Interval Methods for Fault-Tree Analyses in Robotics

Carlos Carreras, *Member IEEE*
Universidad Politécnica de Madrid, Madrid

Ian D. Walker, *Member IEEE*
Clemson University, Clemson

Key Words - Interval methods, Reliability estimation, Fault trees, Robotics.

Summary & Conclusions - This paper describes a novel technique, based on interval methods, for the estimation of reliability using fault trees. The approach encodes inherent uncertainty in the input data by modeling this data in terms of intervals. Appropriate interval arithmetic is then used to propagate the data through standard fault trees to generate output distributions which reflect the uncertainty in the input data. Through a canonical example of reliability estimation for a robot manipulator system, we show how the use of our novel interval method significantly improves the accuracy of reliability estimates over existing approaches to the problem of uncertain input data. We also show that the method avoids the key problem of loss of uncertainty inherent in some previously suggested approaches when applied to non-coherent systems. We further discuss how the method has advantages over approaches based on partial simulation of the input data space as it can provide guaranteed bounds for the estimates in reasonable times.

1. INTRODUCTION

Acronyms and Abbreviations

MCS	Monte Carlo Sampling
LHS	Latin Hypercube Sampling

In the last few years, a key driver for the development and application of new technologies in the field of robotics has been the need to send robots into hazardous and remote environments (space, nuclear, undersea, etc.), where a failure in the robot can have disastrous consequences and repair is often impossible. This has motivated intensive research in the area of fault tolerance (fault detection and isolation) and reliability (modeling and analysis) of robots [30], the latter being the topic of this paper.

The primary tool used for reliability analysis in robotics has been Fault Tree Analysis [30]. Fault Trees [10, 29] support the encoding of fault information from widely disparate subsystems into a single formal framework. With strong motivation and funding coming from the nuclear and hazardous waste clean-up communities in both the USA [26] and Europe [15], numerous Fault Tree studies have been performed over the last few years to identify and analyze key failure paths within different types of robots [12, 15, 30, 32, 31]. These studies have almost exclusively been limited to qualitative analyses (cut set generation, identification of uncovered failure modes, etc.) due, in part, to the difficulty of obtaining good input probability data (from mean time to failure or failure rates of subsystems and components) for quantitative analysis, as robots of interest are typically ‘one of a kind’, or ‘several of a kind’ devices. Thus quantitative estimates have been largely untrusted, which in turn has led to overconservative designs, or even the non-deployment of some systems [32]!

The problem of uncertainty in low-level input fault tree data affecting the accuracy of (and hence confidence in) output reliability estimates from the fault trees has been recognized for some time. The structure of the fault trees themselves has been used by several authors to synthesize attempts to combat this problem. In [20], the standard deviations of the reliability statistics were used to bound the standard deviation of the top-event probability in fault tree analyses. A combination of exact expressions for, and bounds on the values of, the variance of the necessary combinations of events is used to propagate the standard deviations through the trees [20]. However, this analytic approach implies specific knowledge of the closed-form standard deviations of the input data (primary events). This is typically unavailable for many engineering applications such as robotics, and our motivation for this work was to find a method to represent more general uncertainty.

As an attempt to deal with more uncertain situations, the use of Dempster-Shafer theory to model imprecise input data is proposed in [11]. Dempster's rule of combination (as opposed to Bayes rule) is used in [11] to assign *masses* (as opposed to probabilities) to input events and gate logic in fault trees. This allows user uncertainty in both the fault tree logic and the appropriate assignment of probabilities given trial data to be reflected in the calculations. The resulting output information is expressed in terms of the three logical variables True, False, and Uncommitted (i.e. a logical representation of uncertainty). However, in many applications, users would like an output representation that contains more information about the structure of the uncertainty.

Another approach to dealing with uncertainty that has proven popular recently is that of fuzzy logic. Accordingly, several authors [2, 24, 27, 28] have advocated 'fuzzifying' fault tree input data. For example, the idea of fuzzy data has been applied to robot reliability in [32], where

each input was treated as a range (essentially the range of 'unit possibility' of a fuzzy membership function reflecting the uncertainty of the input data), resulting in a corresponding range for the output reliability estimate. However, as shown in [16], this approach can lead to extremely conservative results, and can result in loss of uncertainty in some cases, as is explained in the next section.

Commercial risk assessment computer programs performing uncertainty analysis are usually based on Monte Carlo simulation [25, 21]. This method collects statistical results from the processing of random samples of the input data space. Its mathematical basis establishes essentially that (after a typical implementation), if well-characterized input normal distributions are considered and N output values are obtained from the simulation of N input vectors of random samples (output mean μ and standard deviation σ), it can be assumed that the real output mean is inside the range $[\mu - 3\sigma, \mu + 3\sigma]$ with a probability 0.997, with an error given by $3\sigma/\sqrt{N}$. This error tends to 0 as N increases. However, when considering uncertainty analysis, input distributions are not really known, so they must be approximated with known distributions or described in terms of histograms to obtain estimates of the statistical parameters of the outputs. In general, the accuracy of these estimates increases with the number of simulated samples. But this is only an approximate relationship, as the increase is not strictly monotonic, and it is not known how many samples are required for a given accuracy. In Monte Carlo simulation, this problem is particularly important when input distributions are very different from normal distributions (e.g. multimodal distributions that have several peaks), and when large input spaces are considered. In these situations, estimates based on Monte Carlo methods should be interpreted carefully. Another problem of this approach is that it is not guaranteed that the worst case

scenario (e.g. inputs causing the maximum failure probability of the top event) is actually simulated. Therefore, this method cannot be used to obtain bounds on the output probabilities.

Over the last few years, there has been increasing interest and activity in the theory and application of Interval Arithmetic and Interval Computation [1, 17, 18, 19]. These techniques arose from the need to deal with uncertainty (manipulate imprecise data, keep track of round-off error, etc.). They have been used in numerous applications recently [3, 23] and constitute the formal basis of the uncertainty propagation approach presented here.

