An Evolutionary Algorithm based Approach to Design Optimization using Evidence Theory

Rupesh Kumar Srivastava
Graduate Student
Department of Mechanical Engineering
Indian Institute of Technology Kanpur
PIN 208016, U.P., India
Email: rupeshks@iitk.ac.in

Kalyanmoy Deb
Professor
Department of Mechanical Engineering
Indian Institute of Technology Kanpur
PIN 208016, U.P., India
Email: deb@iitk.ac.in

Rupesh Tulshyan
Research Associate
Kanpur Genetic Algorithms Laboratory
Indian Institute of Technology Kanpur
PIN 208016, U.P., India
Email: tulshyan@iitk.ac.in

KanGAL Report Number 2011006

For problems involving uncertainties in design variables and parameters, a bi-objective evolutionary algorithm (EA) based approach to design optimization using evidence theory is proposed and implemented in this paper. In addition to a functional objective, a plausibility measure of failure of constraint satisfaction is minimized. Despite some interests in classical optimization literature, this is the first attempt to use evidence theory with an EA. Due to EA's flexibility in its operators, non-requirement of any gradient, its ability to handle multiple conflicting objectives, and ease of parallelization, evidence-based design optimization using an EA is promising. Results on a test problem and a couple of engineering design problems show that the modified evolutionary multi-objective optimization (EMO) algorithm is capable of finding a widely distributed trade-off frontier showing different optimal solutions corresponding to different levels of plausibility failure limits. Furthermore, a single-objective evidence based EA is found to produce better optimal solutions than a previously reported classical optimization procedure. The use of a GPU based parallel computing platform demonstrates EA's performance enhancement around 160 to 700 times in implementing plausibility computations. Handling uncertainties of different types are getting increasingly popular in applied optimization studies and this EA based study should motivate further studies in handling uncertainties.

1 Introduction

Considering the effect of uncertainties is often unavoidable during a design optimization task. This is because the physical realization of the design variables and parameters is generally imprecise and this can lead to an optimal design becoming unusable if any constraints get violated. Reliability based design optimization (RBDO) is the name given to an optimization procedure wherein the reliability of a design is also given due consideration, either as a constraint by limiting it to a minimum value [1–6], or as an additional objective [2, 7].

Uncertainties can be classified as 'aleatory' or 'epistemic' depending on their nature. While an aleatory uncertainty is well defined as a probability distribution, epistemic uncertainty represents our lack of information about the nature of the impreciseness. Thus, a well defined probability distribution may not always be available to handle uncertainties in a design optimization problem. In such a case, the usual RBDO methodologies cannot be used and a new approach which can utilize low amount of information about the uncertainty is required. Few such methods like a Bayesian approach for sample-based information [8] and an evidence theory based approach [9, 10] for interval-based information have been proposed, but no study has explored them in an EA-context, except an initial proposal of the authors of this study [11].

In this work, we use the evidence-based approach, assuming that information about the uncertainty is available as evidence of the uncertain variables lying within definite intervals around the nominal value. We then propose that instead of constraining the plausibility of failure to a predefined value, a bi-objective approach may be adopted in order to get a better trade-off information about plausibility and objective values, and discuss other benefits of using an evolutionary algorithm for evidence-based design optimization. We also demonstrate parallelization of the algorithm on a Graphical Processing Unit (GPU), achieving a considerable speed-ups (around 160 to 700 times) which make the proposed EA approach even more practically beneficial.

2 RBDO Problem Formulation

A general formulation of a deterministic optimization problem is given below:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x},\mathbf{p}) \\ \text{subject to:} \quad & g_j(\mathbf{x},\mathbf{p}) \geq 0,\, j = 1,\ldots,J \end{aligned} \tag{1}$$

In this formulation, $\mathbf{x}$ are the n design variables that are varied in the optimization while $\mathbf{p}$ are the m design parameters that are kept fixed. Both $\mathbf{x}$ and $\mathbf{p}$ are assumed to be real-valued. The objective is to minimize a function f subject to J inequality constraints. We do not consider equality constraints here because reliability is not defined for equality constraints in this context.

In reliability-based optimization, uncertainties in the design are embodied as random design variables $\mathbf{X}$ and random design parameters $\mathbf{P}$, and the problem is formulated as:

$$\begin{aligned} \underset{\mu_{\mathbf{X}}}{\text{minimize}} \quad & f(\mu_{\mathbf{X}},\mu_{\mathbf{P}}) \\ \text{subject to:} \quad & Pr[g_j(\mathbf{X},\mathbf{P}) \geq 0] \geq R_j,\, j = 1,\ldots,J \end{aligned} \tag{2}$$

The objective of the problem is to minimize f with respect to the means (μ's) of the random variables given the means of the random parameters. The problem is subject to the constraints that the probability of design feasibility is greater than or equal to R_j, for all $j = 1,\ldots,J$, where R_j is the target reliability for the j^{th} probabilistic constraint. A solution to a reliability-based optimization problem is called an optimal-reliable design.

If the uncertainty in the variables and parameters can be confidently expressed as probability distributions (aleatory uncertainty), the above RBDO formulation is sufficient for reliability analysis. However, it is often found that the uncertainty associated with the variables and parameters of a design optimization problem cannot be expressed as a probability distribution. This is because the only information available might be a certain values of physical realizations of the variables, or expert opinions about the uncertainty. The above RBDO formulation cannot utilize this type of information and therefore, a different approach to uncertainty analysis is called for.

3 Handling Incomplete Information

As stated earlier, several RBDO techniques have been proposed and extensively investigated assuming the uncertainties to be aleatory, having a well-defined, exact probability distribution. In practice too, the general approach is either to ignore the lack of knowledge about epistemic uncertainties, or deal with them in an expensive but still inaccurate manner. The approach labeled 'a' in Figure 1 is generally adopted, wherein the samples of variables and parameters considered epistemic are fit to probability distributions so as to combine them with the aleatory ones, and facilitate a conventional RBDO approach (typically Monte Carlo simulations are performed).

The above approach is not preferable since it does not capture the aspect of incomplete information about the uncertainties effectively. A large number of samples are required to fit any probability distribution with confidence over the data, which is expensive. The uncertainty itself may not be of the form of the general distributions assumed, and this approach will lead to misrepresentation of the uncertainty in such a case. Also, there is no clear cut relationship between the information available and the reliability of results obtained. For the above reasons, the methods labeled 'b' and 'c' are better for handling epistemic uncertainties. These new approaches enable effective handling of epistemic uncertainties, and at the same time provide a well-defined link between the amount of knowledge about the uncertainty and the results obtained. The method 'b' was described in an earlier study [11] while the method 'c' is discussed in this paper, and is an evidence theory based approach, which we discuss next.

4 Basics of Evidence Theory

Evidence theory has recently been used for analysis of design reliability when very limited information about the uncertainty is available, usually in the form of expert opinions. Before we proceed to describe an evidence based design optimization (EBDO) procedure using an evolutionary algorithm, we outline the fundamentals of evidence theory as discussed in [9] in this section.

Evidence theory is characterized by two classes of fuzzy measures, called the *belief* and *plausibility* measures. They are mutually dual, and thus each can be uniquely determined from the other. The plausibility and belief measures act as upper and lower bounds of classical probability to measure the likelihood of events without use of explicit probability distributions. When the plausibility and belief measures are equal, the general evidence theory reduces to the classical probability theory. Therefore, the classical probability theory is a special case of evidence theory. A recapitulation of the basics of fuzzy set theory is essential to understand the nature of plausibility and belief measures.

A universe X represents a complete collection of elements having the same characteristics. The individual elements in the universe X, called singletons, are denoted by x. A set A is a collection of some elements of X. All possible subsets of X constitute a special set called the power set $\wp$.

A fuzzy measure is defined by a function $g : \wp(X) \to [0,1]$. Thus, each subset of X is assigned a number in the unit interval $[0,1]$. The assigned number for a subset $A \in \wp(X)$, denoted by $g(A)$, represents the degree of available evidence or belief that a given element of X belongs to the subset A.

