ulm university universität
uulm

Dissertation
zur Erlangung des Doktorgrades Dr. rer. nat.
der Fakultät für Ingenieurwissenschaften und Informatik
der Universität Ulm

CDF-Intervals: A Probabilistic Interval Constraint Framework to Reason about Data with Uncertainty

---

Aya Hassan Saad

Universität Ulm
Fakultät für Ingenieurwissenschaften und Informatik
Institut für Programmiermethodik und Compilerbau
Oktober 2015

Amtierende Dekanin:	Prof. Dr. Tina Seufert
1. Gutachter:	Prof. Dr. Thom Frühwirth (Universität Ulm, Germany)
2. Gutachter:	Prof. Dr. Carmen Gervet (Université de Montpellier, France)
Tag der Promotion:	21. April 2016

ABSTRACT

We propose a novel framework, the cummulative distribution function (*cdf*)-intervals, that intuitively describes data coupled with uncertainty without losing any information given in the problem definition. Our new proposition brings about a construction of two algebraic convex structures: the *cdf*-intervals and the Probability Box (p-box) *cdf*-intervals. The two proposed structures were driven by the practical usage of reliable approaches in Constraint Programming (CP) and Operation Research (OR) paradigms. These approaches tackle large scale constraint optimization (LSCO) problems associated with data uncertainty in a tractable manner. The key idea is to bound data observed in the problem definition, then perform the computation only on the bounds using interval reasoning techniques. Output solution set from this process satisfies all possible realization of the data sought. Approaches following the convex modeling commonly treat data in their interval representation with an equal weight, thus they do not reflect any possible degree of knowledge about the whereabouts.

Motivated by bringing more knowledge to the realized solution set, we introduced the *cdf*-intervals in [Saad, Gervet, and Abdennadher (2010)]. The bounding points, in the *cdf*-intervals algebraic structure, each is specified by two values: data and its cumulative distribution function (*cdf*) [Saad et al. (2010)]. This new structure attempts to represent data in a 2 dimensions (2D) manner, yet the probability distribution (the 2^{nd} dimension) is an approximated representation of the actual distribution. We further extended the *cdf*-intervals, with the notion of p-box, in order to enclose all available information by two *cdf* distributions [Saad, Gervet, and Fruehwirth (2012b),Saad, Gervet, and Fruehwirth (2012a),Saad, Frühwirth, and Gervet (2014) and Saad (2014)]. The bounding distributions are chosen to be uniform in order to ease the computations over the novel algebraic structure. The probabilities, within those bounds, are ranked based on the stochastic dominance. In this work, we define the formal frameworks for constraint reasoning over the *cdf*-intervals and the p-box *cdf*-intervals. The modeling and reasoning are constructed within the CP paradigm due to its powerful expressiveness. Moreover, we construct a system of global constraints, over the two algebraic structures, by extending Interval Linear Systems (ILS) with a second dimension (the *cdf*). We de-

velop a formal Constraint Logic Programming (CLP) language from the new defined domains and show how the new domains affect the problem variables and the decision process. We implement the new language as a separate solver module in the ECL^{i}PS^e constraint programming environment.

The p-box *cdf*-intervals combine techniques from the convex modeling, to take advantage of their tractability, with approaches revealing quatifiable information from the probabilistc and stochastic world, to take advantage of their expressiveness. We perform a comparison in the data representation and in the reasoning performance over models from the two paradigms and our novel framework. This comparison is further adopted to model two different real-life applications: the Network Traffic Application problem, used in network design problems, and the Inventory Management problem of a manufacturing process. The empirical evaluation of our implementation shows that, with minimal overhead, the output solution set realizes a full enclosure of the data along with tighter bounds on its probabilistic distributions. Solutions sought to be feasible in the real domain are excluded by the p-box *cdf*-intervals reasoning since they are infeasible because they violate the properties of the *cdf*-domain. Additional knowledge, on the data whereabouts, gained by the implementation of our novel formal and practical framework gives rise to a wide range of future research work.

LIST OF TABLES

LIST OF FIGURES

List of Symbols

$F_X(x)$ cummulative distribution function of X. 11

F_x^I *cdf* distribution of an interval variable. 58–60

F^p *cdf* distribution of a point p. 58

F_x^p *cdf* value of a point p having a quantile value equal to x. 62, 189

$f(x)$ density function. 11

$F_{XY}(x,y)$ joint cummulative distribution function. 12

$f_{XY}(z)$ joint density function. 12, 194, 198, 200

glb greatest lower bound. vi, 61, 63, 65, 74, 76, 86, 103, 104, 165, 202, 204

lub least upper bound. vi, 61, 63, 65, 74, 76, 86, 103, 104, 165, 202, 204

p_a point p having a quantile value equal to a a. 58

$\mathbb{R}$ domain of reals. 9, 10, 14, 48, 53, 59, 61–63, 65–67, 69, 70, 72, 74, 163, 164, 192, 194, 197, 201, 214

$R_{\mathcal{U}}$ *cdf* domain of ranges. 75, 85

σ standard deviation. 55, 58

$\Sigma_{\mathcal{U}}$ the signature of the *cdf*-interval formal language. 62, 73

S_x^p *cdf* distribution slope of a point p having a quantile value equal to x. 59, 62, 189

$\mathcal{U}$ domain of discourse when the uncertainty is defined over two dimensional space. 62–66, 73–75

$\mathcal{U}_1$ *cdf*-intervals domain of discourse. vi, 64, 65

$\mathcal{U}_b$ p-box *cdf*-intervals domain of discourse. vi, 64

List of Acronyms

PDF probability distribution function. 10, 11

cdf cummulative distribution function. i, ii, v–vii, xiii, xiv, xvi, 1, 4–6, 11–15, 41, 46–48, 51, 52, 54–83, 86–88, 90–104, 110–113, 118–120, 137–152, 156, 161–167, 169–173, 189–191, 194–214

pdf probability density function. xvi, 10–12, 52, 54, 56, 191, 193–195, 199

2D 2 dimensions. i, vii, 5, 46–48, 61–63, 65, 67–70, 72–74, 77, 78, 80, 83, 103, 113, 118, 161, 163–165

BGP Border Gateway Protocol. 108

CLP Constraint Logic Programming. ii, 5, 165

CP Constraint Programming. i, iii, xv, 1–5, 17, 20, 22, 25, 34, 38–41, 43, 97, 128, 129, 135, 137, 138, 162, 167, 172

CSP Constraint Satisfaction Problem. iii, v, 2, 20, 21, 25–29, 34, 38, 39, 43, 99, 130, 166, 171, 172

DP Dynamic Programming. 126, 128, 129, 131, 140

ELSP Economic Lot Scheduling Problem. 122

EOQ Economic Order Quantity. iii, 122, 124–127, 137–140, 142–146, 148–151, 156

EUCSPT Extended Uncertain CSP Transformation. vi, 98

ILP Interval Linear Programming. 39

ILS Interval Linear Systems. i, 97–100, 103

IP Integer Programming. 23

IS-IS Intermediate System - Intermediate System. 108

LP Linear Programming. 2, 3, 5, 17, 20, 22, 23, 29, 30, 34–36, 39–41, 97, 98, 103, 104, 135, 172

LSCO large scale constraint optimization. i, 1

MIP Mixed Integer Programming. 23, 108, 138, 140, 142, 146–152, 154–156

netflow network flow. 110, 112, 113

NTAP Network traffic flow analysis problem. xvi, 5, 43, 108, 110, 112, 113, 120, 166, 169, 173

OD Origin and Destination. iii, 107–109, 111, 114, 115, 118

OR Operation Research. i, 1–3, 17, 20, 43, 162

OSPF Open Shortest Path First. 108

p-box Probability Box. i, ii, v–vii, xiv, xvi, 4, 5, 14, 15, 47, 48, 51, 52, 55–60, 62–64, 66–83, 87, 88, 90–92, 94, 97, 99–104, 110–113, 118–120, 137–141, 143–150, 152, 156, 161–167, 169–173, 201, 202, 204, 206–214

POLI Positive Orthant Linear Interval. 98

POP Network Point of Presence. 108

poset partially ordered set. 4, 61–64, 165

SCP Stochastic Constraint Programming. xv, 128–130, 140

SCSP Stochastic Constraint Satisfaction Problem. 129, 140, 142, 146–152, 154–156, 171, 172

SNMP Simple Network Management Protocol. 107, 108

TM Network Traffic Matrix. 107

UCSP Uncertain Constraint Satisfaction Problem. xv, 2, 3, 5, 45, 98, 102–104, 111–113, 119, 120, 135, 140–142, 145–148, 150–152, 154–156, 164, 166

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Constraint domains which tackle data uncertainty have received little attention in the Constraint Programming (CP) and in the Operation Research (OR) paradigms. In this research, we introduce a novel framework: the cummulative distribution function (*cdf*)-intervals. This novel framework intuitively encapsulates any probability distribution derived from past measurements or future forecasts into an interval data. It is a multi-paradigm contribution that combines the powerful expressivity of the CP techniques, with the inexpensive computational nature of reliable models, to allow for uncertainty reasoning that is more tractable.

1.1 Problem Definition

Real-world applications, such as in planning, scheduling, diagnosis, or tracking problems, are modeled by large scale constraint optimization (LSCO) problems. They are unavoidably coupled with uncertainty due to unpredictable internal as well as external environmental aspects. Data uncertainty can be found in, but not limited to:

1. Planning: the placement of the renewable energy parks depends on the varying demands for energy and the construction cost of the parks, Gervet and Atef (2013).

2. Scheduling: the inventory management problem, in a manufacturing process, rely on the observed fluctuating information of customer demands, setup costs and item prices, Tarim and Kingsman (2004)

3. Diagnosis: the network traffic management and engineering processes count on the collected traffic flow information. The collection process is often operated in a distributed manner. Accordingly, the measurement can be ill-defined due to packet loss or deviation from designated paths, Grossglauser and Rexford (2005).

4. Tracking: the detailed identification of objects belonging to an image differs in significance in a real-time image recognition problem. It is based on the different interpretation of various color shades and their motions, Deruyver and Hodé (2009).

This uncertainty affects not only the behavior of the problem in hand but also its acquired optimal solution. It is necessary to quantify existing uncertainty in the problem under consideration in order to acquire reliable solutions. Uncertain data in turn can be found: incomplete or following a probabilistic distribution. So far research effort towards formalizing and solving large scale problems coupled with uncertainty factors is quite few. The CP paradigm proved to have a considerable flexibility in formulating optimization problems, whilst Linear Programming (LP) provided a better realization of optimal solutions for scalable problems. Techniques for hybridization have even more incorporated advantages that exist in both paradigms. The CP paradigm, by means of Constraint Satisfaction Problem (CSP) definition, aims at building framework for fixed point semantics which results in narrowing the possible domains of solution. The most recognized CP frameworks which handle uncertainty are listed below:

- Mixed CSPs, Fargier, Lang, and Schiex (1996), seek for one robust solution which satisfies as many realizations as possible. This framework is equivalent to a one stage stochastic CSP in the discrete form
- Interval CSPs, Benhamou and Older (1997), provide a propagation technique for uncertainty represented by real interval domains
- Branching CSPs, Fowler and Brown (2003), handle problems characterized by having variables revealed by time.
- Partial CSPs (valued CSPs and semirings, Bistarelli et al. (1999)) attach an uncertainty value or in other words a degree of preference to constraints. Partial CSP frameworks are adopted explicitly to deal with soft constraints.
- Certainty closure (Uncertain Constraint Satisfaction Problem (UCSP)), Yorke-Smith and Gervet (2009), associates uncertainty to constraint coefficients; thereof, the output solution is characterized to be reliable and can take any of the possible values from the closure of the decision space.
- Stochastic CSPs, Tarim, Manandhar, and Walsh (2006), exhaustively build scenarios based on the input probability distribution; such structure reacts to values suggested by stochastic variables in different manners. Each scenario in turn is considered as a classical CSP on its own.

On the other hand, OR paradigm, found in reliable and robust computation, quest for approximating large scale problems so that they become solvable using LP techniques. Without this approximation LP that inherently lacks flexibility in problem representation

would not be sufficient to handle the uncertainty component in large scale optimization problems. The most well-known OR frameworks can be listed as follows:

- Interval LP, Beaumont (1998), allows for expanding the solution possibilities; hence, an uncertain value is described by bounded intervals instead of a singleton value. Interval linear computations are easily conducted but they do not guarantee the tightest bounds on the resulting solution set.

- Robust optimization, Ben-Tal and Nemirovski (2000), approximates the problem, often by using the variance and expected values, such that all possible scenarios for uncertainty are subsumed in a convex ellipsoidal set. Such representation encapsulates ambiguity provided by erroneous measurements; despite the existence of uncertainty, it makes all possible solution set available.

- Stochastic LP, Sen and Higle (1999), thoroughly formulates a tree that exhibit possible set of actions for a given problem. Each branch in the tree is associated with an expected probable value; therefore unrealistic solutions might be unnecessarily explored.

All of the above mentioned alternatives in: CP and OR, generally, work as follows: some of the models attach a point-wise probability to the values in the discrete domain; some others adopt an approximated known form of probability distribution. Arranged approximations, of the known distribution, are based on the calculation of the variance and the expected values. Solutions are commonly derived from the maximization of the expected value or they are obtained in a set interval. Realizations acquired may not provide the accurate feasible solution with respect to the actual problem since in real-life situations the probability distribution pursues unfamiliar shapes.

1.2 Motivation

Solution reliability and robustness is an important aspect of optimization that is not sufficiently reported in CP and OR, especially for those applications which are tightly coupled with data uncertainty. Our work was driven by the practical usage of reliable approaches in CP which are computationally tractable. These approaches can be found in the UCSP framework in [Yorke-Smith (2004)] to tackle network design problems, and more recently, in the renewable energy park placement problems with uncertain demand and costs. When adopted in real life situations, however, the UCSP lacks the expressivity of the information on the data whereabouts, which is already given in the problem definition. There is a need to extend classical interval coefficient models to account for any potential data distribution available. This dissertation shows how this can be achieved with little overhead, while enriching the solution sets produced with an encapsulation on the data probability distribution.

1.3 Contribution

Indeed our goal is to add expressiveness to the solution sets of reliable models while preserving tractability. The choice of linear enclosure of the data distribution ensures both. The main contribution of this work concerns the formal definition of a new interval arithmetic and its implementation. We show that bounding a random variable distribution, with two tight linear *cdf* distributions, is a safe enclosure at minimal overhead. We also show how intuitively the *cdf*-intervals framework describes values coupled with uncertainty, using native CP techniques to propagate more information to the system of constraints and the variable closure of the solution set.

In the novel framework, uncertainty of data defined in the problem is represented as Probability Box (p-box) *cdf*-interval coefficients which are input to the solver. Solution sets acquire additional quantitative information which adds knowledge of the data whereabouts.

1.3.1 Challenges

- Extend reliable models with quantitative information
- Preserve tractability
- Bounding ill-defined/ uncertain data whereabouts without inferring any assumptions on their distribution
- Reason with *cdf*-intervals by defining variable domains and constraint inference rules.

1.3.2 Mission

Our work aims at addressing the need to reason about data coupled with uncertainty from a language viewpoint. We required a domain ordering, monotonic property, dominance over the *cdf* and p-box notions. Such concepts and definitions are quite new to the CP paradigm. We also show how conventional reliable computing methods can be extended effectively to account for bounded distributions.

We start by defining a new domain for reasoning with uncertain data. The key idea is to combine the usual interval arithmetic approach with a second dimension capturing the *cdf* of the variable whose primary dimension (an interval of quantiles in the real domain) spans the value that the uncertain variable can take. This work introduces this new domain, why it forms a partially ordered set (poset), and how to define the conventional arithmetic operations and their computations on this new domain. The entire exercise makes it possible to define constraints over variables on this domain, where the solution method deliver intervals for variables alongside the relevant fragment of the *cdf*.

The fundamental algebraic structure in our framework is the *cdf*-interval. The main idea behind this work is to express the data by an interval that includes all the information

along with its uncertainty. It is the interval which encloses a set of *cdf*-points defined on 2 dimensions (2D). The *cdf*-Intervals framework is employed to reason about data with uncertainty inexpensively in a Constraint Logic Programming (CLP) system. Reasoning operations are exerted on the convex structure extreme points. They facilitate the shrinking of the interval in order to obtain a solution set, while maintaining the interval properties. Result of these operations, i.e. the solution set, is also an interval enclosed by two bounds.

CP together with LP techniques are adopted to formalize the full system of constraints defined by the *cdf*-intervals. Using the *cdf*-intervals we seek the propagation of values along with their probability distributions to the system of constraints. Due to the new defined propagation algebra more information is introduced to the closure, while values along with their probability of existence are realized. Such realizations could be useful for a better allocation of resources when a solution value and its probability is produced.

1.4 Thesis Organization

The remainder of this dissertation consists of three parts detailing our contributions: where to apply the *cdf* framework, how to construct its algebraic structure and how it works. We also show examples of real life applications to elaborate how to model problems coupled with data uncertainty using the *cdf*-interval framework.

Part I. Context We start by describing the context: the basic concepts forming the pillars which construct the *cdf*-intervals; types for data uncertainty which exist in the real-world applications; and the literature review which diagnoses the different paradigms which tackle data with uncertainty. We end part I with a comparison between the constructions of input data in different convex models: UCSP, *cdf*-intervals and p-box *cdf*-intervals.

Part II. Framework we show how data given in the problem definition is represented in the *cdf*-intervals. We show how to extract the confidence intervals and we compare this presentation with existing approaches. We elaborate the theoretical construction of *cdf*-intervals, and we show how to build them from a language viewpoint. This, in fact, paves the way to shape the language structure, its new domain calculus, and the arithmetic operations which are computed in a $2D$ manner. As a consequence, we implement the novel algebraic structure inference rules, the practical framework and the solver. To complete the full constraint system, we develop a system of global constraints which adopts *cdf*-intervals to express and execute linear systems of constraints.

Part III. Applications to support the proof of concept, we tackle two different real-life applications: the Network traffic flow analysis problem (NTAP) and the inventory management problem. We show in this part how to utilize the *cdf*-intervals framework

in order to easily express the problem. We also compare the *cdf*-intervals framework obtained solution sets, their significance, and the system performance with various existing approaches and techniques commonly employed in such applications. Finally, we support our argument by comparing our framework with existing techniques in terms of expressiveness to model the original problems, additional significance gained, solutions obtained, and system behavior and performance. The three parts of this dissertation are concluded by a discussion section which summarizes the main points elaborated and potential work to be considered in the future. The appendix demonstrates the proofs which form the theoretical framework of the *cdf*-intervals and includes parts of the *cdf*-intervals solver implementation.

PART I

CONTEXT

CHAPTER 2

BASIC CONCEPTS

This chapter recalls fundamental concepts we use to characterize the basic features of our formal framework. These definitions can be found in Stark and Woods (1994); Williamson and Downs (1990); Gubner (2006); Berleant (1993); Glen, Leemis, and Drew (2004). Readers familiar with those concepts can skip this chapter.

2.1 From the Probability Theory

2.1.1 Random Variable

A random variable $X : \Omega \to \mathbb{R}$ defines a mapping from an original sample space Ω to the real line $\mathbb{R}$. In other words, a random variable X is a function whose domain is Ω and whose range is some subset of the real line $\mathbb{R}$.

Property 2.1 (discrete random variable). *assumes values from a finite domain*

A typical example of a discrete random variable can be found in an experiment of tossing a die. The output can be any of the values, given in a list of finite domain, $\{1, 2, 3, 4, 5, 6\}$.

Property 2.2 (continuous random variable). *maps the set of possible values over a continum.*

For instance, a value, that is given from a measurement process of, but not limited to, temperature, height and rainfall, cannot be precise. Random variables involved in these processes are continuous, because they can always obtain infinitesimally more precise values.

Random variables are denoted by capital letters X,Y,Z and values they map are denoted by lower-case letters x,y,z.

Property 2.3 (probability space ($\mathbb{R}, \mathscr{B}, P_x$)).

- $\mathbb{R}$ *is the real line.*
- $\mathscr{B}$ *is the Borel* σ*-algebra of all subsets of* $\mathbb{R}$ *generated by countable unions and intersections of sets with the form* $(-\infty, x]$.
- P_x *is the set function assigning a number* $P_x[A] \geq 0$ *to each set* $A \in \mathscr{B}$.

Definition 2.1 (probability distribution function *PDF*). *assigns a probability to each subset of the possible outcomes of a random variable from a sample space. It is a function of x which contains all the information necessary to compute* $P[E]$ $\forall E$ *in the Borel field of events*

$$F_x(x) = P[X \leq x] = P_x[(-\infty, x)] \tag{2.1}$$

Property 2.4. $F_X(\infty) = 1$ $F_X(-\infty) = 0$ $x_1 \leq x_2 \rightarrow F_X(x_1) \leq F_X(x_2)$

Definition 2.2 (the probability mass function $p(a)$). *is the probability distribution whose sample space is encoded by a discrete random variable X. It is positive for a countable number of values mapped by X.*

$$p(x_i) = \begin{cases} \geq 0 & i = 1, 2, \ldots \\ = 0 & \textit{all other values of } x \end{cases}$$

Example 2.1. *A data set* $X = \{4, 6, 8, 10, 12, 14\}$ *has 6 distinct observations and 12 different readings. The set of corresponding number of obervations per value (frequencies)* $Freq_X = \{4, 2, 2, 2, 1, 1\}$ *shows how many times each distinct value in the set occurs. The probability mass function* $f_X(10) = 0.167$ *is computed by dividing the number of times '10' is observed by the total number of readings '12'.*

Definition 2.3 (the probability density function *pdf*). *describes the relative likelihood a random variable is to take a value from a continuous sample space.*

The probability of the random variable lying within a range of values $\in \mathbb{R}$ is given by the integral of this variables density over that range, i.e. it is given by the area under the density function and above the horizontal axis and between the lowest and greatest values of the range $\in \mathbb{R}$. $f(x)$ is nonnegative $\forall x \in \mathbb{R}$, and its integral over the entire space is equal to one.

$$P\{X \in B\} = \int_B f(x)dx$$

Property 2.5 (probability distribution function of the uniform random variable). *defined over an interval* $(a, b]$

$$F_X(t) = \begin{cases} 0 & t \leq a \\ \frac{t}{b-a} & a \leq t \leq b \\ 1 & t \geq b \end{cases} \tag{2.2}$$

it is known that the distribution is equally (uniformly) likely to occur at any point lying within the interval bounding points $(a, b]$

Property 2.6 (the probability density function (pdf) of a uniformly distributed random variable). *defined over an interval* $[a, b]$

$$f_X(x) = \frac{1}{b-a} \quad \forall x\ a \leq x \leq b \tag{2.3}$$

2.1.2 Cumulative Distribution Function *cdf*

The *cdf*, is the *PDF*, it defines the accumulated probability density so far. More formally, $F_X(x)$ is the probability that a random variable X takes on a value less than or equal to x. On a discrete level, *cdf* is the summation of the probability mass function; equivalently, on the continuous level, it is obtained by integrating the probability density function (*pdf*) up to the item value x.

Definition 2.4 (cumulative distribution function). *Given an item value x, with density function $f(x)$, and an unknown variable (commonly referred to as the real-valued random variable) X, the cdf of x $F_X(x)$ is the function:*

$$\begin{aligned} F_X(x) &= \sum_{X \leq x} p_X(x) && \textit{discrete random variable} \\ F_X(x) &= \int_{-\infty}^{x} f_X(x)dx && \textit{continous random variable} \end{aligned} \tag{2.4}$$

The cdf values range between [0,1].

In example 2.1, the *cdf* value when the variable $X = 10$ is $F_X(10) = 0.833$; this value is computed by accumulating the probability density fucntion of the data values prior to 10.

Property 2.7. *Every* cdf *is monotonically increasing.*

The *cdf* associated with a density function is always increasing. All the data population resides between two points having *cdf* values '0' and '1'. The *cdf* slope increases and decreases together with the population behavior: faster growing *cdf* slopes represent higher proportion of the population residence at lower real values (conventionally named quantiles).

Property 2.8. *The* cdf *of a uniformly distributed random variable over a given interval [a,b] is given by Equation 2.2*

By observation, a uniformly distributed *cdf* is graphically represented by a line whose slope is $\frac{1}{b-a}$ between the two bounds of the interval $[a, b]$. The *cdf* of the variable is 0 for all values below a and it is 1 for all quantiles greater than b.

2.2 Joint Cumulative Distribution Function

Operations can be performed on *cdf*s but carry a different interpretation than operations over standard arithmetic calculus since they relate to probabilities. The joint operation is essential to our solver and is recalled below. The *joint cdf* results from superimposing densities of two random variables in a relation, each exerting a *cdf* on its own.

Definition 2.5 (joint *cdf*). *$F_{XY}(x,y)$ Given two random variables X and Y, their $f_{XY}(z)$*

$$F_{XY}(x,y) = P(X \leq x, Y \leq y) \tag{2.5}$$

For independent variables $P(X \leq x, Y \leq y) = F_X(x)F_Y(y)$.

When two random variables are involved in an addition relation, the convolution operation is an expensive joint pointwise product on their pair of densities and it produces a density function.

Definition 2.6 (convolution operation). *Given $z = x + y$, convolution is the probability density function $f_{XY}(z)$; where both random variables X and Y engaged in the addition take values up to x and y respectively; the resulting distribution is obtained by accumulating the densities for each value of $z = x + y \in (-\infty, \infty)$.*

Appendix A.3.1 details the derivation of the convolution operation which yields the probability distribution of the sum of two random variables.

d*iscrete random variable*

$$f_{XY}(z) = \sum_{x=-\infty}^{+\infty} f_X(x) f_Y(z-x) = \sum_{y=-\infty}^{+\infty} f_X(z-y) f_Y(y)$$

$$F_{XY}(z) = \sum_{x+y=-\infty}^{z} f_{XY}(z)$$

continous random variable

$$f_{XY}(z) = \int_{-\infty}^{+\infty} f_X(x) f_Y(z-x)dx = \int_{-\infty}^{+\infty} f_X(z-y) f_Y(x)dy$$

$$F_{XY}(z) = \int_{-\infty}^{z} f_{XY}(z)dz$$

Example 2.2. *In this example, we compute the convolution between the random variable illustrated in the data set from example 2.1 and another data set $Y = \{12, 15, 18, 21, 24\}$ with corresponding set of frequencies $Freq_Y = \{1, 3, 6, 5, 3\}$. Eventually convolution is computationally expensive because it is a pointwise operation. Figure A.2 depicts the* pdf *and the* cdf *addition distribution of the two discrete random variables.*

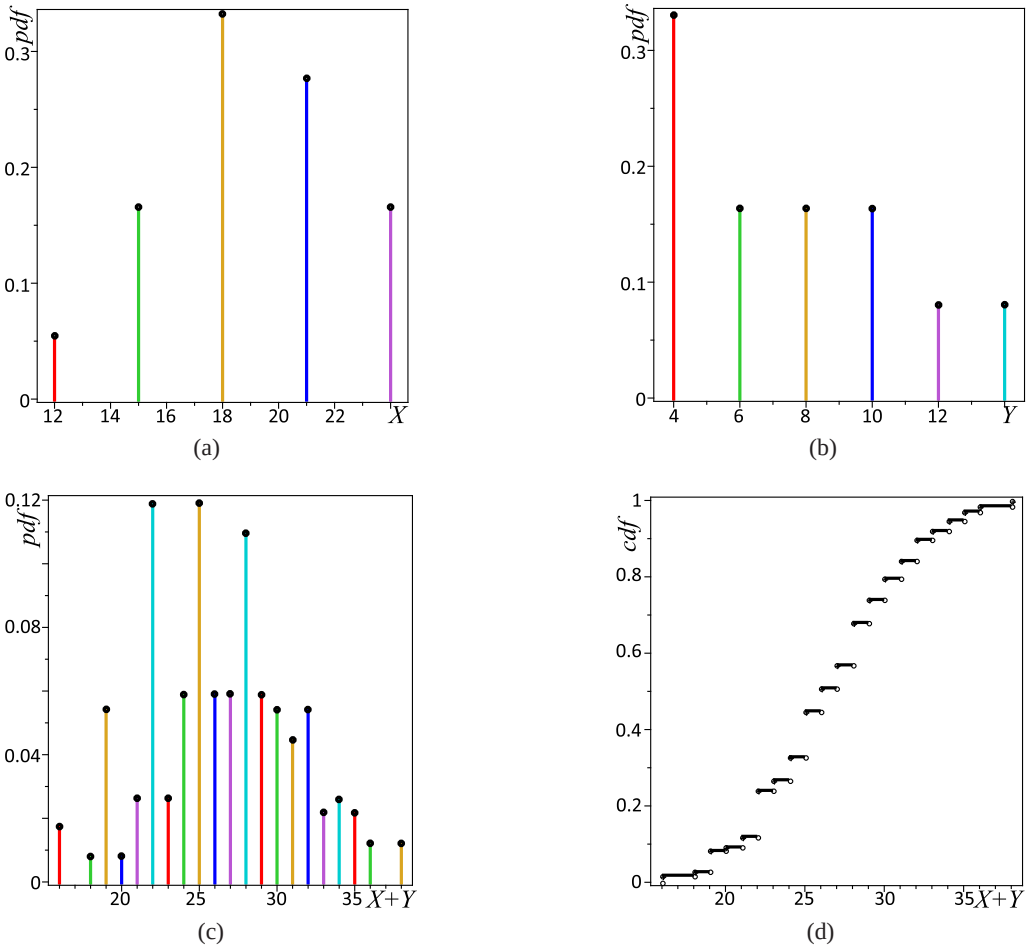


Fig. 2.1: Convolution of two random variable discrete distributions: (a) *pdf* of $X = \{4, 6, 8, 10, 12, 14\}$ with frequencies $\text{Freq}_X = \{4, 2, 2, 2, 1, 1\}$ (b) *pdf* of $Y = \{12, 15, 18, 21, 24\}$ with frequencies $\text{Freq}_Y = \{1, 3, 6, 5, 3\}$ (c) *pdf* of (X+Y) and (d) *cdf* of (X+Y)

2.3 Stochastic dominance

In probability theory the stochastic dominance defines a list of partial ordering on random variables. A random variable is dominating another when it is greater in the ordering.

Definition 2.7 (first order stochastic dominance). *Given X and Y, Y has a first order stochastic dominance over X; or in other words X is dominated by Y when*

$$P(X > x) \leq P(Y > x) \quad \forall x \in (-\infty, +\infty) \tag{2.6}$$

In terms of *cdf*s: $F_Y(x) \leq F_X(x) \quad \forall x \in (-\infty, +\infty)$. Example 2.2 shows that the random variable Y dominates X because computed values $P(X > 12) = 0.08$ and

$P(Y > 12) = 0.94$. Calculations are given by summing up probability densities for quantiles greater than 12. This computation indicates that the variable Y is more likely to occur at higher quantiles when compared to X. This observation is because of the probabilitic property, which enforces the sum of probabilities for a given random variable over the domain of reals $\mathbb{R}$ must always be equal to 1. First order stochastic dominance is illustrated in Fig. 2.2(a) $\forall x \in (-\infty, +\infty)$; it shows that Y is dominating X with respect to the first order stochastic dominance.

In the general case *cdf*s under comparison might not be comparable under the first order stochastic dominance. This illustrated in Fig. 2.2 (b) and (c) for some quantile x: $F_X(x) \leq F_Y(x)$. The second order stochastic dominance handles such cases to defined an ordering among the random variables.

Definition 2.8 (second order stochastic dominance). *If a random variable Y has a second order stochastic dominance over X. Then X is dominated by Y:*

$$F_X \leqslant_S F_Y, \quad \int_{-\infty}^{x} F_Y(x)dx \leq \int_{-\infty}^{x} F_X(x)dx \quad \forall x \in (-\infty, +\infty) \tag{2.7}$$

The integration of *cdf* calculates the area under the curve from $-\infty$ to $+\infty$.

Property 2.9. *Second stochastic ordering indicates that the mean of X is at least as high as that of Y*

Fig. 2.2 shows three examples in which we compare two random variables X and Y: Y has a second order stochastic dominance over X in all cases. We note that the first order stochastic dominance allows us to compare X and Y in the first illustration Fig. 2.2(a) only.

2.4 Probability Box (p-box)

P-boxes are interval-probabilistic bounds adopted in the literature to enclose an imprecisely known probability distribution Ferson, Kreinovich, Ginzburg, Myers, and Sentz (2003); Williamson and Downs (1990). A p-box is a convex structure that embraces a set of probabilities: $[\underline{F}, \overline{F}]$ denotes the set of nondecreasing cumulative distributions. $\underline{F}$ and $\overline{F}$, are respectively the dominated and dominant distribution bounds as depicted in Fig. 2.3

Definition 2.9 (p-box). $[\underline{F}_X, \overline{F}_X]$ *specifies the probability box of a random variable X whose distribution F_X is contained within the p-box bounds:*

$$\underline{F}_X(x) \leqslant_S F_X(x) \leqslant_S \overline{F}_X(x) \quad \forall x \in (-\infty, +\infty) \tag{2.8}$$

Where $\leqslant_S$ is the second order stochastic ordering of the probabilities.

Similar to numeric interval arithmetic, interval-probabilistic arithmetic is applied on random variables. Given two random variables X and Y, $F_X \in [\underline{F}_X, \overline{F}_X]$ and $F_Y \in$

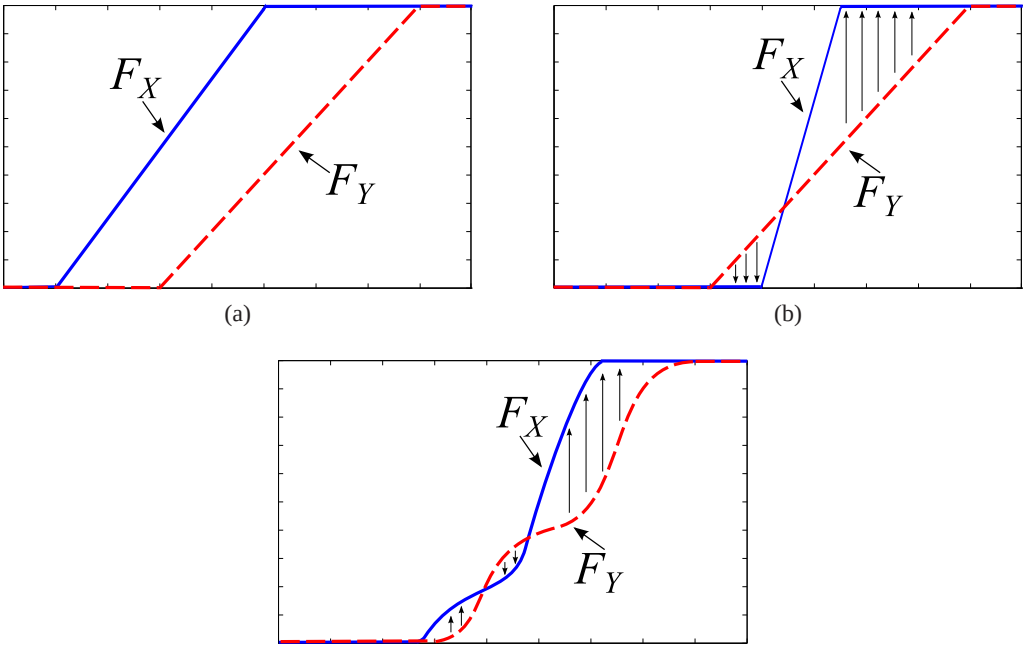


Fig. 2.2: Stochastic dominance: random variables X and Y shaping 2 uniform distributions in (a) where $F_Y(x) \leqslant_S F_X(x) \ \forall x \in (-\infty, +\infty)$; in (b) and (c) $\exists x \in (-\infty, +\infty)$ where $F_Y(x) \not\leq F_X(x)$ but $\int_{-\infty}^{x} F_Y(x)dx \leq \int_{-\infty}^{x} F_X(x)dx \quad \forall x \in (-\infty, +\infty)$

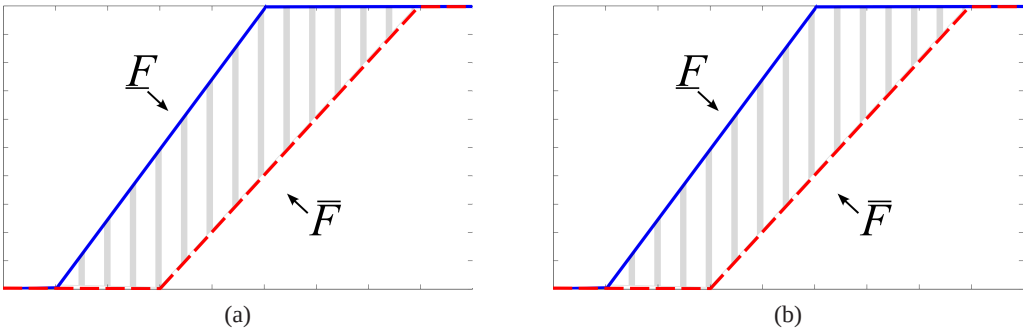


Fig. 2.3: p-box structure: (a) the general case (b) when bounds are uniformly distributed

$[\underline{F}_Y, \overline{F}_Y]$ are the uncertain *cdf* of X and Y respectively. The joint distribution of X and Y F_{XY} is unknown yet it is contained within the joint p-box structure defined by the joint bounds $[\underline{F}_{XY}, \overline{F}_{XY}]$.

$$\underline{F}_{XY} \leqslant_S F_{XY} \leqslant_S \overline{F}_{XY} \tag{2.9}$$

CHAPTER 3

LITERATURE REVIEW

So far research effort towards formalizing and solving large scale problems coupled with uncertainty by means of constraint systems is quite few. In the realm of OR, LP techniques are easily adaptable since they exploit the problem from a global perspective. By means of simple linear inequalities, LPs provide optimal decisions in real-world situations on a scale. On the other hand, the CP paradigm proved to have a considerable flexibility in formulating real-world combinatorial problems. It aims at building frameworks for fixed point semantics which result in narrowing the possible domains of solution. Within the past decade approaches from the CP and LP have been extended to handle forms of data uncertainty.

3.1 Uncertainty in the conceptual world

The representation of uncertainty is debatable. To deal with data surrendered by uncertainty many techniques have been proposed in the literature. The Gum (1995) initiated and introduced the International Organization for Standardization in (1993). Nielsen (2000) emphasized on seeking the root cause of uncertainty in the problem. Uncertainty in the literature is commonly due to two main reasons: insufficient information which yields errors, ill-defined data and ignorance; and fluctuating nature which a result of future forecasts, a stochastic nature or a dynamically changing environment Ferson and Ginzburg (1996). This uncertainty can be associated with the set of variables, domains or constraints in the problem definition Faltings (2006). Hence, it is important to appropriately represent the uncertainty because it affects not only the behavior of the problem in hand but also its acquired optimal solution. Work has been done to attach this uncertainty to the problem definition in order to deal with it in the conceptual world. Fig. 3.1 depicts a classification for the uncertainty found in the real world and how it is identified then represented from the problem definition to the conceptual world.

The set of plausibility measures is the most general and common approach that

subsumes all techniques which represent, deal and reason about data coupled with uncertainty in the conceptual world. They are classified by two main categories: probability and possibility. The set of plausibility measures include: probability measures, Dempster-Shafer belief functions, possibility measures, ranking functions and relative likelihood. The main idea is to map the set of possible worlds * in an algebra to some arbitrary partially ordered set [Halpern (2003)]. We briefly define and distinguish the difference between the concepts in the following:

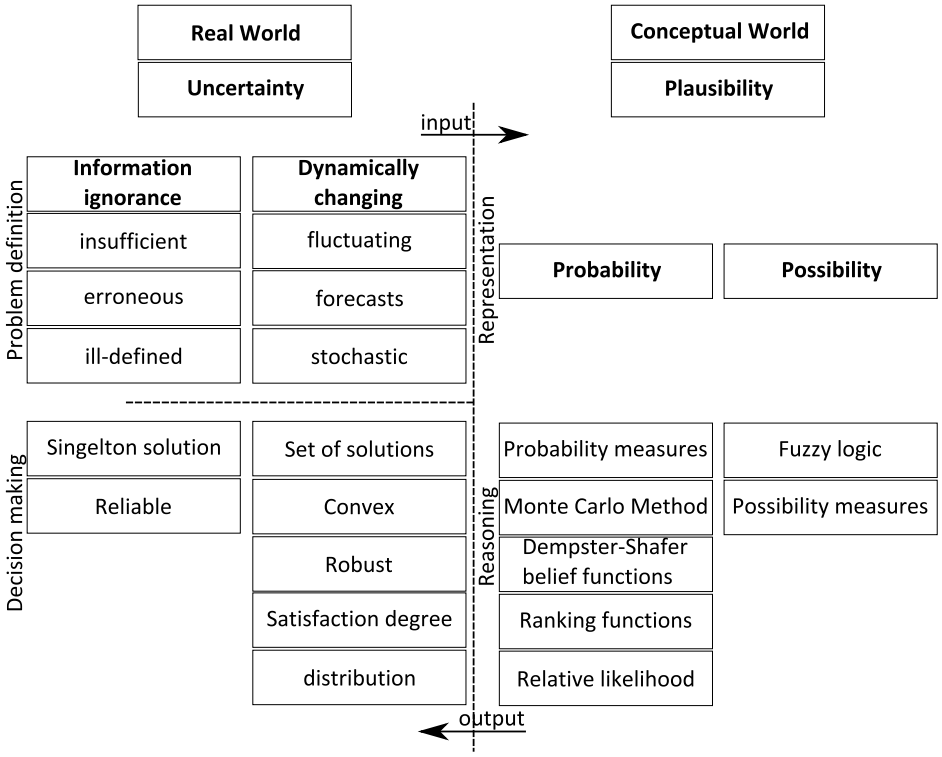


Fig. 3.1: Classification of uncertainty

Probability is a well-known powerful technique that is best utilized when numeric uncertainty is in place. A probability measure maps sets in an algebra over the set of possible worlds to [0,1]. When the probability concept is adopted, two major obstacles reveal: 1. numerical information is sometimes not fully provided in the problem definition. 2. Every two events must be probabilistically comparable which is not always the case.

The sets of probability measures define and concentrate on lower and upper bound probabilities. This concept is more appropriate when it is hard to represent the data whereabouts due to ignorance or lack of information.

*The set of states or elementary outcomes an agent might take into consideration

Monte Carlo analysis [Hammersley, Handscomb, and Weiss (1965)] is the most famous probabilistic approach which incorporates statistical evaluation of mathematical functions using random samples. The method is computation intensive and it requires the application of a large number of independent random samples, each of which seeks an output deterministically. Resulting outcomes are then integrated to form a numerical probabilistic distribution. In this method, the larger the number of random samples taken within a range, the more accurate results are obtained.

The Dempster-Shafer belief functions refer to the theory of evidence which was introduced by Arthur Dempster, then developed by Glenn Shafer. The theory attaches a degree of likelihood to the set of events. It is a modified version of the set of probability measures where belief is consider as the lower bound probability. The degree of evidence varies between [0,1], and the sum of the support over the set of possible worlds must be equal to 1. We classify the Dempster-Shafer belief functions under the umbrella of the probabilistic world since bounds in the theory of evidence are probabilities.

The Possibility measures are based on the well-known *fuzzy* logic Zadeh (1965). The main idea behind this logic is to attach a possibility measure varying between [0,1] to each subset of the possible worlds. In the possibilistic world computation is more structured and well-defined. This fact justifies why research tends to adopt *fuzzy* logic especially when dealing with uncertainty in the conceptual world.

The Ranking functions are utilized to order the set of possible worlds. This approach attaches a natural number or infinity to each and every possible set. Higher numbers mean greater degree of surprise associated with the set.

The Relative likelihood is used to order each and every set in the possible worlds which are assigned a degree of likelihood that ranges between $[0, 1]$

Despite the fact that it is hard to distinguish between the two data types, research commonly differentiate between ill-defined information and fluctuating data when dealing with them. Reasoning about data coupled with uncertainty uses different mathematical propagation techniques: convex models are favored when ignorance takes place while probabilistic models are best adopted when the data has a fluctuating nature. Both techniques yield an identical range of data values. Interval analysis are characterized to be more conservative. They can often consider many unnecessary outcomes along with important ones. Probabilistic approaches add a quantitative information that expresses the likelihood, yet these approaches impose assumptions on the distribution shape in order to conceptually deal with it in a mathematical manner. Table 3.1 shows the relative execution time for all models given that a deterministic instance takes E time units as described in Ferson and Ginzburg (1996).

Model	Execution time
Deterministic point estimate	E
Interval analysis	$4E$
Monte Carlo	NE where $N \in [100, 50000]$
Belief and ranking	K^2E where $K \in [20, 100]$

Table 3.1: Relative execution time. E is the execution time taken by a deterministic model. N is the number of random samples

Solutions to problems coupled with uncertainty follow two main approaches: proactive or reactive. The first approach use all the available knowledge in order to provide solutions which maintain the uncertainty in a robust manner, i.e. irrespective of the whereabouts still a solution is validated for the given uncertain data. Solutions for the reactive approach are flexible, i.e. for any change, solutions might be reused to produce new ones which rely on previous states. Output solutions are classified, as shown in Fig. 3.1 into: a single solution or a set of solutions. The set of solutions obtained are further classified as: convex, robust, having a satisfaction degree or following a probability distribution.

3.2 Implementation Techniques

Systems of constraints are usually embedded in declarative programming languages in order to obtain an intuitive descriptive algebraic structure. They are heavily used in the problem solving environments where we need to find admissible solutions which satisfy a large set of constraints. The main idea behind these solving techniques is to separate logic from control. This separation allows users to easily extend and manipulate existing programs. This is achieved by defining constraints and allow the solver to search for feasible solutions. In a constraint solving framework, variables are constraints over different domains, then diverse query-answer mechanisms from Artificial Intelligence (AI) to Operation Research (OR) are used to find solutions respecting a large number of constraints. Two main competing line of research adopting those mechanisms have emerged, they are classified as: Constraint Programming (CP) and Linear Programming (LP). Techniques for hybridization incorporating advantages existing in both paradigms have been recently evolved [Hooker (2006)].

3.2.1 Constraint Programming line of research

The CP paradigm is a powerful technique that intuitively expresses and formulates decision problems. They aim at building frameworks which result in narrowing domains of variables based on a predefined set of constraints using fixed point semantic techniques. A CSP consists of a set of decision variables over some domain of values and connected by a set of relations (constraints). A constraint solver, given this representation, seeks an assignment to decision variables that satisfy the set of constraints. By means

of predefined inference rule reduction, a constraint solver minimizes the search space to be visited, then it uses backtracking, branch and bound, or local search mechanisms to acquire a solution within the reduced space. This process is NP-Complete. Global constraints are problem specific constraints which might be added to the solver in order to allow for a more efficient and effective search mechanism. Extensions to the CSP framework include: finding optimal solutions, associating preference with constraints and reasoning within a distributed environment.

Definition 3.1 (a classical CSP). *is a triple* $\mathcal{P} = \langle X, D, C \rangle$ *where* X *is the set of n problem variables* $X = \{x_1, x_2, \ldots, x_n\}$, D *is the set of n variable domains* $D = \{D_1, D_2, \ldots, D_n\}$, *and* C *is the set of t constraints* $C = \{C_1, C_2, \ldots, C_t\}$.

Definition 3.2 (a constraint). $C_j = \langle R_{S_j}, S_j \rangle$ *is defined by the relation* R_{S_j} *on the constraint scope* $S_i = scope(C_i)$ *which expresses the Cartesian product of the variable domains in the relation* R_i.

Definition 3.3 (a solution to the CSP $\mathcal{P}$). *is an n-tuple assignment* $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ *where* $a_i \in D_i$ *and which satisfies* C_j *defined over a relation* R_{S_j}.

A given task is to search for one solution or all set of CSP solutions $sol(\mathcal{P})$. This set is empty when the CSP is unsatisfiable. CSP have been tackled by diverse techniques and they became a very interesting topic in many fields of computer science and beyond.

Example 3.1 (Course scheduling). *We illustrate the different constraint programing approaches using the course scheduling example described in Faltings (2006). Consider that we need to schedule a short course, with lectures, practical sessions and tutorials, over three days. This schedule can be modeled in a deterministic CSP using* 9 *different variables labeled as* x_{ij}, *where* i *symbolizes the day number and* j *signifies the session type (*1 *= lecture,* 2 *= practical session and* 3 *= tutorial). The domain of each variable varies between* $\{0, 1, 2, 3, 4, 5\}$ *indicating the number of sessions assigned on a given day. The set of constraints are defined as follows:*

$$\forall i \sum_{j=1}^{3} x_{ij} \geq 2 \tag{3.1}$$

$$\forall j \sum_{i=1}^{3} x_{ij} \in 1, 2, \ldots, 5 \tag{3.2}$$

$$\sum_{i=1}^{3} \sum_{j=1}^{3} x_{ij} \in 10, 11, 12 \tag{3.3}$$

The first constraint indicates that the number of sessions per day cannot be less than 2*; the second constraint states that the schedule must contain between* 1 *to* 5 *sessions per type; and the third constraint ensures that the total number of sessions over the*

three days should be between 10 *to* 12. *Fig. 3.2 depicts the modeling process of the course scheduling problem and one solution instance given from the search space which satisfies all defined constraints in the deterministic case.*

Type

Days \ *J*	*L(1)*	*P(2)*	*T(3)*
1	x_{11}	x_{12}	x_{13}
2	x_{21}	x_{22}	x_{23}
3	x_{31}	x_{32}	x_{33}

(a)

Type

Days \ *J*	*L(1)*	*P(2)*	*T(3)*
1	1	0	1
2	1	1	0
3	0	2	4

(b)

Fig. 3.2: Course scheduling problem: (a) 9 *variables model, (b) one solution instance*

3.2.2 Linear Programming line of research

LP techniques are easily adaptable to provide optimal decisions in real-world situations on a scale. Unlike CP which explores the problem partially through variable domain propagation techniques, LP exploits the problem from a global perspective using simple linear inequalities which geometrically correspond to a convex polyhedron[†]. An LP problem identifies a set of decision variables, a set of linear constraints and a linear objective function.

Definition 3.4. *a classical LP problem is described as:*

$$\begin{aligned} &\min cx && \textit{// objective function (linear)} \\ &\textit{subject to} \quad Ax \geq b && \textit{// resource constraints} \\ &x \geq 0, x \in \mathcal{R}^n \end{aligned}$$

Where A is an $m \times n$ matrix.

Once an LP problem is formalized we can infer its dual. The duality theory is a strong tool that is well-suited for sensitivity analysis and variable domain filtering.

Definition 3.5. *the dual of an LP is described as:*

$$\begin{aligned} &\max \lambda b && \textit{// objective function (linear)} \\ &\textit{subject to} \quad \lambda A \leq c && \textit{// resource constraints} \\ &\lambda \geq 0, \lambda \in \mathcal{R}^m \end{aligned}$$

Property 3.1. *an LP problem is characterized to be:*

[†]The polyhedron is the convex region enclosing the set of feasible solutions and that has vertices each of which is a basic feasible solution

$\mathcal{P}1$. *Unbounded when none of the feasible solutions is optimal*

$\mathcal{P}2$. *Infeasible when no feasible solution exists*

$\mathcal{P}3$. *Have an optimal solution.*

Both the primal problem and its dual yield the same optimal solution when they are bounded and feasible ('strong duality'). Generally the optimal solution of an LP problem is obtained using the Simplex method which iteratively builds a sequence of adjacent basic feasible solutions. This Simplex method is based on the geometric structure of the problem.

Definition 3.6. *an Integer Programming (IP) problem is an LP problem with integrality constraint*

Definition 3.7. *a Mixed Integer Programming (MIP) problem is an LP problem where some variables coupled with integrality constraints*

Definition 3.8. *the convex hull tightest possible convex set containing feasible integral solutions*

The course scheduling problem in example 3.1 can be represented by LP. Yet since the problem consists of 9 variables, we couldn't represent it graphically. To easily visualize the geometrical feature of an LP problem, we refer to a production scheduling problem that consists of two variables and two resources and which is studied thoroughly in [Thipwiwatpotjana (2010)].

Example 3.2. *Production scheduling problem A producer depends on two main resources x_1 and x_2 (supplies of two grades of mineral oil) in the manufacturing of two products A and B and wants to minimize the production cost. The problem information and set of constraints are listed in Table 3.2 and the LP model deduced out of this information is described as follows:*

$$\begin{aligned} & \min z = 2x_1 + 3x_2 \\ \textit{subject to} \quad & 2x_1 + 6.1667x_2 \geq 174.83, \\ & 3x_1 + 3x_2 \geq 161.75, \\ & x_1 + x_2 \leq 100, \\ & x_1, x_2 \geq 0. \end{aligned}$$

Where the first constraint describes the situation where the sum of supplies which form the product A should meet its demands. Similarly, the second constraint ensures product B demands satisfaction. The constraint $x_1 + x_2 \leq 100$ sets the limit on the supply. Fig. 3.3 illustrates the derivation of the optimal solution by drawing the linear inequalities and obtaining their intersections to search for the minimum possible cost. Using this LP approach we obtain one optimal solution $(x_1^*, x_2^*) = (37.84, 16.08)$ that

Mineral Oil (fl.oz.)	*Products*		*Costs (£/fl.oz.)*	*Limit on the processed amount of mineral oil (fl.oz.))*
	Type A (oz./fl.oz.)	*Type B (oz./fl.oz.)*		
x_1	2	3	2	1
x_2	6.1667	3	3	1
	≥	≥	=	≤
	Demands			
	174.83	161.75	z	100

Table 3.2: Production scheduling problem definition

achieves a minimum cost $z^* = \$123.92$. *This optimal solution is derived from the set of feasible solutions bounded by the polyhedron depicted in Fig. 3.3. Note that this solution was obtained using the graphical property of linear inequalities which tackle the problem from a global point of view.*

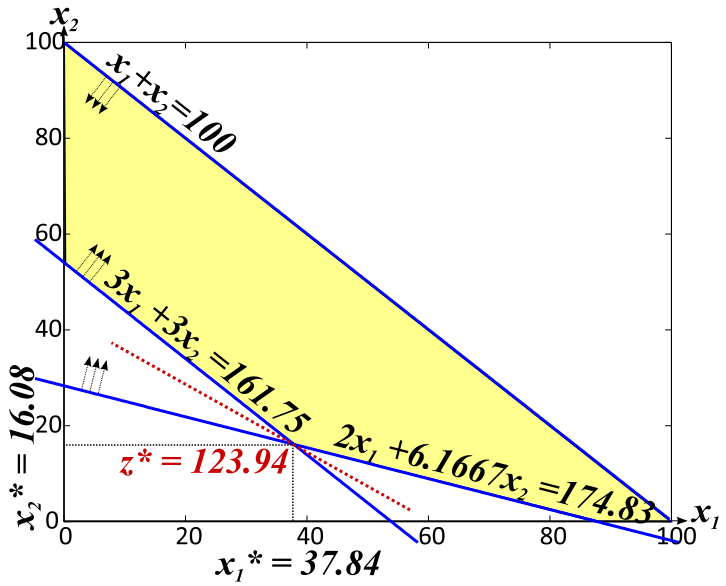


Fig. 3.3: Polyhedron (feasible region) derived for the production scheduling problem in example 3.2

3.3 Modeling and reasoning with uncertainty

We describe existing uncertainty in the conceptual world. We review in the context of uncertainty how different approaches are adopted in the implementation of the models

3.3.1 Probabilistic/Stochastic paradigm

General Description

Probabilistic models are the most generalized statistical frameworks that articulate problems questing decision analysis. They aim at numerically representing the uncertainty brought into the problem under consideration. Uncertainty is described in a probabilistic format assuming a probability value for each scenario. Approaches for stochastic programming search for optimality of one given scenario provided in the problem. By means of statistics and probability distribution properties: expectation, variance and correlation, a stochastic model iteratively generates potential outputs for the set of overwhelming input randomly distributed data; this iterative process produces probable solutions that follow random distributions and accordingly the realization of the projected maximum likelihood outcome is explored.

Input data

Input data, in the problem definition, is often observed in order to accurately incorporate the distribution of the random process, or is based on historical data to serve for a better realization of future forecasts. Statistical tools seek to draw the best fit probability distribution of the problem under consideration.

CP implementations

Probabilistic CSPs define the uncertainty over the presence of constraints. In their model, each constraint is associated with an independent probability value that evaluates its degree of satisfaction. Final decision produces an assignment that maximizes the probability of consistency which satisfies the set of the given constraints. Implementation of the Probabilistic CSPs can be found in the well-known soft constraint frameworks: Valued CSP Schiex, Fargier, and Verfaillie (1995) and semirings Bistarelli et al. (1999).

Example 3.3. *Consider three additional constraints in the course scheduling CSP, detailed in example 3.1, each is associated with a probability value. New constraints are defined as a probabilistic CSP as follows:*

$$\sum_{i=1}^{3} x_{i2} \geq \sum_{i=1}^{3} x_{i1}, (P = 0.6) \tag{3.4}$$

$$x_{32} > x_{12}, (P = 0.5) \tag{3.5}$$

$$\sum_{i=1}^{3} x_{i2} \leq 2, (P = 0.2) \tag{3.6}$$

Where the number of practical sessions must be greater than the number of lectures; on day 3 *the number of practical session is greater than that of day* 1*; and the total*

number of practical sessions is less than 2*. Each constraint is associated with a probability value:* 0.6, 0.5 *and* 0.2 *respectively. The final decision is an assignment to all variables with a maximum probability. Fig. 3.4 shows an assignment with this maximum probability.*

Type

Days i \ j	*L(1)*	*P(2)*	*T(3)*
1	1	0	2
2	1	1	1
3	0	2	2

Fig. 3.4: Course scheduling problem: one solution instance in the probabilistic CSP representation

Mixed CSPs Fargier et al. (1996) describe problems with unknown set of constraints. Variables are divided into controllable and uncontrollable parameters. The second type of parameters might be associated with a probability distribution. A final decision is a reliable solution that satisfies all probable occurrences of parameters with the maximum possible probability value.

Example 3.4. *Referring to example 3.1 where we suppose that tutorials on day* 3 *and lectures on day* 2 *will be determined later upon reveal of the tutor schedule. Fig. 3.5 shows that both variables* x_{13} *and* x_{21} *are uncontrollable and are given possible domains in the initial state* $\{0, 1\}$ *and* $\{0, 1, 2\}$ *respectively. Assume that the probrability distribution over uncontrollable variables is predefined over each value in the domain as* $\{0 : 0.3, 1 : 0.7\}$ *for* x_{13} *and* $\{0 : 0.5, 1 : 0.4, 2 : 0.1\}$ *for* x_{21}*. Accordingly, each solution in the search space is associated with* 6 *possible solutions for both variables* x_{13} *and* x_{21} *in the Mixed CSP, and each combination resulting from the Cartesian product of the uncontrollable variable domains might be associated with a probability value (the intersection/product of probability values involved). The solution suggested in Fig. 3.5 is associated with* 4 *possible domain combinations since* $\{0, 0\}$,$\{1, 2\}$ *should be omitted from the solution space in order to ensure that the last constraint (indicating that the total should be between* $\{10, 11, 12\}$*) in the problem is satisfied. Obtained solution has a maximal probability value equal to* 93%

Dynamic CSPs deal with problems that change over time in a sequential manner. Those changes can affect: variables, domains, constraints in terms of scope and definition. Each problem in the sequence is defined from previous states. Dynamic CSPs in Mittal and Falkenhainer (1990) focus on a subset of variables and constraints that are subject to change (activity constraints). Solving Dynamic CSPs follow three main strategies: minimize the need for change by seeking a robust solution; minimize the cost

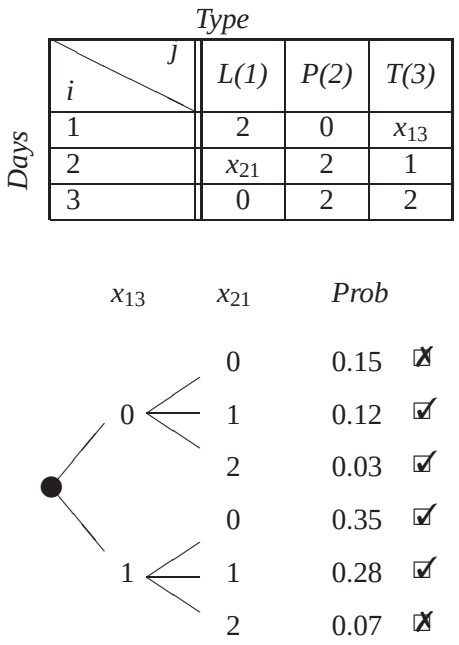


Fig. 3.5: Course scheduling problem: one solution instance in the Mixed CSP representation with a maximum probability 0.93

of change by acquiring a stable solution; and minimize the reaction time by questing quick solutions. Solving strategies are concerned with three main types of problem:

1. Unknown future: this type of problem is tackled using the local repair methods which apply minor modifications on assignments from previous problem states until an acceptable solution is obtained. The min-conflict heuristic detailed in Minton, Johnston, Philips, and Laird (1992) minimizes the number of unsatisfied constraints using problem dependent heuristics. The Local Changes algorithm in Verfaillie and Schiex (1994) resolve the conflict by partitioning the problem variables into three sets based on their assignments: fixed, subject to be modified or unassigned variables. The algorithm tempts to apply modifications on the assignments of variables in the second set until it reaches an admissible solution. Petcu and Faltings (2005) retain solution stability by adding new special constraints to be satisfied. El Sakkout and Wallace (2000) define linear minimal perturbation functions over a solution in a prior state, then they are defined as objective in the new problem states. Barták, Müller, and Rudová (2004) extend these functions to solve over-constrained problems.

2. Unknown type of change: solutions to this type of problems follow the oracle approach Van Hentenryck and Le Provost (1991). In this approach the structure of prior states in the sequence are retained and solutions to new problems main-

tain the same path (fruitless sub-trees are pruned from the search space). Jussien (2003) adds explanations (problem-specific constraints) to the problem structure which support the change during the search.