We are thus motivated herein to represent input data in terms of intervals. The standard *interval* data type is extended with a 'mass' so intervals can be used to represent 'densities' in the form of histograms [4, 5]. Note that closed form densities or distributions are not required, and the data can come from standard reliability data. The method is based on formal interval computations [1, 17, 23] and represents histograms by using predefined interval models called *grids* which allow control of the tradeoff between the accuracy of the results and the complexity of the computation. One major advantage of this interval approach is that in many cases it is possible to simulate the whole interval input space in reasonable times, so the results are safe, meaning that it is guaranteed that the results from the worst case scenario are included in the output histograms. This is a clear advantage over methods based on partial simulation of the input data space (e.g. Monte Carlo), which cannot guarantee generation of the worst case without complete simulation of all possibilities, as opposed to all intervals (significantly less computation) in our method.

The interval approach used in this paper is based on that in [6]. However, the version in [6] is valid only for integer values, and the reliability application requires real numbers.

Modifications to the algorithm to allow consideration of real numbers was performed in [9], in which the structure of estimates from the fuzzy and interval approaches were compared for the robot application for the first time. The case of loss of uncertainty using the fuzzy approach, and the inherent ability of the interval approach to avoid this problem was discussed in [8]. However, the interval estimates in [8, 9] are obtained in 'brute-force' fashion, requiring extensive computation, and in some cases forcing restricted analysis of the problem. In this paper, we introduce a new generalized and fully parameterized version of the method, which allows more complete analysis. Its distinctive feature is the new set of grid models that allow improvement of the efficiency of the computation process.

In the following sections, an illustrative example of a three joint robot with redundant sensors is presented and evaluated in terms of some current approaches (fuzzy, Monte Carlo). Then, the generalized interval method is formally introduced. Results are obtained showing that the interval method improves the accuracy (measured in terms of the (1) in section 4) of the reliability estimates in robot fault tree analyses over those in [32], avoiding the loss of uncertainty detected when considering the aging of the system (e.g. time-varying reliability). This conclusion (discussed further in section 2) was predicted in [8] using the *rare event* approximation ($P(A \cup B) = P(A) + P(B)$) when computing the fault tree, and it is confirmed here with results obtained from the exact fault tree computation accompanied with detailed statistics of the process. We also show that the interval method not only significantly improves the efficiency of algorithms based on exhaustive simulation of the input space [16], but constitutes a feasible alternative to Monte Carlo simulation in terms of computation times, with the advantage of providing guaranteed bounds for the results.

Notation

$\boldsymbol{Si}$	Robot Sensor $\boldsymbol{i}$, Optical Encoder, with Failure Probability s_i
$\boldsymbol{Mi}$	Robot Actuator $\boldsymbol{i}$, Electric Motor, with Failure Probability m_i
$\boldsymbol{Ji}$	Robot Joint $\boldsymbol{i}$ whose Failure Probability is obtained as j_i
λ	Component Failure Rate
t	Mission Time
p	Component Failure Probability
u, v, w, x	Component Failure Probability Interval Points

2. ILLUSTRATIVE EXAMPLE

Assumptions

A. Rare event approximation, $p(A \cup B) = p(A) + p(B)$
B. Independent failure events

The approach will be presented using the example of a planar robot manipulator, illustrated in Figure 1. This type of robot structure is analogous to the 'upper arm - forearm - hand' structure of the human arm, when operated in the plane. The rigid links are connected by three rotational joints, each driven by its own electric motor. Positional feedback is obtained by the use of dual (redundant) optical encoders at each joint. This arrangement is fairly typical for robot manipulator subassemblies in remote environments [32].

The task is to position the end effector arbitrarily in its planar workspace. Since this can be achieved with any two of the joints, the robot has a redundant degree of freedom. For this robot, failure therefore occurs when any *two* of the joints are rendered immobile (joints are considered locked by brakes upon failure - the issue of failures in the brakes themselves is beyond the scope of this example). In turn, each joint fails if either (a) its motor fails; or (b) both of its encoders fail.

The above logic can be easily encoded into a fault tree for the robot [32], with the top event being 'failure of the robot', as in Figure 2. We have chosen this example, first introduced in [32],

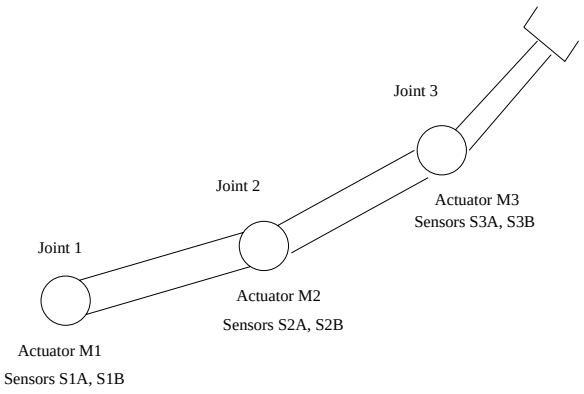

Figure 1: Three joint planar robot with redundant sensors

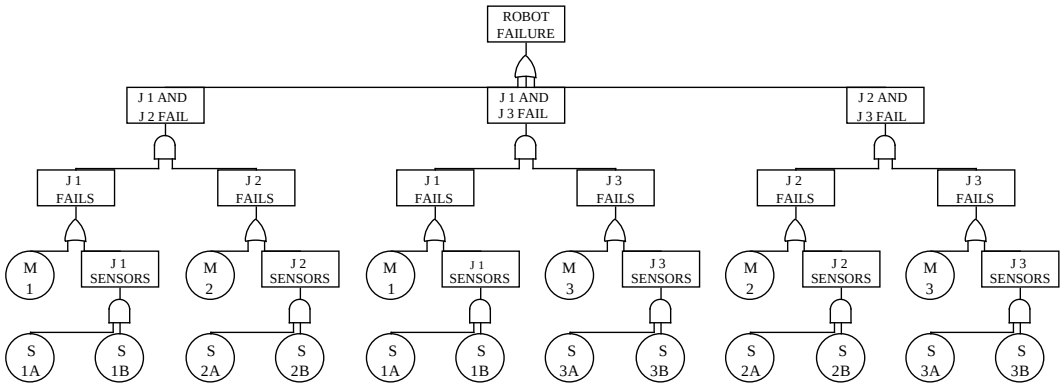

Figure 2: Fault Tree for three joint planar robot with redundant sensors

to compare and contrast our results with those in [32], where the concept of propagating 'failure probability ranges' (rather than fixed values for the probabilities of subsystem failures) through the fault tree was proposed. This concept was shown to be consistent with the notion of fuzzy fault tree analysis introduced in [28], in which the input data is represented by a 'fuzzy trapezoid' describing the *possibility* of the input failure probability, as shown in Figure 3. Note that the probability has a possibility of 1 over a range of values of p, from v to w. Between u and v, and between w and x, the possibility varies linearly between 0 and 1. Thus the set (u, v, w, x) uniquely describes each distribution.

