A Proposal for Computing With Imprecise Probabilities: A Framework for Multiple Representations of Uncertainty in Simulation Software

Michael Spiegel

October 29, 2007

Department of Computer Science
Technical Report CS-2007-16
School of Engineering and Applied Science
University of Virginia
151 Engineer's Way, P.O. Box 400740
Charlottesville, VA 22904-4740

A Dissertation Proposal
Presented in Partial Fulfillment
of the Requirement for the Doctor of Philosophy Degree
at the University of Virginia
School of Engineering and Applied Science
Department of Computer Science

Abstract

We propose the design and construction of a programming language for the formal representation of uncertainty in modeling and simulation. Modeling under uncertainty has been of paramount importance in the past half century, as quantitative methods of analysis have been developed to take advantage of computational resources. Simulation is gaining prominence as the proper tool of scientific analysis under circumstances where it is infeasible or impractical to directly study the system in question. This programming language will be built as an extension to the Modelica programming language, which is an acausal object-oriented language for hybrid continuous and discrete-event simulations [22]. Our language extensions will serve as a platform for the research into representation and calibration of imprecise probabilities in quantitative risk analysis simulations. Imprecise probability is used a generic term for any mathematical model which measures chance or uncertainty without crisp numerical probabilities. The explicit representation of imprecise probability theories in a domain-specific programming language will facilitate the development of efficient algorithms for expressing, computing, and calibrating imprecise probability structures. Computation with imprecise probability structures will lead to quantitative risk analyses that are more informative than analyses using traditional probability theory. We have three primary research objectives: (i) the exploration of efficient representational structures and computational algorithms of Dempster-Shafer belief structures; (ii) the application of the imprecise probabilities to representing variable dependence; and (iii) the exploration of various Dempster-Shafer combination rules for model calibration. At the completion of this dissertation, we will have produced the end-to-end design, implementation, and analysis of a programming language that will facilitate the future exploration of algorithms, software, and theory for quantitative uncertainty analysis in computer science.

1 Introduction

We propose the construction of a language for the formal representation of uncertainty in modeling and simulation. This language will be designed explicitly for quantitative risk assessment. Modeling under uncertainty has been of paramount importance in the past half century, as quantitative methods of analysis have been developed to take advantage of computational resources. More specifically, the methods for uncertainty analysis of a numerical simulation brings together the fields of computer science and risk analysis. Risk analysis provides a set of logical, systematic, and well-defined activities for a decisionmaker to identify, measure, quantify, and evaluate the risk associated with a process or phenomenon [27]. Computer science brings a systematic study of the algorithmic processes that describe and transform information [9]. In order to build advanced simulations for risk analysis, we must apply our insights from computer science to create effective, expressive, and efficient computational tools. Simulation is gaining prominence as the proper tool of scientific analysis under circumstances where it is infeasible or impractical to directly study the system in question. Some prominent public policy examples of uncertainty analysis in simulation include the technical studies for the Yucca Mountain nuclear waste repository [52], the assessment reports of the Intergovernmental Panel on Climate Change [35], and the guidelines from the Office of Budget and Management that recommend formal quantitative uncertainty analysis for any federal regulation involving annual economic effects of $1 billion or greater [53].

According to a February 2006 report of the National Science Foundation Blue Ribbon Panel on Simulation - Based Engineering Science (SBES) : "The development of reliable methodologies — algorithms, data acquisition and management procedures, software, and theory — for quantifying uncertainty in computer predictions stands as one of the most important and daunting challenges in advancing SBES." [48] The most basic quantitative uncertainty simulations consist of three primary components: (*i*) the representation of uncertain quantities using stochastics, (*ii*) Monte-Carlo techniques for sampling the stochastic variables in order to generate expected values for the simulation outputs, and (*iii*) calibration techniques for fitting the stochastic variables to the limited available data. The representation, propagation, and calibration of uncertainty are the three necessary steps for any type of uncertainty analysis. The tools that are used to perform these three steps are built from customized software libraries that are largely incompatible [7]. Application libraries provide a weak barrier for the isolation of stochastic variables from other program information. They provide the functionality of the desired domain semantics, but the resulting programs do not offer a strict separation of concerns of the domain semantics from the program-specific semantics [45]. Our research proposal is to build a programming language that will serve as the foundation on which we will explore the algorithms, software, and theory of quantitative uncertainty analysis.

We will design and build a domain-specific programming language for the purpose of uncertainty analysis in modeling and simulation risk assessment studies. This programming language will be built as an extension to the Modelica programming language, which is an acausal object-oriented language for hybrid continuous and discrete-event simulations [22]. Our language extensions will serve as a platform for the research into representation and calibration of imprecise probabilities in quantitative risk analysis simulations. Due to the emphasis on risk analysis in modeling and simulation, we shall refer to our augmented Modelica programming language as the RiskModelica language. RiskModelica will be our testbed for using imprecise probability theories as alternative representations of stochastic variables. Imprecise probability theories allow us to overcome some of the weaknesses of traditional probability theory. They provide the capacity to express additional information to a decision-maker who uses the simulation output in order to perform some type of risk management. Our language design goals and research objectives are discussed in the remainder of this introduction.

RiskModelica will focus on two primary design goals. The first goal is the representation of continuous and discrete random variables as first-class citizens in a programming language. We shall employ multiple mathematical frameworks for the representation of random variables. Each mathematical framework displays a tradeoff of relative expressive power for ease of use. In the next section, we will show how three mathematical frameworks (traditional probability theory, upper and lower probability bounds, and Dempster-Shafer theory) can be appropriate when varying degrees of information are available. Probability theory suffers from three primary weaknesses when representing uncertainty [60]. First, a precise probability value must be assigned to each element in the set of possible outcomes. It may not be possible to assign exact values or even assign reasonable approximations when little information is available. Second, probability theory imposes Laplace's principle of insufficient reason when no information is available. When n mutually exclusive possible outcomes are indistinguishable except for their names, they must each be assigned a probability of $1/n$. And third, conflicting evidence cannot be represented in traditional probability theory. By assigning probabilities to individual elements, we cannot express incompatibility between multiple sources of information or a cooperative effect between multiple sources of information.

The second design goal of RiskModelica will be the capacity to specify calibration techniques for the uncertainty

representations in all three mathematical frameworks. Modeling under uncertainty implies the absence of perfect information, but often partial information exists in the form of observations on the model's expected behavior. Simulation practitioners expect to make the best possible use of the information available to them. Section 2.1.3 motivates the need for a Bayesian inference engine as a part of our uncertainty language. The Bayesian engine will support the calibration of probability theory, probability boxes, and probability mass functions. The inclusion of model calibration techniques is a vital component of a simulation language that intends on making the most out of limited available data.

The RiskModelica language will serve as our primary vehicle for three research objectives. The first research objective will be the creation of efficient algorithms for the representation, manipulation, and calculation of probability distributions, probability boxes, and Dempster-Shafer belief structures. The second research objective will be the application of the imprecise probabilities to random variable dependence. The third research objective will be the successful integration of the Bayesian inference engine of RiskModelica with the Dempster-Shafer representation of uncertainty. We have developed an evaluation plan to assess the success of our research objectives. The evaluation plan is a series of scientific collaborations with investigators from outside computer science. We will study the computational efficiency, computational costs, and knowledge-gained benefits of our novel approach to uncertainty analysis. The plan will also attempt to assess our contributions to quantitative uncertainty analysis as a whole. Our collaboration with external investigators will highlight the relevance of our research contributions to real-world applications in uncertainty analysis.

Our thesis is the following:

> The explicit representation of imprecise probability theories in a domain-specific programming language will facilitate the development of efficient algorithms for expressing, computing, and calibrating imprecise probability structures, for the purpose of conducting quantitative risk analyses that are more informative than analysis using traditional probability theory.

At the completion of this dissertation, we will have produced the end-to-end design, implementation, and analysis of a programming language that will facilitate the future exploration of algorithms, software, and theory for quantitative uncertainty analysis in computer science. These tasks will be accomplished through a series of external collaborations that will create a formal language for expressing quantitative uncertainty using multiple mathematical representations. These collaborations will produce a series of publications that highlight the merits of our approach to propagating uncertainty in simulation software. We will evaluate the merits of our research based on the criteria of Denning et al. [9], which defines our discipline as the systematic study of the algorithmic processes that describe and transform information, in terms of their theory, analysis, design, efficiency, implementation, and application.

2 Related Work

2.1 Imprecise Probability Theories and Model Calibration

Several different mathematical systems can be used to perform uncertainty analysis. We will focus on probability theory, probability boxes, and the Dempster-Shafer theory of evidence. Probability theory is the most traditional representation of uncertainty and the one most familiar to non-mathematicians. The use of probability theory attempts to provide a quantitative analysis to answer the following three questions: (1) what can go wrong, (2) how likely is it that will happen, and (3) if it does happen, what are the consequences? [39] Probability used as a representation of subjective belief is common in quantitative risk analysis. Safety assessments must deal with rare events and thus it is difficult to assess the relative frequencies of these events [1]. The Bayesian approach to uncertainty analysis is to specify a coherent probability measure as the current state of available knowledge, and use Bayes' theorem to adjust probabilities as new evidence is unveiled.

Imprecise probability is a generic term for any mathematical model which measures chance or uncertainty without crisp numerical probabilities. Two types of imprecise probability, probability boxes and Dempster-Shafer belief structures, offer a more flexible representation of uncertainty over the crisp probabilistic approach. According to a study by Ferson et al. [18], these two mathematical techniques provide an approach to several of the most serious problems that can arise during risk analysis, including: (*i*) imprecisely specified distributions; (*ii*) poorly known or even unknown dependencies; (*iii*) non-negligible measurement uncertainty; (*iv*) non-detects or other censoring in measurements; (*v*) small sample size; (*vi*) inconsistency in the quality of input data; (*v*) model uncertainty; and (*vii*) non-stationarity (non-constant distributions).