3. Uncertain information about the change in the future: using the recurrent CSP approach Wallace and Freuder (1998) to record the source and frequency of the change. The aim is to find a robust solution that maintains all possible changes. This type of solution is characterized to be proactive irrespective of the change. Note that Stochastic CSPs Walsh (2000) and Branching CSPs Fowler and Brown (2003) are different forms of the recurrent CSP approach

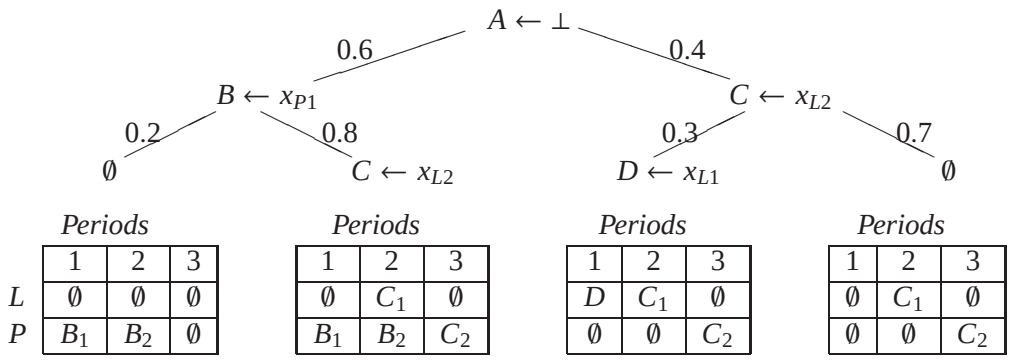
Other forms of Dynamic CSP can be found in Open CSPs Faltings and Macho-Gonzalez (2005). They are generally found in a distributed environment where the set of variables and their constraints are initially defined but domain values and tuple relations are revealed over time. Open CSPs follow an interactive approach Lamma et al. (1999) where changes result in the extension of domains and tuple relations. They are acquired online by querying the network resources for available information. The querying mechanism in this type of problems is usually the most expensive operation.

Stochastic CSPs exist in two forms: policy-based in Walsh (2000) and scenario-based Tarim et al. (2006). The policy-based approach, like Mixed CSPs, classify variables into controllable (decision variables are assigned the OR nodes) and uncontrollable (stochastic variables are assigned the AND nodes). It is a multi-stage probabilistic CSP wherein a probability distribution is associated with the domain of each state variable. Requests alternate states and decision variables. Similar to Probabilistic CSPs a solution is an assignment of decision variables that maximizes the probability of consistency. The scenario-based approach exhaustively builds all possible sets of scenarios based on the input probability distribution. Each path in the tree is associated with a probability value and each scenario is considered as a classical CSP on its own. The scenario-based approach reacts to values suggested by stochastic variables in different manners.

Branching CSPs Fowler and Brown (2003) follow the Markov decision process MDP in Puterman (2009) to model sequential decision problems with a lack of knowledge about input variables and their associated set of constraints. Variables are dealt with as soon as they are revealed by time, in a sequential manner. Absent variables are associated with a probability value. A querying mechanism is exerted, at each step, in which a new added variable is assigned a probability value that maximizes the global expected utility.

Example 3.5. *Consider a modified version of the course scheduling problem in example 3.1. For simplicity, a list of constraints requesting the allocation of one-hour time slot of lectures and practical sessions in one room are revealed in sequence. (A) one hour lecture followed immediately by a one hour practical; (B) two practical hours; (C) one*

hour lecture followed later by a one hour practical; (D) one hour lecture. As soon as they are issued each is given a time slot (revenue) or rejected (no revenue). The aim is to maximize the overall expected revenue. Fig. 3.6 depicts the decision tree and shows how to derive a solution instance for the Branching CSP which has a maximum overall expected revenue equal to 11.88.



$Pr = 0.12, R = 6$ $Pr = 0.48, R = 15$ $Pr = 0.12, R = 12$ $Pr = 0.28, R = 9$

Fig. 3.6: Course scheduling problem: one solution instance in the Branching CSP representation with a maximum overall expected revenue 11.88

LP implementations

Sampling Techniques Sampling techniques approximate a large number of scenarios to be solved in a deterministic manner. Generally, the model is given as:

$$min_x\{c^T x : \mathbf{A}x \leq \mathbf{b}\}, \text{and}, Pr\{(\mathbf{A}, \mathbf{b}) = (\mathbf{A}^s, \mathbf{b}^s)\} = p_s, \forall s = 1, \ldots, S \qquad (3.7)$$

One of the two methods is adopted: 1. estimating coefficients and right-hand sides within a decomposition scheme; 2. generating sample problems which are solved using a deterministic algorithm. The quality of the solution set that differentiates between various approaches is how close the objective value is to the true optimal objective of the problem.

Sampling techniques in the literature can be found in: Crude-Monte-Carlo, Monte-Carlo-Pre-Sampling, Importance Sampling Infanger (1999). The implementation of these approaches can be found in the DECIS/GAMS interface that uses the CPLEX solver callable library Optimization (1989). It is shown that both the Crude-Monte-Carlo and the Importance Sampling approaches extract quantities of the random parameters from their distributions without/with variance reduction techniques respectively. The Monte Carlo pre-sampling method defines the parameters over the sample of the

random distributions. DECIS's implementation computes an approximation of the objective value along with a confidence interval. It was shown in Infanger (1992) that sample size plays a major role in the quality of the solution set obtained. Higher values of the sample size yield solution sets which are closer to the optimal value. However, increasing the sample size is computationally exhaustive.

Other sampling techniques which seek the decrease of the number of LP problems solved are: the Sample Average Approximation and the Stochastic Decomposition methods. The former is based on the well-known Monte-Carlo simulation method. The problem is iteratively solved deterministically on estimations of a random sample. The method uses decomposition and branch-and-cut to compute an approximation of the objective function Verweij, Ahmed, Kleywegt, Nemhauser, and Shapiro (2003). On the other hand, the Stochastic Decomposition is a two-stage stochastic programming algorithm that uses random observations of random variables. These observations are used in a deterministic Benders' decomposition algorithm as a sequence of incumbent solutions which converge to a unique optimal. The stochastic decomposition as pointed out in Sen, Zhou, and Huang (2011) is known to have a faster computational speed when compared to other sampling techniques.

Mean risk model [Markowitz (1952)] characterizes the uncertainty by two different features: the mean, which describes the expected outcome and the risk, which evaluates the dispersion from the mean values. In this model, a trade-off analysis between the mean and the risk is achieved using a multi-objective optimization technique that maximizes the mean outcome while minimizing the risk factor. The objective function of the mean-risk model is rewritten as:

$$\max cx - \lambda x\prime Vx, x \geq 0$$

Where V is the variance coefficient matrix which contains the deviation from the mean values of the variable coefficients.

Chance-constraint model Charnes, Cooper, and Symonds (1958) and Prekopa (1973) searches for feasible solutions which are reliable in the given uncertain environment. Constraints are associated with a probability degree of satisfaction which is maximized in the reasoning process. A chance-constraint LP model is reformulated as:

$$\max cx : s.t.P(Ax \geq b) \geq \alpha, x \geq 0$$

Where $\alpha \in [0, 1]$. We used this model in Chapter 10, as a linear programming interpretation of the inventory control example, in order to exert a performance analysis of models having different uncertainty interpretations.

Recourse/multi-stage model [Dantzig (1955)] partitions the decision variables on two-stages: the first is solvable using a deterministic LP. The second stage is a subsequent of the first-stage decision and the actual realization of the uncertainty parameters.

It is a corrective step which filters infeasible values issued from specific realization of uncertainty that is based on a random distribution. Extensions to the model exist in studying the convexity of the recourse function when the defined random distribution is discrete [Wets (1974)] and developing scenario-based policies through decomposition techniques for continuous uncertain parameters [Birge and Louveaux (1988)]. The two-stage is further extended to a multi-stage formulation which builds a scenario-based tree which uses the uncertainty in the filtration process [Rockafellar and Wets (1991) and Sen and Higle (1999)].

Example 3.6. *Refer back to the production scheduling problem in example 3.2. The manufacturer should decide and plan ahead the production schedule and needed resources to meet the uncertain customer demands before they are realized. This plan needs to maintain the minimal possible production cost. In case when demands are not satisfied a shortage penalty is added. Given that*

$$\hat{a}_{11} = \{1 : 1/4, 2 : 1/2, 3 : 1/4\}, \hat{a}_{12} = \{5 : 1/6, 6 : 1/2, 7 : 1/3\}$$
$$\hat{b}_1 = \{149 : 5/12, 180 : 1/3, 211 : 1/4\}, \hat{b}_2 = \{138 : 1/4, 162 : 1/2, 185 : 1/4\}$$

where $\{v : p\}$ *indicates the value and its probability of occurrence, the two-stage recourse model is formulated as follows:*

$$\begin{aligned} &\min 2x_1 + 3x_2 + E_\zeta Q(x_1, x_2, \zeta) \\ &s.t. x_1 + x_2 \leq 100, x_1, x_2 \geq 0. \end{aligned} \tag{3.8}$$

ζ *is the random vector* $(\hat{a}_{11}, \hat{a}_{12}, \ldots, \hat{a}_{mn}, \hat{b}_1, \hat{b}_2, \ldots, \hat{b}_m)$*, defining* $\hat{a}_{ij}$ *and* $\hat{b}_i$ *as random variables with m number of constraints and N possible scenarios each of which has a different probability of occurrence. Hence for each* $j = 1, 2, \ldots, 81$*, we have:*

$$\begin{aligned} Q(x_1, x_2, \zeta^j) &:= 7\max\{b_1^j - a_{11}^j x_1 - a_{12}^j x_2, 0\} + 12\max\{b_2^j - 3x_1 - 3x_2, 0\} \\ &= \max 7y_1 + 12y_2 \\ s.t. a_{11}^j x_1 + a_{12}^j x_2 + y_1(\zeta^j) \geq b_1^j, &\quad 3x_1 + 3x_2 + y_2(\zeta^j) \geq b_2^j, \quad y_1, y_2 \geq 0. \end{aligned} \tag{3.9}$$

Where 7 *is the penalty cost when there is a shortage in the production of type A. Similarly,* 12 *is the shortage penalty cost of type B. This formulation yields an optimal solution of* $x_1^* = 31.80$ *and* $x_2^* = 29.87$ *with a first-stage minimum cost* 153.21$ *and a sum of costs* 155.38$*. This optimal solution is violated with minimum probability of* $\frac{1}{4}\frac{1}{6}\frac{1}{4} = \frac{1}{96}$ *by the constraint* $x_1 + 5x_2 \geq 211$

The Minimax Regret models Eldar, Ben-Tal, and Nemirovski (2004) seek the mitigation of the given uncertainty rather than anticipating it. This is done by defining the regret function which is the solution deviation from the true objective value of the problem. The model constructs k set of deterministic scenarios. The aim is to minimize the maximum regret function ('worst regret function') that is due to uncertainty and which is identified as: $R(x) = \max\{c^T x - z(\zeta^k)\}, k = 1, 2, \ldots, N$

Example 3.7. *Consider the production planning example with uncertain demands for type A and type B are given by* $b_1 \in \{149, 180, 211\}$ *and* $b_2 \in \{138, 162, 185\}$ *respectively and their production schedule is given by* $a_{11} \in \{1, 2, 3\}$ *and* $a_{12} \in \{5, 6, 7\}$. *The worst-regret function which needs to be minimized is* $R(x_1, x_2) = \max\{2x_1 + 3x_2 - z(a_{11}, a_{12}, b_1, b_2)\}$

$$\begin{aligned} z(\zeta^k) = z(a_{11}, a_{12}, b_1, b_2) := \min\, & 2x_1 + 3x_2 \\ s.t.\quad & a_{11}x_1 + a_{12}x_2 \geq b_1, 3x_1 + 3x_2 \geq b_2, \\ & x_1 + x_2 \leq 100, x_1, x_2 \geq 0. \end{aligned}$$

The minimum value of z in this case is 97.74, *found when* $a_{11} = 3$, $a_{12} = 6$, $b_1 = 149$ *and* $b_2 = 138$. *Consequently the model can be rewritten as:*

$$\begin{aligned} \min\, & R(x_1, x_2) = 2x_1 + 3x_2 - 94.74 \\ s.t.\quad & x_1 + 5x_2 \geq 211, 3x_1 + 3x_2 \geq 185, \\ & x_1 + x_2 \leq 100, x_1, x_2 \geq 0. \end{aligned}$$

Obtained maximum regret from this problem is $R^* = 65.91$ *with* $x_1^* = 24.33$ *and* $x_2^* = 37.33$. *This output solution signifies that not knowing the values given in the uncertain sets (*a_{11}, a_{12}, b_1, *and* b_2*) the worst regret based on the choices of the optimal* x_1 *and* x_2 *is* 65.91$.

Benefits

- Flexible enough to describe the uncertainty nature of real-world problems
- Employed when we involve large data input sets
- Analytical calculations using differential equations are ‘the most expressive behavior representation paradigm‘
- Stochastic models are best candidates when forecasting future expectations are required.

Drawbacks

- Not flexible enough to represent problems with ill-defined data; in this class of problems the probability distribution of random input variables is unknown.
- Representation of large scale systems of differential equations is infeasible
- Exerting point-by-point convolution for probability distributions is an exhaustive computation
- Stochastic models are scenario-based hence the searching process depends on the nature of the current situation or the scenario being explored

- It is based on exhaustive search techniques thus does not guarantee optimal solution exploration in large scale problems.
- Due to its exhaustive search nature, research that seeks integrating stochastic paradigm into other methods is quite few

Output solutions

Analytical calculations are involved in the propagation process. Such calculations include thorough point convolution of the provided input random distributions. They typically rely on differential and integral equations ‘convolution product‘ to derive the output random distribution.

- It is a typical representation of the random element introduced in the problem
- In order to explore the tree-based scenarios, generic search techniques are employed: tabu search and genetic algorithms
- Iteratively produces randomly distributed outcomes from random input data

3.3.2 Possibilistic paradigm

General Description

The aim of introducing *fuzzy* models is to approximate the probability distribution by a set of intervals (called alpha-cuts). This is to ensure simplified computation when Zadeh’s extension possibilistic theorem, in Zadeh (1965), is applied on the produced set for reasoning about the data. Examples of formalizing the possibilistic distribution to approximate real data uncertainty imposed by the measurement process can be found in Mauris, Berrah, Foulloy, and Haurat (2000); Mauris, Lasserre, and Foulloy (2001); Van De Ree and Jager (1993); Urbanski and Wsowski (2003); Ferrero and Salicone (2004); Mauris (2007). *Fuzzy* models are best used when the data is ill-defined; i.e. its probability distribution is unknown. They are conservative by adequately describe uncertainty in measurement resulting from ‘systematic error‘ [Gum (1995)].

Input data

The possibilistic distribution is built based on a computed standard deviation: assuming, in the general case, a unimodal and symmetric distribution if the probability distribution is not known:

- Input data to the model that results from the measurement process is often uncertain or erroneous.
- Data is symbolized by two main intervals (alpha-cuts): worst (kernel) and best (support) bound approximation of the probability distribution. This is provided by assuming a standard deviation for the distribution being observed.

- Formalizing the possibilistic distribution presumes and follows a generic probability distribution: Guassian, Triangular, Rectangular and U-Shaped [Gum (1995)].

CP implementations

***Fuzzy* CSPs** Dubois, Fargier, and Prade (1996) Attaching fuzziness to CP formalism has been thoroughly researched. CP models based on the possibilistic theory aim at providing a degree of 'expressiveness' to traditional CP approaches in order to look for robust solution: a solution which successfully satisfies as many realization of data as possible. However, the degree of imprecision is implemented on constraint tuples: in a discrete manner. There is a lack of investigation for continuous input data and subsequent solution space regarding this approach. CP implementations following the *fuzzy* membership approach are used to describe soft constraints Meseguer, Rossi, and Schiex (2006) and prioritized constraints Schiex (1992). The former attaches a level of satisfaction preference to constraints. Solutions to the problem are those with maximal constraint satisfiability. The priority constraint model sets a priority degree which quantifies the satisfaction degree of the constraint. The aim is to satisfy the most important constraints. *Fuzzy* CSPs can be intuitively integrated into soft constrained frameworks: Valued CSP Schiex et al. (1995) and semirings Bistarelli et al. (1999). During the search process preferences and priorities are used to support the search heuristics which focus on the most promising instances.

Example 3.8. *Consider the* fuzzy *version of the course scheduling problem in example 3.1, where each constraint is associated with a degree of preference. For instance, Professor A is assigned the lectures of the course and she prefers to give four lectures. Dr. B is assigned the practical sessions and he prefers to give three. Finally, Dr. C is assigned the tutorials and prefers to give three of them. Table 3.3 describes the course session assignments along with their preference degree. The rest of the predefined constraints in the problem are classified as hard constraints, hence they should be completely satisfied, i.e. their satisfaction degree provided in the* fuzzy *CSP model is assigned a value of* 1. *Fig. 3.7 illustrates two solutions to the problem with different satisfaction degree. Note that* fuzzy *operations select the minimal degree of satisfaction when constraints in a* fuzzy *relation are in conjunction. Accordingly the first solution yields a* fuzzy *degree of satisfaction equal to* 0.8. *During the search the best solution has the maximal satisfaction degree.*

LP implementations

***Fuzzy* LPs** [Rommelfanger (1996)] define *fuzzy* coefficients on constraints and/or objective function. In order to reduce the cost of gathering the exact information, they solve the problem iteratively. At the beginning a 'compromise solution' is perceived then it is further improved in subsequent steps. They work on finding a 'compromise solution' which iteratively realizes better solutions to the *fuzzy* problem. Variations of

		Sums of Tuples Assigned					
		1	2	3	4	5	*Otherwise*
lectures	$\sum_{i=1}^{3} x_{i1}$	0.4	0.6	0.8	1	0.8	*0*
practicals	$\sum_{i=1}^{3} x_{i2}$	0.6	0.8	1	0.8	0.7	*0*
tutorials	$\sum_{i=1}^{3} x_{i3}$	0.6	0.8	1	0.8	0.7	*0*

Table 3.3: Production scheduling problem: fuzzy *constraints and their satisfaction degrees*

Constraint	*Assignment Sum*	*Sat Degree*
Sessions per Day	2, 4, 4	1.0
Total Sessions	10	1.0
Lectures	3	0.8
Practicals	3	1.0
Tutorials	4	0.8
Overall Satisfaction Degree: 0.8		

Type

Days i \ j	*L(1)*	*P(2)*	*T(3)*
1	1	0	1
2	2	1	1
3	1	2	1

Constraint	*Assignment Sum*	*Sat Degree*
Sessions per Day	2, 4, 4	1.0
Total Sessions	10	1.0
Lectures	4	1.0
Practicals	3	1.0
Tutorials	3	1.0
Overall Satisfaction Degree: 1.0		

Type

Days i \ j	*L(1)*	*P(2)*	*T(3)*
1	1	0	1
2	1	1	0
3	0	2	4

Fig. 3.7: Course scheduling problem: two solution instances with different satisfaction degree

the problem representation can define a combination of crisp and soft constraints. The *fuzzy* LP formulation depends on an 'extended addition' that is based on the 'Yager's parametrized t-norm' and the *fuzzy* LP can be in the form:

$$\widetilde{C}_1 x_1 \oplus \widetilde{C}_2 x_2 \cdots \oplus \widetilde{C}_n x_n \rightarrow \widetilde{Max}$$
$$s.t. \widetilde{A}_{i1} x_1 \oplus \widetilde{A}_{i2} x_2 \cdots \oplus \widetilde{A}_{in} x_n \widetilde{\leq} \widetilde{B}_i, \quad i = 1, \ldots, m,$$
$$x_1, x_2, \ldots, x_n \geq 0.$$

Where $\widetilde{A}_{ij}, \widetilde{B}_i, \widetilde{C}_j$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$ are defined as *fuzzy* sets in $\mathcal{R}$. The $\oplus$ is a *fuzzy* 'extended addition' and $\widetilde{\leq}$ is the *fuzzy* inequality.

Example 3.9. *Recall the production scheduling problem in example 3.2, constraint coefficients in the* fuzzy *version of the problem are defined by* fuzzy *membership function.*

For instance:

$$\hat{a}_{11} = \{1 : 1/3, 2 : 1, 3 : 1/2\}, \hat{a}_{12} = \{5 : 1/3, 6 : 3/4, 7 : 1\}$$
$$\hat{b}_1 = \{149 : 1/2, 180 : 1, 211 : 2/3\}, \hat{b}_2 = \{138 : 1/3, 162 : 3/4, 185 : 1\}$$

where $\{v : p\}$ *represents the value and its possibility. The problem becomes:*

$$min 2x_1 + 3x_2 + EA(s^T \hat{g}^T)$$
$$s.t. x_1 + x_2 \leq 100, x_1, x_2 \geq 0.$$

To simplify let $U_\alpha = S^T(\max\{\hat{b} - \hat{A}, 0\})$*,* $M(U_\alpha)$ *is the average value of the uncertainty at a satisfaction level* α*, and* $EA(\hat{u})$ *is the mean of* $M(U_\alpha)$ *for all* α*-levels. Let* $s^T = [7, 12]^T$*,* $\hat{g}^T = [\hat{g}_1, \hat{g}_2]^T$*,* $\hat{g}_1 = \max\{\hat{b}_1 - \hat{a}_{11}x_1 - \hat{a}_{12}x_2, 0\}$ *and* $\hat{g}_2 = \max\{\hat{b}_2 - 3x_1 - 3x_2, 0\}$*. The problem defines* α*-levels over the different membership functions defined on variable coefficients. Solution to the problem is the set of all possible combinations of the piecewise* α*-levels, each is delt with separately as a simple crisp LP program. The* α*-level sets for* $\hat{b}_1 - \hat{a}_{11}x_1 - \hat{a}_{12}x_2$*:*

$$\alpha \in [0, 1/3] \Rightarrow M(U^1_\alpha) = 180 - 2x_1 - 6x_2$$
$$\alpha \in (1/3, 1/2] \Rightarrow M(U^1_\alpha) = 180 - 2.5x_1 - 6.5x_2$$
$$\alpha \in (1/2, 2/3] \Rightarrow M(U^1_\alpha) = 195.5 - 2x_1 - 6.5x_2$$
$$\alpha \in (2/3, 3/4] \Rightarrow M(U^1_\alpha) = 180 - 2x_1 - 6.5x_2$$
$$\alpha \in (3/4, 1] \Rightarrow M(U^1_\alpha) = 180 - 2x_1 - 7x_2$$

and for $\hat{b}_2 - 3x_1 - 3x_2$

$$\alpha \in [0, 1/3] \Rightarrow M(U^2_\alpha) = 161.67 - 3x_1 - 3x_2$$
$$\alpha \in (1/3, 3/4] \Rightarrow M(U^2_\alpha) = 173.5 - 3x_1 - 3x_2$$
$$\alpha \in (3/4, 1] \Rightarrow M(U^2_\alpha) = 185 - 3x_1 - 3x_2$$

The objective function can be rewritten as :

$$
\begin{aligned}
2x_1 + 3x_2 + 7 \int_0^{\frac{1}{3}} \max\{180 - 2x_1 - 6x_2, 0\} d\alpha \\
+7 \int_{\frac{1}{3}}^{\frac{1}{2}} \max\{180 - 2.5x_1 - 6.5x_2, 0\} d\alpha \\
+7 \int_{\frac{1}{2}}^{\frac{2}{3}} \max\{195.5 - 2x_1 - 6.5x_2, 0\} d\alpha \\
+7 \int_{\frac{2}{3}}^{\frac{3}{4}} \max\{180 - 2x_1 - 6.5x_2, 0\} d\alpha \\
+7 \int_{\frac{3}{4}}^{1} \max\{180 - 2x_1 - 7x_2, 0\} d\alpha \\
+12 \int_0^{\frac{1}{3}} \max\{161.67 - 3x_1 - 3x_2, 0\} d\alpha \\
+12 \int_{\frac{1}{3}}^{\frac{3}{4}} \max\{173.5 - 3x_1 - 3x_2, 0\} d\alpha \\
+12 \int_{\frac{3}{4}}^{1} \max\{185 - 3x_1 - 3x_2, 0\} d\alpha
\end{aligned}
$$

The system yields a unique optimal solution with an objective value $R^* = \$139.37$, $x_1^* = 45.63$ *and* $x_2^* = 16.04$

Output solutions

The authors in [Mauris, Berrah, et al. (2000); Mauris et al. (2001); Van De Ree and Jager (1993); Urbanski and Wsowski (2003); Ferrero and Salicone (2004)] provided different propagation techniques for typical possibilistic distributions using *fuzzy* sets. Equivalent to the joint distribution function in probabilistic domains, the key propagation technique in *fuzzy* models is the T-norm operations. It is an extension of interval-based min/max operations that is performed on each alpha-cut interval in the *fuzzy* possibilistic approximated distribution. Result on the discrete point level is the union of all operations exerted per granule.

Benefits

- Operations on data using the possibilistic theory are simpler than other available techniques which use the probabilistic reasoning.

- Exert the ‘Dominance possibility‘ effect; i.e. it is a pessimistic approach provided that output result is an upper bound of the solution set.

- Unlike interval-based approach, the possibilistic distribution provides worst and best case scenarios represented respectively by the support and the kernel of the possible solution set

Drawbacks

- In the case of a full knowledge of the input data probability distribution, possibility technique is less adequate in expressing the problem under consideration when it is compared to probabilistic approaches

- Propagation is based on interval-based framework (worst case scenario), it is typical to interval-based approaches in its computation and reasoning, and it doesn't have an effect on the uncertainty degree. The solution space is large and further exhaustive search needs to take place.

- The lack of complete knowledge representation yields inaccurate realization of the solution space

3.3.3 Reliable/Robust paradigm

General Description

Data uncertainty in reliable paradigms is bracketed by convex structures: interval or ellipsoidal. This structure in turn assigns safe-guards for erroneous and deviated measurements to guarantee that all data is enclosed in the model. However, it does not provide any knowledge about the population distribution of the input measured data. Reasoning using convex modeling approaches is constructed on the extreme points of the structure, unlike exhaustive point computation found and exerted in other paradigms.

Input data

Ideal model when input data uncertainty comes from ill-defined or erroneous measurement. Data is represented by a convex set which embraces all provided information within its end-points.

CP implementations

Numerical CSP [Benhamou and Older (1997)] uncertain data is embraced within intervals. The framework provides an inference mechanism for interval constraints defined over linear and non-linear systems. It is mainly introduced to solve problems of real-intervals by means of interval-based techniques. In this model the propagation is exerted only on the bounds enveloping real interval domains. Output realizations are in turn interval bounds guaranteed to contain the actual solution of the problem in hand.

Certainty closure [Yorke-Smith and Gervet (2009)] associates uncertainty to constraint coefficients. Interval coefficients in this model are used to bracket the ill-defined data given in the problem definition. This is exerted by shaping a Normal distribution over the measurement (here dealt with as the Normal distribution average). Interval bounds are then assigned the maximum and minimum values to include the majority of the Normal distribution data population. This framework brings together modeling and solving methodologies from the linear programming into the CP paradigm to provide reliable and efficient approaches for uncertain constrain problems.

Example 3.10. *Consider the course scheduling CSP, in example 3.1, the set of constraints defined by the certainty closuse can be reformulated as:*

$$\forall i \sum_{j=1}^{3} w_j x_{ij} \geq 2 \tag{3.10}$$

$$\forall j \sum_{i=1}^{3} w_j x_{ij} \in \{1, \ldots, 5\} \tag{3.11}$$

$$\sum_{i=1}^{3} \sum_{j=1}^{3} w_j x_{ij} = t \tag{3.12}$$

Where $\forall j w_j$ and t are uncertain coefficients each is defined over a real-interval domain. Solution to the problem is a covering set of all possible realizations.

LP implementations

Interval linear programming . Research has thoroughly studied variations of the Interval Linear Programming (ILP) due to their inexpensive computations when employed in large scale systems. Ning and Kearfott (1997) work on the computation of the interval Guassian elimination techniques. Rohn (1993), Hansen (1980),Hansen (1992) extract the inverse of the coefficient matrix when its components are represented by real intervals. Suprajitno and Mohd (2010) and Ramesh and Ganesan (2011) generalized the Simplex method to incorporate real interval computations. Beaumont (1998), Jansson (1997), Oettli (1965) and Aberth (1997) derive inclusion bounds over the solution set. Chinneck and Ramadan (2000) incorporate the uncertainty in the objective function calculations with the hull. The general ILP model can be rewritten as:

$$\min Z = \sum_{j=1}^{n} [\underline{c_j}, \overline{c_j}]$$

subject to

$$\sum_{j=1}^{n} [\underline{a_{ij}}, \overline{a_{ij}}] x_j \geq [\underline{b_i}, \overline{b_i}] \quad \forall i = 1, \ldots, m \tag{3.13}$$

Generally, interval linear computations are easily conducted but they do not guarantee the tightest bounds on the resulting solution set.

Robust optimization [Ben-Tal and Nemirovski (2000) and Mulvey, Vanderbei, and Zenios (1995)] Random components are represented by means of variance reduction techniques. Uncertainty, in this case, is encapsulated within a convex ellipsoidal set which approximates the problem. The model can be rewritten as:

$$\min[\sup_{c,A,b\in\mathcal{U}} c^T x : (Ax \leq b)\forall(c,A,b) \in \mathcal{U}]$$

Where data for (c, A, b) are not known for certain. The model, with 'boundedness assumptions', searches for a solution to the best objective that satisfies all realizations of the constraints. The computed objective seeks the minimization of the worst case scenario.

Interval Expected Value [Thipwiwatpotjana and Lodwick (2008) and Sengupta, Pal, and Chakraborty (2001)] combine the possibility and the probability within the same constraint. Uncertainty is represented by an interval of expected value which is bounded by the smallest and the largest expected values. Intervals are then utilized as coefficients in the interval linear program.

Example 3.11. *Returning to the production planning problem, the interval expected value* $\hat{a}_{11} = [1.6667, 2.5]$ *and* $\hat{b}_1 = [169.6667, 180]$. *The LP model can be reformulated as:*

$$\begin{aligned} \min\, & 2x_1 + 3x_2 \\ s.t. & [1.6667, 2.5]x_1 + 6.1667x_2 \geq [169.6667, 180], \\ & x_1 + x_2 \geq 100, \\ & x_1, x_2 \geq 0. \end{aligned} \tag{3.14}$$

In this example the bounds on the objective function is [\$117.35, \$127.87] *and the optimal solution* $x_1 \in [33.89, 44.41]$ *and* $x_2 \in [9.51, 20.03]$ *is obtained by solving, in this case, four LP problems generated by the extreme points bounding the expected value intervals.*

Output solutions

Results of convex models are reliable and guarantee to carry-out all potential solutions of the problem in-hand. Uncertainty is represented as set of values that are enclosed between extreme points. To derive outer bounds, the model is based on an approximation that is not necessarily reversed. The output solution is a non-tight set. It is characterized to be reliable and can take any of the possible values from the closure of the decision space. Each value in the set has an equal uncertainty degree.

Benefits

- Efficient integration of interval computation methods in CP

- Ideal when partial data is used due to the overwhelming amount of information
- Deal with real data
- Enclose the uncertainty using what is known for sure about the data
- Guarantee the true problem is contained in the model hence described
- Produce robust/reliable solutions
- Efficiently derive the closure to an uncertain constraint problem
- Computationally tractable

Drawbacks

- Result in a solution set with equal uncertainty weights.
- Doesn't reflect any possible degree of knowledge, i.e. the model lacks information about the random distribution embraced by its extreme points
- Acquired solution space can be very wide lacking expressive approximation of the problem in-hand for speedy future search

3.4 Summary

In this chapter we review research efforts to reason about data in the presence of uncertainty. We started by studying the different uncertainty types found in the real-world applications. We then elaborate how research classifies those types and maps them in the conceptual world to deal with them mathematically. Explored mathematical models are adopted consequently to reason about data with uncertainty. Two paradigms were explored: CP and LP. The study of the two paradigms summarizes: 1. how uncertain data is input to the models. 2. CP and LP existing implementation techniques. 3. the property of the output solution set obtained from the reasoning process when different models from both paradigms are adopted. The CP and LP paradigms employ different mathematical concepts to handle and to reason about data with uncertainty. Those mathematical concepts can be listed as: probabilistic, possibilistic, *fuzzy* and convex models. We showcase and elaborate with examples how models from both paradigms with different mathematical concepts behave. We also clarify the advantages and drawbacks found in the literature when each concept is adopted. This study is due to the fact that our framework inherits its properties from the two paradigms: the CP and the LP. We will elaborate more about this inheritance in the rest of the chapters. The following chapter shows how we construct the intervals of the input data with uncertainty in existing convex techniques and how this construction evolved to construct the intervals representing the data with uncertainty in our proposed p-box *cdf*-intervals framework.

CHAPTER 4

CONSTRUCTING THE INTERVALS

The concept of convex modeling, as pointed out in Section 3.3.3, was coined to formalize the idea of enclosing uncertainty sets and yield reliable solutions, i.e guaranteed to contain any solution produced by any possible realization of the data [Chinneck and Ramadan (2000), Ben-Haim and Elishakoff (1995), Yorke-Smith and Gervet (2009)]. Interval coefficients have been introduced in OR and CP to specify and enclose uncertain data in order to provide reliable solutions to convex models. They are at the heart of paradigms such as robust optimization [Ben-Tal and Nemirovski (1999); Hoffman (2000)] in OR as well as mixed CSP [Fargier et al. (1996)], reliable constraint reasoning [Yorke-Smith (2004),Yorke-Smith and Gervet (2009)], and quantified CSP [Zhou, Doyle, and Glover (1996)] in CP. These paradigms specify erroneous and incomplete data using uncertainty sets that denote a deterministic and bounded formulation of an ill-defined data. To remain computationally tractable, the uncertainty sets are approximated by convex structures such as intervals (extreme values within the uncertainty set) and interval reasoning can be applied ensuring effective computations. In this chapter we focus on how convex structures were constructed in the literature. We showcase, with a running example, how the convex representation of data with uncertainty is evolved from reliable models to include additional information about the data whereabouts. Yet they remain tractable by keeping their convex properties.

4.0.1 Constructing reliable intervals

For instance, we consider the traffic volumes in the NTAP. In this problem traffic volumes are monitored in a distributed manner [Grossglauser and Rexford (2005)]. Accordingly, the measurement of one designated flow can differ by milliseconds. Output observations from the measurement process are often erroneous or incomplete due to packet loss or inaccurate deviation. Fig. 4.1 illustrates a snapshot, at a point in time, of an arbitrary network. Numbers provided on the links represent the average of readings

per flow on each link (the flow headed towards node B from node A ($V_{A\to B}$), in this snapshot, is on average equal to 204 packets).

The information available, in the problem definition, is typically point elements along with their frequencies, intensities or probability of occurrence. Out of which existing reliable approaches deduce the average and standard deviation or approximate the given data to the nearest probability distribution. The original list of observations along with their frequencies of occurrence, of the flow from node A to node B, are illustrated in Fig. 4.2.

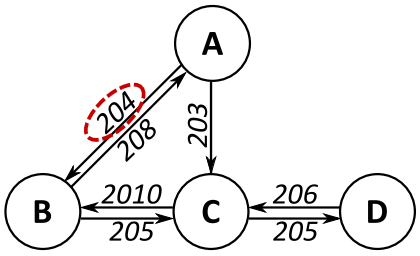


Fig. 4.1: Network of 4-nodes: a snapshot

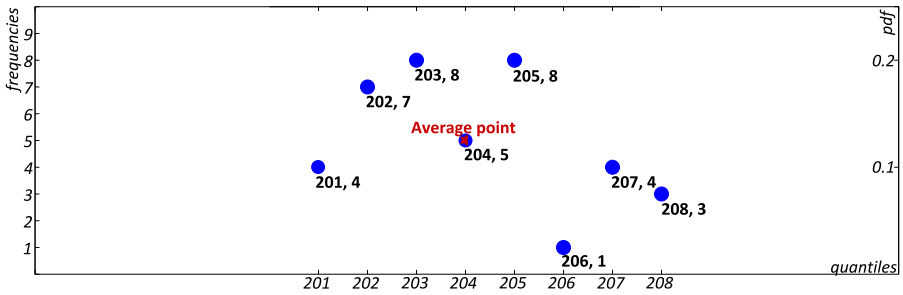


Fig. 4.2: Collected observations of $V_{A\to B}$

To represent ($V_{A\to B}$), reliable models commonly adopt one of the two approaches:

1. Store the minimum and the maximum observed values from the measurement process. In this example, they are 201 and 209 respectively (as illustrated in Fig. 4.3)

2. Derive the probability distribution from the list of values and their occurrences (Fig. 4.4). Out of this derivation, the statistical average and standard deviation can be obtained. Hence, the nearest known probability distribution (in most cases the Normal distribution Fig. 4.5) is computed. The output minimum and maximum values from this process for ($V_{A\to B}$) are respectively 197 and 211. It is obtained by deriving the confidence interval which includes 98% of the statistical data population resulting from the measurement.

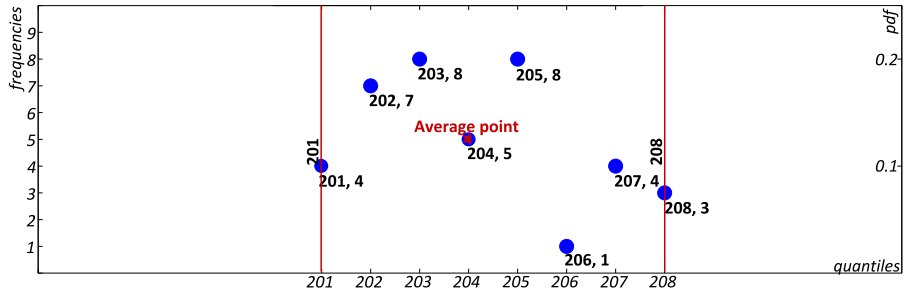


Fig. 4.3: Reliable model representation of $V_{A\to B}$

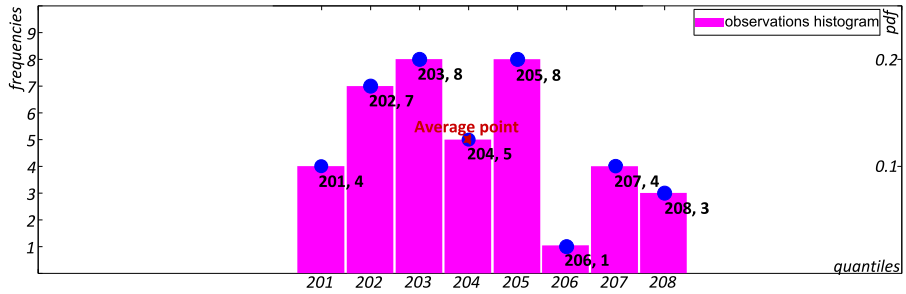


Fig. 4.4: $V_{A\to B}$: probability distribution histogram

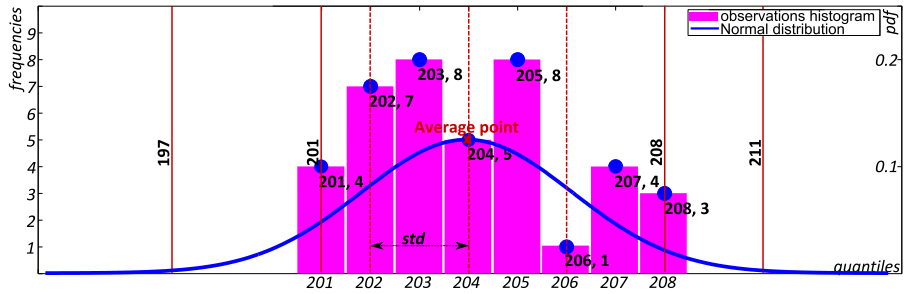


Fig. 4.5: $V_{A\to B}$: nearest Normal distribution

Example 4.1. *The convex representation of traffic volume destined to the node B from the node A is the interval* $V_{A\to B} \in [201, 208]$ *and* $[197, 211]$ *in the two commonly adopted reliable approaches: convex modeling and UCSP.*

Reliable approaches, in their convex representation, do not account for the data whereabouts which are available during the data collection. The outcome of these systems is a solution set that can be refined when more knowledge is acquired about the data population, and does not exclude any potential solution.

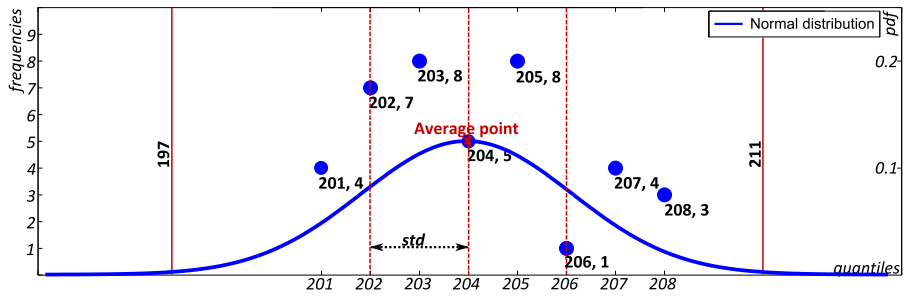


Fig. 4.6: $V_{A\to B}$: reliable model representation of Normal distribution at $\pm 3\sigma$

4.0.2 Constructing the *cdf*-intervals

Motivated by conveying the data whereabouts given in the problem definition, we introduced the *cdf*-intervals in Saad et al. (2010). The *cdf*-intervals structure extends interval data models with a second dimension: a quantitative dimension added to the measured input data. This quantitative dimension provides information about the probability distribution of the data.

We have selected the *cdf* because it has been used, for this purpose, in different models to analyze the distribution of the data whereabouts (e.g. in Gubner (2006) and Tversky and Kahneman (1992)). The *cdf* enjoys three main properties:

1. The *cdf* is a monotone, non-decreasing function, like arithmetic ordering suitable for interval computations and pruning.

2. It directly represents the aggregated probability that a quantity lies within bounds, thus, showing the confidence interval of this uncertain data.

3. It brings flexibility to the problem modeling assumptions (e.g. by bounding data by means of the derived *cdf*. In Saad et al. (2010), we constructed the *cdf*-intervals which is a convex model that approximates the data whereabouts to the nearest uniform *cdf*-distribution within a confidence interval).

Our methodology, in Saad et al. (2010), consists of building data intervals employing 2D points as extreme values. In the *cdf*-intervals model, we assume that with each uncertain data value comes its frequency of occurrence or density function. We then compute the *cdf* over this function. The concept of *cdf*-intervals proved to be capable of representing a new type of convex sets, following the concept of interval coefficients. This requires the decision variables to range over *cdf*-itervals as well. Basically, in the *cdf*-intervals framework, elements of a variable's domain are points in a 2D-space, the first dimension represents the data value, and the second is its aggregated *cdf*-value. It is defined as a *cdf*-interval specified by its lower and upper bounds.

Recall from Example 4.1, the real-intervals representation of $V_{A\to B} \in [201, 209]$ and $[197, 211]$ in the two commonly adopted reliable approaches. In order to illustrate this

interval representation in the *cdf*-intervals framework, we first need to project the observations given in the original definition of the problem onto the *cdf*. Fig. 4.7 illustrates the original probability distribution and its nearest Normal distribution projections onto the *cdf* dimension, and Fig. 4.8 illustrates the traffic volume $V_{A\rightarrow B}$ in the *cdf*-intervals. In this case, it is represented by the interval $[(201, 0.1), (207.73, 0.98)]$. In this example the value 201 is the data value in the real domain $\mathbb{R}$ and 0.1 is its *cdf* value. The *cdf*-interval representation of $V_{A\rightarrow B}$ is approximating the unknown distribution and it indicates that the real values ranges within $[201, 207.73]$, while the *cdf* ranges within $[10\%, 98\%]$.

The main idea, in Saad et al. (2010), is to show that we can preserve the tractability of convex modeling computation while enriching the uncertain data sets with a representation of the degree of knowledge available.

4.0.3 Constructing the p-box *cdf*-intervals

As shown in Fig. 4.8, the *cdf*-intervals model approximates the original probability distribution to the nearest uniform distribution. Despite the fact that this approximation roughly indicates the residence of the data population, it lacks the full encapsulation of the interval in the 2D (i.e. it is not a comprehensive representation of the observed data along with its whereabouts). This led us to further extend the *cdf*-interval model to encapsulate not only the observed data but also its unknown probability distribution Saad et al. (2014). The new 2D interval framework adopts the concept of p-box, detailed in Ferson et al. (2003), to envelop the unknown probability distribution and to ensure the enclosure of any potential solutions along with their whereabouts. We have chosen uniform distributions to exert the interval envelopment due to their tractable computation, compared to other probability distributions.

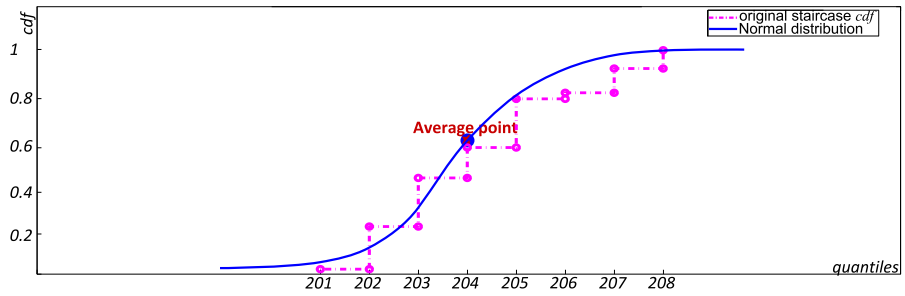


Fig. 4.7: $V_{A\rightarrow B}$: Normal distribution projection onto the *cdf* domain

The p-box augmentation further modifies the point representation. In order to fully represent a line we need at least 3 values. Fig. 4.9 depicts the p-box *cdf* representation of $V_{A\rightarrow B}$ which is represented by the p-box *cdf*-interval
$[(201, 0.1, 0.187), (208, 0.87, 0.128)]$. A point in the p-box *cdf*-intervals framework is a triplet which lists the data value in the real domain $\mathbb{R}$, the *cdf* value and the *cdf* line slope.

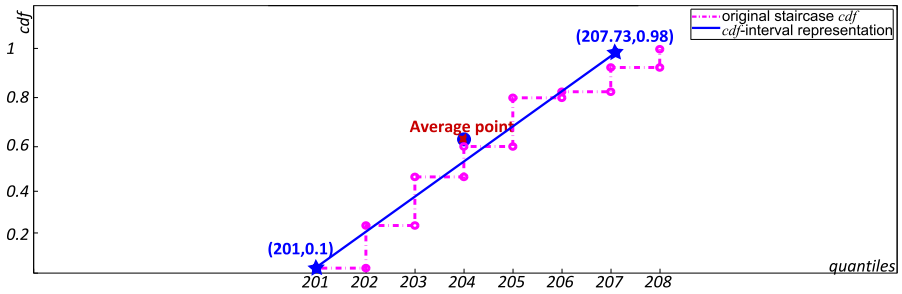


Fig. 4.8: $V_{A \to B}$: *cdf*-interval representation

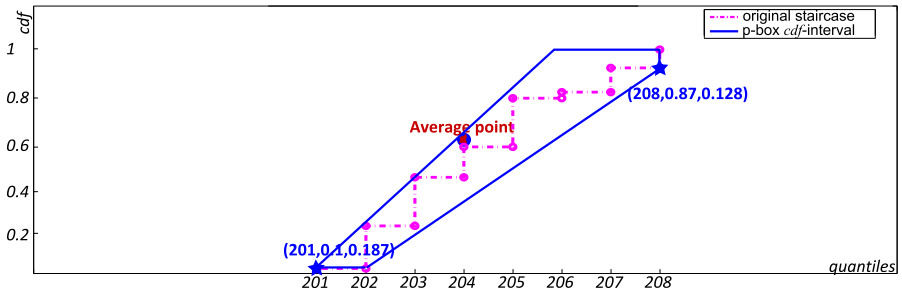


Fig. 4.9: $V_{A \to B}$:p-box *cdf*-interval representation

For the same example they are 201, 0.1 and 0.187 respectively. This p-box *cdf*-interval representation indicates the full encapsulation of the real data between the bounds 201 and 208. The envelopment of the *cdf* indicates that 201 has a chance of occurrence that cannot exceed 10% while 208 has a *cdf* value that is at least 87%. The significance of the interval representation is detailed in Chapter 5.

In summary, the *cdf*-intervals framework specifies elements of a variable domain in a 2D-space, the first dimension represents the data value, whereas the second shows its *cdf*-uniform distribution. A new domain ordering is defined within the 2D-space. This raises the question of performing arithmetic computations over such variables to infer bound consistency. As a result more information is transmitted to the realized solution sets. We augment the *cdf*-interval points by p-boxes (two bounding distributions embracing all possible distributions) to specify new domain bounds on the probability distribution along with data value bounds in the $\mathbb{R}$ domain. In this dissertation we define the constraint domain over which the calculus in this new domain structure can be performed, including the inference rules.

PART II

FRAMEWORK

CHAPTER 5

DATA REPRESENTATION

Quantitative information is usually available during the data collection process, but lost during the reasoning because it is not accounted for in the representation of the uncertain data. This information however is crucial to the reasoning process, and the lack of its interpretation yields erroneous reasoning because of its absence in the produced solution set. It is always necessary to quantify uncertainty that is naturally given in the problem definition in order to obtain robust and reliable solutions.

In this chapter we elaborate how uncertain data, probably collected in a measurement process, is represented; then input to the model. Information available in the problem definition is typically; point elements along with their frequencies of occurrence, intensities or probability distribution. The majority of the existing models, as pointed out in Chapter 3, deduce out of this information the average point and standard deviation; consequently their reasoning is based on expected values [Gum (1995),Kessel (2002) and Nielsen (2000)]. Some probabilistic models approximate the data whereabouts to its nearest probability distribution representation [Kendall et al. (1946)]; yet these models are characterized to have an expensive computation because reasoning, in this case, is exerted in a pointwise manner. Convex models offer reliability and robustness, their computations are tractable, but do not account for quantitative information [Benhamou and Older (1997),Ben-Haim and Elishakoff (1995),Beaumont (1998) and Yorke-Smith (2004)]. *Fuzzy* models have been introduced in order to combine between reliable and probabilistic models, by drawing the possibilistic distribution of the data representation [Mauris, Lasserre, and Foulloy (2000), Ferrero and Salicone (2004) and Mauris (2007)].

We first elaborate how to construct the *cdf*-intervals structure by means of a formal algorithm published in Saad et al. (2010). Then we show how this algorithm has evolved to construct the p-box *cdf*-intervals convex structure, Saad et al. (2014) and Saad (2014). The p-box *cdf*-intervals algebraic structure seeks the full encapsulation of the data along with the whereabouts observed, irrespective of the probability distribution they outline.

In this chapter, we show how a p-box *cdf*-interval bounds the observed probability distribution. The outcome of this process is the established interval; it has two uniformly distributed *cdf*-bounds. Uniform distributions are chosen to ease the computation of storing and reasoning about data. We also compare this new formulation with input coefficients described by reliable, probabilistic and *fuzzy* models Saad et al. (2014).

5.1 Establishing the Confidence Interval

Data obtained from empirical measurement often follows an unknown probability distribution [Gum (1995)]. This data is produced by an instrument, characterized to have components of imprecision. In the general case, it is evaluated in a discrete manner because measurement is performed at specific time points. Computed *cdf*-distribution, in this case, draws a staircase shape and the actual continuous distribution between the measured points is unknown [Smith and La Poutre (1992)].

Given a set of n data series obtained in a measurement process of a population m, $m \neq n$, or possibly randomly generated, we establish a generic construction of the confidence possibilistic/probabilistic interval as follows:

1. Data is collected and n quantiles (data values) are distinguished, each is represented by x_i.

2. The *pdf* of the genuine observations is derived from $\frac{(x_i \text{Freq}_i)}{\sum_1^n x_i \text{Freq}_i}$, where Freq_i is the number of times a quantile x_i is observed.

3. The average $\bar{x} = \frac{x_1 \text{Freq}_1 + \cdots + x_n \text{Freq}_n}{\sum_1^n x_i \text{Freq}_i}$ and their standard deviation $\sigma = \sqrt{\frac{1}{n} \sum_1^n (x_i - \bar{x})^2}$ are computed. Note that other probabilistic tools like the variance can be derived when additional knowledge on the probability distribution is needed.

4. The possibilistic/ probabilistic distributions are derived from the average and the standard deviation values. Based on the Gum (1995) any probability distribution (parametric/non-parametric) is typically approximated to the nearest Normal distribution.

5. Computation and reasoning are based on the derived distributions since pointwise operations are computationally expensive.

5.1.1 The Measurement Process

As a running example, Figure 5.1 illustrates the data collection process which is exerted by means of a measuring instrument. Values, defined as quantiles, are collected along with their frequencies of occurrence. Reliable models can derive the interval representation of this specific sample by storing the minimum and maximum observed values;

in this example the values are 1 and 8 respectively. Consequently, the produced robust interval is signified by two bounding points in the real domain $\mathbb{R}$, as $[1, 8]$.

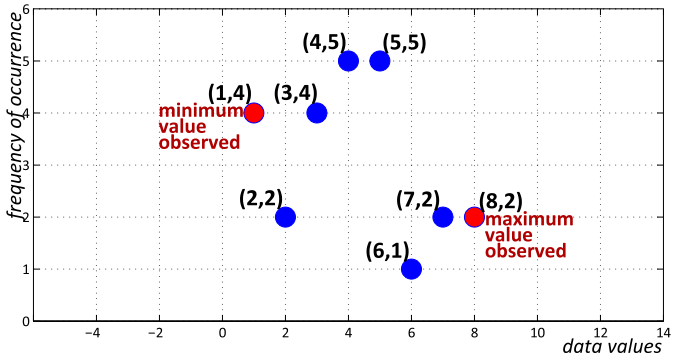
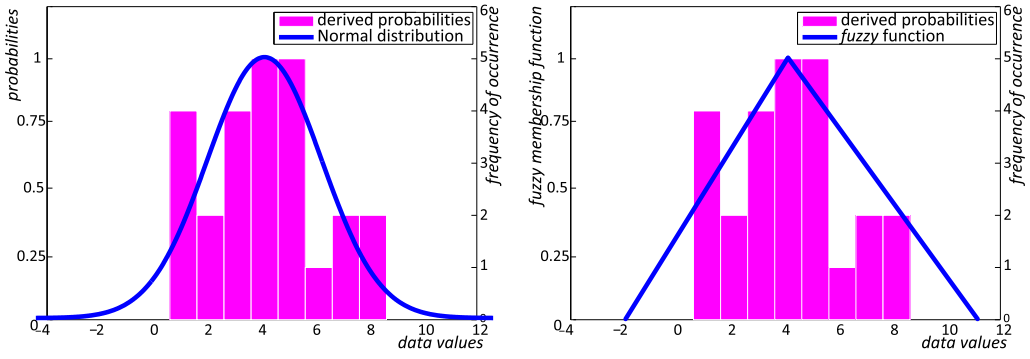


Fig. 5.1: The data observation of the measurand X

5.1.2 Computing the probability distribution

Given the observed data along with its frequency of occurence, we can obtain the density function, described in Definition 2.3, and derive the average and standard deviation of the sample data. The nearest Normal probability distribution and *fuzzy* membership function are consequently derived and illustrated in Figure 5.2. Figure 5.2 shows how the data whereabouts is enveloped within the produced distributions: probabilistic or possibilistic. Then, they are input as coefficients to their designated models. It is worth noting that the Normal distribution representation is given by its average and standard deviation, in this case 4 and 2.11 respectively; while the *fuzzy* membership representation is given by a triplet $[-2, 4, 11]$ if it is a triangular *fuzzy* presentation, or a quartet $[-2, 4, 5, 11]$ if it is a traperzoidal presentation.



(a) (b)

Fig. 5.2: Deriving the probability distribution of the measurand X: (a) Normal distribution, (b) *fuzzy* membership function

5.1.3 Projecting Distributions onto the *cdf*-domain

Recall from Section 2.1 the *cdf* distribution is an aggregated value of the density function; i.e. the *cdf* value of an element is its density value in addition to the sum of densities of all preceding elements. Hence the *cdf* allows us to keep information about the *pdf* in an aggregated manner. Figure 5.3 illustrates the computed *cdf* distribution of the same sample data. This *cdf* has a staircase shape because of the discrete characteristic of the measurement process. Figure 5.4 depicts the projection of the Normal distribution and the *fuzzy* membership function onto the *cdf*-domain in order to visualize accuracy of the studied distribution to encapsulate the observed data whereabouts. Clearly both representations are based on approximation and lack precise point fitting.

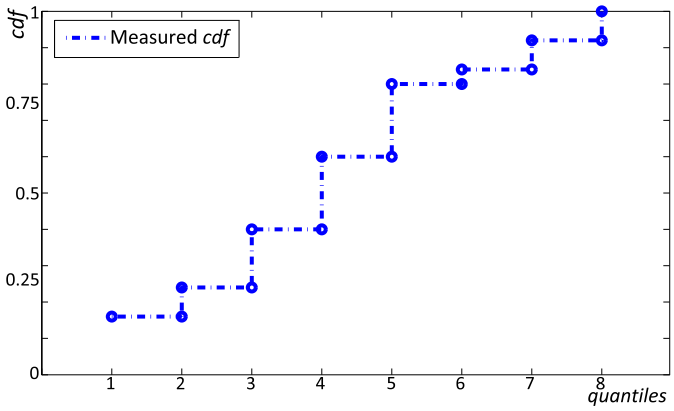
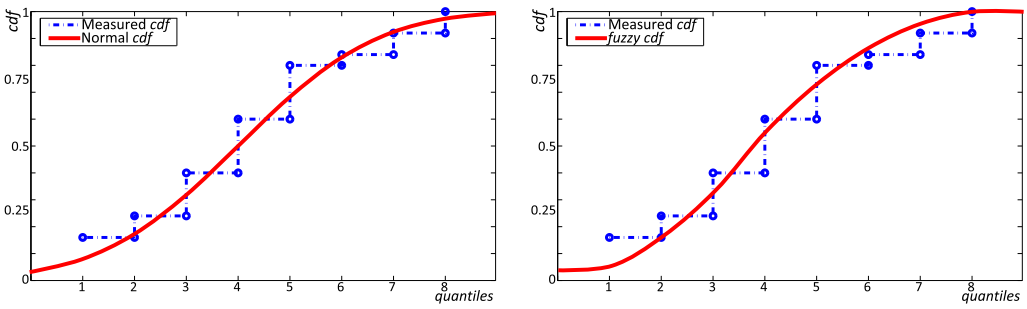


Fig. 5.3: Constructing the *cdf*-distribution of X



(a) (b)

Fig. 5.4: Constructing the Normal and *fuzzy* *cdf*-distributions of the measurand X: (a) Normal *cdf*-distribution, (b) *fuzzy* *cdf*-distribution

5.1.4 Constructing the *cdf*-intervals

Consider that in the set of data illustrated in Fig. 5.1, we construct the *cdf*-interval as detailed in Algorithm 1. The algorithm runs in $O(n)$ where n is the number of distinct

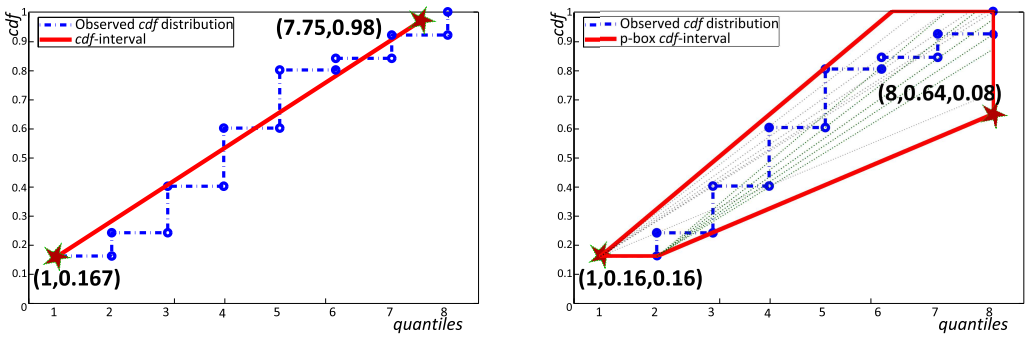
procedure *ConstructCDFIntervalBounds*$(m, Arr[n], \text{Freq}[n])$

1: $\text{cdf}[1] \leftarrow \text{Freq}[1]/m$
2: **for** $i = 2$ to n **do**
3: $\quad \text{cdf}[i] \leftarrow (\text{Freq}[i]/m) + \text{cdf}[i-1]$
4: $i = 1$,
5: **while** $(\text{Freq}[i] \leq 0.02)$ **do**
6: $\quad i \leftarrow i + 1$
7: lowerbound $\leftarrow (Arr[i], \text{cdf}[i])$
8: upperbound $\leftarrow (\text{cdf}^{-1}[0.98], 0.98)$

Algorithm 1: data interval bounds construction

values in the data set. It receives three parameters: the size of the data population, m, a sorted list (ascending order) of the distinct measured data, and a list of their corresponding frequencies. Both lists are of the same size n. The algorithm first computes the *cdf* in a cumulative manner. The turning points are then extracted by recording the data values that have a *cdf* greater than, or equal to, 2%, and the value with *cdf* equal to 98%; the two values were chosen such that they are distant from the average by $\pm 3\sigma$. This algorithm was previously published in Saad et al. (2010).

Fig.5.5 illustrates the *cdf*-interval construction of the sample data given in the running example of this chapter. As listed in Table 5.1, the data set size is $n = 8$, the population size is $m = 25$, $Arr[n] = [1, 2, 3, 4, 5, 6, 7, 8]$, and the corresponding frequencies $Freq[n] = [4, 2, 4, 5, 5, 1, 2, 2]$. The computed *cdf*-interval has the following bounds $[(1, 0.16), (7.75, 0.98)]$. Clearly the resulting interval is an approximation and lacks the full encapsulation of the measured data whereabouts.



(a) (b)

Fig. 5.5: Constructing the p-box *cdf*-intervals bounds of X

	X					
	data	**freq**	***pdf* value**	***cdf***	**(CDF[n]-CDF[1])/ (Arr[n]-Arr[1])**	**(CDF[n-1]-CDF[1])/ (Arr[n]-Arr[2])**
	1	4	0.16	0.16		
	2	2	0.08	0.24	0.08	
	3	4	0.16	0.4	0.12	0.08
	4	5	0.2	0.6	0.15	0.12
	5	5	0.2	0.8	0.16	0.15
	6	1	0.04	0.84	0.14	0.16
	7	2	0.08	0.92	0.13	0.14
	8	2	0.08	1	0.12	0.13
lb	**1**			**0.16**	**0.16**	
ub	**8**			**0.64**		**0.08**

Table 5.1: Preprocessing steps for modeling collected data

5.1.5 Constructing the p-box *cdf*-intervals

Algorithm 2 shows the p-box *cdf*-interval construction steps for the same example. Two parameters are taken into consideration: Arr[n] is an array of n distinct elements (observed and sorted); whereas the second parameter is cdf[n]; the set of their computed *cdf* values.