The idea of using intervals, or 'fuzzy probability ranges', is appealing since this often fits the

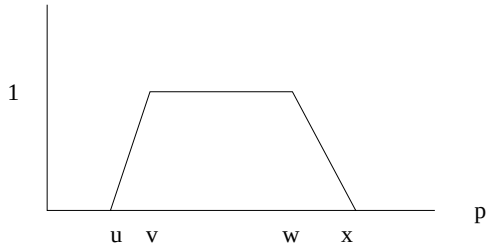

Figure 3: Representation of input data as fuzzy possibility

available data. For example, existing reliability data for electric motors (typically used in robot manipulators) across a wide range of applications reports an overall failure probability of 0.00924, together with information that 68 per cent of cases will be between 0.22 and 4.5 times this value, and 90 per cent of cases will be between 0.08 and 11.8 of it [32]. Thus, for this case, (u, v, w, x) are estimated as $(0.0007392, 0.00203, 0.04158, 0.109032)$, so that 90 per cent of the cases are represented under the 'possibility distribution'. For a typical optical encoder, a corresponding representation of $(0.00124, 0.00341, 0.0698, 0.184)$ is reported in [32].

The above 'fuzzy estimates' can be summed and multiplied using the following rules [28]:

$$\begin{aligned} Addition : (u, v, w, x)_{new} &= (u_1 + u_2, v_1 + v_2, w_1 + w_2, x_1 + x_2) \\ Multiplication : (u, v, w, x)_{new} &= (u_1 u_2, v_1 v_2, w_1 w_2, x_1 x_2) \end{aligned}$$

where the input distributions are given by (u_1, v_1, w_1, x_1) and (u_2, v_2, w_2, x_2).
The use of these rules allow the combination of such fuzzy estimates through the standard fault tree logic gates (AND, OR, etc), while preserving the essential fuzzy structure (note that the estimates above are computed pairwise from the corners of the input trapezoids). That method, which we term the 'fuzzy approach', was used in [32] to evaluate the effects of uncertainty on the reliability of the encoders and motors on the overall robot. For example, given the numerical distributions for electric motor and optical encoders given above, and considering the event E,

'joint Ji fails', its probability is given by $s_{iA} * s_{iB} + m_i$ (using assumption #A). Therefore, if input probabilities are independent (assumption #B), the output trapezoid for E is obtained as $E(u, v, x, w) = (0.0007407, 0.00204, 0.04645, 0.14289)$.

This result reflects the uncertainty in the output, beyond that which is obtained from the traditional single value obtained via a quantitative fault tree analysis, as pointed out in [32]. However, as we shall see, the 'fuzzy' operations used to generate the fuzzy trapezoid above produces a very conservative estimate. Furthermore, such 'fuzzy' manipulations have a severe limitation that was not unveiled until input probabilities varying with time (i.e. taking into account the aging of the system) were considered. This case is reported in [16] where it is shown that the fuzzy approach as in [28, 32] and outlined above led to erroneous reliability range estimates of zero width . As defined here, this means complete loss of uncertainty!

For our robot example, we now consider the event F, 'exactly one joint fails'. From Figure 2, the probability of F, f, is given by (using assumption #B)

$$f = 3 * (j_1 * j_2 * j_3) - 2 * (j_1 * j_2 + j_1 * j_3 + j_2 * j_3) + j_1 + j_2 + j_3$$

where j_1, j_2, j_3 are the probabilities of failure of joints 1, 2, 3 respectively, which are now computed as $s_{iA} * s_{iB} + m_i - s_{iA} * s_{iB} * m_i$ (no rare event approximation). Considering the fuzzy approach of [28, 32] and the failure rates specified above for this example for the time-varying case (using the exponential distribution for component failure probability, i.e. $p(t) = 1 - exp(-\lambda t)$, with λ the failure rates in the distributions [29]), we find that at time $t = 18.55$, the values of v and w (corresponding to Figure 3) are both equal to 0.112. Thus the trapezoid becomes a triangle, or equivalently the 'possibility range' between v and w vanishes!

This exposes a serious limitation with the fuzzy approach, in which the uncertainty present in the physical situation in not reflected in the calculation. Clearly a more accurate approach is called for.

This loss of uncertainty is due to the way the ranges are combined in the fuzzy addition and multiplication. Only the endpoints of the ranges are combined, and then only in a pairwise fashion. However, best and worst case conditions at the endpoints (i.e. the shape of a trapezoid) are only preserved if coherent systems are considered ($\delta P(p_j)_{j=1,N}/\delta p_i \geq 0$ for all $i = 1, N$, where $P(p_j)_{j=1,N}$ is the probability of the system and p_i are the N independent input probabilities). Since the expressions of f and j_i correspond to non-coherent systems, the fuzzy approach can provide erroneous estimates depending on the input data (e.g. data after the aging process).

The solution proposed in [16] to the above problem was brute-force exhaustive sampling of combinations of points within the ranges. However, this was noted to be computationally inefficient (in our example, nine independent input probabilities described with five fractional digits yield an input space of 10^{45} possible combinations of input values), and, in any case, it left unclear how fine the sampling had to be to be effective.

Methods based on Monte Carlo simulation can also be considered to evaluate the fault tree of our example, In this case, a transformation of the available numeric data into well-characterized distributions or into histograms (as is explained in the next section) is first required. The histogram representation is preferred here due to the uncertainty implied in the available data. As for the sampling of the huge input space above, two major choices are provided in existing risk assessment packages such as [13, 25]: Monte Carlo Sampling (MCS) and Latin

Hypercube Sampling (LHS). For MCS, random numbers, usually uniformly distributed, are generated and then transformed to the histogram distribution. For large input spaces, MCS is often time consuming due to the large number of samples typically required [33], as distributions obtained from simulating a very small fraction of the input space can hardly be trusted [14].

