the near-shortest-path enumeration algorithm of Byers and Waterman (1984), to identify only simple paths. This version of the algorithm must recompute the lower bounds $d'(v)$ each time an edge is added or removed from the stack. (Note: The $d'(v)$ and $\bar{d}$ shown here are unrelated to the CPA distance d_{em} given in Section 3.3.2.) We modify the pseudocode in Steps (i), (ii), and (iii).

DESCRIPTION:

An algorithm to solve near-shortest simple paths, NSSP. The format and terminology of this algorithm are from Carlyle and Wood (2005).

INPUT:

A directed graph $G = (V, E)$ in adjacency-list format, s, t, $\mathbf{c} \geq \mathbf{0}$, and $\delta \geq 0$. "firstEdge(v)" points to the first edge in a linked list of edges directed out of v.

OUTPUT:

All simple s-t paths, whose lengths are within a factor of $1 + \delta$ of being shortest.

{

for(all $v \in V$){ $d'(v) \leftarrow$ shortest-path distance from v to t; }

/* The above requires the solution of a single shortest-path problem */

$\bar{d} \leftarrow (1 + \delta)d'(s)$;

for(all $v \in V$) { nextEdge(v) $\leftarrow$ firstEdge(v); }

theStack $\leftarrow s$; $L(s) \leftarrow 0$;

isOnPath(s) $\leftarrow$ TRUE; **for**(all $v \in V - s$){ isOnPath(v) $\leftarrow$ FALSE; }

/* "isOnPath()" is not used in the Byers and Waterman (1984) algorithm */

while(theStack is not empty){

$u \leftarrow$ vertex at the top of theStack;

```
    if( nextEdge(u) ≠ ∅ ) {
        (u, v) ← the edge pointed to by nextEdge(u);
        increment nextEdge(u);
        if( isOnPath(v) = FALSE and
        L(u) + c(u, v) + d'(v) ≤ d̄ ) { /* Step (i) */
            if( v = t ){
                print( theStack ∪ t );
            } else {
                push v on theStack;
                isOnPath(v) ← TRUE;
                L(v) ← L(u) + c(u, v);
                for( all v ∈ V ){ d'(v) ← shortest-path distance from v to t where
                    no vertices v' with isOnPath(v') = TRUE may be traversed;
                    /* Step (ii) */
                }
            }
        }
    } else {
        Pop u from theStack;
        isOnPath(u) ← FALSE;
        nextEdge(u) ← firstEdge(u);
        for( all v ∈ V ){ d'(v) ← shortest-path distance from v to t where
            no vertices v' with isOnPath(v') = TRUE may be traversed;
            /* Step (iii) */
        }
    }
}
```

}

In **MRPA**, edge "lengths" represent risk instead of distance. We modify Step (i) by replacing $L(u) + c(u,v) + d'(v) \leq \bar{d}$ with $r^T(u) + \underline{r}_{uv} + \underline{r}'(v) < \bar{r}$. If the true risk on a sub-path from s to u, $r^T(u)$, plus the lower bound on true risk from u to t, $\underline{r}_{uv} + \underline{r}'(v)$, is less than the upper bound $\bar{r}$, then edge (u,v) is added to the stack. We modify Step (ii) and Step (iii) by replacing the lower bound updates that Carlyle and Wood (2005) perform with the following procedure: **MRPA** performs a backward shortest-path algorithm in which the only mines in the instance are $M \backslash M_{su}$ and all vertices on the current sub-path are forbidden, where M is the set of all mines and M_{su} is the set of mines that impinge upon the sub-path s-u.

APPENDIX B - ADDITIONAL NUMERICAL RESULTS

We construct Problems 6-35 to test the **MRPA** and the integer programming formulations on 30 different minefields composed of the same number of mines and minimum and maximum damage radii. Results from problems 1-5 are given in Chapter 5, except for their LP relaxation objective values. The LP relaxation objective values from integer programming formulations TMR1 and TMR2 for Problem 1-35 are given in Table B.4. CPLEX solves integer programs TMR1 and TMR2 on a Dell PowerEdge R410 machine with 12GB of RAM and a 2.26GHz processor. We test **MRPA** on a MacBook Pro with a 2.26GHz processor and 2GB of RAM.

† indicates that **MRPA** did not solve the problem in 600 seconds, but it does provide an incumbent solution with bounds on the optimal solution.

Table B.1: We report the upper and lower bounds on the initial s-t path that heuristic **H** identifies for problems 6-35. Optimal objective function values obtained by **MRPA** are given in the third column. The optimality gap is given in the fifth column. † indicates that **MRPA** did not solve the problem in 600 seconds.