The two arrays, together, form the staircase function shape with quantiles stored in Arr[] and *cdf* values stored in *cdf*[]. Note that a staircase function defines as set of constant values *cdf*[i] over a set of intervals [Arr[i], Arr[$i+1$]] $\forall i < n$ [Smith La Poutre-Smith La Poutre]. Accordingly, the set of upper and lower bounding points forming the staircase function are {[Arr[i],*cdf*[i]]} $\forall i, 1 \leq i \leq n$ and {[Arr[$i+1$],*cdf*[i]]} $\forall i, 1 \leq i < n$ respectively. The aim of the algorithm is to envelop those observed points with the highest and lowest possible average probabilistic step increase from the first quantile interval of the staircase function. Issuing the slopes from this specific interval is sufficient to compute the bounds due to the *cdf* monotonic property 2.7.

Recall that, from property 2.8, a *cdf* slope is the average step value that indicates how the probability distribution increases. Algorithm 2 starts by computing $2n$ slopes issued from the 2 points, specified as (Arr[1],*cdf*[1]) and (Arr[2],*cdf*[1]), and destined to all other points in the *cdf*-domain. This is to calculate the list of possible average step values between the observed staircase bounding points. Slopes are then sorted to extract the steepest line and the flatest line. The geometric area under the line, computed by the integral, determines the dominated (dominating) *cdf* distribution with maximum (minimum) area as indicated in Section 2.3. Accordingly, the lower bound in the *cdf* domain is the fastest increasing line slope and issued from the 1^{st} quantile observation, and vise versa the upper bound is the least increasing line slope and issued from the maximum quantile value having the minimum observed *cdf* value. This is to guarantee the full encapsulation of all the measured data between the two bounding lines. Upper

procedure *ConstructPBOXCDFIntervalBounds*($Arr[n], CDF[n]$)

// compute the list of slopes between the observed points in the *cdf*-domain
    $j \leftarrow 0$
for $i = 2$ to n **do**
        $\text{slopes}_{lb}[j] \leftarrow (\text{cdf}[i] - \text{cdf}[1])/(\text{Arr}[i] - \text{Arr}[1])$
        $\text{slopes}_{ub}[j] \leftarrow (\text{cdf}[i-1] - \text{cdf}[1])/(\text{Arr}[i] - \text{Arr}[2])$
// find the most increasing lower bound slope $O(nlog(n))$
    $S_{xl} \leftarrow getmax(\text{slopes}_{lb})$
// find the least increasing upper bound slope $O(nlog(n))$
    $S_{xu} \leftarrow getmin(\text{slopes}_{ub})$
// get the lower bound point
    $a \leftarrow \text{Arr}[1]$
    $F_a \leftarrow \text{CDF}[1]$
    $S_a \leftarrow S_{xl}$
// get the upper bound point by projecting the maximum observed quantile
//onto the upper bound slope
    $b \leftarrow \text{Arr}[n]$
    $F_b \leftarrow S_{xu} * (\text{Arr}[n] - \text{Arr}[2]) + \text{CDF}[1]$
    $S_b \leftarrow S_{xu}$
// return cdf-interval
    $[(a, F_a, S_a), (b, F_b, S_b)]$

Algorithm 2: data interval bounds construction

and lower *cdf* uniform distributions are depicted by the red lines in Fig. 5.5 (b), and accordingly we can deduce the p-box *cdf*-interval as published in Saad et al. (2014).

Theorem 5.1. *Algorithm 2 is correct with time complexity* $O(nlog(n))$.

Proof. Computing the $2n$ slopes is exerted by applying the equations:
$(cdf[i] - cdf[1])/(\text{Arr}[i] - \text{Arr}[1])$ and $(cdf[i-1] - cdf[1])/(\text{Arr}[i] - \text{Arr}[2])$. Each of the specified equations is linear with $O(1)$ time complexity. Indexing both arrays: quantiles $Arr[]$ and $cdf[]$ starting from the 1^{st} element up to n (the size of the array is the number of distinct observations which is practically finite). Hence computing the slopes loops over the elements of the array with a time complexity of $O(n)$. The list of calculated slopes is then sorted in $O(nlog(n))$ time complexity. Accordingly, we can conclude that the algorithm is of $O(nlog(n))$ which is the time taken to sort the list of computed slopes. □

Fig.5.5 illustrates interval data construction given by the example displayed in Table 5.1. The data set size $n = 10$; computed p-box *cdf*-interval has the following bounds $[(1, 0.16, 0.16), (8, 0.64, 0.08)]$ for X.

5.2 Interpretation of the confidence interval I

Consider the practical meaning of interval **I** that we have sought to obtain. For a given interval of points specified by $\mathbf{I} = [p_a, p_b]$, p_a and p_b are the extreme points which bound the *cdf*-interval and $\mathbf{I} = [p_a, p_b]$, p_a and p_b are the extreme points which bound the p-box *cdf*-interval.

5.2.1 The *cdf*-interval I =$[p_a, p_b]$

This interval is built according to two main sources of information: 1) the monotonic and non-decreasing properties of the *cdf* curve to account for the degree of knowledge, 2) the extreme turning points over such a curve. Recall that the *cdf*-curve indicates the aggregated distribution function of a data set. Plotting a point on this curve tells us what are the chances that the actual data value lies on or before this point. The extreme turning points, we have sought, are those points distant from the average by $\pm 3\,\sigma$; This correponds to 99.7% of the total population when it is following a Normal distribution. It is also important to note the effectiveness of using the *cdf* as an indicator of the degree of knowledge. Given the measurement of data p_x, such that $p_x = (x, F_x^p)$ is any point, and due to F^p monotone non-decreasing property, we have the following:

$$a \leq x \leq b, \quad F_a^p \leq F_x^p \leq F_b^p$$

Given that $p_a = (a, F_a^p)$ and $p_b = (b, F_b^p)$

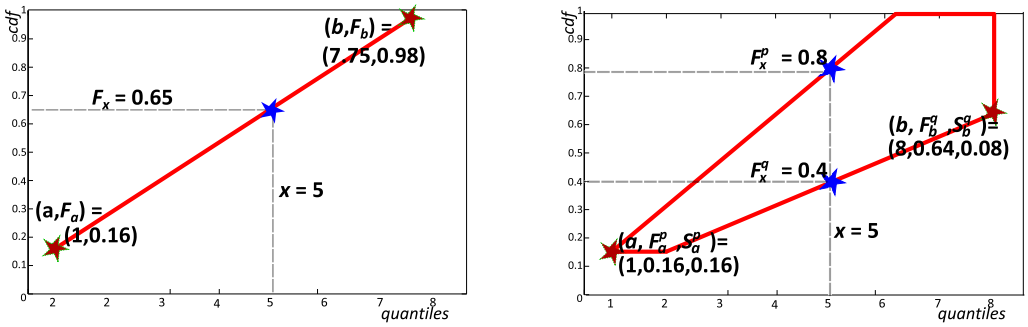
Definition 5.1. *F_x^I is the projected approximated cdf value of p_x onto F^I (the cdf associated to the interval), we will denote $p_x \in \boldsymbol{I}$ as $p_x = (x, F_x^I)$ for any point lying within the **I** interval bounds such that:*

$$a < x < b, \quad F_x^I = \frac{F_b^I - F_a^I}{b - a}.(x - a) + F_a^I \tag{5.1}$$

Property 5.1. *$F_a^p = F_a^I$ and $F_b^q = F_b^I$*

It is worth noting that this projection operation is linear because the F_x^I is derived from the line equation of the *cdf* curve; hence, it takes $O(1)$ to compute the *cdf*-value of a point located within the *cdf*-interval bounds.

Example 5.1. *Fig. 5.6 (a) illustrates the computation of F_x^I. We have $\boldsymbol{I} = [(1, 0.16), (7.75, 0.98)]$. Given a data value $x = 5$ we compute its cdf $F_x^I = 0.65$, and obtain the point $p_x = (5, 0.65)$.*



(a) (b)

Fig. 5.6: Linear approximation within $\mathbf{I} = [p_a, p_b]$ (a) *cdf*-interval bounds and (b) p-box *cdf*-interval bounds

5.2.2 The p-box *cdf*-interval I =$[p_a, q_b]$

One can see that this interval approach does not aim at approximating the curve but rather enclosing it in a reliable manner. The complete envelopment is exerted by means of the *cdf*-bounds; which are depicted by the red curves in Fig. 5.5. It is impossible to find a point that exists outside the formed interval bounds. The *cdf* bounds are chosen to have a uniform distribution because of its monotonic. Each bound is represented by a line with a slope issued from one of the extreme quantiles. Storing the full information of each bound is sufficient to restore the designated interval assignment. Each bound, in turn, is denoted by a triplet point representation, in the 2D space, that guarantees the full information on; the extreme quantile value observed; the *cdf*-line issued from this observed value; and the degree of steepness formed by this line. The slope of the uniform *cdf*-distribution indicates how the probabilistic values accumulate for successive quantiles on the line. Accordingly, the p-box *cdf*-interval point representation:

$p_a = (a, F_a^p, S_a^p)$ and $q_b = (b, F_b^q, S_b^q)$.

Definition 5.2. *S_x^p is the slope of a given cdf-distribution; it signifies the average step probabilistic value. For a given uniform cdf-distribution*

$$S_x^p = \frac{F_b - F_a}{b - a}, \forall a \leq x \leq b \tag{5.2}$$

The average step value, denoted as S_x^p, derives the probabilistic values of consequent quantiles on the real domain $\mathbb{R}$.

Plotting a point p_x within the p-box *cdf*-interval deduces bounds on its possible chances of occurrence.

Definition 5.3. *F_x^I is the interval of values obtained when p_x is projected onto the p-box cdf bounds. For a point $p_x \in \mathbf{I}$ denoted as $p_x = (x, F_x^p, S_x^p)$*

$$a < x < b, \text{ and } F_b^{q'} \geq F_x^I \geq F_a^{p'} \text{ and } S_a^p \geq S_x^p \geq S_b^q \tag{5.3}$$

$F_a^{p'}$ and $F_b^{q'}$ are the possible maximum and minimum *cdf* values p_x can take; both are computed by projecting the point p_x onto the *cdf* distributions passing through points a and b respectively. They are derived using the following linear projections that are computed in $O(1)$ complexity:

$$F_a^{p'} = min(S_a^p(x-a) + F_a^p, 1) \quad \text{and} \quad F_b^{p'} = max(F_b^p - S_b^p(b-x), 0)$$

The equation above guarantees the probabilistic feature of the *cdf*-function by restricting its aggregated value from exceeding the value 1 and having negative values below 0.

Example 5.2. *Fig. 5.6 (b) illustrates the computation of* F_x^I. *We have* $\boldsymbol{I} = [(1, 0.16, 0.16), (8, 0.64, 0.08)]$. *Given a data value* $x = 5$ *we compute its* cdf*-bounds* $F_x^I = [0.4, 0.8]$. *This means that the possible chance of the value to be at most* 5 *is between* 40% *and* 80%, *with an average step probabilistic value between* 8% *and* 16%.

We can conclude from the above examples that the new algebraic structure adds up quantitative information to real intervals. The *cdf*-interval yields one approximated probabilistic value for a given quantile; while the p-box *cdf*-interval representation produces an interval of *cdf* values which encapsulates all chances the designated quantile can possibly occur.

CHAPTER 6

THEORETICAL FRAMEWORK

As demonstrated in Chapter 5, bounds, which specify uncertain data, are constructed in a preprocessing step resulting from the measurement. Established intervals are utilized as input coefficients to the constraint system. The constraint system, in turn, utilizes this additional information to produce a solution set, as opposed to a solution point. The variables thus denote intervals within the structure and constraint processing needs to be extended to perform arithmetic operations within the novel algebraic structure. The *cdf*-interval approach extends real interval arithmetic. It adds a second dimension to each uncertain value, requiring us to define a new ordering among points in a two dimensional space, together with new inference rules.

6.1 Notations

Throughout this thesis we assume that data takes its value in the set of real numbers $\mathbb{R}$, denoted by *a*,*b*,*c*. Data points are denoted by p, q, r possibly subscripted by a data value. p_x is the point p that has a quantile x, i.e. x is its mapping value on $\mathbb{R}$. Variables are denoted by X, Y, Z and intervals of elements from the specified domains are denoted by I, J, K.

6.2 Defining the *cdf* lattice in the 2D-space

The lattice sketches the fundamental features of a formal language description. Lattice, as a poset, is a general algebraic structure which defines for every two elements a partial ordering, a greatest lower bound (*glb*) and a least upper bound (*lub*). The lattice definition of ordering calculus and arithmetic operations are exerted on the computational domain which is the domain of discourse that maps a variable to a measureable quantile together with its knowledge in the 2D-space.

Let $\Sigma_{\mathcal{U}}$ be the signature of the *cdf*-interval formal language. $\Sigma_{\mathcal{U}}$= {$\preccurlyeq_{\mathcal{U}}$, $glb_{\mathcal{U}}$, $lub_{\mathcal{U}}$, $\odot$, $\in_{[p_a,q_b]}$}. $\mathcal{U}$ is subscripted by '1' to characterize the *cdf*-intervals with one approximated *cdf* distribution and by '*b*' to signify the p-box *cdf*-intervals domain of discourse. The $\odot$ is a binary arithmetic operation over the points defined in the 2D-space. $\in_{[p_a,q_b]}$ is the predicate symbol interpreted as the ordering operator of a point p_x as $p_a \preccurlyeq_{\mathcal{U}} p_x \preccurlyeq_{\mathcal{U}} q_b$. The set of *cdf*-interval points extends the Herbrand universe. They express the data and its knowledge in a 2D-space. Points are represented by tuples in the $\mathcal{U}_1$ and by triplets in the $\mathcal{U}_b$ domain of discourse.

6.2.1 Data point structure

The *cdf*-interval computation domain is defined in the 2D-space. The notion of a *cdf* value $F_X(x)$ is associated to the uncertainty value of a given point p. This value quantifies the knowledge about its chance of occurrence. For simplicity, $F_X(x)$ is noted F_x^p, i.e. the *cdf* value of an uncertain data p at value x. A data population, with a uniformly distributed *cdf* curve, creates the set of *cdf*-interval points such that each point p_x is specified by the observed data quantile in the real domain $\mathbb{R}$ along with its *cdf* value. This quantifiable knowledge is issued from a probability distribution, hence its value ranges $\in [0, 1]$.

Definition 6.1 (a *cdf*-interval lattice structure). *is the set of tuples, specified, in the 2D-space, as* $\mathcal{U}_1 = \mathbb{R}\times[0, 1]$ *: real quantiles and their corresponding cdf values. Tuples are partially ordered over the set* $\mathcal{U}_1$ *with unique glb and lub.*

A p-box *cdf*-interval point p_x typically lies on a cumulative uniform distribution function F_X which shapes a line; this line characterizes two values: a *cdf* value and a slope denoted respectively by F_x^p and S_x^{p*}. F_x^p quantifies the aggregated probability of point p at quantile x (by definition $0 \leq F_x^p \leq 1$) and S_x^p signifies the average step chance of occurrence of sequencial quantiles lying on the same *cdf*-distribution. Due to the monotonic property of the cumulative distribution function, S_x^p cannot take negative values. By definition, a p-box *cdf*-interval point p_x is a triplet specified by (x,F_x^p, S_x^p) $\in \mathcal{U}_b$.

Definition 6.2 (a p-box *cdf*-interval lattice structure). *is the set,* $\mathcal{U}_b = \mathbb{R}\times[0, 1]\times\mathbb{R}^+$, *of triplets (observed quantile, cdf and slope) partially ordered, and constitutes a poset with unique glb and lub.*

It is worth noting that the three elements must be present in order to express the full information of the uniform *cdf*-distribution, which is described by a line equation, and issued from an arbitrary quantile. The first element of the triplet is the real quantile; the second element is its *cdf* value; and the last element shows the average step probabilistic value of the distribution.

*For a real interval $[a, b]$, the slope of the *cdf* uniform distribution is given by $\frac{F_b - F_a}{b-a}$, where F_b and F_a are the *cdf* values of quantiles b and a on this *cdf*-distribution respectively

Theorem 6.1. *The p-box cdf-interval lattice is partially ordered, and constitutes a poset with unique glb and lub.*

Proof. Points defined on the p-box *cdf*-interval lattice are specified as triplets $\mathcal{U}_b = \mathbb{R} \times [0,1] \times \mathbb{R}^+$. Elements of the triplet shape a uniform *cdf* distribution issued from the quantiles defined on the real domain $\mathbb{R}$. The list of *cdf* distributions follow the stochastic dominance ordering, defined in Section 2.3, to order, partially, random variables (probabilities). Accordingly, p-box *cdf*-interval triplet points are partially ordered in a 2D manner: reals and probabilities. By definition, partially ordered elements form a poset in which every two elements have a meet *glb* and a join *lub*. Consequently, the poset has a unique *glb* and a unique *lub*. □

6.2.2 Partial Ordering

We can order (partially) points in $\mathcal{U}_1$ and $\mathcal{U}_b$ for the *cdf*-intervals and p-box *cdf*-intervals respectively. Accordingly, we can construct an algebra over variables taking their value in the 2D-space.

The 2D-ordering $\mathcal{U}_1$ arranges *cdf*-interval points such that points with smaller quantiles and less *cdf* values come before those points with higher quantiles and higher aggregated chance to occur.

Definition 6.3 (Ordering over $\mathcal{U}_1, \preccurlyeq_{\mathcal{U}_1}$). *Let $p_x = (x, F_x^p)$, $q_y = (y, F_y^q) \in \mathcal{U}_1$, the ordering $\preccurlyeq_{\mathcal{U}_1}$ is a partial order defined by:*

$$p_x \preccurlyeq_{\mathcal{U}_1} q_y \Leftrightarrow x \leq y \text{ and } F_x^p \leq F_y^q \tag{6.1}$$

Example 6.1. *Consider the three points $p_x = (4, 0.17), q_y = (9, 0.87)$ and $r_z = (2, 0.43)$ depicted in Fig. 6.1 (a). We have $p_x \preccurlyeq_{\mathcal{U}_1} q_y$ and $r_z \preccurlyeq_{\mathcal{U}_1} q_y$, but p_x and r_z are not comparable in the $\preccurlyeq_{\mathcal{U}_1}$ ordering. In this case, the ordering is satisfied in the 1^{st} dimension, quantiles $2 \leq 4$, but it is unfulfilled in the 2^{nd} dimension,* cdf *values $0.43 \not\leq 0.17$.*

Property 6.1. *A* cdf*-interval delimited by two points p_x and q_y is specified by the syntax $[p_x, q_y]$ such that $p_x \preccurlyeq_{\mathcal{U}_1} q_y$.*

Property 6.2. *Since $\mathcal{U}_1$ is a poset then by definition any subset $[p_x, q_y] \in \mathcal{U}_1$ satisfies the following laws:*

$\mathcal{P}1$. *Reflexivity:* $\forall x,\ p_x \preccurlyeq_{\mathcal{U}_1} p_x$

$\mathcal{P}2$. *Antisymmetry:* $(p_x \preccurlyeq_{\mathcal{U}_1} q_y \text{ and, } q_y \preccurlyeq_{\mathcal{U}_1} p_x) \Rightarrow (p_x = q_y)$

$\mathcal{P}3$. *Transitivity:* $(p_x \preccurlyeq_{\mathcal{U}_1} q_y \text{ and, } q_y \preccurlyeq_{\mathcal{U}_1} r_z) \Rightarrow (p_x \preccurlyeq_{\mathcal{U}_1} r_z)$

The ordering $\preccurlyeq_{\mathcal{U}_b}$ arranges p-box *cdf*-interval points such that points at the outset lie on dominated *cdf* distributions, where small data values are more likely to happen at smaller quantiles and vice versa.

Definition 6.4 (Ordering over $\mathcal{U}_b, \preccurlyeq_{\mathcal{U}_b}$). *Let $p_x = (x, F_x^p, S_x^p)$, $q_y = (y, F_y^q, S_y^q) \in \mathcal{U}_b$, the ordering $\preccurlyeq_{\mathcal{U}_b}$ is a partial order defined by:*

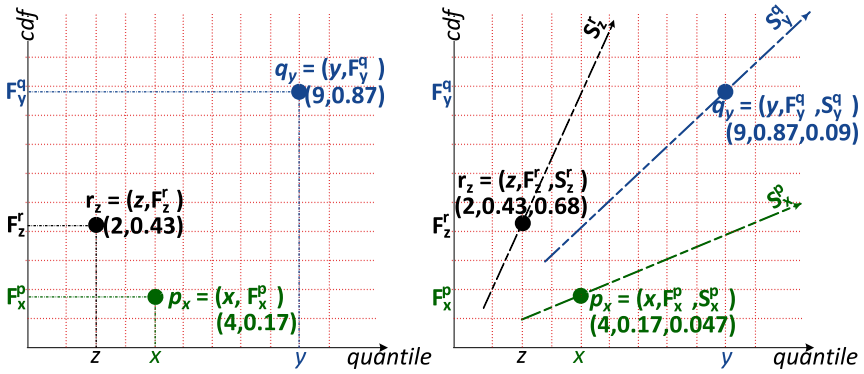
$$p_x \preccurlyeq_{\mathcal{U}_b} q_y \Leftrightarrow x \leq y \text{ and } \int_{-\infty}^{y} F^q dy \leq \int_{-\infty}^{y} F^p dy \tag{6.2}$$

p_x and q_y are located on two different *cdf*-distributions: F^p and F^q respectively. The integral ordering enforces the a second order stochastic dominance of F^q over F^p. Both F^p and F^q are calculated from the stored *cdf* value and slope.

Appendix A.1 shows the integration derivation of linear equations. Integration inequality yields a direct linear substitution in the following

$$\int_{-\infty}^{y} F^q dy \leq \int_{-\infty}^{y} F^p dy \Leftrightarrow F_y^q \leq (y - x)S_x^p + F_x^p \tag{6.3}$$

Example 6.2. *Consider three points $p_x = (4, 0.17, 0.047), q_y = (9, 0.87, 0.09)$ and $r_z = (2, 0.43, 0.68)$ which are depicted in Fig. 6.1 (b). We have $r_z \preccurlyeq_{\mathcal{U}_b} p_x$ and $r_z \preccurlyeq_{\mathcal{U}_b} q_y$, but p_x and q_y are not comparable on the $\mathcal{U}_b$ lattice. If we substitute the values in the inequality, we obtain $4 \leq 9$ but $0.87 \not\leq 0.405$. In this case, quantile ordering is satisfied but the stochastic dominance inequality is not.*



(a) (b)

Fig. 6.1: Points ordering over (a) $\mathcal{U}_1$ (b) $\mathcal{U}_b$

Property 6.3. *A p-box cdf-interval delimited by two points p_x and q_y is specified by the syntax $[p_x, q_y]$ such that $p_x \preccurlyeq_{\mathcal{U}_b} q_y$.*

Property 6.4. *The poset property of $\mathcal{U}_b$ enforces any subset $[p_x, q_y] \in \mathcal{U}_b$ to satisfy the following rules:*

$\mathcal{P}1$. *Reflexivity:* $\forall x,\ p_x \preccurlyeq_{\mathcal{U}_b} p_x$

$\mathcal{P}2$. *Antisymmetry:* $(p_x \preccurlyeq_{\mathcal{U}_b} q_y$ *and,* $q_y \preccurlyeq_{\mathcal{U}_b} p_x) \Rightarrow (p_x = q_y)$

$\mathcal{P}3$. *Transitivity:* $(p_x \preccurlyeq_{\mathcal{U}_b} q_y$ *and,* $q_y \preccurlyeq_{\mathcal{U}_b} r_z) \Rightarrow (p_x \preccurlyeq_{\mathcal{U}_b} r_z)$

6.2.3 Meet and Join Operators

One important task in interval reasoning is the computation of a new interval from arbitrary points or previous intervals, such that it describes the smallest interval containing the possible collection of elements. This is based on the meet and join operators.

Definition 6.5 (Meet and join over $\mathcal{U}_1$). *Given the arithmetic ordering and meet and join operations over the reals $(\mathbb{R}, \leq, min, max)$ and the ordering of cdf values within $([0,1], \leq, min, max)$, the meet lub_1 and join glb_1 operators of two points p_x and q_y in $\mathcal{U}_1$ are defined by:*

$$\begin{aligned} glb_1(p_x, q_y) &= (min(x,y),\ min(F^p_{lb}, F^q_{lb})) \\ lub_1(p_x, q_y) &= (max(x,y),\ max(F^p_{ub}, F^q_{ub})) \end{aligned} \tag{6.4}$$

where $lb = min(x,y)$ and $ub = max(x,y)$

From Equation 6.4, we can deduce that glb_1 is a point in the 2D-space. The 1^{st} dimension (quantile), lb, is a result of applying the minimum operation on the points quantiles. To obtain the *cdf* component of the glb_1, we project lb onto, F^p and F^q, the *cdf* distributions of the points under consideration. The 2^{nd} component of the glb_1 is the minimum value obtained from the projections. Similarly, we compute lub_1 but in that case we replace the minimum by the maximum operation. The consistency property establishes the link between the partial ordering $\preccurlyeq_{\mathcal{U}_1}$ and the pair (glb_1, lub_1) as actual meet and join.

Property 6.5 (Consistency property over $\mathcal{U}_1$).

$$\begin{aligned} p_x \preccurlyeq_{\mathcal{U}_1} q_y &\Leftrightarrow p_x = glb_1(p_x, q_y) \\ p_x \preccurlyeq_{\mathcal{U}_1} q_y &\Leftrightarrow q_y = lub_1(p_x, q_y) \end{aligned} \tag{6.5}$$

Proposition 6.1 ($\bot_{\mathcal{U}_1}$ and $\top_{\mathcal{U}_1}$).

$\bot_{\mathcal{U}_1}$ *is the universal greatest lower bound of cdf-interval points and it is equal to (*$0.0, 0.0$*)*

$\top_{\mathcal{U}_1}$ *is the universal least upper bound of cdf-interval points and it is equal to (*$+\infty, 1.0$*)*

assuming that all random variables are greater than zero.

Proof. On the $\mathbb{R}$ domain where all random variables are observed greater than 0, the minimum and the maximum measured quantiles are respectively 0 and $+\infty$. The *cdf* is a probability distribution, hence 0 and 1 are respectively the minimum and the maximum possible values a *cdf* can obtain. Accordingly, we can conclude that points $\bot_{\mathcal{U}_1} = (0.0, 0.0)$ and $\top_{\mathcal{U}_1} = (+\infty, 1.0)$ □

Example 6.3. *Fig. 6.2 [a] depicts the glb_1 and lub_1 computations of two points $p_x = (4, 0.17)$ and $q_y = (9, 0.87)$. By substituting in Eq. 6.4, the minimum and maximum quantiles are respectively 4 and 9. The minimum and maximum cdf values obtained from the projection operation are 0.09 and 0.8 respectively. Accordingly the points, $glb_1(p_x, q_y) = (2, 0.09)$ and $lub_1(p_x, q_y) = (9, 0.8)$.*

Definition 6.6 (Meet and join over $\mathcal{U}_b$). *Given the arithmetic ordering, meet and join operations over reals $(\mathbb{R}, \leq, min, max)$ and the second order stochastic dominance of cdf distributions, the meet (glb_b) and join (lub_b) operators of two points p_x and q_y in $\mathcal{U}_b$ are defined by:*

$$\begin{aligned} glb_b(p_x, q_y) &= (min(x, y),\ max(F^p_{lb}, F^q_{lb}),\ max(S^p_x, S^q_y)) \\ lub_b(p_x, q_y) &= (max(x, y),\ min(F^p_{ub}, F^q_{ub}),\ min(S^p_x, S^q_y)) \end{aligned} \tag{6.6}$$

where $lb = min(x, y)$ and $ub = max(x, y)$

The following property establishes the link between the partial ordering $\preccurlyeq_{\mathcal{U}_b}$ and the pair (glb_b, lub_b) as actual meet and join.

Property 6.6 (Consistency property over $\mathcal{U}_b$).

$$\begin{aligned} p_x \preccurlyeq_{\mathcal{U}_b} q_y &\Leftrightarrow p_x = glb_b(p_x, q_y) \\ p_x \preccurlyeq_{\mathcal{U}_b} q_y &\Leftrightarrow q_y = lub_b(p_x, q_y) \end{aligned} \tag{6.7}$$

Proposition 6.2 ($\bot_{\mathcal{U}_b}$ and $\top_{\mathcal{U}_b}$).

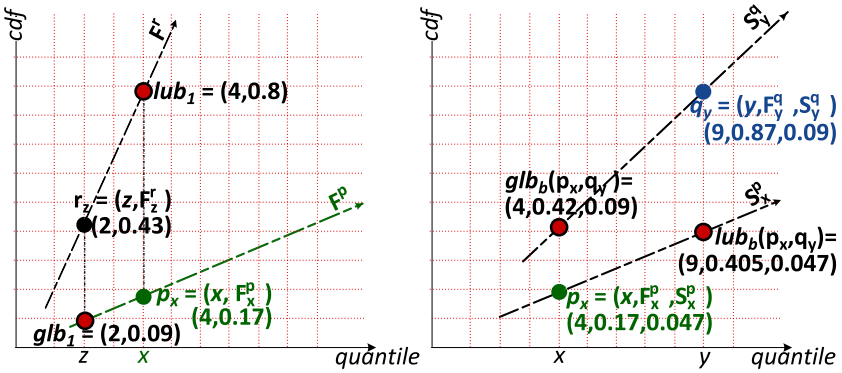
$\bot_{\mathcal{U}_b}$ *is the universal greatest lower bound of p-box cdf-interval points and it is equal to $(0.0, 1.0, \infty)$*

$\top_{\mathcal{U}_b}$ *is the universal least upper bound of p-box cdf-interval points and it is equal to $(+\infty, 0.0, 0.0)$*

assuming that all random variables are greater than zero.

Proof. On the $\mathbb{R}$ domain where all random variables are observed greater than 0, the minimum and the maximum measured quantiles are respectively 0 and $+\infty$. The 2nd order stochastic dominance integral defined in Eq. 6.3 yields 0 for $\bot_{\mathcal{U}_b}$ and ∞ for $\top_{\mathcal{U}_b}$. Accordingly, the proposed universal greatest lower bound and lowest upper bound, both maintain lattice real ordering on the $\mathbb{R}$ domain in addition to the stochastic dominance on the probabilistic domain. □

Example 6.4. *Fig. 6.2 [b] illustrates an example which computes glb_b and lub_b of two points $p_x = (4, 0.17, 0.047)$ and $q_y = (9, 0.87, 0.09)$. By substituting in Eq. 6.6, the minimum and maximum quantiles are respectively 4 and 9; the results of applying the max and min operations on the computed stochastic dominance integration are 0.42 and 0.405 respectively; the maximum and minimum slopes are respectively 0.09 and 0.047. Accordingly we can compute the points, $glb_b(p_x, q_y) = (4, 0.42, 0.09)$ and $lub_b(p_x, q_y) = (9, 0.405, 0.047)$.*



(a) (b)

Fig. 6.2: Meet and join operations (a) glb_1 and lub_1 (b) glb_b and lub_b

6.2.4 Data Point Arithmetic Operations

We extend the computation domain with binary *cdf* point arithmetics. Operations to combine uncertainties in the framework rely on the convolution operation over random variables detailed in Section 2.2. We consider the standard arithmetic operations interpreted over the set of reals $\mathbb{R}$, then, we extend the notion to reason about the *cdf* distribution of the data population.

Theorem 6.2. *For* $\odot \in \{+_{\mathcal{U}}, -_{\mathcal{U}}, \times_{\mathcal{U}}, \div_{\mathcal{U}}\}$ *a binary cdf point arithmetic over the 2D-space,* $p_x \odot q_y = ((x \odot y), F^{p \odot q}_{x \odot y}, S^{p \odot q}_{x \odot y})$*, yields a cdf point defined in the 2D space. Resulting point is a triplet with a quantile value, a cdf distribution and a slope*

Any two points, each lying on a different *cdf* distribution, can be involved in a relation given by a function. This relation outlines a *cdf* that is based on their joint *cdf*-distribution, defined in Section 2.2, by double integrating the product of the two probability distributions they shape Stark and Woods (1994); Williamson and Downs (1990).

We derive the *cdf* equations of the binary arithmetic operations over uniform *cdf* distributions, and described by Definition 2.4, in Appendix A.3, A.4, A.5 and A.6. Notice that uniform distribution computations are linear. Hence, they are inexpensive as opposed to other existing probability distributions.

Lemma 6.1. $+_{\mathcal{U}}$ *is the binary p-box cdf point addition over the 2D-space.*

$p_x +_{\mathcal{U}} q_y = ((x + y), F^{p+q}_{x+y}, S^{p+q}_{x+y})$

$(x + y)$ *is the addition operation of the two points in the real domain* $\mathbb{R}$*,* F^{p+q}_{x+y} *and* S^{p+q}_{x+y} *are respectively the cdf value and the slope of the cdf distribution resulting from*

the addition operation in the probabilistic domain and are derived in Appendix A.3.2

$$
\begin{aligned}
F_{x+y}^{p+q} &= \int_{p_0+q_0}^{x+y} \int_{p_0}^{z-q_0} \frac{1}{(p_1-p_0)(q_1-q_0)} d\tau dz \quad (p_0+q_0) \leq (x+y) \leq (p_0+q_1) \\
&+ \int_{p_0+q_1}^{x+y} \int_{p_0}^{q_1} \frac{1}{(p_1-p_0)(q_1-q_0)} d\tau dz \quad (p_0+q_1) \leq (x+y) \leq (p_1+q_0) \\
&+ \int_{p_1+q_0}^{x+y} \int_{z-q_1}^{p_1} \frac{1}{(p_1-p_0)(q_1-q_0)} d\tau dz \quad (p_1+q_0) \leq (x+y) \leq (p_1+q_1)
\end{aligned} \tag{6.8}
$$

$$
S_{x+y}^{p+q} = \frac{1}{(p_1+q_1)-(p_0+q_0)}
$$

As deribed by Equation A.10, (p_0 and q_0) are the quantile projections of the cdf distributions with cdf value equal to 0 *and (p_1 and q_1) are the quantiles with cdf value equal to* 1.

Proof. Readers can refer to Section A.3 in the Appendix □

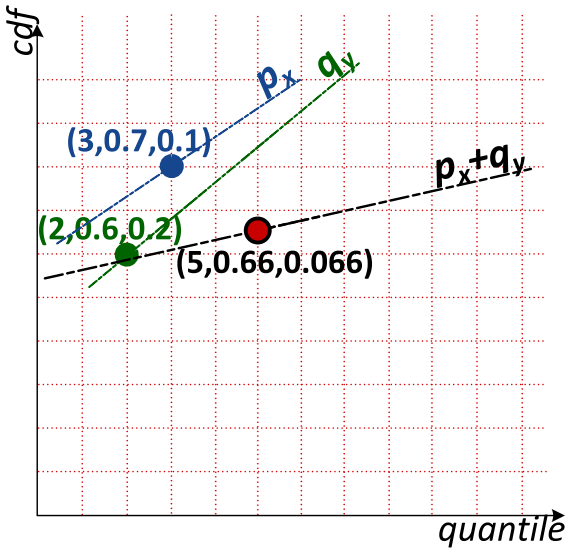


Fig. 6.3: The p-box *cdf*-point addition

Example 6.5. *Let p_x and q_y be two p-box cdf triplet points such that $p_x = (3.0, 0.7, 0.1)$ and $q_y = (2, 0.6, 0.2)$. We can compute the p-box cdf addition of two intervals, given that their bounds are defined in the 2D-space (values and knowledge about their occurrence), as follows:*

1. *Calculate the projected quantiles of the uniform distributions each point lies on. From Equation A.10 $p_0 = -3.99$, $p_1 = 6.0$, $q_0 = -0.99$ and $q_1 = 4.0$.*

2. *Compute the real addition operation $(x + y) = 5$*

3. *Derive the integral value*[†] *of the cdf from Equation 6.20, and as detailed in Appendix A.3.2*

   a) *Calculate the quantiles which bound the regions of the addition operation illustrated in Fig. A.3 for each cdf point respectively.*

   $\{ p_0 + q_0, p_1 + q_0, p_0 + q_1, p_1 + q_1 \} = \{ -4.98, 5.01, 0.01, 10.0 \}$. *Resulting lists are then sorted in order to define the different areas of integration*

   b) *The real addition is located within the quantile addition segments. This is exerted in order to determine the number of regions which will be considered in the integration. For this example, we have* $p_0 + q_1 \leq (x + y) \leq p_1 + q_0$, *the* 2^{nd} *and* 3^{rd} *elements in the computed list, includes* 2 *integration regions.*

   $F_{x+y}^{p+q} = 0.66$

4. *Calculate the slope of the cdf distribution from Equation 6.21.* $S_{x+y}^{p+q} = 0.066$

Fig. 6.3 illustrates the arithmetic cdf point addition relation $(p + q)$. *Result of the operation is specified by the p-box cdf-point* $((x + y), F_{x+y}^{p+q}, S_{x+y}^{p+q}) = (5.0, 0.66, 0.066)$.

Lemma 6.2. $\times_{\mathcal{U}}$ *is the binary p-box cdf point multiplication over the 2D-space.*

$p_x \times_{\mathcal{U}} q_y = ((x \times y), F_{xy}^{pq}, S_{xy}^{pq})$ $(x \times y)$ *is the* $\mathbb{R}$ *domain multiplication operation,* F_{xy}^{pq} *and* S_{xy}^{pq} *are respectively the cdf value and the slope of the cdf distribution resulting from the multiplication operation in the probabilistic domain and are derived in Appendix A.4.2*

$$
\begin{aligned}
F_{xy}^{pq} &= \int_{p_0 q_0}^{xy} \int_{p_0}^{\frac{z}{q_0}} \frac{1}{(p_1 - p_0)(q_1 - q_0)} \frac{1}{|\tau|} d\tau dz \quad (p_0 q_0) \leq (xy) \leq (p_0 q_1) \\
&+ \int_{p_0 q_1}^{xy} \int_{\frac{z}{q_1}}^{\frac{z}{q_0}} \frac{1}{(p_1 - p_0)(q_1 - q_0)} \frac{1}{|\tau|} d\tau dz \quad (p_0 q_1) \leq (xy) \leq (p_1 q_0) \\
&+ \int_{p_1 q_0}^{xy} \int_{\frac{z}{q_1}}^{p_1} \frac{1}{(p_1 - p_0)(q_1 - q_0)} \frac{1}{|\tau|} d\tau dz \quad (p_1 q_0) \leq (xy) \leq (p_1 q_1)
\end{aligned} \tag{6.9}
$$

$$
S_{xy}^{pq} = \frac{1}{(p_1 q_1) - (p_0 q_0)}
$$

Proof. Readers can refer to Section A.4 in the Appendix □

Example 6.6. *The result of multiplying two p-box cdf-points* $p_x = (3.0, 0.7, 0.1)$ *and* $q_y = (2, 0.6, 0.2)$ *is computed in the 2D-space as detailed in Appendix A.4.2.*

1. *The bounding quantiles of the uniform distributions are computed as in the first step of Example 6.5.*

[†] *To compute the integration we calculate the area under the cdf distribution line along the bounds defined on the integral limits*

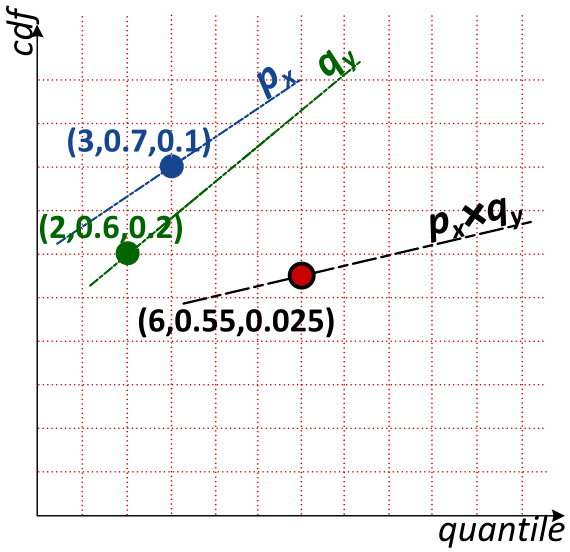


Fig. 6.4: The p-box *cdf*-point multiplication

2. *The computed multiplication in the real domain* $(x \times y) = 6$.

3. *The integration over the joint* pdf *is performed in order to compute the* cdf *bounding values. This is obtained from Equation 6.9 and derived in Appendix A.4.2*

   a) *Quantiles, depicted in Fig. A.4, which bound the different segments of the multiplication are computed then sorted.*

   $\{ p_1q_0, p_0q_1, p_0q_0, p_1q_1 \} = \{ -15.99, -5.99, 3.99, 24.0 \}$.

   b) *xy is located in the* 3^{rd} *segment, hence, the number of integrals to be considered in the* cdf *calculations.*

   $F_{xy}^{pq} = 0.55$

4. *The resulting slope of the* cdf *distribution obtained from Equation 6.10.* $S_{xy}^{pq} = 0.025$

The p-box cdf*-point depicted in Fig. 6.10 is the result of multiplying* p_x *and* $q_y = (6, 0.55, 0.025)$.

Lemma 6.3. $-_{\mathcal{U}}$ *is the binary p-box cdf point subtraction over the 2D-space.*

$p_x -_{\mathcal{U}} q_y = ((x - y), F_{x-y}^{p-q}, S_{x-y}^{p-q})$ $(x - y)$ *is the* $\mathbb{R}$ *domain subtraction operation,* F_{x-y}^{p-q} *and* S_{x-y}^{p-q} *are respectively the cdf value and the slope of the cdf distribution resulting from the subraction operation in the probabilistic domain and are derived in Appendix A.5.2*

$$
\begin{aligned}
F_{x-y}^{p-q} &= \int_{q_0-p_1}^{x-y}\int_{q_0}^{z+p_1}\frac{1}{(q_1-q_0)(p_1-p_0)}d\tau dz \quad (q_0-p_1)\le(x-y)\le(q_0-p_0)\\
&+\int_{q_0-p_0}^{x-y}\int_{q_0}^{q_1}\frac{1}{(q_1-q_0)(p_1-p_0)}d\tau dz \quad (q_0-p_0)\le(x-y)\le(q_1-p_1)\\
&+\int_{q_1-p_1}^{x-y}\int_{z+p_0}^{q_1}\frac{1}{(q_1-q_0)(p_1-p_0)}d\tau dz \quad (q_1-p_1)\le(x-y)\le(q_1-p_0) \qquad (6.10)
\end{aligned}
$$

$$
S_{x-y}^{p-q} = \frac{1}{(q_0-p_1)-(q_1-p_0)}
$$

Proof. Readers can refer to Section A.5 in the Appendix □

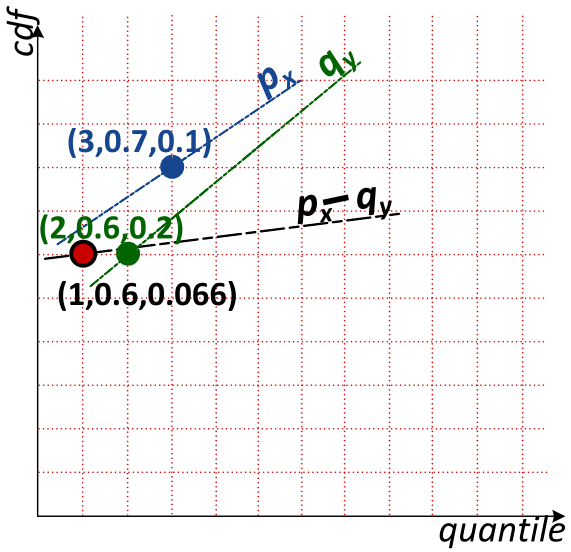


Fig. 6.5: The p-box *cdf*-point subtraction

Example 6.7. *The p-box* cdf*-point resulting from the difference between* p_x *and* q_y *in the relation* $p_x - q_y$ *is computed following the steps listed in A.5.2*

1. *We compute the real difference* $(x-y) = 1$
2. *Bounding values of the* cdf *in Equation 6.10 which result from the integration detailed in Appendix A.5.2* $F_{x-y}^{p-q} = 0.6$
3. *The slopes of the bounding* cdf *distribution are obtained from Equation 6.24.* $S_{x-y}^{p-q} = 0.066$

The p-box cdf*-point output from the subtraction operation* $p_x - q_y$ *is shown in Fig. 6.11 and is equal to* $(1, 0.6, 0.066)$.

Lemma 6.4. *$\div_{\mathcal{U}}$ is the binary p-box cdf point division over the 2D-space.*

$p_x \div_{\mathcal{U}} q_y = ((x \div y), F^{\frac{p}{q}}_{\frac{x}{y}}, S^{\frac{p}{q}}_{\frac{x}{y}})$ $(x \div y)$ is the $\mathbb{R}$ domain division operation, $F^{\frac{p}{q}}_{\frac{x}{y}}$ and $S^{\frac{p}{q}}_{\frac{x}{y}}$ are respectively the cdf value and the slope of the cdf distribution resulting from the division operation in the probabilistic domain and are derived in Appendix A.6.2

$$
\begin{aligned}
F^{\frac{p}{q}}_{\frac{x}{y}} &= \int_{\frac{q_0}{p_1}}^{\frac{x}{y}} \int_{\frac{q_0}{z}}^{p_1} \frac{|\tau|}{(q_1-q_0)(p_1-p_0)} d\tau dz \quad \frac{q_0}{p_1} \leq \frac{x}{y} \leq \frac{q_0}{p_0} \\
&+ \int_{\frac{q_0}{p_0}}^{\frac{x}{y}} \int_{p_0}^{p_1} \frac{|\tau|}{(q_1-q_0)(p_1-p_0)} d\tau dz \quad \frac{q_0}{p_0} \leq \frac{x}{y} \leq \frac{q_1}{p_1} \\
&+ \int_{\frac{p_1}{q_1}}^{\frac{x}{y}} \int_{p_0}^{q_1} \frac{|\tau|}{(q_1-q_0)(p_1-p_0)} d\tau dz \quad \frac{q_1}{p_1} \leq \frac{x}{y} \leq \frac{q_1}{p_0}
\end{aligned} \tag{6.11}
$$

$$
S^{\frac{p}{q}}_{\frac{x}{y}} = \frac{1}{(\frac{q_0}{p_1})-(\frac{q_1}{p_0})}
$$

Proof. Readers can refer to Section A.6 in the Appendix □

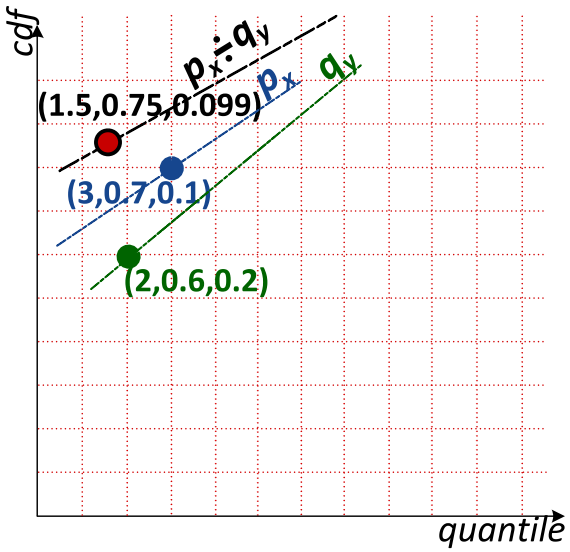


Fig. 6.6: The p-box *cdf*-point division

Example 6.8. *The p-box cdf-point resulting from the division between p_x and q_y in the relation $p_x \div q_y$ is computed by following the steps listed in A.6.2. Depicted in Fig. 6.6 the result of computing the division in the 2D-space and it is equal to* $(1.5, 0.75, 0.099)$.

6.3 Defining the *cdf* interval constraint domain

The *cdf* constraint domain is the structure over which *cdf* interval calculus is exerted. We construct the algebraic structure over the computation domain equipped with the

predicate symbols $\preccurlyeq_{\mathcal{U}}$ and $\in_{[p_a,q_b]}$, the *cdf* constraint relations, which belong to the signature $\Sigma_{\mathcal{U}}$. $\preccurlyeq_{\mathcal{U}}$ is the *cdf* point ordering and $\in [p_a, q_b]$ is interpreted as the constraint of a *cdf* interval variable to take its value within the specified domain. The main idea of the system is to perform interval calculus which guarantees that the domain of any variable is enveloped by a *cdf* interval.

The fundamental algebraic structure of the *cdf*-interval constraint domain is the interval which encloses a set of *cdf*-points characterized in the 2D-space: $\mathcal{U}_1$ or $\mathcal{U}_b$. To remain computationally tractable, we do not maintain a full domain representation of points which define the set of distributions; rather we define the interval domain which encloses a family of probability distributions. Defined *cdf* intervals, in the constraint domain, are employed to reason about data with uncertainty inexpensively in the Constraint Logic Programming (CLP) paradigm.

6.3.1 *cdf*-interval structure

The key element to a *cdf*-interval domain is the uniform *cdf* distribution it lies on. However, to remain computationally tractable we do not maintain a full domain representation of the points defining the distribution. Instead, we approximate the curve by a line whose extreme points are the bounds of the interval. Elements of the interval domain, belonging to $\mathcal{U}_1$, lie on this linear curve.

Definition 6.7 (*cdf*-interval domain). *is a pair* $[p_a, p_b]$ *satisfying* $p_a \preccurlyeq_{\mathcal{U}_1} p_b$ *and denoting the convex structure enclosing the set:*

$$\{p_x = (x, F_x^p) \mid p_a \preccurlyeq_{\mathcal{U}_1} p_x \preccurlyeq_{\mathcal{U}_1} p_b,\ a \le x \le b,\ \textit{and}\ F_x^p = \frac{F_b^p - F_a^p}{b-a}.(x-a) + F_a^p\} \quad (6.12)$$

Property 6.7. *A* cdf *variable* X *in the constraint relation* $X \in [p_a, p_b]$ *takes its value from the constraint domain range enclosed by the two* cdf*-points* p_a *and* p_b.

Example 6.9. *Consider a* cdf*-variable in the relation* $X \in I$ *where* $\boldsymbol{I} = [(1, 0.16), (7.75, 0.98)]$, *illustrated in fig. 5.6[a]. X can take any point value* (x, F_x^I) *such that* $1 \le x \le 7.75$ *and* $F_x^I = \frac{0.98-0.16}{7.75-1}.(x-1) + 0.16$.

Together with real (min, max) operations, the second order stochastic dominance is employed in the p-box framework to characterize the bounds of the interval. Designated bounds are chosen to follow uniform distributions in order to simplify the computations. The enveloped set of p-box *cdf*-points has an upper bound *cdf* with the fastest rising slope and which is dominated by all enclosed distributions in the interval. Conversely, the lower bound *cdf* has the slowest rising slope and it is dominating all *cdf* distributions contained within the interval bounds.

Definition 6.8. *[pbox cdf-interval domain] is a pair* $[p_a, q_b]$ *satisfying* $p_a \preccurlyeq_{\mathcal{U}_b} q_b$ *and denoting the set:*

$$\{p_x = (x, F_x^p, S_x^p) \mid p_a \preccurlyeq_{\mathcal{U}_b} p_x \preccurlyeq_{\mathcal{U}_b} q_b\} \quad (6.13)$$

where $a \le x \le b$, $F_a^{p'} \ge F_x^p \ge F_b^{q'}$ *and* $S_a^p \ge S_x^p \ge S_b^q$

$F_a^{p'}$ and $F_b^{q'}$ are the linear projections of quantile x onto the *cdf*-distribution bounds and they are computed as follows:

$$F_a^{p'} = min(F_a^p - S_a^p(a-x), 1) \quad \text{and} \quad F_b^{q'} = max(F_b^q - S_b^q(b-x), 0) \tag{6.14}$$

Example 6.10. *A cdf-variable X in the constraint relation $X \in \boldsymbol{I}$. $\boldsymbol{I} = [(1, 0.16, 0.16), (8, 0.64, 0.08)]$, illustrated in fig. 5.6 [b]. X can take any point value (x, F_x^I, S_x^I) such that $1 \leq x \leq 8$, F_x^I bounds lie between $min((0.16 - 0.16(1-x)), 1) \geq F_x^I \geq max((0.64 - 0.08(8-x)), 0)$, and the average step probabilistic value is $0.16 \geq S_x^I \geq 0.08$*

Property 6.8. *A cdf variable X is sought to range over a cdf-interval domain $\boldsymbol{I} = [p_a, q_b]$ if and only if $X \in [p_a, q_b]$*

Definition 6.9. *The smallest convex interval which represents two arbitrary points p_x and q_y defined in a 2D-space lattice is: $[glb_{\mathcal{U}_b}(p_x, q_y), lub_{\mathcal{U}_b}(p_x, q_y)]$.*

Theorem 6.3. *A p-box cdf-interval $[p_a, p_b]$ is convex and it forms a lattice structure*

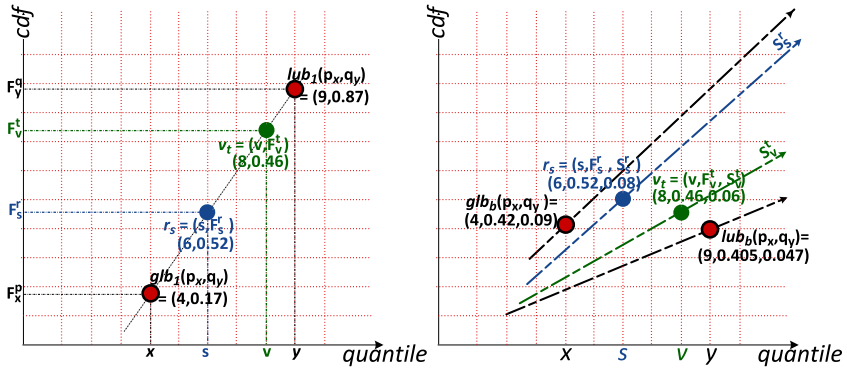
Proof. Let $I = [p_a, p_b]$ be a p-box *cdf*-interval. If r_s and $v_t \in I$ then by definition 6.8 $p_a \preccurlyeq_{\mathcal{U}_b} r_s, v_t \preccurlyeq_{\mathcal{U}_b} p_b$. The bounding points p_a and p_b are the $glb_{\mathcal{U}_b}$ and the $lub_{\mathcal{U}_b}$ of all points lying within the interval including r_s and v_t. One can deduce that all points lying between $glb_{\mathcal{U}_b}(r_s, v_t)$ and $lub_{\mathcal{U}_b}(r_s, v_t)$ exist $\in \mathbf{I}$, hence, the domain range specified as $[glb_{\mathcal{U}_b}(r_s, v_t), lub_{\mathcal{U}_b}(r_s, v_t)] \in \mathbf{I}$. This proofs that the domain $[p_a, p_b]$ is convex with unique *glb* and *lub* which forms, from Theorem 6.1, a lattice structure. □

Property 6.9. *If $r_s, v_t \in \boldsymbol{I} = [glb_{\mathcal{U}_b}(p_x, q_y), lub_{\mathcal{U}_b}(p_x, q_y)]$, then, $glb_{\mathcal{U}_b}(r_s, v_t) \in \boldsymbol{I}$ and $lub_{\mathcal{U}_b}(r_s, v_t) \in \boldsymbol{I}$. One can deduce that all points lying between $glb_{\mathcal{U}_b}(r_s, v_t)$ and $lub_{\mathcal{U}_b}(r_s, v_t)$ exist $\in \boldsymbol{I}$, hence, the domain range specified as $[glb_{\mathcal{U}_b}(r_s, v_t), lub_{\mathcal{U}_b}(r_s, v_t)] \in \boldsymbol{I}$*

Example 6.11. *Fig. 6.7 (b) visualizes Property 6.9. Clearly, the smallest convex interval that forms the two points r_s and v_t: $[glb_{\mathcal{U}_b}(r_s, v_t), lub_{\mathcal{U}_b}(r_s, v_t)]$ $= [(6, 0.52, 0.08), (8, 0.45, 0.06)] \in [(4, 0.42, 0.09), (9, 0.405, 0.047)]$ which, in turn, is the smallest convex p-box cdf-interval that represents p_x and q_y. Similarily, Fig. 6.7 (a) depicts that $[(6, 0.45), (8, 0.73)] \in [(4, 0.17), (9, 0.87)]$ for the cdf-interval domain.*

6.3.2 *cdf*-Interval domain calculus

It is worth noting that the *cdf*-interval domain calculus guarantees an assignment of p-box *cdf*-interval domains to variables. This assignment is essentially a convex *cdf* structure. The convexity of the structure maintains an inexpensive computation of domains in a disjunction. This is due to the fact that interval operations are exerted on the predefined bounds of the given ranges. We can construct an algebra over the variables taking their value in the 2D-space specified as (quantiles × probability distributions), $\mathbb{R}\times[0,1]\times\mathbb{R}^+$ in the $\mathcal{U}_1$ domain and $\mathbb{R}\times[0,1]$ in the $\mathcal{U}_b$ domain.



(a) (b)

Fig. 6.7: Convex interval representation [a] *cdf*-interval [b] p-box *cdf*-interval

Definition 6.10. *The algebraic convex cdf structure is defined in the set of cdf domain of ranges* $R_{\mathcal{U}}$. *Each element in the set* $R_{\mathcal{U}}$ *is a convex subset of* $\mathcal{U}$.

Definition 6.11. *The cdf constraint domain is the algebraic structure of* $\mathcal{R}_{\mathcal{U}}$, *ordered by the* $\preccurlyeq_{\mathcal{U}}$ *and defines the cdf constraint relations* $\odot$ *and* $\in_{[p_a,q_b]}$. $\{\, \mathcal{R}_{\mathcal{U}}, \preccurlyeq_{\mathcal{U}}, \odot_{\mathcal{U}}, \in_{[p_a,q_b]} \,\}$

Definition 6.12. *Let* $\boldsymbol{I} = [p_a, p_b]$ *and* $\boldsymbol{J} = [q_c, q_d]$ *be two interval ranges* $\in \mathcal{R}_{\mathcal{U}}$ *the ordering relation* $\preccurlyeq_{\mathcal{U}}$ *over the two intervals is defined as:*

$$[p_a, p_b] \preccurlyeq_{\mathcal{U}} [q_c, q_d] \Leftrightarrow p_a \preccurlyeq q_c \;\; and \;\; p_b \preccurlyeq q_d \tag{6.15}$$

Definition 6.13. *Two cdf-interval ranges* $\boldsymbol{I} = [p_a, p_b]$ *and* $\boldsymbol{J} = [q_c, q_d] \in R_{\mathcal{U}}$ *are in a disjunction. The convex representation which encloses all elements from the two intervals* $\boldsymbol{I}$ *and* $\boldsymbol{J}$ *is their union*

$$\begin{aligned} lub_{\mathcal{R}_{\mathcal{U}}}([p_a, p_b], [q_c, q_d]) &= [glb_b(p_a, q_c), lub_b(p_b, q_d)] \\ &: glb_b(p_a, q_c) \preccurlyeq_{\mathcal{U}} lub_b(p_b, q_d) \end{aligned} \tag{6.16}$$

Definition 6.14. *The intersection of two cdf-interval ranges* $\boldsymbol{I} = [p_a, p_b]$ *and* $\boldsymbol{J} = [q_c, q_d]$ *encloses common elements,* $\in \mathcal{U}$, *from the two intervals* $\boldsymbol{I}$ *and* $\boldsymbol{J}$

$$\begin{aligned} glb_{\mathcal{R}_{\mathcal{U}}}([p_a, p_b], [q_c, q_d]) &= [lub_b(p_a, q_c), glb_b(p_b, q_d)] \\ &: lub_b(p_a, q_c) \preccurlyeq_{\mathcal{U}} glb_b(p_b, q_d) \end{aligned} \tag{6.17}$$

The meet and join operations over the p-box *cdf*-interval ranges $\in R_{\mathcal{U}}$ should maintain the convex property of the resulting interval. The result of these operations is an element of $R_{\mathcal{U}}$. In other words, the lower bound *cdf* distribution should always be dominated by the upper bound *cdf* uniform representation to ensure the second order stochastic dominance. When an area of conflict takes place, the case when the minimum possible probability distribution is greater than its maximum interpretation, the

p-box *cdf* operation seeks to further prune the resulting domain in order to eliminate the conflict. This elimination is exerted in two steps: 1) Extract the point where the two designated lines intersect; 2) The lower bound *cdf*-points is assigned the point of intersection. All points lying below this intersection are eliminated from the resulting p-box *cdf*-interval domain.

This operation is illustrated in Fig. 6.8 during the intersection operation where the calculated lub_b is $(2, 0.07, 0.24)$. In this illustration lower bound and upper bound distributions intersect at $(2.48, 0.16, 0.24)$. The area of conflict occurs between $(2, 0.07, 0.24)$ and $(2.48, 0.16, 0.24)$, in this area the minimum *cdf*-distribution is greater than its maximum presentation. To maintain the consistency of data (i.e. the convexity of the structure) the algorithm further prunes the resulting domain. This is exerted by shifting the point $(2, 0.07, 0.24)$ to $(2.48, 0.16, 0.24)$. Eventually, the additional calculation enforces the convex envelopment property, where the *cdf*-distribution bounds enclose all points which belong to the interval under consideration.

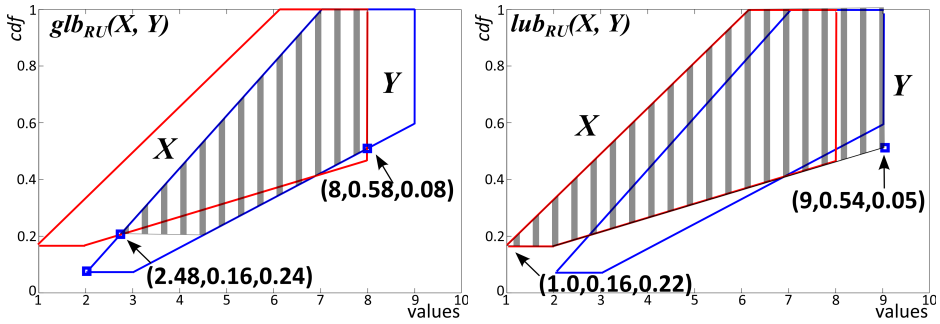


Fig. 6.8: Computing $glb_{\mathcal{R}_\mathcal{U}}$ and $lub_{\mathcal{R}_\mathcal{U}}$

Example 6.12. *Fig. 6.8 shows the computation of the meet and joint* $glb_{\mathcal{R}_\mathcal{U}}$ *and* $lub_{\mathcal{R}_\mathcal{U}}$ *constraints exerted on two variables* X *and* Y. $X \in [(1, 0.16, 0.22), (8, 0.49, 0.05)]$ *and* $Y \in [(2, 0.07, 0.24), (9, 0.67, 0.08)]$

$$glb_{\mathcal{R}_\mathcal{U}}(X, Y) = [(2.48, 0.16, 0.24), (8, 0.58, 0.08)]$$
$$lub_{\mathcal{R}_\mathcal{U}}(X, Y) = [(1.0, 0.16, 0.22), (9, 0.54, 0.05)]$$

6.3.3 *cdf*-interval domains arithmetic operations

Clearly this work follows the real interval arithmetic introduced in Benhamou and Older (1997). In particular, when the degree of knowledge provides equal weight to each data value, the computed intervals are identical.