2.1.1 Probability Boxes

Probability boxes (or p-boxes) define upper and lower boundaries for the probabilities of a set of events [67]. Sometimes we can use these upper and lower boundaries (represented by $\overline{P}(X)$ and $\underline{P}(X)$, respectively) to provide us with additional information that is not available by means of traditional probability theory. The following example will illustrate the potential benefits of probability boxes. Imagine you have found a magical lamp with a genie inside. The genie offers you to play one of two games. In the first game you are given a precision casino six-sided die and ask to roll the die. If you roll a one then the genie will give you \$10,000. In the second game the genie has a stack of six cards that consists of the four aces and two jokers. He presents the stack of cards (face down) and asks you to pick the top card. If the card you select is the ace of hearts, you will win \$10,000. The genie is offering you the choice of playing either one of the two games. Your knowledge of probability informs you that in either case the probability of winning, $P(win)$, is $1/6$. However you feel that somehow the games are not identical to each other. By applying probability boxes, we can deduce that $\underline{P}(win) = 0$ and $\overline{P}(win) = 1$ in the case of the card game. This is because the genie can decide to stack the deck either in your favor or against your favor. Therefore before you measure the event, you can deduce that the genie might know whether or not you will select the ace of hearts. On the other hand in the case of the dice roll game, $\underline{P}(win) = \overline{P}(win) = 1/6$, since there is no additional information available that can be used to categorize the potential states of the system.

A gambler's interpretation of $\underline{P}(X)$ is that it represents the highest price he is willing to pay in order to receive one dollar if X occurs, or receive nothing if X does not occur. Similarly, $\overline{P}(X)$ represents the infimum selling price of an event, which is the lowest price that he is willing to receive in order to sell one dollar if X occurs. Probability boxes are the upper and lower distribution functions ($\underline{F}$ and $\overline{F}$) of an event X where $\underline{F}(x) = \underline{P}(X \leq x)$ and $\overline{F}(x) = \overline{P}(X \leq x)$. Upper and lower distribution functions allow an analyst to make no assumptions about the shape of the true probability distribution function. A series of coherency axioms ensure that $\underline{F}(x) \leq F(x) \leq \overline{F}(x)$ for all real numbers x. Probability boxes enable some separation of epistemic uncertainty and aleatory uncertainty [15, 16]. Under classical probability theory, the principle of indifference dictates one should select a uniform distribution when presented with a lack of information concerning the shape of that distribution. Traditional probabilistic analysis ignores the epistemic uncertainty of the model and thus can result in misleading risk assessment calculations.

Regan et al. [58] looked at EPA calculations for Ecological Soil Screening Levels (Eco-SSLs) in Superfund ecological risk assessments. Eco-SSLs are concentrations of contaminants in soil that are protective of ecological receptors that commonly come into contact with soil or ingest biota that live in or on soil. Eco-SSLs are intentionally conservative in order to provide confidence that contaminants which could present an unacceptable risk are not screened out early in the risk assessment process [66]. The study compared deterministic calculations of Eco-SSLs for the meadow vole (*Microtus pennsylvanicus*) and the northern short-tailed shrew (*Blarina brevicauda*) with a Monte Carlo approach and a probability bounds approach. The results show that Eco-SSL estimates using conservative deterministic methods were greater than estimates using probability bounds methods by approximately 2-3 orders of magnitude. Median-based deterministic calculations resulted in Eco-SSL estimates approximately one order of magnitude greater than conservative deterministic methods. The Monte Carlo simulation fails to produce a conservative estimate due to a combination of assumptions about dependencies between variables, and assumptions about the shape of the probability distribution curves. In the opinion of the authors: "We believe that probability bounds analysis is most useful as a tool for identifying the extent of uncertainty in model application and can assist in reducing this uncertainty."

2.1.2 Dempster-Shafer Theory of Evidence

In the Dempster-Shafer theory of evidence, the concept of imprecise probabilities is extended to account for both nonspecificity and discord of available evidence [61, 41]. Probability boxes account for nonspecificity by propagating lower and upper bounds without specifying the shape of the distribution. But probability boxes require that all the available evidence concludes in one non-overlapping interval. Dempster-Shafer theory allows a decision-maker to reason about several candidate probability intervals for a random process, even when the candidate intervals conflict with one other. Dempster-Shafer theory is formulated in terms of a function known as the basic probability assignment. If Ω is the set of all possible outcomes under consideration and 2^Ω is the power set of Ω (the set of all subsets of Ω), then a basic probability assignment is defined as $m(A) : 2^\Omega \rightarrow [0, 1]$ such that: $m(\emptyset) = 0$ and $\sum_{A \in 2^\Omega} m(A) = 1$.

Evidence theory is a generalization of traditional probability theory where probabilities are assigned to sets of events instead of individual events. A degree of belief, Bel, given to the set A is defined as the sum of all masses that support A: $Bel(A) = \sum_{B \subset A} m(B)$. The degree of plausibility, Pl, quantifies the total amount of belief that might support A: $Pl(A) = Bel(\Omega) - Bel(\overline{A}) = \sum_{B \cap A \neq \emptyset} m(B)$ [62]. The belief and plausibility functions can be viewed as lower and upper bounds of probability in the sense that $Bel(A) \leq P(A) \leq Pl(A)$. It is often the case that some

subsets of Ω will have basic probability assignments of zero. Those sets with nonzero assignments are called the focal elements of m. The set of all focal elements is known as a body of evidence.

Helton et al. [31] present a Dempster-Shafer risk analysis of a hypothetical safety system that is exposed to fire. The safety system consists of one weak link (WL) component and one strong link (SL) component that are both exposed to thermal heating. Both components will ultimately fail at sufficiently high temperatures. The weak link component is designed to fail safe during accidents and render the system inoperational. The strong link component is designed to be robust and resistant to extreme environments. Risk analysis is performed to assess the likelihood that the WL component will fail before the SL component. A time-dependent thermal response curve is used to model the high temperature scenario. The model contains 11 uncertain parameters such as initial temperatures, maximum temperatures, thermal constants, frequency responses, and expected values and standard deviations of normal distributions.

Dempster-Shafer theory enables the expression of several forms of partial information concerning the uncertain parameters. For example, the peak amplitude of the WL temperature transient (T) has been measured experimentally in laboratory environments. Three measurement techniques have resulted in three different recorded intervals for the parameter: $T_1 = -500 \pm 40°\text{C}$, $T_2 = -1000 \pm 60°\text{C}$, $T_3 = -1800 \pm 80°\text{C}$. All three sources are considered equally credible, and yet the intervals give conflicting information. This can be expressed by assigning basic probability assignments of $m(T_1) = m(T_2) = m(T_3) = 1/3$. Evidence theory can also express nested probability structures. The thermal heating time constant (H) of the WL temperature transient is expressed with the following confidence intervals: $H_1 = 0.27 \leq H \leq 0.30\ \text{min}^{-1}$ with 30% confidence, $H_2 = 0.25 \leq H \leq 0.35\ \text{min}^{-1}$ with 50% confidence, and $H_3 = 0.20 \leq H \leq 0.40\ \text{min}^{-1}$ with 100% confidence. H_1, H_2, and H_3 are nested intervals that can be interpreted as plausibility measurements on H. Calculating backwards from the plausibility measurements yields the basic probability assignments $m(H_1) = 0.3$, $m(H_2) = 0.2$, and $m(H_3) = 0.5$.

Computing belief and plausibility measures have been shown to be #P-complete problems [54]. The complexity class #P contains the counting problems which are associated with decision problems that can be solved in polynomial time on a non-deterministic Turing machine (NP problems). It is believed that there does not exist a polynomial-time algorithm for solving #P-complete problems, since this would imply P = NP. As with many suprapolynomial computational problems, a research emphasis has been placed on devising polynomial-time approximation algorithms to computing belief and plausibility measures. Joslyn and Kreinovich [38] describe a Monte Carlo method for approximating belief and plausibility that can produce good estimates when the conflict between evidence is not very high. Dempster-Shafer theory provides a method of reasoning under imprecisely defined sources of information. The belief function is a measure of the extent to which available evidence implies that something is true. And plausibility is a measure of the extent to which evidence implies that something might be true.

2.1.3 Bayesian Inference

Bayesian inference is a technique for updating *a priori* probability distributions based on the observation of data that are relevant to the variables of interest [43]. Let us describe a generic application of Bayesian inference by assuming a model $\bar{y} = f(\bar{x}, \bar{\theta})$, where $\bar{x}$ represents a vector of input quantities to the model, $\bar{\theta}$ represents a vector of scalar parameters to the model, and $\bar{y}$ represents a vector of observable output quantities. The *a priori* beliefs about $\bar{\theta}$ are expressed using a probability distribution function $p(\bar{\theta})$. On the basis of independent observations $X = (X_1, X_2, ..., X_n)$, the probability distribution function is then updated according to Bayes' Law: $p(\bar{\theta}|X) = \frac{p(\bar{\theta})p(X|\bar{\theta})}{p(X)} = \frac{p(\bar{\theta})p(X|\bar{\theta})}{\int p(\bar{\theta})p(X|\bar{\theta})d\bar{\theta}}$.

The *posterior* distribution can be approximated numerically using Monte Carlo integration techniques. Traditional Monte Carlo algorithms perform a series of independent samples from a probability distribution to approximate the desired integral. In order to numerically calculate high-dimensional integrals on large parameter spaces, $\bar{\theta} = (\theta_1, \theta_2, ..., \theta_m)$, a Markov chain Monte Carlo (MCMC) integration technique can be used. MCMC integration techniques are often more efficient than traditional Monte Carlo techniques in high-dimensional parameter spaces.

For example, the Gibbs sampler is a widely used MCMC method for Bayesian inference [23]. In the Gibbs sampler each of the parameters is expressed as a probability conditioned on all of the other parameters. Next, a Markov chain is produced that cycles through all of the conditional probabilities. To perform the Gibbs sampler, start with estimates for $\bar{\theta}^{[0]} = (\theta_1^{[0]}, \theta_2^{[0]}, ..., \theta_m^{[0]})$. Next, begin a cycle by drawing values for each parameter θ_k according to the distribution $\theta_k \sim p(\theta_k|\bar{\theta}_{-k}, X)$, where $\bar{\theta}_{-k}$ indicates the set of parameters without the θ_k coefficient, and X is the set of independent observations. Repeat cycling through the parameter distributions until the process converges on a set $\bar{\theta}^{[n]}$. Any problem that can be expressed in terms of $p(\theta_k|\bar{\theta}_{-k}, X)$ can be solved using the Gibbs sampler.