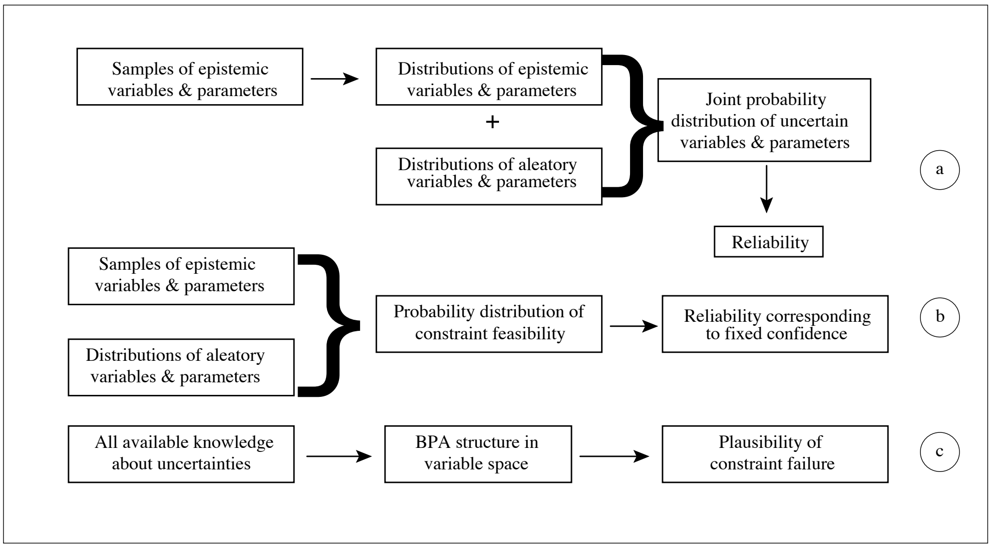
Samples of epistemic variables & parameters
Distributions of epistemic variables & parameters
+
Distributions of aleatory variables & parameters
Joint probability distribution of uncertain variables & parameters
Reliability
a
Samples of epistemic variables & parameters
Distributions of aleatory variables & parameters
Probability distribution of constraint feasibility
Reliability corresponding to fixed confidence
b
All available knowledge about uncertainties
BPA structure in variable space
Plausibility of constraint failure
c

Fig. 1. Various methods of handling incomplete information in RBDO.

In order to qualify as a fuzzy measure, the function g must have certain properties. These properties are defined by the following axioms:

1. $g(\emptyset)=0$ and $g(X)=1$.
2. For every $A,B\in\wp(X)$, if $A\subseteq B$, then $g(A)\leq g(B)$.
3. For every sequence $(A_i\in\wp(X), i=1,2,\ldots)$ of subsets of $\wp(X)$, if either $A_1\subseteq A_i\subseteq\ldots$ or $A_1\supseteq A_2\supseteq\ldots$ (the sequence is monotonic), then $\underset{i\to\infty}{lim} g(A_i)=g(\underset{i\to\infty}{lim} A_i)$.

A belief measure is a function $Bel:\wp(X)\to[0,1]$ which satisfies the three axioms of fuzzy measures and the following additional axiom:

$$Bel(A_1\cup A_2)\geq Bel(A_1)+Bel(A_2)-Bel(A_1\cap A_2). \quad (3)$$

Similarly, A plausibility measure is a function $Pl:\wp(X)\to[0,1]$ which satisfies the three axioms of fuzzy measures and the following additional axiom:

$$Pl(A_1\cap A_2)\leq Pl(A_1)+Pl(A_2)-Pl(A_1\cup A_2). \quad (4)$$

Being fuzzy measures, both belief and its dual plausibility measure can be expressed with respect to the non-negative function:

$$m:\wp(X)\to[0,1], \quad (5)$$

such that $m(\emptyset)=0$, and

$$\sum_{A\in\wp(X)} m(A)=1. \quad (6)$$

The function m is referred to as basic probability assignment (BPA). The basic probability assignment $m(A)$ is interpreted either as the degree of evidence supporting the claim that a specific element of X belongs to the set A or as the degree to which we believe that such a claim is warranted. Every set $A\in\wp(X)$ for which $m(A)\geq 0$ (evidence exists) is called a focal element. Given a BPA m, a belief measure and a plausibility measure are uniquely determined by

$$Bel(A)=\sum_{B\subseteq A} m(B). \quad (7)$$

$$Pl(A)=\sum_{B\cap A\neq 0} m(B). \quad (8)$$

In Equation (7), $Bel(A)$ represents the total evidence corresponding to all the subsets of A, while the $Pl(A)$ in Equation (8) represents the sum of BPA values corresponding to all the sets B intersecting with A. Therefore,

$$Pl(A)\geq Bel(A). \quad (9)$$

Several theories have been proposed to combine evidence obtained from independent sources or experts. If the BPAs m_1 and m_2 express evidence from two experts, the combined evidence m can be calculated using Dempster's rule of combining:

$$m(A)=\frac{\sum\limits_{B\cap C=A} m_1(B)m_2(C)}{1-K}, \quad \text{for } A\neq 0, \quad (10)$$

where

$$K=\sum_{B\cap C=0} m_1(B)m_2(C) \quad (11)$$

represents the conflict between the two independent experts. Other methods of combining evidence may also be used.

5 EBDO Problem Formulation and EA-based Solution Approach

In the previous section, the fuzzy measure of plausibility was introduced which can be used to represent the degree to which the available evidence supports the belief that an element belongs to a set or overlapping sets. In [9], an evidence theory based approach to handle uncertainties in a design problem is discussed. Each probability constraint of the RBDO problem is replaced with a plausibility constraint, limiting the plausibility of failure for each constraint to a predefined value.

In this paper, we propose taking a different approach which treats the problem as a bi-objective problem due to the crucial role of the uncertainty in design. Since the reliability of the design will be an important factor in choosing a design, a trade-off analysis between objective function value and reliability is desirable. Thus, instead of using a plausibility constraint, the plausibility value of any solution **x** can be converted to a second objective function. Since we are evaluating the plausibility of failure, the plausibility value should then be minimized. For multiple constraints, the maximum plausibility over all constraints $Pl_{\max}$ is chosen to be the second objective. According to [9], the plausibility measure is preferred instead of the equivalent belief measure, since at the optimum, the failure domain for each active constraint will usually be much smaller than the safe domain over the frame of discernment. As a result, the computation of the plausibility of failure is much more efficient than the computation of the belief of safe region.

Using an evolutionary algorithm for such a design task has several advantages. Firstly, evidence based design methods have not been investigated adequately due to the high computational complexity of implementation. Since evolutionary algorithms can be efficiently parallelized, a significant amount of computational time and resources can be saved – a matter which we shall revisit in Section 8. Secondly, evolutionary methods are a good choice for EBDO since the optimization task requires an algorithm which is derivative-free and can handle discrete plausibility values. Thirdly, as discussed before, using a bi-objective evolutionary approach, a set of trade-off solutions can be obtained using a multi-objective EA, which can be very useful to a decision maker. These solutions can in turn be utilized for a post-optimality analysis and further insights into the nature of the problem can be developed.

Using the bi-objective evolutionary approach discussed above, the RBDO problem stated in Equation (2) can be formulated using an evidence theory based approach as follows:

$$\begin{aligned} \underset{\mathbf{X},\mathbf{P}}{\text{minimize}} \quad & f(\mathbf{X},\mathbf{P}), \\ \underset{\mathbf{X},\mathbf{P}}{\text{minimize}} \quad & Pl_{\max}, \\ \text{subject to:} \quad & 0 \le Pl_{\max} \le 1, \end{aligned} \tag{12}$$

where (for $g \ge 0$ type constraints),

$$Pl_{\max} = \max_{j=1}^{J}(Pl\,[g_j(\mathbf{X},\mathbf{P}) \le 0]). \tag{13}$$

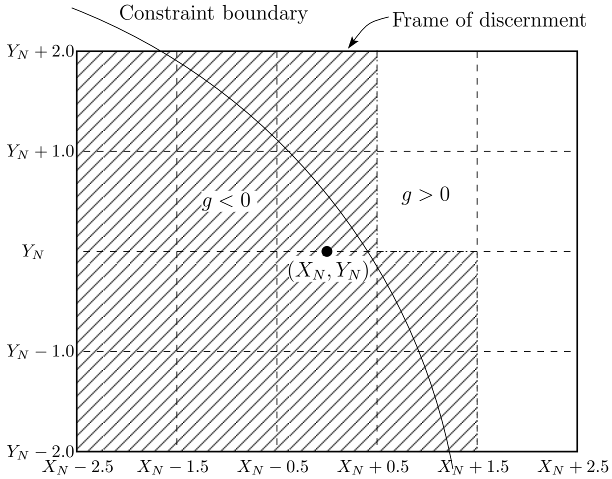

Fig. 2. Focal elements contributing to plausibility calculation.

The plausibility of failure for each member of the population (each design point) must be evaluated in the bi-objective formulation. A step-by-step method for this evaluation is as follows:

1. The frame of discernment (FD) about the design point is identified first. This is the region in the decision variable space for which evidence is available. In a typical BPA structure, the available evidence is expressed corresponding to various intervals of values near the nominal value (denoted by the subscript N) of each variable and parameter. If a conflict exists, a suitable rule for combining evidence must be used. The FD can be viewed as a Cartesian product of these intervals.
2. Each element of the Cartesian product of intervals is referred to as a focal element. This is the smallest unit in the FD for which evidence can be calculated by combining the BPA structures of the variables and parameters. In Figure 2, the focal elements are the rectangles bounded by dashed lines. When the variables/parameters are independent, the BPA of a focal element is simply the product of BPA values corresponding to the intervals that compose it. The BPA structure for the entire FD is thus obtained.
3. According to Equation 8, the focal elements that completely or partially lie in the failure region contribute to the plausibility calculation. Similarly, the focal elements lying completely in the failure region contribute to the belief calculation (Equation 7). This calculation must be performed for each problem constraint. To check if failure occurs within a focal element, the minimum value of the constraint function in the focal element is searched for. If the minimum is negative (for greater-than-equal-to type constraints), then failure occurs in the focal element. Various issues related to identifying the focal elements for plausibility calculation are discussed in Section 5.1.
4. The combined BPA values for the focal elements identified for plausibility calculation are summed to obtained the plausibility for the failure region according to Equa-

tion 8.

