การปรับปรุงวิธีแบบค่าเสียโอกาสมากน้อยที่สุดสำหรับปัญหาค่าคาดหวังของรีคอร์ส ที่มีเวกเตอร์ของความไม่แน่นอน

นางสาวธิภาดา สารปรัง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2559

IMPROVING MINIMAX REGRET APPROACH OF EXPECTED RECOURSE PROBLEM WITH VECTORS OF UNCERTAINTY

Miss Thibhadha Saraprang

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Applied Mathematics and Computational Science

Department of Mathematics and Computer Science

Faculty of Science Chulalongkorn University

Academic Year 2016

Thesis Title	IMPROVING MINIMAX REGRET APPROACH OF EXPECTED RECOURSE PROBLEM WITH VECTORS OF UNCERTAINTY
By	Miss Thibhadha Saraprang
Field of Study	Applied Mathematics and Computational Science
Thesis Advisor	Assistant Professor Phantipa Thipwiwatpotjana, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

. . . . Dean of the Faculty of Science

(Associate Professor Polkit Sangvanich, Ph.D.)

THESIS COMMITTEE

. . . . Chairman

(Assistant Professor Boonyarit Intiyot, Ph.D.)

. . . . Thesis Advisor

(Assistant Professor Phantipa Thipwiwatpotjana, Ph.D.)

. . . . Examiner

(Kitiporn Plaimas, Ph.D.)

. . . . External Examiner

(Rawee Suwandechochai, Ph.D.)

นางสาวธิภาดา สารปรัง: การปรับปรุงวิธีแบบค่าเสียโอกาสมากน้อยที่สุดสำหรับปัญหาค่าคาดหวังของรีคอร์ส ที่มีเวกเตอร์ของความไม่แน่นอน. (IMPROVING MINIMAX REGRET APPROACH OF EXPECTED RECOURSE PROBLEM WITH VECTORS OF UNCERTAINTY) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ.ดร.พันทิพา ทิพย์วิวัฒน์พจนา, 63 หน้า.

ความไม่แน่นอนหมายถึงการที่มีค่ามากกว่าหนึ่งค่าที่สามารถเป็นตัวแทนของสิ่งที่สนใจสิ่งหนึ่ง วิทยานิพนธ์ฉบับนี้ พิจารณารูปแบบพิเศษของความไม่แน่นอน ซึ่งคือช่วงความน่าจะเป็นที่สามารถแทนด้วยเซตสุ่ม แบบจำลอง $\min_{\boldsymbol{x}} \; \boldsymbol{c}^T\boldsymbol{x},\ \boldsymbol{A}^T\boldsymbol{x} \geq \boldsymbol{b},\ \boldsymbol{x} \geq 0$ มีโครงสร้างพิเศษของความไม่แน่นอนนั่นคือ สัมประสิทธิ์ความไม่แน่นอนและค่าความไม่แน่นอนทางขวามือของแต่ละข้อจำกัดจะอยู่ในรูปเวกเตอร์ของความไม่แน่นอน เราแปลงกำหนดการเชิงเส้นซึ่งมีความไม่แน่นอนแบบพิเศษนี้ เป็นปัญหาค่าคาดหวังแบบช่วงของรีคอร์ส แล้วหาค่าเสียโอกาสมากน้อยที่สุดของปัญหานั้นโดยกระบวนผ่อนคลาย กระบวนผ่อนคลายจะถูกปรับปรุงโดยใช้แนวคิดของการเรียงลำดับ และข้อเท็จจริงที่ว่าเราสามารถลดขนาดของเซตความน่าจะเป็น ที่เกิดจากทุกกรณีของการเรียงลำดับที่เป็นไปได้และลดเวลาในการคำนวณ

ภาควิชา	คณิตศาสตร์และวิทยาการคอมพิวเตอร์	ลายมือชื่อนิสิต	....
		ลายมือชื่อ อ.ที่ปรึกษาหลัก	....
สาขาวิชา	คณิตศาสตร์ประยุกต์และวิทยาการคณนา		
ปีการศึกษา	2559		

5772016023: MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

KEYWORDS: TYPESETTING / INTRODUCTION / LATEX

THIBHADHA SARAPRANG : IMPROVING MINIMAX REGRET APPROACH OF EXPECTED RECOURSE PROBLEM WITH VECTORS OF UNCERTAINTY. ADVISOR : Asst. Prof. Phantipa Thipwiwatpotjana, Ph.D., 63 pp.

Uncertainty means that more than one realizations can represent an entity of interest. This thesis concerns a special pattern of uncertainty, which is a probability interval that can be represented as a random set. Our linear optimization model $\min_{\boldsymbol{x}} \boldsymbol{c}^T\boldsymbol{x},\ \boldsymbol{A}^T\boldsymbol{x} \geq \boldsymbol{b},\ \boldsymbol{x} \geq 0$ has a special structure of the uncertainty; i.e., the coefficients and the right hand side of each constraint form vector of uncertainty. We transform a linear program with this special uncertainty into an interval expected recourse problem, then find the minimax regret of this issue by a relaxation procedure. The relaxation procedure finally has been improved by using the idea of ordering and the fact that we can reduce the size of the probability set of all possible ordering cases and reduce the calculation time.

Department	: Mathematics and Computer Science	Student's Signature	....
Field of Study	: Applied Mathematics and Computational Science	Advisor's Signature	....
Academic Year	: 2016		

Acknowledgements

I would like to acknowledge my advisor, Assistant Professor Phantipa Thipwiwatpotjana, Ph.D., not only for her suggestions to complete in my master thesis but also for broadening my academic vision. She also constantly encouraged me to overcome any obstacles I encountered during this work. I could not complete this thesis without her support.

Moreover, I would like to extend my gratitude to all dissertation committees: Assistant Professor Boonyarit Intiyot, Dr. Kitiporn Plaimas and Dr. Rawee Suwandechochai for their comments and suggestions. Furthermore, I would like to thank all teachers who have taught me all valuable knowledge.

Next, I am thankful to the Department of Mathematics and Computer Science, Chulalongkorn University which provides the most valuable moment in my study by giving me a chance to continue higher education.

Last but not least, I would like to thank my friends and colleagues in Chulalongkorn University who have never left me alone. Lastly, I am most grateful to my mother and my sister who are the most important support in my life and give me the opportunity to reach my dream.

Contents

List of Tables

List of Figures

CHAPTER I

INTRODUCTION

In a general form of a linear programming problem min $\boldsymbol{c}^T\boldsymbol{x}\ \ s.t.\ \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}$, if an optimal solution exists, we usually find an optimal solution of this problem by the simplex method. However, we focus on a linear programming problem with vectors of uncertainty in this thesis. In other words, the coefficient matrix $\boldsymbol{A}$ and right hand side $\boldsymbol{b}$ have uncertain information in a vector form. Uncertain entity is an unknown information or a situation that is difficult to explain its realization of that entity. For example, productivities of both brown and white sugar (kg/bag) may have three realizations (Below average, Average and Above average) where the probability of occurrence depends on the weather. The total demand b_i may also have realizations in the same pattern of economic behavior. Therefore, three vector realizations could be Below average (realization 1) : $= (7, 8, 45)$ with probability f_1, Average (realization 2): $= (8, 9, 50)$ with probability f_2 and Above average (realization 3): $= (9, 10, 53)$ with probability f_3, where (α, β, γ) in each realization refers to the productivities (kg/bag) of brown sugar, white sugar and the total demand (kg) of sugar, respectively. This thesis concentrates on such an uncertainty (called vectors of uncertainty) as explained in the example above. There are many types of uncertainty, such as random set, possibility distribution and probability interval. However, we concentrate only on the probability interval type of uncertainty that can be represented as a random set. The references [3], [7] and [8] showed the relationship between a probability interval and a random set and the conditions when they can be represented the same information.

When a problem is a linear program with uncertainty, it is impossible to find an optimal solution directly. A decision maker needs to have a policy on how to use the uncertainty. The methods to solve a linear program with uncertainty by using pessimistic, optimistic, or minimax regret approaches are presented in [2] and [7]. The pessimistic approach is the opposite of the optimistic one. It provides the maximum of the expected recourse values when the original linear program with uncertainty is a minimizing problem. Pessimistic and optimistic solutions provide the boundary of the actual objective value when we do not know the exact realization. The minimax regret approach provides the minimum of the maximum regret due to not knowing the actual probability to establish an expected recourse model. A minimax regret solution would minimize the maximum difference between the actual and the best possible outcome under a particular scenario when we cannot forecast the future result. Comprehensive methods for handling linear programming problems under mixed uncertainty by using pessimistic, optimistic, and minimax regret approaches are stated in [9].

The method to find a minimax regret optimal solution using the relaxation procedure is our main interest in this manuscript. We have an objective to improve the relaxation procedure in [9] by using the idea of ordering. We want to reduce the size of probability boundary (the lower probabilities) of all possible ordering cases and reduce the calculation time. Nevertheless, we limit the number of each of uncertain vectors and uncertain constraints to be only up to five constraints and three realizations. Because for the case of one or two realizations, there are no differences between the original and the improved methods. For the case of more than three realizations, it is not difficult to see that if we apply our method to only the partial of any three realizations and leave the rest normally, we still would be able to reduce the calculation time. Therefore it is not necessary to expand our investigation to more than three realizations. We consider only up to five uncertain constraints because of the similar reason.

This thesis divides into four chapters. In Chapter II, we present the preliminaries and the concept of a stochastic expected recourse model and an interval expected recourse model, which are important to transform the uncertain problem into various approaches. We present the idea of minimax regret of an uncertain expected recourse problem, the algorithm for a general relaxation procedure, the algorithm using the idea of ordering, and the idea how to improve it in Chapter III. We then show a process that we find the average time by MATLAB in Chapter IV. We finally draw the conclusion and suggest for future works in Chapter V.

CHAPTER II

PRELIMINARIES

In this chapter, we present all basic knowledge needed in order to deal with a relaxation procedure for the minimax regret approach. We start with a literature review on minimax regret approach of linear programming problems with uncertainty. We then provide some basic idea of an uncertain information such as probability interval, random set, believe, plausibility, assignment function, and provide the relationship between them. After that, we demonstrate the concept of a stochastic expected recourse model and an interval expected recourse model.

2.1 Literature Reviews

There are some literature that worked on the minimax regret approach of linear programming problems with uncertainty, which tried to show new methods to solve problems under their working conditions. In 1995, Inuiguchi [4] studied a linear programming problem with interval objective function coefficients by using minimax regret criterion. Mausser and Laguna [5] wanted to find the method to reduce the time per iteration of minimax regret for linear programs with interval objective function coefficients. So, they proposed their heuristic method in 1999. This heuristic approach concentrates on the minimax regret problem which can guarantee solving a maximum regret problem to its optimality. The result of this method is that the time per iteration reduces, but the total time increases. Averbakh [1] suggested a method for finding solutions of minimax regret for a group of combinatorial optimization problems with objective functions of minimax type and uncertain objective function coefficients in 2000. The approach based on reducing a problem with uncertainty to some problems without uncertainty. He described the method on bottleneck combinatorial optimization problems, minimax multi-facility location problems and maximum weighted delay scheduling problems with uncertainty. Thipwiwatpotjana and Lodwick [8] presented comprehensive methods for handling a linear programming problem under mixed uncertainty by using pessimistic and optimistic, and minimax regret approaches in 2013.

We know that there are many methods to solve a linear programming problem with uncertainty. However, in this work we interest in improving the minimax regret method for finding an optimal solution by a relaxation procedure. Uncertain information in our issue is probability intervals that can be represented by random sets. We use the idea of belief and plausibility measures to represent a random set, since they are equivalent to each other. We acquaint with probability intervals more than random sets because probability intervals are easier for user to understand. A probability interval and a random set are not equivalent, but they may provide the same information under some conditions, see [3]. Therefore, we can choose to present that information in a way that is more beneficial to us.

We provide the relevant mathematical definitions of uncertain information including a reachability probability interval, random set, believe, and plausibility, in the next section.

2.2 Probability Intervals, Random Set, Believe and Plausibility Measures

In this section, we define set $X = \{x_1, x_2, x_3 \ldots, x_n\}$ to be the set of all n realizations of uncertainty information and $P(X)$ be the power set of X .

Definition 2.2.1**.** Probability interval (see [8]).

Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of all n realizations of an uncertainty information and $L = \{[l_i, u_i]; i \in \{1, 2, 3, ..., n\} \mid 0 \leqslant l_i \leqslant u_i \leqslant 1\}$. We define P^L as the set of probability distributions on X with respect to L as

$$P^L = \Big\{f \ : \ P(X) \to [0,1] \mid l_i \leqslant f(\{x_i\}) \leqslant u_i, \sum_{i=1}^{n} f(\{x_i\}) = 1, \forall i = 1, 2, 3, \ldots, n\Big\},$$

where $f(\{x_i\})$ is a probability density of $\{x_i\}$.
The set L is called the set of probability intervals, or the probability interval, in short. While, the set P^L is the set of all possible probabilities associated with L.

Definition 2.2.2**.** Proper probability interval (see [3]).

A probability interval $L = \{[l_i, u_i] \subseteq [0,1], i = 1, 2, 3, \ldots, n\}$ is called a proper probability interval if P^L is nonempty.

Therefore, $\sum_{i=1}^{n} l_i \leqslant 1 \leqslant \sum_{i=1}^{n} u_i$ always holds under the proper probability interval. For each $A \subseteq X$, the reference [3] defines the smallest and the largest probability of set A as

$$\begin{aligned} l(A) &= \min_{f \in P^L} f(A), \text{ and} \\ u(A) &= \max_{f \in P^L} f(A). \end{aligned}$$

Functions l and u are not probabilities in general, since $l(A) + l(A^c)$ may be less than 1 and $u(A) + u(A^c)$ could be greater than 1, where A^c is the complement of A.

The next definition is the condition of a proper probability interval which ensures that for each i, the lower bound l_i and the upper bound u_i, can be reached by some probabilities in the set P^L.

Definition 2.2.3**.** Reachability (see [3]).

A proper probability interval $L = \{[l_i, u_i], i = 1, 2, 3, ..., n\}$ is reachable if

$$\sum_{j \neq i} l_j + u_i \leqslant 1 \text{ and } \sum_{j \neq i} u_j + l_i \geqslant 1, \ \forall i.$$

However, when L is a reachable probability interval, it was proved in [2] that

(i). $l(\{x_i\}) = l_i$ and $u(\{x_i\}) = u_i, \ \forall i \in \{1, 2, 3, \ldots, n\}$

(ii). $l(A) = \max \left\{ \sum_{x_i \in A} l_i, 1 - \sum_{x_i \in A^c} u_i \right\}, \forall A \subseteq X,$

(iii). $u(A) = \min \left\{ \sum_{x_i \in A} u_i, 1 - \sum_{x_i \in A^c} l_i \right\}, \forall A \subseteq X.$

We use $l(A)$ and $u(A)$ to represent the boundary of probabilities of a set A;
$\{f \mid l_i \leq f(\{x_i\}) \leq u_i, \sum_{i=1}^{n} f(\{x_i\}) = 1, \forall i\} = \{f \mid l(A) \leq f(A) \leq u(A), \ \forall A \in P(X)\}.$

Random set is probability over the power set of X represented by a basic probability assignment function $m : P(X) \to [0, 1]$.

Definition 2.2.4. Probability basic assignment function and random set (see [6]).

Let $P(X)$ be the power set of X. A probability basic assignment function m is a mapping $m\colon P(X) \to [0, 1]$, such that $\sum_{A\in\mathcal{F}} m(A) = 1$, generates a random set $(\mathcal{F}, m)$, where $\mathcal{F} = \{A \in P(X) : m(A) > 0\}$.

For example, an opinion poll for a governor's election, let $X = \{a, b, c, d, e\}$ be the set of all candidates. There are 10,000 individuals providing their preferences. They may not have made their final choice, since the poll takes place well before the election. Suppose that 3,500 individuals support candidates a and b, 4,500 individuals support candidates c, d and e, 500 individuals support candidates a, 500 individuals support candidates b and d, and 1,000 individuals support all candidates (have no opinion yet).This example can be presented as subsets of X shown in figure below.

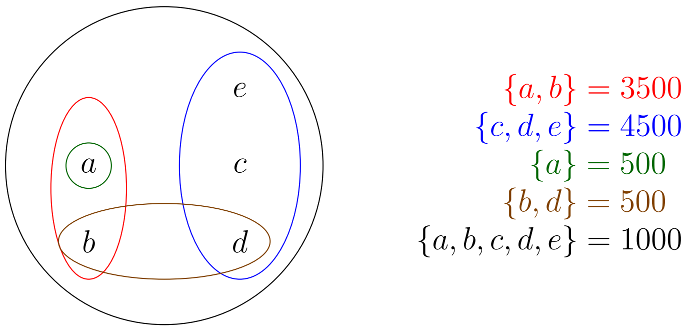

Figure 2.1: Random set example (a governor's election)

We can find the value of a basic probability assignment function m as follows:

$m(\{a, b\}) = 0.35,\qquad m(\{c, d, e\}) = 0.45,$

$m(\{a\}) = 0.05,\qquad m(\{b, d\}) = 0.05,$

$m(\{a, b, c, d, e\}) = 0.1.$

We will not get into the details of a random set but provide the definition of believe and plausibility measures which are closely related to a random set. Belief and plausibility measures can be characterized by a basic probability assignment function m as follows.

$$Bel(A) = \sum_{E, E \subseteq A} m(E),$$

and

$$Pl(A) = 1 - Bel(A^c) = \sum_{E, E \cap A \neq \emptyset} m(E).$$

It is well known as shown in [2] that

$$Bel(A) \leq f(A) \leq Pl(A),$$

where f is an unknown probability of A, given a random set of information. However, reader can read more details in [3] and [6]

Definition **2.2.5.** Belief (see [3]).

Let $P(X)$ be the power set of X. A belief measure is a function $Bel : P(X) \to [0, 1]$ that satisfies the following properties:

- $Bel(\emptyset) = 0, Bel(X) = 1$, and
- Super-additive property: for $A_1, A_2, A_3, \ldots, A_t \subseteq X$,

$$\begin{aligned} Bel(A_1 \cup \cdots \cup A_t) \quad &\geq \sum_{1 \leq j \leq t} Bel(A_j) - \sum_{1 \leq j \leq k \leq t} Bel(A_j \cap A_k) \\ &+ \cdots + (-1)^{t+1} Bel(A_1 \cap A_2 \cap A_3 \cap \cdots \cap A_t). \end{aligned}$$

Definition **2.2.6.** Plausibility (see [3]).

Let $P(X)$ be the power set of X. A plausibility measure is a function $Pl : P(X) \to [0, 1]$ that satisfies the following properties;

- $Pl(\emptyset) = 0, Pl(X) = 1$, and
- Super-additive property: for $A_1, A_2, A_3, \ldots, A_t \subseteq X$,

$$
\begin{aligned}
Pl(A_1 \cup \cdots \cup A_t) \quad &\leq \sum_{1 \leq j \leq t} Pl(A_j) - \sum_{1 \leq j \leq k \leq t} Pl(A_j \cap A_k) \\
&+ \cdots + (-1)^{t+1} Pl(A_1 \cap A_2 \cap A_3 \cap \cdots \cap A_t).
\end{aligned}
$$

The definitions above mean that belief and plausibility measures lose the additive property of probability measure. Thus, we have

$$
\begin{aligned}
Bel(A) + Bel(A^c) \quad &\leq 1, \text{ and} \\
Pl(A) + Pl(A^c) \quad &\geq 1.
\end{aligned}
$$

Boodgumarn [2] provided the meaning of belief and plausibility in a general context as follows. $Bel(A)$ means a user's beliefs that one of the elements in A could happen for sure with the proportion $Bel(A)$. $Pl(A)$ means it possible that one of the elements in A could happen with the proportion $Pl(A)$.

A given probability interval can be converted into a random set that has the same information when it is

1. a reachable probability interval,
2. having at most two indices, say i_1 and i_2, such that $\sum_{j \neq i_1} l_j + u_{i_1} < 1$ and $\sum_{j \neq i_2} l_j + u_{i_2} < 1$,
3. bounded by $Bel(A) = l(A)$ and $Pl(A) = u(A), \forall A \in P(X)$.

We will not get into the details of the proof of this statement. More details can be read in [3], [7], and [8].

In the next section, we explain a general linear programming problem with uncertainty and its associated recourse model. A recourse model that we use in our work is an interval expected recourse model. Uncertainty in the model is limited to a probability interval uncertainty that can be presented as a random set.

2.3 Recourse Models for Solving Linear Programming Problems with Uncertainty

2.3.1 Stochastic Expected Recourse Model

We change the pattern of uncertainty to be a vector form. Let $\boldsymbol{x}$ be the vector of variables of size n and $\boldsymbol{c}$ be the vector of coefficients of size n. Consider the following linear programming problem with uncertainty:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.} \quad & \hat{\boldsymbol{A}}_1^T\boldsymbol{x} \geq \hat{b}_1 \\ & \vdots \\ & \hat{\boldsymbol{A}}_m^T\boldsymbol{x} \geq \hat{b}_m \\ & \mathbf{B}\boldsymbol{x} \geq \mathbf{d} \\ & \boldsymbol{x} \geq \mathbf{0}, \end{aligned} \tag{2.3.1}$$

where $(\hat{\boldsymbol{A}}_i, \hat{b}_i)$ is a vector of uncertainty with k_i realizations, for each $i = 1, 2, 3, \ldots, m$. $\mathbf{B}\boldsymbol{x} \geq \mathbf{d}$ is a deterministic constraint. We can write all k_i realizations of $(\hat{\boldsymbol{A}}_i, \hat{b}_i)$ as $(\boldsymbol{A}_{i_1}, b_{i_1}), (\boldsymbol{A}_{i_2}, b_{i_2}), \ldots,$ $(\boldsymbol{A}_{i_{k_i}}, b_{i_{k_i}})$. Each j^{th} realization of $(\hat{\boldsymbol{A}}_i, \hat{b}_i)$ has the associated probability f_{i_j} for $j = 1, 2, 3, \ldots, k_i$. In our work, we will skip $\mathbf{B}\boldsymbol{x} \geq \mathbf{d}$ and consider only uncertain constraints because it has no effect on the complexity of the uncertain problem. An example of an uncertain constraint is as follows. Suppose x_1 and x_2 are the number of bags of brown sugar and white sugar (at the same fixed price per bag), respectively. The i^{th} constraint can be referred to as to a demand constraint, where $\hat{\boldsymbol{A}}_i$ is an uncertain vector of weight (kg) for each bag of brown sugar and white sugar and b_i is an uncertain demand (kg). Note that x_1 and x_2 may not need to be the whole numbers, a customer could get a half a bag of brown sugar for example. The productivities of both types of sugar may have three realizations (Below average, Average and Above average) depending on the weather. The demand b_i also has realizations in the same pattern under economic behavior that means if the productivities is a below average the demand is a below average demand, too, since a manager could set a higher price per bag of sugar to reduce the customer's demand. Therefore, three vector realizations are as follows.

Below average (realization 1)	$(\boldsymbol{A}_{i_1}, b_{i_1}) = (7, 8, 45)$ with probability f_{i_1}
Average (realization 2)	$(\boldsymbol{A}_{i_2}, b_{i_2}) = (8, 9, 50)$ with probability f_{i_2}
Above average (realization 3)	$(\boldsymbol{A}_{i_3}, b_{i_3}) = (9, 10, 53)$ with probability f_{i_3}.