2.2 Probabilistic Programming Language

The dissertation work of Sungwoo Park of Carnegie Mellon University describes the design and implementation of PTP, a ProbabilisTic Programming language [55]. PTP is an extension of λ-calculus that uses sampling functions to specify probability distributions . A sampling function takes as input an infinite sequence of random numbers drawn independently from $U(0.0; 1.0]$, consumes zero or more random numbers, and returns a sample with the remaining sequence. There are three primary advantages to using sampling functions as the mathematical basis for probabilistic programming. First, the use of sampling functions provides a unified representation scheme for discrete distributions, continuous distributions, and distributions belonging to neither group. Second, probability distributions can be expressed over a rich domain, including infinite discrete domains, continuous domains, and even unusual domains such as infinite data structures and cyclic domains. Third, there can be more than one way to specify a probability distribution, and the more we know about it, the better we can encode it.

PTP has been applied to three applications in robotics: robot localization, people tracking, and robotic mapping [56]. Robot localization is the problem of estimating the position and orientation of a robot when a map of the environment is available. People tracking is an extension of robot localization that estimates the position of unidentified objects on the environment map. Robotic mapping is the problem of building a map of the environment from sensor readings. All three applications use Bayes filters to update the estimates of unknown state information. A Bayes filter estimates the state s of a system from a sequence of actions and measurements, where an action a induces a change to the state and a measurement m gives information on the state.

PTP provides language primitives for discrete probability distributions, continuous probability distributions, the expectation operator of some function with respect to a distributions, and the Bayes operator applied to two probability distributions. The probabilistic language is based on an augmented lambda calculus whose mathematical basis is sampling functions. Park et al. [56] claim that PTP is the only probabilistic language with a formal semantics that has been applied to practical problems involving continuous distributions.

2.3 Interval Analysis

Interval analysis is a branch of mathematical analysis that operates on closed connected sets of the real numbers. Interval analysis became a topic of interest for computational analysis in the 1950s and 1960s, when it became obvious that numerical imprecision of floating point computation could lead to misleading program output in certain instances [20]. Interval computation is application of interval mathematics to floating-point computation. Interval computation has been implemented in several language compilers [64, 65] and language libraries [40, 44, 6]. Using interval arithmetic results in a computation that satisfies certain correctness criterion at the expense of output precision. Correctness is guaranteed by the fundamental theorem of interval arithmetic, which states that an arithmetic expression evaluated using intervals computes to a resulting interval that contains all the results of the original expression when evaluated using the points in the input intervals. Precision is sacrificed because of the fundamental theorem of interval arithmetic. Interval computation contains all possible floating-point evaluations of the points along an interval, including those combinations where rounding errors combine in an unfavorable way.

Modern implementations of interval computation can take advantage of the IEEE 754 floating point standard to enforce the fundamental theorem of interval arithmetic. The directed rounding modes allow for a simple, portable, and efficient implementation of the basic arithmetic operations for interval computation [8]. It is possible to use the IEEE 754 standard and implement interval computation such that the properties of correctness, totality, closedness, optimality, and efficiency are preserved [32].

Interval computation is a deterministic representation of uncertainty that preserves arithmetic correctness (the fundamental theorem of interval arithmetic) at the expense of precision. As we discussed in Section 3.1, interval computation can be expressed in Dempster-Shafer theory by assigning a probability of 1.0 to the desired interval. In fact, Monte Carlo sampling algorithms are currently used in ensemble meteorological forecast simulation to evaluate interval computation. Ensemble forecasting techniques simulate many different weather models in parallel, with slightly different initial conditions or model configurations [24]. The initial conditions are sampled from candidate intervals that are representative of current weather conditions. The initial conditions are considered deterministic in nature despite the use of sampling techniques to perform the analysis. We will have support for interval computation through the employment of Dempster-Shafer arithmetic when the probability of the desired interval is assigned all the weight of the probability mass function.

2.4 Modelica

Modelica is an object-oriented equation-based programming language. It is designed for large, complex and heterogeneous physical systems. Modelica programs are declarative, mathematical systems of equations that specify acausal relationships among state variables. Acausal programming is a programming paradigm in which program data flow is not explicitly represented. The primary operator in acausal programming is the equality operator. In traditional imperative programming the primary operator is the assignment operator which has defined inputs, the right-hand side, and outputs, the left-hand side. The equality operator does not express neither input nor output. Instead it states that two expressions containing one or more variables are equivalent. Acausal programming allows for the expression of higher-order mathematical properties to be observed and preserved. For example, acausal equality statements obey the transitive property, such that $(a = b) \wedge (b = c) \rightarrow (a = c)$, but imperative assignment statements do not obey this property, $(a := b) \wedge (b := c) \nrightarrow (a := c)$. The acausal programming paradigm is most closely related to constraint-based programming techniques. However constraint programming techniques have historically focused on either finite domains or integer and rational domains [36].

The Modelica language allows for the expression of a system of equations, and then the Modelica compiler will translate this system of equations into a traditional imperative C program. It is the purpose of the Modelica compiler to determine the appropriate data flow and control flow that will solve the system of equations. An application developer can express the problem as a system of equations and allow the programming tools to produce an executable program that will solve the equations.

Modelica allows for the expression of differential algebraic equations (DAEs) that are commonly found in the modeling of physical systems. A differential algebraic equation is a system of equations that includes both differential equations and algebraic equations. The general form of a DAE is $f(\frac{dx}{dt}, x, y, t) = 0$, where x is a vector of variables in $\Re^n$ for which derivatives are present, y is a vector of variables in $\Re^m$ for which no derivatives are present, and t is a scalar that represents the independent variable time. Differential algebraic equations arise in Newtonian mechanics, where constraints are coming from the fact that particles are not completely free to move with respect to each other. Examples of these multibody dynamics are a double pendulum or the movement of a robot arm. Another typical class of problems comes from electrical circuits, where the Kirchhoff equations again lead to a DAE with voltages and currents as unknowns.

Modelica also allows for the expression of hybrid discrete event and continuous systems. Hybrid differential algebraic equations allow for the expression of discontinuous changes in system state. Modelica allows us to express models that combine discrete event behavior and continuous behavior. Many real-world problems behave continuously until some threshold is crossed, at which point a sharp discontinuity in system state occurs.

DAEs are solved numerically by transforming the system of equations into a set of differential, algebraic, and discrete equations. The differential equations can be solved by explicit integration methods, and these solutions can be used to solve the algebraic equations. The main difference between numerical integration of ODEs and DAEs is that solving the DAE may include differentiation. This is because algebraic equations may express constraints in variables that are not directly expressed in their derivatives. A reduction technique is used to simplify the DAE system into a set of ODEs, which can be solved traditionally. Numerical integration is used during regions of continuous change. When a discrete event is triggered, the integration scheme is halted and a set of algebraic equations are solved for the current time instant. Only algebraic equations must be solved during event calculation, as all differential equations are eliminated in a time interval with zero duration. After the event has been processed, the numerical integration scheme must be restarted and continues until the next event is generated.

3 Research Objectives and Plan

We have identified three research objectives that will constitute novel work in the field of computer science. Our first research objective comes from the design decisions that will be made in the construction of RiskModelica. The objective is the creation of efficient algorithms for the representation, manipulation, and calculation of Dempster-Shafer belief functions. As we have discussed in section 2.1.2, Dempster-Shafer theory assigns probability to subsets of all possible outcomes. Assume we are assigning probability to a distribution with possible outcomes from the real numbers, then we could have a Dempster-Shafer belief function with a domain of cardinality $\aleph_2$. That belief function could not be represented by traditional sampling-based methods using a transformation on the uniform distribution. Although it is unclear whether such a large belief function would ever be used in practice, the efficient expression and manipulation of (smaller) Dempster-Shafer structures remains an open research topic. Section 3.1 presents a survey of the different types of Dempster-Shafer belief structures that are used in risk analysis studies.

The second research objective is the application of probability boxes to representing dependence and correla-

tion. Variable dependence has been known to have substantial influence on quantitative risk assessments in some instances. The most common way of specifying variable dependence uses a scalar value, a correlation metric, that can be misleading. The accurate modeling of random variable dependence has a substantial influence on quantitative risk assessments. A correlation metric does not fully specify the true dependence relationship between two or more random variables. The correct relationship is specified using a copula, and the boundaries of a copula can be expressed using probability boxes. We will define the copula function in section 3.2, and show that probability boxes can be used to express the boundaries of any arbitrary copula function.

The final research objective investigates the various Dempster-Shafer combination rules for the application of model calibration. Dempster-Shafer theory contains multiple possible rules in which evidence from various sources can be combined. The flexibility of Dempster-Shafer theory arises from a choice on the assignment of conflicting evidence. By assigning probabilities to sets of possible outcomes, evidence theory can express incompatibility between multiple sources of information or a cooperative effect between multiple sources of information. The evidence combination rules we shall consider occupy a continuum between conjunction (AND-based on set intersection) and disjunction (OR-based on set union) [11]. A conjunctive combination is appropriate in circumstances where all sources are considered reliable. A disjunctive combination is appropriate when there is only one reliable source among many sources. Section 3.3 presents some of the more common combination rules and compares their strengths and weaknesses.

3.1 Representation of Belief Structures

Dempster-Shafer belief structures are represented using one of two main types of data structures, depending on the cardinality of the set of possible outcomes. One representation scheme is reserved for finite sets while the other representation scheme is used for infinite sets. For a finite set S of possible outcomes with cardinality N, there are at most 2^N unique basic probability assignments. We have found that rarely is a full set of 2^N unique basic probability assignments used in practice. Within the research literature there exists four common subclasses of basic probability assignments for finite sets [5, 42]. These subclasses are enumerated as follows:

1. The trivial case of total ignorance where $m(S) = 1$ and $m(X) = 0$ if-and-only-if $X \neq S$. This is highlighted as a more accurate representation of total ignorance when compared to traditional probability theory, which must apply Laplace's principle of indifference in these circumstances. In this instance, there is exactly one non-zero probability mass assignment used in the belief structure.