5. The maximum plausibility value over all constraints is obtained as the second objective function value to be minimized, according to Equation 13.

The illustration in Figure 2 shows the FD around a candidate point for a 2-D problem lying close to the constraint boundary in the infeasible region. The dashed lines divide the FD into focal elements according to the intervals of the BPA structure along each axis. Thus a combined BPA structure is built for the FD. The shaded focal elements contribute to the plausibility calculation since inside these elements, the constraint becomes infeasible. During an optimization procedure, this FD will be built around each candidate solution, and the contributions to plausibility from the infeasible focal elements will be summed to yield the plausibility of failure for that point. This value can then be minimized or checked for constraint violation depending on the solution approach.

5.1 Plausibility calculation and related issues

It is evident that the calculation of plausibility is a computation bottle-neck in a practical implementation of the evidence-based design optimization approach, and this is a major reason that such methods have not been investigated or used as much as others in the past. In this section, we discuss some algorithmic modifications to tackle this computational complexity, making the approach less time-consuming and more practical. Later in Section 8 we shall demonstrate the use of a plausible hardware implementation for much faster computation.

An intuitive approach to the problem of reducing computations is to use an algorithm that does not need to search all the individual focal elements, so that several focal elements belonging to the feasible or infeasible region can be identified together. Such a method will circumvent the local search effort in each focal element and is thus expected to reduce computation substantially. For this purpose, subdivision techniques such as [12] can be used to eliminate a portion of the search region in each step. A similar algorithm has been demonstrated for evidence-based design optimization in [9]. However, it must be noted that if the computation cost of the identification of focal elements still remains high and the procedure uses a complex serial procedure, then such an algorithm cannot be made faster using a parallel computing platform.

Parallelization or distributed computing is certainly a viable option for the search process required for plausibility calculation. Focal elements belonging to a frame of discernment can be searched in parallel threads and all the elements contributing to the plausibility calculation can be identified simultaneously. Thus, a high amount of computational speed-up is expected, making the approach most suitable when the information available about the uncertainty is more crisp. In this paper, a GPU (Graphical Processing Unit) based parallelization technique is used for this purpose and is described in Section 8.

Whether a subdivision technique is used or not, a subtask of finding the failure plausibility at a design point is to classify a region (containing one or several focal elements) as feasible or infeasible with respect to all the constraints. As previously stated, this can be done by searching for the minimum value of the constraint function in the region, and checking if it is negative (implying infeasibility for $g() \geq 0$ type constraints). There are various techniques which can be used for this search:

Grid-point evaluation: The constraint function can be evaluated at all points forming a hyper-grid across the dimensions of the search space and then the minimum of the values obtained can be used as the minimum of the constraint function. Although the minimum obtained through this technique will not be accurate, it may be sufficient for checking the negativity condition.

Sampling-based evaluation: Instead of evaluating the constraint at uniform grid-points, techniques like optimum symmetric Latin hypercube sampling [13] may be used to evaluate the constraint at fewer, representative points in the search space. Both the grid-based method using a coarse grid and sampling techniques are suitable when the search regions are not large enough that major variations in the constraint values may be expected.

Local search: A local search for the minimum can be performed using a vertex method, gradient-based method or a solver like KNITRO [14] depending on the complexity of constraints. Some of these methods may significantly raise the computation cost and thus should be carefully chosen. To simplify the evaluation of the constraints, a response surface or kriging models may be generated using sampling techniques as in [15]. For this paper, KNITRO is used to confirm the results obtained from a grid-point evaluation method.

Finally, it is seldom the case that a design optimization task is a black-box operation and no information about the behavior of the constraints is available. Quite often, designers are capable of predicting the behavior of the constraint functions with respect to different variables and parameters, and this information should be utilized effectively to reduce the computation cost for such methods. For example, a deflection constraint might always reach extrema at the variable bounds for the material property, and there is no need to perform a search operation for the region under consideration with respect to this variable. This type of information utilization can significantly reduce the computational effort in practice.

Table 1. BPA Structure for variable x_1 the numerical test problem.

Interval	BPA
$[x_{1,N}$ - 1.0, $x_{1,N}$ - 0.5$]$	4.78 %
$[x_{1,N}$ - 0.5, $x_{1,N}]$	45.22 %
$[x_{1,N}, x_{1,N}$ + 0.5$]$	45.22 %
$[x_{1,N}$ + 0.5, $x_{1,N}$ + 1.0$]$	4.78 %

Table 2. BPA Structure for variable x_2 the numerical test problem.

Interval	BPA
$[x_{2,N}$ - 1.0, $x_{2,N}$ - 0.5$]$	4.78 %
$[x_{2,N}$ - 0.5, $x_{2,N}]$	45.22 %
$[x_{2,N}, x_{2,N}$ + 0.5$]$	45.22 %
$[x_{2,N}$ + 0.5, $x_{2,N}$ + 1.0$]$	4.78 %

6 Numerical Test Problem

We first demonstrate the approach on a two-variable test problem described as follows:

$$
\begin{aligned}
\underset{\mathbf{X}}{\text{minimize}} \quad & f(\mathbf{X}) = X_1 + X_2, \\
\text{subject to:} \quad & g_1 : 1 - \frac{X_1^2 X_2}{20} \leq 0, \\
& g_2 : 1 - \frac{(X_1 + X_2 - 5)^2}{30} - \frac{(X_1 - X_2 - 12)^2}{120} \leq 0, \\
& g_3 : 1 - \frac{80}{X_1^2 + 8X_2 - 5} \leq 0 \\
& 0 \leq X_1 \leq 10, \\
& 0 \leq X_2 \leq 10.
\end{aligned}
\qquad (14)
$$

The deterministic optimum for this problem is at $f^* = 5.179$. To demonstrate evidence based analysis, it is assumed that the variables X_1 and X_2 are epistemic, and the evidence available about the variation of uncertainty corresponding to each can be represented by a BPA structure such as that described in Tables 1 and 2. Due to the uncertainties, the optimal solution to the evidence based design optimization is likely to be worse than the deterministic optimum. For simulation purposes, the value of BPA assigned to a particular interval may be obtained directly from an assumed probability distribution. The subscript N denotes the nominal value for a decision variable or parameter.

The problem is converted to a bi-objective problem as developed in the previous section, where an additional objective of minimizing the plausibility of failure is now considered. The problem is solved using the well-known NSGA-II procedure [16] – a multi-objective evolutionary algorithm. We use a population size of 100 (to obtain as distributed a front as possible) and run it for 100 generations. Other NSGA-II parameters are set as a standard practice, as follows: recombination probability of 0.9, SBX index of 10, polynomial mutation probability of 0.5 and mutation index of 20. The resulting trade-off front is shown in Figure 3.

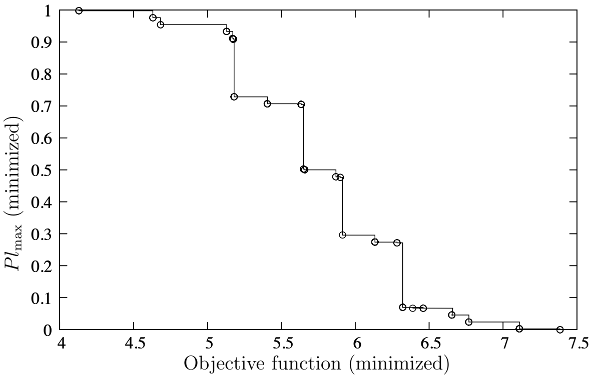

Fig. 3. Results obtained for the two-variable test problem.

It should be noted that the front is discontinuous due to the discrete nature of the BPA structure; only certain values of plausibility are possible, and the front does not exist for intermediate values. Also, several design points will have the same plausibility of failure, and the algorithm is able to find the points for which the first objective is minimized while the second remains constant. The design points with failure plausibility closer to unity lie in the infeasible region with respect to their nominal values, and it is only due to the uncertainty that a few feasible realizations may be expected for these designs.

The jumps in the plausibility values obtained in the final front are due to the jumps in the BPA structures of the variables. To demonstrate this, we modify the BPA structures to be more *smooth*, distributing the high values of evidence for the central intervals as in Table 3. The final front obtained in Figure 4 is seen to be much smoother and without large jumps in plausibility values. This reflects our intuition that more information about the uncertainty distribution will lead to better decision making.