The knowledge representation in the *cdf*-intervals is the tightest possible uniformly distributed *cdf*-bounds enveloping an unknown probability distribution. This encapsulation describes the data whereabouts minimum and maximum possible chance of occurrence.

For $\odot_{\mathcal{RU}} \in \{+_{\mathcal{RU}}, -_{\mathcal{RU}}, \times_{\mathcal{RU}}, \div_{\mathcal{RU}}\}$ a binary interval arithmetic over the 2D-space we seek:

$$[p_a, p_b] \odot_{\mathcal{RU}} [q_c, q_d] = \{p_X \odot q_Y \mid p_X \in [p_a, p_b], q_Y \in [q_c, q_d]\} \tag{6.18}$$

Theorem 6.4. *A binary interval arithmetic relation exerted over two p-box cdf-intervals yields a convex structure.*

Proof. The proof follows the work of Stark and Woods (1994); Williamson and Downs (1990) and noted in Section 2.2. Any two ranges, each enclosing a different *cdf* distribution, can be involved in a relation given by a function. This relation encloses their joint *cdf*-distribution. Probabilities resulting from all possible pairwise relations between two ill-defined random variables are contained within the output computation exerted on the upper and the lower *cdf* distribution bounds which are supported in Theorem 6.3 to maintain a convex structure. □

From this generic methodology we derive *cdf* equations for the binary arithmetic operations listed in Equation 6.18. Such operations seek the computation of the maximum and the minimum possible *cdf* bounds that encapsulate the probability distribution resulting from the random binary arithmetic operation. We have chosen the bounding *cdf*-distributions to be uniform, as described by Equation 2.2, due to its inexpensive linear computation as opposed to other existing probability distributions.

Addition '$+_{\mathcal{RU}}$'

Consider two intervals $\mathbf{I} = [p_a, p_b]$ and $\mathbf{J} = [q_c, q_d]$, their arithmetic addition is a result of adding every two points p_x and q_y, each lying within one of the intervals which belong to the binary computation. It is worth noting that this computation is very expensive if it is exerted in a pointwise manner. Hence, we rely on the convex property of the p-box *cdf*-interval to perform this computation on the interval bounds only. The result of this computation yields a p-box *cdf* convex structure which encloses all possible quantiles as well as probability distributions that can be output from the arithmetic addition operation. The addition of the *cdf*-intervals is exerted on the bounding quantile values using real interval arithmetic addition, the joint *cdf* addition is conducted on the two *cdf* bounding distributions separately, The resulting convex interval is specified by

$$[(lb_+, F^{I+J}_{lb_+}, S^{I+J}_{lb_+}), (ub_+, F^{I+J}_{ub_+}, S^{I+J}_{ub_+})]$$

Real interval arithmetic addition is applied to compute lower and upper quantile bounds respectively denoted as lb_+ and ub_+. The resulting *cdf*s, $F^{I+J}_{lb_+}$ and $F^{I+J}_{ub_+}$ are obtained by superimposing lower bound distributions (F^p_a and F^q_c) and upper bound distributions (F^p_b and F^q_d) respectively. The computation of the *cdf* values and slopes is based on the convolution operation previously discussed in section 2.2.

$$lb_+ = \min(a + c, a + d, b + c, c + d) \text{ and } ub_+ = \max(a + c, a + d, b + c, c + d) \tag{6.19}$$

$$\begin{aligned} F^{(I+J)}_{lb_+} &= \int_{a_{lb}+c_{lb}}^{lb_+} f_{IJ}(z)dz \\ F^{(I+J)}_{ub_+} &= \int_{a_{ub}+c_{ub}}^{ub_+} f_{IJ}(z)dz \end{aligned} \tag{6.20}$$

$$\begin{aligned} S^{I+J}_{lb_+} &= \frac{1}{(b_{lb}+d_{lb})-(a_{lb}+c_{lb})} \\ S^{I+J}_{ub_+} &= \frac{1}{(b_{ub}+d_{ub})-(a_{lb}+c_{lb})} \end{aligned} \tag{6.21}$$

Appendix A.3 describes the proof of this finding. It shows that the result of the integration in the above equations is a linear computation; knowing the bounds of the uniform distributions we directly substitute their values in order to obtain the bounds of the *cdf* distributions. a_{lb} and c_{lb} are quantiles with *cdf* value equal to 0; b_{ub} and d_{ub} are quantiles with *cdf* value equal to 1. These quantiles are located respectively on the lower bound and upper bound uniform *cdf* distributions.

The interval addition operation can be further generalized to the n-ary relation case.

Definition 6.15 (n-ary *cdf*-interval addition). *n cdf-intervals involved in the addition operation is denoted by*

$$[(lb^n_+, F^{n+}_{lb_+}, S^{n+}_{lb_+}), (ub^n_+, F^{n+}_{ub_+}, S^{n+}_{ub_+})]$$

$$lb^n_+ = (a_{lb_1} + ... + a_{lb_n}) \text{ and } ub^n_+ = (a_{ub_1} + ... + a_{ub_n})$$

$$\begin{aligned} F^{n+}_{lb_+} &= \frac{1}{n!}\frac{(a_{lb_1}+...+a_{lb_n}-lb^n_+)^n}{(b_{lb_1}-a_{lb_1})...(b_{lb_n}-a_{lb_n})} \\ F^{n+}_{ub_+} &= 1-\frac{1}{n!}\frac{(a_{ub_1}+...+a_{ub_n}-ub^n_+)^n}{(b_{ub_1}-a_{ub_1})...(b_{ub_n}-a_{ub_n})} \end{aligned}$$

$$\begin{aligned} S^{n+}_{lb_+} &= \frac{1}{((b_{lb_1}+...+b_{lb_n}))-(a_{lb_1}+...+a_{lb_n})} \\ S^{n+}_{ub_+} &= \frac{1}{(b_{ub_1}+...+b_{ub_n})-(a_{ub_1}+...+a_{ub_n})} \end{aligned}$$

Example 6.13. *Let **I** and **J** be two* cdf*-intervals such that* $\boldsymbol{I} = [(1, 0.8, 0.099), (2, 0.03, 0.02)]$ *and* $\boldsymbol{J} = [(3, 0.7, 0.1), (6, 0.02, 0.01)]$. *The computed p-box* cdf *interval addition is exerted on the* cdf *bound points defined in the 2D-space (values and knowledge about their occurrence). Fig. 6.9 illustrates the arithmetic addition p-box* cdf*-interval relation* $\boldsymbol{I}+\boldsymbol{J}$. *Result of the operation is specified by the* cdf*-interval* $[r_{a+c}, r_{b+d}] =$

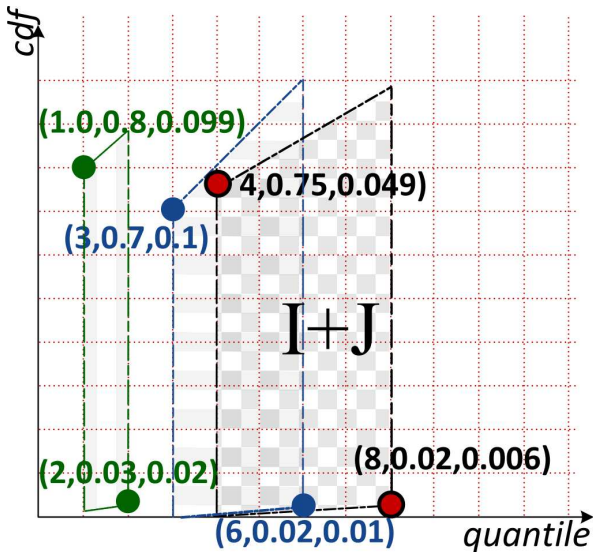


Fig. 6.9: The *cdf*-interval addition

$[(4, 0.75, 0.049), (8, 0.02, 0.006)]$. *Note that in the absence of* cdf *knowledge distribution, where candidate intervals have equal uncertainty weights, we obtain a real interval arithmetic addition* $[4, 8]$. *The additional knowledge p-box* cdf*-interval representation introduces is the fact that any point lying within the interval bounds has an average step probability value varying within* $[0.6\%, 4.9\%]$ *with a minimum probability value between* $[0\%, 2\%]$. *Clearly, the span of the p-box* cdf*-interval domain resulting from the addition operation increased along the quantiles. It indicates that the* cdf *bounded distributions, covering more real quantiles in a uniform manner, have decreased in values (minimum and maximum probability distributions) along with slopes (the average step probabilistic values).*

Multiplication '$\times_{\mathcal{RU}}$'

The multiplication operation, of $\mathbf{I} = [p_a, p_b]$ and $\mathbf{J} = [q_c, q_d]$, is exerted on pair of points belonging to the intervals in a binary relation. The result of this operation is enveloped by a p-box *cdf*-interval whose bounds are defined as

$$[(lb_{\times}, F^{I\times J}_{lb_{\times}}, S^{I\times J}_{lb_{\times}}), (ub_{\times}, F^{I\times J}_{ub_{\times}}, S^{I\times J}_{lb_{\times}})]$$

Operation on quantile values follows the conventional real interval arithmetic multiplication. The lower and upper bounds are defined by $lb_{\times}$ and $ub_{\times}$. Recall that data values can be negative:

$$lb_{\times} = \min(a \times c, a \times d, b \times c, c \times d) \text{ and } ub_{\times} = \max(a \times c, a \times d, b \times c, c \times d)$$

The *cdf*-distributions which envelop the data whereabouts of the interval resulting from the multiplication operation of $\mathbf{I} \times \mathbf{J}$ can be computed as detailed in A.4.2. They

are defined as:

$$F_{lb_\times}^{I\times J} = \int_{a_{lb}\times c_{lb}}^{lb_\times} f_{IJ}(z)dz$$
$$F_{ub_\times}^{I+J} = \int_{a_{ub}\times c_{ub}}^{ub_\times} f_{IJ}(z)dz \tag{6.22}$$

The slope of the *cdf*-distributions are

$$S_{lb_\times}^{I\times J} = \frac{1}{IJ_{lb}^u - IJ_{lb}^l} \; S_{ub_\times}^{I\times J} = \frac{1}{IJ_{ub}^u - IJ_{ub}^l}$$

Where $IJ_{lb}^l = \min\{a_{lb}c_{lb}, a_{lb}d_{lb}, b_{lb}c_{lb}, b_{lb}d_{lb}\}$, $IJ_{lb}^u = \max\{a_{lb}c_{lb}, a_{lb}d_{lb}, b_{lb}c_{lb}, b_{lb}d_{lb}\}$, $IJ_{ub}^l = \min\{a_{ub}c_{ub}, a_{ub}d_{ub}, b_{ub}c_{ub}, b_{ub}d_{ub}\}$ and $IJ_{ub}^u = \max\{a_{ub}c_{ub}, a_{ub}d_{ub}, b_{ub}c_{ub}, b_{ub}d_{ub}\}$. Appendix A.4 derives the proof of the above equations. To compute, a_{lb}, c_{lb}, b_{lb}, d_{lb}, a_{ub}, c_{ub}, b_{ub}, d_{ub} , the bounds of the *cdf* distributions which envelop **I** and **J** we refer to the linear Equation A.10. Each of these bounds has a *pdf* value equal to 0.

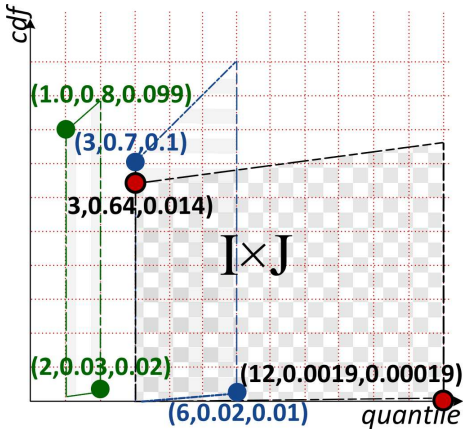


Fig. 6.10: The *cdf*-interval multiplication

Example 6.14. *The p-box* cdf*-interval which results from the multiplication operation over* $\boldsymbol{I}\times\boldsymbol{J} = [(1, 0.8, 0.099), (2.0, 0.03, 0.02)]\times[(3.0, 0.7, 0.1), (6, 0.02, 0.01)]$ *is computed in the 2D-space by computing the p-box* cdf*-point multiplication of the interval bounding points as detailed in Appendix A.4.2.*

The result of multiplying $\boldsymbol{I}\times\boldsymbol{J}$ *is the depicted p-box* cdf*-interval in Fig. 6.10.* $[r_{lb_\times}, r_{ub_\times}] = [(3, 0.64, 0.014), (12, 0.0019, 0.00019)]$. *Observably, the span of quantiles along this interval has increase and the average step values obtained are* [0.019%, 1.4%]. *Probabilities, in this case, are fairly distributed along a higher range of quantiles.*

Subtraction '$-_{\mathcal{R}_\mathcal{U}}$'

The p-box *cdf*-interval which results from the difference operation, of $\mathbf{I} = [p_a, p_b]$ and $\mathbf{J} = [q_c, q_d]$ is defined as

$$[(lb_-, F_{lb_-}^{I-J}, S_{lb_-}^{I-J}), (ub_-, F_{ub_-}^{I-J}, S_{lb_-}^{I-J})]$$

$lb_{-} = \min(a-c, a-d, b-c, c-d)$ and $ub_{-} = \max(a-c, a-d, b-c, c-d)$

$$F^{I-J}_{lb_{-}} = \int_{a_{lb}-d_{lb}}^{lb_{-}} f_{IJ}(z)dz$$
$$F^{I-J}_{ub_{-}} = \int_{a_{ub}-d_{ub}}^{ub_{-}} f_{IJ}(z)dz \tag{6.23}$$

$$S^{I-J}_{lb_{-}} = \frac{1}{IJ^{u}_{lb} - IJ^{l}_{lb}} \; S^{I-J}_{ub_{-}} = \frac{1}{IJ^{u}_{ub} - IJ^{l}_{ub}}$$

Where $IJ^{l}_{lb} = \min\{a_{lb}-c_{lb}, a_{lb}-d_{lb}, b_{lb}-c_{lb}, b_{lb}-d_{lb}\}$, $IJ^{u}_{lb} = \max\{a_{lb}-c_{lb}, a_{lb}-d_{lb}, b_{lb}-c_{lb}, b_{lb}-d_{lb}\}$, $IJ^{l}_{ub} = \min\{a_{ub}-c_{ub}, a_{ub}-d_{ub}, b_{ub}-c_{ub}, b_{ub}-d_{ub}\}$ and $IJ^{u}_{ub} = \max\{a_{ub}-c_{ub}, a_{ub}-d_{ub}, b_{ub}-c_{ub}, b_{ub}-d_{ub}\}$.

The derivation of the above equations are shown in Appendix A.5. Subtractions on the bounding points are exerted. Resulting p-box *cdf*-points enclose all possible *cdf* distributions resulting from the point-wise subtractions of points lying within the two intervals under consideration.

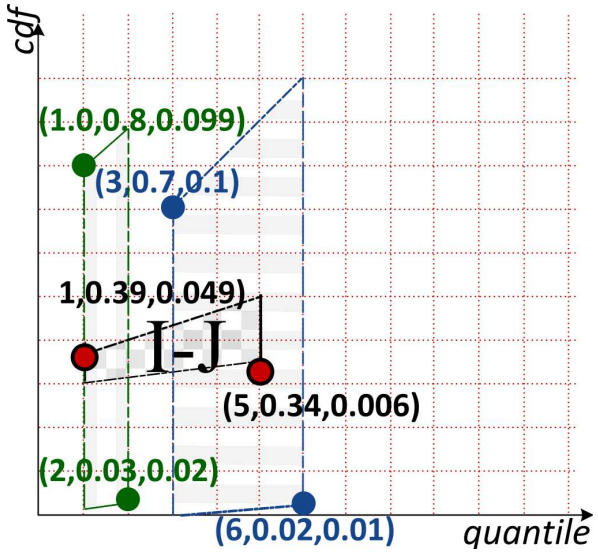


Fig. 6.11: The *cdf*-interval subtraction

Example 6.15. *The p-box* cdf*-interval difference between* ***I*** *and* ***J*** *in the relation* $\boldsymbol{J-I}$ *computes the p-box* cdf*-interval which envelops the output of the subtraction operation* $\boldsymbol{J-I}$ *is shown in Fig. 6.11.* $[r_{lb_{-}}, r_{ub_{-}}] = [(1, 0.39, 0.049), (5, 0.34, 0.006)]$. *Any point lying within the interval bounds has an average step probability value varying within* $[6\%, 4.9\%]$ *with a minimum* cdf *value* 0.3. *Notice that this* cdf *value is obtained by projecting the quantile of the lower bound onto the dominant* cdf *distribution.*

Division '$\div_{\mathcal{RU}}$'

The ratio operation, of $\mathbf{I} = [p_a, p_b]$ and $\mathbf{J} = [q_c, q_d]$ yields a p-box *cdf*-interval whose bounds are defined as

$$[(lb_{\div}, F^{I\div J}_{lb_{\div}}, S^{I\div J}_{lb_{\div}}), (ub_{\div}, F^{I\div J}_{ub_{\div}}, S^{I\div J}_{lb_{\div}})]$$

$$lb_{\div} = \min(a \div c, a \div d, b \div c, c \div d) \text{ and } ub_{\div} = \max(a \div c, a \div d, b \div c, c \div d)$$

$$\begin{aligned} F^{I\div J}_{lb_{\div}} &= \int_{\frac{a_{lb}}{d_{lb}}}^{lb_{\div}} f_{XY}(z)dz \\ F^{I\div J}_{ub_{\div}} &= \int_{\frac{a_{ub}}{d_{ub}}}^{ub_{\div}} f_{XY}(z)dz \end{aligned} \tag{6.24}$$

$$S^{I\div J}_{lb_{\div}} = \frac{1}{IJ^u_{lb} - IJ^l_{lb}} \; S^{I\div J}_{ub_{\div}} = \frac{1}{IJ^u_{ub} - IJ^l_{ub}}$$

Where $IJ^l_{lb} = \min\{\frac{a_{lb}}{c_{lb}}, \frac{a_{lb}}{d_{lb}}, \frac{b_{lb}}{c_{lb}}, \frac{b_{lb}}{d_{lb}}\}$, $IJ^u_{lb} = \max\{\frac{a_{lb}}{c_{lb}}, \frac{a_{lb}}{d_{lb}}, \frac{b_{lb}}{c_{lb}}, \frac{b_{lb}}{d_{lb}}\}$, $IJ^l_{ub} = \min\{\frac{a_{ub}}{c_{ub}}, \frac{a_{ub}}{d_{ub}}, \frac{b_{ub}}{c_{ub}}, \frac{b_{ub}}{d_{ub}}\}$ and $IJ^u_{ub} = \max\{\frac{a_{ub}}{c_{ub}}, \frac{a_{ub}}{d_{ub}}, \frac{b_{ub}}{c_{ub}}, \frac{b_{ub}}{d_{ub}}\}$.

Appendix A.6 details the proof of the above equations.

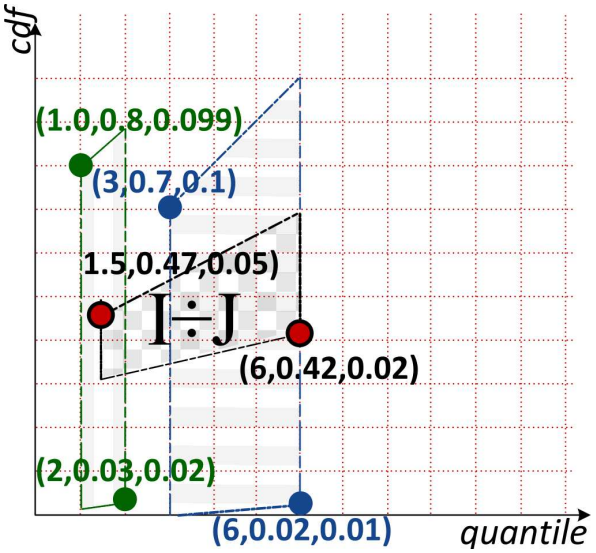


Fig. 6.12: The *cdf*-interval division

Example 6.16. *The computation of the p-box* cdf *ratio* $\boldsymbol{J \div I}$ *is listed in Appendix A.6.2*

The resulting p-box cdf*-interval is depicted in Fig. 6.12. It is specified by the cdf-interval* $[r_{lb_{\div}}, r_{ub_{\div}}] = [(1.5, 0.47, 0.05), (6, 0.42, 0.02)]$. *Clearly the slope of the lower bound distribution increased in this case. This is due to the fact that the span of the quantile bounds have shrinked because of the division operation.*

Noticeably, the core operations, computed over p-box *cdf*-intervals, adopt the computation in the 2D-space: real arithmetic and probabilistic computation. The latter operations are proved to be linear, in Appendix A.3, A.4, A.5 and A.6 due to the *cdf* monotonic property and since we exert these operations on linear uniform distributions which enclose the intervals under consideration. The computation seek to find two values for the *cdf* bounds and their slopes. Obtained domains are proved to have a convex algebraic structure which encloses an unknown probability distribution. Examples 6.13, 6.14, 6.15 and 6.16 further ellaborate that shrinking the real quantile bounds yields an increase in the range of *cdf* distributions bounding the resultant interval and vise versa. This is due to the fact that the uniform distribution fairly divides the average step probabilistic value over the quantile range. Yet the output interval encloses the unknown probability distribution due to the probability theory in Ferson et al. (2003); Williamson and Downs (1990).

CHAPTER 7

PRACTICAL FRAMEWORK

In the constraint programming paradigm relations between variables are specified as constraints. A set of rules and algebraic semantics, defined over the list of constraints, formalize the reasoning about the problem. As a fundamental language component in CLP, these set of rules, with a syntax of definite clauses, form the language scheme Jaffar and Lassez (1987). The constraint solving scheme is intuitively and efficiently utilized in the reasoning over the computation domain. The scheme formally attempts at assigning to variables a suitable domain of discourse equipped with an equality theory together with a least and a greatest model of fix-point semantics. Starting from an initial state the reasoning scheme follows a local consistency technique which attempts at constraining each variable over the *cdf*-interval domain while excluding values not belonging to the feasible solution.

7.1 The *cdf*-interval language scheme

Definition 7.1. *The system of constraints assigns each variable a cdf-interval range from* $R_{\mathcal{U}}$

1. *function symbols, mapping variables to cdf-intervals, are* $\in \{+_{\mathcal{U}}, \times_{\mathcal{U}}, -_{\mathcal{U}}, \div_{\mathcal{U}}\}$

2. *relation symbols are* $\in \{=_{\mathcal{U}}, \preccurlyeq_{\mathcal{U}}\}$

3. $X \in [p_a, p_b]$ *constrains the variable X to a domain range such that*
   $p_a \preccurlyeq_{\mathcal{U}} p_x \preccurlyeq_{\mathcal{U}} p_b$

4. $C_{\mathcal{U}} \preccurlyeq_{\mathcal{U}} C_{\mathcal{U}1}$ *is in the set of constraints where* $C_{\mathcal{U}}$ *and* $C_{\mathcal{U}1}$ *are specified as expressions*
   (i.e. constants, variables and operations $\in \{glb_{\mathcal{U}}, lub_{\mathcal{U}}, \odot\}$*)*

The constraint system behaves like a solver over real intervals. It is based on the relational arithmetic of real intervals where arithmetic expressions are interpreted as relations Cleary (1987). Variables in a relation are expressed in terms of others and subject to *cdf*-interval domain refinement using local consistency techniques. To simplify the complexity of the operations, n-ary constraints are decomposed into primitive ones which contain at most two *cdf*-interval variables in a relation.

The notion of arc consistency in Mackworth (1977) ensures that, for a binary relation, every pair of points enveloped by the domain of variables beloging to the relation must be satisfied. This notion, however, consists of an infinite number of checks, when adopted in the *cdf*-interval algebraic structure, since the domain encloses an infinitesimal number of elements in the 2D-space. The *cdf*-interval scheme instead adequately seeks the local consistency on the interval bounds and this implies that every pair of points lying within the bounds satisfy the system local consistency. Domain holes are not taken into consideration because they increase the system computational complexity.

Property 7.1. *Let $C_1 \preccurlyeq_{\mathcal{U}} C_2$ be a primitive* cdf *constraint. This binary constraint is local consistent if and only if:*

*C*1. $glb_{\mathcal{U}}(C_1) \preccurlyeq_{\mathcal{U}} glb_{\mathcal{U}}(C_2)$ *and*

*C*2. $lub_{\mathcal{U}}(C_1) \preccurlyeq_{\mathcal{U}} lub_{\mathcal{U}}(C_2)$

7.2 The *cdf*-interval inferrence rules

Operations which refine the *cdf*-interval domain exclude elements from the domain in order to satisfy the conditions defined by the local consistency notion. This refinement is also called domain pruning and it is intuitively exerted on the *cdf*-interval domain bounds defined in the 2D-space. Relations are handled using the transformation rules which extend those defined over the real intervals coupled with inferences over the bounds of the *cdf* distributions. The solver converges to a fixed point or infers failure. We ensure termination of the generic constraint propagation algorithm because the *cdf*-interval domain ordering $\preccurlyeq_{\mathcal{U}}$ is reflexive, antisymmetric and transitive. Hereafter we present the main transformation rules for the basic arithmetic operations. When a domain remains unchanged we will use the following notation: $\mathbf{I} = [p_a, p_b]$, $\mathbf{J} = [q_c, q_d]$ and $\mathbf{K} = [r_e, r_f]$. The *cdf*-variables are denoted by X, Y and Z and their initial binding is set to $X = \mathbf{I}$, $Y = \mathbf{J}$ and $Z = \mathbf{K}$. We attach a ($'$) to points on the bounds when they are subject to domain prunning (adjustment due to the operation). Failure is detected if some domain bounds do not preserve the ordering $\preccurlyeq_{\mathcal{U}}$.

7.2.1 Ordering constraint $X \preccurlyeq_{\mathcal{U}} Y$

To infer the local consistency of the binary ordering constraint, we extend the lower *cdf*-bound of X and contract the upper *cdf*-bound of Y. This is to maintain the domain

interval ordering described previously in Section 6.3.2. To infer about the ordering constraint we use the following rule:

$$\frac{p_b' = glb(p_b, q_d), q_c' = lub(p_a, q_c)}{\{X \in \mathbf{I}, Y \in \mathbf{J}, X \preccurlyeq_{\mathcal{U}} Y\} \longmapsto \{X \in [p_a, p_b'], Y \in [q_c', q_d], X \preccurlyeq_{\mathcal{U}} Y\}}$$

Example 7.1. *Let the initial* cdf *domain binding for* $X \in [(2.0, 0.4, 0.8), (6.0, 0.2, 0.05]$ *and* $Y \in [(1.0, 0.6, 0.7), (5.0, 0.1, 0.06)]$*, the inequality constraint* $X \preccurlyeq_{\mathcal{U}} Y$*, as shown in Fig. 7.1, prunes the domain of* X *from the upper bound and the domain of* Y *from the lower bound. The output domains from this operation are* $X \in [(2.0, 0.4, 0.8), (5.0, 0.1, 0.06)]$ *and* $Y \in [(2.0, 0.4, 0.8), (5.0, 0.1, 0.06)]$*. Clearly resulting domains preserve the convex property of the p-box* cdf*-intervals.*

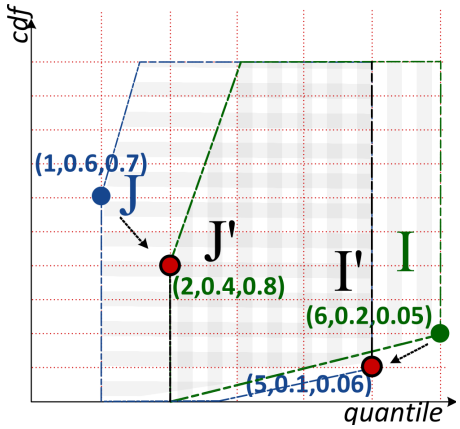


Fig. 7.1: Ordering constraint execution. Initial bindings are $X \in \mathbf{I}$, $Y \in \mathbf{J}$.

7.2.2 Equality constraint $X = Y$

The binary equality constraint seeks at refining extreme points bracketing the domains of both variables such that resulting domains are identical. Resulting domains from the equality are $X, Y \in [p_a', p_b']$ after conducting the intersection on the original domains. This is exerted using the following inference rule:

$$\frac{p_b' = glb(p_b, q_d), p_a' = lub(p_a, q_c)}{\{X \in \mathbf{I}, Y \in \mathbf{J}, X = Y\} \longmapsto \{X \in [p_a', p_b'], Y \in [p_a', p_b'], X = Y\}}$$

7.2.3 Ternary addition constraints $X +_{\mathcal{U}} Y = Z$

The addition operation is implemented by summing up pair of points, defined in the 2D space and located within the *cdf*-interval bounds which enclose the domain ranges of X and Y. This operation is convex and can be computed from the end points of the

domains involved in the addition. The *cdf*-domain of Z is updated to envelop all points defined in that range. The inference rule applied in this operation maps the Cartesian product of the *cdf* initial binding domains involved in the relation ($\mathbf{I} \times \mathbf{J} \times \mathbf{K}$) to a new Cartesian product ($\mathbf{I}' \times \mathbf{J}' \times \mathbf{K}'$). The resulting Cartesian product $\mathcal{R}^3_{\mathcal{U}}$ includes all possible three-point-set which satisfy the local consistency of the constraint and which is specified as $+_{\mathcal{U}} \equiv \{(p_x, q_y, r_z) : p_x, q_y, r_z \in \mathcal{U}, p_x + q_y = r_z\}$. Notice that initial bounds, enveloping the domain of Z, affect this triplet-point set. The resulting full set of solutions is given by $+_{\mathcal{U}} \bigcap \mathbf{I} \times \mathbf{J} \times \mathbf{K}$. To obtain the output domain bindings, we project the Cartesian product onto each variable domain involved in the relation. The output Cartesian product when projected on the domain of X yields the set of elements specified as $\{p_x : \exists q_y, r_z (p_x, q_y, r_z) \in +_{\mathcal{U}} \bigcap \mathbf{I} \times \mathbf{J} \times \mathbf{K}\}$. To infer about the *cdf* ternary addition constraint we use the following:

$$\frac{r_f{}' = (ub_+, F^{I+J}_{ub_+}, S^{I+J}_{ub_+}), r_e{}' = (lb_+, F^{I+J}_{lb_+}, S^{I+J}_{lb_+})}{\{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in \mathbf{K}, Z = X +_{\mathcal{U}} Y\} \longmapsto \{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in [r_e{}', r_f{}'], Z = X +_{\mathcal{U}} Y\}}$$

$$\frac{p_b{}' = (ub_-, F^{K-J}_{ub_-}, S^{K-J}_{ub_-}), p_a{}' = (lb_-, F^{K-J}_{lb_-}, S^{K-J}_{lb_-})}{\{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in \mathbf{K}, X = Z -_{\mathcal{U}} Y\} \longmapsto \{X \in [p_a{}', p_b{}'], Y \in \mathbf{J}, Z \in \mathbf{K}, Z = Z -_{\mathcal{U}} Y\}}$$

The projection onto the Y domain is symmetrical.

Example 7.2. *Table 7.1 and Fig. 7.2 depict the execution steps of the* cdf *ternary addition inference rules, exerted on the variable domains involved in the relation $Z = X +_{\mathcal{U}} Y$. Observe that domain pruning is performed in a* 2 *dimensional manner: quantile and* cdf*. The addition of the two variables X and Y is performed on the bounds of their predefined domains then it is projected onto the initial bindings. The first row in Fig. 7.2 shows output domains from the addition $I + J$, $K - J$ and $K - I$. Domain operations are exerted on the extreme points. The second row illustrates the intersection of the output domains with the initial bindings, assigned to Z, X and Y. Obtained domains from the ternary addition operation are $\boldsymbol{K}'$, $\boldsymbol{J}'$ and $\boldsymbol{I}'$. Clearly in this example pruning real quantile bounds is identical to that of real domains and since output domains preserve the stochastic dominance property no further pruning takes place.*

Example 7.3. *Table 7.1 and Fig. 7.3 show another example which incorporates different* cdf *distribution bounds issued from the same real quantiles. The property of these distributions were tweaked to affect the output domains pruning. This is due to the fact that the model seeks to preserve the convexity of the p-box* cdf *intervals. Pruned domains resulting from the operation detailed in this example have different real lower bound quantiles:* 1.68 *and* 2.55 *for X and Y respectively.*

Initial bindings (Example 7.2)	**X**	**Y**	**Z**
$X +_{\mathcal{U}} Y =$ $[(0, 0.6, 0.099), (2, 0.03, 0.01)] +_{\mathcal{U}}$ $[(1, 0.7, 0.098), (3, 0.1, 0.04)]$ $Z -_{\mathcal{U}} X =$ $[(4, 0.8, 0.05), (6, 0.05, 0.008)] -_{\mathcal{U}}$ $[(0, 0.6, 0.099), (2, 0.03, 0.01)]$ $Z -_{\mathcal{U}} Y =$ $[(4, 0.8, 0.05), (6, 0.05, 0.008)] -_{\mathcal{U}}$ $[(1, 0.7, 0.098), (3.0, 0.1, 0.04)]$	 $[(1, 0.56, 0.033),$ $(5, 0.2, 0.006)]$	 $[(2, 0.6, 0.033),$ $(6, 0.467, 0.004)]$	 $[(1, 0.65, 0.049),$ $(5, 0.044, 0.008)]$
Output variable domain bounds	$[(1, 0.56, 0.033),$ $(2.0, 0.03, 0.01)]$	$[(2, 0.6, 0.033),$ $(3.0, 0.1, 0.04)]$	$[(4.0, 0.8, 0.05),$ $(5.0, 0.044, 0.008)]$
Initial bindings (Example 7.3)	**X**	**Y**	**Z**
$X +_{\mathcal{U}} Y =$ $[(0.0, 0.4, 0.25), (2.0, 0.1, 0.06)] +_{\mathcal{U}}$ $[(1.0, 0.5, 0.2), (3.0, 0.2, 0.05)]$ $Z -_{\mathcal{U}} X =$ $[(4.0, 0.3, 0.9), (6.0, 0.04, 0.02)] -_{\mathcal{U}}$ $[(0.0, 0.4, 0.25), (2.0, 0.1, 0.06)]$ $Z -_{\mathcal{U}} Y =$ $[(4.0, 0.3, 0.9), (6.0, 0.04, 0.02)] -_{\mathcal{U}}$ $[(1.0, 0.5, 0.2), (3.0, 0.2, 0.05)]$	 $[(1.62, 0.23, 0.16),$ $(5.0, 0.28, 0.014)]$	 $[(2.45, 0.23, 0.195),$ $(6.0, 0.28, 0.015)]$	 $[(1.0, 0.45, 0.11),$ $(5.0, 0.15, 0.027)]$
Output variable domain bounds	$[(1.62, 0.23, 0.16),$ $(2, 0.1, 0.06)]$	$[(2.45, 0.23, 0.195),$ $(3.0, 0.2, 0.05)]$	$[(4, 0.78, 0.11),$ $(5.0, 0.15, 0.027)]$

Table 7.1: Execution steps of ternary addition constraint $Z = X +_{\mathcal{U}} Y$.

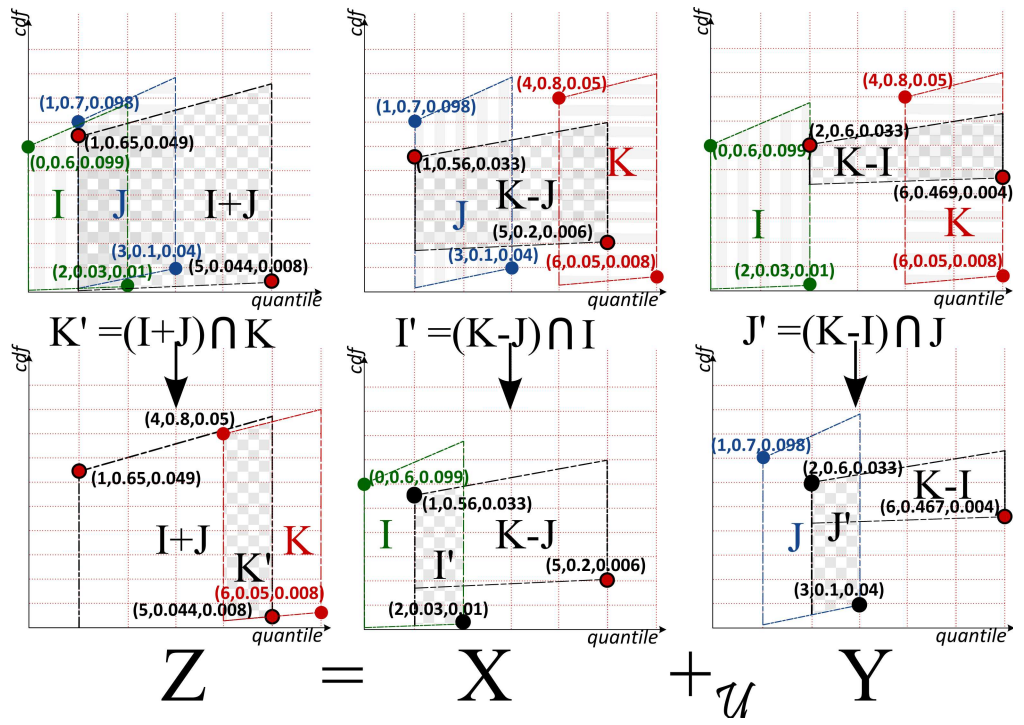


Fig. 7.2: Ternary addition inference rule execution. Initial bindings are $X \in \mathbf{I}$, $Y \in \mathbf{J}$ and $Z \in \mathbf{K}$. Final bindings are $X \in \mathbf{I}'$, $Y \in \mathbf{J}'$ and $Z \in \mathbf{K}'$.

7.2.4 Ternary multiplication constraint $X \times_{\mathcal{U}} Y = Z$

The ternary multiplication is not naturally convex, yet Cleary (1987) introduced an algorithm which seeks an output convex domain from the multiplication. The algorithm is based on splitting the domains, especially when they are bounded by quantiles equal to 0; the problem, in this case, is solved as separate sub-problems; then, resulting domains are unified. Ternary multiplication on p-box *cdf*-intervals domains is first performed on real quantiles using real interval domain multiplication. Resulting quantile bounds are then projected on the *cdf* domain. Inferring about the product of 3 *cdf* bounding domains yields a set of all possible triplets taking their values from the domains ($\mathbf{I}' \times \mathbf{J}' \times \mathbf{K}'$) and specified by the set $\times_{\mathcal{U}} \equiv \{(p_x, q_y, r_z) : p_x, q_y, r_z \in \mathcal{U}, p_x \times q_y = r_z\}$. Elements of X projected on the output Cartesian product are specified as $\{p_x : \exists q_y, r_z (p_x, q_y, r_z) \in \times_{\mathcal{U}} \bigcap \mathbf{I} \times \mathbf{J} \times \mathbf{K}\}$. The ternary multiplication is described by the following inference rules.

$$\frac{r_f{}' = (ub_\times, F^{I\times J}_{ub_\times}, S^{I\times J}_{ub_\times}), r_e{}' = (lb_\times, F^{I\times J}_{lb_\times}, S^{I\times J}_{lb_\times})}{\{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in \mathbf{K}, Z = X \times_{\mathcal{U}} Y\} \longmapsto \{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in [r_e{}', r_f{}'], Z = X \times_{\mathcal{U}} Y\}}$$

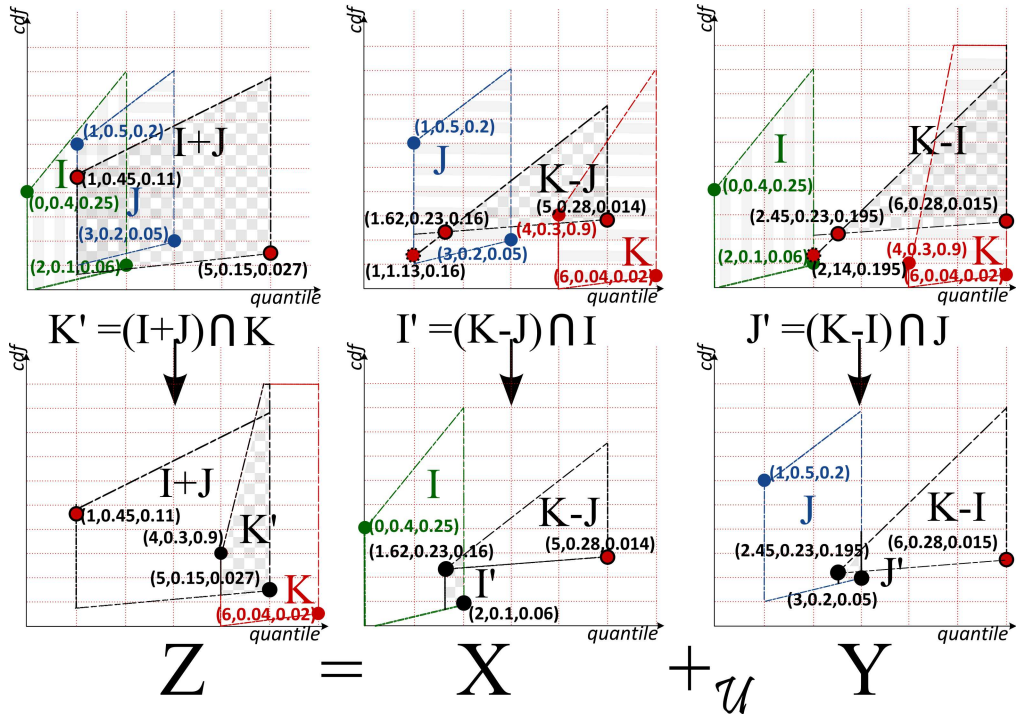


Fig. 7.3: Ternary addition inference rule execution. Initial bindings are $X \in \mathbf{I}$, $Y \in \mathbf{J}$ and $Z \in \mathbf{K}$. Final bindings are $X \in \mathbf{I'}$, $Y \in \mathbf{J'}$ and $Z \in \mathbf{K'}$. Initial bindings have identical quantiles in the real domain but final bindings are further pruned to maintain the *cdf* stochastic ordering property

$$\frac{p_b{}' = (ub_{\div}, F^{K \div J}_{ub_{\div}}, S^{K \div J}_{ub_{\div}}),\ p_a{}' = (lb_{\div}, F^{K \div J}_{lb_{\div}}, S^{K \div J}_{lb_{\div}})}{\{X \in \mathbf{I}, Y \in \mathbf{J}, Z \in \mathbf{K}, X = Z \div_{\mathcal{U}} Y\} \longmapsto \{X \in [p_a{}', p_b{}'], Y \in \mathbf{J}, Z \in \mathbf{K}, Z = Z \div_{\mathcal{U}} Y\}}$$

The projection onto Y's domain is symmetrical.

Example 7.4. *Fig. 7.4 depicts the execution of the ternary multiplication* $Z = X \times_{\mathcal{U}} Y$ *where initial domain bindings are specified as* $X \in [(0, 0.6, 0.099), (2, 0.02, 0.01)]$, $Y \in [(-1, 0.7, 0.098), (3, 0.1, 0.04)]$ *and* $Z \in [(-3.0, 0.8, 0.05), (7.0, 0.05, 0.008)]$. *Observably, domains of X and Y remain unchanged but domain of Z was pruned to* $\in [(-2, 0.37, 0.012), (6, 0.042, 0.008)]$ *due to the multiplication operation.*

7.3 Operational semantics of the *cdf*-intervals

In practice, a problem has a set constraints system, each constraint is defined over a set of variables, each of which is defined over an interval domain. Constraints may share some variables. The p-box *cdf*-interval domains, involved in a relation, are subject

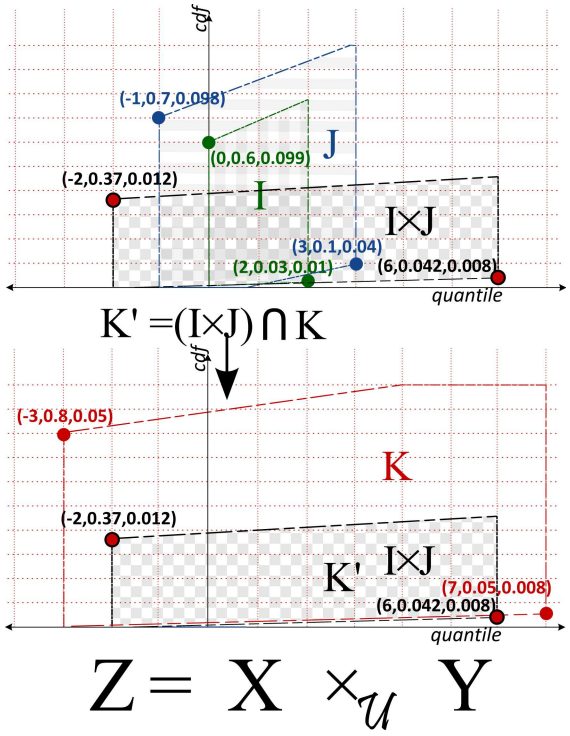


Fig. 7.4: Ternary multiplication inference rule execution.

to be pruned by the list of constraints incorporating variables bind to these domains. The operational semantic of the p-box *cdf*-intervals follows the relaxation algorithm of fixed-point semantics proposed for real intervals in [Lee and Van Emden (1993)]. The algorithm, as described in Algorithm 3, is performed as a list of state transitions. It defines two sets of constraint list: passive $\mathcal{P}_{\mathcal{U}}$ in which all constraints are stable (i.e. approved to be local consistent) and active $\mathcal{A}_{\mathcal{U}}$ which contains all constraints subject to the consistency check. The set of domain constraints is symbolized by $\mathcal{R}_{\mathcal{U}}$. An arbitrary state i is specified as $\langle \mathcal{A}_{\mathcal{U}i}, \overrightarrow{\mathcal{R}_{\mathcal{U}i}}, \overrightarrow{\mathcal{P}_{\mathcal{U}i}} \rangle$. The initial state lists all the problem constraints in $\mathcal{A}_{\mathcal{U}}$, it is specified as $\langle \mathcal{A}_{\mathcal{U}0}, \emptyset, \emptyset \rangle$ and the final state is either a 'fail' if one of the domain constraints is empty or $\langle \emptyset, \mathcal{R}_{\mathcal{U}i}', \mathcal{P}_{\mathcal{U}}' \rangle$, where all constraints have been checked for local consistency and moved from the active list to the constraints store $\mathcal{C}_{\mathcal{U}}$. The only difference in the operation lies in the execution of inference rules which is performed in the two 2D-space. For any system of constraints, if its defined inference rules are characterized to contract variable domains while being idempotent, Older and Vellino (1993) proved that fixed-point semantic algorithm always terminates irrespective of the ordering of the inference rules execution.

For the set of p-box *cdf*-interval variables $\mathcal{V}_{\mathcal{U}}$, the relaxation algorithm is described in Algorithm 3

1: Initialize $\mathcal{A}_{\mathcal{U}}$ to the list of all constraints in the network
2: Initialize $\mathcal{P}_{\mathcal{U}}$ to the empty list
3: **while** $\mathcal{A}_{\mathcal{U}}$ is not empty **do**
4: remove the first constraint $\langle \varrho, \overrightarrow{\mathcal{V}_{\mathcal{U}}} \rangle$ from $\mathcal{A}_{\mathcal{U}}$
5: apply domain narrowing using *cdf* inference rules on $\langle \varrho, \overrightarrow{\mathcal{V}_{\mathcal{U}}} \rangle$ to obtain $\overrightarrow{\mathcal{V}'_{\mathcal{U}}}$.
6: **if** interval narrowing fails **then**
7: Exit with failure
8: **else if** $\overrightarrow{\mathcal{V}_{\mathcal{U}}} \neq \overrightarrow{\mathcal{V}_{\mathcal{U}}'}$ **then**
9: $\overrightarrow{\mathcal{V}_{\mathcal{U}}} \leftarrow \overrightarrow{\mathcal{V}_{\mathcal{U}}'}$
10: **for each** constraint $\langle \xi, \overrightarrow{\mathcal{Y}_{\mathcal{U}}} \rangle$ in $\mathcal{P}_{\mathcal{U}}$ **do**
11: **if** $\overrightarrow{\mathcal{V}_{\mathcal{U}}}$ and $\overrightarrow{\mathcal{Y}_{\mathcal{U}}}$ share narrowed variable(s) **then**
12: remove $\langle \xi, \overrightarrow{\mathcal{Y}_{\mathcal{U}}} \rangle$ from $\mathcal{P}_{\mathcal{U}}$ and append it to $\mathcal{A}_{\mathcal{U}}$
13: append $\langle \varrho, \overrightarrow{\mathcal{V}_{\mathcal{U}}} \rangle$ to the end of $\mathcal{P}_{\mathcal{U}}$ which maps the *cdf* interval variable domain $\mathcal{V}_i$ to $\mathcal{R}_{\mathcal{U}}$

Algorithm 3: relaxation algorithm

7.3.1 The design of the *cdf*-interval solver

An implementation of the constraint system was established as a separate module in the ECL^{i}PS^e constraint programming environment ECRC (1994). ECL^{i}PS^e provides two major components to build the solver: attributed variable data structure and suspension handling mechanism. Fundamentally, attributed variables are specific data structures which attach more than one data type. Together they permit for a new definition of unification which extends the well-known Prolog unification Le Huitouze (1990); Holzbaur (1992). A *cdf*-interval point is implemented in an attributed variable data structure which encompasses three main constituents: quantile, *cdf* value and slope. Whilst constraints suspension handling is a highly flexible mechanism that aims at controlling user defined atomic goals. This is achieved by waiting for user-defined conditions to trigger specific goals. Appendix B describes part of the solver source code and shows a list of query/answer examples which are implemented using the list inference rules detailed in Section 7.2.

The *cdf*-interval solver roots and derivations

The *cdf*-interval language inherits its syntactic features from Prolog. A function or a predicate can be implemented equivalently in two formats: infix or postfix. This feature allows us to implement arithmetic operators in the infix format (for example $+(p_a, q_b)$ and $p_a + q_b$ are equivalent). Hence we can seek for the development of a data-driven language which intuitively realizes meta-programming.

cdf-interval domains are defined over reals:

1. quantiles range $[-\infty, +\infty]$, the *cdf* values range $[0, 1]$ and the slope values range $[0, +\infty]$, by definition.

2. The ‘`...`‘ *cdf* domain range operator take the bounding *cdf* points as parameters to indicate that the interval ranges between the bounding points. This operator is defined in the infix format

3. The { `~`, `|` } operators append the uncertainty components to the real quantile: *cdf* value and slope

For example a *cdf*-interval domain range is expressed by the formula ‘`2~0.7|0.86...3~0.1|0.028`‘ in *cdf*-interval solver. We define *cdf*-interval linear constraints to incorporate the following:

1. *cdf*-interval constant expressions of the form ‘`2~0.7|0.86...3~0.1|0.028`‘

2. *cdf*-interval coefficients each multiplied by the *cdf*-variables

3. ‘`X.::A~FA|SA...B~FB|SB`‘ is semantically equivalent to $p_a \preccurlyeq_{\mathcal{U}} X \preccurlyeq_{\mathcal{U}} q_b$ given that p_a and q_b are p-box *cdf*-interval points specified as (a, F_a^p, S_a^p) and (b, F_b^q, S_b^q) respectively.

4. Binary relations are specified as { `.<, .=< , .=, .>, .>=` }

5. { `+,-` } are binary operations defined by the infix notation. { `+` } and { `-` } are left associative and have the same binding order *.

A linear constraint is specified by the formula $(s \ \text{op} \ t)$ where s and t are linear expressions and op $\in$ { `.<, .=< , .=, .>, .>=` } For example ‘`(3~0.8|0.96...4~0.4|0.034)*X + (2~0.7|0.86...3~0.1|0.028) * (Y + X + (6~0.9|0.85...7~0.5|0.029)`‘ is an arithmetic expression and ‘`(3~0.8|0.96...4~0.4|0.034)*X + (2~0.7|0.86...3~0.1|0.028)*Y + X .=< (4~0.9|0.85...5~0.5|0.029)*Z + (6~0.7|0.86...7~0.1|0.028) + Y`‘ is a linear constraint. Arithmetic constraints are the fundamental language feature of the modeling in the constraint programming paradigm. Arithmetic constraints are linear constraints augmented by the binary arithmetic multiplication { `*` }. Any expression of the form ‘`X * Y + (6~0.7|0.86...7~0.1|0.028) .=< (4~0.9|0.85...5~0.5|0.029) * Z`‘ is an arithmetic constraint.

The *cdf*-interval delay mechanisms

We use the Eclipse suspend library in our user-defined goals implementation. Goals containing *cdf* variables are suspended until they are initialized and they are subject to the waking condition of the active constraint when variable domain bounds change. Accordingly, we use ‘`X1->inst`‘, ‘`X1->min`‘ and ‘`X1->max`‘as suspending parameters which

*Binding ordering of arithmetic operators is detailed in [Apt and Wallace (2006)]

identify any changes in the variable domain. By means of constraint propagation techniques we refine the variable domain, moving the bounding *cdf* points. Once this occurs, the list of all system constraints attached to this variable are woken and further domain pruning may be applied. This is exerted by calling the ECL^{i}PS^e built-in '`wake/0`'. The local consistency mechanism ensures variable domain refinement by applying the *cdf*-interval inference rules.

Example 7.5. *The goal* '`X.::3~0.8|0.7...5~0.1|0.02 , Y.::2~0.7|0.87...4~0.4|0.017`', `X.=<Y .`' *produces the refined domains:* '`X.::3~0.8|0.7...4~0.4|0.017`' '`Y.::3~0.8|0.7...4~0.4|0.017`' *and the delayed goal:* '`X .=< Y`'. *More examples can be found in Appendix B*

7.4 Empirical evaluation

To evaluate the added value of the new constraint domain, we considered an example provided in Yorke-Smith and Gervet (2009), and attached *cdf* value and slope to the interval bounds. We tested the system for coefficients lying on the positive quadrant (i.e. they are assigned positive values). Example 8.1 aims at solving a system of linear equations which has *cdf* coefficients and unknown variables having no certainty degree defined, i.e. the lower bound points are $'(0, 1.0, \infty)'$ whereas upper-bound of the variable interval is $'(\infty, 0, 0)'$. Shown below are pruned domains of the variables at fixed point using our inference rules. The *cdf*-intervals attached to the data (here coefficients) were propagated onto the *cdf*-variables, X_1 and X_2. We can see that inferences on the quantile component of the 3-D space point yield similar pruning on the resulting variable domains and the additional information coming from the *cdf* and slope components demonstrate the information gained on the density of occurrence for the resulting points within the *cdf*-domains.

Example 7.6. *Consider the system of linear equations* $(\boldsymbol{A}, \mathfrak{R}, \boldsymbol{b})$ *shown below:*

$$\boldsymbol{A} = \begin{bmatrix} [(-2.0, 0.5, 0.2), (2.0, 0.01, 0.095)] & [(1.0, 0.3, 0.32), (2.0, 0.02, 0.083)] \\ [(-2.0, 0.7, 0.1), (-1.0, 0.01, 0.087)] & [(-1.0, 0.2, 0.3), (-1.0, 0.01, 0.087)] \\ [(6.0, 0.9, 0.98), (6.0, 0.01, 0.018)] & [(1.5, 0.1, 0.6), (3.0, 0.06, 0.034)] \end{bmatrix},$$

$$\mathfrak{R} = \begin{bmatrix} \preccurlyeq_{\mathcal{RU}} \\ = \\ = \end{bmatrix} \text{ and } \boldsymbol{b} = \begin{bmatrix} [(3.0, 0.88, 0.4), (4.0, 0.4, 0.088)] \\ [(-5.0, 0.85, 0.1), (5.0, 0.02, 0.013)] \\ [(4.0, 0.9, 0.02), (15.0, 0.01, 0.001)] \end{bmatrix}$$

*Fig. 7.5 illustrates two skewed intersecting boxes, each encloses the output solution population residence for X*1 *and X*2*. The shown boxes are the result of applying* cdf *propagation techniques on the* cdf *linear equations provided in this example. The black box is the representation of X*1 *solution domain bounded by* $[(0.0, 1.0, +\infty), (5.0, 0.46, 0.07)]$ *and the yellow one is the solution domain of the variable X*2 *given by* $(0.0, 1.0, +\infty), (2.5, 0.28, 0.07)]$*. Clearly, both solutions intersect in the X*1*-X*2 *2D space*

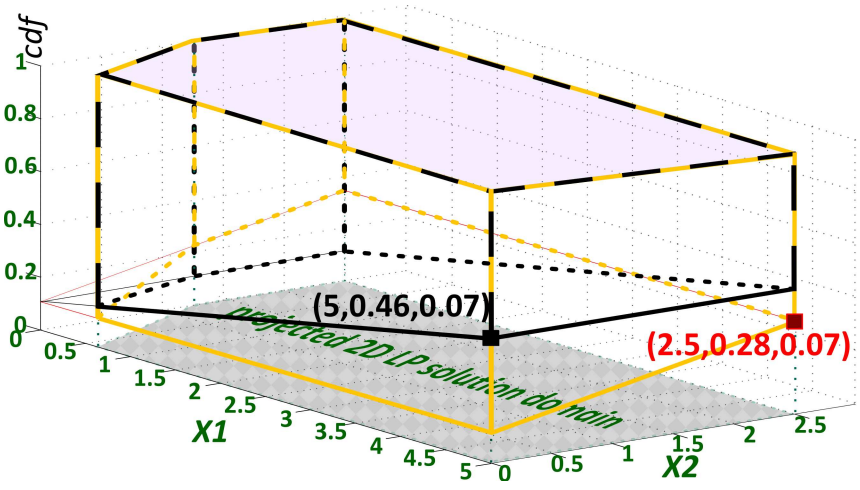


Fig. 7.5: Example 8.1: Solution set resulting from the *cdf* computations

as illustrated by their projection depicted by the shaded checkerboard region. Typically, the 2D projection plane matches their 2D LP solution domain and it represents the real-interval arithmetic solution. This is due to the fact that lateral areas of the boxes share common 2D planes in the 3D space, yet each variable encapsulates a different probabilty distribution. Section B.5, in Appendix B, shows the query/answer syntax for the system of linear equations input to/output from our solver implementation.

CHAPTER 8

P-BOX CDF-INTERVALS GLOBAL CONSTRAINTS

Global constraints have been widely used in the CP literature in order to enhance the reasoning process in terms of efficiency and effectiveness. As part of a constraint solver implementation, they take advantage of the constraints modeling semantics to reason about problem specific data from a global perspective W.-J. van Hoeve and Katriel (2006). Many propositions to formulate global constraints of the CP modeling by means of LP representations have been thoroughly researched in the literature Refalo (2000); Milano, Ottosson, Refalo, and Thorsteinsson (2001); Hnich, Rossi, Tarim, and Prestwich (2011). Such hybridizations proved to inherit the intuitive expressiveness of the CP paradigm while applying the powerful LP optimization techniques. These approaches aim at integrating the LP paradigm into the CP and they proved to perform very well in the deterministic case.

In this chapter, we define the system of global constraints over the p-box *cdf*-intervals algebraic structures, by extending Interval Linear Systems (ILS) with a second dimension: the *cdf*. This new proposition inherits its characteristics from the hybridization techniques found in the literature and surveyed by Refalo (2000); Milano et al. (2001); Hnich et al. (2011).

8.1 P-Box CDF-Intervals LP

The CP characteristic of the model comes from adopting our p-box *cdf*-intervals framework. We build the global constraints, by transforming the problem into an equivalent set of linear equations on the domain of quantiles which, in turn, are solved by the Simplex method Chvátal (1983). We show how the p-box *cdf*-intervals constraint model does generalize the ILS Hansen (1979). Hence, linear systems with p-box *cdf*-interval coefficients and variables can be solved by a simple polynomial transformation into a

linear model. The approach is similar to the ILS with positive coefficients (also called Positive Orthant Linear Interval (POLI) Beaumont (1998). This approach was adopted by Yorke-Smith and Gervet (2009) to extract real bounds of the ILS with interval coefficients in their UCSP algebraic structure. Resulting extreme quantiles, obtained from the POLI, are then projected onto the probability domain in order to deduce bounds on the *cdf* bounding distributions. We show that extreme points obtained in UCSP and *cdf*-ILS are equivalent.

8.1.1 Extended UCSP Transformation (EUCSPT) Algorithm

The transformation algebra prunes output solution domains by extracting extreme points in the interval hull. Hence, we can obtain the full closure of the ILS; which represents the problem in hand along with its data whereabouts.