Table 2.1: Realizations of an uncertain demand constraint

When a linear programming problems is deterministic, it can be solved by using a simplex method by adding slack/surplus variables. If a constraint is uncertain, we remodel the problem as a two-stage stochastic expected recourse model. We can rewrite (2.3.1) to be the stochastic expected recourse model as follows.

$$
\begin{aligned}
\min \quad & \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i) \\
\text{s.t.} \quad & \boldsymbol{A}_{1_1}^T\boldsymbol{x} + w_{1_1} \geq b_{1_1} \\
& \boldsymbol{A}_{1_2}^T\boldsymbol{x} + w_{1_2} \geq b_{1_2} \\
& \vdots \\
& \boldsymbol{A}_{1_{k_1}}^T\boldsymbol{x} + w_{1_{k_1}} \geq b_{1_{k_1}} \\
& \vdots \\
& \boldsymbol{A}_{m_1}^T\boldsymbol{x} + w_{m_1} \geq b_{m_1} \\
& \boldsymbol{A}_{m_2}^T\boldsymbol{x} + w_{m_2} \geq b_{m_2} \\
& \vdots \\
& \boldsymbol{A}_{m_{k_m}}^T\boldsymbol{x} + w_{m_{k_m}} \geq b_{m_{k_m}} \\
& \boldsymbol{x}, \boldsymbol{w} \geq \boldsymbol{0},
\end{aligned}
\tag{2.3.2}
$$

where $\boldsymbol{x}$ is the first stage decision vector and $\boldsymbol{w}_i = (w_{i_1}, w_{i_2}, w_{i_3}, \ldots, w_{i_{k_i}})$ is the second stage (recourse) decision vector according to the original i^{th} constraint, with the corresponding probability $\boldsymbol{f}_i = (f_{i_1}, f_{i_2}, f_{i_3}, \ldots, f_{i_{k_i}})$. Each w_{i_j} defines as $w_{i_j} = \max\{(b_{i_j} - \boldsymbol{A}_{i_j}^T\boldsymbol{x},\ 0)\}$. The term $\sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i)$ is the expected recourse value, when $\boldsymbol{s} = (s_1, s_2, s_3, \cdots, s_m)$ is a penalty vector with respect to the original constraints. Example 2.3.1 shows the corresponding stochastic expected recourse problem of the demand constraint of brown sugar and white sugar with minimization of its transportation cost.

Example 2.3.1**.** Consider a linear programming P_1 with uncertain vector $(\hat{\boldsymbol{A}}_1, \hat{b}_1)$

$$
\begin{aligned}
P_1\colon \quad \min \quad & 3x_1 + 2x_2 \\
\text{s.t.} \quad & \hat{\boldsymbol{A}}_1^T \boldsymbol{x} \geq \hat{b}_1 \\
& x \geq 0.
\end{aligned}
$$

Suppose we know realizations of $(\hat{\boldsymbol{A}}_1, \hat{b}_1)$ and their probabilities as shown in Table 2.1. The corresponding stochastic expected recourse problem with recourse variables (w_1, w_2, w_3) of the demand constraint is

$$
\begin{aligned}
P_2 : \min \quad & 3x_1 + 2x_2 + s_1 f_{1_1} w_{1_1} + s_1 f_{1_2} w_{1_2} + s_1 f_{1_3} w_{1_3} \\
\text{s.t.} \quad & 7x_1 + 8x_2 + w_{1_1} \geq 45 \\
& 8x_1 + 9x_2 + w_{1_2} \geq 50 \\
& 9x_1 + 10x_2 + w_{1_3} \geq 53 \\
& \boldsymbol{x}, \boldsymbol{w}_1 \geq \boldsymbol{0}.
\end{aligned}
\tag{2.3.3}
$$

□

How do we interpret a result of the expected recourse problem (2.3.3) if we do not know the exact probability, for example, if $f_{1_1} \in [\frac{1}{3}, \frac{1}{2}], f_{1_2} \in [\frac{1}{6}, \frac{2}{3}]$ and $f_{1_3} \in [\frac{1}{6}, \frac{1}{2}]$? One of the approaches is to use an idea of an interval expected value, which will be presented in the next section.

2.3.2 **Interval Expected Recourse Model**

We know that a linear program with uncertainty cannot have an exact optimal solution. Some common approaches for such a problem are to find the maximum/minimum expected objective value or a minimum of maximum regret solution. In this thesis, we define Ω to be the set of all feasible solutions of problem (2.3.2) and consider only uncertainties that are in the form of interval probabilities which can be written as random sets. Thus, our general uncertain expected recourse problem with unknown probability in probability interval is stated as follows, using the same feasible set Ω as (2.3.2).

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{w}_i)\in\Omega} \quad & \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i), \\ \text{where} \quad & \boldsymbol{f}_i = (f_{i_1}, f_{i_2}, f_{i_3}, \ldots, f_{i_{k_i}}) \text{ and} \\ & f_{i_j} \in [l_{i_j}, u_{i_j}] \;\text{ for all } i = 1,2,3,\ldots,m \text{ and } j = 1,2,3,\ldots,k_i. \end{aligned} \tag{2.3.4}$$

Let $\hat{t}$ be an uncertain information of our concern, and $T = \{t_1, t_2, t_3, \ldots, t_k\}$ be the set of all realizations of $\hat{t}$ where $t_1 \leq t_2 \leq t_3 \leq \cdots \leq t_k$. We use the idea of Bel and Pl for finding the smallest and the largest probabilities on set A, for any $A \subseteq T$ as follows.

$$Bel(A) = l(A) = \max\left(\sum_{t_i\in A} l_i, 1 - \sum_{t_i\in A^c} u_i\right), \tag{2.3.5}$$

$$Pl(A) = u(A) = \min\left(\sum_{t_i\in A} u_i, 1 - \sum_{t_i\in A^c} l_i\right), \tag{2.3.6}$$

where the probability of $\{t_i\}$ is bounded in $[l_i, u_i]$.

Moreover, Nguyen [6] showed the probabilities generating the smallest and the largest expected values of $\hat{t}$ when we have a random set uncertainty as follows.

$$\begin{aligned} \underline{f}(t_1) = \quad & Bel(\{t_1, t_2, \cdots, t_k\}) - Bel(\{t_2, t_3, \cdots, t_k\}) \\ & \vdots \\ \underline{f}(t_i) = \quad & Bel(\{t_i, t_{i+1}, \cdots, t_k\}) - Bel(\{t_{i+1}, t_{i+2}, \cdots, t_k\}) \\ & \vdots \\ \underline{f}(t_k) = \quad & Bel(\{t_k\}) \\ \\ \overline{f}(t_1) = \quad & Bel(\{t_1\}) \\ & \vdots \\ \overline{f}(t_i) = \quad & Bel(\{t_1, t_2, \cdots, t_i\}) - Bel(\{t_1, t_2, \cdots, t_{i-1}\}) \\ & \vdots \\ \overline{f}(t_k) = \quad & Bel(\{t_1, t_2, \cdots, t_k\}) - Bel(\{t_1, t_2, \cdots, t_{k-1}\}), \end{aligned} \tag{2.3.7}$$

where $\underline{f}$ and $\overline{f}$ are probabilities that provide the smallest and the largest expected values, respectively. The boundary of expected value of $\hat{t}$ is $\left[\sum_{j=1}^{k} t_j \underline{f}(t_j), \sum_{j=1}^{k} t_j \overline{f}(t_j)\right]$. This boundary is called an interval expected value. Sometimes we will call $\underline{f}$ as the lower probability. Hence we can find $\underline{f}$ and $\overline{f}$ in the form of l_i and u_i when we have probability intervals represented as random sets. Thus in case of three realizations $t_1 \leq t_2 \leq t_3$, we know that

$$\begin{aligned}
\underline{f}(t_3) &= Bel(\{t_3\}) \\
&= \max(l_3, 1 - u_1 - u_2) \\
\underline{f}(t_2) &= Bel(\{t_2, t_3\}) - Bel(\{t_3\}) \\
&= \max(l_2 + l_3, 1 - u_1) - \max(l_3, 1 - u_1 - u_2) \\
\underline{f}(t_1) &= Bel(\{t_1, t_2, t_3\}) - Bel(\{t_2, t_3\}) \\
&= 1 - \max(l_2 + l_3, 1 - u_1) \\
\overline{f}(t_1) &= Bel(\{t_1\}) \\
&= \max(l_1, 1 - u_2 - u_3) \\
\overline{f}(t_2) &= Bel(\{t_1, t_2\}) - Bel(\{t_1\}) \\
&= \max(l_1 + l_2, 1 - u_3) - \max(l_1, 1 - u_2 - u_3) \\
\overline{f}(t_3) &= Bel(\{t_1, t_2, t_3\}) - Bel(\{t_1, t_2\}) \\
&= 1 - \max(l_1 + l_2, 1 - u_3).
\end{aligned} \tag{2.3.8}$$

For the rest of the thesis, we will work on only three realizations of uncertain vector for each uncertain constraint as stated in the end of chapter 1. We can apply the idea of System (2.3.8) to our vector of uncertainty when its realizations can be ordering; i.e, let $(\hat{\boldsymbol{A}}_i, \hat{b}_i)$ be a vector of uncertainty with k_i realizations: $(\boldsymbol{A}_{i_1}, b_{i_1}), (\boldsymbol{A}_{i_2}, b_{i_2}), \cdots,$ and $(\boldsymbol{A}_{i_{k_i}}, b_{i_{k_i}})$, we have $(\boldsymbol{A}_{i_j}, b_{i_j}) \leq (\boldsymbol{A}_{i_l}, b_{i_l})$ when $j \leq l$. Please note that $(\boldsymbol{A}_{i_j}, b_{i_j}) \leq (\boldsymbol{A}_{i_l}, b_{i_l})$ if and only if $\boldsymbol{A}_{i_j} \leq \boldsymbol{A}_{i_l}$ and $b_{i_j} \leq b_{i_l}$.

Let us continue using Example 2.3.1 to explain how to find probabilities that provide an interval expected recourse model with interval probability (with the properties of random set information) by using the idea (2.3.8).

Example 2.3.2. Consider the expected recourse problem P_2 in Example 2.3.1, when we do not know the probability $\boldsymbol{f}_1 = (f_{1_1}, f_{1_2}, f_{1_3})$ for certain, but we know that $f_{1_1} \in [\frac{1}{3}, \frac{1}{2}], f_{1_2} \in [\frac{1}{6}, \frac{2}{3}]$, and $f_{1_3} \in [\frac{1}{6}, \frac{1}{2}]$. Therefore, to find an interval expected value to the objective function of problem P_2 with the unknown probability $(f_{1_1}, f_{1_2}$ and $f_{1_3})$, we have to know the ordering of w_{1_1}, w_{1_2} and w_{1_3} which are corresponding to vectors of realizations in Table 2.1. All possible ordering cases of w_{1_1}, w_{1_2} and w_{1_3} with their corresponding lower/upper probabilities to get the boundary of interval expected value are in the table below.

Ordering cases	Probability $\underline{\boldsymbol{f}}_i$ provides the smallest expected value			Probability $\bar{\boldsymbol{f}}_i$ provides the largest expected value		
	$\underline{f}_{1_1}$	$\underline{f}_{1_2}$	$\underline{f}_{1_3}$	$\overline{f}_{1_1}$	$\overline{f}_{1_2}$	$\overline{f}_{1_3}$
$w_{1_1} \leq w_{1_2} \leq w_{1_3}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$
$w_{1_1} \leq w_{1_3} \leq w_{1_2}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$
$w_{1_2} \leq w_{1_1} \leq w_{1_3}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$
$w_{1_2} \leq w_{1_3} \leq w_{1_1}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{6}$
$w_{1_3} \leq w_{1_1} \leq w_{1_2}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{3}$
$w_{1_3} \leq w_{1_2} \leq w_{1_1}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{2}$

□

However, a decision maker may prefer to have a solution that can present a reasonable regret for not knowing an exact probability, which leads us to the minimax regret approach. In the next chapter, we provide the minimax regret of an uncertain expected recourse problem, the idea of a relaxation procedure and its improvement using the ordering idea.

CHAPTER III

MINIMAX REGRET OF UNCERTAIN EXPECTED RECOURSE PROBLEM WITH ORDERING

The objective of our work is to improve the relaxation procedure of the minimax regret method by using the idea of ordering. In our work, we use only three realizations of each uncertain vector and up to five uncertain constraints. In this chapter, we first explain the minimax regret of uncertain expected recourse problem. After that we provide a general relaxation procedure, and a relaxation procedure using ordering from the reference [9]. We give a modified version of the relaxation procedure by ordering in the last section.