Problem Instance	Optimal $P(Survival)$	Optimal Objective	Lower Bound	Upper Bound	Optimality Gap (%)
6	0.503	0.687	0.334	0.687	51.4%
7	0.836	0.179	0.070	0.238	70.6%
8	0.473	0.749	0.333	0.749	55.5%
9	0.806	0.216	0.083	0.216	61.6%
10	0.528	0.639	0.475	0.639	25.7%
11	0.892	0.114	0.046	0.114	59.6%
12	0.711	0.341	0.136	0.341	60.1%
13	0.525	0.644	0.374	0.675	44.6%
14	0.544	0.609	0.357	0.609	41.4%
15	0.534	0.627	0.378	0.627	39.7%
16	0.867	0.143	0.065	0.143	54.5%
17	0.446	0.807	0.460	0.844	45.5%
18	0.385	0.955	0.534	1.044	48.9%
19	0.914	0.090	0.040	0.090	55.6%
20	0.749	0.289	0.120	0.289	58.5%
21	†	†	0.566	0.901	37.2%
22	0.472	0.751	0.378	0.909	58.4%
23	0.917	0.087	0.028	0.087	67.8%
24	†	†	0.319	0.732	56.4%
25	0.842	0.172	0.089	0.195	54.4%
26	0.781	0.247	0.098	0.247	60.3%
27	0.627	0.467	0.274	0.511	46.4%
28	0.757	0.278	0.097	0.293	66.9%
29	0.461	0.774	0.350	0.832	57.9%
30	0.903	0.102	0.031	0.102	69.6%
31	0.917	0.087	0.031	0.087	64.4%
32	0.852	0.160	0.054	0.160	66.3%
33	0.735	0.308	0.167	0.308	45.8%
34	0.509	0.675	0.322	0.764	57.9%
35	0.630	0.462	0.247	0.485	49.1%

Table B.2: Optimal probability of survival from problems 6-35 computed through **MRPA** and through the integer programs TMR1 and TMR2. "SP True" is the probability of survival of the SP solution evaluated under the threat-additive model. "SP Gap (%)" is the percentage difference between "Optimal $P(Survival)$" and "SP True." The number of mines that impinge upon the optimal route found by **MRPA** is given in the last column.

Problem Instance	Optimal $P(Survival)$	SP True	SP Gap (%)	Number of Mines
6	0.503	0.503	0.0%	3
7	0.836	0.789	5.6%	2
8	0.473	0.473	0.0%	4
9	0.806	0.806	0.0%	2
10	0.528	0.528	0.0%	1
11	0.892	0.892	0.0%	1
12	0.711	0.711	0.0%	4
13	0.525	0.510	2.9%	4
14	0.544	0.544	0.0%	3
15	0.534	0.534	0.0%	3
16	0.867	0.867	0.0%	2
17	0.446	0.430	3.6%	2
18	0.385	0.352	8.6%	4
19	0.914	0.914	0.0%	1
20	0.749	0.749	0.0%	2
21	0.406	0.372	8.4%	2
22	0.472	0.472	0.0%	4
23	0.917	0.917	0.0%	2
24	0.482	0.481	0.2%	9
25	0.842	0.824	2.1%	1
26	0.781	0.781	0.0%	2
27	0.627	0.601	4.1%	3
28	0.757	0.747	1.3%	3
29	0.461	0.435	5.6%	8
30	0.903	0.903	0.0%	3
31	0.917	0.917	0.0%	1
32	0.852	0.852	0.0%	2
33	0.735	0.735	0.0%	2
34	0.509	0.466	8.4%	3
35	0.630	0.616	2.2%	1

Table B.3: Solution times ("Run Time"), network reductions, and number of mines for problems 6-35. The number of edges and vertices present after preprocessing **P** and network reductions **R** is given in Columns 5 and 6, respectively. The original model has 963 vertices and 2792 edges. The number of mines that impinge upon the optimal route found by **MRPA** is given in the last column. † indicates that **MRPA** did not solve the problem in 600 seconds. No corresponding SP solution time exceeds 0.01 seconds.

Problem	Run Times (sec)			**P** and **R**		Number of
Instance	TMR1	TMR2	**MRPA**	Edges	Vertices	Mines
6	1.41	39.61	<.01	722	195	3
7	0.67	5.54	1.33	375	118	2
8	2.88	156.57	0.05	706	226	4
9	0.67	1.98	<.01	488	111	2
10	0.63	80.02	<.01	244	76	1
11	0.42	1.00	<.01	186	59	1
12	0.86	2.60	0.01	294	119	4
13	5.65	95.66	127.39	647	266	4
14	1.27	65.11	0.01	493	163	3
15	0.96	61.00	0.18	613	207	3
16	0.51	0.64	0.05	257	79	2
17	1.80	119.16	0.08	492	189	2
18	13.19	167.59	0.06	1053	402	4
19	0.49	0.38	<.01	79	25	1
20	0.99	0.92	<.01	288	97	2
21	13.58	177.97	†	1119	366	2
22	8.06	95.96	33.15	1364	462	4
23	0.60	0.37	<.01	78	32	2
24	7.53	70.59	†	882	295	9
25	1.14	2.46	1.76	375	105	1
26	0.71	2.73	0.01	309	104	2
27	1.15	22.38	<.01	451	158	3
28	0.63	1.46	<.01	512	108	3
29	19.39	154.37	0.34	946	369	8
30	0.61	0.38	<.01	155	50	3
31	0.28	0.34	0.01	167	66	1
32	0.96	0.56	0.01	285	75	2
33	0.81	4.19	0.25	496	151	2
34	2.90	12.40	<.01	383	167	3
35	1.67	58.73	24.66	723	271	1

Table B.4: LP relaxation objective values from integer programs TMR1 and TMR2. TMR2 has a tighter LP relaxation. LP relaxation objective values for TMR1 and TMR2 are given in Columns 2 and 3, respectively. These objective values, translated into the probability of survival, are given in Columns 5 and 6, respectively. The percentage increase ("Gap %") in the LP relaxation of TMR2 for the objective and for the objective translated into a probability is in Columns 4 and 7, respectively.