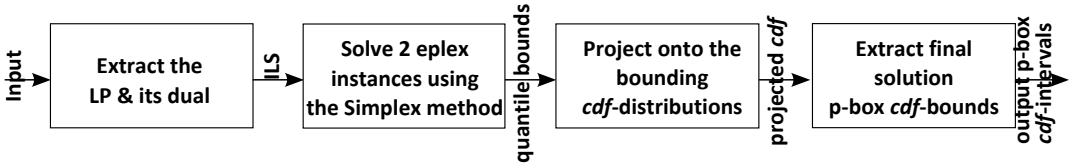


Fig. 8.1: EUCSPT Algorithm

The algorithm as depicted in Fig.8.1 is composed of four main steps:

1. Extract the LP and its dual
2. Solve 2 eplex instances
3. Project the quantiles onto the *cdf* bounding distributions
4. Extract the final solution *cdf*-interval bounds

Definition 8.1. *Let $\mathcal{V}$ be a set of n p-box cdf-variables, and C a set of m linear constraints over $\mathcal{V}$ with p-box cdf-interval coefficients. A p-box cdf-interval Linear System UILS induced by C over $\mathcal{V}$ is a tuple $\langle \boldsymbol{A}, \Re, \boldsymbol{B} \rangle$, where $\boldsymbol{A} \in (\mathbb{R} \times [0..1] \times \mathbb{R}^+)^{m \times n}$ is the matrix of the LHS intervals $[glb_A, lub_A]$, $\boldsymbol{B} \in (\mathbb{R} \times [0..1]\mathbb{R}^+)^m$ is the vector of the RHS intervals $[glb_B, lub_B]$ and $\Re_i \in \{<_{\mathcal{U}}, \leqslant_{\mathcal{U}}, =, \geqslant_{\mathcal{U}}, >_{\mathcal{U}}\}$ $\forall i = 1, .., m$ is a list of m relations defined over p-box cdf-intervals.*

Theorem 8.1. *The Positive Orthant Linear transformation of the cdf-intervals (POLI-CDF) $\rho = \langle \boldsymbol{A}, \Re, \boldsymbol{B} \rangle$ is the complete solution set of the interval linear inequality system $\langle \boldsymbol{A}`X\{\Re\}\boldsymbol{B}` \rangle$ and it is defined as follows:*

$$(A_i`, B_i`) = \left\{ \begin{array}{ll} (glb_{A_i}, lub_{B_i}) & if \ \{<_{\mathcal{U}}, \leqslant_{\mathcal{U}}\} \in \Re_i \\ (lub_{A_i}, glb_{B_i}) & if \ \{>_{\mathcal{U}}, \geqslant_{\mathcal{U}}\} \in \Re_i \end{array} \right\}$$

Proof. According to the p-box *cdf*-intervals notation, any point p_x that belongs to the interval lies between the *glb* and *lub* with respect to the ordering $\preccurlyeq_{\mathcal{U}}$: $glb \preccurlyeq_{\mathcal{U}} p_x \preccurlyeq_{\mathcal{U}} lub$. Both $\{\succ_{\mathcal{U}}, \succcurlyeq_{\mathcal{U}}\} \in \mathfrak{R}_i$ are the mirror of $\{\prec_{\mathcal{U}}, \preccurlyeq_{\mathcal{U}}\} \in \mathfrak{R}_i$.
The proof demonstrates that all realizations obtained in $S_1 = \Sigma(\mathbf{A}, \mathbf{B})$, which is the complete solution set of the UILS, coincide with $S_2 = \Sigma(\mathbf{A}', \mathbf{B}')$, which is the solution set resulting from the transformation.
For any point $p_x \in S_2$, $\mathbf{A}' \in \mathbf{A}$ and $\mathbf{B}' \in \mathbf{B}$, $\mathbf{B}'$ and $\mathbf{B}'$ are the *glb* and *lub* elements of the original interval solution set. Hence, $p_x \in S_1$.
On the other hand, for any point $p_x \in S_1$, $\exists A \in \mathbf{A}$, $B \in \mathbf{B}$ such that $AX \preccurlyeq_{\mathcal{U}} B$ for $p_x \succcurlyeq_{\mathcal{U}} \perp$, where $\perp$ is the *glb* of all points. For each linear p-box *cdf*-interval inequality though the operator $\preccurlyeq_{\mathcal{U}}$ is monotonic. Thereof, inequality holds with a decrease in the LHS, an increase in the RHS, or both. For any point in $\mathbf{B}$, $B_i \preccurlyeq_{\mathcal{U}} lub_{B_i}$ and $A_i \succcurlyeq_{\mathcal{U}} glb_{A_i}$, any point $p_x \succcurlyeq_{\mathcal{U}} \perp$, the inequality $A_i p_x \succcurlyeq_{\mathcal{U}} glb_{A_i} p_x$ holds. □

This transformation proves the equivalence of the models. Hence, solving the produced list of constraints yields the full closure of the original CSP problem along with its maximum and minimum probabilities of occurrence. For a given optimization problem, we seek to extract the interval bounds enclosing the underlined objective function $\mathbf{Z}$. The maximization of the designated function exploits its upper bound. Similarly, the minimization, introduced by the duality theory Von Neumann (1947), searches for the lower bound. $min(\mathbf{Z}) \preccurlyeq_{\mathcal{U}} (\mathbf{Z}) \preccurlyeq_{\mathcal{U}} max(\mathbf{Z})$. Chinneck and Ramadan (2000) proved that this transformation yields the interval hull of the solution set for the ILS with interval coefficients. This approach is adopted in Bertsimas, Pachamanova, and Sim (2004); Bertsimas and Sim (2004) in order to develop robust optimization problems for ILS under uncertainty. In the first step of the algorithm, we show how to extract the maximization problem and its dual from the system of uncertain linear equations formulated by the p-box *cdf*-intervals.

Extract the LP and its dual.

Consider the following maximization problem over the p-box *cdf*-coefficients and variables:

$$\text{Maximize } Z =_{\mathcal{R}_{\mathcal{U}}} \sum_{j=1}^{n} [p_{e_j}, p_{f_j}] X_j$$

$$\text{subject to } \sum_{j=1}^{n} [p_{a_{ij}}, p_{b_{ij}}] X_j \preccurlyeq_{\mathcal{R}_{\mathcal{U}}} [p_{c_i}, p_{d_i}] \quad \forall i = 1, 2, \ldots, m$$

$$\forall j, \quad X_j \in [p_{x_j}, q_{x_j}], \text{ and } p_{x_j}, q_{x_j} \in \mathbb{R} \times [0..1] \times \mathbb{R}^+$$

The transformation of the above model according to Theorem 8.1

$$\max\ Z = \sum_{j=1}^{n}[p_{f_j}]X_j \qquad \min\ Z = \sum_{i=1}^{m}[p_{c_i}]Y_i$$

$$\text{s.t.}\ \sum_{j=1}^{n}[p_{a_{ij}}]X_j \preccurlyeq_{\mathcal{U}} [p_{d_i}]\ \ \forall i = 1,2,\ldots,m \quad \text{s.t.}\ \sum_{i=1}^{m}[p_{b_{ji}}]Y_i \succcurlyeq_{\mathcal{R}} [p_{e_j}]\ \ \forall j = 1,2,\ldots,n$$

$$\forall j,\quad p_{x_j} \preccurlyeq_{\mathcal{U}} X_j \preccurlyeq_{\mathcal{U}} q_{x_j} \qquad \forall i,\quad p_{y_i} \preccurlyeq_{\mathcal{U}} Y_i \preccurlyeq_{\mathcal{U}} q_{y_i} \tag{8.1}$$

n is the number of variables and m is the number of constraints.

Upper and lower bounds of the p-box *cdf*-intervals are dictated based on Theorem 8.1. The complexity of this splitting step is $O(2m)$. The transformation of the above model yields (2^{n+1}) inequalities over p-box *cdf*-interval bounding points. The produced solution set is $S_i = \{S_i^k | k = 1,2,...,2^{n+1}\}$, where the upper-bound value range is $S_i = lub_{k=1}^{2^{n+1}} S_i^k$, and the lower-bound value range is $S_i = glb_{k=1}^{2^{n+1}} S_i^k$. Despite the fact that this presentation can be solved in the p-box *cdf*-interval constraint solver, it is not as scalable as the Simplex method adopted to solve ILS. An intermediate step needs to be introduced in order to extract linear equations which yield equivalent solution bounds

Example 8.1. *Consider the maximization problem Maximize*$(X_1 + X_2)$, *st*.$(\boldsymbol{A}, \preccurlyeq_{\mathcal{R}_{\mathcal{U}}}, \boldsymbol{B})$ *shown below:*

$$\boldsymbol{A} = \begin{bmatrix} [(-2,0.5,0.2),(2,0.01,0.095)] & [(1,0.3,0.32),(2,0.02,0.083)] \\ [(-3,0.7,0.1),(-1,0.01,0.087)] & [(1,0.2,0.3),(1.5,0.01,0.087)] \\ [(6,0.9,0.98),(6,0.01,0.018)] & [(1.5,0.1,0.6),(3,0.06,0.034)] \end{bmatrix},$$

$$\mathfrak{R} = \begin{bmatrix} \preccurlyeq_{\mathcal{R}_{\mathcal{U}}} \\ \preccurlyeq_{\mathcal{R}_{\mathcal{U}}} \\ \preccurlyeq_{\mathcal{R}_{\mathcal{U}}} \end{bmatrix} \text{ and } \boldsymbol{B} = \begin{bmatrix} [(3,0.88,0.4),(4,0.4,0.088)] \\ [(1,0.85,0.1),(5,0.02,0.013)] \\ [(4,0.9,0.02),(15,0.01,0.001)] \end{bmatrix}$$

The output of the first step in the transformation will be: Maximize(X_1+X_2), *st*.$(\boldsymbol{A}, \preccurlyeq_{\mathcal{U}}, \boldsymbol{B})$

$$\boldsymbol{A} = \begin{bmatrix} (-2,0.5,0.2) & (1,0.3,0.32) \\ (-3,0.7,0.1) & (1,0.2,0.3) \\ (6,0.9,0.98) & (1.5,0.1,0.6) \end{bmatrix}, \mathfrak{R} = \begin{bmatrix} \preccurlyeq_{\mathcal{U}} \\ \preccurlyeq_{\mathcal{U}} \\ \preccurlyeq_{\mathcal{U}} \end{bmatrix} \text{ and } \boldsymbol{B} = \begin{bmatrix} (4,0.4,0.088) \\ (5,0.02,0.013) \\ (15,0.01,0.001) \end{bmatrix}$$

and the dual is formed from the maximization of the lower bounds Minimize$((3,0.88,0.4)*Y_1 + (1,0.85,0.1)*Y_2 + (4,0.9,0.02)*Y_3)$, *st*.$(\boldsymbol{A}, \succcurlyeq_{\mathcal{U}}, \boldsymbol{B})$

$$\boldsymbol{A}_D^T = \begin{bmatrix} (2,0.01,0.095) & (2,0.02,0.083) \\ (-1,0.01,0.087) & (1.5,0.01,0.087) \\ (6,0.01,0.018) & (3,0.06,0.034) \end{bmatrix}, \mathfrak{R}_D = \begin{bmatrix} \succcurlyeq_{\mathcal{U}} \\ \succcurlyeq_{\mathcal{U}} \end{bmatrix} \text{ and } \boldsymbol{b}_D = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Solve 2 **eplex instances.**

The Simplex method is then computed on the maximization and the dual of the minization problem. The computations is exerted over the first component of the coefficient

triplet p-box *cdf*-interval point. Calculations yield the bounding quantiles of the objective function as well as the variables under consideration. Despite the fact that Simplex method worst case complexity is exponential, it is worth noting from the literature that it performs well in the average case. In example 8.1, the calculated quantile bounds of the objective function, x_1 and x_2 are $[1.08, 7]$, $[0.25, 1]$ and $[0.83, 6]$ respectively.

Project the quantiles onto the *cdf* bounding distributions.

We perform $n \times m$ p-box *cdf*-intervals division operations, as demonstrated by Lemma 6.4, provided by the set of m contraints. Since the bounding *cdf*-distributions are uniform, i.e. forming a line equation, the division operation exerted in the probabilistic dimension is of $O(1)$ complexity. The obtained quantile bounds extracted from the previous step are then projected onto the resulting domain of the divisions. This projection has, in turn, a linear computation. It is exerted by calculating the *cdf*-value of a given quantile, when it is located on a *cdf*-distribution.

Extract the final solution bounds.

Extracted solution bound triplets are selected from the projected list of the *cdf*-distributions. This list is ordered in the probability domain by means of the second order stochastic dominance (Definition 2.8). The lower and the upper *cdf*-bounds per variable are the dominated and dominant distributions respectively. Possibly, further domain pruning is exerted to preserve the *cdf*-stochastic dominance properties in order to ensure that the maximum *cdf*-value of a variable quantile should be greater than, or equal to, its minimum probability of occurrence.

Proposition 8.1. *The p-box cdf-intervals bounding the solution domain of the variable* X_i.

$$X_i \in [glb^m_{j=1}(p^j_{x_i}), lub^m_{j=1}(q^j_{x_i})] \forall i \in 1 \dots n$$

For each variable X_i in the system of linear constraints, the result of the division operation is given by the intersection of all p-box *cdf*-intervals $[p^j_{x_i}, q^j_{x_i}]$, i is the index of the variable in the variable list and j is the index of its involved constraint. $p^j_{x_i}$ and $q^j_{x_i}$ are the lists of lower and upper bounding points respectively. In order to extract p_{x_i} and q_{x_i} for each variable such that $p_{x_i} \preccurlyeq_{\mathcal{U}} X_i \preccurlyeq_{\mathcal{U}} q_{x_i}$, we compute the *glb* and the *lub* on $p^j_{x_i}$ and $q^j_{x_i}$ respectively resulting from all given constraints.

In Example 8.1 the resulting p-box *cdf*-intervals for X_1 and X_2 are $[(0.25, 0.85, 0.1), (1, 0.75, 4.44e-17)]$ and $[(0.83, 0.86, 0.14), (6, 0.081, 0.0055)]$ respectively. We can also compute the objective function Z by applying the p-box *cdf*-intervals addition operation $X_1 + X_2$. It is given in this example as $[(1.08, 0.85, 0.06), (7, 0.75, 4.44e - 17)]$.

8.1.2 Algorithm Complexity

We compare our new proposed algorithm for global constraint relaxation with the one introduced in Yorke-Smith and Gervet (2009). Note that this algorithm is constructed

over the objective function (minimization/maximization) existing as part of the problem characteristics. This initial step bypasses the computation of the $2n$ eplex instances solved in the algorithm proposed by Yorke-Smith and Gervet (2009) to compute the interval hull of the UCSP algebraic structure. This set of operations was necessary in order to extract the bounds of the solution sought for each variable. Our algorithm further adds the bounds on the likelihood of each variable. This additional information comes from the p-box *cdf*-intervals property which guides the knowledge about the search space not only from the real domain perspective but also from a probabilistic viewpoint. Table 8.1 shows the computation complexity of each step in our transformation algorithm. Clearly, we have gained an improved complexity when compared to the algorithm introduced in Yorke-Smith and Gervet (2009). Moreover, the additional probabilistic information is exerted linearly.

p-box *cdf*-intervals		UCSP	
Extract LP and its dual	$O(m)$	Generate linear inequalities	$O(2m)$
Solve 2 eplex	$O(2^{n+1})$	Solve 2n eplex	$O((2n)(2^{n+1}))$
Projection	$O(nm))$		
Extract solution bounds	$O(2n)$	Extract solution bounds	$O(2n)$

Table 8.1: Computation complexity of the transformation algorithm

8.1.3 Performance Comparison

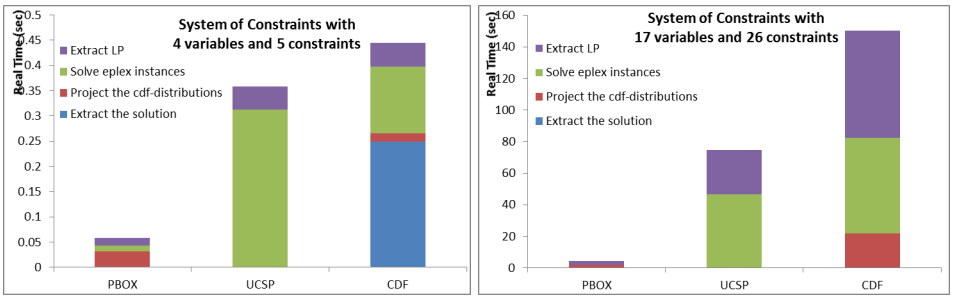


Fig. 8.2: Real-time performance

The framework was tested on two systems of constraints with densities: 12 (4 variables and 5 constraints) and 127 (17 variables and 26 constraints). Performance is given in terms of seconds: the real-time taken by the algorithm to compute the bounds of the final solution set. We used the ECL^{i}PS^e profiling in order to calculate the percentage of the total time, each predicate takes to carry-out its designated task ECRC (1994).

		p-box *cdf*-intervals	UCSP	*cdf*-intervals
12 variables	Extract LP	26.4%	13.04%	8.1%
	Solve eplex instances	20.68%	86.95%	45.9%
	Project the *cdf*-distributions	52.87%	N/A	2.7%
	Extract the solution	0%	N/A	43.24%
	Total real-time (sec.)	**0.059**	**0.358**	**0.44**
127 variables	Extract LP	52.92%	37.65%	37.91%
	Solve eplex instances	0.19%	62.34%	21.77%
	Project the *cdf*-distributions	46.34%	N/A	3.87%
	Extract the solution	0.535%	N/A	36.44%
	Total real-time (sec.)	**20.39**	**86.45**	**222.24**

Table 8.2: Real-time execution

Real-time performance of the p-box *cdf*-intervals framework is compared to the UCSP and the *cdf*-intervals with one approximated *cdf*-uniform distribution. Coefficient bounds of the ILS are choosen to be the same, in order to check for the ability of the algebraic structure to shrink the bounds of the solution set. In this experiment we employ the UCSP transformation algorithm adopted for the UCSP in Yorke-Smith and Gervet (2009) and for the *cdf*-intervals with one approximated *cdf*-uniform distribution Saad et al. (2010). The UCSP transformation algorithm consists of four main steps. It starts with extracting the set of LP inequality constraints on the real bounds of the UCSP and on the first component of the *cdf*-interval algebraic structure. The second step of the transformation algorithm, as in Yorke-Smith and Gervet (2009), computes 2 eplex instances per variable, and it creates one maximization and one minimization objective function over each variable. Note that in this step the optimization of the genuine objective function defined in the problem is not included in the computation of the interval hull of the solution set. The third step in the algorithm computes the minimum and the maximum values, in the domain of reals, resulting from the second step instances. The computation over the UCSP stops at this steps and the real interval hull bounding the solution set can be derived in the real domain. However, and since the *cdf*-intervals algebraic structure works on a 2D space, 2 extra steps are added. The first additional step computes $n \times m$ *cdf* division operations. Then the *cdf*-intervals *glb* and *lub* computations are exerted to deduce the *cdf* bounding points.

Table 8.2 shows a comparison between the proposed EUCSPT algorithm for the p-box *cdf*-intervals and the UCSP transformation algorithm adopted for the UCSP in Yorke-Smith and Gervet (2009) and for the *cdf*-intervals with one approximated *cdf*-uniform distribution Saad et al. (2010). Obviously, the EUCSPT algorithm is less expensive in terms of complexity and real runtime taken in seconds. This is due to the

fact that, in our proposed transformation algorithm, we run 2 simplex instances on the main problem and its dual. On the other hand, the UCSP transformation algorithm runs 2 simplex method per variable in order to extract their bounding real points. Fig. 8.2 and Table 8.2 illustrate that the projection onto the *cdf*-domain takes less real time execution when compared to solving the eplex instances per variable in the UCSP and the *cdf* intervals algebraic structure.

Table 8.2 shows the percentage of the real-time taken by each predicate to execute. The third column in the table represents the execution of the *cdf*-intervals algebraic structure with one approximated *cdf*-uniform distribution. Clearly, both solving the $2n$ eplex instances and computing the solution set *glb* and the *lub* of the *cdf*-interval bounding points take almost the same time range. Noticeably, the $n \times m$ *cdf* division operations take 2.7% and 3.87% of the execution time for both the 12 and the 127 problem densities respectively. In the case of the p-box *cdf*-interval algebraic structure, shown in the first column, they take 0.03 seconds and 9.44 seconds when 12 and 127 problem densities are involved. We can conclude from these observations that the division operations are not expensive in terms of execution time.

Another advantage of the proposed algorithm is that it considers the optimization of the objective function defined in the genuine problem. Moreover, p-box *cdf*-interval bounding points of the objective function are computed and extracted by this proposed algorithm.

8.2 Summary

This chapter introduces a relaxation technique for the global chance constraint. It defines the LP relaxation approach utilized for global constraints over the p-box *cdf*-intervals. The p-box *cdf*-intervals were employed because they guarantee a full encapsulation of the observed information along with its whereabouts. This method is a preprocessing propagation technique that analyzes the problem from a global perspective and in a tractable manner. Compared to existing reliable techniques, it suggests tighter as well as more accurate bounds on the search space in a two dimensional manner, enclosing the data along with its probability of occurrence. This new domain propagation proved to be efficient and effective.

PART III

APPLICATIONS

CHAPTER 9

NETWORK TRAFFIC ANALYSIS PROBLEM

In the field of telecommunication the Network Traffic Matrix (TM) is an essential tool that identifies traffic flows between all possible Origin and Destination (OD)-pairs in a given network. It is widely used to help network administrators monitor the actual network traffic and issue cost of service reports. TM is typically used in network design problems to seek the realization of all possible flows by installing links at minimum cost; while satisfying the link capacity constraints (i.e. the flow cannot exceed the installed link capacity). Accordingly, TM is employed in capacity planning, traffic engineering, reliability analysis, network management and other network adminitrative tasks.

Exploiting TM with complete OD-pair combinations, for a real-world large network, is an expensive computation. This matrix computation is directly proportional to the square of the matrix dimension; in this case, the number of OD-pair entries. To obtain TM data two measurement mechanisms can be found: direct and indirect. The direct approach is based on NetFlow (Claise, Sadasivan, Valluri, and Djernaes (2004)) and sFlow (Pheal (1992)) mechanisms; it is more accurate; but more expensive. The direct mechanism mainly exploits the packet content at several network protocol layers. On the other hand, the indirect approach is widely used and supported in actual networks. It is based on measuring the network link-counts; and it is exerted by the Simple Network Management Protocol (SNMP) (Case, Fedor, Schoffstall, and Davin (1990)). The task of measuring link-counts of a network is done at ease and its dedicated instruments are widespread. Thereof, for real-world large networks, research has focused on algorithms that calculate the TM based on the link-counts (Jin et al. (2008)).

Eventually, traffic volumes in the network fluctuate. Observed fluctuations urge network planners to overestimate the flow in order to ensure adequate network sustainability. Statistical methods seek at estimating demands based on prior measurements and

building assumptions on the population of traffic distribution.

Convex modeling, such as in Koster, Kutschka, and Raack (2010), typically formulate the network problem in MIP. They seek at exploiting minimum and maximum bounds which encapsulate all feasible realizations of the end-to-end network traffic flow. Convex frameworks seek at exploiting extreme bounds that encapsulate all feasible realizations of the end-to-end network traffic flow. Estimated resulting flow has an equal degree/weight of uncertainty and lacks the portrayal of the data whereabouts.

To the best of our knowledge, available techniques lack the ability to provide an accurate model which describes natural fluctuations of the traffic flow. Hence, a representation that account for the encapsulation of the actual distribution of data, provided in the problem definition, is needed.

9.1 Modeling the NTAP problem

Typically, data available for the NTAP is:

1. Measurements of the traffic flow in each link.

2. Routing matrix that specifies the OD-pair network usage.

The measurement process of the link-counts is exerted by the SNMP (Case et al. (1990)). Consider Y a vector of size r which identifies the measured traffic volumes (link-counts) for each link at a given time t. r is the number of links in the network. Successive measurements can be repeatedly taken over time. The vector Y_t denotes measured link-counts at time t. The frequency of t is generally determined by the network operator and it can vary from less than 5 mins to a month time interval. The number of origin-destination pairs in the network is: $c = n(n-1)$; where n is the number of Network Point of Presence (POP); POP can be either a routing or an end node. The routing matrix A is $r \times c$; it is normally obtained from the Border Gateway Protocol (BGP) configurations, through the Open Shortest Path First (OSPF), or Intermediate System - Intermediate System (IS-IS) link weights, at the router interface. Each column in A represents a traffic flow and each row corresponds to a link. In the simple case, A_{ij} is set to 1 if the OD-pair i uses the link j and 0 otherwise. Accordingly, the problem $Y = AX$ searches for values of OD-pairs in X. Computed values should be able to reproduce link-counts which are close in value to the measured ones. Generally, the NTAP is under-constrained because the number of links is significantly small, compared to the number of traffic demands, ($r \ll c$). This fact yields a feasible solution set that can be infinite.

We study a fragment of a network with three nodes (A, B, C) and two bidirectional links, as illustrated in Fig. 9.1. In this network $c = 6$ traffic flow variables and $r = 4$ directed link loads.

Fig. 9.2 is the mathematical representation of the 3 nodes problem in the deterministic case as explained in Airoldi and Faloutsos (2003). Nodes are connected by

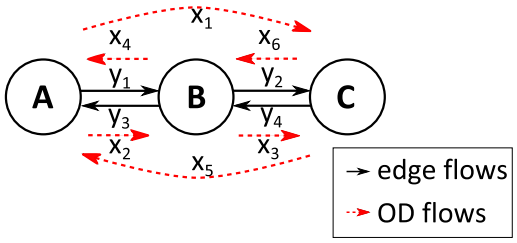


Fig. 9.1: Network of 3-nodes and 2-bidirectional-links

bidirectional links. Non-observable OD flows are represented by dash circles. Measurable traffic flows are represented by solid circles and each is pointing to the OD flow that is possibly utilizing its corresponding link.

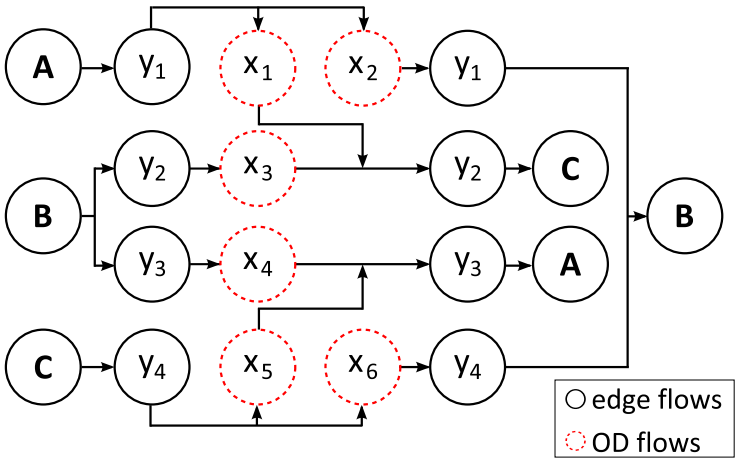


Fig. 9.2: Mathematical model of 3-nodes and 2-bidirectional-links

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

Generally, the network is subject to three types of constraints:

1. Link traffic constraints: on each link the sum of flow is equal to the measured traffic volume

2. Traffic conservation constraints: at each router, the flow leaving the network is destined to the router and the flow coming in the network is issued by the router.

3. Flow conservation constraints: at each node, total incoming flow is equivalent to the total outgoing flows.

We basically use this model and extend it to describe data anticipated by *cdf*-intervals and p-box *cdf*-intervals. In our model, the matrix A will contain an identity p-box *cdf*-element when the flow is assigned to the link and a zero element when it is not.

9.2 Input Dataset Representation

To test the model we used the Matlab toolbox Matlab (2010) to generate and convert measurable data into *cdf*-intervals and p-box *cdf*-intervals. Under uncertainty, data is provided by either a single demand matrix or a set of n distinct measured-demand matrixes. In the single matrix case provided demands are often based on approximating the original measured values, forecasting traffic volumes or computing statistical assumptions on the data population. Whereas in the original problem a set of multiple demand matrixes are provided each representing the demand measurement at a given time point. The data corpus available in Orlowski, Pióro, Tomaszewski, and Wessäly (2007) and Orlowski, Pióro, Tomaszewski, and Wessäly (2010) provide two types of datasets: single and dynamic set of traffic matrixes. As illustrated in Fig. 9.3, we formed the p-box *cdf*-intervals from the original dynamic data and we generated both Normal and Poisson distributions for each element in the single demand matrix to build the p-box *cdf*-interval coefficients. Output coefficients from this operation are then utilized to model the NTAP problem and form the Y vector. Recall that the A matrix has an identity p-box *cdf*-interval element if the corresponding traffic flow uses the link and a zero p-box *cdf*-interval element otherwise. The formed matrix coefficients is input to the solver which in turn seeks a p-box *cdf*-interval output representation for each network flow (netflow) variable. Resulting solution sets are then compared with the original demand matrix of the problem in order to validate our methodology.

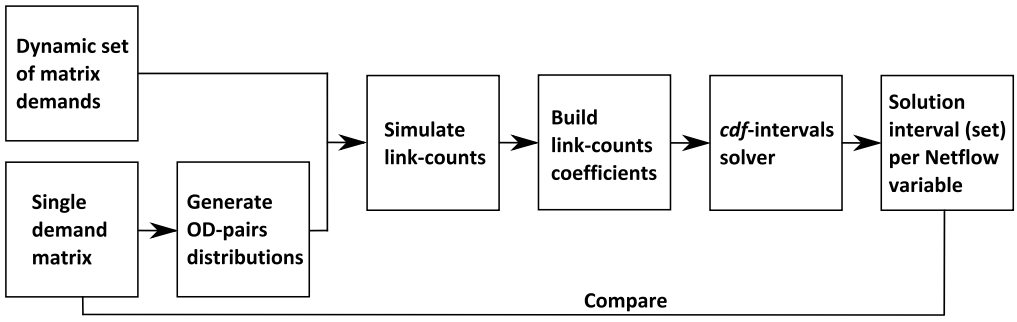


Fig. 9.3: Simulating the traffic loads

9.2.1 Simulated TM

To compute the distributions, we assume that the measurement provided in the single demand matrix is the mean value. We then set n as the number of readings/measurements; (i takes values from 1 to n); for each OD-pair we generate n values based on either Poisson or Normal distributions. As proposed by Medina, Taft, Salamatian, Bhattacharyya, and Diot (2002), the Poisson distribution computation per element is as follows:

1. Generate a λ_i value from the uniformly distributed interval $[100, 500]$
2. Apply the Poisson distribution function over the generated λ_i to get the value of a network demand $X_i = Poisson(\lambda_i)$

And the Normal distribution generation per value is as follows:

1. Generate μ_i (mean value) for each OD-pair from the uniformly distributed interval $[100, 500]$
2. Set the variance $\sigma_i^2 = 40$ for all
3. Apply the Normal distribution function over the generated μ_i to get the value of a network demand $X_i = Normal(\mu_i, \sigma_i)$

9.2.2 Interval coefficients formulation

In the two cases (single and dynamic demand matrixes), the measurement operation yields a set of n evaluations per flow variable; we simulate this scenario for each element in the TM; we aggregate the probability distribution generated in Section 9.2.1 then we construct the flow set *cdf* distribution. This operation yields, for each flow variable, an array of n distinct quantiles along with their corresponding n *cdf* values; such array is input to the preprocessing steps of each framework.
The interval representation of each flow variable is established as follows:

1. Minimum and maximum values are recorded to represent the UCSP interval bounds.
2. *cdf*-distribution of n-steps is computed based on generated Poisson or Normal distributions provided in Section 9.2.1.
3. Algorithm 1 and Algorithm 2 are applied to contruct the *cdf*-interval and p-box *cdf*-interval bounds. In the first case, one *cdf*-uniform distribution represents the interval and in the second case the p-box *cdf*-interval structure encapsulates all possible occurrences of the data measured.

9.3 An instance: a network with 4 nodes

For a detailed demonstration, we utilize a small hypothetical network with 4 nodes. This network is depicted in Fig. 9.4 and it is used in Medina et al. (2002) and Yorke-Smith and Gervet (2009). We employ this network to visualize how data is manipulated on the set of netflow variables. As shown in Fig. 9.4 all links are bidirectional except for the link between nodes A and C. Fig. 9.4 is a snapshot at a point in time where the true values of traffic volumes are displayed on the links.

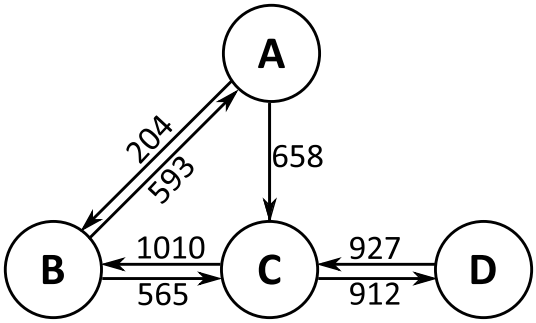


Fig. 9.4: 4-nodes instance

Link-counts are then simulated to form the interval bounds in the 3 algebraic structures:

1. Error correction model $(\mu, 40)$ represented by UCSP
2. *cdf*-intervals with 1 approximated *cdf*-distribution
3. p-box *cdf*-intervals encapsulated by two *cdf*-distributions

Table 9.1 displays the calculated interval bounds when the data is following a Normal distribution. Clearly, computed p-box *cdf*-intervals encapsulate the data whereabouts; the framework also adds more information about the maximum and minimum probabilities a quantile value can obtain.

Clearly intervals derived in the three models under consideration encapsulate the actual measured value. The second column in Table 9.1 lists the hypothetical mean values measured on the links specified in the first column. And since this measurement is based on a Normal probability function we constructed their corresponding distributions $(\mu, 40)$ as in Section 9.2.1; we accordingly establish data within interval-bounds in an exhaustive manner. Recognized data from this operation simulate an adequate hypothetical representation of a standard NTAP problem. The rest of Table 9.1 are derived bounds for the three models; clearly, UCSP which employs an error correction model over-estimates the extreme points of the interval as shown in the last two columns; the UCSP model seeks to encapsulate all possible data; As shown in Yorke-Smith and Gervet (2009) the model assumes data is following a Normal distribution which is not always the case

in a typical NTAP problem. Moreover, intervals provided in UCSP have an equally weighted knowledge of data whereabouts. In the *cdf*-intervals with one approximated uniform distribution, deduced input intervals are the most pruned; they also contain the true measurement within their extreme bounds; unlike the UCSP the steepness of the *cdf* distribution is an acceptable indication of the data whereabouts specially when we need to express data in an uncertain environment. However, this model considers the approximation of the probability to a uniform distribution and lacks the representation of the possible maximum and minimum probability data can happen. The p-box *cdf*-intervals framework encapsulates all possible data along with their possible probabilities using two uniformly distributed probabilities that are issued from the extreme points.

Example 9.1. *Consider in Table 9.1 the traffic flow from node A to node B:* V_{AB}*; its generated Normal distribution mean value is* 204*. This value is bounded by the interval of quantiles* $[171.02, 226.08]$ *in the UCSP model. In the* cdf*-intervals model* V_{AB} *ranges between quantiles* 174.49 *and* 206.75 *with an average step value of* 2.9%*; i.e. derived* cdf*-value of quantile* 175.49 *is* 0.0664*. p-box* cdf*-intervals representation is a full encapsulation of the actual data; i.e. data cannot exist outside the interval quantile and probabilistic bounds. p-box* cdf*-interval of the netflow variable* V_{AB} *shows that quantiles cannot lie outside the interval* $[174.49, 208.12]$*. It is worth noting that this interval of quantiles has tighter bounds, when it is compared to the UCSP interval representation. The p-box* cdf*-intervals structure indicates that probabilistic average step values lie between* $[2.5\%, 5.3\%]$*. Given this information, we can deduce that the* cdf*-value of* 176.49 *is between a minimum value of* 0.09 *and a maximum value of* 0.94*.*

Table 9.2 demonstrates solution sets resulting from Table 9.1 input interval coefficients. Each solver reasons about its corresponding input intervals and deduce the values of the unknown Netflow variables; solutions in turn are compared with the original Netflow true values that are listed in column 2. Positively provided solution sets in all the models under consideration contain the designated Netflow true values.

Similarly, the model was generated based on a Poisson distribution in order to monitor the problem behavior under different hypothetical distributions. Generated input interval coefficients are listed in Table 9.3, and the output generated from the 3 models is listed in Table 9.4. Output solution sets in all cases adequately encapsulate the original Netflow data.

For the same coefficient input bounds, Fig. 9.5 illustrates the result of pruning the output netflow variable F_{AC} in the 2D space. Noticeably quantile bounds are typical in the three models under consideration. The UCSP shows an equal degree of occurrence for all quantiles that lie within the interval bounds. The approximated *cdf* distribution of the *cdf*-intervals algebraic model lies between the two maximum and minimum uniform probability bounds of the p-box *cdf*-interval representation.

Similarly, Fig. 9.6 demonstrates how the netflow variable F_{AD} is pruned. UCSP and *cdf*-intervals with one approximated distribution generate the same extreme quantiles.

		Derived measured bounds											
	True Values	p-box *cdf*-intervals						*cdf*-intervals				UCSP err correction model $(\mu, 40)$	
		lb	cdf_{lb}	slope_{lb}	*ub*	cdf_{ub}	slope_{ub}	*lb*	cdf_{lb}	*ub*	cdf_{ub}	*lb*	*ub*
V_{AB}	204	174.49	0.037	0.053	208.12	0.19	0.025	174.49	0.037	206.75	0.98	171.02	226.08
V_{BA}	593	503.23	0.01	0.012	604.12	0.006	6.40E-05	525.88	6E-04	600	1	492.82	658.0
V_{AC}	658	598.43	0.007	0.014	665.69	0.01	0.001	603.92	8E-04	665.69	1	591.49	701.61
V_{BC}	565	505.32	0.01	0.014	572.58	0.09	0.004	505.32	2E-04	572.58	1	498.39	608.50
V_{CB}	1011	890.79	0.01	0.009	1025.31	0.009	6.40E-05	913.44	7E-04	1021.19	0.99	876.92	1097.15
V_{CD}	913	822.51	0.009	0.009	923.4	0.001	0.001	825.26	1E-04	923.40	1	812.11	977.28
V_{DC}	927	837.42	0.02	0.016	938.30	0.006	6.50E-05	858.69	1.4E-03	932.81	0.99	827.01	992.18
TA_{in}	863	772.92	0.01	0.011	873.81	0.16	1.53E-03	778.41	1E-04	872.43	1	762.51	927.69
TA_{out}	593	503.23	0.01	0.012	604.12	0.006	6.40E-05	525.88	6E-04	600	1	492.82	658.0
TB_{in}	977	887.03	0.02	0.012	987.91	0.08	3.57E-03	887.03	1E-04	983.79	1.0	876.62	1041.79
TB_{out}	1034	943.91	0.04	0.017	1044.64	0.05	7.99E-03	943.75	9E-04	1039.15	0.99	933.35	1098.52
TC_{in}	750	660.24	0.01	0.01	761.13	0.17	1.59E-03	664.36	1E-04	760.44	1	649.83	815.01
TC_{out}	978	888.10	0.01	0.012	988.99	0.16	1.50E-03	893.59	7E-04	986.93	0.99	877.70	1042.87
TD_{in}	927	837.42	0.024	0.016	938.3	0.006	6.50E-05	858.69	1.4E-03	932.81	0.99	827.01	992.18
TD_{out}	913	822.51	0.01	0.009	923.4	0.019	1.77E-03	825.26	1E-04	923.40	1.0	812.11	977.28

Table 9.1: Link-counts when OD-pairs are based on a Normal distribution

		Solution set bounds											
	True Values	p-box *cdf*-intervals						*cdf*-intervals				UCSP err correction model $(\mu, 40)$	
		lb	cdf_{lb}	slope_{lb}	*ub*	cdf_{ub}	slope_{ub}	*lb*	cdf_{lb}	*ub*	cdf_{ub}	*lb*	*ub*
F_{AB}	204	175.87	0.11	5.32E-02	208.12	0.06	1.53E-03	174.49	0	206.75	0.98	171.02	226.08
F_{AC}	338	274.48	0.01	1.48E-02	375.37	0.16	1.50E-03	279.97	0	375.37	0.99	249.58	414.76
F_{AD}	320	293.79	0.11	2.94E-02	323.95	0.06	1.53E-03	290.32	0	323.95	1	286.85	341.91
F_{BA}	412	314.44	0.01	1.26E-02	482.59	0.01	6.40E-05	337.09	0	455.82	1	268.12	543.41
F_{BC}	194	97.22	0.01	1.47E-02	265.37	0.16	1.50E-03	97.22	0	265.37	0.99	50.89	326.18
F_{BD}	371	307.22	0.01	1.47E-02	408.10	0.09	1.77E-03	307.22	0	408.1	1	282.32	447.49
F_{CA}	88	0.0	0.0	1.48E-02	188.78	0.01	6.40E-05	0	0	188.78	0.99	0	224.71
F_{CB}	441	343.97	0.04	1.79E-02	511.95	0.01	6.40E-05	348.61	0	506.46	0.99	297.48	572.77
F_{CD}	221	124.09	0.01	1.09E-02	292.24	0.17	1.59E-03	126.84	0	292.24	1	77.77	353.06
F_{DA}	94	0	0.02	1.63E-02	188.78	0.01	6.40E-05	0	0	188.78	0.99	0	224.71
F_{DB}	388	358.2	0.06	7.67E-02	391.83	0.01	6.40E-05	358.2	0	388.39	0.98	354.73	409.79
F_{DC}	446	280.99	0.02	1.63E-02	580.1	0.01	6.50E-05	286.49	0	574.61	0.99	192.52	637.45

Table 9.2: Netflow variables: comparison between output solution sets and true values when OD-pairs input coefficients are based on a Normal distribution

	Derived measured bounds											
	p-box *cdf*-intervals						UCSP		*cdf*-intervals			
	lb	cdf_{lb}	slope_{lb}	*ub*	cdf_{ub}	slope_{ub}	*lb*	*ub*	*lb*	cdf_{lb}	*ub*	cdf_{ub}
V_{AB}	174.49	0.0374	0.0532	208.12	0.095	2.59E-02	174.49	208.12	174.49	0.03744	208.12	1
V_{BA}	503.23	0.0114	0.0126	604.12	6.8E-03	6.40E-05	503.23	604.12	503.23	0	604.12	1
V_{AC}	598.43	7.2E-03	0.0148	665.69	0.1130	1.62E-03	598.43	665.69	598.43	0	665.69	1
V_{BC}	505.32	0.0107	0.0147	572.58	0.0886	4.13E-03	505.32	572.58	505.32	2.3E-04	572.58	1
V_{CB}	890.79	0.0152	9.6E-03	1025.31	9E-03	6.40E-05	890.79	1025.31	890.79	0	1025.31	1
V_{CD}	822.51	8.8E-03	9.8E-03	923.40	0.1855	1.77E-03	822.51	923.40	822.51	0	923.40	1
V_{DC}	837.42	0.0235	0.0163	938.30	6.9E-03	6.5E-05	837.42	938.30	837.42	0	938.30	1
TA_{in}	772.92	0.0137	0.0116	873.81	0.1599	1.53E-03	772.92	873.81	772.92	0	873.81	1
TA_{out}	503.23	0.0114	0.0126	604.12	6.8E-03	6.4E-05	503.23	604.12	503.23	0	604.12	1
TB_{in}	887.03	0.0183	0.0126	987.91	0.3777	3.57E-03	887.03	987.91	887.03	6.6E-05	987.91	1
TB_{out}	943.91	0.0407	0.0179	1044.64	0.0456	7.99E-03	943.91	1044.64	943.91	2.56E-03	1044.64	1
TC_{in}	660.24	0.0108	0.0109	761.13	0.1661	1.59E-03	660.24	761.13	660.24	0	761.13	1
TC_{out}	888.10	0.0140	0.0122	988.99	0.1574	1.50E-03	888.10	988.99	888.10	0	988.99	1
TD_{in}	837.42	0.0235	0.0163	938.30	6.9E-03	6.50E-05	837.42	938.30	837.42	0	938.30	1
TD_{out}	822.51	8.8E-03	9.8E-03	923.40	0.1855	1.77E-03	822.51	923.40	822.51	0	923.40	1

Table 9.3: Link-counts: bounds on input coefficient values that are based on a Poisson distribution

	Solution set bounds											
	p-box *cdf*-intervals						UCSP		*cdf*-intervals			
	lb	cdf_{lb}	slope_{lb}	*ub*	cdf_{ub}	slope_{ub}	*lb*	*ub*	*lb*	cdf_{lb}	*ub*	cdf_{ub}
F_{AB}	175.87	0.1107	0.0532	208.12	0.1599	1.53E-03	174.49	208.12	174.49	0	208.12	1
F_{AC}	274.48	7.2E-03	0.0148	375.37	0.1574	1.50E-03	274.48	375.37	274.48	0	375.37	1
F_{AD}	293.79	0.1139	0.0294	323.95	0.1599	1.53E-03	290.32	323.95	290.32	0	323.95	1
F_{BA}	314.44	0.0114	0.0126	482.59	6.8E-03	6.40E-05	314.44	482.59	314.45	0	482.59	1
F_{BC}	97.22	0.0107	0.0147	265.37	0.1574	1.50E-03	97.22	265.37	97.22	0	265.37	1
F_{BD}	307.22	0.0107	0.0147	408.10	0.1855	1.77E-03	307.22	408.10	307.21	0	408.10	1
F_{CA}	0	2.6E-03	0.0148	188.78	6.8E-03	6.4E-05	0	188.78	0	0	188.78	1
F_{CB}	343.97	0.0407	0.0179	511.95	9E-03	6.4E-05	343.97	511.95	343.96	0	511.95	1
F_{CD}	124.09	0.0108	0.0109	292.24	0.1661	1.59E-03	124.09	292.24	124.09	0	292.24	1
F_{DA}	0	0.0235	0.0163	188.78	6.8E-03	6.4E-05	0	188.78	0	0	188.78	1
F_{DB}	358.20	0.0553	0.0767	391.83	9E-03	6.4E-05	358.20	391.83	358.20	0	391.83	1
F_{DC}	280.99	0.0235	0.0163	580.10	6.9E-03	6.50E-05	280.99	580.10	280.99	0	580.10	1

Table 9.4: Netflow variables: output solution sets when input coefficients are based on a Poisson distribution

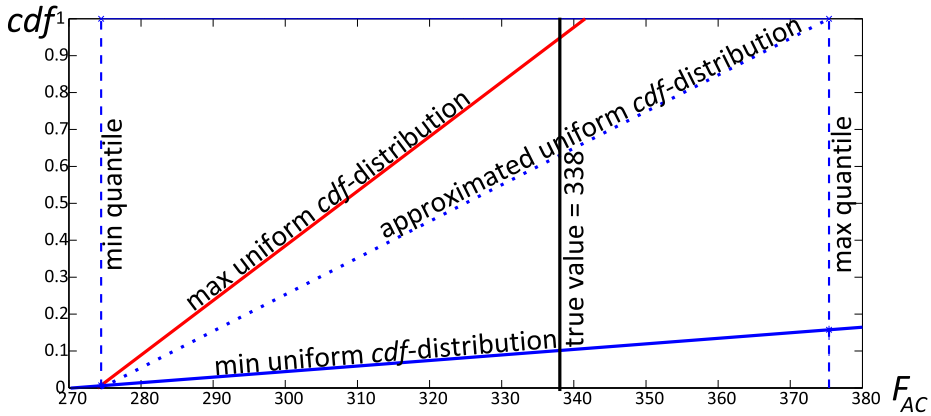


Fig. 9.5: Pruning F_{AC} in the 2D space

However, the two bounding distributions of the p-box *cdf*-intervals representation intersect; this intersection yields a conflict in the *cdf* property: the maximum *cdf* distribution at small quantiles is less in value than their minimum distribution; in other words, any point that lies in the 2D space before 293.79 will have a minimum *cdf* that is greater than its maximum distribution; hence, it is impossible to find a solution point in this inconsistent area; this conflict allows the solver to further prune the domain. Accordingly, quantiles of F_{AD} can be 293.79 at 11% but they cannot be less than the value 293.79.

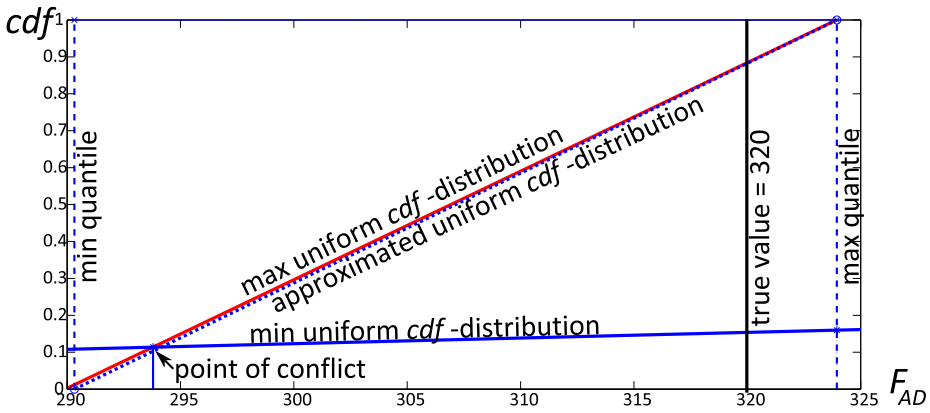


Fig. 9.6: Pruning F_{AD} in the 2D space

We have generated 80 instances of the above 4-nodes network problem: 40 instances based on Poisson distributions and the other 40 are based on Normal distributions of OD-pairs. Link-counts were simulated. Output solution sets of all maneuvers witnessed a total encapsulation of the generated Netflow data. The p-box *cdf*-intervals algebraic structure can further prune the output solution domains to prohibit the upper bound probability from being dominated by lower bound distribution. The solver detected an area of conflict in 33% -42% of the cases when Poisson distribution was employed; and 16% - 25% of the cases were observed when Normal distribution was taken under consider-

ation; in these cases output solution domains from the p-box *cdf*-intervals were further pruned when compared to bounded realizations resulting from UCSP and *cdf*-intervals; this is because p-box *cdf*-intervals disregard unrealistic quantiles from the search space.

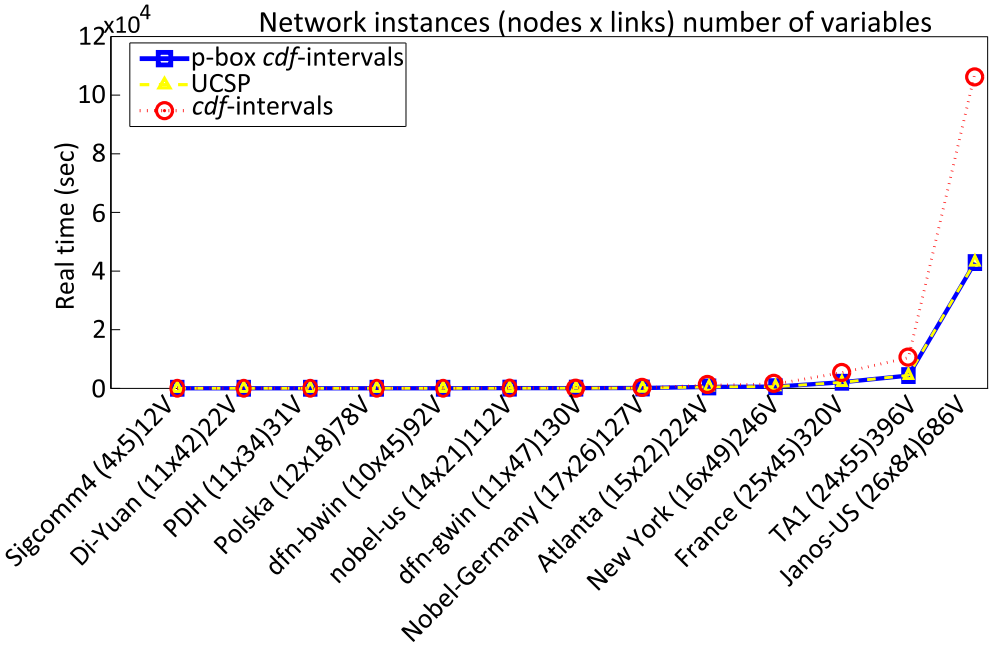


Fig. 9.7: Real-time comparison

9.4 Scalability Test

In this experiment we employed the corpus available in Orlowski et al. (2007, 2010). As shown in Fig. 9.7 and Fig. 9.8 twelve networks with varying densities were employed:

1. Number of variables: 12 to 688
2. Number of nodes: 4 to 26
3. Number of links: 5 to 84

Clearly the real-time taken to find the problem solution domains is directly proportional to the network density. The problem density is specified by the number of unknown/ variables the solver needs to exploit based on the given link-counts.

Fig. 9.7 and Fig. 9.8 illustrate the real-time spent by the three solvers in seconds and in log scale. Fig. 9.7 show that p-box *cdf*-intervals exploit variable solution domains with almost the same real-time as UCSP; and this is further observed even in the log scale Fig. 9.8. Hence, the p-box *cdf*-intervals framework is cost effective and does not add up computational cost for large problems. Moreover, it adds up information about the minimum and maximum distributions of data; accordingly, user (domain expert) can

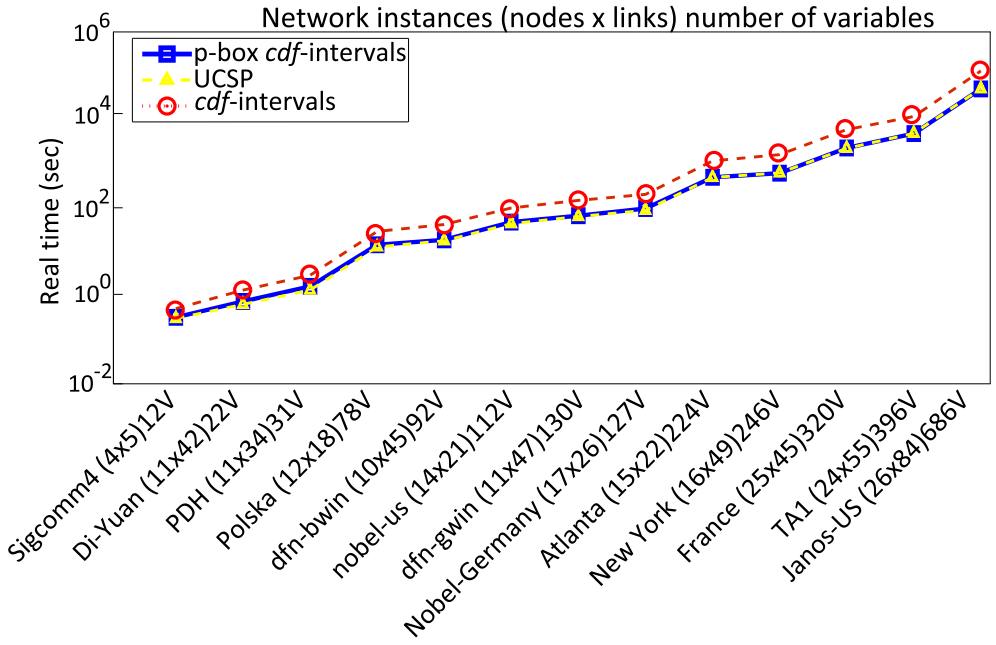


Fig. 9.8: Real-time comparison (log scale)

choose to further constraint the output domain of the variable by specifying a probability limit to restrict the search space.

9.5 Summary

In this chapter we compare the p-box *cdf*-intervals with convex models. We proved that the new algebraic structure can easily formulate problems that are modeled by means of convex intervals such as the UCSP. We support our argument by a case study using the consortium of NTAP problems. The new structure incorporates knowledge on data whereabouts along with the interval of quantiles. This enables decision makers to include additional information that is already given in the problem then they can use this knowledge to reason about the data. Experimental results show that p-box *cdf*-intervals can further prune the domain of intervals; this occurs when stochastic dominance probabilistic property is violated; in this case the algebraic structure of the p-box *cdf*-intervals seeks to find the nearest possible point that satisfies the probabilistic dominance. We proved in practice that p-box *cdf*-intervals employ this behavior in a cost effective manner without adding significant computation cost for large problems.

CHAPTER 10

MANAGEMENT OF INVENTORY

Inventory management involves a large class of real life problems; the daily ordering of newspapers; the booking of airline flights; the ordering of items in a supply chain of a manufacturing process. In this consortium of real life applications, determining an adequate quantity to be kept in-stock, with minimal cost, given the stochasticity of the demand/order environment, is an open-ended area of research (Jacob, Chase, and Aquilano (2009)).

Companies seek to design a cost effective model that synchronizes production plans with supplier's orders. In a manufacturing environment, producers schedule ahead orders of raw materials to meet their promised delivery timings with minimum possible cost. However, the schedule of orders is based on two main factors which are fluctuating: customer demands and market prices. Those two factors in turn change the size, cost and time of orders unpredictably.

10.1 Problem definition

The inventory problem is simplified to a one stage replenishment cycle of one item and it resembles the case of a newsvendor. Typically and on a daily basis, a newsvendor needs to determine the number of newspapers to buy before observing demands. Excess ordering yields an overhead cost since overstock newspapers on the second day become, by nature, obsolete. On the other hand, unmet demand leads to profit loss and possibly additional unforeseen 'customer switching cost'. This model has been thoroughly researched due to its importance and applicability in real life.

More extensions to the model incorporate variations on the following:

- setup time
- production (batches, single machines, parallel machines, open shops, flow shops, job shops)

- inventory (minimum service level, backlogging)
- customer demands

10.2 Evolution of current models

The Economic Order Quantity (EOQ) is the first model for the management of multi-stage inventory introduced in Harris (1913). A typical EOQ model deals with a single item and a single manufacturing machine. In his model Harris derives the optimal quantity to order based on a set of predefined constants: demand rate, setup cost, cost per item for a continuous time scale and infinite time horizon.

Wagner and Whitin (1958) studied the first mathematical lot-sizing model. Their model is based on demands of a single item with inventory holding and setup costs, which differ over the set of N periods of time. The model seeks a minimum total cost of inventory management. Subsequently a variety of the model extensions were built to take into consideration multiple items and multiple stages.

In the Economic Lot Scheduling Problem (ELSP) (Elmaghraby (1978)) multiple items with constant demand rate are considered. The model divides the manufacturing process into cycles that follow similar patterns. ELSP seeks an overall minimum cost over the time horizon and determines the start and finish times, the processing sequence and the machine loading for each job.

Potts and Van Wassenhove (1992) built lot-sizing schedules for multiple items while batching similar jobs in order to minimize setup times and costs. In their model, items are grouped based on setup costs. Kuik, Salomon, and Van Wassenhove (1994) group items of the manufacturing process to induce time-phased production that seeks servicing for fluctuating demand patterns

The production inventory management problem is naturally coupled with uncertainty and randomness in its customers' demand, setup-times and suppliers' capacity. This uncertainty has a large impact on the cost of inventory, due to either excess in inventory sizes that are carried out, or on the contrary, stock-out that yields unsatisfied demands. In planning, there is a need to consider actual and forecasted information that is under a dynamic environment. To deal with uncertainty and unforeseen demands, existing models simplify fluctuating information to create deterministic versions of the problem that is easier to deal with. The effect of modeling using the deterministic case inevitably yields errors that impact the behavior and cost of the production procedure.

10.3 Modeling Aspects of Inventory Management

The management of inventory defines the structure of policies involved in monitoring inventory levels. Accordingly, planning the time and quantity of items to be ordered should maintain adequate inventory levels. Modeling the operational aspects of inventory defines the setup time, the production process, the inventory levels, the replenish-

ment policies, the customers' demand and the rolling time horizon. More strategic models define the lot sizing problem as a substructure that is integrated in the manufacturing and distribution planning process, which might include supplier selection.

10.3.1 Lead/Setup Time

Lead time is the time taken once the order is placed until the item is processed. Setup time may vary for different operations at different times and it adds up cost on the inventory operation. Basic models often either neglect cost that is due to the setup time or consider it unchangeable over time for simple computation.

10.3.2 Replenishment Policies

Inventory management modeling depends on the policy adopted by the management. Replenishment policies play a major role in availing material to the production process. Policies determine the number, time and quantity of replenishments. Commonly utilized inventory replenishment policies are periodical and continuous. Periodical replenishment policy sets constant time intervals between ordering requests; whilst continuous replenishment policy seeks to monitor the stock level, and triggers an ordering request when it reaches a predefined threshold. The challenge is to find an optimal replenishment policy that meets customer demands with minimal cost. Replenishment policy determines the reorder point, i.e. the point an order takes place. It is commonly symbolized by T and R respectively for periodic time and continuous reorder point policies.

10.3.3 Customer Demands

Customer demands are obviously the most unpredictable component of the inventory management problem. Stochastic models use a variation of forecasting methods to plan and schedule ahead their orders; observations output from the forecasted demands are generally simplified to the nearest probability distribution (usually the Normal distribution) and they are delt with as points of expectations; the model is then approximated to the deterministic version using resulting points of expectation in order to simplify the computation.

10.4 Measured Output

Existing models of the inventory management seek to find the optimal quantity to order at a specific time that achieves the minimum possible cost. To the best of our knowledge current modeling results are commonly given and delt with as point values. The problem has a fluctuating nature that yields a probability distribution for each output.

10.4.1 Quantity to Order

The well known model EOQ of Harris (1913) builds basic equations for the quantity to order under a deterministic and a predefined set of demands. Derived equations seek to find the optimum quantity which must be ordered and that achieves the minimum overall cost. The model and its extensions consider the deterministic case, while produced outputs are point values even in stochastic and *fuzzy* versions.

10.4.2 Reorder Point

Reoder point defines a threshold on the level of inventory that when reached an order should take place. This level is a safe guard lead time that ensures the smooth transaction of the order without interrupting the manufacturing process. Both reorder point and lead time can vary over time because of external environmental circumstances such as the time taken by the shipment placement.

10.4.3 Inventory Costs

The cost of inventory incorporates:

- Holding cost of excess ordered quantity in stock
- Setup cost of preparing product components
- Ordering cost of purchasing production items
- Shortage cost of unmet customer demands which include unforeseen switching cost

Total cost is defined as the sum of all previously specified costs.

10.5 Basic Model

The basic inventory management deterministic model EOQ draws cost equations and adds them up to find the optimal quantity to order which is the lowest point on the curve resulting from the addition (Harris (1913)). Fig. 10.1 shows that cost in general depends on the size of the order. For instance it is cheaper to order items in batches. Also, the cost for holding items in the inventory increases over time due to interest charges and depreciation. The basic model bases its equations on a predefined constant demand rate, setup cost, and cost per item. The equation of the total cost is

$$TC = DC + \frac{D}{Q}S + \frac{Q}{2}H \tag{10.1}$$

Equation 10.1 shows that total cost is the sum of three components:

- purchase cost DC the average annual demand multiplied by the cost per item

- ordering cost $\frac{D}{Q}S$ the number of orders placed multiplied by the cost of each order
- holding cost $\frac{Q}{2}H$ the average level of inventory througout the year multiplied by the holding cost per item

Equation 10.1 is differentiated over the quantity to order Q in order to get the minimum point value for $Q = \sqrt{\frac{2DS}{H}}$; where:

- TC: total cost
- D: annual demand rate
- C: cost per unit item
- S: setup cost
- H: annual holding cost

Reorder point is signified as $R = \bar{d}L$ where $\bar{d}$ is the average daily demand and L is the daily lead time; clearly both components are constants.

Example 10.1. *Harris (1913) shows the application of the EOQ on two different manufacturing items: a copper connector and a stud. The former is an example of a cheap item and the latter is a more expensive part.*

	Connector	*Stud*
Monthly demand rate	1,230	30
Cost per item	\$0.0135	\$5.65
Setup cost	\$2.15	\$1.85
Annual interest and depreciation cost	10%	10%
daily lead time	*2hrs*	*2hrs*
Quantity to order	6,856	49
Reorder point	82 *units*	2 *units*
Ordering cost	\$387	\$13.72
Holding cost	\$342.85	\$2.45
Purchase cost	\$199.26	\$2034
Total manufacturing cost	\$929.11	\$2050.17

Table 10.1: EOQ deterministic model for two manufacturing items: copper connector and stud; an example of a cheap item and an expensive one

From Table 10.1 we can observe that the optimal quantities to order are respectively 6,856 *and* 49 *units of copper connector and stud items. The model suggests issuing the replenishment order when the inventory level reaches* 82 *copper connectors and* 2 *stud units. It is worth noting that the model demonstrates that cheaper items with higher*

demand rate can be kept in the inventory in larger quantities when compared to more expensive items. Derived optimal quantities yield a total manufacturing cost of $929.11 *and* $2050.17 *for the copper connector and the stud respectively.*

Example 10.1 recommends keeping the lowest possible inventory levels especially when items are expensive. This is to avoid the overhead of the holding cost that results from the depreciation and the interest rates. On the other hand managers of the manufacturing process should avoid understock in order to satisfy customer demands.