3.1 Minimax Regret of Uncertain Expected Recourse Problem

A minimax regret of an uncertain expected recourse problem is a problem trying to minimize the maximum regret solution over all unknown probability vectors in the set of all probabilities generated by probability intervals, M, where

$$M = \{(\boldsymbol{f}_1,\boldsymbol{f}_2,\boldsymbol{f}_3,\dots,\boldsymbol{f}_m) \mid \boldsymbol{f}_i = (f_{i_1}, f_{i_2}, f_{i_3}, \cdots, f_{i_{k_i}}) \text{ where } f_{i_j} \in [l_{i_j}, u_{i_j}]\}.$$

Consider the stochastic expected recourse model

$$\begin{aligned} \min \quad & \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i) \\ \text{s.t.} \quad & \boldsymbol{A}_{i_j}^T\boldsymbol{x} + w_{i_j} \geq b_{i_j} \text{ for } i = 1,2,3\dots,m \text{ and } j = 1,2,3,\dots,k_i \\ & \boldsymbol{x},\boldsymbol{w} \geq \boldsymbol{0}. \end{aligned} \tag{3.1.1}$$

The objective function of problem (3.1.1) is to minimize

$$z(\boldsymbol{f},\boldsymbol{x},\boldsymbol{w}) := \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i\boldsymbol{f}_i^T\boldsymbol{w}_i,$$

where $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3 \dots, \boldsymbol{w}_m)$ and $\boldsymbol{w}_i = (w_{i_1}, w_{i_2}, w_{i_3}, \dots, w_{i_{k_i}})$.

We now consider the problem

$$\min_{(\boldsymbol{x},\boldsymbol{w})\in\Omega} z(\boldsymbol{f},\boldsymbol{x},\boldsymbol{w}).$$

Given a probability $\boldsymbol{f}\in M$, the regret shows the amount of the objective value that a candidate solution $(\boldsymbol{x},\boldsymbol{w})$ deviates from the true objective value, which can be expressed by

$$r(\boldsymbol{f},\boldsymbol{x},\boldsymbol{w}) := \Big(z(\boldsymbol{f},\boldsymbol{x},\boldsymbol{w}) - \min_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} z(\boldsymbol{f},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\Big),$$

where $\min\limits_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} z(\boldsymbol{f},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})$ is the true objective value with this given probability $\boldsymbol{f}$.

The best of the worst regret (minimax regret) over all $\boldsymbol{f}\in M$ is

$$\min_{(\boldsymbol{x},\boldsymbol{w})\in\Omega}\max_{\boldsymbol{f}\in M}\Big(z(\boldsymbol{f},\boldsymbol{x},\boldsymbol{w}) - \min_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} z(\boldsymbol{f},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\Big). \tag{3.1.2}$$

A method to find an optimal solution of the minimax regret for an uncertain expected recourse problem has four general steps as follows.

Algorithm 1: General relaxation procedure (see [9]).

1. Initialization. Choose $\boldsymbol{f}^{(1)}\in M$ and solve $\min\limits_{(\boldsymbol{x},\boldsymbol{w})\in\Omega} z(\boldsymbol{f}^{(1)},\boldsymbol{x},\boldsymbol{w})$ to obtain its optimal solution $(\bar{\boldsymbol{x}}^{(1)},\bar{\boldsymbol{w}}^{(1)})$, then set $p=1$.

2. Solve the following current relaxed problem to obtain an optimal solution $(R^{(p)};(\boldsymbol{x}^{(p)},\boldsymbol{w}^{(p)}))$.

$$\begin{aligned}\min_{R;(\boldsymbol{x},\boldsymbol{w})\in\Omega}\ & R\\ \text{s.t. } & R\geq 0\\ & R\geq\Big(z(\boldsymbol{f}^{(i)},\boldsymbol{x},\boldsymbol{w}) - z(\boldsymbol{f}^{(i)},\bar{\boldsymbol{x}}^{(i)},\bar{\boldsymbol{w}}^{(i)})\Big), i=1,2,3,\ldots,p.\end{aligned} \tag{3.1.3}$$

3. Obtain an optimal solution $(\boldsymbol{f}^{(p+1)},\bar{\boldsymbol{x}}^{(p+1)},\bar{\boldsymbol{w}}^{(p+1)})$ where its optimal value $Z^{(p)}$ is

$$Z^{(p)} = \max_{\boldsymbol{f}\in M;(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega}\Big(z(\boldsymbol{f},\boldsymbol{x}^{(p)},\boldsymbol{w}^{(p)}) - z(\boldsymbol{f},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\Big).$$

4. If $Z^{(p)} \leq R^{(p)}$, terminate the procedure. An optimal solution to the minimax expected regret model is $(R^{(p)};\boldsymbol{x}^{(p)},\boldsymbol{w}^{(p)})$. Otherwise, set $p = p+1$ then return to step 2 . $\square$

3.2 The idea of ordering

Let $\hat{t}$ be an uncertain information of our concern, $T = \{t_1, t_2, t_3, \ldots, t_k \,| t_i \in \mathbb{R}\}$ be the set of all realizations of $\hat{t}$, where $\hat{t}$ can be ordering as $t_1 \leq t_2 \leq t_3 \leq \cdots \leq t_k$. Boodgumarn [3] said that we can use the idea of Bel and Pl for finding the smallest and the largest values of probability on set A, for any $A \subseteq T$ from equation (2.3.5) and (2.3.6). When a probability is bounded in $[l_i, u_i]$, we know all possible ordering cases of all realizations of a probability interval uncertainty that can be represented by a random set. We can use a method in Nguyen [6] to find the largest and the smallest expected values of $\hat{t}$ from system (2.3.7). We apply this idea to step 3 of Algorithm 1 when we distribute the term $Z^{(p)}$ as follows.

$$
\begin{aligned}
Z_{\boldsymbol{f}}^{(p)} &= \max_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} \Big(z(\boldsymbol{f},\boldsymbol{x}^{(p)},\boldsymbol{w}^{(p)}) - z(\boldsymbol{f},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}}) \Big) \\
&= \max_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} \Big(\boldsymbol{c}^T(\boldsymbol{x}^{(p)} - \bar{\boldsymbol{x}}) + s_1 \sum_{j=1}^{k_1} f_{1_j}(w_{1_j}^{(p)} - \bar{w}_{1_j}) + \cdots + s_m \sum_{j=1}^{k_m} f_{m_j}(w_{m_j}^{(p)} - \bar{w}_{m_j}) \Big).
\end{aligned}
$$

We start step 3 by using $\boldsymbol{f}$ and the optimal solution $(\boldsymbol{x}^{(p)},\boldsymbol{w}^{(p)})$ from step 2. We then find the different values of terms $(w_{1_j}^{(p)} - \bar{w}_{1_j}), (w_{2_j}^{(p)} - \bar{w}_{2_j}), (w_{3_j}^{(p)} - \bar{w}_{3_j}), \ldots, (w_{m_j}^{(p)} - \bar{w}_{m_j})$ and make the ordering of them. We can find probability h_{i_j} that corresponding to the ordering when $\boldsymbol{h} = (\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3, \cdots, \boldsymbol{h}_m) \in \overline{M}$, $\boldsymbol{h}_i = (h_{i_1}, h_{i_2}, h_{i_3}, \cdots, h_{i_{k_i}})$ and h_{i_j} is corresponding to the order of $(w_{i_j}^{(p)} - \bar{w}_{i_j,\boldsymbol{f}})$. Moreover $(\bar{\boldsymbol{x}}_{\boldsymbol{h}}, \bar{\boldsymbol{w}}_{\boldsymbol{h}})$ is an optimal solution for $\min_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} z(\boldsymbol{h},\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})$ and $Z_{\boldsymbol{f}}^{(p)} \leq Z_{\boldsymbol{h}}^{(p)}$. We continue to find a new h_{i_j} by using the following system.

$$
\begin{aligned}
Z_{\boldsymbol{f}}^{(p)} &= \boldsymbol{c}^T(\boldsymbol{x}^{(p)} - \bar{\boldsymbol{x}}_{\boldsymbol{f}}) + s_1 \sum_{j=1}^{k_1} f_{1_j}(w_{1_j}^{(p)} - \bar{w}_{1_j,\boldsymbol{f}}) + \cdots + s_m \sum_{j=1}^{k_m} f_{m_j}(w_{m_j}^{(p)} - \bar{w}_{m_j,\boldsymbol{f}}) \\
&\leq \boldsymbol{c}^T(\boldsymbol{x}^{(p)} - \bar{\boldsymbol{x}}_{\boldsymbol{f}}) + s_1 \sum_{j=1}^{k_1} h_{1_j}(w_{1_j}^{(p)} - \bar{w}_{1_j,\boldsymbol{f}}) + \cdots + s_m \sum_{j=1}^{k_m} h_{m_j}(w_{m_j}^{(p)} - \bar{w}_{m_j,\boldsymbol{f}}) \\
&\leq \boldsymbol{c}^T(\boldsymbol{x}^{(p)} - \bar{\boldsymbol{x}}_{\boldsymbol{h}}) + s_1 \sum_{j=1}^{k_1} h_{1_j}(w_{1_j}^{(p)} - \bar{w}_{1_j,\boldsymbol{h}}) + \cdots + s_m \sum_{j=1}^{k_m} h_{m_j}(w_{m_j}^{(p)} - \bar{w}_{m_j,\boldsymbol{h}}) \\
&= \max_{(\bar{\boldsymbol{x}},\bar{\boldsymbol{w}})\in\Omega} (z(\boldsymbol{h}, \boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)}) - z(\boldsymbol{h}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{w}})) \\
&= Z_{\boldsymbol{h}}^{(p)}.
\end{aligned} \tag{3.2.1}
$$

3.3 Relaxation Procedure by Ordering

Thipwiwatpojana and Lodwick [9] used the ordering idea (3.2.1) to improve the general relaxation procedure by changing the set M into $\overline{M}$, where $\overline{M}$ is the set of probabilities of all possible ordering cases of realizations that provide the smallest expected values in step 3 of the p^{th} iteration of Algorithm 1. We follow step 1 and 2 from the Algorithm 1 and use the idea (3.2.1) of ordering in step 3, which helps to obtain a modified version of the relaxation procedure in Algorithm 2 as follows.

Algorithm 2: Relaxation procedure by ordering of realizations see ([9]).

1. Initialization. Choose $\boldsymbol{f}^{(1)} \in M$ and solve $\min\limits_{(\boldsymbol{x},\boldsymbol{w})\in\Omega} z(\boldsymbol{f}^{(1)}, \boldsymbol{x}, \boldsymbol{w})$ to obtain its optimal solution $(\bar{\boldsymbol{x}}^{(1)}, \bar{\boldsymbol{w}}^{(1)})$, then set $p = 1$.

2. Solve the following current relaxed problem to obtain an optimal solution $(R^{(p)}; (\boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)}))$.

$$
\begin{aligned}
&\min_{R;(\boldsymbol{x},\boldsymbol{w})\in\Omega} R \\
&\qquad \text{s.t. } R \geq 0 \\
&\qquad\quad R \geq \left(z(\boldsymbol{f}^{(i)}, \boldsymbol{x}, \boldsymbol{w}) - z(\boldsymbol{f}^{(i)}, \bar{\boldsymbol{x}}^{(i)}, \bar{\boldsymbol{w}}^{(i)})\right), i = 1, 2, 3, \ldots, p.
\end{aligned} \tag{3.3.1}
$$

3. Start with $\boldsymbol{f}^{(p)}$ and work on the system (3.2.1) in order to find $\boldsymbol{h}$ then calculate $Z_{\boldsymbol{h}}^{(p)}$ and

its optimal solution $(\bar{\boldsymbol{x}}^{(p)}, \bar{\boldsymbol{w}}^{(p)})$.

4. If $Z_{\boldsymbol{h}}^{(p)} > R^{(p)}$, set $p = p + 1$ and $\boldsymbol{f}^{(p)} = \boldsymbol{h}$, then return to step 2. Otherwise, go to next step.