6.1 Post-optimality analysis for choosing preferred points

The mirrored ‘S’ shaped form of the trade-off frontier is evident from these plots. Such fronts usually offer some preferred points, if particularly the mirrored ‘S’ curve is steep, as in Figure 4. We discuss this aspect here.

Figure 5 shows a fitted mirrored ‘S’ shaped curve on the front shown in Figure 4, as a trade-off frontier for our discussion here. For choosing a preferred set of solutions from a trade-off frontier between two objectives f_1 and f_2, one of the popular methods is the use of a linear utility function [17]

Table 3. Modified BPA structure for the numerical test problem.

x_1

Interval	BPA
$[x_{1,N}$ - 1.0, $x_{1,N}$ - 0.5]	4.78 %
$[x_{1,N}$ - 0.5, $x_{1,N}$ - 0.25]	15.22 %
$[x_{1,N}$ - 0.25, $x_{1,N}]$	30 %
$[x_{1,N}, x_{1,N}$ + 0.25]	30 %
$[x_{1,N}$ + 0.25, $x_{1,N}$ + 0.5]	15.22 %
$[x_{1,N}$ + 0.5, $x_{1,N}$ + 1.0]	4.78 %

x_2

Interval	BPA
$[x_{2,N}$ - 1.0, $x_{2,N}$ - 0.5]	4.78 %
$[x_{2,N}$ - 0.5, $x_{2,N}$ - 0.25]	15.22 %
$[x_{2,N}$ - 0.25, $x_{2,N}]$	30 %
$[x_{2,N}, x_{2,N}$ + 0.25]	30 %
$[x_{2,N}$ + 0.25, $x_{2,N}$ + 0.5]	15.22 %
$[x_{2,N}$ + 0.5, $x_{2,N}$ + 1.0]	4.78 %

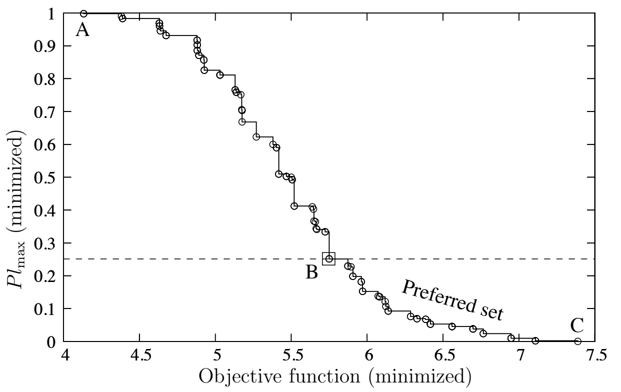

Fig. 4. Results obtained for the modified two-variable test problem.

of the following type:

$$\mathcal{U}(\mathbf{f}) = w_1 f_1 + w_2 f_2, \qquad (15)$$

where (w_1, w_2) is an arbitrary non-zero weight vector (with non-negative weights). For a particular weight vector, the point that minimizes the above utility function becomes the preferred trade-off point. Using this argument, it can be shown that for all non-negative weight combinations (except $(w_1, w_2) \neq (0,0)$), the mirrored 'S' shaped trade-off frontier will make the minimum-f_1 point (A) and a region near the minimum-f_2 points as preferred points. The point B is such a point on the trade-off frontier that the tangent at B passes through A. This is because for any weight vector that causes a slope (w_1/w_2) of contour line of $\mathcal{U}$ higher than the slope of line AB, the resulting preferred point is A and for a weight vector that that causes a slope smaller than that of line AB the preferred point lies in the range BC. A representative scenario is illustrated in the figure. When the weight vector causes the slope to be equal to the slope of line AB, the resulting preferred point is either A or B. Thus, from the perspective of a linear utility function, the preferred point is either the minimum-f_1 point (if allowed, ignoring objective f_2) or a point in the vicinity of the minimum-f_2 point.

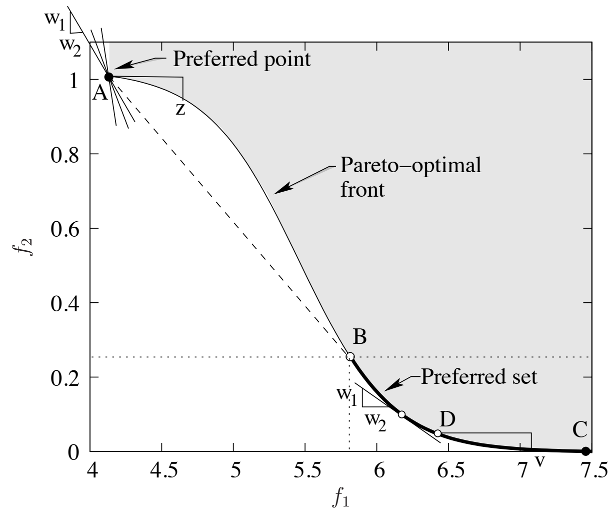

Fig. 5. A mirrored 'S' shaped Pareto-optimal front and likely preferred point (A) and set (BC) are illustrated. The point B is obtained by drawing a tangent line from A on to the Pareto-optimal front.

We now discuss its implications for the evidence-based design optimization point of view. Solutions near the minimum-f point (deterministic optimum of f, that is, point A) are usually close to boundary of certain constraints and would make large failures due to uncertainties in decision variables and parameters (having a large failure plausibility value). However, any small improvement in failure plausibility (from moving from point A to say, point z, shown in the figure) demands for a large sacrifice of f. Thus, in the neighborhood of the minimum-f solution, there is not much motivation for the designer to choose any point (such as point z) other than the minimum-f point. Since the neighborhood of the minimum-f solution is observed to be non-convex, the extreme points of a non-convex front are usually the preferred choices [18]. But since the minimum-f solution ignores the second objective, it may not be preferable from a consideration of both objectives. Thus, we turn to the region near minimum failure plausibility point for a trade-off preferred solution. This region (B to C) corresponds to low values of plausibility of failure, despite having to sacrifice somewhat on the weight of the structure. Again, the trade-off beyond another point D may be such that a small gain in failure plausibility value comes from a large sacrifice from weight, as shown with a point v from the front. We argue that choosing a particular solution from the resulting region BD is user-specific, but this study from a linear utility function perspective narrows down the choice for choosing a preferred solution from the entire non-dominated range of AC to BD.

The location of the point B is difficult to find for a non-smooth trade-off frontier, like the one shown in Figure 4. However, instead of computing the gradient of the frontier, we locate the point B using the following procedure. For every point $\mathbf{z}$ on the trade-off frontier, we create a line joining the minimum-f_1 point (A) and the point $\mathbf{z}$. If all other trade-off points lie on or above this line, then the point $\mathbf{z}$ becomes the preferred point B. For the modified numerical test problem, this procedure yields the point B marked in Figure 4. This point corresponds to a maximum plausibility of failure of 25.11% having $f_1 = 5.750$. Thus, our analysis using a linear utility function theory, any solution having a plausibility of failure of 25.11% or less is preferred and it now depends on the designer to choose from this narrow set of trade-off points found by NSGA-II procedure. The location of D can also be found using the methods described in a recent knee-based study [18], but in this paper we do not discus this aspect.

7 Results for Engineering Design Problems

In this section, we apply the evidence based EA optimization procedure to a couple of engineering design problems.

7.1 Cantilever beam design

The first engineering design problem we solve is a cantilever beam design for vertical and lateral loading [19]. In this problem, a beam of length $L = 100$ in. is subjected to a vertical load Y and lateral load Z at its tip. The objective of the problem is to minimize the weight of the beam which can be represented by $f = w \times t$, where w is the width and t the thickness of the beam (both represented in inches).

The problem is modeled with two constraints which represent two non-linear failure modes. The first mode is represented by failure at the fixed end of the beam, while the second mode is the tip displacement exceeding a maximum allowed value of $D_0 = 2.5$ in. The deterministic optimization problem formulation is given as follows:

$$\begin{aligned}
\underset{w,t}{\text{minimize}} \quad & f = wt, \\
\text{subject to:} \quad & g_1 : \sigma_y - \left(\frac{600Y}{wt^2} + \frac{600Z}{w^2t}\right) \geq 0, \\
& g_2 : D_0 - \frac{4L^3}{Ewt}\sqrt{\left(\frac{Y}{t^2}\right)^2 + \left(\frac{Z}{w^2}\right)^2} \geq 0, \\
& 0 \leq w \leq 5, \\
& 0 \leq t \leq 5.
\end{aligned} \tag{16}$$

To demonstrate the principle of the evidence theory based EA approach, the variables w and t are taken to be deterministic, while the parameters Y = vertical load (lb), Z = lateral load (lb), σ_y = yield strength (psi) and E = Young's modulus (psi) are assumed to be epistemic. In general, expert opinions and prior information about the uncertainty will yield a BPA structure for the epistemic parameters. However, in order to facilitate comparison with an RBDO result, the distributions of the parameters as used in an RBDO study [20] are used to obtain the BPA structure for them (Table 4). Thus, normal distributions $Y \sim N(1000, 100)$ lb, $Z \sim N(500, 100)$ lb, $\sigma_y \sim N(40000, 2000)$ psi and $E \sim N(29(10^6), 1.45(10^6))$ psi are assumed and the area under the corresponding PDF for each interval is taken as the BPA for each parameter. Since these epistemic parameters are not decision variables in this problem, we do not need to use the subscript 'N' for them.