2. Every assignment in the belief structure is made to a singleton of the set S. As a consequence there must be exactly N assignments made within this belief structure. Additionally, $Bel()$ and $Pl()$ are equal to each other for all subsets of S and they correspond to a traditional probability measure on the set S.

3. Every assignment in a belief structure is made to a nested set of assignments. In other words, for every two assignments X and Y such that $m(X) > 0$ and $m(Y) > 0$, then $X \subset Y$ or $Y \subset X$. This structure has the property that $Bel()$ is also a necessity measure on S, which has the property that $Bel(X \cap Y) = min\{Bel(X), Bel(Y)\}$. It is also clear that there are can be at most N assignments made within this belief structure. This arrangement of nested assignments in known as possibility theory.

4. Every assignment in a belief structure is made to a discrete contiguous frame (d.c.f) of elements.
   If $\{a_1, a_2, ..., a_N\} \subseteq S$ is an ordered set of elements that is used a contiguous frame, then for each pair of consecutive elements (a_i, a_{i+1}) in the set, there does not exist another element $a_j \in S$ such that $a_i < a_j < a_{i+1}$. This organization is represented graphically in Figure 1 in a representation that is attributed to Strat [63]. The figure illustrates how each contiguous frame is recorded in one of the entries of the upper-diagonal square. In the figure we have highlighted one sample contiguous frame that corresponds to the interval $\{a_i, a_{i+1}, a_{i+2}, ..., a_j\}$. It is evident that this representation uses at most $O(N^2)$ basic probability assignments. The motivation for discrete contiguous frames is that it is common in practical applications, when one has a sequence of possible outcomes that has been ordered, $\{a_1, a_2, ..., a_N\}$ and there exists information concerning a subset of these outcomes, $\{a_1, ..., a_4\}$, is will be unusual that this information does not also concerns the intermediate elements a_2 and a_3.

In our survey of real-world applications of Dempster-Shafer theory to infinite sets, we have found that only one type of belief structure is currently used. This structure is a continuous contiguous frame (c.c.f) of elements and it is the infinite analog to the discrete contiguous frame. The contiguous frame of elements assigns basic probability statements to the closed intervals $[u, v]$ as shown in Figure 2. Unlike the d.c.f., the c.c.f is not required to have lower

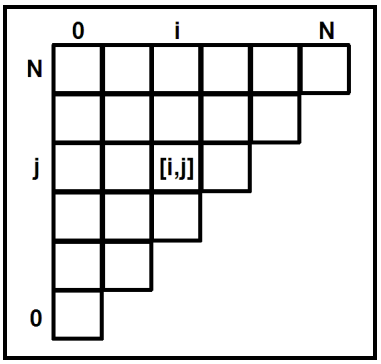

Figure 1: Discrete contiguous frames

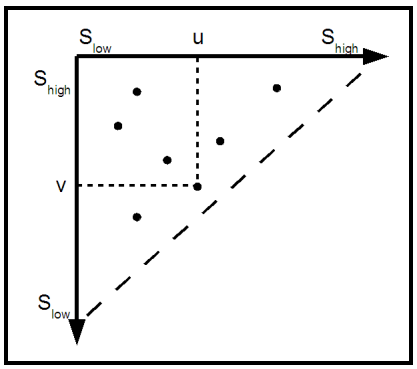

Figure 2: Continuous contiguous frames with a finite point set

and upper bounds that restrict that values of u and v. However, in our literature survey we have found that c.c.f.'s were restricted to a finite number of basic probability assignments. In the case of finite basic probability assignments, we can extract S_{low} and S_{high} which correspond to the lowest and highest elements of S used in an interval assignment. The Helton et al. [31] study of Dempster-Shafer risk analysis that was presented in Section 2.1.2 is an application of continuous contiguous frames. One subclass of continuous contiguous frames is the deterministic set of interval assignments. In the deterministic case, one interval receives the total basic probability assignment, ie. $P([u, v]) = 1$ and $P(X) = 0$ if $X \neq [u, v]$. Thus, it is possible to use Dempster-Shafer theory to represent a deterministic interval computation. Interval analysis will be discussed under related work in section 2.3.

3.2 Dependence and Correlation

As an application of alternative representations of uncertainty, we highlight the use of probability boxes to express variable correlation and dependence. Two or more random variables are dependent when knowledge of the value of one variable gives information on the value of the other variables. More formally, two variables are independent if and only if their joint distribution function is the product of their marginal distributions. If two variables are not independent then they experience some type of dependence. The accurate modeling of random variable dependence has a substantial influence on quantitative risk assessments [33]. Haas [25] cites several examples of correlated input random variables in ecological modeling, including age and weight [4], age and water consumption [59], and levels of dioxin and furan congeners in emissions [28]. The tail probabilities of output distributions are more likely to be sensitive to variable dependencies, which can result in the probability of extreme events being underestimated [14].

The dependence between two random variables can be measured using a number of correlation coefficients. The two most popular correlation coefficients are Spearman's correlation coefficient ρ and Kendall's correlation coefficient τ. Correlation coefficients do not contain enough information to specify the dependence relationship between two random variables. In order to specify the dependence a mathematical function known as the copula is necessary. However a correlation coefficient is often used as an approximation for the dependence relation. The popular software packages Crystal Ball and @Risk use an algorithm developed by Iman and Conover [34] to produce correlated random variables with any set of marginal distributions and a specified value of τ.

3.2.1 Copulas

Copulas are functions that join multivariate distribution functions to their one-dimensional marginal distribution functions [50]. A bivariate copula is defined as a function from $[0, 1] \times [0, 1] \rightarrow [0, 1]$, such that both marginal distributions are uniform over the interval $[0, 1]$. If X and Y are random variables with continuous marginal distribution functions $F(x)$ and $G(y)$, respectively, and (X, Y) has the joint distribution function $H(x, y)$, then there exists a unique copula $C(u, v)$ such that $H(x, y) = C(F(x), G(y))$. The copula contains all of the information on the dependence between the two random variables, but gives no information on the marginal distributions.

One approach for estimating the dependence between random variables is to make no assumptions about shape of the copula function [19]. In this case it is known only that the copula must lie within the Fréchet-Hoeffding boundaries, which are the upper and lower probability distributions for dependence relations: $W(u, v) \leq C(u, v) \leq M(u, v)$. Using the Fréchet-Hoeffding boundaries, we can leverage our implementation of probability boxes from Section 2.1.1 to relax the assumption that all random variables are independent. When we relax this assumption and make no

assumptions about the nature of the dependence relation, the probability distributions are transformed into probability boxes using the Fréchet-Hoeffding boundaries. The dependence relations $M(u, v)$ and $W(u, v)$ are known as perfect dependence and opposite dependence, respectively. Perfect dependence occurs whenever one of the random variables is completely eclipsed by the other variable. It is characterized by the joint distribution function $M(u, v) = min(u, v)$. Opposite dependence occurs whenever the random variables occupy the smallest possible area of overlap. Opposite dependence is characterized by the joint distribution function $W(u, v) = max(0, u + v - 1)$.

One problem associated with propagating the Fréchet-Hoeffding boundaries is that the resulting probability boxes encompass large probability spaces. A solution to this problem is to use partially specified information to approximate the dependence between random variables [17]. This arises in circumstances when there is not enough information to completely specify the desired copula, but there is some available information to select a subset of the Fréchet-Hoeffding boundaries. A common example is the knowledge that two random variables experience positive quadrant dependence, which means that $H(x, y) \geq F(x)G(y)$ for all x and y. Positive quadrant dependence means that the joint probability that the variables are simultaneously small (or large) is at least as great as the joint probability as if the variables were independent. Positive quadrant dependence implies non-negative Spearman and Kendall correlations. It allows us to tighten the lower boundary of the copula to $\Pi(u, v) \leq C(u, v) \leq M(u, v)$, where $\Pi(u, v) = uv$.

3.3 Combination of Evidence in Dempster-Shafer Theory

Dempster-Shafer theory contains multiple possible rules in which evidence from various sources can be combined. The flexibility of Dempster-Shafer theory allows for the variety of combination rules, in contrast with traditional probability theory. Bayes's theorem proves that there exists one unique combination method of combining evidence in traditional probability theory. The flexibility of Dempster-Shafer theory arises from a choice on the assignment of conflicting evidence. Sentz and Ferson [60] provide an excellent survey of current combination rules in Dempster-Shafer theory and discuss the appropriateness of the rules depending on the context of the problem. We shall summarize three of these rules, Dempster's rule, Yager's rule, and Dubois and Prade's disjunctive rule, and observe how conflicting evidence can be either (a) completely ignored, (b) assigned to the universal set of evidence, or (c) taken into consideration using consensus, respectively.

Dempster's rule of combination was developed as an integral part of of the original Dempster-Shafer theory of evidence. The rule combines multiple belief functions through their basic probability assignments. The joint basic probability assignment is calculated by aggregating information over all non-conflicting evidence sources: $m_{12}(A) = \sum_{B \cap C = A} m_1(B)m_2(C)/(1 - K)$, where $K = \sum_{B \cap C = \emptyset} m_1(B)m_2(C)$. K represents basic probability mass associated with conflict. It is an aggregation of the bpa's of all probability assignments where the intersection is null. The joint basic probability assignment is calculated by throwing out all of the conflicting evidence. K is used in the equation for m_{12} to normalize the joint bpa so that the sum over the elements of the power set is 1. The elimination of conflicting evidence can lead to counterintuitive results, as illustrated in this example by Lotfi Zadeh [70]. Imagine a patient suffering from neurological symptoms goes to visit two doctors. The first doctor believes that either the patient has meningitis with odds 99 to 1 or the patient has a brain tumor. The second doctor believes the patient has a concussion with odds 99 to 1 or the patient has a brain tumor. Dempster's rule states that the joint probability mass function $m_{12}(\text{brain tumor}) = 1$. This implies complete endorsement of a joint diagnosis that both of the doctors considered to be unlikely, but it was the only diagnosis upon which both doctors agreed.