Clearly Example 10.1 demonstrates that in the basic EOQ model demand rate is given as an average annual value; it is a constant value in the deterministic model; moreover, the item cost and the lead time are unchangeable over time and the interest rate is incorporated in the model for both items as one value: 10%.

Table 10.2 shows a simulation of the EOQ when applied in a real life situation: the stud manufacturing process. The first row monitors time cycles over the year. Monthly customer demand is varying and it is given in the second row but it is on average 30 items per month. We started the simulation with an empty inventory level. Negative values of inventories signify unsatisfied demands in the observed cycle. The simulation considers two cases: 1. we allow backlogs to satisfy unmet demands and 2. no backlogs are allowed in the second situation. In the latter case unsatisfied demands are not served in the following cycle and cause a penalty which is evaluated by a shortage cost.

Clearly when backlogs are allowed, more replenishments are issued and ordering cost increases. The order size exeeds the value 49 in order to satisfy unmet demands from previous cycles; yet inventory holding cost decreases because excess ordered items are promptly processed to cover backlogs. Shortage cost is 0 because all demands are satisfied in this case. In general the overall total manufacturing cost when backlogs are allowed is $1013.65 and it is $1037.25 when no backlogs are allowed. In both cases, the total manufacturing cost is 49% – 50% of the figure obtained from the EOQ determistic model. We can conclude from the simulation results that the total cost obtained from the deterministic case is not accurately calculated. Managers of the manufacturing process need to know in advance, the order quantities and the total costs that reflect real life situations in a better way, because they are used in their decision making and budget allocation procedures.

10.6 Existing Approaches

10.6.1 Dynamic programming line of research

Dynamic Programming (DP) was the first line of research that considers the problem of inventory management. Since DP is applied in problems that contain a sequence of interrelated decisions, the algorithm seeks optimization by dividing the problem into a more simplified set of nested subproblems. The DP model sets the objective cost function that needs to be minimized, a recursive formulation of the cost function at state i, and boundary condition at the initial state. It is worth noting that the DP models

		$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$
	Actual Monthly Demands	26	36	23	28	32	30	29	37	25	34	
No backlogs	Reorder point	1	0	1	0	1	0	1	0	1	0	0
	Order size	49		49		49		49		49		
	Inventory ending level		23	-13	26	-2	17	-13	20	-17	24	-10
	Costs	Setup		Holding		Purchase			Shortage		Total Cost	
		$11.1		**$10.93**		**$1384.25**	**-$617.545**		**$259.335**		**$1048.07**	
Backlogs are allowed	Reorder point	1	0	1	0	1	1	0	1	0	1	1
	Order size	49		36		34	51		41		28	43
	Inventory ending level		23	-13	13	-15	2	21	-8	4	-21	-6
	Costs	Setup		Holding		Purchase			Shortage		Total Cost	
		$12.95		**$6.3**		**$1350.35**	**-$355.95**		**$0**		**$1013.65**	

Table 10.2: Simulation of the EOQ on actual customer demands for 10 cycles

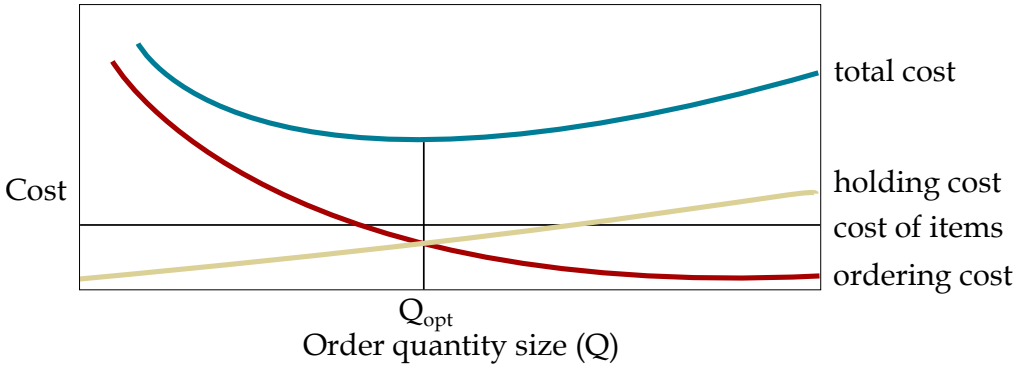


Fig. 10.1: Size of the quantity to order

do not scale well in a stochastic environment. Existing DP models such as those in Clark and Scarf (1960) and Moon and Gallego (1994) approximate the problem to the deterministic case in order to overcome the complexity of the recursive formulation. They assume full knowledge of the demand distribution. In addition, they are restricted to the specification of serial supply chain network topology. It is worth noting that DP utilizes myopic policies for search; i.e. it is restricted to finding the minimum cost of the current time period. Recently, hybrid optimization techniques incorporationg DP and CP have been proposed in Rossi (2008).

10.6.2 CP line of research

CP has been successfully utilized for deterministic planning and scheduling models. Yet the technique lacks an accurate representation of the model, particularly when randomness and uncertainty are introduced in the problem.

To model the problem stated in Example 10.1, a CP classical model defines the total cost minimization constraint deterministically and describes the quantity to order as a variable that is assigned a domain of integers. CP searches for the quantity to order which achieves a total minimum cost. Table 10.3 illustrates the search that is exerted by enumerating all possible values of the quantity to order within the assigned domain of integers. The enumerated values are then substituted in the total cost constraint equation. The result of the enumeration is 362 and 795 which give total minimum costs of $278.93 and $2070.49 for the connector and the stud manufacturing items respectively.

10.6.3 Stochastic Constraint Programming (SCP) line of research

The stochastic newsvendor model refers back to the work introduced by the prominant economist Edgeworth in (Edgeworth (1888)) who mathematically represented the amount of money to be deposited in a bank account with a random cash withdrawal.

Due to its powerful expressiveness, SCP is an intuitive way of modeling combinatorial problems coupled with stochasticity. It is generally used in CP for optimization

Connector		Stud	
Q	TC	Q	TC
600	282.15	200	2077.3
601	282.11	201	2077.18
...	...	...	...
795	278.93	362	2070.49
...	...	...	...
812	278.94	370	2070.5
...	...	...	...
817	278.95	270	2072.16

Table 10.3: CP enumeration of the quantity to order size for the total cost calculation

under uncertainty. Stochastic models aim at numerically representing the uncertainty brought into the problem under consideration. They iteratively generate potential outputs for the set of input randomly distributed data in a pointwise manner. This iterative process in turn produces probable solutions that follow random distributions and accordingly realization of the maximum likelihood of projected outcomes are explored.

SCP divides the problem into decision stages; each stage consists of a pair (V_i, S_i) where V_i and S_i are the sets of decision variables and stochastic variables respectively. One-stage Stochastic Constraint Satisfaction Problem (SCSP) (V, S_i) assigns decision variables in V for each stochastic variable value in S. This assignment should satisfy hard-constraints and all possible scenarios specified by stochastic variables. the m-stage SCP partitions the problem into disjointed sets of multiple stages (V_i, S_i). Decision variables in a stage are realized by observing former stages. The solution is represented by a policy tree where each path represents variable assignments to a given scenario.

Solution methods are based on two approaches: policy based and scenario based. The former, in Walsh (2002), establishes the tree paths by assigning values to variables in order to satisfy possible scenarios. Decision variables can take only one value, hence they are assigned to the OR nodes, whilst stochastic variables list all possible events and they are given the AND nodes. The model realizes one value that satisfies all possible constraints. The scenario based approach, in Tarim et al. (2006), exhaustively builds the tree of all possible sets of scenarios in a list of paths. Each scenario (path) is solved with conventional deterministic CP on its own. A path in the tree is associated with a probability value. In this model the number of scenarios grows exponentially with the number of stages. A global chance constraint, introduced in Rossi (2008), combined SCP with DP to calculate the cost using the minimum distance algorithm. It is worth noting that SCP is more flexible and powerful in modeling inventory management problems coupled with randomness. Yet SCP does not scale well when it involves solving the actual problem.

Table 10.4 shows the scenario-based approach enumeration of the quantity to order

given in Example 10.1. In this example, each cost component (the ordering, the holding and the purchasing costs) is associated with a probability distribution function. The SCP approach builds the scenario tree by associating each path with a probability of occurrence. This probability is computed by multiplying each and every combination of the probabilities given for the cost components. The total cost with the highest probability of occurrence is then evaluated. Finally, the quantity to order size selected is the most likely to occur and at the same time it satisfies the total minimum cost constraint. It is shown in bold fonts as 197 and 3216 for the stud and connector manufacturing items. Their highest probability values are: 0.18 and 0.264 respectively. As shown by this example, the stochastic computation can be infeasible because they are exerted on the given probability distributions exhaustively, i.e. in a point-by-point manner.

	Interest & depreciation	**Setup cost**	**Prob**	**Q**	**Ordering cost**	**Holding cost**
		$12.5	0.02	85	$62.50	$6.53
	4.45%	$18.5	0.05	103	$74.00	$9.65
		$26.1	0.03	122	$78.30	$13.61
		$12.5	0.12	136	$37.50	$6.36
Stud	11.60%	$18.5	0.3	166	$55.50	$9.50
		$26.1	0.18	197	$52.20	$13.43
		$12.5	0.06	179	$37.50	$6.33
	20.10%	$18.5	0.15	259	$37.00	$22.97
		$26.1	0.09	259	$52.20	$13.33
		$1.075	0.039	1608	$10.75	$0.564
	1.10%	$2.15	0.082	2274	$15.05	$1.15
		$4.3	0.049	3216	$21.5	$2.36
		$1.075	0.13	2274	$7.5	$0.56
Connector	2.20%	$2.15	0.264	3216	$10.75	$1.13
		$4.3	0.16	4548	$17.2	$2.3
		$1.075	0.064	3216	$5.38	$0.55
	4.40%	$2.15	0.134	6432	$6.45	$4.4
		$4.3	0.08	6432	$12.9	$2.26

Table 10.4: Scenario-based CSP enumeration of the quantity to order size for the total cost calculation

10.6.4 Mixed integer programming line of research

Bookbinder and Tan (1988) used the static-dynamic uncertainty strategy to determine the optimal policy. Their model formulates the problem on two stages: first determine the replenishment periods; then seek the actual orders upon demand realization. In their model Bookbinder and Tan assume that demand is a random variable following a normal probability distribution. They build their deterministic equivalent model on the demand

expectations for each period and they assumed constant item cost, which has no role in the replenishment policy. Hence it has no effect on determining the best solution of the schedule.

Definition 10.1. *The uncertain linear programming inventory model for mixed integer programming is formulated as follows:*

$$\begin{aligned} \textit{minimize} \quad E\{TC\} = \int_{d_1}\int_{d_2}\cdots\int_{d_N}\sum_{t=1}^{N}(a\delta_t + hI_t + vX_t) \\ \times g_1(d_1)g_2(d_2)\ldots g_N(d_N) \\ \textit{subject to} \quad \delta_t = \left\{ \begin{array}{ll} 1 & \textit{if } X_t > 0 \\ 0 & \textit{otherwise} \end{array} \right\} \\ I_t = I_0 + \sum_{i=1}^{t}(X_i - d_i) \\ X_t, I_t \geq 0, \quad t = 0, 1, \ldots, N \end{aligned} \tag{10.2}$$

Bookbinder and Tan (1988) used this model in order to reflect the fluctuations of the inventory levels in each cycle over the time horizon. In this model the three cost components are:

- The ordering cost denoted by a (the cost per item) and mutiplied by δ_t which represents the number of times an order is issued. $\delta_t = 1$ when an order is issued and $\delta_t = 0$ otherwise.

- The holding cost depends on the inventory level I_t at cycle t, where t is a given period/cycle in a time horizon from $\{1 \ldots N\}$.

- The purchase cost varies based on the item cost v multiplied by the quantity to order size R_t at cycle t.

The customer demand d_t defined over a time period t is often given by a known probability density function $g(d_t)$. The inventory level I_t is the difference between the order quantity R_t, and the customer demands d_t in a given cycle. The problem in this case has a total time horizon equal to N cycles. And the model seeks the minimization of the total cost expected value that is symbolized by $E\{TC\}$.

Bertsimas and Thiele (2006) proved that mixed-integer programming outperforms DP in terms of computation and cost results. They used a mixed-integer programming model to find bounds on the solution of the cost. In their computation, a convex representation which encapsulates the unknown probability distribution is derived; yet the output obtained does not contain any information on the data whereabouts.

10.6.5 Probabilistic programming line of research

Probabilistic approaches are flexible enough to describe the uncertainty in real-world problems. They add a quantitative information that expresses the likelihood. Operations are exerted on probability distributions which approximate the observed data.

Probabilistic models often define the uncertainty over the presence of constraints. They impose assumptions on the probability distributions shape to deal with them mathematically by means of the expected values and the standard deviation. Final decision is an assignment that maximizes the probability of consistency which satisfies the set of the given constraints. Tarim and Kingsman (2004) developed a probabilistic mixed-integer programming algorithm based on the work introduced by Bookbinder and Tan and which combines the multi-steps into a single one. The probabilistic inventory model proposed by Tarim and Kingsman (2004) is defined as follows:

$$
\begin{aligned}
\text{minimize} \quad & E(TC) = \sum_{t=1}^{N} (a\delta_t + hE(I_t) + vE(X_t)) \\
\text{subject to} \quad & E(I_t) = E(X_t) - E(d_t), \\
& E(X_t) \geq E(I_{t-1})), \\
& E(X_t) - E(I_{t-1}) \leq M\delta_t, \\
& E(I_t) \geq \sum_{j=1}^{t} (G^{-1}_{d_{t-j+1}+\cdots+d_t}(\alpha) - \sum_{k=t-j+1}^{t} E(d_k))P_{tj}, \\
& \sum_{j=1}^{t} P_{tj} = 1, \\
& P_{tj} \geq \delta_{t-j+1} - \sum_{k=t-j+2}^{t} \delta_k \\
& E(X_t), E(I_t) \geq 0, \\
& \delta_t, P_{tj} \in \{0, 1\}, \\
& t = 1, \ldots, N \\
& j = 1, \ldots, t
\end{aligned}
\tag{10.3}
$$

In this model, an additional service level constraint is added to guarantee that the inventory levels are always kept positive with a probability greater than a value α ($Pr(I_t \geq 0) \geq \alpha$). This is represented by the equation which relates the inventory level I_t with the variations of the monthly demand d_k by means of the cumulative probability distribution function $G_{D(t)} = G_{d_1} \ldots G_{d_t}(\alpha)$ defined for $\{d_1 \ldots d_t\}$. In this equation variations of the monthly demand are associated with the inventory level and their constraint relation is satisfied with a certain probability value symbolized by P_{tj}. The set of constraints in this system are defined over the expected values of the inventory $E(I_t)$ and the monthly demand $E(d_k)$.

Table 10.5 shows the input of the stud manufacturing item detailed in Example 10.1. The number of calculated ordering placements is 4 and the computed total cost is $2142.15. The ordering quantity differs with each order placement depending on the monthly demand and the level of inventory reached.

10.6.6 *Fuzzy* programming line of research

Fuzzy models introduce the vague representation of demands and costs. They approximate the probability distribution by a set of intervals called 'alpha-cuts'. The possibilistic theorem, Zadeh (1965), is applied on the alpha-cuts to reason about the data in a simplified way. In the inventory management problem, *fuzzy* sets are estimated subjectively to represent the uncertainty observed in demands and costs. The possibilistic distribution of the input data is built based on a computed standard deviation: assuming, in the general case, a unimodal and symmetric distribution of the probability Gum (1995). Demands and costs are described by *fuzzy* membership functions. Dutta, Chakraborty, and Roy (2005) use a triangular membership *fuzzy* function in their model, and Xu and Hu (2012) model uncertain customer demands by means of random *fuzzy* variables.

The *fuzzy* model formulation depends on an 'extended addition' ($\oplus$) that is based on the 'Yagers parametrized t-norm', Yager (1997). Operations on data using the possibilistic theory exert the 'Dominance possibility'. The possibilistic distribution accordingly provides worst and best case scenarios represented respectively by the support and the kernel of the possible solution set. In order to obtain the resulting solutions, cost and demand are subject to fuzzification and defuzzification algorithms to generate the expected profit, then to calculate bounds on the optimal quantity to order. This operation is exerted, while maximizing the expected profit and minimizing the expected cost. As a result, the optimum quantity to order is calculated in a discrete manner, i.e. for each possible given demand with a *fuzzy* representation of the holding and ordering costs.

Definition 10.2. *The fuzzy inventory model is formulated as follows:*

$$\begin{aligned} \textit{minimize} \quad & TC = \sum_{t=1}^{N} (a\delta_t \oplus hI_t \oplus vX_t) \\ \textit{subject to} \quad & \delta_t = \left\{ \begin{array}{ll} 1 & \textit{if } X_t > 0 \\ 0 & \textit{otherwise} \end{array} \right\} \\ & I_t = I_0 \oplus \sum_{i=1}^{t} (X_i \ominus d_i)\mu_i \\ & X_t, I_t \geq 0, \quad t = 0, 1, \ldots, N \end{aligned} \tag{10.4}$$

Table 10.6 shows results of running the *fuzzy* model on the stud manufacturing item listed in Example 10.1. In this model the variations of the monthly demand follow triangular shape distributions, and constraint preferences are represented by means of *fuzzy* membership functions μ_i. The calculated total cost is $2702.93.

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$	$t = 6$	$t = 7$	$t = 8$	$t = 9$	$t = 10$	$t = 11$
Monthly demands	32	30	26	36	23	29	37	25	34	28	32
Reorder point	1	0	1	0	0	1	0	0	1	0	0
Order quantity	62	0	85	0	0	91	0	0	94	0	0
Initial I_t	62	30	0	59	23	0	62	25	0	60	32
Ending I_t	30	0	59	23	0	62	25	0	60	32	0
Setup cost											$74
Holding cost											$34.15
Purchase cost											$2034
Total cost											$2142.15

Table 10.5: Probabilistic model calculation of the quantity to order size and the total cost of the stud manufacturing item

	$t = 1$	...	$t = 5$	$t = 6$	...
Monthly demands	[23.9, 24.3, 25]	...	[29.7, 29.95, 30.3]	[31.3, 32.2, 32.9]	...
Membership function	[0.08, 0.16, 0.1]	...	[0.08, 0.16, 0.04]	[0.02, 0.12, 0.06]	...
Reorder point	1	...	1	0	...
Order quantity	56	...	89	0	...
Ending I_t	30	...	57	0	...
Setup cost					$67.2
Holding cost					$9.05
Purchase cost					$2626.68
Total cost					$2702.93

Table 10.6: *fuzzy* model calculation of the quantity to order size and the total cost of the stud manufacturing item

10.6.7 Reliable programming line of research

Reliable techniques suggest convex structures, interval or ellipsoidal, to guarantee a full data enclosure in the model when uncertainty takes place. The certainty closure (UCSP) introduced by Yorke-Smith and Gervet (2009) associates the uncertainty to constraint coefficients. Interval coefficients are used to bracket the ill-defined data. This framework brings together modeling and solving methodologies from LP into the CP paradigm to provide reliable and efficient approaches for uncertain constrain problems.

This convex model is a modified version of the probabilistic model proposed by Tarim and Kingsman (2004) as listed in Equation 10.3 but it incorporates convex structures which are boldified to represent interval coefficients. Variables are no longer given as expected values because in the convex model, we seek the bounds which encapsulate realized solution sets. The inverse of the cumulative probability distribution $G^{-1}_{D(t)}$ is replaced by the monthly demand intervals in the service level constraint. Input data interval coefficients are produced by shaping a Normal distribution over the observations. Interval bounds are then assigned the maximum and minimum values to include the majority of the Normal distribution data population.

Definition 10.3. *The convex model formulation of the inventory management problem is defined as follows:*

$$
\begin{aligned}
\textit{minimize} \quad & TC = \sum_{t=1}^{N} (\boldsymbol{a}\delta_t + \boldsymbol{h}I_t + \boldsymbol{v}X_t) \\
\textit{subject to} \quad & I_t = X_t - \boldsymbol{d_t}, \\
& X_t \geq I_{t-1}, \\
& X_t - I_{t-1} \leq M\delta_t, \\
& I_t \geq \sum_{j=1}^{t} \boldsymbol{d_{t-j+1}} + \cdots + \boldsymbol{d_t} - \sum_{k=t-j+1}^{t} \boldsymbol{d_k}, \\
& \sum_{j=1}^{t} P_{tj} = 1, \\
& P_{tj} \geq \delta_{t-j+1} - \sum_{k=t-j+2}^{t} \delta_k \\
& X_t, I_t \geq 0, \\
& \delta_t, P_{tj} \in \{0, 1\}, \\
& t = 1, \ldots, N \\
& j = 1, \ldots, t
\end{aligned}
\tag{10.5}
$$

Reasoning using convex models is a tractable computation because it is constructed over the extreme points of the algebraic structure, unlike exhaustive point computation found and exerted in other paradigms. Results of convex models are reliable and guarantee to carry-out all potential solutions of the problem in-hand. To derive outer bounds, the model is based on an approximation that is not necessarily reversed. Acquired reliable realizations can be very wide lacking an expressive approximation of the problem in-hand and missing possible degree of knowledge because each value in the solution set has an equal uncertainty degree.

Table 10.7 shows results of running the convex model on the stud manufacturing item listed in Example 10.1. In this model the variations of the monthly demand are given as interval bounds. The number of ordering placements is given an interval representation. The model at $t = 10$ suggests a no replenishment at the lower bound an order issuing at the upper bound. The total setup, holding and purchase costs are given by an interval representations. This means that each value within the interval has the same probability of occurrence. The calculated total cost is any point lying within the interval bounds [\$739.5, \$4458.3]. Despite the fact that the resulting solution set guarrantees a full encapsulation of the data, convex models lack the expressiveness of the probabilistic information.

	$t=1$	...	$t=5$	...	$t=10$	...
Monthly demands	[23.9, 25]		[29.7, 31.3]		[37.6, 39.1]	
Reorder point	[1, 1]	...	[0, 0]	...	[0, 1]	
Ending I_t	[114.6, 115.7]	...	[0.0, 6.4]	...	[0, 1.5]	
Setup cost						[$22.4, $66.3]
Holding cost						[$16.65, $190.29]
Purchase cost						[$1496.7, $2077.18]
Total cost						[$1535.7, $2333.77]

Table 10.7: Convex model calculation of the quantity to order size and the total cost of the stud manufacturing item

10.7 Modeling Inventories with p-box *cdf*-intervals representation

This section details the construction of the p-box *cdf*-interval model for the inventory management problem and which is an extension of the convex model representation listed in Equation 10.5. Recall Equation 10.1 of the EOQ model, the cost function has three main components: purchase cost, holding cost and ordering cost. Each cost component has a stochastic nature; for instance the purchase cost varies on the size of the order; the ordering cost depends on number of times orders are issued; and the holding cost varies with the size of items in-stock. The p-box *cdf*-intervals CP expressive nature enables us to build the EOQ set of constraints without introducing any approximation. Accordingly, probability distribution of demands, ordering sizes, inventory levels, reorder points and cost components are encapsulated and represented in a reliable manner in order to consider all possible realizations of the problem in hand.

In this section the subscript t denotes the snapshot of the problem at a given time period/cycle. I is the inventory level, X is the quantity to order and d is the customer demands. Boldified characters are p-box *cdf*-interval representations of constraint coefficients.

10.7.1 Ordering Cost

The ordering cost incorporates the number of orders and the cost of issuing an order. Recall from Equation 10.1 the EOQ model assumes the number of issued orders as D/Q where D is the annual average customer demand and Q is the total size of ordered quantity (Harris (1913)). This assumption is based on the expected discrete value of the demand which is by nature unpredictable. In our model we symbolize the issuing of a replenishement with δ; it is an unknown variable assigned from the discrete domain of values $\{0, 1\}$. δ takes the value 1 when a replenishment is scheduled in the current time period and 0 otherwise. δ is then multiplied by the ordering cost **a** that depends on the

fluctuating market prices and which is represented by a p-box *cdf*-interval that bounds the real distribution of the cost.

Definition 10.4. *The total ordering cost of the inventory management problem for N cycles is*

$$OC = \Sigma_{t=1}^{N} \boldsymbol{a}\delta_t \tag{10.6}$$

where a is the p-box cdf-interval representation of the ordering cost, $t = 1..N$ *and* $\delta \in [0, 1]$

10.7.2 Holding Cost

Inventory levels and depreciation charges are the main aspects that affect the holding cost. The EOQ model assigns a discrete value for the average inventory level equal to $Q/2$. However, in reality, the current inventory level is the difference between the ordered quantity and the realized consumer demands; it is defined as $I_t = X_t - d_t$. Depreciation and interest charges are in turn fluctuating based on market prices and they are denoted by **h**.

Definition 10.5. *The total holding cost of the inventory management problem for N cycles is*

$$HC = \Sigma_{t=1}^{N} \boldsymbol{h} I_t \tag{10.7}$$

where $\boldsymbol{h}$ *is the p-box cdf-interval representation of the holding cost.*

$$I_t = X_t - d_t \tag{10.8}$$

where X_t *is the size of the order variable at cycle t and* d_t *is the p-box cdf-interval representation of the customer demand in cycle t*

10.7.3 Purchase Cost

When a replenishment is scheduled, the cost of purchasing depends on the order size is X and the varying cost per unit item is **v**.

Definition 10.6. *The purchase cost of the inventory management problem for N time periods is*

$$PC = \Sigma_{t=1}^{N} \boldsymbol{v} X_t \tag{10.9}$$

where $\boldsymbol{v}$ *is the p-box cdf-interval representation of the unit item cost.*

10.7.4 The Model

The p-box *cdf*-intervals model is an extension of the convex model representation listed in Equation 10.5. The model hybridizes the CP high expressivity of uncertainty given by the p-box *cdf*-intervals representation with the fast and easy way of solving systems of linear equations given by the MIP to find the bounds on the quantity to order and

the total cost. Output realizations envelop the actual solution set of the problem along with its data whereabouts. The aim is to schedule ahead replenishement periods and find the optimal quantity bounds that achieve minimum total manufacturing cost using cost components introduced in this section. A reorder point with order size X_t should meet customer demands up to the next point of replenishment.

To maintain a certain service level, inventory management models either assign a penalty cost on shortage or maintain inventory levels to a minimum threshold in order to ensure no backlogs. The former criteria is presented in Equation **??** and the latter adds a new constraint $Pr\{I_t > 0\} \geqslant \alpha$; the additional constraint maintains a positive inventory level with a probability greater than a predefined constant α.

Definition 10.7. *The set of p-box cdf-intervals constraints that define an inventory management problem given N cycles time horizon are*

$$
\begin{aligned}
\textit{minimize} \quad & TC = \sum_{t=1}^{N} (\boldsymbol{a}\delta_t + \boldsymbol{h}I_t + \boldsymbol{v}X_t) \\
\textit{subject to} \quad & I_t = X_t - \boldsymbol{d_t}, \\
& X_t \geq I_{t-1}, \\
& X_t - I_{t-1} \leq M\delta_t, \\
& I_t \geq \sum_{j=1}^{t} \boldsymbol{d_{t\text{-}j+1}} + \cdots + \boldsymbol{d_t} - \sum_{k=t-j+1}^{t} \boldsymbol{d_k}, \\
& \sum_{j=1}^{t} P_{tj} = 1, \\
& P_{tj} \geq \delta_{t-j+1} - \sum_{k=t-j+2}^{t} \delta_k \\
& X_t, I_t \geq 0, \\
& \delta_t, P_{tj} \in \{0, 1\}, \\
& t = 1, \ldots, N \\
& j = 1, \ldots, t
\end{aligned}
\tag{10.10}
$$

N can be set to one in order to characterize the EOQ p-box *cdf*-intervals with one cycle hence results can be easily compared with previous simple models. It is worth noting that the set of constraints in Definition 10.7 symbolize demands, costs, and inventory levels by p-box *cdf*-intervals; i.e. they encapsulate the unknown fluctuating nature of the problem.

10.8 Evaluation of the model

Recall the stud item in Example 10.1, Figure 10.2 illustrates the demand observations along with their corresponding probabilities over a time horizon equals to 10 cycles.

The dotted mid-line represents the mean demands; this dotted line is in turn enveloped by two probabilistic upper and lower bounds. Clearly, demands and input variable costs vary within each cycle. In this section we study the observed information for 10 different models to compare and study the effect of adopting deterministic, probabilistic, *fuzzy*, SCSP, MIP, UCSP, *cdf*-intervals and p-box *cdf*-intervals on the inventory problem quality of solution. Table 10.8 lists different input variable formats to the models under consideration. Demands depicted in Figure 10.2 are shown in the last column of Table 10.8. Our empirical evaluation considers the implementation of the following list of models:

- EOQ is the basic deterministic inventory management model detailed in Section 10.5. This model takes one average value of the observed demands over the year as shown in Equation 10.1 and in this case it is 30 items.

- Probabilistic, is listed in the set of constraints defined by Equation 10.3. This model commonly assumes that data is normally distributed; in each cycle the probabilistic model takes the mean value of the observed customer demand and this is shown in Table 10.8.

- *fuzzy* (Dutta), as shown by the set of constraints listed by Equation 10.4, uses the *fuzzy* approach adopted in Dutta et al. (2005); *fuzzy* customer demands are listed in the last column of Table 10.8; each input in the demand set is itself a *fuzzy* membership function; the model, in turn, generates an overall *fuzzy* membership function of the optimal quantity to order that achieves the maximum possible expected profit.

- Petrović I, (Petrović, Petrović, and Vujošević (1996)), is another *fuzzy* approach that represents customer demands by *fuzzy* sets; the approach uses *fuzzification* and *defuzzification* techniques in order to deduce one value for the optimal quantity to order.

- Petrović II in Petrović et al. (1996) has an additional advantage over Petrović I: it deals with imprecise inventory costs: overage and shortage costs; they are in this case represented as *fuzzy* sets

- MIP in Tarim and Kingsman (2004), is a MIP approach listed by Equation 10.2. In this model, input demands are represented by their observed distribution mean per cycle.

- SCSP in Rossi (2008), a constraint based programming approach which combines SCP with DP to calculate the overall inventory management cost using the minimum distance algorithm.

- UCSP in Yorke-Smith and Gervet (2009), uncertainty closure uses the set of constraints defined by Equation 10.5 to calculate the bounds on the cost and optimal

quantity to order; demands in UCSP can be input as a set of intervals representing observed bounds per cycle or as an overall interval bounding the measured data over the year.

- *cdf*-intervals model is an extension of the UCSP model where demands are represented by an approximated set of *cdf*-intervals

- p-box *cdf*-intervals has the unknown distribution of demands bounded by p-box *cdf*-intervals representation to encapsulate all observed data along with its whereabouts.

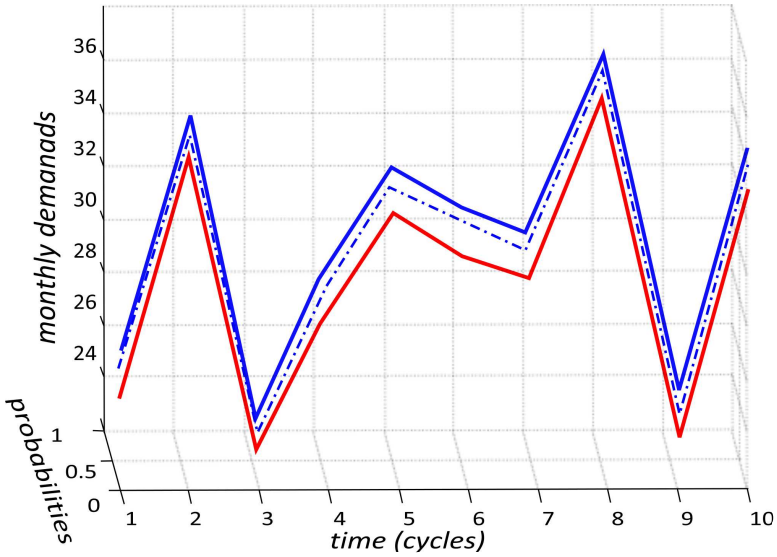


Fig. 10.2: Demand observations over the year

In summary, customer demand information illustrated in Figure 10.2 along with input variables detailed in Table 10.8 can be viewed and delt with from two perspectives:

- Each replenishment order size and time reveals detailed information at each cycle.

- The overall quantity to be ordered and total cost over the year help desision makers allocate and schedule ahead long-term manufacturing budget and process.

10.8.1 Input Coefficients

We study the stud item of Example 10.1. Table 10.8 lists input coefficients to models under consideration. For any observed distribution of data, coefficients are presented as bounds in the 3 models: the UCSP, the *cdf*-intervals and the p-box *cdf*-intervals.

	ordering cost	holding cost	cost per item	Monthly demands
EOQ	1.85	10%	5.56	30
Probabilistic, *fuzzy* Petrović I, SCSP, MIP	1.85	10%	5.56	[26.2, 35.4, 23.2, 27.8, 32.3, 30.8, 29.3, 36.9, 24.7, 33.9]
fuzzy Dutta	1.85	10%	5.56	[[25.5, 26.2, 27], [34.6, 35.4, 36.1], [23, 23.2, 23.9], [27, 27.8, 28.5], [31.6, 32.3, 33.1], [30, 30.8, 31.6], [28.5, 29.3, 30], [36.1, 36.9, 37.4], [23.9, 24.7, 25.5], [33.1, 33.9, 34.6]]
fuzzy Petrović II	1.85	[4.65%, 10.37%, 16.1%]	5.56	[26.2, 35.4, 23.2, 27.8, 32.3, 30.8, 29.3, 36.9, 24.7, 33.9]
UCSP	[1.25, 2.61]	[4.65%, 16.1%]	[1.06, 11.1]	[23.2, 36.9]
cdf-intervals	[[1.25, 0.04], [2.56, 0.98]]	[[4.65%, 0.08], [15.45%, 0.98]]	[[1.06, 0.08], [9.99, 0.98]]	[[23.2, 0.04], [36.15, 0.98]
p-box *cdf*-intervals	[[1.25, 0.04, 0.8], [2.61, 0.36, 0.26]]	[[4.65%, 0.08, 12.07], [16.1%, 0.88, 7.87]]	[[1.06, 0.08, 0.14], [11.1, 0.4, 0.04]]	[[23.2, 0.04, 0.09], [36.9, 0.43, 0.03]]

Table 10.8: Input variables for the stud manufacturing item: costs and monthly demands observed for 10 cycles

Customer demands/cycle

We consider the raw measured customer demands without imposing approximation on its stochastic nature. We focus on one cycle (cycle 7) given the distribution of demands over the year illustrated in Figure 10.2. Figure 10.3 zooms into the observed customer demands in cycle 7. Figure 10.3[*a*] shows a comparison between the actual measured data and its nearest derived Normal probability distribution. Figure 10.3[*b*] illustrates the *fuzzy* membership function that represents the observed data in cycle 7.

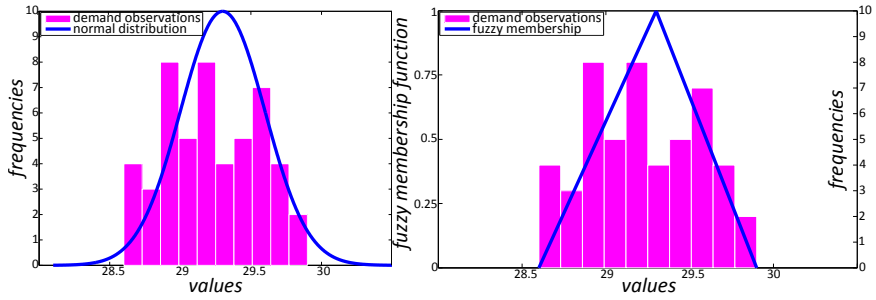


Fig. 10.3: Observed customer demand of the stud item in cycle 7 (a) nearest Normal distribution (b) *fuzzy* membership function of the collected demand

Figure 10.4 illustrates a projection of the observed customer demand in cycle 7 in the *cdf* domain. We use this figure to compare between 4 different approaches: probabilistic, *fuzzy*, *cdf*-interval and p-box *cdf*-interval; and we show how accurate data is encapsulated and presented in each model. Clearly the p-box *cdf*-interval envelops all existing data along with its whereabouts. Customer demand in cycle 7 ranges between quantiles 28.6 and 29.9 with a step value * at least 0.7 and at most 1.1. Using the slope of the *cdf*-interval we can judge where data is accumulated over the interval of quantiles: the average step value in this example is 0.75; *fuzzy* representation approximately encapsulates the data whereabouts, as illustrated in Fig. 10.4(b). However, the Normal distribution, Fig. 10.4(a), which is commonly used in probabilistic models, does not follow the natural shape of the observed data distribution in the *cdf* domain because probabilities have to be between the values $[0, 1]$ for the quantiles in the real domain of $[-\infty,\infty]$.

Similarly, demands for each cycle are derived and compiled into a single set. The produced demand set is then input to the different models, to compare output results and performance. This process can be performed at each cycle as a separate EOQ model myopically in order to minimize the cost belonging to the cycle under consideration.

Forecasted customer demands of a set of cycles that ranges within a predefined time horizon seek to calculate:

*The average step value is the slope of the given *cdf*-distribution in this case its minimum value is equal to 0.7; i.e. if quantile 28.6 has a *cdf*-value 0.08, then the *cdf*-value of quantile 29.6 is 0.78 at minimum

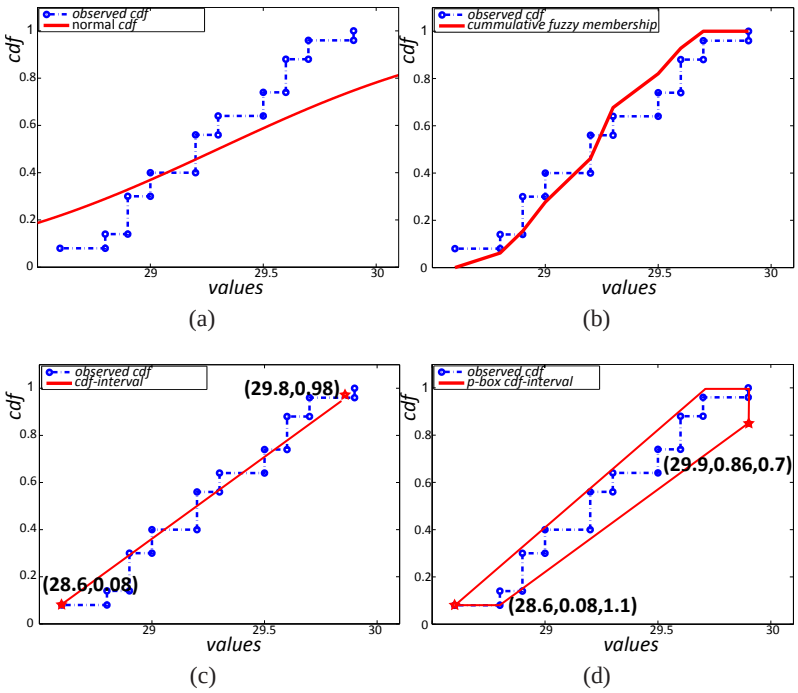


Fig. 10.4: Projection of the collected data in the *cdf* domain (a) Normal *cdf* distribution (b) projection of the *fuzzy* membership function on the *cdf* domain (c) *cdf*-interval representation of demand in cycle 7 (d) p-box *cdf*-interval representaion of customer demand in cycle 7

- reorder points
- each cycle inventory initial and ending levels
- each cycle order up to level
- each cycle setup, overage, purchase, shortage and total costs

Customer demands over a time horizon

Decision makers seek to look at the problem over a given time horizon so that they can schedule ahead a long-term manufacturing process. To showcase this process, we consider the observed data over a 10 cycle time horizon and we monitor the effect of modeling the unknown measured distribution on the representation of data input to the models. Observed customer demands are considered as: one overall mean value in the EOQ model, a set of mean normal distribution value/cycle in the probabilistic model and a set of *fuzzy* membership functions in *fuzzy* models.

On the other hand, models with convex structures are capable of seeking a long term plan in one step/iteration. This method is the fastest and most appropriate when we

need to include the fluctuating nature of the problem over the year while seeking to plan ahead:

- the overall inventory levels to plan for the inventory capacity
- the overall total cost to plan long term budget allocation

Cost components

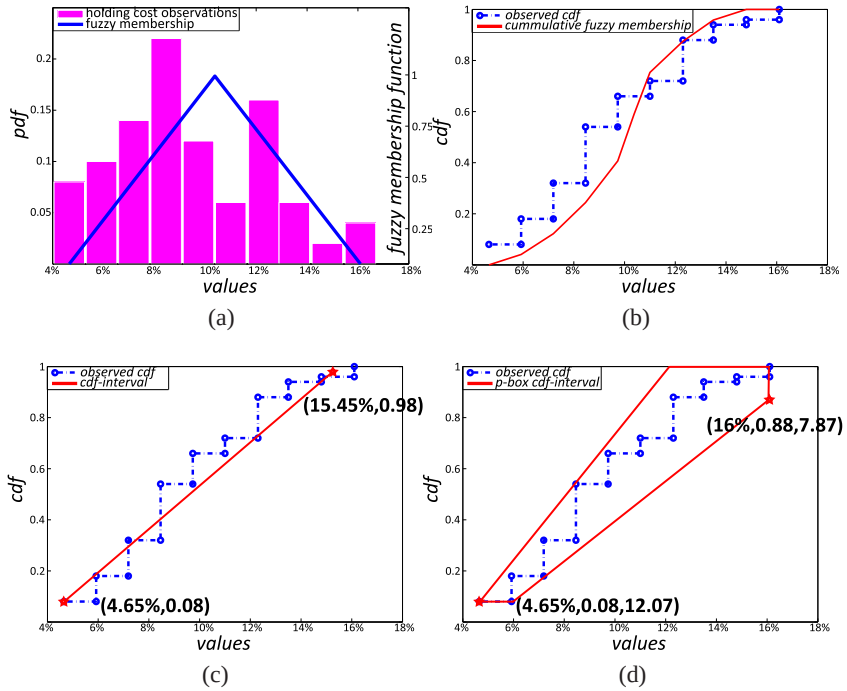


Fig. 10.5: Input measurement of the stud item holding cost over a time horizon of 10 cycles (a) observed measurement and its nearest *fuzzy* membership function (b) projection of the *fuzzy* membership function on the *cdf* domain (c) *cdf*-interval representation (d) p-box *cdf*-interval representaion

Figure 10.5 illustrates how holding cost data is encapsulated in the studied models. The EOQ, stochastic, *fuzzy* Dutta, PetrovićI can deal only with one value, the mean measured value. This is because these models by nature cannot deal with varying distribution or bounds. PetrovićII's model as shown in Table 10.8 handles a *fuzzy* representation of the holding cost in its calculations; and Figure 10.5[*a*] depicts its membership function which ranges within the *fuzzy* interval [4.75%, 10.375%, 16%].

Clearly, holding cost ranges between quantiles 4.75% and 16% which are considered in the UCSP model as interval bounds. The *cdf*-interval model shows that this interval has an average step value of 8.3. The p-box *cdf*-interval further bounds the unknown data distribution by means of two uniform *cdf*-distributions with an average step value

ranging between 7.87 and 12.07; this means that for each quantile above 4.75% and below 16% an increase of 1% has an expected probability value between 0.0787 and 0.1207. Data in this case is totally encaspsulated along with its whereabouts information.

10.8.2 Output Solution

We simulate the models under two different conditions: when backlogs are either allowed or not allowed. In our simulation we utilize real demand data bounds illustrated in Figure 10.2 over a time horizon of 10 cycles. Values in Table 10.8 are input to the models and yield optimal size to order, inventory levels and total cost presented in Tables 10.9, 10.10 and 10.11. We note that incorporated item, setup and variable costs are given as bounds for convex models but they are treated as a single value in the rest of the models under consideration. This is because non-convex models do not represent variables in terms of bounds.

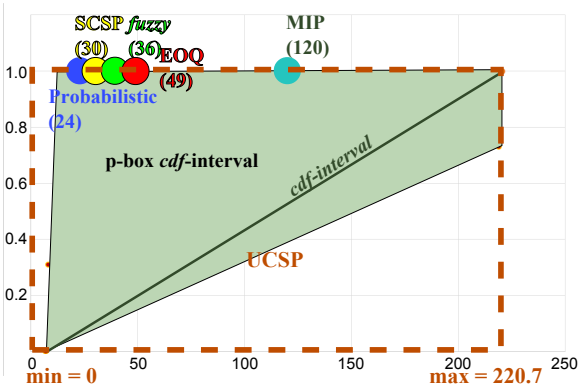
Reorder Points

Reorder points are represented in Table 10.9 column 1 by the set of cycles where an order should be issued. This set ensures that the overall process demands are met and satisfied. Reorder points are recommended by the models to seek the minimization of the overall cost. The output set of reorder points in turn affects the setup and purchase costs and this is illustrated in Table 10.10 columns 1 and 3 respectively. UCSP, *cdf*-intervals and p-box *cdf*-intervals models output lower and upper bound sets in their recommendations. The rest of the studied models each suggests one set in their output solution. For instance, SCSP proposes only two reorder points: at cycles 1 and 3; MIP suggests cycles $\{1, 3, 6, 9\}$ to minimize the overall cost of the stud item manufacturing process example.

Quantity to order

UCSP, *cdf*-intervals and p-box *cdf*-intervals output a convex representation for this quantity: the optimum quantity to order cannot be outside the obained bounds and the stochasticity is captured within input as well as output representation of variables. Fig. 10.6 depicts the graphical representation of the quantity to order output solution from the different models under consideration. The p-box *cdf*-interval is illustrated by the shaded region and the bounds of its convex representation draw the dotted rectangle. Clearly, the solution set obtained from the p-box *cdf*-intervals model, when compared with the outcome of the convex model, realize and additional knowledge (i.e. tighter bounds in the *cdf* domain). This solution set is opposed to a one value proposed as 49 item by the EOQ model due to its deterministic nature. Noticeably, Probabilitic and *fuzzy* models suggest equivalent reorder point value that lies within the same range of quantiles: 24 and 30 items when no-backlogs are allowed and 27 and 58 items when backlogs are allowed. The reason behind the increase in the latter case is due to the need to fulfill

unmet demands from previous cycles. SCSP and MIP propose less number of cycles in the set of reorder points hence to meet customer demands the quantity of the order size is higher: it is 91 and 182 items respectively when backlogs are allowed.



(a)

Fig. 10.6: Output solution representation of the order up to level

Inventory levels and unmet demands

A decision taken on the set of reorder points along with the quantity to order in each cycle control the levels of inventory. Table 10.9 lists bounds on the set of inventory levels over the 10 cycles. In the shown results the upper bound value of the inventory level affects the holding cost of the manufacturing process as listed in Table 10.10 column 2. Negative inventory levels signify unmet customer demands in the observed cycle. This value differs from one model to another since each proposes a different set of reorder points as well as quantity to order. For instance, when no-backlogs are allowed in the manufacturing process, the SCSP value of unmet demands is the highest compared to other models; it reaches a value of 252 items which in turn imposes the highest shortage penalty value that lies within the interval range: $[5910.28, 6371.76] as shown in Table 10.10 column 4. Probabilistic and *fuzzy* models have a similar behavior and obtained unmet demands value reaches: 72 items and their shortage penalty is: $[1576.35, 2050.95]. In general, unmet demands results in shortage penatly when backlogs are allowed is less because in this case unsatisfied demands from previous cycles are usually ordered and met in subsequent cycles. The set of possible solutions is encapsulated in the UCSP model since the model envelops all possible realizations of the problem in hand. However output solution range obtained from the UCSP model is extremely wide. *cdf*-intervals and p-box *cdf*-intervals models seek to meet all observed demands thus they exclude output realizations that contradict the stochastic nature of the problem.

		reorder points	inventory initial levels	inventory ending levels	order up to levels
no-backlogs are allowed	EOQ	$\{1,2,3,4,5,6,7,8,9\}$	$[-15.6, 26]$	$[-15.6, 26]$	49
	Probabilistic	$\{1,2,3,4,5,6,7,8,9\}$	$[-61, 0]$	$[-72, -2]$	24
	SCSP	$\{1,3\}$	$[-217, 4]$	$[-252, 4]$	30
	fuzzy Dutta	$\{1,2,3,4,5,7,8,9\}$	$[0, 37]$	$[1, 37]$	36
	fuzzy Petrović I	$\{1,2,3,4,5,6,7,8,9\}$	$[-61, 0]$	$[-72, -2]$	24
	fuzzy Petrović II	$\{1,2,3,4,5,6,7,8,9\}$	$[-61, 0]$	$[-72, -2]$	24
	MIP	$\{1,3,6,9\}$	$[0, 149]$	$[24, 149]$	120
	UCSP	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[-236, 1222]$	$[-270, 1222]$	$[0, 220.7]$
	cdf-intervals	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[(0, 0), (1239, 1)]$	$[(0, 0), (1239, 1)]$	$[(0, 0), (220.7, 1)]$
	p-box *cdf*-intervals	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[(0, 1, \infty), (1215, 1, 0.43)]$	$[(0, 1, 0.43), (1215, 1, 0.43)]$	$[(6.99, 0, 0.31), (219.9, 1, 0.27)]$
backlogs are allowed	EOQ	$\{1,2,3,4,5,6,7,8,9\}$	$[-40.3, 23.5]$	$[-40.3, 23.5]$	49
	Probabilistic	$\{1,2,3,4,5,6,7,8,9\}$	$[0, 24]$	$[0, 24]$	$[27, 58]$
	SCSP	$\{1,3\}$	$[-38, 67]$	$[-38, 67]$	91
	fuzzy Dutta	$\{1,2,3,4,5,7,9\}$	$[-38, 74]$	$[-38, 74]$	100
	fuzzy Petrović I	$\{1,2,3,4,5,6,7,8,9\}$	$[0, 24]$	$[0, 24]$	$[27, 58]$
	fuzzy Petrović II	$\{1,2,3,4,5,6,7,8,9\}$	$[0, 24]$	$[0, 24]$	$[27, 58]$
	MIP	$\{1,3,6,9\}$	$[-38, 152]$	$[-38, 152]$	182
	UCSP	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[-44, 211]$	$[-69, 211]$	$[0, 238]$
	cdf-intervals	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[(0, 0), (210, 1)]$	$[(0, 0), (210, 1)]$	$[(0, 0), (237, 1)]$
	p-box *cdf*-intervals	$[\{1\}, \{1,2,3,4,5,6,7,8,9\}]$	$[(0, 1, \infty), (210, 1, 0.43)]$	$[(0, 1, 0.43), (210, 1, 0.43)]$	$[(0, 1, 0.31), (237, 1, 0.27]$

Table 10.9: The stud manufacturing item bounds on the inventory levels and reorder points simulated for 10 cycles

Total cost

Total cost listed in Table 10.11 incorporates four components: setup, holding, purchase and shortage costs listed in Table 10.10. Cost components vary and depend on the order size and the inventory levels. Clearly from Table 10.10, purchase cost is the highest portion of the total cost; its interval representation is wider when backlogs are allowed. Setup and holding costs are equal in both cases: backlogs and no-backlogs. Output values from the EOQ, Probabilitic, *fuzzy* are located within realized interval bounds.

p-box *cdf*-interval bounds encapsulate all possible realizations of minimal cost and yield a range of quantile values between $993.67 and $6646.55 when backlogs are allowed. The lower bound quantile obtained indicates that for the same range of input data the model can achieve a better minimal cost value. In case when no-backlogs are allowed in the model this range increases with an upper bound of $9884.87, this is mainly due to the additional shortage cost caused by unmet customer demands. The p-box *cdf*-interval representation further explains that an increase of $1 has an expected probability value that ranges between 0.2% and 12%. Generally, bounds on the total cost obtained help decision makers better plan their budget allocation in comparison with a single value that might not be applied when unconvex models are implemented in real-time.

10.8.3 Scalability of the model

In this section we compare the above studied solvers in terms of scalability. We have implemented 5 data sets with variations of demand distributions.

- Randomly generated monthly demands for 7,10 and 24 cycles Fig. 10.7
- P1 set: demand distribution mean value per cycle is $50(1 + \sin(\pi t/6))$
- P2 set: demand distribution mean value per cycle is $50(1 + \sin(\pi t/6)) + t$
- P3 set: demand distribution mean value per cycle is $50(1 + \sin(\pi t/6)) + (52 - t)$
- P4 set: demand distribution mean value per cycle is $50(1 + \sin(\pi t/6)) + \min(t, 52 - t)$

where t is the cycle number.

We have generated a random demand distribution for each given mean value in the above list. Recall that convex and *fuzzy* models studied in this chapter bound each of the values presented. On the other hand SCSP, probabilistic and MIP models can only deal with one value for each run. In this scalability test each solver seeks to find the optimal inventory levels, ordering points and order-up-to-level values for each cycle in the given

		setup cost	overage cost	purchase cost	shortage cost
no-backlogs are allowed	EOQ	[11.7, 25.3]	[10.8, 12.84]	[1025.48, 1132.26]	[187.58, 227.695]
	Probabilistic	[11.7, 25.3]	[0, 0]	[0, 1356]	[1576.35, 2050.95]
	SCSP	[2.34, 5.06]	[0.186, 0.483]	[311.36, 316.92]	[5910.28, 6371.76]
	fuzzy Dutta	[10.53, 22.77]	[9.3465, 24.633]	[1830.6, 1830.6]	[0, 0]
	fuzzy Petrović I	[11.7, 25.3]	[0, 0]	[1356, 1356]	[1576.35, 2050.95]
	fuzzy Petrović II	[22, 22]	[0, 0]	[1356, 1356]	[1576.35, 2050.95]
	MIP	[4.68, 10.12]	[41.2455, 129.283]	[2246.24, 2246.24]	[0, 0]
	UCSP	[1.25, 26.1]	[0.0465, 1375.423]	[150.12, 8520.7]	[0, 7394.8]
	cdf-intervals	[(1.25, 0), (25.6, 0.9)]	[(0.047, 0.01), (1346.47, 1)]	[(1049.67, 0), (8622.1, 1)]	[(0, 0.35), (7394.8, 1)]
	p-box *cdf*-intervals	[(1.25, 0.04, 0), (26.1, 0.36, 0.267)]	[(0.0465, 0.65, 0.23), (1172.08, 1, 0.17)]	[(988.44, 0, 0.003), (8491.232, 1, 0]	[(0, 0.65, 0.005), (7394.8, 1, 0)]
backlogs are allowed	EOQ	[12.95, 12.95]	[4.44, 3.3]	[829.99, 869.54]	[470.08, 607.94]
	Probabilistic	[11.7, 25.3]	[5.58, 19.32]	[1678.05, 1762.8]	[0, 0]
	SCSP	[2.34, 5.06]	[4.42, 15.62]	[272.44, 283.56]	[1378.88, 1451.16]
	fuzzy Dutta	[7.02, 15.18]	[12.741, 44.919]	[2502.95, 2582.05]	[723.2, 757.1]
	fuzzy Petrović I	[11.7, 25.3]	[5.58, 19.32]	[1678.05, 1762.8]	[0, 0]
	fuzzy Petrović II	[22, 22]	[12.45, 12.45]	[1678.05, 1762.8]	[0, 0]
	MIP	[4.68, 10.12]	[22.3665, 78.246]	[3319.32, 3374.92]	[1006.36, 1061.96]
	UCSP	[1.25, 26.1]	[1.26, 138.14]	[294.68, 5804.64]	[0, 1506.76]
	cdf-intervals	[(1.25, 0), (25.6, 0.9)]	[(1.26, 0.19), (125.76, 1)]	[(294.68, 0), (5932.52, 1)]	[(328.04, 0.8), (1506.76, 1)]
	p-box *cdf*-intervals	[(1.25, 0.04, 0), (26.1, 0.36, 0.27)]	[(1.26, 0.99, 0.27), (137.5, 1, 0.2)]	[(1123.66, 0, 0.004), (5799.08, 1, 0]	[(0, 0.65, 0.005), (1506.76, 1, 0)]

Table 10.10: The stud manufacturing item output costs calculated for 10 cycles

	when no-backlogs are allowed		when backlogs are allowed	
	o/p min cost	bounds on total cost	o/p min cost	bounds on total cost
EOQ	\$1048.07	[1235.56, 1398.1]	\$1013.65	[1355.865, 1455.315]
Probabilistic	\$3052.55	[2944.05, 3432.25]	\$1714.7	[1695.33, 1807.42]
SCSP	\$6325.46	[6224.17, 6694.22]	\$1672.66	[1658.08, 1755.4]
fuzzy Dutta	\$1849.08	[1850.4765, 1849.08, 1878.003]	\$3262.6	[3245.91, 3262.6, 3399.25]
fuzzy Petrović I	\$3052.55	[2944.05, 3432.25]	\$1714.7	[1695.33, 1807.42]
fuzzy Petrović II	\$3056.05	[2954.35, 3428.95]	\$1729.45	[1712.5, 1797.25]
MIP	\$2262.33	[2292.17, 2385.64]	\$4360.14	[4352.73, 4525.25]
UCSP	\$[151.42, 9922.22]		[1803.95, 6669.44]	
cdf-intervals	\$[(1050.96, 0.23), (9994.14, 1)]		[(1086.22, 0.8), (6411.92, 1)]	
p-box *cdf*-intervals	\$[(1056.3, 0, 0.1), (9884.87, 0, 0.002)]		[(993.67, 0, 0.12), (6646.55, 0, 0.002)]	

Table 10.11: The stud manufacturing item optimal quantity to order effect on the total cost observed for 10 cycles

time horizon that realize the minimum total cost. We run the tests using a Core2 Duo CPU, 2.53GHz and 3GB RAM under a windows environment.

The first three rows in Table 10.12 show the real time taken by each model in seconds to generate the output solution of the total cost given the 3 randomly generated demand sets over 7, 10 and 24 cycles. Generally real-time increases with the increase of the number of cycles in the time horizon. From the given results we can observe that the slope of the SCSP time curve is the steepest; this indicates that the SCSP solver times out at earlier stages when it is compared to *fuzzy* and convex models. Solvers that implement *fuzzy* approaches all have almost the same time performance and they are faster than probabilistic solver. For this specific data set convex models realize bounds on the output solution in less than 50% the time taken by *fuzzy* solvers; MIP solver is faster when time horizons are 7 and 10 cycles.

Two other measurements, the shared heap used and the control stack used, are taken into consideration in order to study the memory consumption of each model. The shared heap used is the memory allocated to store compiled Prolog code and its related variables and necessary buffers. The control stack used is utilized to hold backtracking information. Table 10.12 demonstrates that stochastic model memory consumption grows exponentially when scaling-up the problem, it reaches 100% for a time horizon $t = 24$. The p-box *cdf*-intervals behavior is similar to convex models. Probabilistic and *fuzzy* models have the best shared heap utilization. Clearly the percentage of the control stack utilized in the stochastic model is the highest. This is due to the behavior of stochastic techniques which exhaustively build the solution scenarios in order to reach a solution. It is worth noting that convex models and p-box *cdf*-intervals do not need to build this tree since output solution set is provided within an interval range that is encapsulating all possible output scenarios.

Furthermore, Tables 10.13 and 10.14 show the time performance comparison of the models under consideration with the 4 variations of the randomly distributed demands of our imperical evaluation. The real-time taken to output the final solution set per model is measured in seconds. It differs from one model to another and depends mainly on the set of input demand distributions and the number of cycles foreseen in the given time horizon. Results are not recorded after a time-out of 2 hours.

Clearly from Fig. 10.8 and Tables 10.13 and 10.14 the SCSP was the first to times out and the probabilistic model comes in second place followed by the MIP then *fuzzy* models. The best model in terms of scalability is the UCSP. Evidently p-box *cdf*-intervals model finds bounds on the output solution within almost a similar time range realized by the UCSP. Moreover, solution bounds obtained by the p-box *cdf*-intervals incorporate knowledge on the data whereabouts.