LHS was developed as an effort to reduce the number of such samples [33, 25]. LHS divides the range of each of the k input variables into n non-overlapping intervals, randomly selects n values (one value from each interval) for each of the k variables, and combines them randomly into n k-tuples which are used as input vectors. While LHS reduces the number of samples, they are much harder to compute, and, in the words used in [22, 33], LHS only becomes particularly useful when an *exceedingly* sparse N-dimensional space must be sampled at M points. Consequently, both methods are regarded as time consuming when many samples must be generated as is the case in the example in this paper. In addition, there is no guarantee that the selected samples include the worst case scenario, so no safe output bounds can be derived from these methods (bounds can be estimated from the largest simulated results, but without any guarantee to be correct).

In the following section, we review and discuss an alternative approach using interval methods that avoids the problems detected in the fuzzy method while providing guaranteed output bounds as opposed to MCS and LHS. Although LHS may resemble the interval method presented next, it is quite different as the interval method is based on a different computation model with different data types (i.e. intervals).

3. A NOVEL INTERVAL APPROACH

We now propose a new approach, building on our initial work in [9], in which the input data

is reformulated in terms of an estimate of the probability densities of the inputs in terms of histograms [4, 5]. From these histograms we compute the probability density of the event of interest (top event of the fault tree, etc.). For example, for the electric motor data discussed in the previous section, an input distribution (histogram) is constructed as follows:

$$
\begin{aligned}
m &= (0, 0.000739) : 0.05 \\
m &= (0.000739, 0.00203) : 0.11 \\
m &= (0.00203, 0.00924) : 0.34 \\
m &= (0.00924, 0.04158) : 0.34 \\
m &= (0.04158, 0.10903) : 0.11 \\
m &= (0.10903, 1) : 0.05
\end{aligned}
$$

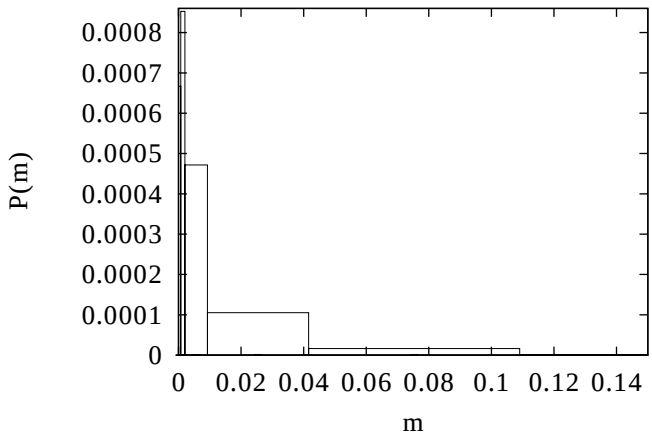

Figure 4: Motor Probability of Failure

Thus the function m (representing the failure probability of an actuator Mi) has 0.05 of its probability mass between 0 and 0.000739, etc. A plot of the corresponding histogram is presented in Figure 4. Only the range $[0, 0.15]$ is included to allow a more detailed representation. Note that in the above type of distribution, in contrast to the fuzzy trapezoid, *all* the probability mass is included under the distribution. These histograms are described in terms of intervals which are defined as:

Definition 1. An interval $[a, b]/p$ is the set of $N = (b - a + 1)$ integers x that verify $a \leq x \leq b$ with an associated probability of mass p.

We consider that p is uniformly distributed in $[a, b]$, so that the probability of any $x \in [a, b]$ can be computed as p/N. Note that the interval definition above is based on integer numbers, so some preprocessing (and postprocessing) is required for its application to the reliability problem where the data is based on rational numbers. In particular, the range $[0, 1]$ is partitioned into M discretization steps of equal size represented by a number (midpoint) and a probability (obtained from the original distribution). Then, the $[0, 1]$ range is scaled into the range $[0, M - 1]$ and mapped to an integer distribution in the same range, where each integer n has the probability of the subrange $[n - 0.5, n + 0.5]$ in the scaled distribution. It should be noted that the scaling may require some additional operations (multiplication or division by M) when computing the fault tree algorithm in order to maintain the original behavior. Also, the value of M plays an important role in the computation as it determines the numeric accuracy. In general, M is selected here according to the maximum precision provided in the input data so that no information is lost in the scaling process. For example, if the input data is provided with m fractional digits, a value $M = 10^m$ is used for the scaling.

The most distinctive feature of this interval approach is that it is heavily based on predefined interval models called grids to represent the histograms through the computation [7]. Grids allow selecting the size of the data ranges (granularity) even at the operation level, thus providing a mechanism to control the tradeoff between accuracy and computation times. In general, the complexity of an interval-based computation is determined by the size of the interval input space and by the process of collecting the result of each run in a global output histogram. If unbounded

interval models are used, exact collection of results tends to increase the number of bars of the output histogram (merging two intervals with non-empty intersection produces three output intervals) and the collection time easily becomes prohibitive. Instead, bounded interval models based on grids are used. Grids are lists of disjoint adjacent intervals covering the whole numeric range of interest. In this situation, the collection process is governed by two rules:

- *Merge* rule: all interval results occurring inside the same interval of the grid are represented as a single interval with probabilities added.
- *Split* rule: any interval result spanning over several intervals of the grid is partitioned into as many intervals with proportional probabilities before being merged in the output histogram.

Several types of grids can be defined to control, not only the collection process, but also the size of the interval input space (i.e. grids are also applied to the input histograms).

Of course, the user may decide on what histogram bars should be used to represent inputs and outputs as in [4]. We call these custom grids. They are tedious to define and usually require some previous knowledge about the shape of the (output) distributions to be effective. In our interval approach, regular analytic grids are mainly considered. They are characterized by four parameters: a type, a granularity (g), a center of symmetry (c), and a focus (f). The type determines the analytic expression of the grid. The granularity represents the size of its intervals. The grid is symmetric with respect to c. Finally, the focus reduces the number of intervals of the grid to a finite number around c. This last parameter is a distinctive feature from previous versions of grid models [6, 7]. It reduces the complexity of the computation by limiting the region of the numeric space which is analyzed in detail.

Two types of regular grids are considered: linear grids (formed by intervals of equal size)

and geometric grids (formed by intervals with exponentially increasing sizes as they depart from the center of symmetry).