Table 4. BPA Structure for the cantilever design problem

Z (lb)		σ_y ($\times 10^3$ psi)	
Interval	BPA	Interval	BPA
[200, 300]	2.2 %	[35, 37]	6.1 %
[300, 400]	13.6 %	[37, 38]	9.2 %
[400, 450]	15 %	[38, 39]	15 %
[450, 500]	19.2 %	[39, 40]	19.2 %
[500, 550]	19.2 %	[40, 41]	19.2 %
[550, 600]	15 %	[41, 42]	15 %
[600, 700]	13.6 %	[42, 43]	9.2 %
[700, 800]	2.2 %	[43, 45]	7.1 %

Y (lb)		E ($\times 10^6$ psi)	
Interval	BPA	Interval	BPA
[700, 800]	2.2 %	[26.5, 27.5]	10 %
[800, 900]	13.6 %	[27.5, 28.5]	21 %
[900, 1000]	34.1 %	[28.5, 29]	13.5 %
[1000, 1100]	34.1 %	[29, 29.5]	13.5 %
[1100, 1200]	13.6 %	[29.5, 30.5]	21 %
[1200, 1300]	2.4 %	[30.5, 31.3]	21 %

The problem is converted to the bi-objective formulation developed in Section 5, and solved using a population size of 60 for 100 generations of NSGA-II. The resulting trade-off is shown in Figure 6. The mirrored 'S' shaped front is obtained.

Table 6. Cantilever design: Comparison of results for $\mathbf{p_{lim}}$ = 0.0013. NSGA-II solution makes $Pl(g_1)$ constraint have a value close to this limit.

Variable	Original [9]	NSGA-II
w (in)	2.5298	2.4142
t (in)	4.1726	4.1247
	Values obtained	
$Pl(g_1)$	0.000032	**0.001274**
$Pl(g_2)$	0.000000	0.000000
f (in^2)	10.556	**9.957998**

7.1.1 Comparison with an earlier study

In order to compare to the results with a previous study [9], which used a single objective of minimizing f alone, NSGA-II is used to solve for the single-objective formulation, setting the limit of failure plausibility for each constraint equal to p_{lim} as follows:

$$\begin{aligned} \underset{w,t}{\text{minimize}} \quad & f = wt, \\ \text{subject to:} \quad & Pl_{\max} \leq p_{lim}, \\ & 0 \leq w \leq 5, \\ & 0 \leq t \leq 5. \end{aligned} \tag{17}$$

The NSGA-II results are compared with the previous results [9] in Table 5. It is notable that NSGA-II finds a better design point for all three values of p_{lim}, demonstrating the non-optimality of previously reported solutions and a better performance of an evolutionary algorithm for this complex optimization task. The plausibility and objective function values obtained for $p_{lim} = 0.0013$ corresponding to the design point reported in [9] and the point obtained by NSGA-II are compared in Table 6. The superiority of the NSGA-II solution comes from the fact that NSGA-II solution is able to find a solution for which the plausibility failure of first constraint is almost same as the desired p_{lim}, whereas the solution reported in the existing study is far from this limit.

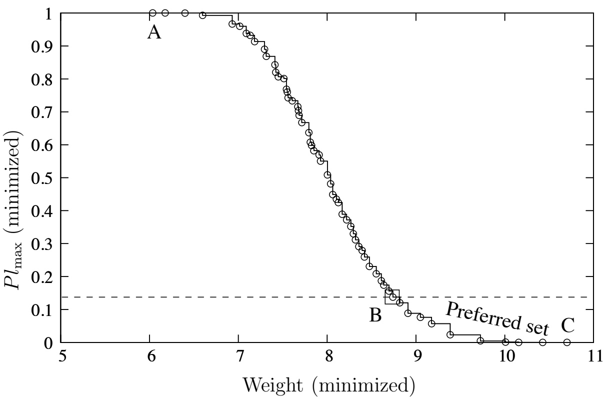

Fig. 6. Obtained trade-off front for the cantilever design problem.

Table 7. Five preferred solutions of the cantilever design problem, as an outcome of the post-optimality analysis.

w	t	Weight	$Pl_{\max}$
(in)	(in)	(in^2)	(%)
2.160	4.047	8.740	13.736
2.287	3.897	8.911	8.840
2.305	4.072	9.387	2.299
2.410	4.036	9.726	0.472
2.701	3.706	10.009	0.127

As expected, the design points obtained under uncertainty are worse than the deterministic optimum, since objective function value must be sacrificed in order to gain in the reliability of the design. Also, the value of weight obtained for $p_{lim} = 0.0013$ (corresponding to $R = 0.9987$) for the RBDO study is worse than the reliable optimum. This is because evidence theory based optimization uses less information about uncertain variables/parameters than that in RBDO, and this loss of information affects the optimum weight value that can be obtained by the EBDO approach.

7.1.2 A post-optimality analysis

A post-optimality analysis (discussed in subsection 6.1) reveals that the designer is better-off choosing a preferred solution having a plausibility of failure smaller than or equal to 13.74%, which corresponds to a weight of 8.74 in^2. This solution is marked as point B in Figure 6. From a set of 58 different trade-off solutions obtained by NSGA-II, only 11 solutions are recommended by the post-optimality analysis for further processing. These solutions correspond to a plausibility of failure ranging from 13.74% to 0%, with corresponding weights ranging from 8.74 in^2 to 10.70 in^2. This is a significant reduction in choice of solutions for a final subjective decision-making task. Investigating further these 11 solutions, five preferred points with good trade-offs are chosen and presented in Table 7. It is interesting to note that in order to make the designs more safe against uncertainties in four problem parameters, the width (w) must be made bigger, whereas the thickness (t) needs to be fixed at a high value.

7.2 Pressure vessel design

The second engineering design problem is design of a pressure vessel [9]. The objective of the problem is to maximize the volume of the pressure vessel which is a function of design variables R, the radius and L, the mid-section length. A third design variable is the thickness t of the pressure vessel wall. All three design variables are considered epistemic here, thereby representing their nominal values with a subscript 'N'. The problem is modeled with five constraints which represent failure in the form of yielding of material

Table 5. Comparison of results for the Cantilever design problem.

	Evidence theory based results				Reliable optimum	Deterministic optimum
	Algorithm	$p_{lim} =$ 0.2	$p_{lim} =$ 0.1	$p_{lim} =$ 0.0013	$R = 0.9987$	
Weight	Original [9]	8.6448	10.217	10.556	9.5212	7.6679
	NSGA-II	8.5751	8.8832	9.9580		

in circumferential and radial directions, and violation of geometric constraints. The deterministic problem formulation is as follows:

$$\begin{aligned}
\underset{R_N, L_N, t_N}{\text{maximize}} \quad & f = \frac{4}{3}\pi {R_N}^3 + \pi {R_N}^2 L_N, \\
\text{subject to:} \quad & g_1 : 1.0 - \frac{P(R+0.5t)SF}{2t\sigma_y} \geq 0, \\
& g_2 : 1.0 - \frac{P(2R^2+2Rt+t^2)SF}{(2Rt+t^2)\sigma_y} \geq 0, \\
& g_3 : 1.0 - \frac{L+2R+2t}{60} \geq 0, \\
& g_4 : 1.0 - \frac{R+t}{12} \geq 0, \\
& g_5 : 1.0 - \frac{5t}{R} \geq 0, \\
& 0.25 \leq t_N \leq 2, \\
& 6 \leq R_N \leq 24, \\
& 10 \leq L_N \leq 48.
\end{aligned} \qquad (18)$$

Besides the design variables, the parameters internal pressure P, and yield strength σ_y are also assumed to be epistemic for this problem (making a total of five epistemic variables). The assumed normal distribution for P and σ_y are $P \sim N(1000, 50)$ and $\sigma_y \sim N(260000, 13000)$, respectively, while the normal distributions for R, L and t are assumed to have standard deviations equal to 1.5, 3.0 and 0.1, respectively. The factor of safety SF is considered to be 2. The units of these parameters were not specifically mentioned in the original study [9] and the chosen magnitudes are difficult to justify using any standard unit system. Thus, in this problem, we refrain from using any unit for any of the problem parameters. Keeping the problem identical to that in the original study helps us compare the results of our study with the original study. Importantly, the absence of a unit system does not come on the way of demonstrating the working of our proposed methodology.