5. If $Z_{\boldsymbol{h}}^{(p)} \leq R^{(p)}$, select $\boldsymbol{f}^{(p)} \in \overline{M} \neq \emptyset$ that has not been used in this iteration of step 3 and reprocess System (3.2.1) until we obtain $\boldsymbol{h}$ such that $Z_{\boldsymbol{h}}^{(p)} > R^{(p)}$, then continue working on step 4. Otherwise, $\overline{M} = \emptyset$ then we terminate the procedure. Finally, an optimal solution of the minimax expected regret model (3.1.2) is $(R^{(p)}; \boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)})$. □

We continue modifying Algorithm 2, by removing $\boldsymbol{f}^{(i)}; i = 1, \cdots, p-1$, in step 2 from the set $\overline{M}$ before using it in step 3, since we had already knew the value of these $(z(\boldsymbol{f}^{(i)}, \boldsymbol{x}, \boldsymbol{w}) - z(\boldsymbol{f}^{(i)}, \bar{\boldsymbol{x}}^{(i)}, \bar{\boldsymbol{w}}^{(i)}))$ from (3.3.1), we only need to check $Z_{\boldsymbol{h}}$ for the rest of $\boldsymbol{h} \in \overline{M}$. Checking the correspondence between ordering of $(w_{i_j} - \bar{w}_{i_j,\boldsymbol{h}})$ and the probability $\boldsymbol{h}$ when every $Z_{\boldsymbol{h}}^{(p)} \leq R^{(p)}$ at step 5, if they are corresponding to each order, we stop the algorithm, i.e., we do not need to keep doing until getting $\overline{M} = \emptyset$.

3.4 Modify Relaxation Procedure by Ordering

Algorithm 3: Modify relaxation procedure by ordering of realization and reducing the size of $\overline{M}$.

1. Initialization. Choose $\boldsymbol{f}^{(1)} \in M$ and solve $\min\limits_{(\boldsymbol{x},\boldsymbol{w}) \in \Omega} z(\boldsymbol{f}^{(1)}, \boldsymbol{x}, \boldsymbol{w})$ to obtain its optimal solution $(\bar{\boldsymbol{x}}^{(1)}, \bar{\boldsymbol{w}}^{(1)})$, then set $p = 1$.

2. Solve the following current relaxed problem to obtain an optimal solution $(R^{(p)}; (\boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)}))$.

$$\begin{aligned} &\min_{R;(\boldsymbol{x},\boldsymbol{w}) \in \Omega} R \\ &\quad \text{s.t. } R \geq 0 \\ &\quad\quad R \geq \left(z(\boldsymbol{f}^{(i)}, \boldsymbol{x}, \boldsymbol{w}) - z(\boldsymbol{f}^{(i)}, \bar{\boldsymbol{x}}^{(i)}, \bar{\boldsymbol{w}}^{(i)})\right), i = 1, 2, 3, \ldots, p. \end{aligned} \tag{3.4.1}$$

3. If $\overline{M} = \emptyset$, $(R^{(p)}$ then $\boldsymbol{x}^{(p)}; \boldsymbol{w}^{(p)})$ is an optimal solution. If $\overline{M} \neq \emptyset$, start with $\boldsymbol{f}^{(p)}$ and set $\overline{M} = \overline{M} \backslash \{ \boldsymbol{f}^{(i)} \mid i = 1, 2, 3, \ldots, p\}$, then work on the system (3.2.1) in order to find $\boldsymbol{h}$.

If $\boldsymbol{h} \notin \overline{M}$ then $(R^{(p)}; \boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)})$ is an optimal solution for our minimax regret problem. Else, calculate $Z_{\boldsymbol{h}}^{(p)}$ and its optimal solution $(\bar{\boldsymbol{x}}^{(p)}, \bar{\boldsymbol{w}}^{(p)})$.

4. If $Z_{\boldsymbol{h}}^{(p)} > R^{(p)}$, set $p = p + 1$, and $\boldsymbol{f}^{(p)} = \boldsymbol{h}$, then return to step 2. Otherwise go to the next step.

5. If $Z_{\boldsymbol{h}}^{(p)} \leq R^{(p)}$, set $M_1 = \overline{M}$. Select $\boldsymbol{f}^{(p)} \in M_1 \neq \emptyset$ that has not been used in this iteration of step 3, and reprocess the system (3.2.1) until we find $\boldsymbol{h}$ such that $Z_{\boldsymbol{h}}^{(p)} > R^{(p)}$, then continue to step 4. Otherwise, $M_1 = \emptyset$ and we terminate the procedure with an optimal solution to the minimax expected regret model □

We already show three algorithms of relaxation procedure. We then implement the codes and find the average time in MATLAB using computer with Processor: Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz (8 CPUs), 3.4GHz Memory: 8192MB RAM, Windows 10. In the next chapter, we first provide the step of setting up our algorithm and finally provide the results and discussions of all algorithm.

CHAPTER IV

RELAXATION PROCEDURE WITH MATLAB

From previous chapter, we know that we can reduce the size of probability boundary (the lower probabilities, $\underline{f}$) of all possible ordering cases by using the idea of ordering (3.2.1). In this chapter, we want to reduce the calculation times by implementing all algorithms in MATLAB. We first random 100 uncertain examples each for up to five decision variables and up to five uncertain constraints. Thus we have the total of 25 cases. Each case contains 100 random uncertain examples. Then we find an optimal solution and record the time used for each example in each algorithm. Finally, we find the average time of each case by using the process below.

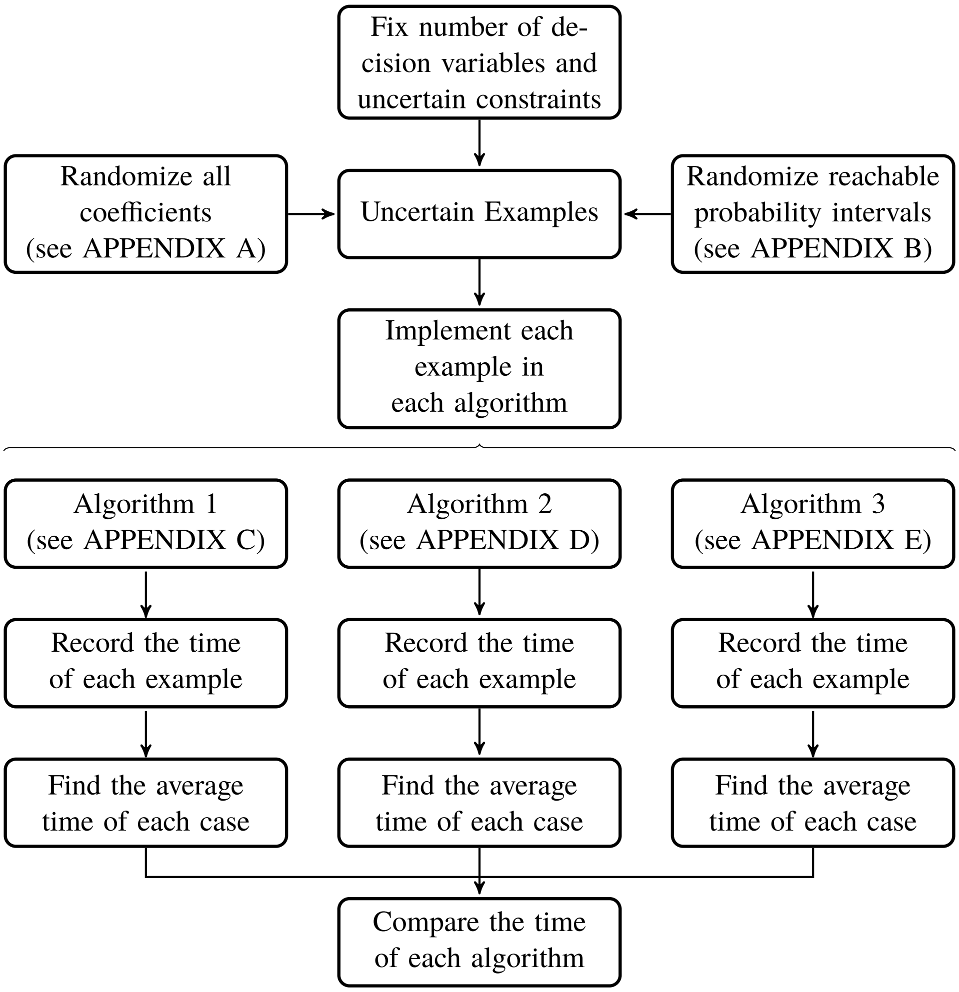

Figure 4.1: The process to find the average time used in Algorithms 1, 2 and 3.

Examples in our work are stochastic expected recourse problems that we limit the number of each realizations of uncertain vectors and uncertain constraints to be only three realizations and up to five constraints. Uncertain entities in our example are probability intervals that can be represented as random sets. A problem of five constraints and three realizations for each constraint, the total scenarios of a linear programming with uncertainty are 15 scenarios. If we reduce the number of constraints, the scenarios decreases corresponding to the constraints.

4.1 **The Process for Finding an Average Time**

Step 0: Realizations of uncertain vectors is fixed to be three realizations.

Step 1: Fix a number of up to five decision variables and up to five uncertain constraints. Then randomize $\boldsymbol{c} \in [-100, 100]$ and each component of an uncertain vector $(\boldsymbol{A}, \boldsymbol{b})$ in $[-100, 100]$. Check that all realizations can be ordering. Also randomize s_i in $[1, 100]$ because the problem is minimizing. Probability interval must be able to be represented by the random set (see APPENDIX A). A generated uncertain recourse example is as follows.

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{w}_i)\in\Omega} \quad & \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i), \\ \text{s.t.} \quad & \boldsymbol{A}_{i_j}^T\boldsymbol{x} + w_{i_j} \geq b_{i_j}, \\ & \boldsymbol{x}, \boldsymbol{w} \geq \boldsymbol{0}, \\ \text{where} \quad & f_{i_j} \in [l_{i_j}, u_{i_j}]; \ \text{for} i = 1, 2, \ldots, m, \ \ j = 1, 2, \ldots, k_i. \end{aligned}$$

Check the feasibility of the uncertain recourse example. Keep randomize all parameters until we have 100 feasible uncertain recourse examples.

Step 2: Implement each of the feasible uncertain recourse examples with Algorithm 1, 2 and 3. We start with the same process in step 1 and step 2 of each algorithm. We then use an optimal solution $(R^{(p)}; (\boldsymbol{x}^{(p)}, \boldsymbol{w}^{(p)}))$ in step 3. This step is very computationally intensive. We start with Algorithm 1 for the original relaxation procedure (see APPENDIX C). We apply the idea of ordering to reduce size of M in the Algorithm 2 (see APPENDIX D) and modify the Algorithm 2 to reduce size of M and reduce calculate time in the Algorithm 3 (see APPENDIX E).

Step 3: We then record the times of each of 100 examples and finally find the average time of the fixed case. (see APPENDIX A).

Step 4: Redo step 1-3 using the other fixed number of decision variables and uncertain constraints, until complete all 25 cases. Average times and regrets of each case are shown in APPENDIX F.

4.1.1 **An Experimental**

We compare times of all three algorithm for each case. Note that the optimal regret got from three algorithms (by using the same problem) are the same, please see APPENDIX F. The average time of each case is shown as follows.

Problems with 1 variable	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.1711	0.3919	3.3691	24.9854	166.9415
Algorithm 2	0.0700	0.5548	1.5061	9.2054	65.9210
Algorithm 3	0.0585	0.1957	0.9731	4.8609	37.7121

Table 4.1: The average time of 100 examples of one decision variable and up to five uncertain constraints

Problems with 2 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.0515	0.4867	2.8889	20.2407	106.8274
Algorithm 2	0.0534	0.3502	1.7999	15.4527	96.9212
Algorithm 3	0.0507	0.1953	1.0525	6.1747	48.3515

Table 4.2: The average time of 100 examples of two decision variables and up to five uncertain constraints

Problems with 3 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.0906	0.4333	2.6587	23.4222	128.7291
Algorithm 2	0.0755	0.3074	2.0604	13.0668	91.6097
Algorithm 3	0.0600	0.2387	1.2470	7.8081	52.0429

Table 4.3: The average time of 100 examples of three decision variables and up to five uncertain constraints

Problems with 4 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.1069	0.4218	3.3827	18.5621	138.7532
Algorithm 2	0.0713	0.2563	2.3472	15.1171	112.7372
Algorithm 3	0.0655	0.1966	1.3934	10.5068	54.9573

Table 4.4: The average time of 100 examples of four decision variables and up to five uncertain constraints

Problems with 5 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.0811	0.4398	2.7904	22.2362	181.4090
Algorithm 2	0.0646	0.2742	2.1166	16.7340	119.8803
Algorithm 3	0.0589	0.2043	1.3776	9.7868	75.2131

Table 4.5: The average time of 100 examples of five decision variables and up to five uncertain constraints

Problems with 100 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.1080	0.2031	2.9017	51.0962	567.6350
Algorithm 2	0.0984	0.1531	2.4377	25.5042	269.3758
Algorithm 3	0.0805	0.1439	1.3570	15.1525	145.1933

Table 4.6: The average time of 100 examples of 100 decision variables and up to five uncertain constraints

From Table 4.1- 4.5, we can see the trend that the lowest average time happens when using Algorithm 3. We also illustrate in Table 4.6 with the large amount of variables (100 variables) to show that the number of decision variables would effect the average time in the case of five constraints. This means, our algorithm should be reasonable for a moderate amount of variables.