Definition 2. A linear grid is a set of adjacent intervals $[A, B]$, each of them uniquely identified by integer n, that verify one of the following identities:

$$[A,B] = \begin{cases} [-Inf, c + g(-f+1)] & (n \leq -f) \\ [c+gn+1, c+g(n+1)] & (-f < n < -1) \\ [c-g+1, c-1] & (n=-1) \\ [c,c] & (n=0) \\ [c+1, c+g-1] & (n=1) \\ [c+g(n-1), c+gn-1] & (f > n > 1) \\ [c+g(f-1), Inf] & (n \geq f) \end{cases}$$

Definition 3. A geometric grid is a set of adjacent intervals $[A, B]$, each of them uniquely identified by integer n, that verify one of the following identities:

$$[A,B] = \begin{cases} \left[-Inf, c - g^{(f-1)}\right] & (n \leq -f) \\ \left[c - g^{-n} + 1, c - g^{-(n+1)}\right] & (-f < n < 0) \\ [c,c] & (n=0) \\ \left[c + g^{(n-1)}, c + g^n - 1\right] & (f > n > 0) \\ \left[c + g^{(f-1)}, Inf\right] & (n \geq f) \end{cases}$$

The quantities g and f are positive integers. If the focus feature is not used, the previous expressions revert to those in [6] ($f = Inf$ in the expressions above). It should be noted that the ability to set the center of symmetry to any point of the range is particularly important in the geometric grid as the center is the point where the model is most accurate.

Once the input distributions have been transformed to integer ranges and represented in terms of a predefined grid, the computation method is adapted from [4] as:

1. Consider the input space $N_1 \times \ldots \times N_I$ where N_i is the set of intervals describing the

histogram of input i.

2. For each vector $(\ldots, [a_{ij}, b_{ij}]/p_{ij}, \ldots)$ of the input space, where $[a_{ij}, b_{ij}]/p_{ij}$ represents the j-th bar of the histogram describing input i:
   (a) Compute its probability $P = \prod_{i=1}^{I} p_{ij}$.
   (b) Execute the operations using interval arithmetic.
   (c) Assign P to each resulting interval.
   (d) Collect the results in output histograms applying the split and merge rules.

Standard definitions of operations between intervals are used for the computation [17, 23]. In the case of positive intervals:

$$\begin{aligned} [x_1, y_1] + [x_2, y_2] &= [x_1 + x_2, y_1 + y_2] \\ [x_1, y_1] - [x_2, y_2] &= [x_1 - y_2, y_1 - x_2] \\ [x_1, y_1] * [x_2, y_2] &= [x_1 * x_2, y_1 * y_2] \\ [x_1, y_1] \,/\, [x_2, y_2] &= [x_1/y_2, x_2/y_1] \end{aligned}$$

It is clear that assuming uniform distributions in the output intervals is an approximation that should be considered in the analysis of the estimates.

Since the interval method computes the fault tree for all possible input interval combinations, it produces a much more complete estimate of the output distribution than the methods in [28, 32], where only the trapezoid corners are considered and in a pairwise fashion. In fact, non-coherent systems can now be evaluated, as interval computations guarantee that the interval result of an operation contains all the results that can be obtained when operating values from the input intervals. Therefore, even though best and worst conditions may not be represented by the interval endpoints, they are included in the histogram distribution collecting all the interval results and the interval method can be used to obtain bounds for the output probabilities. If such bounds are the only result of interest, the computation can be simplified to perform Skelboe's

algorithm. This algorithm is based on the cyclic bisection of only the input intervals providing the largest output probabilities. Cyclic bisection provides tighter bounds on each iteration so there is no limitation in the accuracy that can be obtained. (For a detailed explanation see [17]).

It is clear that guaranteed bounds are a clear advantage of the interval method over Monte Carlo simulation and the method in [16]. It remains to be seen how they compare in terms of computation times. From our experiments, it is observed that, in general, interval computations do not add significant complexity. In fact, in several experiments the overhead caused by the collection process when using intervals can be comparable to the overhead caused by random sample generation in Monte Carlo (except when very detailed output grids are used). So, in this situation, each interval run of the fault tree takes approximately twice as much as each Monte Carlo run.

The interval approach is particularly interesting for computing large input spaces dominated by large input ranges (i.e. many digits in the input probability values). However, they are less effective when input spaces are dominated by a large number of inputs. This is also a problem for Monte Carlo methods. In the reference manual of [25], it is suggested to consider correlation classes: sets of inputs with identical distributions who share the same sample in each run (heuristic). Another approach suggested in [6] is to partition the fault tree into independent (or quasi-independent) subtrees with less inputs which are computed separately. For example, the fault tree of the previous section can be obtained from the independent computation of the probabilities of events 'dual sensor (Si_A, Si_B) fails', 'joint Ji fails', and 'robot arm fails', where each computation takes as input the output of the previous one. It can be observed that the original nine inputs are reduced to two, two and three respectively for each subtree. If subtrees are

not truly independent, some oversize will appear in the interval results, but they will still be guaranteed bounds of the distribution.

4. RESULTS USING INTERVAL METHODS

We now present the results of the novel interval method presented here as applied to the robot example in section 2. No rare event approximation has been applied, as opposed to some initial results in [8]. The distributions obtained show much greater detail, and thus provide significantly more information, than previous estimation results.

In a first set of experiments, the impact of using different types of grids during the estimation process is analyzed through the distribution obtained for E, the event 'joint Ji fails'. Figure 5 represents the distribution obtained when using standard intervals (no grid). Figure 6 uses geometric grids (grain 2, center 0, no focus) for both the input representation and the collection of results. Figure 7 uses the same geometric input grid but results are collected through a linear grid (grain 100, center 0, no focus). Grid parameters take into account that the input distributions are scaled to a range $[0, 10^5]$ (i.e. probability values given with five fractional digits). The output distributions are only represented in the range $[0, 0.2]$.