Yager's rule is a modification of Dempster's rule (it is also known as the modified Dempster's rule) that assigns all conflicting evidence to the universal set. The probability assignment to the universal set is interpreted as a measure of ignorance among the different sources of evidence [69]. Yager introduces the ground probability mass assignment (q) which behaves very similarly to the basic probability mass assignment. The ground probability assignment is calculated using an unnormalized Dempster rule: $q(A) = \sum_{B \cap C = A} m_1(B)m_2(C)$. The ground probability assignment for the null set is assigned to the normalizing constant K, $q(\emptyset) = \sum_{B \cap C = \emptyset} m_1(B)m_2(C)$. Then a series of operations is applied to transform ground probability assignments to basic probability assignments. For the universal set, an arbitrary set, and the empty set, these operations are $m^Y(X) = q(X) + q(\emptyset)$, $m^Y(A) = q(A)$, and $m^Y(\emptyset) = 0$, respectively. The notation m^Y denotes basic probability assignments associated with Yager's rule. Conflicting evidence is preserved using Yager's transformation, and all conflicting evidence is attributed the universal set which serves as an unbiased source for depositing conflicting evidence.

Dubois and Prade's disjunctive rule is based on set union, in contrast to the previous rules that were based on set intersection [10]. The union of basic probability assignments is defined as $m_\cup(A) = \sum_{B \cup C = A} m_1(B)m_2(C)$. The union does not generate any conflicting evidence and it does not reject any of the information from the individual basic probability assignments. The usefulness of the disjunctive rule is questionable, as it assumes that only one of the independent information sources is reliable and it is unknown which is the reliable source. The conjunctive rules

assume that all of the information sources are reliable, which is a stronger constraint on the joint basic probability assignment. Dubois and Prade's rule can produce a more imprecise joint basic probability assignment than desirable.

3.4 Research Plan

Our research plan consists of the sequence of actions that must be completed in order to fulfill the three research objectives that were explained in sections 3.1, 3.2, and 3.3. To meet our research objectives we have two different kinds of tasks that must be accomplished. These two tasks are (a) the design and implementation of the RiskModelica infrastructure, and (b) the application of the infrastructure to accomplish our objectives. These two tasks will be interleaved, based on the understanding that orthogonal components of RiskModelica can be designed independently. A detailed analysis of the RiskModelica design and implementation is presented in the next section. The initial steps of our research plan will follow the PTP language design from Carnegie Mellon and use sampling expressions to express probability distribution functions. Next, we shall extend this work by implementing the Bayesian component of our project to allow for model calibration techniques using traditional probability theory. The exact types of calibration techniques will be dependent on our research collaborators. At this point we have built a framework for expressing and calibrating uncertainty for traditional probability theory. After this foundation has been constructed we will design and implement the expression of imprecise probability structures for the propagation and calibration of uncertainty. We have identified two projects that employ imprecise probability structures. One project is the employment of probability boxes to express random variable dependence as an improvement over traditional methods of correlation coefficients. The other project is the calibration of Dempster-Shafer belief structures and the selection of an appropriate evidence combination law to cope with conflicting evidence. We have laid out a series of three projects in this section that each fulfill one of our three research objectives. The expected challenges and expected contributions of our research are in sections 5 and 6. Our measures of success are written to answer the following overarching question: does the additional computation required for propagating imprecise probability theories provide enough novel information to a decision-maker? Do the benefits of the novel analysis outweigh the costs of extended computing? The related work we have cited suggests that the answer is in the affirmative [18, 25, 31, 58]. While the benefits of the new analysis are in a subjective domain, we can and will quantify the costs of the analysis in our evaluation plan that is discussed in section 3.6.

3.5 Implementation Plan

3.5.1 Language Design

Our uncertainty framework will be designed as a series of language extensions to the Modelica programming language. These language extensions will facilitate the two primary design goals listed in the previous section. To distinguish between the host language (Modelica) and the new language features, we will refer to the new language as RiskModelica. RiskModelica will contain new data types and new operators for these types to manipulate random variables in all three frameworks of imprecise probabilities. The basic support for probability density functions will use a calculus of sampling functions [55]. The flexibility of sampling functions as a representation scheme for probability density functions has been show in the PTP language. In order to construct sampling functions, we introduce the sampling expression S. S consumes a random number from an infinite sequence drawn independently from $U(0.0, 1.0]$, and S evaluates to the random number that has been consumed. All functions in Modelica are pure functions, which means that evaluation of the function does not produce any side effects. Sampling functions in RiskModelica will retain the property of pure functions, with the exception that sampling functions may draw an arbitrary number of times from the sampling expression S. A sampling function will behave as a pure function if the sampling expression S is reset and recycled across multiple executions. The RiskModelica compiler will ensure that the sampling expression is invoked only from sampling functions.

Whereas a sampling function is sufficient to represent a probability density function, a pair of sampling functions can be used to represent a probability box, such that $\underline{F}(x) \leq F(x) \leq \overline{F}(x)$ for all x. RiskModelica will need specialized data types in order to represent the various types of basic probability assignments used in Dempster-Shafer theory (see section 3.1). Once probability boxes have been added to RiskModelica, this will allow for the expression of copulas using the Fréchet-Hoeffding boundaries (see section 3.2).

In addition to the representation of random variables in all three frameworks of imprecise probabilities, we will also add language features for specifying calibration techniques in all three frameworks. As discussed in section 2.1.3, Bayesian techniques allow the combination of prior probabilities with calibration data to create a posterior probability model. RiskModelica will include support for specifying which resources to use as calibration data (ie. file or I/O streams), and which algorithm to use for computing Bayes' Law. MCMC techniques will be used to calibrate

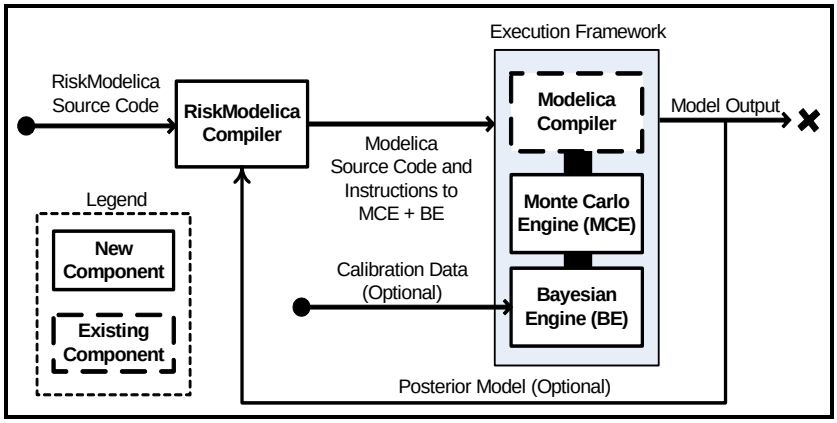

Figure 3: RiskModelica compiler infrastucture

probability density functions and probability boxes. These techniques rely on the specification of the conditional probability distributions on the parameters of interest. The calibration of probability belief structures using variations of Bayes' Theorem in Dempster-Shafer theory is one of our research objectives. As we build and test appropriate combination rules (see Section 3.3), these rules shall become a part of the calibration framework in RiskModelica.

RiskModelica will be designed as a language extension to Modelica by introducing novel primitive types and type qualifiers to the Modelica language. A type qualifier is a form of subtyping where a supertype `T` is combined with a qualifier `Q` such that some semantic property is enforced on all instances of the subtype `Q T` [21]. Some canonical examples of type qualifiers are `const` and `volatile` from the ANSI C standard. A variable declared with the type qualifier `const` can be initialized but it cannot be modified. The type qualifier `volatile` means that the value of the instantiated variable can change asynchronously of the current execution thread. Type qualifiers can be used as a mechanism for implementing a variety of compile-time and run-time semantic properties. One popular example is Splint, the static analysis tool of Evans and Larochelle, that uses program annotations to implement type qualifiers that can check for memory usage errors at compile-time [13]. Another example is the CCured translator, built by Necula et al., that extends C's type system with pointer qualifiers [49]. Pointer qualifiers are type qualifiers that modify pointer types. CCured uses a combination of static analysis and run-time checking to add memory safety guarantees to C programs.

RiskModelica will extend the rich type qualifier system available in the Modelica language. Modelica contains twelve type qualifiers, preserving semantic properties at compile-time such as encapsulation [`public`/`private`], causality [`input`/`output`], and variability [`constant`/`parameter`/`discrete`], etc. RiskModelica will extend Modelica as necessary to support (*i*) probability bounds analysis, (*ii*) Dempster-Shafer belief structures, (*iii*) Bayesian inference, and (*iv*) model calibration in Dempster-Shafer theory. RiskModelica will maintain the type properties of Modelica in order for the new language features to integrate well with the host language. Modelica is statically typed, explicitly typed, and type safe for a subset of the language [2].

Type qualifiers will allow the necessary semantic properties for stochastic computation to be enforced, and type qualifiers allow greater flexibility of static analysis techniques. Static analysis has already been applied in Modelica to detect overconstrained or underconstrained systems of equations in models [3]. We will apply static analysis techniques as much as possible to enforce the semantic properties of our novel type qualifiers. Wherever it is impossible to use static analysis, we will revert to runtime checking to enforce the required properties. The success in implementing the type properties as compiler transformation techniques, and thus avoiding costly runtime analysis, is one of the performance criteria for our evaluation plan. As future work, RiskModelica programs could be checked by novel static analysis tools for detecting desirable properties of stochastic computation.