If we look at the overall manufacturing process, UCSP, CDF1 and PBOX outperform the rest of the models in terms of time taken to find bounds on the plan. In the three models data is collected and bounds on the probability distribution of the input coefficients are composed. Derived input coefficients are then utilized as input to the models

	time horizon *t*	stochastic	probabilistic	*fuzzy*	*cdf*	p-box	convex
real time (sec)	7	0.43	3.71	3.06	0.81	0.78	0.5
	10	70.14	6.08	5.77	3.28	3.28	3.06
	24	1683.36	175.22	159.05	55.41	32.5	31.2
control stack used	7	62.87%	46.71%	23.35%	0%	0%	0%
	10	89.82%	46.71%	23.35%	0%	0%	0%
	24	100%	46.71%	23.35%	0%	0%	0%
shared heap used	7	0.21%	0.4%	0.29%	6.87%	6.86%	6.82%
	10	0.21%	0.5%	0.29%	9.68%	9.67%	9.62%
	24	100%	0.9%	0.7%	22.93%	23.04%	22.79%

Table 10.12: Real-time taken and measurement of memory consumption

	24	26	28	30	32	34	36	38	40	42	44	46
						P1 set						
SCSP	2497.22	3127.24	3828.05	4599.65	5442.04	6355.23						
PROB	196.93	446.42	641.57	1882.5	1710.91	2207.96	6557.76					
MIP	49.23	97.68	197.53	572.62	1081.1	1594.68	3294.46	6933.5	6556.86			
PETROII	139.68	351.52	577.55	1138.5	1228.8	1479.68	1697.76	1869.98	2129.6	2328.48	5265.93	
DUTTA	124.2	239.98	590.61	1213.92	1597.44	1942.08	2284.2	2512.56	2700.0	2698.92	3252.48	
PETROI	124.04	257.8	721.28	1189.5	1873.92	1447.89	2148.12	1898.86	2096.0	2302.02	4123.69	
CDF1	82.06	265.85	600.75	1244.6	865.78	642.75	891.5	1130.0	1351.67	2289.59	2340.78	
PBOX	58.24	260.77	574.62	1187.45	675.77	586.81	874.12	1110.59	1256.86	1955.72	2119.47	
UCSP	56.33	212.72	482.03	1111.26	432.88	553.48	778.28	961.24	1088.4	1800.23	1844.06	1828.4
						P2 set						
SCSP	2525.0	3162.02	3870.62	4650.8	5502.57	6425.92						
PROB	225.88	326.56	829.57	1422.0	3242.4	5248.25						
MIP	65.68	106.02	449.73	823.76	875.81	4395.8	7272.81					
PETROII	176.4	314.34	717.36	1620.0	2088.96	2653.02	3311.28	3869.92	5136.0	6615.0		
DUTTA	148.32	180.27	672.53	945.0	1451.52	1742.67	1963.44	2317.62	2388.0	2725.38		
PETROI	155.24	243.37	682.43	1389.0	1661.44	1846.71	1967.58	5732.69	5004.0	2055.06		
CDF1	211.92	428.46	619.6	1465.45	775.08	538.88	854.55	1285.74	1922.06	2102.92		
PBOX	171.9	363.05	603.42	1376.66	669.89	520.13	813.36	1211.82	1663.99	1985.7		
UCSP	119.77	309.54	517.18	1238.79	440.33	468.82	693.04	1095.12	1371.14	1814.8		

Table 10.13: Real-time taken to solve instances where the set of demands is given as in P1 and P2

	24	26	28	30	32	34	36	38	40	42	44	46
						P3 set						
SCSP	2492.17	3120.91	3820.3	4590.34	5431.04	6342.38						
PROB	216.22	419.97	1048.32	1773.75	2444.8	4722.27	6156.0					
MIP	60.09	53.79	136.02	469.03	736.8	3873.24	4682.94	5188.97	8467.79			
PETROII	209.52	312.09	728.0	1696.5	2216.96	3034.5	3777.85	4194.83	5192.0	7003.09		
DUTTA	143.28	232.09	757.12	1003.5	1551.36	1864.05	2031.48	2263.47	2568.0	3003.21		
PETROI	178.57	263.66	989.15	1199.25	1231.36	1430.55	2998.8	5736.3	2568.0	4423.24		
CDF1	171.7	370.84	627.6	1195.14	888.15	622.29	1073.09	1372.47	1775.58	2435.39		
PBOX	157.94	368.78	625.1	1047.68	840.45	532.45	920.0	1172.04	1567.14	2147.39		
UCSP	144.15	298.46	531.96	897.83	743.92	529.05	848.64	1144.34	1548.07	2091.32		
						P4 set						
SCSP	2499.74	3130.39	3831.91	4604.29	5447.54	6361.65						
PROB	357.86	325.14	1241.24	2259.0	2672.64	4404.36						
MIP	55.19	104.72	314.78	590.79	1732.67	3547.5	6042.37					
PETROII	197.28	315.46	1024.05	1831.5	2319.36	3063.4	3531.6	4534.16	5368.0	6368.04		
DUTTA	156.96	248.99	822.64	1053.0	1697.28	2020.11	2400.84	2469.24	2712.0	3082.59	4414.08	
PETROI	200.74	292.49	861.22	1531.5	1367.04	1684.87	1640.16	2714.72	2124.0	3245.76	5382.08	
CDF1	251.81	377.27	614.4	1357.18	800.11	605.21	922.54	1127.69	1379.99	1990.82	2051.26	
PBOX	209.19	376.52	595.77	1156.04	664.42	601.99	813.17	1010.49	1186.99	1698.63	1684.34	
UCSP	127.79	307.38	520.14	1155.23	442.47	519.8	697.55	968.99	1177.76	1570.67	1449.6	1668.72

Table 10.14: Real-time taken to solve instances where the set of demands is given as in P3 and P4

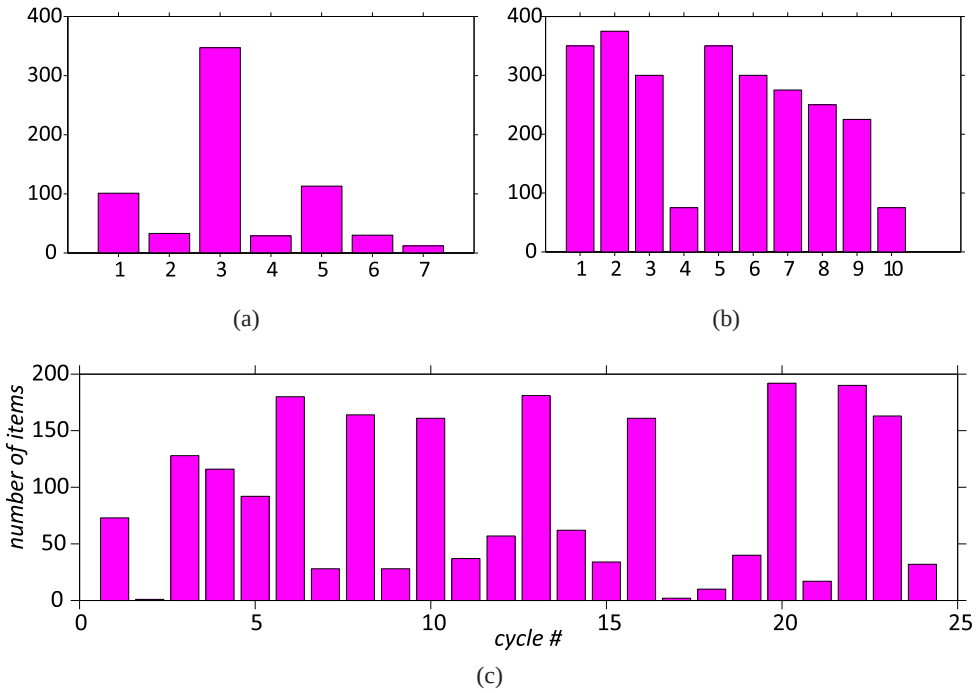


Fig. 10.7: Input demand distributions with mean sets over (a) 7, (b) 10 and (c) 24 cycles

and solution is deduced in one stage; i.e. all foreseen cycles at once. Output solution is an envelopment of an unknown distribution.

10.9 Summary

In this chapter, we discussed the inventory management problem and its application in the manufacturing process. We described how different models formulate the inventory problem and what is the useful solution output. Generally, input to the model is: customer demands, setup cost, variable item cost and holding costs. Noticeably, input to the problem has a stochastic nature and variables with fluctuating values are usually approximated in order to simplify the complexity of the algorithm that seeks to find the optimal solution. Output from the models is basically the schedule and size of orders, inventory levels and the total cost. Obtained output helps decision makers to plan ahead for the inventory and order capacities as well as the budget of the manufacturing process. We elaborated the evolution of the model formulation in the literature starting from the very basic deterministic EOQ model. We exerted a comparison between SCSP, probabilitic, *fuzzy*, MIP, UCSP, *cdf*-intervals and p-box *cdf*-intervals models. SCSP and probabilitic models, in practice, proved to be the slowest models. While studied *fuzzy* models have almost the same time performance. Petro II incorporates *fuzzy* cost variable in the model hence it allows for reasoning while considering *fuzzy* encapsulation of the given data.

We practically proved that convex models UCSP, *cdf*-intervals and p-box *cdf*-intervals

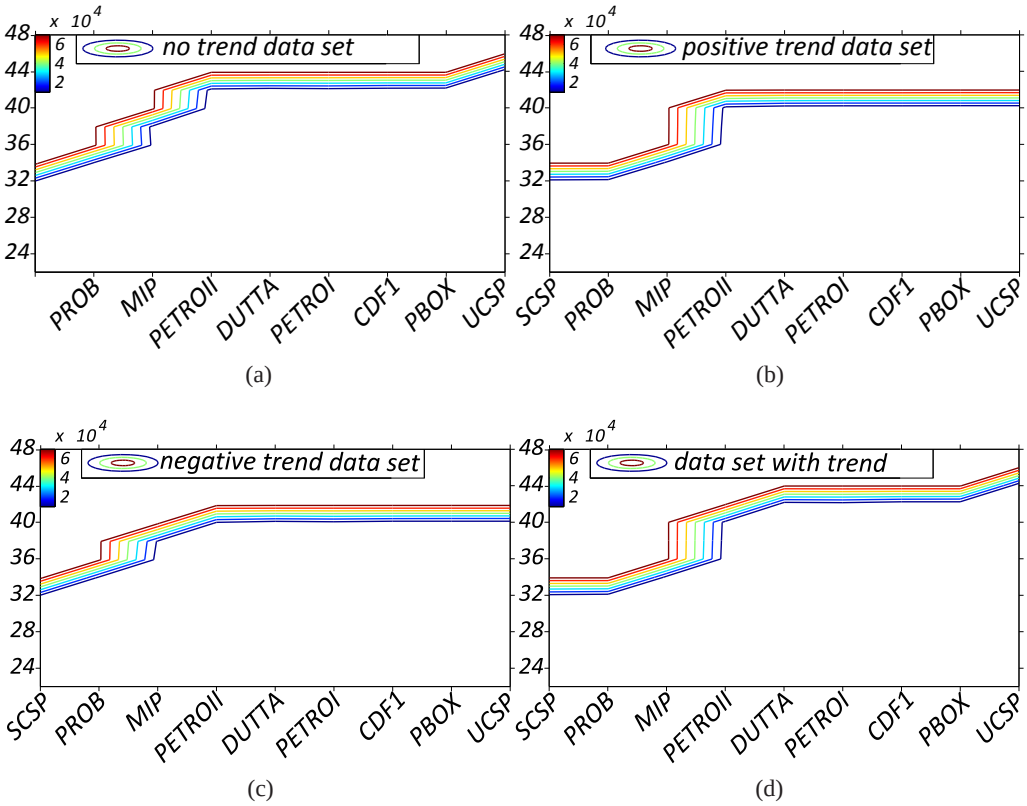


Fig. 10.8: Model scalability comparison for (a) P1 set of demand distributions with no trend over the time horizon,(b) P2 set of demand distributions with positive trend over the time horizon,(c) P3 set of demand distributions with negative trend and (d) P4 set of demand distributions with trend

encapsulate each demand distribution in a convex representation in order to incorporate the problem inputs fluctuating nature. Clearly results show that convex models are very fast and outperform the rest of the models. Our approach suggests a range of quantities to order along with an idea of data whereabouts. This information will provide all options to the decision maker; hence they will be given a range of quantities to order along with the range of costs for each decision.

PART IV

DISCUSSION

CHAPTER 11

CONCLUSION

In this dissertation, we propose a novel framework and a formal constraint logic programming language over *cdf*-intervals to reason about data with uncertainty. The key idea is to extend convex models with probabilistic information in order to realize aditional quantifiable knowledge on the data whereabouts. Our aim is to intuitively describe data coupled with uncertainty or following unknown distributions without losing any knowledge given in the problem definition.

We introduced in Saad et al. (2010) the *cdf*-intervals structure which was our first attempt to represent data in a 2D domain representation: real and *cdf*. Despite the fact that the *cdf*-intervals structure represents and reasons about data alongside its whereabouts in the 2D, it is an approximation of the probability distribution, hence it lacks the full encapsulation of the actual probability distribution of the data. We extend the *cdf*-intervals with a p-box structure to obtain a safe enclosure in which data along with its whereabouts are enveloped by two probability distributions [Saad et al. (2012b),Saad et al. (2012a),Saad et al. (2014) and Saad (2014)]. Bounding *cdf*-distributions are chosen to be uniform because they have linear computations. In this chapter we summarize the thesis list of contributions.

11.1 Related work

In Chapter 3, we provide the technical background that supports the scheme and proofs the correctness of the theory behind our framework. We recall all probabilistic concepts which establish and explain the *cdf* and the p-box, the main building blocks of our algebraic structure. We also demonstrate how probabilistic operations are exerted in the probability paradigm. These operations are the core maneuver in the reasoning process when it is concerned with probabilistic distributions.

We reviewed how data uncertainty is usually tackled and dealt with in different paradigms. The literature categorizes the models to follow either probabilistic or pos-

sibilistic approaches. We studied the two approaches and their diverse implementations in the CP and LP paradigms. We also provide their advantages and drawbacks when they are implemented in real-life problems. There is so much to do in the literature concerning reasoning about real-life situations coupled with uncertainty. Representations and solving techniques are often catered to specific situations and cannot be generalized. There is no one existing integrated paradigm that enables researchers to represent and deal with all types of uncertainty.

We are interested in systems of constraints in CP and OR because they are easily embedded in declarative programming languages. Such systems are heavily used in the problem solving environments, while separating logic from control. The CP paradigm is characterized to be flexible in expressing optimization problems. On the other hand, the LP paradigm provides a better realization of optimal solutions when problems are scalable. Systems of constraints in CP and OR define a set of constraints through an intuitive descriptive algebraic structure along with the domains allowed for variables in a constraint relation. The aim is to find an admissible solution or set of solutions to the problem under consideration. This is achieved by an appropriate definition of a query-answering mechanism following AI and/or OR techniques.

11.2 Uncertain Data Representation

Alternative approaches found in CP and OR, generally, follow diverse approximation techniques to represent data uncertainty/population distribution observed in the problem definition. Some approaches deal with uncertainty by associating a point-wise probability to values in the discrete domain. Some others seek to approximate the observations to the nearest known probability distribution using statistical tools such as the expected value and variance. Approaches following convex techniques enclose all possible data and treat them with an equally weighted knowledge (probability of occurrence). This is to ensure the tractability of the model by performing the reasoning on the interval bounding values (extreme points).

In Chapter 5, we show how observed data in the problem definition is collected and represented by the algebraic structure defined in our model. Data whereabouts often forms an unknown random distribution. Since this distribution is unknown we enclose it between two uniform distributions. Bounding distributions along with the minimum and maximum quantile values construct a p-box. We show that by issuing lines with a maximum (minimum) slopes from the minimum (maximum) measured quantile values, we guarantee the full encapsulation of the probability distribution being measured and following an unknown distribution. We then compare between the different representations of input data to show how accurate its whereabouts are enveloped in the models (probabilistic, *fuzzy*, *cdf*-intervals and p-box *cdf*-intervals). The p-box *cdf*-intervals representation out of this comparison is the most safe enclosure which guarantees the full encapsulation of available data along with its whereabouts. A point lying in the p-box *cdf*-interval bounds has a chance of occurrence lying between the uniform distri-

butions bounding the p-box. Accordingly the p-box representation allows for an additional quantitative information about bounds on the possible chances of occurrence of quantile values. Table 11.1 lists the main differences between the convex representation in real-intervals, *cdf*-intervals and p-box *cdf*-intervals.

11.3 New domain definition

After the data collection process we form the p-box *cdf*-interval which is as safe enclosure that identifies the confidence interval in a two-dimensional manner: values and bounds on their chance of occurrence. We then propose a new domain for reasoning with uncertain data. The key idea is to combine the usual interval arithmetic approach with a second dimension capturing the cumulative distribution function *cdf* of the variable whose primary dimension (the real domain $\mathbb{R}$) spans the value that the uncertain variable can take. We define the *cdf*-intervals domain and how the interval arithmetic operations are exerted over this new domain. The entire exercise makes it possible to define constraints over variables defined on this domain. Solution method of these constraints deliver intervals for the variables with the *cdf* in our first approach, and alongside an enveloped *cdf* in the p-box *cdf*-interval approach.

As opposed to points defined over the lattice of reals $\mathbb{R}$, a point in the *cdf*-interval is defined by a line (the uniform *cdf*-distribution) in a 2D-space manner. Our first *cdf*-intervals structure aligns all points lying between the interval bounds into one line that approximates the whereabouts to the nearest *cdf*-uniform distribution. This representation is revisited such that each point is associated with one *cdf*-uniform distribution. In a 2D-space, a line is generally constructed following one of the two approaches: four values (two values on the x axis and their corresponding values on the y axis) or three values (an x value, a y value and a slope). The first approach defines the *cdf*-interval domain proposed to define our first algebraic structure. The second approach defines triplet points bounding the p-box *cdf*-interval. Each triplet point corresponds to: the quantile value, the *cdf* and the slope of the uniform distribution which shows how data is scattered along the quantiles (on the x axis). Storing a triplet for each bounding uniform distribution is sufficient to construct the p-box *cdf*-interval, hence the minimum and the maximum quantiles alongside bounds on the chance of occurrence any point lying between the interval bounds can happen are all stored by the bounding triplet representation.

Points of the new defined domains are consequently ranked in a two-dimensional manner. Distributions lying towards the upper bound are dominating those preceding them. This fact helps us defining our computation domain in the 2D-space wherein the *cdf* is associated to the uncertainty value of a point. The core operations computed over p-box *cdf*-intervals extends real interval arithmetic with a probabilistic computation over the bounding *cdf*-uniform distributions which is in turn a linear computation hence adding a minimal overhead while performing the calculations over the interval bounds.

	Reliable intervals	*cdf*-intervals	p-box *cdf*-intervals
Initial data representation	Bounds on the observed data are recorded but its whereabouts is not	It is one point lying within the p-box *cdf*-interval	encapsulates all data presented/measured and provided in the problem under consideration
	All values provided within the interval bounds are equally weighted	Assumes that data is uniformly distributed across the interval. If this is the case then it is an exact presentation of the data probability and values	*cdf*-bounds restrict the possibility of data to occur outside the interval limits
	The interval encapsulates and deals with only one domain: the domain of reals $\mathbb{R}$	When the data is not uniformly distributed, estimated *cdf*-uniform distribution is an approximation of the data realization	The interval is a full encapsulation of data and its whereabouts
	Lattice property on one dimension: the real domain $\mathbb{R}$	Lattice property for each dimension. The framework makes use of the monotonic property of the *cdf*. The steepness of the distribution slope roughly indicates the whereabouts	Lattice property for each dimension: quantile and probability
	Maintains an interval-based approach in the real domain $\mathbb{R}$	Maintains an interval-based approach in the quantifiable dimension (probability) along with the real dimension	Maintains an interval-based approach in the quantifiable dimension (probability) along with the real dimension
Point representation	A point is a singleton value in the real domain $\mathbb{R}$	A point is a tuple. The monotonic property of the *cdf* seeks to store only two values per bound: quantile and *cdf*. Thus we do not store the full probability information	A point is a triplet. This triplet indicates a $2D$ line which is the *cdf*-uniform distribution
Reasoning	Output solution set is an interval of quantile bounds (like in UCSP)	Builds 2 sets of constraints that enable us to reason in a 2D manner. Hence, output solution set is not only interval quantile bounds but also it shows an approximated uniform distribution of the data whereabouts	Builds the set of constraints on the real domain $\mathbb{R}$ then project obtained real solution onto the *cdf*-domain

Table 11.1: Convex structures

Resulting domains from these operations are proved to have a convex algebraic structure which encloses the unknown probability distribution.

11.4 The *cdf*-intervals Language and Constraint Solver

We define the new domain over a lattice structure. As a poset, a lattice defines for every two elements a partial ordering, a *glb* and *lub*. Those definitions outline the fundamental features of a formal language description. The new ordering calculus and arithmetic operations are exerted on the computational domain: the domain of discourse that maps a variable to a measureable quantile together with its knowledge in the 2D-space. Operations are performed on the bounding uniform distributions. They are linear because uniform distributions each is shaping a line.

We develop a formal CLP language from the new defined domains and show how this new domain affects the problem variables and the decision process. The *cdf*-intervals CLP language intuitively formalizes the set of rules and fix-point semantics to reason about data coupled with uncertainty. The reasoning scheme follows a local consistency approach which attempts at constraining each variable over the *cdf*-interval domain while excluding infeasible solutions.

We implement the new language as a separate solver module in the ECL^{i}PS^e constraint programming environment. We use the ECL^{i}PS^e environment due to its data-driven programming facility which intuitively realizes meta-programming. This is accomplished by using the attributed variables to specify the *cdf*-interval data structures, and the suspension library to control the triggering mechanism for executing delayed clauses. We provide in our implementation an extension to the Prolog unification, constraint ordering relations and arithmetic operations over the *cdf*-intervals variable domains.

Empirical evaluation shows that solutions from constraint systems over p-box *cdf*-intervals domains intersect with those output from the real intervals, especially when real bounds are the same. Moreover, the violation of the *cdf* ordering property may shrink the interval domain. Hence the realized solution space can further be pruned. This is an added value the p-box *cdf*-intervals introduce: solutions sought to be feasible in the real domain are excluded since they are infeasible because they violate the properties belonging to the *cdf*-domain.

11.5 Definition of new Global Constraints

We define global constraints over systems of linear equations described by the p-box *cdf*-intervals algebraic structure. The new constructed global constraints extend interval linear systems with a second dimension (the probability). In the linear systems of equations, p-box *cdf*-intervals are introduced as variable coefficients. They can be solved by simple polynomial transformation into a linear model which is then sent to the Simplex method. We run 2 Simplex to compute interval bounds. This approach is often used to

solve Interval Linear Systems and it is very cost effective. Interval quantiles are then projected onto the *cdf* domain in order to obtain bounds on the *cdf*-distributions. The p-box *cdf*-intervals closure realizes quantile values along with bounds on their whereabouts. We show that extreme triplet points resulting from this operation is equivalent to those obtained from the CSP version of the solver.

11.6 Practical Evaluation

We apply the novel language to model two different real-life applications: NTAP used in network design problems and the inventory management problem.

11.6.1 NTAP Problem

The NTAP problem is used in traffic monitoring and network diagnostic which serve in capacity planning, traffic engineering, reliability analysis, network management and other network administrative tasks. The observed traffic volumes in the problem definition have a fluctuating nature and most of the techniques adopted to serve this problem are either reliable (they seek bounds that encapsulate all possible realizations) or robust (they approximate the problem to the nearest possible deterministic case). NTAP is a large scale optimization problem which is usually under constrained (the number of links is significantly small compared to the number of traffic demands) and has a solution set characterized to be infinitesimal. Most of the existing approaches serving this consortium of applications lack the full encapsulation of the actual distribution of data that is provided in the problem definition.

We describe how to intuitively model the NTAP problem in the p-box *cdf*-intervals and compare this representation with the deterministic, certainty closure and the approach of the *cdf*-intervals with one approximated uniform distribution. We showcase our concept with hypothetical data and network topologies collected from the sndlib data corpus and a sample network instance of 4 nodes. Intervals derived in three models encapsulate the actual measured value. The UCSP provide a solution set with equal knowledge weights. The *cdf*-intervals approach yields an interval with one approximated uniform distribution. The p-box *cdf*-intervals realizations incorporate additional knowledge on the data whereabouts and might further prune the resulting domains in order to comply with the probability ordering properties. Hence the p-box *cdf*-intervals approach can exclude solutions which might exist in convex model yet they are impossible to happen.

11.6.2 Inventory Management Problem

The class of inventory management problem is widely used in real-life situations which can be listed but not limited to: the daily ordering of newspapers, the booking of airline flights and the ordering of items in a supply chain of a manufacturing process. In this type of problems decision makers need to schedule ahead the ordering of items which

fulfill uncertain consumer demands, associated by fluctuating market prices, with a minimal possible cost. We provide the evolution of the existing models which help solving this consortium of applications. The main modeling aspects which are input to the model and which define the structure of policies involved in monitoring the inventory levels are: the setup time that adds extra cost in the manufacturing process, the replenishment policies which vary between periodical and continuous monitoring of the inventory levels, and the customer demands characterized to be unpredictable yet they need to be fulfilled and satisfied in a timely manner. Modeling the inventory problem yields output factors which help the decision makers to take further steps like: defining the reorder point and determining the quantity to order. Both actions lead to a set of costs (holding cost, setup cost, ordering cost and shortage cost) which affect the overall cost of the manufacturing/ production process.

We studied the different modeling approaches which tackle the inventory management problem: the deterministic version (EOQ model), dynamic programming, constraint programming and its stochastic version, uncertainty closure framework, mixed integer programming, and *fuzzy* programming. We show how the p-box *cdf*-intervals intuitively envelop the uncertain data found in different modeling aspects with minimum overhead. To evaluate our model and its solution we compare the p-box *cdf*-intervals inventory management model version with all of the above mentioned approaches by means of a process with a finite time horizon. In practice and based on our findings, stochastic constraint programming and probabilistic models are the slowest. Fuzzy models with variable representation proved to have the same time performance and their output solutions are characterized to be reliable, i.e. they seek the satisfaction of all possible realizations. Convex models of the certainty closure, the *cdf*-intervals and the p-box *cdf*-intervals encapsulate all possible distributions of the demands in a convex representation. They seek bounds on the solution sets. This fact allows for building inventories with appropriate capacities and budget. Moreover, the p-box *cdf*-intervals framework provides a range of quantities to order and a range of costs for each decision along with bounds on its whereabouts.

To the best of our knowledge, p-boxes have never been implemented in the CP paradigm, yet they are very good candidates to deal with and reason about uncertainty in the probabilistic paradigm, especially when the data is shaping an unknown distribution. The concept of p-boxes relies on the probabilistic approach that ranks probability distributions based on their stochastic dominance. It is a safe envelopment of the data whereabouts especially when it follows an unknown distribution. The *cdf* was selected due to its aggregated nature which enables the propagation of the information to the interval bounds. In addition to its capability of easily ranking probability distributions within a p-box domain. We have implemented the p-box *cdf*-intervals in the ECL^{i}PS^e constraint programming environment, and we have shown that, with minimal overhead, obtained solutions gained tighter bounds in the probabilistic domain when compared with convex models.

CHAPTER 12

FUTURE WORK

The introduction of a novel framework to reason about data coupled with uncertainty due to ignorance or based on variability, paves the way to many fruitful research directions. We can list many in: studying models having variables following dependent probability distributions, further investigating the model structure to deal with disjoint domains, suggesting a list of global constraints, exploring different search techniques, generalizing the framework to deal with all types of uncertainty, revisiting the framework within a dynamically changing environment, and last but not least applying the model to a variety of large scale optimization problems which target real-life engineering and management applications.

12.1 Dependencies of variables

When we construct our model, we assume that all variables have independent probability distributions. This is to ease the computation and the reasoning about the distributions. For future work we need to account for data dependencies if exist in the problem representation. In the NTAP, we can study, for instance, the network traffic flows which split into more than one link. The probability distribution that shapes each flow variable traversing a link will depend on another link-flow variable.

12.2 Disjoint domains

The p-box *cdf*-intervals is a convex structure which sets bounds on the data and its whereabouts. We did not tackle examples where we have an uncertain variable with disjoint domains. We expect that disjoint domains will add up more probabilistic computation on the *cdf* bounding distributions because more triplet points will be involved. The question for future work is how to balance between representing the full set of domains in a convex structure as opposed to dealing with them as multi-intervals in disjunction.

This should be coupled with a full study on the price with respect to the complexity of the computation of disjoint domains since in this case the number of enumerated domains is larger.

12.3 Global constraints

Global constraints are pre-filtering algorithms which capture problem specific structures in order to reduce the search space by disregarding infeasible realizations. They utilize different paradigms which view the problem with a global perspective such as in: the graph theory constructs the problem visually using a graph (directed/undirected) defined over a set of vertices and nodes; and the linear programming paradigm incorporates continuous variables in a set of linear equalities and inequalities, along with an objective function that needs to be optimized [W. van Hoeve and Katriel (2006)]. Examples of global constraints can be found in: the sum and knapsack constraints associates a scalar to the sum of the set of variables; the global cardinality constraint assigns every value to a variable [Régin (1996), Régin (1999)]; the element constraint states that a value is equal to the i^{th} variable in a constraint [Van Hentenryck and Carillon (1988)]; the Alldifferent constraint is based on the matching theory for efficient filtering [Régin, Petit, Bessière, and Puget (2000)]; and the Cumulative constraint allocates and schedules resources in a cumulative manner [Baptiste, Laborie, Le Pape, and Nuijten (2006)]. The p-box *cdf*-intervals framework is aligned with the concepts of global constraints. Using the propagation techniques following the p-box *cdf* algebraic structure, we can eliminate values from the domains that do not lead to an admissible solution before the search takes place. The probability dominance property is an essential tool, in this case, it introduces probabilistic constraints which eliminate unnecessary values which contradict the satisfaction of the problem set of constraints. We have adopted techniques from the linear programming paradigm to represent bounding *cdf* distributions of the variable constraint coefficients. This technique provides an efficient propagation technique that prunes the domain of a variable to a convex representation that excludes all unnecessary values from the p-box *cdf*-interval. Further investigations need to take place to study the integration of the p-box *cdf*-intervals within a global graphical representation of a given problem when it is subject to uncertainty.

12.4 Search Techniques

The p-box *cdf*-interval algebraic structure constrains variable domains in a two dimensional manner. The probabilistic dimension adds an extra constraint in the constraint network in order to maintain the probabilistic dominance property. The novel structure can be plugged into existing search techniques for problem instances existing in the uncertain and stochastic world. The challenge here is to improve the search methods by adding information bounding the probability distribution of a given event.

Techniques for solving CSP can be found in: backtracking search and local search. Solving algorithms can be either complete or incomplete. The backtracking search is a complete algorithm (eventually it leads to a solution) whereas the local search is incomplete (might not find the optimal solution of the problem). Backtracking often requires an exponential amount of space and time. Research to improve the backtracking algorithms is quite large and usually it is accompanied by heuristics which often order the search tree with the most promising branch to be visited first. A typical backtracking algorithm is a set of branching constraints that build a tree. The variations of such branches are formed by enumeration, binary choice point or domain splitting [van Beek (2006)]. An example of a future work can be combining the p-box *cdf*-intervals with value ordering heuristics [Ginsberg (1993), Vernooy and Havens (1999)]. This algorithm seeks to find the best next value for a given variable that is more likely to be part of the solution. The choice of this value can be supported by the stored probabilistic value within the *cdf*-interval structure. Another possible integration can be associated with the 'randomization and restart' search strategies [Harvey (1995), Gomes, Selman, Kautz, et al. (1998)]. The p-box *cdf*-interval structure can assist in finding the best point in time wherein the restart should take place. This can be done, for instance, when the search with the variable domain has a very low probability of satisfaction.

A detailed survey about local search techniques adopted in CSPs can be found in Hoos and Tsang (2006). Local search by means of meta-heuristics is one of the most powerful search techniques for solving large problem instances in many practical applications. The most prominent local search techniques are: stochastic local search, simulated annealing, tabu search and dynamic local search. To find an optimal solution, the algorithm iteratively utilizes a randomization in order to improve the realization when the solution reached is infeasible, sub-optimal or incomplete. The key idea is to select the new candidate solution in a stochastic manner. The probabilistic property stored in the p-box *cdf*-interval algebraic structure can assist in this selection: the most promising choice that is more likely leads to a solution.

12.5 Modeling Uncertainty

Uncertainty, in the literature, as shown in Chapter 3 can be due to ignorance or stochastic nature of the problem. To represent uncertainties, the model selection process is overwhelmed with an infinite number of proposed models each targeting a specific uncertainty type. There is an urgent need to deal and reason about all types of uncertainty and variability in a single framework. Uncertainty is assessed either by enumeration (requiring expensive calculations) or bounds (acquiring less expensive computation). The challenge is to construct a system which identifies the uncertainty type, builds an adequate problem structure and finds the most effective and efficient resolution form. Ideally, all proposed frameworks should be brought together this is by integrating the *cdf*-intervals with semiring CSPs [Bistarelli et al. (1999)], valued CSPs [Schiex et al. (1995)], mixed CSPs [Fargier et al. (1996)], SCSPs [Walsh (2000)], dynamic CSPs [Falt-

ings and Macho-Gonzalez (2005)], and *fuzzy* CSPs [Dubois et al. (1996)]. As a result we can introduce an intuitive expressiveness of the uncertainty associated with the problem definition. Furthermore, we should seek the integration of the hybridization techniques which bring the LP power into the CP for better optimization and solution findings.

12.6 Dynamically changing environment

In this dissertation, we studied the *cdf*-intervals algebraic structure to model uncertain environment in a single time snapshot. However, uncertainty can also be found in a changing environment. As pointed out in Chapter 3, solving problems which change over time has two main strategies: minimize the need for change by seeking a reliable solution; minimize the cost of change by acquiring a stable solution; and minimize the reaction time by questing for quick solutions. Since the *cdf*-intervals are convex structures they seek a robust solution which envelop all possible realizations of the uncertain problem, hence they can assist in the modeling and solving of problems using the first listed strategy.

A possible future line of research is to study the integration of the *cdf*-intervals with existing local repair methods targeting applications with unknown future. For instance, when the min-conflict heuristic is used to minimize the number of unsatisfied constraints [Minton et al. (1992)] or to solve over-constrained problems [Barták et al. (2004)], the additional knowledge, which is provided by the *cdf*-intervals about the whereabouts, can be propagated to guide the heuristics towards minimizing the possible chance of occurrence of unsatisfied constraints in the problem.

When the type of change is unknown, the *cdf*-intervals can be a good candidate in the integration with the oracle approach [Van Hentenryck and Le Provost (1991)]. This is due to the ability of p-boxes to retain a convex structure that stores previous observations. In this case prior states in the sequence are retained while excluding all unnecessary sub-trees from the search space. Solutions to new problems can maintain the same path.

A third type of dynamic environment exists when information about the change is uncertain. This type of problems often use the recurrent CSP approach to record the source and frequency of the change. The p-box *cdf*-intervals are characterized to be proactive since they store all observed information (data and its whereabouts) in a convex structure. Techniques to integrate recurrent CSP Wallace and Freuder (1998) with the p-box *cdf*-intervals should take place in order to enrich the framework with a tractability property. This integration can be generalized to incorporate the SCSPs [Walsh (2000)] and the Branching CSP [Fowler and Brown (2003)] for the same reason.

In a distributed environment, uncertainty can be found in the variable domains since they are revealed over time in a distributed manner. The p-box *cdf*-intervals structure can define bounds on such domains. Accordingly, they can be intuitively integrated with Open CSPs [Faltings and Macho-Gonzalez (2005)]. The p-box *cdf*-intervals restrict the search space to the possible realizations. When they are used interactively they

exclude all infeasible solutions from the search space hence we expect a less expensive computation.

12.7 Applications

Tackling real-world applications, coupled with data uncertainty, is a broad and diverse field of research. As pointed out in Chapter 3, they are the heart of many Engineering and Management problems such as in: planning, scheduling, diagnostic and tracking. These problems can be found in a diverse list of applications such as in risk assessment, financial markets trading and product design reliability analysis. The given uncertainty varies in its definition and representation between ill-defined and fluctuating. Even though we developed a model for the NTAP and the inventory management problems, there is a lot to be done in this area and the applications tackled in this thesis in particular. For instance, we need to study the model variations when distributed information about the link-flow in the NTAP is observed and for several adopted measurement techniques. Another example in the inventory management problem the model can differ based on the implemented replenishment policy of inventory levels.

The key idea is to identify the source of uncertainty and seek bounds on the given knowledge (data and its whereabouts). Then the representation of the uncertainty using the convex structure should envelop all the observed information. Third, the problem is modeled in terms of variables, domains and constraints. This will provide the p-box *cdf*-intervals problem algebraic structure which follow the bound consistency propagation techniques in order to prune the search space. The output from this operation is a solution set per variable having a p-box *cdf*-interval convex structure. The p-box *cdf*-intervals framework can be utilized to model and solve an endless list of future optimization problems coupled with uncertainty.

REFERENCES

Aberth, O. (1997). The solution of linear interval equations by a linear programming method. *Linear algebra and its applications*, *259*, 271–279.

Airoldi, E., & Faloutsos, C. (2003). *Advances in Network Tomography* (Tech. Rep.). Citeseer.

Apt, K. R., & Wallace, M. (2006). *Constraint logic programming using ecl^{i}ps^e*. Cambridge University Press.

Baptiste, P., Laborie, P., Le Pape, C., & Nuijten, W. (2006). Constraint-based scheduling and planning. In F. Rossi, P. Van Beek, & T. Walsh (Eds.), (chap. 22). Handbook of Constraint Programming.

Barták, R., Müller, T., & Rudová, H. (2004). A new approach to modeling and solving minimal perturbation problems. In *Recent advances in constraints* (pp. 233–249). Springer.

Beaumont, O. (1998). Solving interval linear systems with linear programming techniques. *Linear Algebra and its Applications*, *281*(1-3), 293–309.

Ben-Haim, Y., & Elishakoff, I. (1995). Discussion on: A non-probabilistic concept of reliability. *Structural Safety*, *17*(3), 195–199.

Benhamou, F., & Older, W. (1997). Applying interval arithmetic to real, integer, and boolean constraints. *The Journal of Logic Programming*, *32*(1), 1–24.

Ben-Tal, A., & Nemirovski, A. (1999). Robust solutions of uncertain linear programs. *Operations Research Letters*, *25*(1), 1–14.

Ben-Tal, A., & Nemirovski, A. (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming*, *88*(3), 411–424.

Berleant, D. (1993). Automatically verified reasoning with both intervals and probability density functions. *Interval Computations*, *2*, 48–70.

Bertsimas, D., Pachamanova, D., & Sim, M. (2004). Robust linear optimization under general norms. *Operations Research Letters*, *32*(6), 510–516.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations research*, *52*(1), 35–53.

Bertsimas, D., & Thiele, A. (2006). A robust optimization approach to inventory theory. *Operations Research*, *54*(1), 150-168.

Birge, J. R., & Louveaux, F. V. (1988). A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, *34*(3), 384–392.

Bistarelli, S., Montanari, U., Rossi, F., Schiex, T., Verfaillie, G., & Fargier, H. (1999). Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison. *Constraints*, *4*(3), 199–240.

Bookbinder, J., & Tan, J. (1988). Strategies for the probabilistic lot-sizing problem with service level constraints. *Management Science*, *34*, 1096-1108.

Case, J., Fedor, M., Schoffstall, M., & Davin, J. (1990). Rfc 1157: Simple network management protocol (snmp).

Charnes, A., Cooper, W. W., & Symonds, G. H. (1958). Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. *Management Science*, *4*(3), 235–263.

Chinneck, J., & Ramadan, K. (2000). Linear programming with interval coefficients. *Journal of the Operational Research Society*, 209–220.

Chvátal, V. (1983). *Linear programming. a series of books in the mathematical sciences.* WH Freeman and Company, New York.

Claise, B., Sadasivan, G., Valluri, V., & Djernaes, M. (2004). *Cisco systems netflow services export version 9* (Tech. Rep.). RFC 3954, October. Retrieved from http://www.ietf.org/rfc/rfc3954.txt

Clark, A., & Scarf, H. (1960). Optimal policies for a multi-echelon inventory problem. *Management Science*, *6*(4), 475–490.

Cleary, J. (1987). Logical arithmetic. *Future Computing Systems*, *2*(2), 125–149.

Dantzig, G. B. (1955). Linear programming under uncertainty. *Management science*, *1*(3-4), 197–206.

Deruyver, A., & Hodé, Y. (2009). Qualitative spatial relationships for image interpretation by using a conceptual graph. *Image and Vision Computing*, *27*(7), 876–886.

Dubois, D., Fargier, H., & Prade, H. (1996). Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty. *Applied Intelligence*, *6*(4), 287–309.

Dutta, P., Chakraborty, D., & Roy, A. (2005). A single-period inventory model with fuzzy random variable demand. *Mathematical and Computer Modelling*, *41*(8), 915–922.

ECRC. (1994). *Eclipse (a) user manual, (b) extensions of the user manual* (Tech. Rep.). Author. http://www.eclipseclp.org/.

Edgeworth, F. (1888). The mathematical theory of banking. *Journal of the Royal Statistical Society*, *51*(1), 113–127.

Eldar, Y. C., Ben-Tal, A., & Nemirovski, A. (2004). Linear minimax regret estimation of deterministic parameters with bounded data uncertainties. *Signal Processing, IEEE Transactions on*, *52*(8), 2177–2188.

Elmaghraby, S. (1978). The economic lot scheduling problem (elsp): review and extensions. *Management Science*, *24*(6), 587–598.

El Sakkout, H., & Wallace, M. (2000). Probe backtrack search for minimal perturbation in dynamic scheduling. *Constraints*, *5*(4), 359–388.

Faltings, B. (2006). Uncertainty and change. In F. Rossi, P. Van Beek, & T. Walsh (Eds.), (chap. 21). Handbook of Constraint Programming.

Faltings, B., & Macho-Gonzalez, S. (2005). Open constraint programming. *Artificial Intelligence*, *161*(1), 181–208.

Fargier, H., Lang, J., & Schiex, T. (1996). Mixed constraint satisfaction: A framework for decision problems under incomplete knowledge. In *Proceedings of the national conference on artificial intelligence* (pp. 175–180).

Ferrero, A., & Salicone, S. (2004). The random-fuzzy variables: A new approach to the expression of uncertainty in measurement. *Instrumentation and Measurement, IEEE Transactions on*, *53*(5), 1370–1377.

Ferson, S., & Ginzburg, L. R. (1996). Different methods are needed to propagate ignorance and variability. *Reliability Engineering & System Safety*, *54*(2), 133–144.

Ferson, S., Kreinovich, V., Ginzburg, L., Myers, D., & Sentz, K. (2003). *Constructing Probability Boxes and Dempster-Shafer structures, Sandia National Laboratories* (Tech. Rep.). Technical report SANDD2002-4015.

Fowler, D., & Brown, K. (2003). Branching constraint satisfaction problems and Markov decision problems compared. *Annals of Operations Research*, *118*(1), 85–100.

Gervet, C., & Atef, M. (2013). Optimal allocation of renewable energy parks: A two–stage optimization model. *RAIRO-Operations Research*, *47*(02), 125–150.

Ginsberg, M. L. (1993). Dynamic backtracking. *arXiv preprint cs/9308101*.

Glen, A., Leemis, L., & Drew, J. (2004). Computing the distribution of the product of two continuous random variables. *Computational statistics & data analysis*, *44*(3), 451–464.

Gomes, C. P., Selman, B., Kautz, H., et al. (1998). Boosting combinatorial search through randomization. *AAAI/IAAI*, *98*, 431–437.

Grossglauser, M., & Rexford, J. (2005). Passive Traffic Measurement for Internet Protocol Operations. *The Internet as a Large-Scale Complex System*, 91.

Gubner, J. (2006). *Probability and Random processes for electrical and computer Engineers*. Cambridge Univ Pr.

Gum, I. (1995). Guide to the expression of uncertainty in measurement. *BIPM, IEC, IFCC, ISO, IUPAP, IUPAC, OIML*.

Halpern, J. Y. (2003). Reasoning about uncertainty.

Hammersley, J. M., Handscomb, D. C., & Weiss, G. (1965). Monte carlo methods. *Physics today*, *18*, 55.

Hansen, E. (1979). Global optimization using interval analysis: the one-dimensional case. *Journal of Optimization Theory and Applications*, *29*(3), 331–344.

Hansen, E. (1980). Global optimization using interval analysisthe multi-dimensional case. *Numerische Mathematik*, *34*(3), 247–270.

Hansen, E. (1992). Bounding the solution of interval linear equations. *SIAM journal on numerical analysis*, *29*(5), 1493–1503.

Harris, F. (1913). How many parts to make at once. *Operations Research*, 947–950.

Harvey, W. D. (1995). *Nonsystematic backtracking search*. Unpublished doctoral dissertation, Citeseer.

Hnich, B., Rossi, R., Tarim, S. A., & Prestwich, S. (2011). A survey on cp-ai-or hybrids for decision making under uncertainty. In *Hybrid optimization* (pp. 227–270). Springer.

Hoffman, K. (2000). Combinatorial Optimization: Current successes and directions for the future. *Journal of computational and applied mathematics*, *124*(1-2), 341–360.

Holzbaur, C. (1992). Metastructures vs. attributed variables in the context of extensible unification - applied for the implementation of clp languages. In *In 1992 international symposium on programming language implementation and logic programming* (pp. 260–268). Springer Verlag.

Hooker, J. N. (2006). Operations research methods in constraint programming. *Foundations of Artificial Intelligence*, *2*, 527–570.

Hoos, H., & Tsang, E. (2006). Local search methods. In F. Rossi, P. Van Beek, & T. Walsh (Eds.), (chap. 8). Handbook of Constraint Programming.

Infanger, G. (1992). Monte carlo (importance) sampling within a benders decomposition algorithm for stochastic linear programs. *Annals of Operations Research*, *39*(1), 69–95.

Infanger, G. (1999). *Gams/decis users guide.* Dr. Gerd Infanger. Retrieved from http://www.gams.com/dd/docs/solvers/decis/

Jacob, R., Chase, & Aquilano. (2009). *Operations and supply management* (12th ed.). McGraw-Hill International Edition.

Jaffar, J., & Lassez, J.-L. (1987). Constraint logic programming. In *Proceedings of the 14th acm sigact-sigplan symposium on principles of programming languages* (pp. 111–119).

Jansson, C. (1997). Calculation of exact bounds for the solution set of linear interval systems. *Linear Algebra and Its Applications*, *251*, 321–340.

Jin, Y., Jiang, D., Yuan, S., Cao, J., Wang, L., & Zhou, G. (2008). Od count estimation based on link count data. *Challenges for Next Generation Network Operations and Service Management*, 217–226.

Jussien, N. (2003). *The versatility of using explanations within constraint programming*. Unpublished doctoral dissertation, Université de Nantes.

Kendall, M. G., et al. (1946). The advanced theory of statistics. *The advanced theory of statistics.*(2nd Ed).

Kessel, W. (2002). Measurement uncertainty according to ISO/BIPM-GUM. *Thermochimica acta*, *382*(1-2), 1–16.

Koster, A., Kutschka, M., & Raack, C. (2010). *Towards robust network design using integer linear programming techniques.*

Kuik, R., Salomon, M., & Van Wassenhove, L. (1994). Batching decisions: structure and models. *European Journal of Operational Research*, *75*(2), 243–263.

Lamma, E., Mello, P., Milano, M., Cucchiara, R., Gavanelli, M., & Piccardi, M. (1999). Constraint propagation and value acquisition: why we should do it interactively. In *Ijcai* (Vol. 99, pp. 467–473).

Lee, J. H.-M., & Van Emden, M. (1993). Interval computation as deduction in chip. *The Journal of Logic Programming*, *16*(3), 255–276.

Le Huitouze, S. (1990). A new data structure for implementing extensions to Prolog. In *Programming language implementation and logic programming* (pp. 136–150).

Mackworth, A. (1977). Consistency in networks of relations. *Artificial intelligence*, *8*(1), 99–118.

Markowitz, H. (1952). Portfolio selection*. *The journal of finance*, *7*(1), 77–91.

Matlab. (2010). *version 7.10.0 (r2010a)*. Natick, Massachusetts: The MathWorks Inc.

Mauris, G. (2007). Approximating coverage intervals of well-known continuous probability distributions by possibility distributions. In *Advanced methods for uncertainty estimation in measurement, 2007 ieee international workshop on* (pp. 5–9).

Mauris, G., Berrah, L., Foulloy, L., & Haurat, A. (2000). Fuzzy handling of measurement errors in instrumentation. *Instrumentation and Measurement, IEEE Transactions on*, *49*(1), 89–93.

Mauris, G., Lasserre, V., & Foulloy, L. (2000). Fuzzy modeling of measurement data acquired from physical sensors. *Instrumentation and Measurement, IEEE Transactions on*, *49*(6), 1201–1205.

Mauris, G., Lasserre, V., & Foulloy, L. (2001). A fuzzy approach for the expression of uncertainty in measurement. *Measurement*, *29*(3), 165–177.

Medina, A., Taft, N., Salamatian, K., Bhattacharyya, S., & Diot, C. (2002). Traffic matrix estimation: Existing techniques and new directions. *ACM SIGCOMM Computer Communication Review*, *32*(4), 161–174.

Meseguer, P., Rossi, F., & Schiex, T. (2006). Soft constraints. *Foundations of Artificial Intelligence*, *2*, 281–328.

Milano, M., Ottosson, G., Refalo, P., & Thorsteinsson, E. S. (2001). Global constraints: When constraint programming meets operation research. In *Informs journal on computing, special issue on the merging of mathematical programming and constraint programming.*

Minton, S., Johnston, M. D., Philips, A. B., & Laird, P. (1992). Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, *58*(1), 161–205.

Mittal, S., & Falkenhainer, B. (1990). Dynamic constraint satisfaction. In *Proceedings eighth national conference on artificial intelligence* (pp. 25–32).

Moon, I., & Gallego, G. (1994). Distribution free procedures for some inventory models. *The Journal of the Operational Research Society*, *45*(6), 651–658.

Mulvey, J. M., Vanderbei, R. J., & Zenios, S. A. (1995). Robust optimization of large-scale systems. *Operations research*, *43*(2), 264–281.

Nielsen, H. (2000). Know your uncertainty. *Quality*, *39*(4), 30–37.

Ning, S., & Kearfott, R. B. (1997). A comparison of some methods for solving linear interval equations. *SIAM Journal on Numerical Analysis*, *34*(4), 1289–1305.

Oettli, W. (1965). On the solution set of a linear system with inaccurate coefficients. *Journal of the Society for Industrial & Applied Mathematics, Series B: Numerical Analysis*, *2*(1), 115–118.

Older, W., & Vellino, A. (1993). Constraint arithmetic on real intervals. In *Constraint logic programming* (pp. 175–195).

Optimization, C. (1989). *Inc.(1989–1997): Using the cplex callable library, 930 tahoe blvd*. Bldg. 802.

Orlowski, S., Pióro, M., Tomaszewski, A., & Wessäly, R. (2007, April). SNDlib 1.0–Survivable Network Design Library. In *Proceedings of the 3rd international network optimization conference (inoc 2007), spa, belgium*. Retrieved from http://www.zib.de/orlowski/Paper/OrlowskiPioroTomaszewskiWessaely2007-SNDlib-INOC.pdf.gz (http://sndlib.zib.de, extended version accepted in Networks, 2009.)

Orlowski, S., Pióro, M., Tomaszewski, A., & Wessäly, R. (2010). SNDlib 1.0–Survivable Network Design Library. *Networks*, *55*(3), 276–286. Retrieved from http://www3.interscience.wiley.com/journal/122653325/abstract doi: 10.1002/net.20371

Petcu, A., & Faltings, B. (2005). Optimal solution stability in continuous-time optimization. In *Changes' 05 international workshop on constraint solving under change and uncertainty*.

Petrović, D., Petrović, R., & Vujošević, M. (1996). Fuzzy models for the newsboy problem. *International Journal of Production Economics*, *45*(1), 435–441.

Pheal, P. (1992). Inmon corporation's sflow: A method for monitoring traffic in switched and routed networks. *RFC3176, IETF*. Retrieved from http://www.ietf.org/rfc/rfc3176.txt

Potts, C., & Van Wassenhove, L. (1992). Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity. *Journal of the Operational Research Society*, 395–406.

Prekopa, A. (1973). Contributions to the theory of stochastic programming. *Mathematical Programming*, *4*(1), 202–221.

Puterman, M. L. (2009). *Markov decision processes: discrete stochastic dynamic programming* (Vol. 414). Wiley. com.

Ramesh, G., & Ganesan, K. (2011). Interval linear programming with generalized interval arithmetic. *a=[a, a]*, *1*(2), 1–2.

Refalo, P. (2000). Linear formulation of constraint programming models and hybrid solvers. In *Principles and practice of constraint programming–cp 2000* (pp. 369–383). Springer.

Régin, J.-C. (1996). Generalized arc consistency for global cardinality constraint. In *Proceedings of the thirteenth national conference on artificial intelligence-volume 1* (pp. 209–215).

Régin, J.-C. (1999). Arc consistency for global cardinality constraints with costs. In *Principles and practice of constraint programming–cp99* (pp. 390–404).

Régin, J.-C., Petit, T., Bessière, C., & Puget, J.-F. (2000). An original constraint based approach for solving over constrained problems. In *Principles and practice of constraint programming–cp 2000* (pp. 543–548). Springer.

Rockafellar, R. T., & Wets, R. J.-B. (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research*, *16*(1), 119–147.

Rohn, J. (1993). Cheap and tight bounds: The recent result by e. hansen can be made more efficient. *Interval Computations*, *4*(13-21), 2.

Rommelfanger, H. (1996). Fuzzy linear programming and applications. *European Journal of Operational Research*, *92*(3), 512–527.

Rossi, R. (2008). *Constraint programming for optimization under uncertainty in inventory control*. Unpublished doctoral dissertation, the National University of Ireland.

Saad, A. (2014). Cdf-intervals: A reliable framework to reason about data with uncertainty. *The 10th ICLP Doctoral Consortium*.

Saad, A., Frühwirth, T., & Gervet, C. (2014). The p-box cdf-intervals: Reliable constraint reasoning with quantifiable information. *Theory and Practice of Logic Programming*, *14*(4-5), 461–475.

Saad, A., Gervet, C., & Abdennadher, S. (2010). Constraint Reasoning with Uncertain Data Using CDF-Intervals. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, 292–306.

Saad, A., Gervet, C., & Fruehwirth, T. (2012a). CDF-Intervals: Reliable Constraint Reasoning with Quantifiable Information. *The Proceeding of the Doctoral Program of the 2012 International Conference on Principles and Practice of Constraint Programming*.

Saad, A., Gervet, C., & Fruehwirth, T. (2012b). CDF-Intervals Revisited. *The Eleventh International Workshop on Constraint Modelling and Reformulation - ModRef2012*.

Schiex, T. (1992). Possibilistic constraint satisfaction problems or how to handle soft constraints? In *Proceedings of the eighth international conference on uncertainty in artificial intelligence* (pp. 268–275).

Schiex, T., Fargier, H., & Verfaillie, G. (1995). Valued constraint satisfaction problems: Hard and easy problems. In *International joint conference on artificial intelligence* (Vol. 14, pp. 631–639).

Sen, S., & Higle, J. (1999). An introductory tutorial on stochastic linear programming models. *Interfaces*, 33–61.

Sen, S., Zhou, Z., & Huang, K. (2011). Stochastic decomposition and extensions. In *Stochastic programming* (pp. 57–66). Springer.

Sengupta, A., Pal, T., & Chakraborty, D. (2001). Interpretation of inequality constraints involving interval coefficients and a solution to interval linear programming. *Fuzzy Sets and systems*, *119*(1), 129–138.

Smith, W., & La Poutre, H. (1992). *Approximation of staircases by staircases* (Tech. Rep.). Citeseer.

Stark, H., & Woods, W. (1994). *Probability, Random Processes and Estimation Theory for Engineers* (2nd ed.). Prentice Hall.

Suprajitno, H., & Mohd, I. B. (2010). Linear programming with interval arithmetic. *International Journal of Contemporary Mathematical Sciences*, *5*(5-8), 323–332.

Tarim, S., & Kingsman, B. (2004). The stochastic dynamic production/inventory lot-sizing problem with service-level constraints. *International Journal of Production Economics*, *88*, 105–119.

Tarim, S., Manandhar, S., & Walsh, T. (2006). Stochastic constraint programming: A scenario-based approach. *Constraints*, *11*(1), 53–80.

Thipwiwatpotjana, P. (2010). *Linear programming problems for generalized uncertainty*. Unpublished doctoral dissertation, University of Colorado.

Thipwiwatpotjana, P., & Lodwick, W. A. (2008). Algorithm for solving optimization problems using interval valued probability measure. In *Fuzzy information processing society, 2008. nafips 2008. annual meeting of the north american* (pp. 1–5).

Tversky, A., & Kahneman, D. (1992). Advances in prospect theory: Cumulative representation of uncertainty. *Journal of risk and uncertainty*, *5*(4), 297–323.

Urbanski, M., & Wsowski, J. (2003). Fuzzy approach to the theory of measurement inexactness. *Measurement*, *34*(1), 67–74.

van Beek, P. (2006). Backtracking search algorithms. In F. Rossi, P. Van Beek, & T. Walsh (Eds.), (chap. 4). Handbook of Constraint Programming.

Van De Ree, R., & Jager, R. (1993). Control-data representation at knowledge level. In *Systems, man and cybernetics, 1993.'systems engineering in the service of humans', conference proceedings., international conference on* (pp. 702–706).

Van Hentenryck, P., & Carillon, J.-P. (1988). Generality versus specificity: An experience with ai and or techniques. In *Aaai* (pp. 660–664).

Van Hentenryck, P., & Le Provost, T. (1991). Incremental search in constraint logic programming. *New Generation Computing*, *9*(3-4), 257–275.

van Hoeve, W., & Katriel, I. (2006). Global constraints. In F. Rossi, P. Van Beek, & T. Walsh (Eds.), (chap. 7). Handbook of Constraint Programming.

van Hoeve, W.-J., & Katriel, I. (2006). Global constraints. *Handbook of constraint programming*, 169–208.

Verfaillie, G., & Schiex, T. (1994). Solution reuse in dynamic constraint satisfaction problems..

Vernooy, M., & Havens, W. S. (1999). *An examination of probabilistic value-ordering heuristics*. Springer.

Verweij, B., Ahmed, S., Kleywegt, A. J., Nemhauser, G., & Shapiro, A. (2003). The sample average approximation method applied to stochastic routing problems: a computational study. *Computational Optimization and Applications*, *24*(2-3), 289–333.

Von Neumann, J. (1947). Morgenstern, 0.(1944) theory of games and economic behavior. *Princeton: Princeton UP*.

Wagner, H., & Whitin, T. (1958). Dynamic version of the economic lot size model. *Management Science*, *50*(12), 1770–1774.

Wallace, R. J., & Freuder, E. C. (1998). Stable solutions for dynamic constraint satisfaction problems. In *Principles and practice of constraint programming-cp98* (pp. 447–461). Springer.

Walsh, T. (2000). Stochastic constraint programming. *constraints*, *2*(I5U), V8.

Walsh, T. (2002). Stochastic constraint programming. *Proceedings of the 15th Eureopean Conference on Artificial Intelligence*, 111–115.

Wets, R. J.-B. (1974). Stochastic programs with fixed recourse: The equivalent deterministic program. *SIAM Review*, *16*(3).

Williamson, R., & Downs, T. (1990). Probabilistic arithmetic. I. Numerical methods for calculating convolutions and dependency bounds* 1. *International Journal of Approximate Reasoning*, *4*(2), 89–158.

Xu, Y., & Hu, J. (2012). Random fuzzy demand newsboy problem. *Physics Procedia*, *25*, 924–931.

Yager, R. R. (1997). On a class of weak triangular norm operators. *Information Sciences*, *96*(1), 47–78.

Yorke-Smith, N. (2004). *Reliable constraint reasoning with uncertain data*. Unpublished doctoral dissertation, IC-Parc, Imperial College London.

Yorke-Smith, N., & Gervet, C. (2009). Certainty closure: Reliable constraint reasoning with incomplete or erroneous data. *ACM Transactions on Computational Logic (TOCL)*, *10*(1), 3.

Zadeh, L. A. (1965). Fuzzy sets. *Information and control*, *8*(3), 338–353.

Zhou, K., Doyle, J. C., & Glover, K. (1996). *Robust and optimal control*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Appendix

CHAPTER A

PROOFS SUPPORTING THE THEORETICAL FRAMEWORK

A.1 Stochastic ordering of p-box *cdf*-intervals points

Assume two points $p_x = (x, F_x^p, S_x^p)$ and $q_y = (x, F_y^q, S_y^q)$, each lying on a different uniform *cdf*-distribution. We need to prove that

$$F_Y^p \leqslant_S F_X^q : \int_{-\infty}^{y} F^q dy \leq \int_{-\infty}^{y} F^p dy \Leftrightarrow F_y^q \leq (y-x)S_x^p + F_x^p \tag{A.1}$$

Proof. Recall from Definition 2.8, to identify a second order stochastic dominance of a random variable over another, we perform the integration over the *cdf*-distribution, by calculating the area under the designated curves. The slope equations are:

$$\begin{aligned} S_x^p &= \frac{F_y^{p\prime} - F_x^p}{y-x} \\ S_y^q &= \frac{F_y^q - F_x^{q\prime}}{y-x} \end{aligned} \tag{A.2}$$

where $F_y^{p\prime}$ is the projected *cdf* value of the point q_y quantile y onto the *cdf*-distribution of point p_x. Similarily, $F_x^{q\prime}$ is the projected *cdf* value of the point p_x quantile x onto the *cdf*-distribution of point q_y. From the slope equations, we can derive the two projected values:

$$\begin{aligned} F_y^{p\prime} &= (y-x)S_x^p + F_x^p \\ F_x^{q\prime} &= (y-x)S_y^q - F_y^q \end{aligned} \tag{A.3}$$

The integrations over the given *cdf*-distributions are:

$$\int_{-\infty}^{y} F^p dy = F_y^{p\prime}$$
$$\int_{-\infty}^{y} F^q dy = F_y^q \tag{A.4}$$

If we substitute $F_y^{p\prime}$ with its derived value that includes the observed slope and *cdf*, the resulting linear equations are:

$$\int_{-\infty}^{y} F^p dy = (y - x)S_x^p + F_x^p$$
$$\int_{-\infty}^{y} F^q dy = F_y^q \tag{A.5}$$

□

A.2 Computing the p-box *cdf*-interval points of projection

For two real intervals $\mathbf{I} \in [a_0, b_1]$ and $\mathbf{J} \in [c_0, d_1]$ data is uniformly distributed between each interval bounds. Hence, due to the probability distribution function definition, from Equation 2.3, we have:

$$f_I(x) = \begin{cases} 0 & x < a_0 \\ \frac{1}{b_1 - a_0} & a_0 \leq x \leq b_1 \\ 0 & x > b_1 \end{cases} \tag{A.6}$$

$$f_J(y) = \begin{cases} 0 & y < c_0 \\ \frac{1}{d_1 - c_0} & c_0 \leq y \leq d_1 \\ 0 & y > d_1 \end{cases} \tag{A.7}$$

Consider two random variables X and Y defined over real intervals such that $X \in [a, b]$ and $Y \in [c, d]$, $X \subseteq \mathbf{I}$ and $Y \subseteq \mathbf{J}$, and $a_0 \leq a \leq b \leq b_1$, $c_0 \leq c \leq d \leq d_1$. The *cdf*-values for a, b, c, and d, taking their values from the *cdf* distributions formed by **I** and **J**, are F_a,F_b,F_c, and F_d respectively. The *cdf*-distribution functions defined for X and Y are:

$$F_X = \begin{cases} 0 & x < a \\ \frac{x}{b_1 - a_0} & a \leq x \leq b \\ 1 & x > b \end{cases} \tag{A.8}$$

$$F_Y = \begin{cases} 0 & y < c \\ \frac{y}{d_1 - c_0} & c \leq y \leq d \\ 1 & y > d \end{cases} \tag{A.9}$$

The red lines in Fig. A.1 illustrates the characteristic functions which represent X and Y distributions. Clearly, each has a line equation within the points bounding the

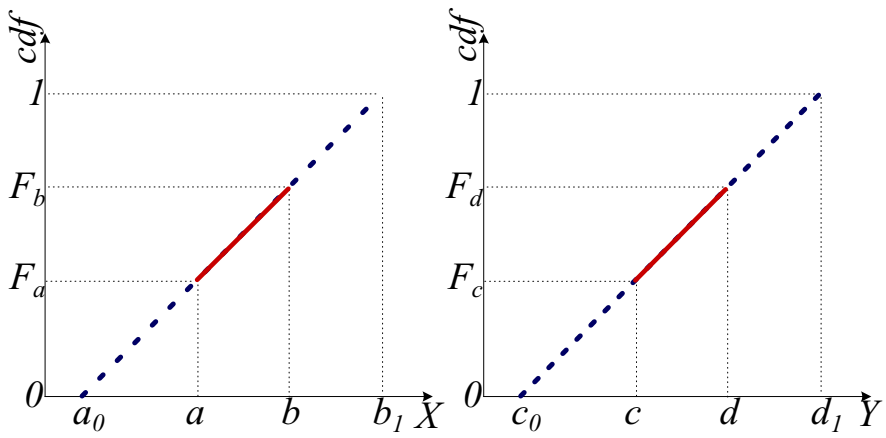


Fig. A.1: The characteristic functions of $X \subseteq \mathbf{I}$ and $Y \subseteq \mathbf{J}$

intervals $[a, b]$ and $[c, d]$. Given the values of a, b,c, d and their corresponding *cdf* values F_a, F_b,F_c, F_d, we can calculate the projections on the X-axis a_0, c_0, b_1 and d_1 of the intervals **I** and **J** from the slope (by projecting the line equation onto the '0' and '1' *cdf*-axis values). Accordingly we can deduce the following:

$$\begin{aligned} a_0 &= a - F_a \frac{(b-a)}{(F_b - F_a)} \\ b_1 &= b + \frac{(1-F_b)(b-a)}{(F_b - F_a)} \end{aligned} \tag{A.10}$$

Similarily

$$\begin{aligned} c_0 &= c - F_c \frac{(d-c)}{(F_d - F_c)} \\ d_1 &= d + \frac{(1-F_d)(d-c)}{(F_d - F_c)} \end{aligned} \tag{A.11}$$

A.3 Addition of two *cdf* uniform distributions

As noted in Section 2.1, the *cdf* is based on the integration of the *pdf*. We therefore derive the joint *pdf* over the interval of the addition. The resulting *pdf* is then integrated to obtain the joint *cdf*.