A comparison of some parameters of interest obtained from the simulations is presented in Table 1, including (1) the number of interval operations, (2) the number of output intervals which gives an estimate of the memory requirements, (3) the output ranges which, in general, exceed the $[0, 1]$ range due to the oversize implied by the interval computations, (4) the probability mass inside the $[0, 1]$ range, and (5) the simulation time. The table also includes results when a focus value is used to limit the linear output grid to the range $[0, 0.2]$, thus reducing the collection time. It should be noted that the interval computations are performed in the integer domain by scaling

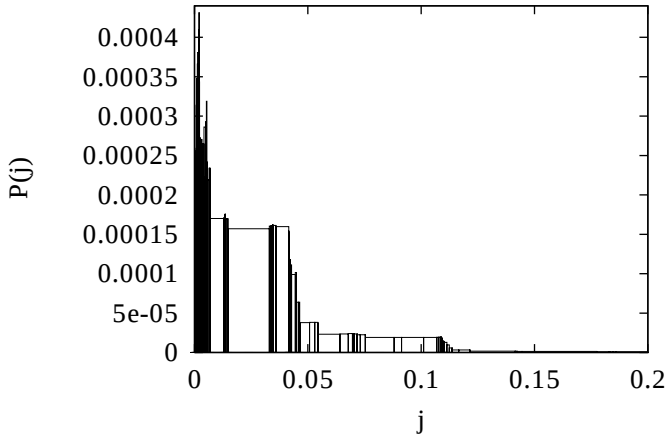

Figure 5: Joint Probability of Failure (no grid)

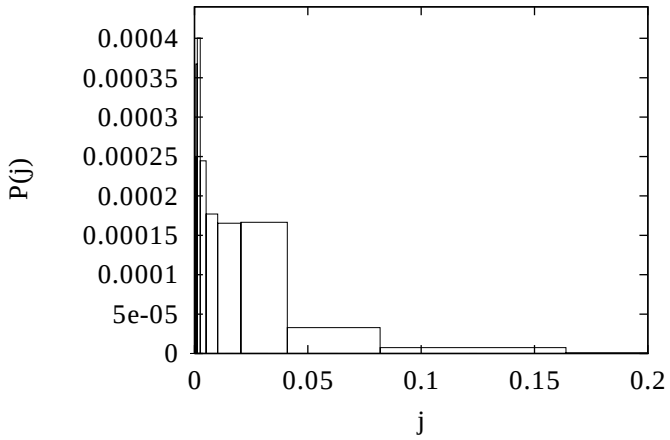

Figure 6: Joint Probability of Failure (geometric input and output grids)

the $[0, 1]$ range to a $[0, 10^5]$ range. Since interval operations only provide bounds for the output ranges with an accuracy determined by the data dependencies involved, it is possible that some results fall outside the $[0, 10^5]$ range, thus leading to parameter (4) above. (Times should be taken only as relative measurements, as performance is not optimized in the current software implementation of the approach).

Considering a simulation strategy based on individual samples (i.e. Monte Carlo), the size of the input space is 3×10^5 input vectors that would require 15×10^5 operations in case of

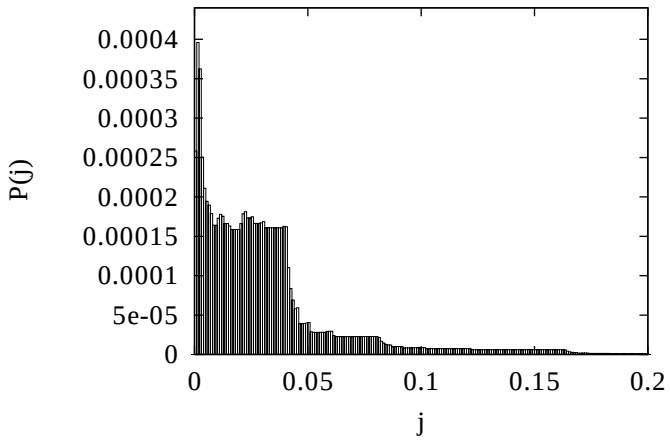

Figure 7: Joint Probability of Failure (geometric input grid; linear output grid)

Input grid	Output grid	Interval operations	Output intervals	Output range	Prob. in [0,1]	Time (ms)
none	none	380	541	[-0.70695,1.97994]	0.999453	1170
geo(2,0,-)	geo(2,0,-)	1024	404	[0,1.57053]	0.999746	330
geo(2,0,-)	lin(100,0,-)	1024	1958	[0,1.57053]	0.999746	2280
geo(2,0,-)	lin(100,0,0.2)	1024	487	[0,1.57053]	0.999746	390

Table 1: Statistics from the simulations for event 'Joint Ji fails'

exhaustive computation. This number is greatly reduced in all interval-based simulations, thus proving their efficiency. When no grids are considered (Figure 5) a great level of detail can be obtained. However, merging individual results in the output histogram increases substantially the simulation time even for this simple case (more complex simulations show that this time quickly becomes too large). In addition, since the number of input intervals used for the computation is determined by the available data, the oversize of the output range is much greater than when using grids that enforce more input intervals (although in all cases the probability mass outside $[0, 1]$ is very small, showing little impact of the data dependencies on the accuracy of the interval results). Geometric grids (Figure 6) allow fast computations while memory requirements remain low. A

combination of a geometric input grid with a linear output grid (Figure 7) maintains the computational complexity while providing greater detail in the results. Naturally, this implies an increase in the computation time due to the collection process which is mainly a function of the granularity of the grid. Therefore it can be adjusted depending on the complexity of the computation, as opposed to the simulations without grids where no control mechanism is available.

The event R, 'robot arm fails' is analyzed in a second set of experiments. This is a more complex case where the failure probability is given by

$$r = j_1 * j_2 + j1 * j3 + j2 * j3 - 2 * (j_1 + j_2 + j_3)$$

where j_i is the probability of failure of joint Ji represented in Figure 6. The results obtained using geometric input and output grids with grain 2 and center 0 are represented in Figure 8. Results with other grid combinations display similar features. The number of operations is 61440 and the probability mass outside the $[0, 1]$ range still remains very low (0.999857). The simulation time is 24590 msec.