Problem with up to 5 variables	Average time (second)				
# of constraints	1	2	3	4	5
Algorithm 1	0.1002	0.4347	3.0180	21.8893	144.5320
Algorithm 2	0.0670	0.3486	1.9660	13.9152	97.3539
Algorithm 3	0.0587	0.2016	1.2087	7.8275	53.6554

Table 4.7: The average time of 100 examples of up to five decision variable and up to five constraints

By using randomized 100 problems of up to five decision variables and up to five uncertain constraints, we also show in Table 4.7 and Fig 4.2 that Algorithm 3 will always provide the best average time.

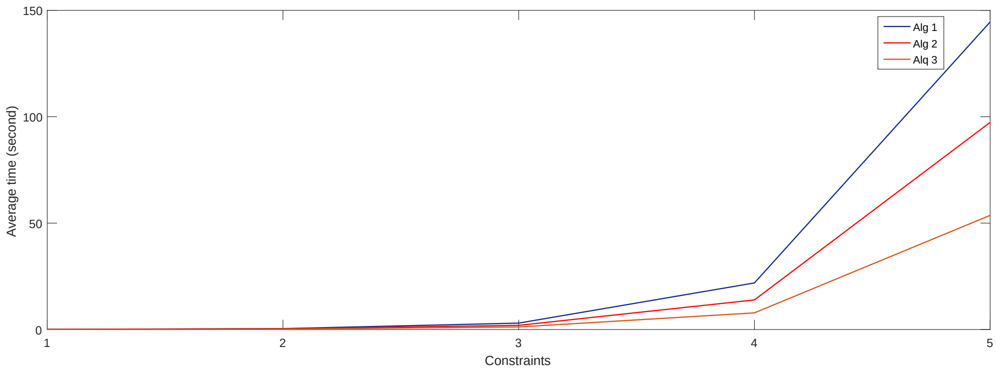

Figure 4.2: The average time of 100 problems of up to five decision variable and up to five uncertain constraints

Although the average time in each table shows the best results when using Algorithm 3, in some cases Algorithm 2 or 3 may provide higher average times than Algorithm 1 because it we may use all possible cases of probabilities in Algorithm 2 or 3 to check ordering in the worst cases.

We show some regrets (R) for each case of up to five decision variables and up to five uncertain constraints (see APPENDIX F). However, we skip reporting the associated optimal solutions since each problem may not have a unique optimal solution. The regret got from three algorithms are the same for each problem while the average time is the best in Algorithm 3. It means that we can use the idea of ordering to reduce the size of the probability set of all possible cases and reduce the calculate times.

CHAPTER V

CONCLUSION

In this thesis, we improve the relaxation procedure by applied the analysis written in [9] to our vectors of uncertainty. Therefore the uncertain entities in this thesis is said to be a special case of general uncertainty in [9] that means we change the form of uncertainty to be vectors of uncertainty and assume that $(\hat{\boldsymbol{A}}_i, \hat{b}_i)$ be a vector of uncertainty with k_i realizations: $(\boldsymbol{A}_{i_1}, b_{i_1}), (\boldsymbol{A}_{i_2}, b_{i_2}), \cdots,$ and $(\boldsymbol{A}_{i_{k_i}}, b_{i_{k_i}})$, such that $(\boldsymbol{A}_{i_j}, b_{i_j}) \leq (\boldsymbol{A}_{i_l}, b_{i_l})$ when $j \leq l$. All vectors of uncertainty together with probability intervals that can be represented by the random sets. We use only up to three realizations and up to five constraints. Finally we can reduce the average calculation time of the relaxation procedure by using the idea of ordering and reduce the size of probability boundary (the lower probabilities) of all possible ordering cases at each iteration. In minimax regret approach, we clearly improved Algorithm 2. by reducing the set of $\overline{M}$ in each iteration that we already explain in Chapter III and IV.

The result can guarantee that if the realizations can be ordering and uncertain information is probability intervals that can be represented by random sets. We can use the idea of ordering to reduce the size of $\overline{M}$ and reduce the calculate time.

References

1. Igor Averbakh. Minmax regret solutions for minimax optimization problems with uncertainty. Operations Research Letters, 27(2):57–65, 2000.

2. Peeraporn Boodgumarn. Solving linear programming problem with uncertainty: Probability interval and random set parameters. Master's thesis, Chulalongkorn University, 2012.

3. Peeraporn Boodgumarn, Phantipa Thipwiwatpotjana, and Weldon A Lodwick. When a probability interval is a random set. ScienceAsia, 39(3):319–326, 2013.

4. Masahiro Inuiguchi and Masatoshi Sakawa. Minimax regret solution to linear programming problems with an interval objective function. European Journal of Operational Research, 86(3):526–536, 1995.

5. Helmut E Mausser and Manuel Laguna. A heuristic to minimax absolute regret for linear programs with interval objective function coefficients. European Journal of Operational Research, 117(1):157–174, 1999.

6. Hung T Nguyen. An introduction to random sets. CRC press, 2006.

7. Phantipa Thipwiwatpotjana and Weldon A Lodwick. The relationship between a probability interval and a random set. In 2011 Annual Meeting of the North American Fuzzy Information Processing Society, 2011.

8. Phantipa Thipwiwatpotjana and Weldon A Lodwick. A relationship between probability interval and random sets and its application to linear optimization with uncertainties. Fuzzy Sets and Systems, 231:45–57, 2013.

9. Phantipa Thipwiwatpotjana and Weldon A Lodwick. Pessimistic, optimistic, and minimax regret approaches for linear programs under uncertainty. Fuzzy Optimization and Decision Making, 13(2):151–171, 2014.

APPENDICES

APPENDIX A: Comparing time of all examples

The stochastic expected recourse problem in our work is

$$
\begin{aligned}
\min_{(\boldsymbol{x},\boldsymbol{w}_i)\in\Omega} \quad & \boldsymbol{c}^T\boldsymbol{x} + \sum_{i=1}^{m} s_i(\boldsymbol{f}_i^T\boldsymbol{w}_i), \\
\text{s.t.} \quad & \boldsymbol{A}_{i_j}^T\boldsymbol{x} + w_{i_j} \geq b_{i_j}, \\
& \boldsymbol{x}, \boldsymbol{w} \geq \boldsymbol{0}, \\
\text{where} \quad & f_{i_j} \in [l_{i_j}, u_{i_j}]; \text{ for } i = 1, 2, \ldots, m, \;\; j = 1, 2, \ldots, k_i.
\end{aligned}
$$

```
function Run
t1=[]; t2=[]; t3=[];         % Time used in Algorithm 1,2,3
r1=[]; r2=[]; r3=[];         % Regrat of Algorithm 1,2,3
count = 0;                   % Count number of examples
r = @() 100*(2*rand-1);      % Random value blw [-100,100]
q = @() [r() r() r()];       % Random vector
N = 100;                     % Number of examples
while count!=N
   a=[]; c=[]; fs=[];
   s=[]; b=[]; fl=[];
   n = input('Nuber of variables : '); % #Variables
   m = input('Nuber of constraints : '); % #Const (1 to 5)
   [count+1 n m]
   for i = 1:n
      e = [];
      for j = 1:m;
         e = [e q()];
      end
      a = [a e'];            % Coef of x in constraint
      c = [c r()];           % Coef of x in objective
   end
   for i = 1:m
      s = [s 100*rand];      % Penalty
      b = [b q()];           % Right hand side
      [fsml,flar] = Prob(); % Probability f
      fs = [fs fsmal];       % f smallest expected value
      fl  = [fl flarg];      % f largest expected value
```

```
    end
% Find the solution (x,w) from the algorithm 1,2,3
    [T1 R1 X1] = Algorithm1(m,n,a,c,s,b,fs);
    [T2 R2 X2] = Algorithm2(m,n,a,c,s,b,fs,fl);
    [T3 R3 X3] = Algorithm3(m,n,a,c,s,b,fs,fl);
% Check unbound of the algorithm 1,2,3
    if abs([R1,R2,R3])<10^7
        t1 = [t1 T1]; r1 = [r1 R1];
        t2 = [t2 T2]; r2 = [r2 R2];
        t3 = [t3 T3]; r3 = [r3 R3];
        count = count+1;
    end
end
% Display the regrat values and time of the problem
[r1; r2; r3]
A1 = [1 mean(t1)];
A2 = [2 mean(t2)];
A3 = [3 mean(t3)];
formatSpec = 'Algorithm %d : Time = %.4f\n';
fprintf(formatSpec,A1,A2,A3);
% Plot the time that used to solve the problem
plot(1:N,t1); hold on;
plot(1:N,t2)
plot(1:N,t3)
```

The following functions are subroutines of the Algorithm 1, 2 and 3, which these subroutines are used often in all algorithms. Thus, we bring them to create the functions that including; sort descending of vector with three elements, generate all possible cases of probabilities and permutation ($P_{n,r}$), respectively, as the following.

```
function ord = Ordering(x) % Sort descending of x
for i = 1:length(x)
    if      x(i)==min(x);  ord(3) = i;
    elseif  x(i)==max(x);  ord(1) = i;
    else                   ord(2) = i;
    end
end
```

```
function F = Expand(m,f)  % Generate all cases of prob
if size(f,1)==6
   k = GenOrdf(6,m);
   for i = 1:length(k)
      h = [];
      for j = 1:m
         h = [h f(k(i,j),(1:3)+3*(j-1))];
      end
      F(i,:) = h;
   end
else
   F = [];
   for i = 1:m
      F = [F repmat(f(i),1,3)];
   end
end
```

```
function P = GenOrdf(n,r)   % Permutation P(n,r)
if n==1
   P = ones(1,r);
elseif r==1
   P = (1:n)';
end
L = n^r;                   % Number of rows in outputs
P = zeros(L,r);            % Pre allocation matrix
v = [-(n-1) ones(1,n-1)]'; % These values put into P
T = v(:,ones(L/n,1));      % Instead of repmatting
P(:,r) = T(:);             % We don't need to do 2 loops
P(1:n^(r-1):L,1) = v;      % The 1st col is the simplest
for i = r-1:-1:2
   R = 1:n^(i-1):L;        % Index into rows for this col
   X = length(R);          % Find length of R
   T = v(:,ones(X/n,1));   % Match dimension
   P(R,r-i+1) = T(:);      % Build it up, insert values
end
P(1,:) = 1;                % For proper cumsumming
P = cumsum(P);             % This is the time hog
```

APPENDIX B: Find reachable probability

This program finds reachable probability. We then find probability f_i that they provide the smallest expected value. We use the concept of (2.3.8) and imply it as follows.

```
function [fsmal flarg] = Prob()
Cord = 1;                   % Check ordering
Crcb = true;                % Check reachable
p = perms([3 2 1]);         % All possible ordering
while Cord!=0
   while Crcb
% Random prob intervals a,b,c
a = [rand rand]; b = [rand rand]; c = [rand rand];
L = [min(a) min(b) min(c)];
U = [max(a) max(b) max(c)];
% Check prob intervals are reachable
CL = [U(1)+L(2)+L(3) U(2)+L(1)+L(3) U(3)+L(1)+L(2)];
CU = [L(1)+U(2)+U(3) L(2)+U(1)+U(3) L(3)+U(1)+U(2)];
if CL(1)>1||CL(2)>1||CL(3)>1||CU(1)<1||CU(2)<1||CU(3)<1
      Crcb = true;
else  Crcb = false; end
end
for i = 1:length(p)
l = L(p(i,:));
u = U(p(i,:));
% Prob f provides the smallest expected
fs(p(i,1)) = 1-max(l(2)+l(3),1-u(1));
fs(p(i,2)) = max(l(2)+l(3),1-u(1))-max(l(3),1-u(1)-u(2));
fs(p(i,3)) = max(l(3),1-u(1)-u(2));
fsmal(i,:) = fs;
% Prob f provides the largest expected
fl(p(i,1)) = max(l(1),1-u(2)-u(3));
fl(p(i,2)) = max(l(1)+l(2),1-u(3))-max(l(1),1-u(2)-u(3));
fl(p(i,3)) = 1-max(l(1)+l(2),1-u(3));
flarg(i,:) = fl;
q(i,:) = Ordering(fs);
end
Cord = norm(p-q);
Crcb = true;
end
```