A.3.1 Deriving the joint *pdf* over the interval of the addition

Consider an interval Z, such that, $Z = X + Y$. X and Y are the intervals illustrated in Fig. A.1. The event $Z = z$ occurs if and only if values from X and Y, summed up together, are equal to z. In this relation, if $X = \tau$ then $Z = z$ if and only if $Y = z - \tau$, τ and z are arbitrary real values. Accordingly, the event $Z = z$ occurs when both events $X = \tau$ and

$Y = z - \tau$ happen together. τ can take any value from the real domain $\mathbb{R}$: $-\infty \leq \tau \leq +\infty$. The probability distribution function of Z can be obained from the following convolution operations derived as follows:

$$F_Z(z) = \iint\limits_{x+y\leq z} f_{XY}(x,y)\,dx\,dy = \int_{-\infty}^{+\infty}(\int_{-\infty}^{z-y} f_{XY}(x,y)dx)dy$$

substituting

$$G_{XY}(x,y) = \int f_{XY}(x,y)dx$$

then

$$F_Z(z) = \int_{-\infty}^{+\infty}[G_{XY}(z-y,y) - G_{XY}(-\infty,y)]dy \tag{A.12}$$

$$F_Z(z) = \int_{-\infty}^{+\infty}\int f_{XY}(z-y,y) - f_{XY}(-\inf,y)dy \tag{A.13}$$

for independent random variable

$$f_{XY}(-\infty,y) = f_{XY}(-\infty)\,f_{XY}(y) \tag{A.14}$$

and by definition $f_{XY}(-\infty) = 0$ then $f_{XY}(-\infty,y) = 0$ and

$$F_Z(z) = \int_{-\infty}^{+\infty}\int f_{XY}(z-y,y)dy \tag{A.15}$$

which is the convolution operation defined by the following equation:

$$f_Z(z) = \int_{-\infty}^{+\infty} f_X(\tau)\,f_Y(z-\tau)d\tau = \int_{-\infty}^{+\infty} f_X(z-\tau)\,f_Y(\tau)d\tau \tag{A.16}$$

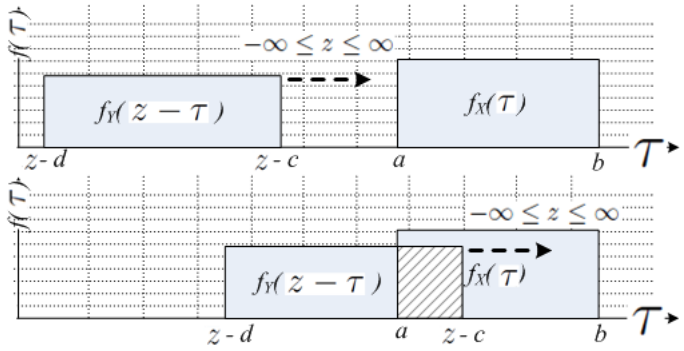


Fig. A.2: The convolution operation

Fig. A.2 illustrates the convolution operation on two uniformly distributed intervals $X \in [a,b]$ and $Y \in [c,d]$. The visual convolution operation exerts the following steps:

1. Each distribution is expressed in terms of a variable τ: $f_X(\tau)$ and $f_Y(\tau)$

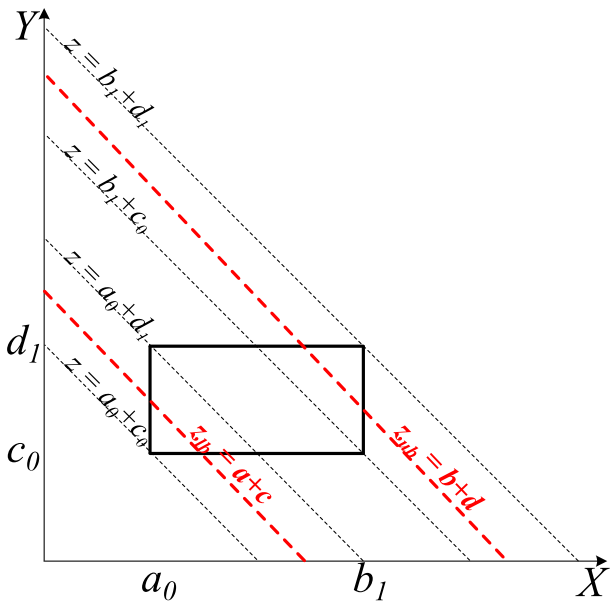


Fig. A.3: Z values as function of quantiles taken from X and Y given that $Z = X + Y$

2. One of the two functions is mirrored around the real 0: $f_Y(-\tau)$

3. An offset is added to the mirrored function: $f_Y(z - \tau)$

4. The offset z takes values from $-\infty$ to $+\infty$ to allow $f_Y(z - \tau)$ to slide along the τ axis; as shown by the sliding arrow in Fig. A.2

5. As illustrated in Fig. A.2, the shaded regions are superimposed to compute the integration

Values of the random variable Z, as a function of X and Y, form the set of declined lines drawn in Fig. A.3. Since both X and Y are random variables mapped onto two real intervals bounded by $[a, b]$ and $[c, d]$ respectively, Z is mapped onto the real interval bounds $[a + c, b + d]$. The characteristic function of Z is a piecewise linear function defined by the 5 different regions. Bounds differenciating each region are determined by the dotted blue lines: $z = a_0 + c_0$, $z = a_0 + d_1$, $z = b_1 + c_0$ and $z = b_1 + d_1$. The *pdf* of Z is computed as follows:

$$z_{lb} = \min(a_0 + d_1, c_0 + b_1) \quad \text{and} \quad z_{ub} = \max(a_0 + d_1, c_0 + b_1) \tag{A.17}$$

Then $f_Z : \mathbb{R} \to [0, \infty)$ is the piecewise linear function given by:

$$f_Z(z) = \begin{cases} 0, & z < a_0 + c_0 \\ \frac{z-(a_0+c_0)}{(b_1-a_0)(d_1-c_0)}, & a_0 + c_0 \leq z \leq z_{lb} \\ \frac{z_{lb}-(a_0+c_0)}{(b_1-a_0)(d_1-c_0)}, & z_{lb} \leq z \leq z_{ub} \\ \frac{(b_1+d_1)-z}{(b_1-a_0)(d_1-c_0)}, & z_{ub} \leq z \leq b_1 + d_1 \\ 0, & z > b_1 + d_1 \end{cases} \tag{A.18}$$

A.3.2 Deriving the *cdf* over the interval of the addition

By definition, the *cdf* distribution is obtained by integrating the *pdf*. Due to its monotonic property, we only need to calculate the *cdf* of the bounding quantiles: $z = a + c$ and $z = b + d$, the calculated minimum and maximum quantiles respectively. The events $z = a + c$ and $z = b + d$ are located within the set of regions depicted in Fig. A.3. Note that $(a_0 + c_0) \leq (a + c)$ and that $(b + d) \leq (b_1 + d_1)$ because $[a, b] \subseteq [a_0, b_1]$ and $[c, d] \subseteq [c_0, d_1]$. For an arbitrary $z = x + y$ where $x \in [a, b]$ and $y \in [c, d]$:

$$F_Z(z = x + y) = \int_{a_0+c_0}^{x+y} f_{XY}(z)dz$$

We can replace $f_{XY}(z)$ by its constituents from Equation A.18.

$$\begin{aligned} F_Z(z = x + y) \quad &= \int_{a_0+c_0}^{x+y} \int_{a_0}^{z-c_0} \tfrac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (a_0 + c_0) \leq (x + y) \leq (a_0 + d_1) \\ &+ \int_{a_0+d_1}^{x+y} \int_{c_0}^{d_1} \tfrac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (a_0 + d_1) \leq (x + y) \leq (b_1 + c_0) \\ &+ \int_{b_1+c_0}^{x+y} \int_{z-d_1}^{b_1} \tfrac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (b_1 + c_0) \leq (x + y) \leq (b_1 + d_1) \end{aligned} \tag{A.19}$$

$$F_Z(z = x + y) = \frac{1}{2} \frac{(a_0 + c_0 - x - y)^2}{(b_1 - a_0)(d_1 - c_0)} \quad (a_0 + c_0) \leq (a + c) \leq (a_0 + d_1)$$

$$\begin{aligned} F_Z(z = x + y) \quad &= \tfrac{1}{2} \tfrac{(c_0-d_1)}{(b_1-a_0)} + \tfrac{(x+y-a_0-d_1)}{(b_1-a_0)} \\ &(a_0 + d_1) \leq (x + y) \leq (b_1 + c_0) \end{aligned}$$

$$\begin{aligned} F_Z(z = x + y) \quad &= \tfrac{1}{2} \tfrac{(c_0-d_1)}{(b_1-a_0)} + \tfrac{(b_1+c_0-a_0-d_1)}{(b_1-a_0)} \\ &+ \tfrac{1}{(b_1-a_0)(d_1-c_0)} \{ \tfrac{(a_0+c_0)^2}{2} - \tfrac{(b_1+c_0)^2}{2} + (b_1 + d_1)(x + y - b_1 - d_1) \} \\ &(b_1 + c_0) \leq (x + y) \leq (b_1 + d_1) \end{aligned}$$

From the equations listed above, and knowing the projected values of the uniform distribution bounds obtained in Equation A.10, we compute the result of the convolution linearily.

The slope, by definition, is the average step value the *cdf* distribution takes. We compute the slope of the *cdf* distribution resulting from the addition operation from Equation A.23 as follows:

$$S_z = \frac{1}{(b_1+d_1)-(a_0+c_0)}$$

This is due to the fact that $(a_0 + c_0)$ and $(b_1 + d_1)$ are respectively the lower and upper bounds of the distribution and their *cdf* values are equal to '0' and '1'.

A.4 Multiplication of two *cdf* uniform distribution

The derivation of the binary multiplication operation is similar to the addition we provided in Section A.3. However, in this case, for $Z = XY$, the event $Z = z$ occurs if and only if values from X and Y, multiplied together, are equal to z. Deriving the *cdf* of the multiplication is based on the work introduced in [Glen, Leemis, DrewGlen et al.].

A.4.1 Deriving the joint *pdf* over the interval of the product

In this section, we show how to derive the *pdf* of the product of the two random variables X and Y, illustrated in Fig. A.1, and which belong to a product operation $Z = XY$. The joint *pdf* of Z can be obained using the following formula [Stark and Woods (1994)]:

$$f_{XY}(z) = \int_{-\infty}^{+\infty} \frac{1}{|\tau|} f_X(\tau) f_Y\left(\frac{z}{\tau}\right) d\tau = \int_{-\infty}^{+\infty} \frac{1}{|\tau|} f_X\left(\frac{z}{\tau}\right) f_Y(\tau) d\tau \tag{A.20}$$

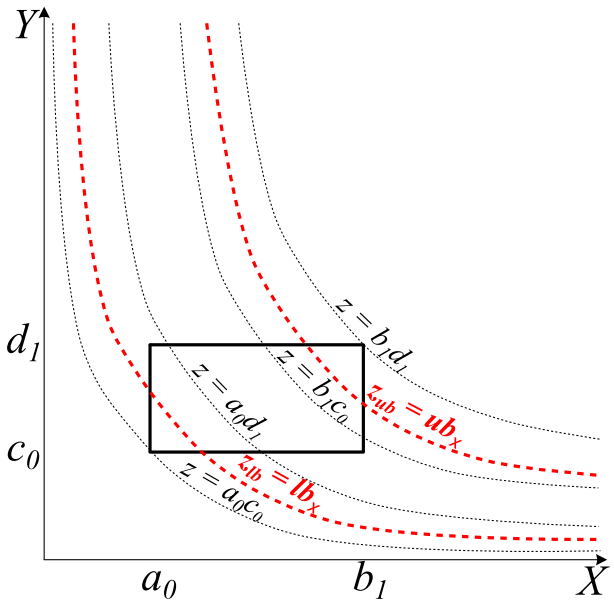


Fig. A.4: Z values as function of quantiles taken from X and Y given that $Z = XY$

According to Glen et al. (2004), we divide the domain of $Z = XY$ into 3 main segments as depicted in Fig. A.4. They are bounded by the dotted lines: $z = a_0c_0$, $z = a_0d_1$, $z = b_1c_0$ and $z = b_1d_1$. When $(a_0d_1) \leq (b_1c_0)$, the probability distribution function of Z is described as follows:

$$f_Z(z) = \begin{cases} 0, & z < (a_0c_0) \\ \int_{a_0}^{\frac{z}{c_0}} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau & (a_0c_0) \leq z \leq (a_0d_1) \\ \int_{\frac{z}{d_1}}^{\frac{z}{c_0}} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau & (a_0d_1) \leq z \leq (b_1c_0) \\ \int_{\frac{z}{d_1}}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau & (b_1c_0) \leq z \leq (b_1d_1) \\ 0, & z > b_1d_1 \end{cases} \tag{A.21}$$

A.4.2 Deriving the *cdf* over the interval of the multiplication

To calculate the *cdf* on the quantile bounds of the multiplication: $lb_\times = \min(ac, ad, bc, bd)$ and $ub_\times = \max(ac, ad, bc, bd)$. The events $z = lb_\times$ and $z = ub_\times$ are located within the set of intersecting segments depicted in Fig. A.4. For an arbitrary $z = xy$ where $x \in X$ and $y \in Y$:

$$F_Z(z = xy) = \int_{a_0c_0}^{xy} f_{XY}(z)dz$$

Replacing $f_{XY}(z)$ by its constituents, we obtain the following:

$$\begin{aligned} F_Z(z = xy) &= \int_{a_0c_0}^{xy} \int_{a_0}^{\frac{z}{c_0}} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau dz \quad (a_0c_0) \leq (xy) \leq (a_0d_1) \\ &+ \int_{a_0d_1}^{xy} \int_{\frac{z}{d_1}}^{\frac{z}{c_0}} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau dz \quad (a_0d_1) \leq (xy) \leq (b_1c_0) \\ &+ \int_{b_1c_0}^{xy} \int_{\frac{z}{d_1}}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} \frac{1}{|\tau|} d\tau dz \quad (b_1c_0) \leq (xy) \leq (b_1d_1) \end{aligned} \tag{A.22}$$

$$F_Z(z = xy) = \frac{xy\left(\ln\left(\frac{xy}{a_0c_0}\right)\right) + a_0c_0}{(b_1 - a_0)(d_1 - c_0)} \quad (a_0c_0) \leq (xy) \leq (a_0d_1)$$

$$\begin{aligned} F_Z(z = xy) &= \frac{a_0d_1\left(\ln\left(\frac{a_0d_1}{a_0c_0}\right)\right)+a_0c_0}{(b_1-a_0)(d_1-c_0)} + \frac{\ln\left(\frac{d_1}{c_0}\right)(xy-a_0d_1)}{(b_1-a_0)(d_1-c_0)} \\ &\quad (a_0d_1) \leq (xy) \leq (b_1c_0) \end{aligned}$$

$$\begin{aligned} F_Z(z = xy) &= \frac{a_0d_1\left(\ln\left(\frac{a_0d_1}{a_0c_0}\right)\right)+a_0c_0}{(b_1-a_0)(d_1-c_0)} + \frac{\ln\left(\frac{d_1}{c_0}\right)(b_1c_0-a_0d_1)}{(b_1-a_0)} \\ &+ \frac{xy\ln\left(\frac{b_1d_1}{xy}\right)-b_1c_1\ln\left(\frac{b_1d_1}{b_1c_0}\right)+xy-b_1c_0}{(b_1-a_0)(d_1-c_0)} \qquad (b_1c_0) \leq (xy) \leq (b_1d_1) \end{aligned}$$

The slope of the *cdf* distribution resulting from the multiplication operation is defined as follows:

$$S_z = \frac{1}{(b_1 d_1)-(a_0 c_0)}$$

Note that, in this case, $(a_0 c_0)$ and $(b_1 d_1)$ are respectively the quantile bounds of the distribution with *cdf* values equal to '0' and '1'.

A.5 Subtraction of two *cdf* uniform distribution

A.5.1 Deriving the joint *pdf* over the interval of the difference

Consider an interval Z, such that, $Z = X - Y \equiv Z = X + (-Y)$. If $X = \tau$, then, the event $Z = z$ occurs only if $Y = z + \tau$ takes place. τ can take any value from the real domain $\mathbb{R}$: $-\infty \leq \tau \leq +\infty$. The probability distribution function of Z can be obained as follows:

$$f_{XY}(z) = \int_{-\infty}^{+\infty} f_X(\tau) f_Y(z+\tau) d\tau \tag{A.23}$$

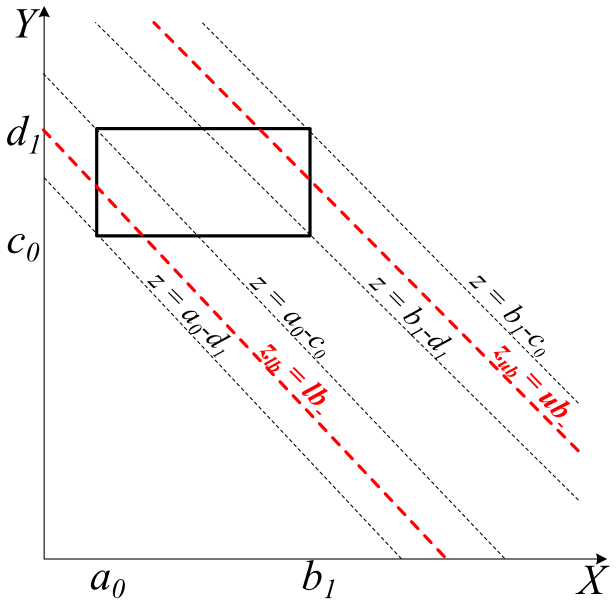


Fig. A.5: Z values as function of quantiles taken from X and Y given that $Z = X - Y$

Depicted in Fig. A.5, the domain of Z in terms of the X and Y real interval domains. When $(a_0 - c_0) \leq (b_1 - d_1)$, the probability density function of Z is defined as follows:

$$f_Z(z) = \begin{cases} 0, & z < a_0 - c_0 \\ \int_{a_0}^{z+d_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau & (a_0 - d_1) \le z \le (a_0 - c_0) \\ \int_{a_0}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau & (a_0 - c_0) \le z \le (b_1 - d_1) \\ \int_{z+c_0}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau & (b_1 - d_1) \le z \le (b_1 - c_0) \\ 0, & z > b_1 - d_1 \end{cases} \tag{A.24}$$

A.5.2 Deriving the *cdf* over the interval of the subtraction

The *cdf* of the bounding quantiles formed by the distribution of Z are $z = lb_-$ and $z = ub_-$, the minimum and the maximum quantiles respectively.

$lb_- = \min(a-d, a-c, b-c, b-d)$ and $ub_- = \max(a-d, a-c, b-c, b-d)$

$$F_Z(z = (x-y)) = \int_{a_0-d_1}^{(x-y)} f_{XY}(z) dz$$

$f_{XY}(z)$ is computed as follows:

$$\begin{aligned} F_Z(z = (x-y)) &= \int_{a_0-d_1}^{(x-y)} \int_{a_0}^{z+d_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (a_0 - d_1) \le (x-y) \le (a_0 - c_0) \\ &+ \int_{a_0-c_0}^{(x-y)} \int_{a_0}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (a_0 - c_0) \le (x-y) \le (b_1 - d_1) \\ &+ \int_{b_1-d_1}^{(x-y)} \int_{z+c_0}^{b_1} \frac{1}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad (b_1 - d_1) \le (x-y) \le (b_1 - c_0) \end{aligned} \tag{A.25}$$

$$F_Z(z = (x-y)) = \frac{1}{2} \frac{((x-y) - (a_0 - d_1))^2}{(b_1 - a_0)(d_1 - c_0)} \quad (a_0 - d_1) \le (x-y) \le (a_0 - c_0)$$

$$\begin{aligned} F_Z(z = (x-y)) &= \tfrac{1}{2} \tfrac{(b_1-a_0)}{(d_1-c_0)} + \tfrac{((x-y)-(a_0-c_0))}{(d_1-c_0)} \\ &(a_0 - c_0) \le (x-y) \le (b_1 - d_1) \end{aligned}$$

$$\begin{aligned} F_Z(z = (x-y)) &= \tfrac{1}{2} \tfrac{(b_1-a_0)}{(d_1-c_0)} \\ &+ \tfrac{1}{(b_1-a_0)(d_1-c_0)} \{ \tfrac{(b_1-d_1)^2}{2} - \tfrac{((x-y))^2}{2} + (b_1 - c_0)((x-y) - (b_1 - d_1)) \} \\ &(b_1 - d_1) \le (x-y) \le (b_1 - c_0) \end{aligned}$$

The slope of the *cdf* distribution resulting from the subtraction operation is:

$$S_z = \tfrac{1}{(a_0-d_1)-(b_1-c_0)}$$

$(a_0 - d_1)$ and $(b_1 - c_0)$ are the distribution quantile bounds and their *cdf* values are equal to 0 and 1.

A.6 Division of two *cdf* uniform distribution

A.6.1 Deriving the joint *pdf* over the interval of the division

Consider the two intervals X and Y, illustrated in Fig. A.1, and belonging to a division operation $Z = X \div Y$. The *pdf* distribution of Z is defined in [Stark and Woods (1994)] as follows:

$$f_{XY}(z) = \int_{-\infty}^{+\infty} |\tau|\, f_X(\tau) f_Y(z\tau)\, d\tau = \int_{-\infty}^{+\infty} |\tau|\, f_X(z\tau)\, f_Y(\tau) d\tau \tag{A.26}$$

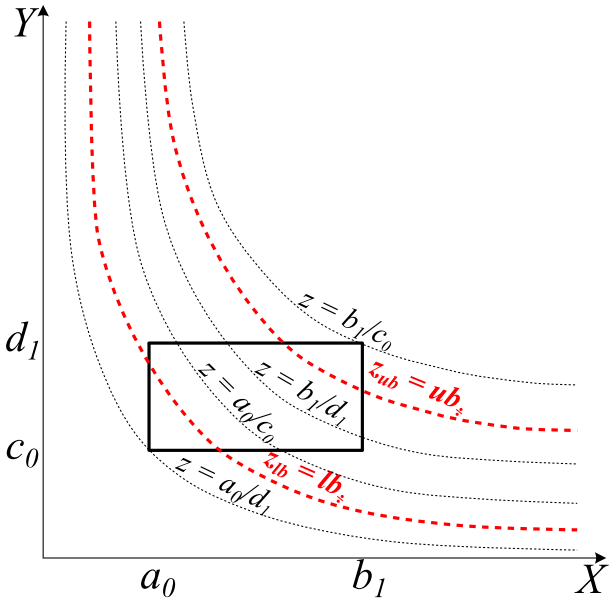


Fig. A.6: Z values as function of quantiles taken from X and Y given that $Z = X \div Y$

Given that $\frac{a_0}{c_0} \leq \frac{b_1}{d_1}$, the *pdf* of Z is defined as follows:

$$f_Z(z) = \begin{cases} 0, & z < \frac{a_0}{c_0} \\ \int_{\frac{a_0}{z}}^{d_1} \frac{|\tau|}{b_1 - a_0} \frac{1}{d_1 - c_0} d\tau & \frac{a_0}{d_1} \leq z \leq \frac{a_0}{c_0} \\ \int_{c_0}^{d_1} \frac{|\tau|}{(b_1 - a_0)(d_1 - c_0)} d\tau & \frac{a_0}{c_0} \leq z \leq \frac{b_1}{d_1} \\ \int_{c_0}^{\frac{b_1}{z}} \frac{|\tau|}{(b_1 - a_0)(d_1 - c_0)} d\tau & \frac{b_1}{d_1} \leq z \leq \frac{b_1}{c_0} \\ 0, & z > \frac{b_1}{d_1} \end{cases} \tag{A.27}$$

A.6.2 Deriving the *cdf* over the interval of the division

The *cdf* of the bounding quantiles formed by the distribution of Z are $z = lb_{\div}$ and $z = ub_{\div}$, the minimum and the maximum quantiles respectively.

$$F_Z(z = \frac{x}{y}) = \int_{\frac{a_0}{d_1}}^{\frac{x}{y}} f_{XY}(z)dz$$

$f_{XY}(z)$ is computed as follows:

$$\begin{aligned} F_Z(z = \frac{x}{y}) &= \int_{\frac{a_0}{d_1}}^{\frac{x}{y}} \int_{\frac{a_0}{z}}^{d_1} \frac{|\tau|}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad \frac{a_0}{d_1} \leq \frac{x}{y} \leq \frac{a_0}{c_0} \\ &+ \int_{\frac{a_0}{c_0}}^{\frac{x}{y}} \int_{c_0}^{d_1} \frac{|\tau|}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad \frac{a_0}{c_0} \leq \frac{x}{y} \leq \frac{b_1}{d_1} \\ &+ \int_{\frac{b_1}{d_1}}^{\frac{x}{y}} \int_{c_0}^{\frac{b_1}{z}} \frac{|\tau|}{(b_1-a_0)(d_1-c_0)} d\tau dz \quad \frac{b_1}{d_1} \leq \frac{x}{y} \leq \frac{b_1}{c_0} \end{aligned} \tag{A.28}$$

$$F_Z(z = \frac{x}{y}) = \frac{1}{2} \frac{d_1^2 \left(\frac{x}{y} - \frac{a_0}{d_1}\right) + a_0^2 \left(\frac{1}{\frac{x}{y}} - \frac{d_1}{a_0}\right)}{(b_1 - a_0)(d_1 - c_0)} \quad \frac{a_0}{d_1} \leq \frac{x}{y} \leq \frac{a_0}{c_0}$$

$$\begin{aligned} F_Z(z = \frac{x}{y}) &= \frac{1}{2} \frac{d_1^2\left(\frac{a_0}{c_0} - \frac{a_0}{d_1}\right) + a_0^2\left(\frac{c_0}{a_0} - \frac{d_1}{a_0}\right)}{(b_1-a_0)(d_1-c_0)} \frac{1}{2} \frac{d_1^2 - c_0^2}{(b_1-a_0)(d_1-c_0)} \left(\frac{x}{y} - \frac{a_0}{c_0}\right) \\ &\frac{a_0}{c_0} \leq \frac{x}{y} \leq \frac{b_1}{d_1} \end{aligned}$$

$$\begin{aligned} F_Z(z = \frac{x}{y}) &= \frac{1}{2} \frac{d_1^2\left(\frac{a_0}{c_0} - \frac{a_0}{d_1}\right) + a_0^2\left(\frac{c_0}{a_0} - \frac{d_1}{a_0}\right)}{(b_1-a_0)(d_1-c_0)} \frac{1}{2} \frac{d_1^2 - c_0^2}{(b_1-a_0)(d_1-c_0)} \left(\frac{x}{y} - \frac{a_0}{c_0}\right) \\ &+ \frac{1}{2} \frac{1}{(b_1-a_0)(d_1-c_0)} \left(d_1 - \frac{b_1}{\frac{x}{y} + c_0^2\left(\frac{b_1}{d_1} - \frac{x}{y}\right)}\right) \\ &\frac{b_1}{d_1} \leq \frac{x}{y} \leq \frac{b_1}{c_0} \end{aligned}$$

The slope of the *cdf* distribution resulting from the division operation is:

$$S_z = \frac{1}{(\frac{a_0}{d_1}) - (\frac{b_1}{c_0})}$$

Given that $\frac{a_0}{d_1}$ and $\frac{b_1}{c_0}$ are bounds of the distribution resulting from the division operation and their *cdf* values are equal to ‘0‘ and ‘1‘.

CHAPTER B

THE P-BOX *cdf*-INTERVALS SOLVER IMPLEMENTATION

The p-box *cdf*-intervals solver is implemented as a separate module in the ECL^{i}PS^e constraint programming environment ECRC (1994). The solver follows the structure of the ic library* ECRC (1994), it consists of two main modules imported using the ‘use_module(cdfi)‘ directive. The first module, ‘cdfi_kernel‘ details all the core operations which are exerted on the *cdf*-intervals. The second module, ‘cdfi‘ defines the list of constraints and their behavior.

B.1 Syntax

A triplet point in the p-box *cdf*-interval is denoted as ‘`Q~F|S`‘, where ‘`Q`‘ is the quantile value ‘`F`‘ is the *cdf* value and ‘`S`‘ is the slope of the *cdf* distribution. By means of the ‘`~`‘ and ‘`|`‘ operators we separate the elements of a point which can take any value from $\mathbb{R}$.

A p-box *cdf*-interval domain is input to the solver as ‘`Glb ... Lub`‘. This representation indicates that we have a convex p-box *cdf*-interval which is bounded by the two triplet points ‘`Glb`‘ and ‘`Lub`‘ and separated by the ‘`...`‘ operator

Operators are accordingly declared in the solver as follows

```
% declaring the infix representation of the cdf domain operators
:- export
  op(550,xfx,'.::'),
  op(500,xfx,'...'),
  op(450,xfx,'~'),
  op(400,xfx,'|').
```

*The ic library is the constraint solver implementation of the integer/real interval arithmetic in ECL^{i}PS^e

B.2 Core operations

The core operations are implemented in the ‘cdfi-kernel‘ module. They detail all computations exerted on the *cdf* distributions. Listed below part of this implementation incorporating: the projection of a quantile onto the *cdf*-distribution, determining the dominance of a *cdf*-distribution when compared to another, extracting triplet points out of the *glb* and *lub* operations, exerting the intersection operation between two *cdf* intervals, and computing the arithmetic operations on a pair of p-box *cdf*-intervals.

The projection operation computes the value of the *cdf* for a given real quantile located within the interval bounds.

```
% ---- Projection
% find_cdfproj(?X,?Y,-Fxy)
% X is a cdf variable, Y is a cdf variable
% Fxy is the cdf projection of X onto Y
% returns the cdf ground value of X variable
% quantile when projected onto the cdf line of Y
find_cdfproj(X~_FX|_SX, Y~FY|SY, FXP):-
  FXP = FY - SY * (Y - X).
```

Extracting the quantiles when the *cdf* is ‘0‘ or ‘1‘ is reversing the projection operation. Both are utilized when we need to compute the real interval overwhich the *cdf* uniform distribution is defined[†].

```
% get quantile value @ 0 cdf
get_quantileat0(A~FA|SA, A0):-
  A0 is A - (FA / SA).
```

```
% get quantile value @ 1 cdf
get_quantileat1(A~FA|SA, A1):-
  A1 is A - (FA - 1)/ SA).
```

The stochastic dominance compares between two uniform *cdf* distributions. As detailed in Section 2.3, it is determined by integrating the *cdf*-curves over $\mathbb{R}$. This integration is a result of calculating the areas enveloped by the *cdf* distributions. Accordingly, the dominant *cdf* is the distribution enclosing a minimum area, whereas the dominated distribution is encapsulating the maximum area.

```
% ---- Stochastic Dominance
% stochastic_dominance(?VarX,?VarY,-DominatedVar,-DominantVar)
% given two cdf lines issued from the same quantile
% return the dominated and the dominant cdf
% VarX is a cdf variable, VarY is a cdf variable,
```

[†]Readers can refer to Equation A.10 that defines the uniform distribution over an interval $[a, b]$

```
% DominatedVar is the stochastic dominated variable
% DominantVar is the stochastic dominant variable
% calculate the area under the cdf curve
% over the real domain from -1.0Inf to +1.0Inf
% by computing quantiles of each curve having cdf 0 and 1
% the area having a minimum value is the dominant and vice versa
stochastic_dominance(X~FY|SY,X~FX|SX,X~Flb|Slb,X~Fub|Sub):-
  get_quantileat1(X~FX|SX, X1), get_quantileat1(X~FY|SY, Y1),
  ((SY >= +1.0Inf) -> AREA1 is +1.0Inf;
    get_quantileat0(X~FY|SY, Y0),
    % calculate the area under the cdf between Y0 and Y1
    AREA10 is 0.5*(Y1 - Y0) ),
  ( (SX >= +1.0Inf) -> AREA2 is +1.0Inf;
    get_quantileat0(X~FX|SX, X0),
    % calculate the area under the cdf between X0 and X1
    AREA20 is 0.5*(X1 - X0) ),
  % find the maximum quantile then add up
  % the rest of the area under the cdf curve
  ((Y1 > X1) ->
      AREA1 = AREA10,  AREA2 is AREA20 + Y1
      ;
      AREA2 = AREA20, AREA1 is AREA10 + X1 ),
  ((AREA1 > AREA2) ->
    Flb is FY, Slb is SY, Fub is FX, Sub is SX;
    Flb is FX, Slb is SX, Fub is FY, Sub is SY).
```

The *glb* is the min quantile value projected onto the dominated *cdf* bounding the max area.

```
% ---- Greatest Lower Bound
% glb(?VarX,?VarY,-GLB)
% VarX is a cdf variable, VarY is a cdf variable,
% GLB is the greatest lower bound
% the point with minimum quantile value and dominated cdf
% 1. project X onto the cdf curve of Y
% 2. Glb is the min quantile value projected
%    onto the dominated cdf bounding the max area
glb(X~FX|SX , Y~FY|SY , Glb~Flb|Slb) :-
  ( X =< Y ->
    % project X onto the cdf curve of Y
    find_cdfproj(X~FX|SX, Y~FY|SY, FXY),
    % Glb is the min quantile value projected
    % onto the max area
    stochastic_dominance(X~FXY|SY,X~FX|SX,Glb~Flb|Slb,_~_|_)
    ;
    % project X onto the cdf curve of Y
    find_cdfproj(Y~FY|SY, X~FX|SX, FYX),
```

```
    stochastic_dominance(Y~FYX|SX,Y~FY|SY,Glb~Flb|Slb,_~_|_)
  ).
```

The *lub* is the max quantile value projected onto the dominant *cdf* bounding the min area.

```
% ---- Least Upper Bound
% lub(?VarX,?VarY,-LUB)
% VarX is a cdf variable, VarY is a cdf variable,
% LUB is the least upper bound
% the point with maximum quantile value and dominant cdf
% 1. project X onto the cdf curve of Y
% 2. Lub is the max quantile value projected
%    onto the dominant cdf bounding the min area
lub(X~FX|SX , Y~FY|SY , Lub~Fub|Sub) :-
  ( X =< Y ->
    % project Y onto the cdf curve of X
    find_cdfproj(Y~FY|SY, X~FX|SX, FYX),
    % Lub is the max quantile value projected onto the min area
    stochastic_dominance(Y~FYX|SX,Y~FY|SY,_~_|_,Lub~Fub|Sub)
    ;
    % project X onto the cdf curve of X
    find_cdfproj(X~FX|SX, Y~FY|SY, FXY),
    % Lub is the max quantile value projected onto the min area
    stochastic_dominance(X~FXY|SY,X~FX|SX,_~_|_,Lub~Fub|Sub)
  ).
```

The intersection seeks to find the p-box *cdf*-interval resulting from joining two intervals. This operation starts by extracting the *lub* of the intervals lower bounds, then, it computes the *glb* of the upper bounds. Finally, it detects whether an area of conflict[‡] needs to be removed.

```
% ---- CDFI Interval Intersection Operation
% intersect(?VarX,?VarY,-Result)
% VarX is a cdf interval (A~FA|SA...B~FB|SB),
% VarY is a cdf interval (X0~FX0|SX0...X1~FX1|SX1),
% Result is a cdf interval (XL~FXL|SXL ...XU~FXU|SXU)
intersect(VarX ,VarY ,I):-
  lub(A~FA|SA, X0~FX0|SX0, XL0~FXL0|SXL0),
  % project XL onto upper bounds
  find_cdfproj(XL0~FXL0|SXL0, B~FB|SB, FXL1),
  find_cdfproj(XL0~FXL0|SXL0, X1~FX1|SX1, FXL2),
  FXLmax is max(FXL1,FXL2),
  ((FXLmax = FXL1) -> SXLmax is SB; SXLmax is SX1),
```

[‡]Readers can refer to Section 6.3.2 for a detailed information about the area of conflict

```
  ((FXL0 > FXLmax) ->
    XL is XL0, FXL is FXL0, SXL is SXL0
    ;
    % extract the intersecting quantile point
    get_xi(XL0~FXL0|SXL0,XL0~FXLmax|SXLmax,XLI~_|_),
    XL is XLI,
    find_cdfproj(XLI~FXLmax|SXLmax, XL0~FXLmax|SXLmax, FXL),
    SXL is SXL0
  ),

  glb(B~FB|SB, X1~FX1|SX1, XU0~FXU0|SXU0),
  % project XU onto lower bounds
  find_cdfproj(XU0~FXU0|SXU0, A~FA|SA, FXU1),
  find_cdfproj(XU0~FXU0|SXU0, X0~FX0|SX0, FXU2),

  FXUmin is min(FXU1,FXU2),
  ((FXUmin = FXU1) -> SXUmin is SA; SXUmin is SX0),
  ((FXU0 < FXUmin) ->
    XU is XU0, FXU is FXU0, SXU is SXU0
    ;
    % extract the intersecting quantile point
    get_xi(XU0~FXU0|SXU0,XU0~FXUmin|SXUmin,XUI~_|_),
    XU is XUI,
    find_cdfproj(XLI~FXUmin|SXUmin, XU0~FXUmin|SXUmin, FXU),
    SXU is SXU0
).
```

Arithmetic operations compute the addition, multiplication, subtraction and division operations on a pair of *cdf*-uniform distributions. Each operation is implemented to work on the interval bounding triplet points in pairs, i.e. operations on the two interval lower bounds yield the resultant interval lower bound, and vice versa.

```
% ---- cdf-intervals arithmetic addition
% arith_addition(?VarX,?VarY,-Result)
% VarX is a cdf interval (A~FA|SA...B~FB|SB),
% VarY is a cdf interval (C~FC|SC...D~FD|SD),
% VarZ is the result cdf interval (Zl~Fl|Sl...Zu~Fu|Su)
arith_addition(VarX,VarY,VarZ):-
  Zl is A+C, Zu is B+D,
  % get the bounds of the cdf uniform distributions
  get_quantileat0(A~FA|SA, A1),get_quantileat1(A~FA|SA, B1),
  get_quantileat0(C~FC|SC, C1),get_quantileat1(C~FC|SC, D1),
  % apply the addition over the two uniform distributions
  % the cdf value of (A+C) is Fl
  get_cdf_add([A,B,C,D],[A1,B1,C1,D1],Fl,_Fu1),
  % calculate the slope knowing that the upper bound
```

```
  % has cdf value 1 and lower bound cdf value 0
  % @ A1+C1 cdf is 0 @ B1+D1 cdf is 1
  Sl is (Fl)/((A+C) - (A1+C1)),

  % get the bounds of the cdf uniform distributions
  get_quantileat0(B~FB|SB, A2),get_quantileat1(B~FB|SB, B2),
  get_quantileat0(D~FD|SD, C2),get_quantileat1(D~FD|SD, D2),
  % apply the addition over the uniform distributions
  % the cdf value of (B+D) is Fu
  get_cdf_add([A,B,C,D],[A2,B2,C2,D2],_Fl2,Fu),
  % calculate the slope knowing that the upper bound
  % has cdf value 1 and lower bound cdf value 0
  %% @ B2+D2 cdf is 0 @ B2*D2 cdf is 1
  Su = (Fu - 1)/((B+D) - (B2+D2)).
```

```
% calculate the cdf values resulting from
% adding two uniform distributions
% get_cdf_add(?IntervalBounds,?UniformDistBounds,-Fl,-Fu)
% IntervalBounds is the list of interval bounds
% UniformDistBounds is the list of quantiles
% bounding the two uniform distributions
% and having cdf values 0 and 1
% -Fl lower bound cdf value of the addition
% -Fu upper bound cdf value of the addition
get_cdf_add([A,B,C,D],[A1,B1,C1,D1],Fl,Fu):-
  Zl is A+C, Zu is B+D,
  Fl is 0.5*(A1+C1-A-C)*(A1+C1-A-C)/((B1-A1)*(D1-C1)),
  Fu is 1-(0.5*(B1+D1-B-D)*(B1+D1-B-D)/((B1-A1)*(D1-C1))).
```

Similarily, we have defined in the ‘cdfi_kernel‘ module the *cdf*-intervals arithmetic multiplication, subtraction and division.

```
% ---- cdf-intervals arithmetic multiplication
% arith_multiplication(?VarX,?VarY,-Result)
% ---- cdf-intervals arithmetic subtraction
% arith_subtraction(?VarX,?VarY,-Result)
% ---- cdf-intervals arithmetic division
% arith_division(?VarX,?VarY,-Result)
```

B.3 The solver

The solver aims at creating an expressive development environment for programmers to state their problems intuitively and in a declarative manner. It implements interval propagation techniques used to solve problems over p-box *cdf*-intervals. The key idea is to envelop the true value and its whereabouts within a p-box *cdf*-interval bounded by two triplet points. All arithmetic operations are then performed using these bounds,

hence the resulting interval is widened to take into account any possible solution to the problem in hand.

Interval propagation is implemented by means of the attributed variable data structure together with the suspension handling mechanism. We use both techniques to define a new unification algorithm over p-box *cdf*-intervals which extends the Prolog unification Le Huitouze (1990); Holzbaur (1992).

A *cdf*-interval implementation in an attributed variable data structure consists of: quantiles, *cdf* values and slopes. We append the notation of ‘lb‘ and ‘ub‘ to denote the representation of lower and upper bounds respectively. The constraint suspension mechanism depends on two members of the attributed variable structure: ‘min‘ and ‘max‘. These two elements are the waiting conditions which define the suspension list. When they are triggered, i.e. assigned a wake condition, they do activate[§] the constraint over which the interval variable is defined.

```
% p-box cdf-interval data structure
:- export struct(cdfi(
    qlb, qub, % q for quantile values
    flb, fub, % f for the cdf value
    slb, sub, % s for the slope
    min, % suspensions: wake on update of lo
    max  % suspensions: wake on update of hi
    )).
```

The attributed variable declares: the data structure that defines the p-box *cdf*-interval variable domain, the unification mechanism which extends the Prolog unification algorithm Le Huitouze (1990); Holzbaur (1992), the handler of the suspensions predicate that aims at querying the list of suspensions attached to a variable, the handler of the delayed goals that returns the number of all suspended goals found in this attributed variable, the print predicate which accesses and prints the p-box *cdf*-interval domain, the get bounds predicate that retrieves the triplet points bounding the interval, and the update bounds predicate which updates the bounds of this attributed variable p-box *cdf*-interval domain. Note that calling the update bounds predicate triggers the wake conditions of the suspension list.

```
:- meta_attribute(cdfi, [unify:unify_cdfi/3,
    test_unify:test_unify_cdfi/2,
    suspensions:suspensions_cdfi/3,
    delayed_goals_number:delayed_goals_number_cdfi/2,
    print:print_cdfi/2,
    get_bounds: get_cdfi_bounds/3,
    set_bounds: update_cdfi_bounds/3
    ]).
```

[§]When a constraint is activated, it is moved from the passive list of the active list as detailed in Algorithm 3

B.4 Constraint predicates

`?Var .:: ++Domain` , constrain ‘`Var`‘ to a p-box *cdf*-interval domain, where ‘`++Domain`‘ is defined in terms of its constituents as: ‘`Min~CDFMin|SMin...Max~CDFMax|SMax`‘. If ‘`Var`‘ is already assigned a p-box *cdf*-interval domain, then bounds of ‘`Var`‘ will be updated. Updated bounds are then checked for their consistency to preserve the convex property of the p-box *cdf*-interval domain. In our implementation, we allow the domain to take the ‘untyped‘ bound values: ‘`-1.0Inf`‘ and ‘`+1.0Inf`‘ since quantile bounds are defined in $\mathbb{R}$.

```
% creating a cdfi variable
X .:: Min ~ CDFMin | SMin ... Max ~ CDFMax | SMax :-
   impose_cdfi_bounds(X, Min~CDFMin|SMin, Max~CDFMax|SMax).
```

`?ExprX .=< ?ExprY` , ‘ExprX‘ is less than or equal to ‘ExprY‘ given that each Expr yields ap-box *cdf*-interval domain. Note that this constraint can be equivalently written as ‘`le(?ExprX,?ExprY)`‘.

`?ExprX .= ?ExprY` , ‘`ExprX`‘ is equal to ‘`ExprY`‘ given that each Expr yields a p-box *cdf*-interval domain. Note that this constraint is equivalently written as ‘`eq(?ExprX,?ExprY)`‘.

`?ExprX + ?ExprY` , the addition of two constraint p-box *cdf*-interval domains expressed by ‘`ExprX`‘ and ‘`ExprY`‘.

`?ExprX * ?ExprY` , the multiplication of two constraint p-box *cdf*-interval domains expressed by ‘`ExprX`‘ and ‘`ExprY`‘.

Binary p-box *cdf*-interval constraints are declared in the ‘cdfi‘ module as follows:

```
:- export (.=<)/2, (.=)/2.
% defining the infix representation of constraints
% local is used to override the definition of '+' and '*'
% NB: for the operator definition:
%     A lower priority number indicates a tighter binding
:- local (+)/2, (*)/2, (+)/3, (*)/3.
:- export
  op(400,yfx,'*'),  % tighter than the +
  op(430,yfx,'+'),
  op(700,xfx,'.='),
  op(700,xfx,'.=<').
```

The p-box *cdf*-interval solver is not limited to linear constraints rather it can be used in general problems like ‘`X*X*X + Y*Y .=< Z*Z`‘ where ‘`X`‘, ‘`Y`‘ and ‘`Z`‘ are variables defined over p-box *cdf*-interval domains. The ‘`cdfi_eval`‘ serves at evaluating constraint expressions and follows a decomposition approach to deal with complex constraint format.

```
% cdfi_eval(?ExprX, ?ExprY, -Type)
% evaluating constraint expressions
% for constraint decomposition
% utilized to decompose complex constraint
% expressions into binary ones
% when the expression is an addition constraint
cdfi_eval(CDFI1 + CDFI2, CDFI, _S):-,
  +(CDFI1, CDFI2, CDFI).
% when the expression is a multiplication constraint
cdfi_eval(CDFI1 * CDFI2, CDFI,_S) :-
  *(CDFI1,CDFI2,CDFI).
% when the expression is a constant cdf domain
cdfi_eval(CDFI, CDFI1, 1)?- !,
((is_bounded(CDFI); is_domain(CDFI)) -> true;
  CDFI.:: -1.0Inf~1.0| 1.0Inf... +1.0Inf~0.0|0.0 ),
  (var(CDFI1) ->
    get_cdfi_bounds(CDFI,GlbX~FlbX|SlbX,LubX~FubX|SubX),
    CDFI1.:: GlbX~FlbX|SlbX... LubX~FubX|SubX
  ; true).

cdfi_eval(CDFI1, CDFI, 2)?- !,
((is_bounded(CDFI); is_domain(CDFI)) -> true;
  CDFI.:: -1.0Inf~1.0| 1.0Inf... +1.0Inf~0.0|0.0 ),
  (var(CDFI1) ->
    get_cdfi_bounds(CDFI,GlbX~FlbX|SlbX,LubX~FubX|SubX),
    CDFI1.:: GlbX~FlbX|SlbX... LubX~FubX|SubX
    ; true).

% when both expressions are constant cdf domain
cdfi_eval(GlbX~_FlbX|SlbX...LubX~_FubX|SubX,
 GlbY~_FlbY|SlbY...LubY~_FubY|SubY,3) :-
  GlbX =< GlbY, LubX =< LubY,
  SlbX >= SlbY, SubX >= SubY.
% need an exact match even if they are bounded cdfi
cdfi_eval(CDFI , CDFI,0).
```

We showcase the implementation of the inequality and addition constraints. The rest of the system constraints follow the same behavior. The inequality constraint, when it is applied on two p-box *cdf*-interval variables, it constrains the first variable to take values from the domain that are less than or equal to values in the second variable. This operation follows the p-box *cdf*-interval ordering detailed in Section 6.2.2. As a result the system should maintain the bound-consistency property. To reach the goal ‘`X .=< Y`‘, domains of ‘`X`‘ and ‘`Y`‘ are updated from the upper bound and the lower bound respectively. This constraint is triggered by the suspension list conditions: the ‘`min`‘ and the ‘`max`‘ of the attributed variables ‘`X`‘ and ‘`Y`‘ reciprocally. When ‘`min`‘

of ‘X‘ changes, the domain of ‘Y‘ accordingly should be updated in order to satisfy the ‘X .=< Y‘ condition, i.e. all values in the domain of ‘Y‘ should be greater than those in the domain of ‘X‘ and vice versa. The suspension list, in turn, causes a goal wake up when the upper/lower bound changes. We annotate the goal with a special predicate called the ‘demon‘. Unlike a normal goal, which disappears from the resolvent list once it is woken, a goal declared using a ‘demon‘ annotation remains in the resolvent list when it is woken until it is explicitly killed using the ‘kill_suspension‘ built-in predicate. The constraint solver handles bounded p-box *cdf*-intervals by means of the ‘var_type/3‘ predicate. This type of domain intervals has constant (unchangeable) bounding points.

```
%-------------------------
% the inequality constraint
CDFI1 .=< CDFI2 :-
  le(CDFI1,CDFI2).
%-------------------------
% le constraint
le(X,Y) :-
  var_type(X,Y,S),
  init_domain(X,X1), init_domain(Y,Y1),
  (S = 3 -> SuspList = [X1->inst, Y1->inst] ; true),
  (S = 2 -> SuspList = [X1->inst, Y1->cdfi:max]; true),
  (S = 1 -> SuspList = [Y1->inst, X1->cdfi:min]; true),
  (S = 0 -> SuspList = [X1->cdfi:min, Y1->cdfi:max]; true),

  suspend(le(X1,Y1,MySusp), 0, SuspList, MySusp),
  le(X1,Y1,MySusp),

  (S = 2 ->  cdfi_eval(Y,Y1,2); true),
  (S = 1 ->  cdfi_eval(X,X1,2); true),
  (S = 0 ->  cdfi_eval(X,X1,2), cdfi_eval(Y,Y1,2); true).

:- demon le/3.
le(X,Y,MySusp) :-
  get_cdfi_bounds(X,GlbX~FlbX|SlbX,LubX~FubX|SubX),
  get_cdfi_bounds(Y,GlbY~FlbY|SlbY,LubY~FubY|SubY),
  ( (is_cdfinterval(X), is_cdfinterval(Y)) ->
    true        % implicitly re-suspend
    ;
    kill_suspension(MySusp)
  ),
  glb(LubX~FubX|SubX, LubY~FubY|SubY, LubNew~FubNew|SubNew),
  lub(GlbX~FlbX|SlbX, GlbY~FlbY|SlbY, GlbNew~FlbNew|SlbNew),
  update_cdfi_max(X,LubNew~FubNew|SubNew),
  update_cdfi_min(Y,GlbNew~FlbNew|SlbNew).
```

Similarly, the ternary addition constraint is implemented in our solver over three

p-box *cdf*-interval variables. The implementation covers also the case when one or more variable has constant (unchangeable) bounds. The assignment of the suspension list in this operation differs based on the type of each variable involved. This is followed by the execution of p-box *cdf*-interval addition and subtraction operations as shown by the inference rules detailed in Section 7.2. An intersection operation over each interval variable with corresponding resultants takes place. Then the output of the intersections update the bounds of the variables. Changed domains, in turn, trigger the constraints based on their defined suspension list.

```
%------------------------
% the ternary addition constraint
CDFI1 + CDFI2 :-
  +(CDFI1,CDFI2,_CDFI).
%------------------------
% the addition constraint
addition(X, Y, Z) :-
  var_type(X,Y,S),
  (S = 3 ->  init_domain(X,X1), init_domain(Y,Y1); true),
  (S = 2 ->  init_domain(X,X1), cdfi_eval(Y,Y1,1); true),
  (S = 1 ->  cdfi_eval(X,X1,1), init_domain(Y,Y1); true),
  (S = 0 ->  cdfi_eval(X,X1,1), cdfi_eval(Y,Y1,1); true),
  (is_bounded(Z) -> init_domain(Z,Z1); cdfi_eval(Z,Z1,1)),
  % suspension list
  Sx = [X1->cdfi:min, X1->cdfi:max],
  Sy = [Y1->cdfi:min, Y1->cdfi:max],
  Sz = [Z1->cdfi:min, Z1->cdfi:max],
  (is_bounded(X) -> SuspListx = [X1->inst]; SuspListx0 = [Sz,Sy],
                    flatten(SuspListx0,SuspListx)),
  (is_bounded(Y) -> SuspListy = [Y1->inst]; SuspListy0 = [Sz,Sx],
                    flatten(SuspListy0,SuspListy)),
  (is_bounded(Z) -> SuspListz = [Z1->inst]; SuspListz0 = [Sx,Sy],
                    flatten(SuspListz0,SuspListz)),

  suspend(addz(X1,Y1,Za,MySuspz), 0, SuspListz, MySuspz),
  suspend(addx(Xa,Y1,Z1,MySuspx), 0, SuspListx, MySuspx),
  suspend(addx(Ya,X1,Z1,MySuspy), 0, SuspListy, MySuspy),
  % applying the cdfi addition operation
  addx(Xa,Y1,Z1, MySuspx),
  addx(Ya,X1,Z1, MySuspy),
  addz(X1,Y1,Za, MySuspz),

  cdfi_eval(X1,Xa,2), cdfi_eval(Y1,Ya,2), cdfi_eval(Z1,Za,2)
  cdfi_eval(X,X1,2), cdfi_eval(Y,Y1,2), cdfi_eval(Z,Z1,2).

:- demon addz/4.
addz(X,Y,Z,MySusp) :-
  ( (is_cdfinterval(X), is_cdfinterval(Y)) ->
```

```
        true  % implicitly re-suspend
        ;
        kill_suspension(MySusp)
  ),
  arith_addition(X, Y, Z).

:- demon addx/4.
addx(X,Y,Z,MySusp) :-
  ( (is_cdfinterval(Z), is_cdfinterval(Y)) ->
        true  % implicitly re-suspend
        ;
       kill_suspension(MySusp)
  ),
  arith_subtraction(Y,Z,X).
```

B.5 Examples

In this section we demonstrate how the programmer can input user-defined constraints to the p-box *cdf*-interval solver in an expressive manner. We also show the solver behavior and its output when p-box *cdf*-interval propagation mechanisms are adopted.

```
%-------------------------
%% Intersection
?- X .:: 3.0 ~ 0.8 | 0.9 ... 5.0 ~ 0.5 | 0.06,
   X .:: 2.0 ~ 0.6 | 0.7 ... 4.0 ~ 0.1 | 0.08.
X = X{[(3.0, 0.8, 0.9) ... (4.0, 0.44, 0.06)]}
```

The unification of two p-box *cdf*-interval variables exerts an intersection operation then update the bounds over the two intervals.

```
%-------------------------
%% Unification
?- X .:: 3.0 ~ 0.8 | 0.9 ... 5.0 ~ 0.5 | 0.06,
   Y .:: 2.0 ~ 0.9  |0.3 ... 4.0 ~ 0.3  |0.09, Y = X.
X = X{[(3.0, 0.8, 0.9) ... (4.0, 0.3, 0.09)]}
Y = X{[(3.0, 0.8, 0.9) ... (4.0, 0.3, 0.09)]}
```

The inequality constraint over two p-box *cdf*-intervals ‘X‘ and ‘Y‘, ‘X .=< Y‘ updates the upper bound of ‘X‘ and lower bound of ‘Y‘. The solution obtained has one delayed goal: ‘le(X, Y}, ’SUSP-_1632-susp’)‘ because resulting ‘X‘ and ‘Y‘ have a p-box *cdf*-interval format.

```
%-------------------------
%% Inequality constraint
?- X .:: 2.0 ~ 0.4| 0.8 ... 6.0 ~ 0.2 | 0.05,
   Y .:: 1.0 ~ 0.6 | 0.7 ... 5.0 ~ 0.1 | 0.06 , X .=< Y.
X = X{[(2.0, 0.4, 0.8) ... (5.0, 0.1, 0.06)]}
```

```
Y = Y{[(2.0, 0.4, 0.8) ... (5.0, 0.1, 0.06)]}
There is 1 delayed goal.
```

Consider Example 7.2 which shows the execution of the ternary addition inference rule. Initial bindings to domains of ‘X‘, ‘Y‘ and ‘Z‘ are given in the query along with the ternary addition or ‘X + Y .= Z‘ (shuffling the variable order in the constraint yields the same output solution sets).

```
%-------------------------
%% Ternary addition
?- X.::0.0~0.6|0.099...2.0~0.03|0.01,
   Y.::1.0~0.7|0.098...3.0~0.1|0.04,
   Z.::4.0~0.8|0.05...6.0~0.05|0.008,
   Z.= X+Y.
X = X{[(1.0, 0.56, 0.033) ... (2.0, 0.03, 0.01)]}
Y = Y{[(2.0, 0.6, 0.033) ... (3.0, 0.1, 0.04)]}
Z = Z{[(4.0, 0.8, 0.05) ... (5.0, 0.044, 0.008)]}
```

The ternary multiplication operation ‘Z .= X * Y‘ when initial bindings for
‘X.:: 0.0 ~ 0.6 | 0.099 ... 2.0 ~ 0.03 | 0.01‘,
‘Y .:: -1.0 ~ 0.7 | 0.098 ... 3.0 ~ 0.1 | 0.04‘ and
‘Z .:: -3.0 ~ 0.8 | 0.05 ... 7.0 ~ 0.05 | 0.008‘, yields the following modification in the domains of ‘X‘, ‘Y‘ and ‘Z‘ all together. Final obtained domains after applying this operation are:
‘X.::0.0~0.6|0.099...2.0~0.03|0.01‘,
‘Y.::-1.0~0.7|0.098 ... 3.0~0.1|0.04‘ and
‘Z.::-2.0~0.37|0.012 ... 6.0~0.042|0.008‘

```
%-------------------------
%% Ternary multiplication
?- X .:: 0.0 ~ 0.6 | 0.099 ... 2.0 ~ 0.03 | 0.01,
   Y .:: -1.0 ~ 0.7 | 0.098 ... 3.0 ~ 0.1 | 0.04,
   Z .:: -3.0 ~ 0.8 | 0.05 ... 7.0 ~ 0.05 | 0.008,
   Z .= X * Y.
X = X{[(0.0, 0.6, 0.099) ... (2.0, 0.03, 0.01)]}
Y = Y{[(-1.0, 0.7, 0.098) ... (3.0, 0.1, 0.04)]}
Z = Z{[(-2.0, 0.37, 0.012) ... (6.0, 0.042, 0.008)]}
```

Consider the system of linear equations provided in Example 8.1, one can provide the set of the system constraints to the solver as shown below. Both variables ‘X1‘ and ‘X2‘ in this system lie within domains of positive reals, i.e. $[0, \infty]$. They are both constrained to the domain ‘X1.::0.0~1.0| +1.0Inf ... +1.0Inf~0.0|0.0‘ in order to ensure this fact. The linear inequalities and equalities are intuitively input to the solver using the ‘.=<‘ and ‘.=‘ constraints. The right hand side of the equations are defined over bounded p-box *cdf*-intervals, while the left hand side of the inequalities/equalities are the addition of p-box *cdf*-interval coefficients multiplied by the variables.

```
linear_equations([X1,X2]) :-
  X1.::0.0~1.0| +1.0Inf ... +1.0Inf~0.0|0.0,
  X2.::0.0~1.0| +1.0Inf ... +1.0Inf~0.0|0.0,
  (-2.0~0.5|0.2...2.0~0.01|0.095) * X1 +
  (1.0~0.3|0.32...2.0~0.02|0.083) * X2 .=<
  (3.0~0.88|0.4...4.0~0.04|0.088),

  (-2.0~0.7|0.1... -1.0~0.01|0.087) * X1 +
  (-1.0~0.2|0.3... -1.0~0.01|0.087) * X2 .=
  (-5.0~0.85|0.1... 5.0~0.02|0.013),

  (6.0~0.9|0.98...6.0~0.01|0.018) * X1 +
  (1.5~0.1|0.6...3.0~0.06|0.034) * X2
  .= (4.0~0.9|0.02...15.0~0.01|0.001),

  nl, write('X1 is '), write(X1),
  nl, write('X2 is '), write(X2), nl,
```

The solver prunes the p-box *cdf*-interval domains of '`X1`' and '`X2`' like the ic solver in the $\mathbb{R}$ domain. Additional information about the whereabouts is also propagated in order to elaborate on the data stochastic property. Output domains '`X1`' and '`X2`' are illustrated in Figure 7.5.

```
?- linear_equations([X1, X2]).
X1 = X1{[(0.0, 1.0, 1.0Inf) ... (2.5, 0.28, 0.07)]}
X2 = X2{[(0.0, 1.0, 1.0Inf) ... (5.0, 0.46, 0.07)]}
```