In this set of experiments, the interval-based results have been compared to 'exact' results (in the integer domain) obtained from exhaustive computation of all individual integer inputs in order to evaluate the error introduced in the interval results. Exhaustive computation has been possible by using scaling factors N below 10^3 (probabilities with up to three fractional digits) and considering that all sensors and all motors behave identically (otherwise the size of the input

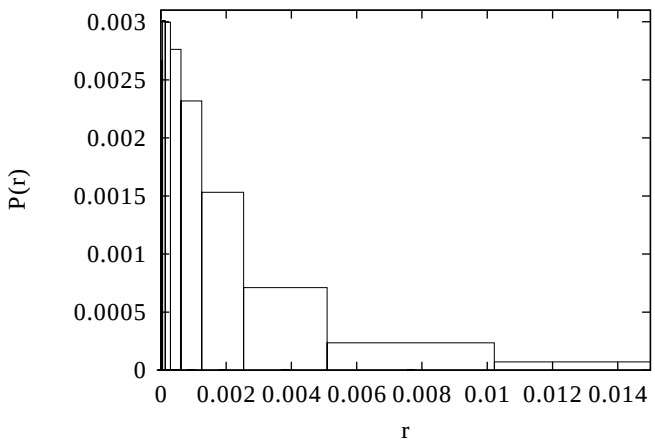

Figure 8: Output distribution (probability estimate) of event 'Robot arm fails'

space would have been too large). The error $\mathcal{E}$ has been obtained as

$$\mathcal{E}(\%) = \sum_{\forall i} \frac{\mid p_{i,ex} - p_{i,int} \mid}{2p_{i,ex}} \times 100$$

where $\mathcal{E}$ is a percentage of the exact output distribution, $p_{i,ex}$ and $p_{i,int}$ are the probabilities of the i-th bar of the exact and interval-based output histograms, both represented in terms of the interval grid, and the factor 2 accounts for the fact that each misplaced results causes a difference in the distributions of twice its probability. An error below 7% has been obtained in all comparisons, while computation times have been reduced as much as four orders of magnitude when using geometric grids. In [9] it is also shown that, if fixed computation times are considered, interval results provide smaller errors $\mathcal{E}$ than methods based on random sampling like Monte Carlo (i.e. they are faster in getting closer to the true distributions). This is due to the fact that interval input vectors can be sorted so the combinations with higher probabilities are computed first. In general, this is not feasible if individual numeric samples are selected.

The last set of simulations investigates the effect of aging input probabilities in the reliability

of the system. In particular, we are interested in the results provided by the interval approach when the fuzzy approach breaks down. The output distribution for event F provided by the interval method using a linear input grid with grain 3000 and a linear output grid with grain 100 is represented in Figure 9. In this case, the joint failure probabilities used in the computation are previously obtained also using input and output linear grids. This provides much more detail at the cost of higher requirements in memory and computation times. In particular, the simulation providing the distribution in Figure 9 computes 589560 interval operations in 1600 seconds (including input generation and output collection times). The probability mass outside the $[0, 1]$ range still remains very low (0.986358). This result confirms that the fuzzy method, which shows a loss of uncertainty and peak at 0.112 for this example as explained in section 2, underestimates the true result.

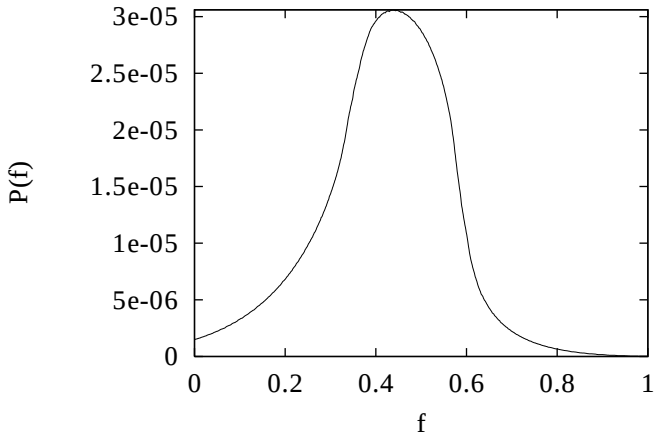

Figure 9: Output distribution (probability estimate) of event ‘Exactly one joint fails’ at time 18.55

In summary, the examples confirm that the interval methods introduced here result in more accurate and detailed estimates than with the fuzzy approach, and also produce consistent and meaningful results in cases where the fuzzy method does not (i.e. loss of uncertainty). In addition,

by the appropriate use of grids, tradeoffs between accuracy and computational complexity can be easily managed, as demonstrated in the examples. This allows comparable performance to Monte Carlo methods with the advantage that guaranteed distribution bounds are provided.

5. APPLICABILITY AND FUTURE WORK

The interval-based reliability estimation algorithm introduced in this paper had been illustrated using examples from robotics. We anticipate that the robotics arena, in which dealing with uncertain input data has proved a major obstacle in quantitative reliability estimation [32], is one area in which the method introduced here is likely to be particularly applicable. However, the interval approach to reliability estimation presented here is applicable to any system modeled by fault trees with input uncertainty represented by intervals. The ability of the approach to formally include the effects of the uncertainty in the input data in the output reliability estimate make the method very appealing across a range of applications. Our ongoing work focuses on the effects of different scalings and representations of the interval data on the output distributions.

6. ACKNOWLEDGMENTS

The first author is supported by projects ESPRIT 24137 and CICYT TIC97-0928. The second author is supported by the National Science Foundation (NSF) under grant CMS-9796328, by NSF/EPSCoR grant EPS-9630167, and by the U.S. Department of Energy (through Foster-Miller Inc.) under contract #DE-FG07-97ER14830, subcontract #805-05002.

REFERENCES

[1] G. Alefeld and J. Herzberger. *Introduction to Interval Computations*. AP, NY, 1983.

[2] J.A.B. Geymayr amd N.F.F. Ebecken. Fault-Tree Analysis: A Knowledge-Engineering Approach. *IEEE Transactions on Reliability*, 44(1):37–45, 1995.

[3] APIC'95. International Workshop on Applications of Interval Computations, El Paso, Texas, February 1995. http://cs.utep.edu/interval-comp/apictoc.html.

[4] D. Berleant. Automatically Verified Reasoning with Both Intervals and Probability Density Functions. *Interval Computations*, 1993(2):48–70, 1993.

[5] D. Berleant. Bounding the Results of Arithmetic Operations on Random Variables of Unknown Dependency using Intervals. *Reliable Computing, Kluwer*, 4(2):147–165, 1998.

[6] C. Carreras, J.A. Lopez, and O. Nieto. Bit-width Selection in Data-path Implementations. In *Proc. 12th IEEE International Symposium on System Synthesis*, pages 114–119, San Jose, CA, Nov 1999.

[7] C. Carreras, O. Nieto, J.M. Moya, and F. Moya. Fast Data Flow Characterization. In *Proceedings VIII BELSIGN Workshop*, Oct 1998.