3.5.2 Compiler Infrastructure

The RiskModelica language will be implemented as a superset of the Modelica language, and we will create a RiskModelica compiler to translate RiskModelica into an executable program. The output of our RiskModelica compiler will be a Modelica program, along with special instructions for the RiskModelica execution framework (see Figure 3). The execution framework consists of the Modelica compiler, the Monte Carlo engine, and the Bayesian engine. Therefore it is the function of the RiskModelica compiler to translate RiskModelica-specific language constructs, and pass along the remaining Modelica constructs downstream to the Modelica compiler. The RiskModelica compiler

will function primarily as a compiler frontend, as an intermediate language representation will not be generated and code optimization will not be performed. The RiskModelica compiler will build from the grammar specification that is available from the OpenModelica project, which uses ANTLR [57], a predicated $LL(k)$ parser generator, to construct the Modelica parser. As to the Modelica compiler, we shall use either the open source OpenModelica compiler designed by Programming Environment Laboratory of Peter Fritzson [22], or the commercial Dymola compiler produced by Dynasim AB of Sweden [12]. The Modelica compiler translates the Modelica source into C code which is subsequently compiled and executed.

The remaining units of the RiskModelica execution framework, the Monte Carlo engine and the Bayesian engine, are responsible for directing execution flow in order to analyze the uncertainty information that has been encoded in the original RiskModelica program. The Monte Carlo engine and the Bayesian engine receive instructions from the RiskModelica compiler, and use these directions to control the execution of the compiled Modelica program. The Monte Carlo (MC) engine will be responsible for setting up multiple iterations of the model, and consequently the MC engine will be responsible for managing the sampling expression S and whatever sampling constructs will be created for probability belief structures. The Bayesian engine will be responsible for applying Bayes' Law or the variations of Dempster's rule of combination to calibrate RiskModelica models. When the Bayesian engine has been engaged, the output of the RiskModelica execution framework will not consist of model output. Instead the Bayesian engine will output a calibrated RiskModelica program which can be subsequently compiled and executed at a later time. The Bayesian engine thus operates as a one-time iterative example of self-modifying code. The Bayesian engine has only iteration by default, as the output of the the engine is a calibrated program that has turned off Bayesian inference, although this inference could be manually turned back on when additional calibration data became available.

3.6 Evaluation Plan

Our planned evaluation will measure the degree of success to which we have addressed our thesis statement and the three research objectives. The evaluation plan serves to bind the research objectives into a series of published studies that will show the merits of our contributions. When completed, RiskModelica will have the capacity to represent, propagate, and calibrate stochastic variables in three mathematical frameworks. We will conduct a series of comparative studies between stochastic simulations using traditional probability theory and stochastic simulations using imprecise probability theories. In the comparative studies we can study the benefits of imprecise probability analysis and highlight what additional information is provided to a decision-maker by using imprecise probability theories. The relative costs of imprecise probability analysis will be evaluated by comparing the slowdown in execution time to traditional probabilistic analysis. In addition, the overhead of the RiskModelica compiler will be studied when a preexisting stochastic simulation is available. As such, a three-way program comparison can be created: (*i*) a stochastic simulation written using pre-existing program languages, (*ii*) quantitative uncertainty analysis of a simulation in RiskModelica using probability theory, and (*iii*) quantitative uncertainty analysis of a simulation in RiskModelica using imprecise probability theories. In addition to execution time in the three-way comparison, other metrics will be used to assess their differences, such as memory footprint, algorithmic complexity, and compilation time.

We expect to publish at least three comparative studies, one study per research objective identified in this section. The first study will focus on using probability boxes and Dempster-Shafer theory to propagate random variables in simulation software. The second study will use probability boxes to propagate the Fréchet-Hoeffding boundaries of dependence relations. And the third study will explore how Bayesian inference can be applied calibrating Dempster-Shafer belief structures. For each study we will work with an external collaborator on a problem within their application domain. The previous efforts of our research group in developing modeling and simulation technologies has provided us with contacts to the departments of environmental science, materials science, mechanical and aerospace engineering, physics, and neurological surgery at the University of Virginia. Our goal is to use these contacts to find existing stochastic simulations that can be extended using the novel methods of RiskModelica. The underlying support of RiskModelica to represent, propagate, and calibrate stochastic variables using (precise) probability theory is not a significant research accomplishment, but this support allows us to benchmark RiskModelica against pre-existing simulations that also have these abilities.

We will consider RiskModelica a success if our compilation infrastructure (see section 3.5.2) runs in a modest completion time and memory footprint, and the generated simulation code from the RiskModelica $\rightarrow$ Modelica $\rightarrow$ C compiler chain performs comparably to pre-existing simulations. The measure of success against pre-existing simulation code will be the comparative resource requirements of the two programs while performing a Monte Carlo program sampling and Markov Chain Monte Carlo program calibration. The success of our contributions to quantitative uncertainty analysis will be contingent upon the benefits of our new insights into the behavior of a stochastic system

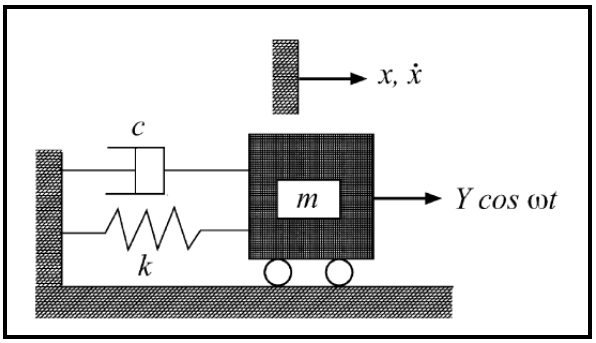

Figure 4: Mass-spring-damper system. Figure 2 of Oberkampf et al. [51].

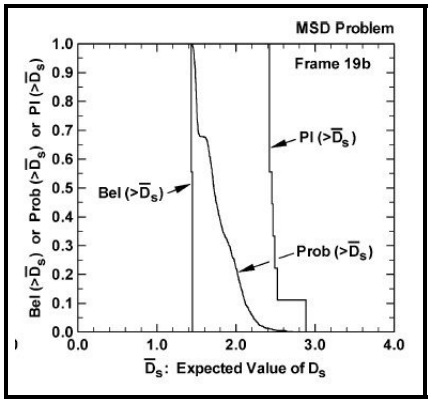

Figure 5: Complementary cumulative distribution functions of $\overline{D}_s$. Figure 19b of Helton et al. [30].

versus the additional resource requirements of these new techniques. We will compare the resource requirements of our implementation in RiskModelica against the theoretical algorithmic complexity of the underlying algorithms that will be developed. One of the fundamental goals for the development of RiskModelica is the requirement that sampling algorithms and evidence combination rules remain computationally modest as functions of the sample size used in Monte Carlo calculations, the number of entries in the Dempster-Shafer belief structures, and the size of the dataset for model calibration.

4 Preliminary Work

In this section, we present an uncertainty analysis using imprecise probabilities for a system of differential equations. The purpose of this analysis is to illustrate the merits of using imprecise probabilities in simulation software. The calculations presented in this section did not use a general framework for imprecise probabilities; instead the computation was hand-coded for this particular example. As preliminary work, this exercise is intended to show the proof of concept of computing with imprecise probabilities in dynamic systems. We have chosen to extend one of the challenge problems presented in a workshop on uncertainty representation at Sandia National Laboratories in the summer of 2002, with a summary of the workshop published by Oberkampf et al. [51] in 2004. The workshop created seven time-independent input/output relationships and discussed various types of generalized uncertainty analyses of these models. One study, published by Helton et al. [30], was an analysis of these models using probability theory, possibility theory (see section 3.1), and evidence theory. The seventh model of this workshop was inspired by a dynamic system known as the "Mass-spring-damper" (MSD) system. In the Mass-spring-damper system, a mass is attached to a cart on wheels, which is attached to a wall with both a spring and a linear damper. A periodic external force is applied to the cart (see figure 4). The behavior of the mass-spring-damper system is characterized by the equation [Eqn 4.1] $m\ddot{x} + c\dot{x} + kx = Y\cos\omega t$. The original challenge problem identifies a static property of this system known as the steady-state magnification factor, D_s, which is defined as the ratio of the amplitude of the steady-state response to the system to the static displacement of the system induced by a force of magnitude Y: [Eqn 4.2] $D_s = \frac{k}{\sqrt{(k-m\omega^2)^2+(c\omega)^2}}$. We have extended the challenge problem to include the complementary cumulative probability, belief, and plausibility distribution functions of the position of the mass over time. We have calculated the position distribution functions of the mass using both numerical integration techniques, and using an analytical derivation of the position of the mass in order to verify the numerical integration scheme.

The parameters of the mass-spring-damper system are represented as a combination of aleatory uncertainty and epistemic uncertainty. The parameters $\{m, k, \omega\}$ are assumed to be of an aleatory character and they are represented by traditional probability distributions. k and ω are represented by triangular distributions, and the input parameters to these triangular distributions are assumed to be of an epistemic nature. The parameters $\{k_{min}, k_{mod}, k_{max}, c, Y, \omega_{min}, \omega_{mod}, \omega_{max}\}$ are represented by Dempster-Shafer belief structures using continuous intervals, where α_{min}, α_{mod}, α_{max} are the three input parameters to a triangular distribution for some parameter α. The parameters k and ω are represented using a two-tier system of uncertainty, using belief-structures as input parameters to the probability distributions of these parameters. The belief and plausibility measures for this example each consist of a probability space. In order to represent these probability spaces, Helton et al. [30] have chosen to plot the mean value of the D_s distributions. Complementary cumulative probability functions, belief functions, and plausibility functions for $\overline{D}_s$

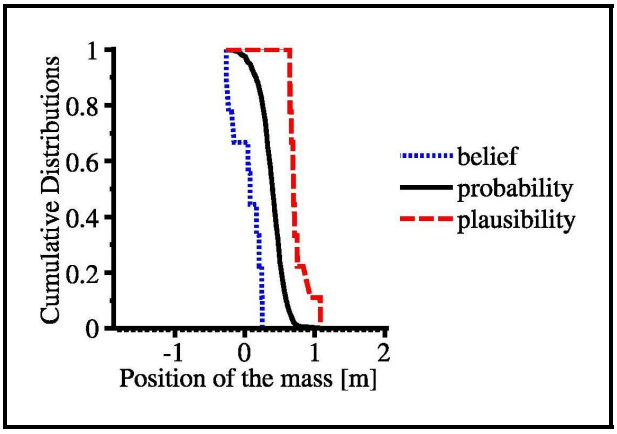

Figure 6: Position at time t = 3 seconds.