Tables 8 and 9 show the BPA structure used for variables and parameters of the problem [9]. The bi-objective formulation of the problem is solved using a population size of 60 for 100 generations of NSGA-II and the resulting trade-off is shown in Figure 7. The observed jumps in the front are expected due to the jumps in the BPA structure of the problem variables, as explained earlier in Section 6.

Table 9. BPA Structure for the pressure vessel design problem (parameters).

P	σ_y	BPA
[800, 850]	[208000, 221000]	0.13%
[850, 900]	[221000, 234000]	2.15%
[900, 1000]	[234000, 260000]	47.72%
[1000, 1100]	[260000, 286000]	47.72%
[1100, 1150]	[286000, 299000]	2.15%
[1150, 1200]	[299000, 312000]	0.13%

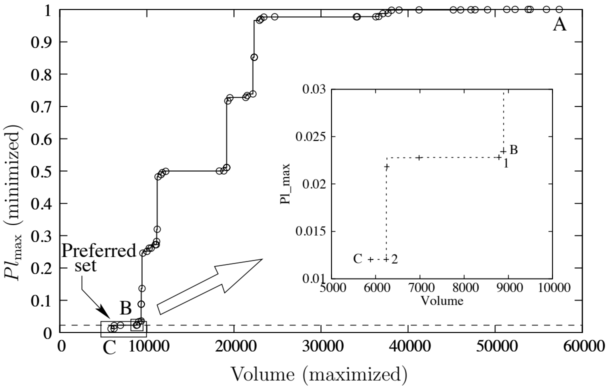

Fig. 7. Obtained trade-off front for the Pressure Vessel design problem.

7.3 Comparison with existing studies

A similar comparison as in the previous example shows that NSGA-II is able to obtain better solution in terms of objective function value for the same value of p_{lim} than that reported in the original study [9]. Table 10 shows that the results obtained are consistent with expected values, being worse the deterministic optimum as well as the corresponding reliability based optimum ($p_{lim} = 0.015$ corresponding to $R = 0.985$), reflecting the lack of information about the uncertainty.

Table 11 compares the design point reported previously [9] and the design point obtained by NSGA-II in terms of plausibility and objective function values.

Table 8. BPA Structure for the pressure vessel design problem (variables).

R	L	t	BPA
[R_N - 6.0, R_N - 4.5]	[L_N - 12, L_N - 9]	[t_N - 0.4, t_N - 0.3]	0.13%
[R_N - 4.5, R_N - 3.0]	[L_N - 9, L_N - 6]	[t_N - 0.3, t_N - 0.2]	2.15%
[R_N - 3.0, R_N]	[L_N - 6, L_N]	[t_N - 0.2, t_N]	47.72%
[R_N, R_N + 3.0]	[L_N, L_N + 6]	[t_N, t_N + 0.2]	47.72%
[R_N + 3.0, R_N + 4.5]	[L_N + 6, L_N + 9]	[t_N + 0.2, t_N + 0.3]	2.15%
[R_N + 4.5, R_N + 6.0]	[L_N + 9, L_N + 12]	[t_N + 0.3, t_N + 0.4]	0.13%

Table 10. Comparison of results for the Pressure Vessel design problem.

		Evidence theory based results			Reliable Optimum	Deterministic optimum
	Algorithm	$p_{lim} = 0.3$	$p_{lim} = 0.2$	$p_{lim} = 0.015$	$R = 0.985$	
Volume	Original [9]	1.101e4	0.9053e4	0.6137e4	1.605e4	2.240e4
	NSGA-II	1.126e4	0.9495e4	0.6307e4		

Table 11. Pressure vessel design: Comparison of results for $\mathbf{p_{lim}}$ = 0.015. NSGA-II solution makes $Pl(g_2)$ constraint have a value close to this limit.

Variable	Original [9]	NSGA-II
R	7.074	7.127
L	29.626	30.019
t	0.413	0.364
	Values obtained	
$Pl(g_1)$	0.001298	0.001301
$Pl(g_2)$	0.001318	**0.014732**
$Pl(g_3)$	0.012020	0.012020
$Pl(g_4)$	0.012050	0.012050
$Pl(g_5)$	0.012020	0.012020
f	6.137e3	**6.307e3**

Table 12. Three preferred solutions of the pressure vessel design problem, as an outcome of the post-optimality analysis.

Soln.	R	L	t	Volume	$Pl_{\max}$
B	8.296	30.074	0.324	8894.9	2.34
1	8.251	30.074	0.338	8784.4	2.28
2	7.118	29.732	0.382	6242.6	1.21

7.4 A post-optimality analysis

A post-optimality analysis reveals that designs having a plausibility of failure smaller than or equal to 2.34% are preferred. This limiting design corresponds to a volume of 8894.9. This design is marked in Figure 7. From a set of 58 different trade-off solutions, the linear utility based theory reduces the choice to only six trade-off solutions (plausibility of failure of 1.20% to 2.34% with corresponding volumes ranging from 5881.6 to 8894.9). The inset plot in Figure 7 marks these solutions. Of six solutions, solutions B, 1 and 2 have significant trade-off among them and are presented in Table 12. It is quite interesting to note that in order to reduce the failure plausibility in design, the thickness of the pressure vessel t must have to be increased, whereas the radius R and length L of the vessel must be reduced. This pattern of design modifications to improve against uncertainties – reduce the volume of vessel with an increased thickness of the shell – derived from the optimization study are valuable for the designers in gaining useful insights to the pressure vessel design problem.

8 Parallelization of Evolutionary Evidence-based Design Optimization

As mentioned earlier, EAs have an inherent potential to be parallelized due to their population approach. In this section, we discuss the areas of parallelization using an emerging GPU based computing platform.

8.1 Motivation

Parallelism is inherent in any evolutionary algorithm by its design. The population-based approach and the existence of operators working independently on mutually exclusive subsets of population data, makes these algorithms amenable to parallelizations of all sorts, besides handing them a key advantage over classical algorithms which cannot be so easily parallelized. It is not a surprise therefore, that this aspect of EAs has attracted abundant attention from researchers in the past and continues to be an active research field.

It has to be however, noted that the benefits of parallelization can be obtained either by designing parallel evolutionary algorithms or by parallelizing computationally heavy sections of the existing implementations. While the two methodologies are both dependent on the parallel hardware architecture and have intersection of ideas, their operational goals differ. The designing of parallel EAs typically involves modifying an original EA algorithm so that it can be implemented efficiently on a particular parallel computing architecture. As such, apart from parallelizing the basic EA tasks, research in this domain focuses on evaluating the performance of EAs which may have multiple populations with varying hierarchies, varying topologies, and different migration strategies. A good survey on such parallel EA models can be found in [21].

On the other hand, if an algorithm spends majority of its time in a particular operation, then it seems prudent to only parallelize the time-consuming operation without modifying the rest of the algorithm. Also, this methodology does not pose questions on the consistency of results and can be faster to implement at times. In the current study on evidence-based design optimization it is observed that the procedure for computing the plausibility term arising from all constraints is highly computation-intensive and consumes more than 99% of the algorithm runtime and therefore is an ideal candidate for parallelization. In the following sections, we first introduce our parallel computing hardware, discuss its intricacies and then show some scale-up results.

8.2 GPU Computing

In recent years, interest has steadily developed in the area of using GPUs for general-purpose computing (GPGPU). A graphics processing unit (GPU) is a SIMD (single instruction multiple device) co-processor unit designed for doing image processing tasks facilitating the rendering of computer generated imagery. Since, the majority of image processing tasks are *data-parallel*[1], the GPUs have traditionally favored multi-threaded many-core architectures, whereby more transistors are devoted to ALUs (Arithmetic Logic Unit) compared to a CPU core at the expense of having fewer cache and control-flow units. Lately, as a result of the growing needs of the gaming industry of simulating more and more immersive and detailed environments, the GPUs have advanced very aggressively. Hence, a GPU of today, though limited in its functionality, is far ahead of the CPUs in terms of the number of cores and hence the raw computational power.

Though the concept of using a GPU for scientific computations is quite old, till recently the restrictions in a GPU in terms of operations and programming made writing GPU codes very complex. The advancements in technology have since then seen the addition of programmable stages, high precision arithmetic and other capabilities to the GPU rendering pipelines, thereby improving their utility. Today, software developers are able to efficiently use the computational powers of GPUs with the help of easy to learn programming APIs, the most popular being Nvidia's CUDA [22].

8.2.1 CUDA Programming

The CUDA programming model has been specifically designed to allow control over the parallel processing capabilities of the GPUs for general purpose applications, and to take GPU computing to the mass. CUDA provides an API which extends the industry standard programming languages like C and Fortran, providing constructs for parallel *threads*, shared memory and barrier synchronization. This ensures that a CUDA programmer can start utilizing the parallel architecture readily after familiarization with these key abstractions without needing to understand the graphics rendering process of a standard Nvidia GPU.