Problem	LP Relaxation Objective			LP Relaxation $P(Survival)$		
Instance	TMR1	TMR2	Gap %	TMR1	TMR 2	Gap %
1	0.063	0.148	57.4%	0.939	0.862	8.2%
2	0.062	0.125	50.6%	0.940	0.883	6.1%
3	0.107	0.233	54.3%	0.899	0.792	11.9%
4	0.229	0.449	49.0%	0.795	0.638	19.7%
5	0.147	0.316	53.6%	0.864	0.729	15.6%
6	0.172	0.340	49.3%	0.842	0.712	15.4%
7	0.077	0.169	54.6%	0.926	0.844	8.8%
8	0.202	0.422	52.1%	0.817	0.656	19.7%
9	0.063	0.131	52.2%	0.939	0.877	6.6%
10	0.192	0.291	33.9%	0.825	0.748	9.4%
11	0.035	0.091	61.2%	0.965	0.913	5.4%
12	0.123	0.269	54.1%	0.884	0.764	13.5%
13	0.149	0.326	54.5%	0.862	0.721	16.3%
14	0.155	0.295	47.3%	0.856	0.745	13.0%
15	0.161	0.274	41.1%	0.851	0.760	10.6%
16	0.074	0.143	48.3%	0.929	0.867	6.7%
17	0.149	0.339	56.1%	0.862	0.712	17.3%
18	0.199	0.403	50.7%	0.820	0.669	18.4%
19	0.037	0.090	59.5%	0.964	0.914	5.2%
20	0.142	0.289	50.8%	0.868	0.749	13.6%
21	0.172	0.336	48.8%	0.842	0.715	15.1%
22	0.151	0.278	45.4%	0.859	0.758	11.9%
23	0.041	0.086	52.9%	0.960	0.917	4.5%
24	0.166	0.331	49.8%	0.847	0.718	15.2%
25	0.058	0.128	54.6%	0.943	0.879	6.8%
26	0.078	0.191	59.3%	0.925	0.826	10.7%
27	0.111	0.242	54.1%	0.895	0.785	12.2%
28	0.075	0.171	55.9%	0.927	0.843	9.1%
29	0.196	0.467	58.0%	0.822	0.627	23.7%
30	0.089	0.102	12.9%	0.915	0.903	1.3%
31	0.048	0.087	44.6%	0.953	0.917	3.8%
32	0.097	0.160	39.4%	0.908	0.852	6.1%
33	0.117	0.220	46.8%	0.889	0.802	9.8%
34	0.200	0.429	53.4%	0.819	0.651	20.5%
35	0.099	0.194	48.7%	0.905	0.824	9.0%

Table B.5: This table reports results from **MRPA** enumerating a near-minimum risk path (NMRP) that is within 5% of optimality. "Objective Gap" is the percent change between the near-minimum risk objective value and the optimal objective value. Column 3 is the percent change between the near-minimum risk $P(Survival)$ and the optimal $P(Survival)$. Column 4 ("Run Time Gap") is the number of seconds that NMRP enumeration saves. † indicates that **MRPA** did not solve the problem in 600 seconds.

Problem Instance	Objective Gap	$P(Survival)$ Gap	Run Time Gap (sec)
1	0.00%	0.00%	0.01
2	0.00%	0.00%	0.00
3	0.00%	0.00%	0.02
4	0.00%	0.00%	0.12
5	0.00%	0.00%	0.60
6	0.00%	0.00%	0.00
7	0.00%	0.00%	0.28
8	0.00%	0.00%	0.01
9	0.00%	0.00%	0.00
10	0.00%	0.00%	0.00
11	0.00%	0.00%	0.00
12	0.00%	0.00%	0.01
13	4.58%	3.05%	3.09
14	0.00%	0.00%	0.00
15	0.00%	0.00%	0.05
16	0.00%	0.00%	0.00
17	4.33%	3.59%	0.00
18	2.15%	2.08%	0.01
19	0.00%	0.00%	0.00
20	0.00%	0.00%	0.00
21	†	†	†
22	4.39%	3.39%	2.84
23	0.00%	0.00%	0.00
24	†	†	†
25	0.00%	0.00%	0.16
26	0.00%	0.00%	0.00
27	0.00%	0.00%	0.00
28	5.00%	1.45%	0.00
29	0.00%	0.00%	0.10
30	3.25%	0.33%	0.00
31	0.00%	0.00%	0.00
32	0.00%	0.00%	0.00
33	0.00%	0.00%	0.09
34	0.00%	0.00%	0.00
35	4.64%	2.22%	1.18