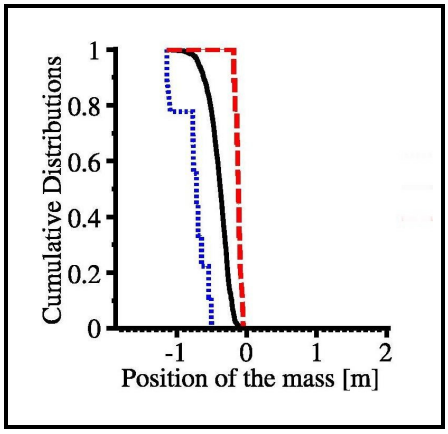

Figure 7: Position at time t = 6 seconds.

have been calculated and are shown in figure 5.

Plausibility and belief for Ds and $x(t)$ are approximated by making the observation that $Pl_y(B) = Pl_z(f^{-1}(B))$ and $Bel_y(B) = Bel_z(f^{-1}(B))$ where $y = f(z)$ is a functional transformation from z to y [37]. $f(z)$ is [Eqn 4.1] for determining $x(t) = f(m, k, \omega, c, Y, t)$ and $f(z)$is [Eqn 4.2] for determining $Ds = f(m, k, \omega, c)$. $Pl_y(B)$ and $Bel_y(B)$ are approximated by sampling the input space $\overline{S}(X) = x_i \in X, 1 \leq i \leq n$ and $y_i = f(x_i)$ and then defining the sampled belief and plausibility measures as $\hat{Pl}_y(B) = Pl_x\big(\hat{f}^{-1}\big(\overline{S}(B)\big)\big)$ and $\hat{Bel}_y(B) = 1 - \hat{Pl}_y(B)$. It can be shown that the sampled belief and plausibility measures converge to their true values in the limit: $\lim_{n\to\infty} \hat{Pl}_y(B) = Pl_y(B)$ and $\lim_{n\to\infty} \hat{Bel}_y(B) = Bel_y(B)$. This type of sampling is referred to as a "black box" sampling approach because no additional information is available about the transformation function $f(z)$. The flexibility of the black box approach allows this technique to be applied identically to both Ds and $x(t)$.

Figure 5 shows the computation of complementary cumulative belief (CCBF) and plausibility functions (CCPF) for Ds as calculated by Helton et al. [30]. A complementary cumulative probability density function (CCDF) has also been calculated, operating under the assumption that the evidence intervals specified in the belief structures are interpreted as uniform probability distributions. We have extended this work to calculate CCBF, CCDF, and CCPF for the position of the mass, $x(t)$, at time slices of $t = 0.05$ seconds over an interval of 30 seconds. We begin at time $t = 0$ when the object is at rest. Figures 6 and 7 show the movement of the mass over time $t = 3$ and $t = 6$ seconds, respectively. An animated short film showing the dynamic behavior of the probability distribution functions is available online[1]. In this film, we can observe the initial oscillation of the probability measures as the object is placed into motion, followed by a steady-state behavior of the CCBF, CCDF, and CCPF functions following the initial transient period. Thus our preliminary work has shown that it is possible to propagate plausability and belief measures through a dynamic simulation over time using the same black box sampling approaches that have been applied to static models.

5 Expected Challenges

- Exact computation of plausibility and belief of Dempster-Shafer belief structures are known to be #P-complete problems [54]. Efficient approximation algorithms are necessary in order to make reasonable approximations in polynomial time. Several methods have been proposed, such as latin hypercube sampling [29], importance sampling [47], Markov Chain Monte Carlo [46, 68], and resource bounded anytime-algorithms [26]. I will explore the various trade-offs of these approximation algorithms and identify those that will meet the needs of our application domains during the course of our research.
- The explicit representation of random variable dependence in RiskModelica is a desired language feature, but it is a non-trivial design question as to how this dependence should be expressed. I will explore several possible design alternatives and study their merits on the basis of correctness, completeness, and compactness before this language feature can be implemented.
- A suite of algorithms have been developed for Bayesian inference, such as unbiased Monte Carlo sampling,

[1] http://www.cs.virginia.edu/~ms6ep/mastri/preliminary-work/movies/evidence_1x.avi

importance sampling, Gibbs sampling, the Metropolis-Hastings algorithm, etc. I will explore the translation of these methods into to an effective implementation of the Dempster-Shafer combination laws. It is unclear whether any of the three common combination laws are computationally faster than the other combination laws when applied to Bayesian inference problems.

6 Expected Contributions

- A formal programming framework that will support the representation, propagation, and calibration of uncertainty using probability theory, probability boxes, and Dempster-Shafer's theory of evidence.
- The development of reliable methodologies — algorithms, data acquisition and management procedures, software, and theory — for quantifying uncertainty in computer predictions (from the NSF Blue Ribbon Panel on Simulation-Based Engineering Science [48]).
- The creation of efficient algorithms for the representation, manipulation, and calculation of Dempster-Shafer belief functions. And the application of probability boxes to more accurately express random variable dependence than current single-valued correlation metrics.
- The creation of efficient algorithms for applying Dempster-Shafer combination rules. An investigation of which rules are appropriate when calibrating belief structures given multiple sources of information that are possibly conflicting.
- An evaluation plan that will measure the benefits of additional information that is conveyed in an analysis that uses imprecise probability theories, versus the costs associated with additional computation.

7 Conclusion

We have proposed the construction of a language for the formal representation of uncertainty in modeling and simulation. The explicit representation of imprecise probability theories in a domain-specific programming language will facilitate the development of efficient algorithms for expressing, computing, and calibrating imprecise probability structures, for the purpose of conducting quantitative risk analyses that are more informative than analysis using traditional probability theory. Our preliminary work has shown that it is possible to extend current techniques of evaluating imprecise probability distributions into the realm of dynamic simulations using a Monte Carlo sampling-based approach. We have three primary research objectives: (*i*) the exploration of efficient representational structures and computational algorithms of Dempster-Shafer belief structures; (*ii*) the application of the imprecise probabilities to representing variable dependence; and (*iii*) the exploration of various Dempster-Shafer combination rules for model calibration. We have developed an evaluation plan that will measure the degree of success to which we have addressed our thesis statement and three research objectives. The evaluation plan will study the computational efficiency, computational costs, and knowledge-gained benefits of our novel approach to uncertainty analysis. It will also assess our contributions to quantitative uncertainty analysis as a whole.

We shall target three potential avenues of publication to disseminate the results of this research. The first set includes journals and conferences in the modeling and simulation domain, such as *ACM Transactions on Modeling and Computer Simulation (ACM TOMACS), ACM Transactions on Mathematical Software (ACM TOMS), Simulation: Transactions of The Society for Modeling and Simulation International, Proceedings of the International Modelica Conference, Proceedings of International Workshop on Principles of Advanced and Distributed Simulation (PADS)*, and *Proceedings of the Winter Simulation Conference (WSC)*. The second set includes avenues from the field of quantitative risk analysis, such as *Risk Analysis, Reliability Engineering and System Safety,* and *Reliable Computing.* And our final set outlets for research dissemination will be the conferences and journals of our external collaborators in their application domain.

Modeling under uncertainty has been of paramount importance in the public and private sector in the past half century, as quantitative methods of analysis have been developed to take advantage of computational resources. Simulation is gaining prominence as the proper tool of scientific analysis under circumstances where it is infeasible or impractical to directly study the system in question. At the completion of this dissertation, we will have produced the end-to-end design, implementation, and analysis of a programming language that will facilitate the future exploration of algorithms, software, and theory for quantitative uncertainty analysis in computer science. In the systematic study of the algorithmic processes that describe and transform information, RiskModelica will bring the tools to evaluate and solve problems under the conditions of limited available information. As computational tools are being increasingly used to study problems of limited information, we will serve these needs by providing an underlying infrastructure that formally represents, transforms, and calibrates uncertainty for information processing.

References

[1] George Apostolakis. The concept of probability in safety assessment of technological systems. *Science*, 250: 1359–1364, December 1990.

[2] David Broman, Peter Fritzson, and Sebastien Furic. Types in the modelica language. In *Fifth International Modelica Conference*, 2006.

[3] Peter Bunus and Peter Fritzson. Automated static analysis of equation-based components. *SIMULATION*, 80 (7-8):321–345, July/August 2004.

[4] David E. Burmaster and E. A. Crouch. Lognormal distributions for body weight as a function of age for males and females in the united states, 1976-1980. *Risk Analysis*, 17:499–505, 1997.

[5] William F. Caselton and Wuben Luo. Decision making with imprecise probabilities: Dempster-shafer theory and application. *Water Resources Research*, 28(12):3071–3083, 1992.

[6] CGAL. *Computational Geometry Algorithms Library User and Reference Manual*. CGAL Editorial Board, 2006.

[7] Barry A. Cipra. Revealing uncertainties in computer models. *Science*, 287(5455):960–961, February 2000.

[8] Michael Clemmesen. Interval arithmetic implementations using floating point arithmetic. *ACM SIGNUM Newsletter*, 19(4):2–8, October 1984.

[9] Peter J. Denning, D. E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. Computing as a discipline. *Communications of the ACM*, 32(1):9–23, January 1989.

[10] Didier Dubois and Henri Prade. A set-theoretic view on belief functions: Logical operations and approximations by fuzzy sets. *International Journal of General Systems*, 12:193–226, 1986.

[11] Didier Dubois and Henri Prade. On the combination of evidence in various mathematical frameworks. In J. Flamm and T. Luisi, editors, *Reliability Data Collection and Analysis*, pages 213–241, Brussels, 1992.

[12] Dynasim. Dymola (dynamic modeling laboratory) version 6.0b. Ideon Research Park, SE-223 70 Lund, Sweden, 2006. Available at `www.dynasim.se`.