[8] C. Carreras and I.D. Walker. Interval Methods for Improved Robot Reliability Estimation. In *Proceedings IEEE Annual Reliability and Maintainability Symposium, Los Angeles, CA*, pages 22–27, Jan 2000.

[9] C. Carreras, I.D. Walker, O. Nieto, and J.R. Cavallaro. Robot Reliability Estimation Using Interval Methods. In *Proceedings 1999 Workshop on Applications of Interval Analysis to Systems and Control*, pages 371–385, Girona, Spain, 1999.

[10] Joanne Bechta Dugan and Stacy A. Doyle. New Results in Fault-Tree Analysis. In *Proceedings Annual Reliability and Maintainability Symposium Tutorial Notes*, pages 1–23, Las Vegas, NV, 1996.

[11] M.A.S. Guth. A Probabilistic Foundation for Vagueness and Imprecision in Fault-Tree Analysis. *IEEE Transactions on Reliability*, 40(5):563–571, 1991.

[12] B.M. Harpel, J.B. Dugan, I.D. Walker, and J.R. Cavallaro. Analysis of Robots for Hazardous Environments. In *Proceedings IEEE Annual Reliability and Maintainability Symposium*, pages 111–116, Philadelphia, PA, 1997.

[13] Item Software. AvSim. http://www.itemsoft.com/Monte_Carlo.html.

[14] Janne Pesonen et al. Interval Approach Challenges Monte Carlo Simulation. In *Proceedings of Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN-95)*, 1995.

[15] K. Lauridsen, P. Christensen, and H. E. Kongsø. Assessment of the Reliability of Robotic Systems for Use in Radiation Environments. *Reliability Engineering and System Safety*, 53(3):265–276, 1996.

[16] M.L. Leuschen, I.D. Walker, and J.R. Cavallaro. Robot Reliability Through Fuzzy Markov Models. In *Proceedings IEEE Annual Reliability and Maintainability Symposium*, pages 209–214, Anaheim, CA, 1998.

[17] R.E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, 1979.

[18] K. Nickel. *Interval Mathematics 1980*. Academic Press, 1990.

[19] K. Nickel. *Interval Mathematics 1985*. Springer-Verlag, 1996.

[20] L.B. Page and J.E. Perry. Standard Deviation As an Alternative to Fuzziness in Fault Tree Models. *IEEE Transactions on Reliability*, 43(3):402–407, 1994.

[21] Palisade Corp. @RISK. http://www.palisade.com/html/risk_for_project.html.

[22] W.H. Press, S.A. Teukolsky, W.T.Vetterling, and B.P. Flannery. *Numerical Recipes in FORTRAN: The Art of Scientific Computing*. Cambridge Univ. Press, New York, 1992.

[23] H. Ratschek and J. Rokne. *Computer Methods for the Range of Functions*. Ellis-Horwood, Chichester, 1988.

[24] I.D. Roberts and A.E. Samuel. The Use of Imprecise Component Reliability Distributions in Reliability Calculations. *IEEE Transactions on Reliability*, 45(1):141–144, 1996.

[25] SAPHIRE. IRRAS. http://saphire.inel.gov/saphire_main.html.

[26] E.J. Shen. Retrieval of Underground Storage Tank Wastes: The Hanford Challenge. In *Proceedings American Nuclear Society Topical Meeting on Robotics and Remote Systems*, pages 549–553, Monterey, CA, 1995.

[27] D. Singer. A Fuzzy Set Approach to Fault Tree and Reliability Analysis . *Fuzzy Sets and Systems*, 37(2):267–286, 1990.

[28] H. Tanaka, L.T. Fan, F.S. Lai, and K. Toguchi. Fault-Tree Analysis by Fuzzy Probability. *IEEE Transactions on Reliability*, 32(5):453–457, 1983.

[29] W. E. Vesley, F. F. Goldberg, N. H. Roberts, and D. F. Haasi. Fault Tree Handbook. NUREG 0492, US Nuclear Regulatory Commission, Washington DC, January 1981.

[30] M.L. Visinsky, J.R. Cavallaro, and I.D. Walker. Robot Fault Detection and Fault Tolerance: A Survey. *Reliability Engineering and System Safety*, 46(2):139–158, 1994.

[31] I.D. Walker and J.R. Cavallaro. Failure Mode Analysis for a Hazardous Waste Clean-Up Manipulator. *Reliability Engineering and System Safety*, 53(3):277–290, 1996.

[32] I.D. Walker and J.R. Cavallaro. The Use of Fault Trees for the Design of Robots for Hazardous Environments. In *Proceedings IEEE Annual Reliability and Maintainability Symposium*, pages 229–235, Las Vegas, NV, 1996.

[33] C.N. Zeeb and P.J. Burns. A Comparison of Failure Probability Estimates by Monte Carlo Sampling and Latin HyperCube Sampling. Dept. Mechanical Engineering, Colorado State Univ., 1998, Report for Sandia Nat. Labs., http://www.colostate.edu/ pburns/monte/documents.html.

AUTHORS

Dr. Carlos Carreras; Departamento de Ingeniería Electrónica E.T.S.I. Telecomunicación; Universidad Politécnica de Madrid; 28040 Madrid, SPAIN

Internet (e-mail): carreras@die.upm.es

Carlos Carreras received the engineering degree in 1986 from the Universidad Politécnica de Madrid (U.P.M.) and the M.S. degree in 1989 from the University of Texas at Austin, both in Electrical Engineering. He obtained his PhD degree also in Electrical Engineering in 1993 again from the U.P.M. He has worked as staff engineer for Bull-Madrid in 1987 and for Schlumberger-Austin from 1989 to 1991. Since then, he is an Associate Professor of Digital Electronics and Systems at the U.P.M., in Madrid, Spain.

Dr. Ian D. Walker; Department of Electrical and Computer Engineering; Clemson University; Clemson, SC 29634 USA

Internet (e-mail): ianw@ces.clemson.edu

Ian D. Walker received the B.Sc. in Mathematics from the University of Hull, England, in 1983. He received the M.S. in 1985, and the Ph.D. in 1989, both in Electrical Engineering, from the University of Texas at Austin. From 1989 to 1997 he was in the Department of Electrical/Computer Engineering at Rice University, in Houston, Texas, where he held the positions of Assistant and Associate Professor. Since 1997 he has been an Associate Professor in the Department of Electrical/Computer Engineering at Clemson University, in Clemson, SC.