APPENDIX C: Algorithm 1

```
function [T R xwp] = Algorithm1(m,n,a,c,s,b,fs)
opt = optimset('Display','off');
tic
N = n+3*m;                % #Unknown variables x,w
s = Expand(m,s);
f = Expand(m,fs);
X = [];
% Step 1 ----------------------------------------------
F = f(1,:);
z = [c s.*F(1,:)];
A = [-a -eye(3*m)];
lb = zeros(1,N);
xwb = linprog(z,A,-b,[],[],lb,[],[],opt);
X = [X xwb];
p = 1;                    % Number of iterations
% Step 2 ----------------------------------------------
Check = true;
while Check
% --------------------- Find R(p) ---------------------
   z = [zeros(1,N) 1];
   A = [-a -eye(3*m) zeros(3*m,1)];
   B = -b;
   for i = 1:p
      A = [A; c s.*F(i,:) -1];
      q = c*X(1:n,i)+s.*F(i,:)*X(n+1:end,i);
      B = [B q];
   end
   lb = zeros(1,N+1);
   [xwp,R(p)] = linprog(z,A,B,[],[],lb,[],[],opt);
% --------------------- Find Z(p) ---------------------
   for i = 1:6^m
      z1(i) = c*xwp(1:n)+s.*f(i,:)*xwp(n+1:end-1);
      z = [c s.*f(i,:)];
      A = [-a -eye(3*m)];
      lb = zeros(1,N);
      [xw,z2(i)] = linprog(z,A,-b,[],[],lb,[],[],opt);
      xwb(:,i) = xw;
```

```
    end
    [Z(p),k] = max(z1-z2);
    F = [F; f(k,:)];
    X = [X xwb(:,k)];
% -------------------- Check Unbound --------------------
    if abs([Z(p),R(p)])>10^7
       break;
    end
% ------------------- Check Z(p)>R(p) -------------------
    if Z(p)>R(p) && p<=50
          p = p+1;
          Check = true;
    else  Check = false;
    end
end
T = toc;
R = R(end);
```

APPENDIX D: Algorithm 2

```
function [T R xwp] = Algorithm2(m,n,a,c,s,b,fs,fl)
opt = optimset('Display','off');
tic
N = n+3*m;                  % #Unknown variables x,w
d = []; X = [];
p = perms([3 2 1]);
for i = 1:m
   d = [d p(:,3) p(:,2) p(:,1)];
end
d = Expand(m,d);
s = Expand(m,s);
fl  = Expand(m,fl);
fs = Expand(m,fs);
zval = @(x,f) c*x(1:n)+s.*f*x(n+1:end-1);
% Step 1 ---------------------------------------------
F = fs(1,:);
z = [c s.*F(1,:)];
A = [-a -eye(3*m)];
lb = zeros(1,N);
xwb = linprog(z,A,-b,[],[],lb,[],[],opt);
X = [X xwb];
p = 1;                      % Number of iterations
% Step 2 ----------------------------------------------
Check = true;
while Check
% --------------------- Find R(p) ---------------------
   z = [zeros(1,N) 1];
   A = [-a -eye(3*m) zeros(3*m,1)];
   B = -b;
   for i = 1:p
      A = [A; c s.*F(i,:) -1];
      q = c*X(1:n,i)+s.*F(i,:)*X(n+1:end,i);
      B = [B q];
   end
   lb = zeros(1,N+1);
   [xwp,R(p)] = linprog(z,A,B,[],[],lb,[],[],opt);
% --------------------- Find Z(p) ---------------------
```

```
    z = [c s.*F(p,:) -zval(xwp,F(p,:))];
    A = [-a -eye(3*m) zeros(3*m,1)];
    Aeq = [zeros(1,N) 1]; beq = 1;
    xwb = linprog(z,A,-b,Aeq,beq,lb,[],[],opt);
    h = ProbProvMax(m,n,xwp-xwb,fl,d);
    z = [c s.*h -zval(xwp,h)];
    [xwb,val] = linprog(z,A,-b,Aeq,beq,lb,[],[],opt);
    Z(p) = -val;
    F = [F; h];
    X = [X xwb(1:end-1)];
% ------------------ Check Z(p)<=R(p) -------------------
    if Z(p)<=R(p)
       A = [-a -eye(3*m)];
       lb = zeros(1,N);
       z3=[]; f=[]; x=[];
       for i = 1:6^m
          Check2 = Chkf(fs(i,:),F);
          if Check2
             z1 = zval(xwp,fs(i,:));
             z = [c s.*fs(i,:)];
             [xw,z2] = linprog(z,A,-b,[],[],lb,[],[],opt);
             z3 = [z3 z1-z2];
             f = [f; fs(i,:)];
             x = [x xw];
          end
       end
       if length(z3)!=0
          [Z(p),k] = max(z3);
          F(p+1,:) = f(k,:);
          X(:,p+1) = x(:,k);
       end
    end
% -------------------- Check Unbound --------------------
    if abs([Z(p),R(p)])>10^7
       break;
    end
% ------------------- Check Z(p)>R(p) -------------------
    if Z(p)>R(p) && p<=50
          p = p+1;
          Check = true;
```

```
    else  Check = false;
    end
end
T = toc;
R = R(end);
```

The following function is the subroutine for finding the probabilities providing largest expected values for all ordering cases that used in Algorithm 2.

```
function h = ProbProvMax(m,n,dx,f,d)
h = []; ord = [];
dif = dx(n+1:end-1);
for i = 1:m
    k = (1:3)+3*(i-1);
    ord = [ord Ordering(dif(k))];
end
for i = 1:6^m
    if ord == d(i,:)
        h = f(i,:);
    end
end
```

APPENDIX E: Algorithm 3

```
function [T R xwp] = Algorithm3(m,n,a,c,s,b,fs,fl)
opt = optimset('Display','off');
tic
N = n+3*m;                    % #Unknown variables x,w
s = Expand(m,s); X=[];
fs = Expand(m,fs);
zval = @(xw,f) c*xw(1:n)+s.*f*xw(n+1:end-1);
% Step 1 ----------------------------------------------
F = fs(1,:); M = fs(1,:);
z = [c s.*F(1,:)];
A = [-a -eye(3*m)];
lb = zeros(1,N);
xwb = linprog(z,A,-b,[],[],lb,[],[],opt);
X = [X xwb];
p = 1;                        % Number of iterations
% Step 2 ----------------------------------------------
Check1 = true;
while Check1
% --------------------- Find R(p) ---------------------
   z = [zeros(1,N) 1];
   A = [-a -eye(3*m) zeros(3*m,1)];
   B = -b;
   for i = 1:p
      A = [A; c s.*F(i,:) -1];
      q = c*X(1:n,i)+s.*F(i,:)*X(n+1:end,i);
      B = [B q];
   end
   lb = zeros(1,N+1);
   [xwp,R(p)] = linprog(z,A,B,[],[],lb,[],[],opt);
% --------------------- Find Z(p) ---------------------
   z = [c s.*F(p,:) -zval(xwp,F(p,:))];
   A = [-a -eye(3*m) zeros(3*m,1)];
   Aeq = [zeros(1,N) 1]; beq = 1;
   xwb = linprog(z,A,-b,Aeq,beq,lb,[],[],opt);
   [h,I1] = ModifiedPPM(m,n,xwp-xwb,fl);
   z = [c s.*h -zval(xwp,h)];
   [xwb,val] = linprog(z,A,-b,Aeq,beq,lb,[],[],opt);
```

```
    Z(p) = -val;
    F = [F; h];
    X = [X xwb(1:end-1)];
% ----------------- Check Z(p)<=R(p) ------------------
    if Z(p)<=R(p)
        I2 = ChkOrd(m,n,xwp-xwb);
        if isequal(I1,I2)
            break;
        else
            f=[]; z2=[]; x=[];
            for i = 1:6^m
                Check2 = Chkf(fs(i,:),F);
                if Check2
                    z = [c s.*fs(i,:) -zval(xwp,fs(i,:))];
                    [xw,z1] = linprog(z,A,-b,Aeq,1,lb,[],[],
                        opt);
                    Z(p) = -z1;
                    z2 = [z2 Z(p)];
                    x = [x xw(1:end-1)];
                    f = [f; fs(i,:)];
                    if Z(p)>R(p)
                        F(p+1,:) = fs(i,:);
                        X(:,p+1) = xw(1:end-1);
                        break;
                    elseif !Chkf(fs(i,:),F)
                        Check1 = false;
                        break;
                    end
                end
            end
        end
    end
% ------------------- Check Unbound -------------------
    if abs([Z(p),R(p)])>10^7
        break;
    end
% ------------------ Check Z(p)>R(p) ------------------
    if Z(p)>R(p) && p<=50
            p = p+1;
            Check1 = true;
```

```
    else  Check1 = false;
    end
end
T = toc;
R = R(end);
```

The following functions are the subroutines used in Algorithm 3 which including; modified probability providing largest expected values for all ordering cases (ModifiedPPM), construct the ordering (ChkOrd) and check that a probability $f^{(i)}$ is used in step 2 (Chkf), respectively, as the following.

```
function [h,I] = ModifiedPPM(m,n,dx,fl)
p = perms([3 2 1]);
d = [p(:,3) p(:,2) p(:,1)];
dif = dx(n+1:end-1);
h = []; I = [];
for i = 1:m
    k = (1:3)+3*(i-1);
    ord = Ordering(dif(k));
    for j = 1:6
        if ord == d(j,:)
            h = [h fl(j,k)];
            I = [I p(j,:)];
            break;
        end
    end
end
```

```
function I = ChkOrd(m,n,dx)
I = [];
dif = dx(n+1:end-1);
for i = 1:m
    k = (1:3)+3*(i-1);
    I = [I Ordering(dif(k))];
end
```

```
function Check = Chkf(f,F)
Check = true;
for i = 1:size(F,1)
   if isequal(f,F(i,:))
      Check = false;
      break;
   end
end
```

APPENDIX F: Regret average time for each case up to five decision variables and up to five uncertain constraints

Since we random 100 examples for each case of variables and constraints, we present here only 10 of these 100 examples to show that the regrets got from three algorithms are the same.