[13] David Evans and David Larochelle. Improving security using extensible lightweight static analysis. *IEEE Software*, 19(1):42–51, January/February 2002.

[14] S. Ferson and M. Burgman. Correlations, dependency bounds and extinction risks. *Biological Conservation*, 73: 101–105, 1995.

[15] Scott Ferson and Lev R. Ginzburg. Different methods are needed to propagate ignorance and variability. *Reliability Engineering and System Safety*, 54:133–144, 1996.

[16] Scott Ferson and Janos G. Hajagos. Arithmetic with uncertain numbers: rigorous and (often) best possible answers. *Reliability Engineering and System Safety*, 85:135–152, 2004.

[17] Scott Ferson and Janos G. Hajagos. Varying correlation coefficients can underestimate uncertainty in probabilistic models. *Reliability Engineering & System Safety*, 91:1461–1467, 2006.

[18] Scott Ferson, Vladik Kreinovich, Lev Ginzburg, Davis S. Myers, and Kari Sentz. Constructing probability boxes and dempster-shafer structures. Sandia National Laboratories, Albuquerque, New Mexico, January 2003. SAND2002-4015.

[19] Scott Ferson, Roger B. Nelsen, Janos Hajagos, Daniel J. Berleant, Jianzhong Zhang, W. Troy Tucker, Lev R. Ginzburg, and William L. Oberkampf. Dependence in probabilistic modeling, dempster-shafer theory, and probability bounds analysis. Sandia Report SAND2004-3072, Sandia National Laboratories. Alburquerque, New Mexico., 2004.

[20] George E. Forsythe. Pitfalls in computation, or why a math book isn't enough. *The American Mathematical Monthly*, 77(9):931–956, November 1970.

[21] Jeffrey S. Foster, Manuel Fahndrich, and Alexander Aiken. A theory of type qualifiers. In *Proceedings of SIGPLAN '99 (PLDI)*, 1999.

[22] Peter Fritzson. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. IEEE Press and Wiley-Interscience, 2004.

[23] Jeff Gill. *Bayesian Methods: A Social and Behavioral Sciences Approach*. Chapman & Hall/CRC, 2002.

[24] Tilmann Gneiting and Adrian E. Raftery. Weather forecasting with ensemble methods. *Science*, 310:248–249, 2005.

[25] Charles N. Haas. On modeling correlated random variables in risk assessment. *Risk Analysis*, 19(6):1205–1213, 1999.

[26] Rolf Haenni and Norbert Lehmann. Resource bounded and anytime approximation of belief function computations. *International Journal of Approximate Reasoning*, 31(1-2):103–154, October 2002.

[27] Yacov Y. Haimes. *Risk Modeling, Assessment, and Management*. Wiley-Interscience, second edition edition, 2004.

[28] D. Hattis and D. E. Burnmaster. Assessment of variability and uncertainty distributions for practical risk analysis. *Risk Analysis*, 14:713–730, 1994.

[29] J.C. Helton, J.D. Johnson, W.L. Oberkampf, and C.B. Storlie. A sampling-based computational strategy for the representation of epistemic uncertainty in model predictions with evidence theory. Sandia National Laboratories, Alburquerque, New Mexico, 2006.

[30] Jon C. Helton, Jay D. Johnson, and William L. Oberkampf. An exploration of alternative approaches to the representation of uncertainty in model predictions. *Reliability Engineering & System Safety*, 85:39–71, 2004.

[31] Jon C. Helton, William L. Oberkampf, and Jay D. Johnson. Competing failure risk analysis using evidence theory. *Risk Analysis*, 25(4):973–995, 2005.

[32] T. Hickey, Q. Ju, and M.H. Van Emden. Interval arithmetic from principles to implementation. *Journal of the ACM*, 48(5):2001, September 2001.

[33] T.P. Hutchinson and C.D. Lai. *Continuous Bivariate Distributions, Emphasising Applications*. Rumsby Scientific Publishing, Adelaide, South Australia, 1990.

[34] Ronald L. Iman and W. J. Conover. A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics-Simulation and Computation*, 11(3):311–334, 1982.

[35] IPCC. Intergovernmental panel on climate change fourth assessment report. Intergovernmental Panel on Climate Change, 2007.

[36] J. Jaffar and M. J. Maher. Constraint logic programming: A survey. *Journal of Logical Programming*, 19/20: 503–581, 1994.

[37] Cliff Joslyn and Jon C. Helton. Bounds on belief and plausibility of functionally propagated random sets. In *2002 Conference of the North American Fuzzy Information Processing Society ([NAFIPS] 02)*, 2002.

[38] Cliff Joslyn and Vladik Kreinovich. Convergence properties of an interval probabilistic approach to system reliability estimation. Technical report, Los Alamos National Laboratory, 2002.

[39] Stanley Kaplan and B. John Garrick. On the quantitative definition of risk. *Risk Analysis*, 1(1):11–27, 1981.

[40] R. Baker Kearfott. Algorithm 763: Interval arithmetic: a fortran 90 module for an interval data type. *ACM Transactions on Mathematical Software*, 22(4):385–392, December 1996.

[41] G. J. Klir. Measures of uncertainty in the dempster-shafer theory of evidence. In Ronald R. Yager, Mario Fedrizzi, and Janusz Kacprzyk, editors, *Advances in the Dempster-Shafer theory of evidence*. Wiley, 1994.

[42] Ivan Kramosil. *Probabilistic Analysis of Belief Functions*. Kluwer Academic/Plenum Publishers, New York, New York, 2001.

[43] Peter M. Lee. *Bayesian Statistics: An Introduction*. Hodder Arnold, third edition edition, 2004.

[44] Guillaume Melquiond, Sylvain Pion, and Hervé Brönnimann. *Boost C++ Interval Arithmetic Library*, 2004.

[45] Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344, 2005.

[46] Serafin Moral and Nic Wilson. Markov chain monte-carlo algorithms for the calculation of dempster-shafer belief. In *Proceedings of the twelfth national conference on Artificial Intelligence*, 1994.

[47] Serafin Moral and Nic Wilson. Importance sampling monte-carlo algorithms for the calculation of dempster-shafer belief. In *Proceedings of the IMPU'96*, 1996.

[48] National Science Foundation. Simulation-based engineering science : Revolutionizing engineering science through simulation. Report of the NSF Blue Ribbon Panel on Simulation-Based Engineering Science, February 2006.

[49] George C. Necula, Jeremy Condit, Matthew Harren, Scott McPeak, and Westley Weimer. CCured: Type-safe retrofitting of legacy software. *ACM Transactions on Programming Languages and Systems*, 27(3):477–526, 2005.

[50] Roger B. Nelsen. *An Introduction to Copulas*. Springer Series in Statistics. Springer, second edition edition, 2006.

[51] William L. Oberkampf, Jon C. Helton, Cliff A. Joslyn, Steven F. Wojtkiewicz, and Scott Ferson. Challenge problems: uncertainty in system response given uncertain parameters. *Reliability Engineering & System Safety*, 85:11–19, 2004.

[52] OCRWM. Yucca mountain science and engineering report REV 1. DOE/RW-0539-1. U.S. Department of Energy, Office of Civilian Radioactive Waste Management, Las Vegas, Nevada, 2002.

[53] OMB. Circular a-4, regulatory analysis. Office of Management and Budget, September 2003.

[54] Pekka Orponen. Dempster's rule of combination is #P-complete. *Artificial Intelligence*, 44(1-2):245–253, 1990.

[55] Sungwoo Park. *A Programming Language for Probabilistic Computation*. PhD thesis, School of Computer Science. Carnegie Mellon University., Pittsburgh, PA, August 2005.

[56] Sungwoo Park, Frank Pfenning, and Sebastian Thrun. A probabilistic language based on sampling functions. *ACM Transactions on Programming Languages and Systems*, V(N):1–45, August 2006.

[57] T. J. Parr and R. W. Quong. ANTLR: A predicated-LL(k) parser generator. *Software-Practice and Experience*, 25(7):789–810, July 1995.

[58] Helen M. Regan, Brad E. Sample, and Scott Ferson. Comparison of deterministic and probabilistic calculation of ecological soil screening levels. *Environmental Toxicoloy and Chemistry*, 21(4):882–890, 2002.

[59] A. M. Roseberry and D. E. Burnmaster. Log-normal distributions for water intake by children and adults. *Risk Analysis*, 12:99–104, 1992.

[60] Kari Sentz and Scott Ferson. Combination of evidence in dempster-shafer theory. Sandia National Laboratories, 2002.

[61] Glenn Shafer. *A mathematical theory of evidence*. Princeton University Press, 1976.

[62] Phillippe Smets. What is dempster-shafer's model? In Ronald R. Yager, Mario Fedrizzi, and Janusz Kacprzyk, editors, *Advances in the Dempster-Shafer theory of evidence*. Wiley, 1994.

[63] T.M. Strat. Continuous belief functions for evidential reasoning. In *Fourth National Conference on Artificial Intelligence*, pages 308–313, Austin, Texas, 1984. American Association for Artificial Intelligence.

[64] Sun Microsystems. *Interval Arithmetic in Forte Fortran Technical White Paper*. Palo Alto, California, 2000.

[65] Sun Microsystems. *Interval Arithmetic in High Performance Technical Computing*. Santa Clara, CA, September 2002. urlwww.sun.com/processors/whitepapers.

[66] U.S. EPA. Guidance for developing ecological soil screening levels. Office of Solid Waste and Emergency Response, Washington, DC, November 2003. OSWER Directive 9285.7-55.

[67] Peter Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, 1991.

[68] Nic Wilson and Serafin Moral. Fast markov chain algorithms for calculating dempster-shafer belief. In *Proceedings of the European Conference on Artificial Intelligence*, 1997.

[69] Ronald R. Yager. On the dempster-shafer framework and new combination rules. *Information Sciences*, 41: 93–137, 1987.

[70] Lofti Zadeh. Review of books: A mathematical theory of evidence. *The AI Magazine*, 5(3):81–83, 1984.