To do computations on the GPU, the user first needs to copy the relevant data to the GPU device. The GPU device has its own DRAM[2] (hereafter referred to as *global memory*) for the storage of large data, required throughout the execution of a program. The user then needs to specify the layout and the number of threads to create, and invoke a GPU method (called as *kernel*). A *kernel* contains the instruction set which is to be simultaneously executed by all the threads, albeit depending on their unique indices. The layout specification includes defining a grid of thread-blocks, wherein each thread-block contains a given number of threads. The arrangement of threads in a thread-block and the arrangement of thread-blocks in a grid, both can be specified to be in a 1D, 2D or 3D lattice (with some restrictions), depending on which three-dimensional unique indices are computed.

During compilation, each thread is allotted the required number of 32-bit memory registers, depending on the *kernel* code. While different thread-blocks function independently, threads inside a thread-block can synchronize their execution and are also allowed to share data using a device-specific limited shared memory. Thus, the user is provided with flexibility of two levels of parallelization, which (s)he needs to efficiently exploit. After computations, the result data needs to be copied back to the CPU memory to be used further.

While the software layer abstraction deals in grids and blocks, the hardware architecture is built around a scalable array of multi-threaded Streaming Multiprocessors (SMs). When a *kernel* is invoked, the blocks of the grid are enumerated and distributed to multiprocessors with available execution capacity (a block cannot be shared by two multiprocessors). A multiprocessor consists of eight scalar processor cores, two special transcendentals, a multi-threaded instruction unit, and on-chip shared memory. The multiprocessor creates, manages, and executes concurrent threads in

[1] the same set of instructions are to be performed for different pieces of distributed data

[2] Dynamic Random Access Memory

hardware with zero scheduling overhead. The threads of a block execute with time-sharing and are scheduled and given instructions in batches of 32 called as *warps*. A *warp* executes one common instruction at a time thereby making data-dependent loops and if-else constructs inefficient within threads in a *warp*. As thread blocks terminate, new blocks are launched on the vacated multiprocessors. Detailed descriptions and illustrations of the architecture are available in [22].

The limitations of a GPU device stem from its limited memory spaces with different latencies and optimal access patters. The threads work in parallel and should agree on their instruction path (avoiding memory conflicts) to give optimal speed-ups. Additionally, GPUs currently are unable to handle double-precision computations efficiently. To sum up, overall performance optimization strategies for a CUDA code include:

1. Structuring the algorithm to exploit maximum parallel execution and high arithmetic intensity.
2. Minimizing data transfers with low-bandwidth which includes transfers between the host and the device and the read-write to the global memory.
3. Achieving global memory coalescing - simultaneous memory accesses by a group of threads can be performed as a single memory transaction. Assuming global memory to be partitioned and aligned into segments of 32, 64 or 128 bytes, coalescing occurs when 16 threads of a half-warp access variables of same bit size, lying in the same segment of global memory.
4. Keeping the multiple cores busy. This includes creating enough number of blocks with enough number of threads and minimizing the resource usage so that concurrent blocks can run on one GPU core.
5. Maximizing the usage of shared memory for frequently used data albeit avoiding access conflicts.
6. Minimizing the number of local variables per thread.
7. Minimizing the branching of threads within a *warp*.
8. Minimizing the need of explicit synchronization among threads.

8.3 Parallel implementation of EBDO

The evidence-based design optimization approach as demonstrated in the previous sections offers several levels of parallelization when an evolutionary approach is used:

1. Each population member (each point at which failure plausibility is to be calculated) can be evaluated independent of others.
2. Failure plausibility for each constraint can be evaluated independently.
3. The focal elements can be checked for feasibility in parallel.

Keeping the GPU architecture in mind, it seems reasonable to distribute the population evaluation among different blocks, as the evaluations can be done independently and in any order. Also, it allows us to effectively use the shared memory to store an individual specific information in a thread-block. Further, at the thread-block level, parallel threads share the task of checking feasibility in different focal elements. Each thread stores the value of each constraint function (for the focal elements in its share) in a specified location in the shared memory, while the focal element is searched for feasibility. The benefit of using shared memory here is that the summation of probabilities over all focal elements can later be done using parallel reduction technique within the *kernel* itself.

The constraint-level parallelization presents a trade-off. While its ideal to handle all constraints simultaneously when searching a focal element, doing so may increase the shared-memory requirements of a thread-block considerably, which may result in lesser number of concurrent thread-blocks per GPU core. On the other hand, distributing different constraints to different thread-blocks means creating more number of thread-blocks which is expected to increase the overall computation time. Also, for sufficiently high number of constraints the shared-memory available might not be enough for all constraints to be handled within a single thread-block. Besides, while the total number of thread-blocks would also depend on the population size, the resources used per thread-block would depend on the number of threads in a block too. Optimal configuration of constraint-level parallelization, hence will vary with problems and can be found only by experimentation.

To sum up, for parallelizing the EBDO approach, it is sufficient to parallelize the objective evaluations, since it takes the majority of time. To do this on GPU, we make a grid of *popsize* thread-blocks containing a pre-specified *tcount* number of threads per block. The threads iteratively find the failure plausibility for each constraint over different focal elements and store it in the shared memory. After summing the values, and finding the constraint with maximum plausibility, the corresponding value is written into the *global memory* for transfer to the host. In the next section, we show the achieved speed-ups and other results of this parallelization study.

8.4 Results using GPUs

We use a Tesla C1060 Nvidia processor with 30 SMs for our simulations. The codes are run on a AMD Phenom(tm) II X4 945 processor system using only one CPU core. It has to be noted that while the modern GPUs are improving the efficiency of double-precision computations, our Tesla GPU is an order of magnitude slower when doing double-precision arithmetic. Hence, single-precision computations are done for the parallel code, whereas the sequential code uses double-precision computations.

8.4.1 Validation

For any parallelization study, it is imperative that the results of the parallel code be validated against the sequential code results. In the current study, the results are not expected to change since only evaluation of an objective has been parallelized without modifying the rest of the algorithm. However, the difference in the computation precision between the

Table 13. Comparison of results of sequential and parallel code for the cantilever problem.

Pop.	$hverr_{ngen}$		
size	$ngen = 25$	$ngen = 50$	$ngen = 100$
60	6.660894e-11	6.217420e-05	5.307487e-05
120	9.190401e-10	1.707969e-10	8.848040e-10
180	6.995547e-06	4.258202e-06	7.442100e-05

Table 14. Comparison of results of sequential and parallel code for the pressure vessel problem.

Pop.	$hverr_{ngen}$		
size	$ngen = 15$	$ngen = 30$	$ngen = 45$
60	4.400588e-03	7.527060e-04	1.817406e-03
120	6.049134e-04	2.142371e-04	2.590431e-04
180	1.498034e-04	2.714034e-04	8.221551e-05

sequential and parallel code may cause some aberrations in results, in which case validation of results is desired.

For this validation, we use the hypervolume measure [23] to compare GA population sets of the sequential and parallel code across different generations. To compute the hypervolume both the objectives are normalized to $[0,1]$ based on their extremum values achieved within the bound constraints, for both the problems. The point $(1.1,1.1)$ is taken as the reference point. Then, taking the absolute sum of difference in hypervolumes as an error, i.e.

$$hverr_i = |Hypervol_{seq_i} - Hypervol_{par_i}|,$$

the corresponding error values are tabulated for different generation levels and different population sizes. Tables 13, 14 show the observations for a particular run of the algorithm.

The results of the hypervolume study suggest that the difference in precision does not affect the evolution of GA population considerably. Henceforth, we will only use the number of generations and population size as the determining criteria for time comparison studies.

8.4.2 Effect of number of threads in CUDA

The number of threads per block (hereafter referred to as *tcount*) is an important parameter to be considered when making parallel code on CUDA. While sufficient number of threads are required to hide the latency of memory access and arithmetic operations, a very high value of *tcount* increases the register usage of a thread-block considerably leading to lower number of concurrent thread-blocks per core. Here, we do a study varying *tcount* to find the optimal number of threads per block. The results for the cantilever problem are shown in Figure 8. It is observed that an evaluation takes longer time for $tcount = 64$ than for $tcount = 256$, therefore for small population sizes where there are not enough blocks to keep the GPU busy, its better to have large number of threads per block. However, with more threads per block, the resource requirement get high and not much gain is obtained by having concurrent blocks running on one core. As the figure shows, for higher population sizes, $tcount = 64$ has a slight edge over $tcount = 256$. Similar results are obtained for the pressure vessel problem. Hence, 64 threads per thread-block is taken as the optimal number for our subsequent simulations.

It is interesting to note that the plot shows jumps at multiples of 30 due to the fact that the Tesla GPU used in our experiments has 30 multiprocessors. Thus, no gain in speed-up is expected when a fraction of a block of 30 multiprocessors are used used in the computation process.