1 variable	Regret (R)										Average time
1 constraint	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	751.2027	0	0	466.3738	0	9.4757	0	122.5838	440.8971	0	0.1711
Algorithm 2	751.2027	0	0	466.3738	0	9.4757	0	122.5838	440.8971	0	0.0700
Algorithm 3	751.2027	0	0	466.3738	0	9.4757	0	122.5838	440.8971	0	0.0585

Table 1: The regret and average time for one decision variable and one uncertain constraint

1 variables	Regret $(R) * (1.0e + 03)$										Average time
2 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.2723	0.0160	0	0	0.0870	0.0406	0.0143	0.0003	0.9967	1.2379	0.3919
Algorithm 2	0.2723	0.0160	0	0	0.0870	0.0406	0.0143	0.0003	0.9967	1.2379	0.2248
Algorithm 3	0.2723	0.0160	0	0	0.0870	0.0406	0.0143	0.0003	0.9967	1.2379	0.1957

Table 2: The regret and average time for one decision variable and two uncertain constraints

1 variables	Regret $(R) * (1.0e + 03)$										Average time
3 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.0147	0.0883	0	0	1.9846	0.4918	0.5839	2.3727	0.2203	0	3.3691
Algorithm 2	1.0147	0.0883	0	0	1.9846	0.4918	0.5839	2.3727	0.2203	0	1.5061
Algorithm 3	1.0147	0.0883	0	0	1.9846	0.4918	0.5839	2.3727	0.2203	0	0.9731

Table 3: The regret and average time for one decision variable and three uncertain constraints

1 variables	Regret $(R) * (1.0e + 03)$										Average time
4 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.8145	1.4377	0.5915	3.0158	1.6776	0	0.0279	0.0984	0.2366	0.6779	24.9854
Algorithm 2	1.8145	1.4377	0.5915	3.0158	1.6776	0	0.0279	0.0984	0.2366	0.6779	9.2054
Algorithm 3	1.8145	1.4377	0.5915	3.0158	1.6776	0	0.0279	0.0984	0.2366	0.6779	4.8609

Table 4: The regret and average time for one decision variable and four uncertain constraints

1 variables	Regret $(R) * (1.0e + 03)$										Average time
5 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.6511	1.7951	1.9166	0.9353	0.8797	0	0	0	1.8718	1.2700	166.9415
Algorithm 2	1.6511	1.7951	1.9166	0.9353	0.8797	0	0	0	1.8718	1.2700	65.9210
Algorithm 3	1.6511	1.7951	1.9166	0.9353	0.8797	0	0	0	1.8718	1.2700	37.7127

Table 5: The regret and average time for one decision variable and five uncertain constraints

2 variables	Regret $(R) * (1.0e + 03)$										Average time
1 constraint	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	280.8024	0	164.8673	0	0	630.0676	28.9899	0	252.8209	0.0515
Algorithm 2	0	280.8024	0	164.8673	0	0	630.0676	28.9899	0	252.8209	0.0534
Algorithm 3	0	280.8024	0	164.8673	0	0	630.0676	28.9899	0	252.8209	0.0507

Table 6: The regret and average time for two decision variables and one uncertain constraint

2 variables	Regret $(R) * (1.0e + 03)$										Average time
2 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	104.4580	16.6488	72.6255	588.3791	0	36.3101	168.0688	689.0189	0	0.4867
Algorithm 2	0	104.4580	16.6488	72.6255	588.3791	0	36.3101	168.0688	689.0189	0	0.3502
Algorithm 3	0	104.4580	16.6488	72.6255	588.3791	0	36.3101	168.0688	689.0189	0	0.1953

Table 7: The regret and average time for two decision variables and two uncertain constraints

2 variables	Regret $(R) * (1.0e + 03)$										Average time
3 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.9788	2.2941	0.2016	1.5303	0.3956	0.2327	1.0540	0.7086	0.0000	0.5720	2.8889
Algorithm 2	0.9788	2.2941	0.2016	1.5303	0.3956	0.2327	1.0540	0.7086	0.0000	0.5720	1.7999
Algorithm 3	0.9788	2.2941	0.2016	1.5303	0.3956	0.2327	1.0540	0.7086	0.0000	0.5720	1.0525

Table 8: The regret and average time for two decision variables and three uncertain constraints

2 variables	Regret $(R) * (1.0e+03)$										Average time
4 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.4940	2.6464	4.2096	0.1269	1.2598	0.9280	1.0749	1.5939	0.8351	1.5715	20.2407
Algorithm 2	1.4940	2.6464	4.2096	0.1269	1.2598	0.9280	1.0749	1.5939	0.8351	1.5715	15.4527
Algorithm 3	1.4940	2.6464	4.2096	0.1269	1.2598	0.9280	1.0749	1.5939	0.8351	1.5715	6.1747

Table 9: The regret and average time for two decision variables and four uncertain constraints

2 variables	Regret $(R) * (1.0e+03)$										Average time
5 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.5492	1.3780	0.9573	1.7370	1.4260	1.9608	2.9881	1.0010	1.6997	0.8628	106.8274
Algorithm 2	0.5492	1.3780	0.9573	1.7370	1.4260	1.9608	2.9881	1.0010	1.6997	0.8628	96.9212
Algorithm 3	0.5492	1.3780	0.9573	1.7370	1.4260	1.9608	2.9881	1.0010	1.6997	0.8628	48.3515

Table 10: The regret and average time for two decision variables and five uncertain constraints

3 variables	Regret $(R) * (1.0e+03)$										Average time
1 constraint	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	0.0004	0.0001	0.6447	0.0001	0.0017	0	0	0	0.0009	0.0906
Algorithm 2	0	0.0004	0.0001	0.6447	0.0001	0.0017	0	0	0	0.0009	0.0755
Algorithm 3	0	0.0004	0.0001	0.6447	0.0001	0.0017	0	0	0	0.0009	0.0600

Table 11: The regret and average time for three decision variables and one uncertain constraint

3 variables	Regret $(R) * (1.0e+03)$										Average time
2 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	0.9661	0	1.5438	0.7393	0	1.8529	0.1326	2.0811	0.2230	0.4333
Algorithm 2	0	0.9661	0	1.5438	0.7393	0	1.8529	0.1326	2.0811	0.2230	0.3074
Algorithm 3	0	0.9661	0	1.5438	0.7393	0	1.8529	0.1326	2.0811	0.2230	0.2387

Table 12: The regret and average time for three decision variables and two uncertain constraints

3 variables	Regret $(R) * (1.0e+03)$										Average time
3 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.8541	2.5989	4.8891	0.3205	0.6902	1.5040	0.4697	0.0125	0.8515	2.1872	2.6587
Algorithm 2	0.8541	2.5989	4.8891	0.3205	0.6902	1.5040	0.4697	0.0125	0.8515	2.1872	2.0604
Algorithm 3	0.8541	2.5989	4.8891	0.3205	0.6902	1.5040	0.4697	0.0125	0.8515	2.1872	1.2570

Table 13: The regret and average time for three decision variables and three uncertain constraints

3 variables	Regret $(R) * (1.0e+03)$										Average time
4 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.8029	1.2349	0.9004	0.0139	1.2559	0.4011	1.4114	0.8593	0.8528	1.5798	23.4222
Algorithm 2	0.8029	1.2349	0.9004	0.0139	1.2559	0.4011	1.4114	0.8593	0.8528	1.5798	13.0668
Algorithm 3	0.8029	1.2349	0.9004	0.0139	1.2559	0.4011	1.4114	0.8593	0.8528	1.5798	7.8081

Table 14: The regret and average time for three decision variables and four uncertain constraints

3 variables	Regret $(R) * (1.0e + 03)$										Average time
5 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.5376	2.7453	2.7432	1.2381	1.3521	1.8797	1.2449	2.4090	2.1467	1.4328	128.7291
Algorithm 2	1.5376	2.7453	2.7432	1.2381	1.3521	1.8797	1.2449	2.4090	2.1467	1.4328	91.6097
Algorithm 3	1.5376	2.7453	2.7432	1.2381	1.3521	1.8797	1.2449	2.4090	2.1467	1.4328	52.0429

Table 15: The regret and average time for three decision variables and five uncertain constraints

4 variables	Regret $(R) * (1.0e + 03)$										Average time
1 constraint	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.0002	0.0002	0.0105	1.1662	0	0.0064	0.0064	0.0133	0	0.0035	0.1069
Algorithm 2	0.0002	0.0002	0.0105	1.1662	0	0.0064	0.0064	0.0133	0	0.0035	0.0713
Algorithm 3	0.0002	0.0002	0.0105	1.1662	0	0.0064	0.0064	0.0133	0	0.0035	0.0.0655

Table 16: The regret and average time for four decision variables and one uncertain constraint

4 variables	Regret $(R) * (1.0e + 03)$										Average time
2 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	1.6690	0	0	0.4632	0.8693	1.2886	0.1433	0	0.1474	0.4218
Algorithm 2	0	1.6690	0	0	0.4632	0.8693	1.2886	0.1433	0	0.1474	0.2563
Algorithm 3	0	1.6690	0	0	0.4632	0.8693	1.2886	0.1433	0	0.1474	0.1966

Table 17: The regret and average time for four decision variables and two uncertain constraints

4 variables	Regret $(R) * (1.0e+03)$										Average time
3 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.310	0.2483	4.7501	0.6437	3.6403	0.6307	2.9055	0.7824	0.5378	1.8325	3.3827
Algorithm 2	0.310	0.2483	4.7501	0.6437	3.6403	0.6307	2.9055	0.7824	0.5378	1.8325	2.3472
Algorithm 3	0.310	0.2483	4.7501	0.6437	3.6403	0.6307	2.9055	0.7824	0.5378	1.8325	1.3934

Table 18: The regret and average time for four decision variables and three uncertain constraints

4 variables	Regret $(R) * (1.0e+03)$										Average time
4 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.0838	5.4309	0.0110	1.5226	0	0.5008	1.2725	0.5698	1.7556	3.2432	18.5621
Algorithm 2	0.0838	5.4309	0.0110	1.5226	0	0.5008	1.2725	0.5698	1.7556	3.2432	15.1171
Algorithm 3	0.0838	5.4309	0.0110	1.5226	0	0.5008	1.2725	0.5698	1.7556	3.2432	10.5068

Table 19: The regret and average time for four decision variables and four uncertain constraints

4 variables	Regret $(R) * (1.0e+03)$										Average time
5 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.8819	0.5880	0.7826	0.8497	1.5189	2.5162	0.1368	1.4879	0.2725	2.0057	138.7532
Algorithm 2	0.8819	0.5880	0.7826	0.8497	1.5189	2.5162	0.1368	1.4879	0.2725	2.0057	112.7372
Algorithm 3	0.8819	0.5880	0.7826	0.8497	1.5189	2.5162	0.1368	1.4879	0.2725	2.0057	54.9573

Table 20: The regret and average time for four decision variables and five uncertain constraints

5 variables	Regret $(R) * (1.0e + 03)$										Average time
1 constraint	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	2.7679	0	0.0046	0	0	0.3373	0	0.2031	0	0.0811
Algorithm 2	0	2.7679	0	0.0046	0	0	0.3373	0	0.2031	0	0.0646
Algorithm 3	0	2.7679	0	0.0046	0	0	0.3373	0	0.2031	0	0.0589

Table 21: The regret and average time for five decision variables and one uncertain constraint

5 variables	Regret $(R) * (1.0e + 03)$										Average time
2 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0	1.0264	0	0	0.0254	0	0.0237	1.1546	0.1918	0	0.4398
Algorithm 2	0	1.0264	0	0	0.0254	0	0.0237	1.1546	0.1918	0	0.2742
Algorithm 3	0	1.0264	0	0	0.0254	0	0.0237	1.1546	0.1918	0	0.2043

Table 22: The regret and average time for five decision variables and two uncertain constraints

5 variables	Regret $(R) * (1.0e + 03)$										Average time
3 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.2504	2.7764	0.0621	1.5114	1.2619	2.4972	0.0871	0.0218	0.4235	2.1828	2.7904
Algorithm 2	0.2504	2.7764	0.0621	1.5114	1.2619	2.4972	0.0871	0.0218	0.4235	2.1828	2.1166
Algorithm 3	0.2504	2.7764	0.0621	1.5114	1.2619	2.4972	0.0871	0.0218	0.4235	2.1828	1.3776

Table 23: The regret and average time for five decision variables and three uncertain constraints

5 variables	Regret $(R) * (1.0e+03)$										Average time
4 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	1.0707	1.5042	0.7458	0.7975	1.2330	1.4603	5.4942	3.8356	2.2596	1.5996	22.2362
Algorithm 2	1.0707	1.5042	0.7458	0.7975	1.2330	1.4603	5.4942	3.8356	2.2596	1.5996	16.7340
Algorithm 3	1.0707	1.5042	0.7458	0.7975	1.2330	1.4603	5.4942	3.8356	2.2596	1.5996	9.7868

Table 24: The regret and average time for five decision variables and four uncertain constraints

5 variables	Regret $(R) * (1.0e+03)$										Average time
5 constraints	Ex1	Ex2	Ex3	Ex4	Ex5	Ex6	Ex7	Ex8	Ex9	Ex10	
Algorithm 1	0.7084	0.6301	3.3242	0.8987	3.1591	1.4350	2.2404	2.7510	1.8445	4.8095	181.4090
Algorithm 2	0.7084	0.6301	3.3242	0.8987	3.1591	1.4350	2.2404	2.7510	1.8445	4.8095	119.8803
Algorithm 3	0.7084	0.6301	3.3242	0.8987	3.1591	1.4350	2.2404	2.7510	1.8445	4.8095	75.2131

Table 25: The regret and average time for five decision variables and five uncertain constraints

Biography

Thibhadha Saraprang
Bachelor's degree of Mathematics, 2014,
Department of Mathematics and Computer Science,
Faculty of Science,
Chulalongkorn University