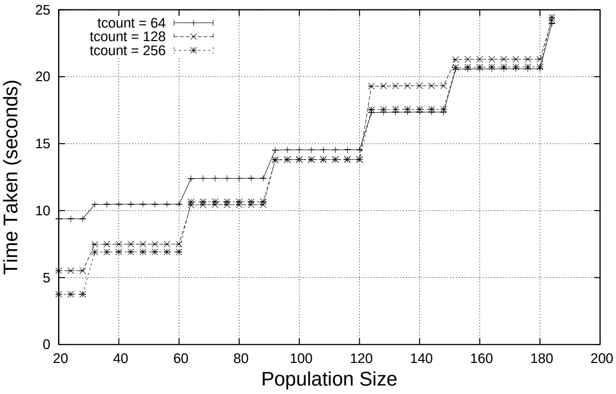

Fig. 8. Effect of varying number of threads per block ($tcount$) for the cantilever problem run for 30 generations.

8.4.3 Speed-ups

Now, we conduct a study of speedups observed for varying population sizes taking $tcount = 64$ as optimal number of threads per block. The results are shown in Table 15 and Table 16. It is observed that for a small population size of 60 the GPU is not being utilized efficiently which results in a low speed-up. For higher population sizes, once we have sufficient number of thread-blocks running, the speed-ups achieved are much high and almost similar. Also, the speed-ups for the pressure vessel problem are much higher owing to the fact that more number of constraints are being evaluated in parallel, thereby providing a large gain in speed-up compared to the serial implementation.

9 Conclusions

In this paper, we have proposed a new evolutionary approach to design optimization using the evidence theory. The use of evidence theory for uncertainty analysis in design optimization has been limited in the past primarily due to the

Table 15. Computational times and speed-ups for the cantilever problem with $tcount = 64$ run for 30 generations.

Popsize	CPU time (secs)	GPU time (secs)	Speed-up
60	1716.78	10.48	163.81
120	3653.55	14.55	251.10
180	5379.99	20.61	261.04
240	7147.29	27.48	260.09
300	9116.30	37.95	240.47

Table 16. Computational times and speed-ups for the pressure vessel problem with $tcount = 64$ run for 10 generations.

Popsize	CPU time (secs)	GPU time (secs)	Speed-up
60	3878.79	11.27	344.17
120	7917.84	13.05	606.73
180	11711.76	17.31	676.59
240	16131.02	22.83	706.57
300	19678.12	33.85	581.33

high computation cost involved. However, due to availability of better optimization algorithms and parallel computing platforms, they are getting more and more tractable and such practice-oriented methods are bound to become more popular. An evolutionary approach also offers the possibility of parallelization and an algorithmic flexibility which can drastically reduce the computation time required for analysis.

Our proposed approach is based on a bi-objective formulation and the solution methodology is based on the NSGA-II approach. Since a number of trade-off solutions can be found by such an approach, designers will have a plethora of information relating to designs having different plausible failure limits. Such information is not only important to choose a single design using a post-optimality analysis, the knowledge of variation of solutions for different limiting failure levels would be most valuable to the designers. Results for numerical and engineering design problems show the effectiveness of the proposed approach, and the capability of an EA to find better solutions in the discrete objective space. Moreover, due to better handling of constraints within NSGA-II, the proposed approach has been able to find a better design compared to an existing classical optimization algorithm.

Computation of plausibility values are computationally demanding and the use of parallel computing is important for such problem-solving tasks. The use of optimization algorithms that allow such parallel computations, such as EAs, remains as key methodologies for solving such problems. Here, we have employed an emerging GPGPU multi-threaded computing platform to demonstrate a computational time advantage of 150 to 700 times than that needed with a serial computer.

Handling uncertainties in design variables and parameters is a must if optimization methods have to be used routinely in engineering design activities. The use of evolutionary multi-objective optimization algorithm and a parallel computing platform exploits the currently available software-hardware technologies to show promise in solving optimization problems with uncertainties in variables and parameters. However, the uncertainties in some parameters may be known by means of a probability distribution and uncertainties in some other parameters may only be available sparingly. A methodology for handling combined such cases (combining reliability-based and evidence-based design) would be the next step. Hybridizing the concepts with meta-modeling based optimization techniques should reduce the computational burden further and must be explored next.

Acknowledgments

The study is supported by Department of Science and Technology, Government of India under SERC-Engineering Sciences scheme (No. SR/S3/MERC/091/2009).

References

[1] Cruse, T. R., 1997. *Reliability-based mechanical design*. New York: Marcel Dekker.

[2] Deb, K., Gupta, S., Daum, D., Branke, J., Mall, A., and Padmanabhan, D., 2009. “Reliability-based optimization using evolutionary algorithms”. *IEEE Trans. on Evolutionary Computation,* **13**(5), pp. 1054–1074.

[3] Du, X., and Chen, W., 2004. “Sequential optimization and reliability assessment method for efficient probabilistic design”. *ASME Transactions on Journal of Mechanical Design,* **126**(2), pp. 225–233.

[4] Ditlevsen, O., and Bjerager, P., 1989. “Plastic reliability analysis by directional simulation”. *Journal of Engineering Mechanics, ASCE,* **115**(6), pp. 1347–1362.

[5] Gu, L., and Yang, R., 2006. “On reliability-based optimisation methods for automotive structures”. *Int. J. Materials and Product Technology,* **Vol. 25, Nos. 1/2/3**, pp. 3–26.

[6] Salazar, D. E., and Rocco, C. M., 2007. “Solving advanced multi-objective robust designs by means of multiple objective evolutionary algorithms (moea): A reliability application”. *Reliability Engineering & System Safety,* **92**(6), pp. 697–706.

[7] Daum, D., Deb, K., and Branke, J., 2007. “Reliability-based optimization for multiple constraint with evolutionary algorithms”. In Congress on Evolutionary Computation, IEEE, pp. 911–918.

[8] Gunawan, S., and Papalambros, P. Y., 2006. “A bayesian approach to reliability-based optimization with incomplete information”. *ASME Journal of Mechanical Design,* **128**, pp. 909–918.

[9] Mourelatos, Z. P., and Zhou, J., 2006. “A design optimization method using evidence theory”. *ASME Journal of Mechanical Design,* **128**, pp. 901–908.

[10] Rao, S. S., 2008. “Evidence-based fuzzy approach for the safety analysis of uncertain systems”. *AIAA Journal,* **46**, pp. 2383–2387.

[11] Srivastava, R., and Deb, K., 2010. “Bayesian reliability analysis under incomplete information using evolutionary algorithms”. In Simulated Evolution and Learning, Berlin, Heidelberg: Springer, pp. 435–444. Lecture Notes in Computer Science, 6457.

[12] Dellnitz, M., Schütze, O., and Sertl, S., 2002. “Finding zeros by multilevel subdivision techniques”. *IMA Journal of Numerical Analysis,* **22**(2), pp. 167–185.

[13] Ye, K. Q., Li, W., and Sudjianto, A., 2000. “Algorithmic construction of optimal symmetric latin hypercube designs”. *Journal of Statistical Planning and Inferences,* **90**, pp. 145–159.

[14] Byrd, R. H., Nocedal, J., and Waltz, R. A., 2006. *KNITRO: An integrated package for nonlinear optimization*. Springer-Verlag, pp. 35–59.

[15] Zou, T., Mourelatos, Z. P., Mahadevan, S., and Tu, J., 2008. “An indicator response surface method for simulation-based reliability analysis”. *ASME Journal of Mechanical Design,* **130**(7), pp. 071401–1–11.

[16] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T., 2002. “A fast and elitist multi-objective genetic algorithm: NSGA-II”. *IEEE Transactions on Evolutionary Computation,* **6**(2), pp. 182–197.

[17] Keeney, R. L., and Raiffa, H., 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York: Wiley.

[18] Deb, K., and Gupta, S., in press. “Understanding knee points in bicriteria problems and their implications as preferred solution principles”. *Engineering Optimization*.

[19] Wu, Y. T., Shin, Y., Sues, R., and Cesare, M., 2001. “Safety-factor based approach for probabilistic-based design optimization”. In 42nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference. AIAA-2001-1522.

[20] Liang, J., Mourelatos, Z. P., and Tu, J., 2004. “A single-loop method for reliability-based design optimization”. *ASME Conference Proceedings,* **2004**(46946), pp. 419–430.

[21] Cant-Paz, E., 1998. “A survey of parallel genetic algorithms”. *Calculateurs Paralleles Reseaux Et Systems Repartis,* **10**.

[22] NVIDIA, 2008. *Nvidia Compute Unified Device Architecture C Programming Guide 3.2*. http://developer.nvidia.com/cuda-toolkit-32-downloads.

[23] Zitzler, E., and Thiele, L., 1999. “Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach”. *IEEE Transactions on Evolutionary Computation,* **3**(4), pp. 257–271.