Solving constrained optimization problems via Subset Simulation

Hong-Shuang Li and Siu-Kui Au
Department of Building and Construction, City University of Hong Kong
83 Tat Chee Avenue, Kowloon, Hong Kong
e-mail: siukuiau@cityu.edu.hk

Abstract: This paper extends the application of Subset Simulation (SS), an advanced Monte Carlo algorithm for reliability analysis, to solve constrained optimization problems encountered in engineering. The proposed algorithm is based on the idea that an extreme event (optimization problem) can be considered as a rare event (reliability problem). The Subset Simulation algorithm for optimization is a population-based stochastic global optimization approach realized with Markov Chain Monte Carlo and a simple evolutionary strategy, and so it does not require initial guess or gradient information. The constraints are handled by a priority-based fitness function according to their degree of violation. Based on this constraint fitness function, a double-criterion sorting algorithm is used to guarantee that the feasible solutions are given higher priority over the infeasible ones. Four well studied constrained engineering design problems in the literature are studied to investigate the efficiency and robustness of the proposed method. Comparison is made with other well-known stochastic optimization algorithms, such as genetic algorithm, particle swarm optimization and evolutionary strategy.

Keywords: Subset Simulation, constrained optimization, feasibility-based rule, Markov Chain Monte Carlo

1. Introduction

Many constrained optimization problems arise from modern engineering design process, with systems often modeled with nonlinear, multimodal or even discontinuous behavior. They are complex and difficult to solve by traditional optimization methods (Ravindran, Ragsdell, & Reklaitis, 2006). Stochastic search algorithms have received increasing attention and have been successfully applied in many research fields in recently years (Spall, 2003). Genetic Algorithm (GA) (Holland, 1975) and Simulated Snnealing (SA) (Kirkpatrick, Gelatt, & Vecchi, 1983) are amongst the most two successful heuristic techniques. Other stochastic optimization algorithms include Ant Colony Optimization (ACO) (Maniezzo, Dorigo, & Colorni, 1996), Particle Swarm Optimization (PSO) (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995) etc. It is commonly believed that there is no single universal method capable of solving all kinds of global optimization problems efficiently because each method has its own definition and heuristics to improve efficiency.

Subset Simulation (SS) (Au & Beck, 2001; Au & Beck, 2003; Au, Ching, & Beck, 2007) is an efficient Monte Carlo technique originally developed for structural reliability problems. The method takes advantage of Markov Chain Monte Carlo (MCMC) (Robert & Casella, 2004) and a simple evolutionary strategy (Schuëller, 2009). Based on the idea that an extreme event (optimization problem) can be considered as a rare event (reliability problem), the aim of this paper is to explore application of Subset Simulation to solve constrained optimization problems. Motivated by Dong et al. (2005), a modified feasibility based rule is

4th International Workshop on Reliable Engineering Computing (REC 2010)
Edited by Michael Beer, Rafi L. Muhanna and Robert L. Mullen

ISBN: 978-981-08-5118-7. *Published by* Research Publishing Services.
doi:10.3850/978-981-08-5118-7_069

presented to handle general constraints, which segregates feasible and infeasible solutions and optimizes of the objective function simultaneously in a natural way. Simulation results of four well studied constrained engineering design problems illustrate the robustness and efficiency of the proposed method.

2. Connection between optimization problem and reliability problem

Consider the constrained optimization problem formulated as

$$\max\ h(\mathbf{x}) \quad \text{s.t.}\ g_i(\mathbf{x}) \le 0, i = 1,\ldots,L,\ \ \mathbf{x} \in S \tag{1}$$

where $h: R^n \to R$ is the objective function; $g_i: R^n \to R\left(i = 1,\ldots,L\right)$ is the ith inequality constraint, L is the number of inequality constraints; and $S \subseteq R^n$ is the definition domain or the search space. To view an optimization problem in the context of a reliability problem, consider the input variables $\mathbf{x}$ as random variables so that the objective function $h(\mathbf{x})$ is now random and has its own probability density function (PDF), and cumulative distribution function (CDF). This concept of 'augmenting' deterministic variables as random ones is merely a computational technique for solving complex problems taking advantage of Monte Carlo Simulation (Au 2005). From probability theory, every CDF is monotone, non-decreasing and right-continuous. Let h_{opt} be the global maximum of h, where $\mathbf{x} = \mathbf{x}_{opt}$. By definition the CDF value at $h = h_{opt}$ is unity. Consider the reliability problem of finding the "failure probability" P_F defined as

$$P_F = P(F) = P(h(\mathbf{x}) \ge h_{opt}) \tag{2}$$

where the failure event is $F = \left\{h(\mathbf{x}) \ge h_{opt}\right\}$. Clearly, this is zero because h_{opt} is the global maximum. In the context of an optimization problem, the failure probability is of little concern, however; the attention is on the point or region where the objective function attains the largest value(s). The rare failure region in the reliability problem corresponds to the region where the objective function attains its global maximum. This suggests the possibility of converting a global optimization problem into a rare event simulation problem, where the feasible potentially optimal solutions are analogously the rare event samples close to failure in the reliability problem. The rare event simulation problem is in turn handled by Subset Simulation.

3. The proposed methodology

Along the same spirit of Subset Simulation for reliability analysis (Au & Beck, 2001; Au & Beck, 2003; Au et al., 2007), $P\left(h(\mathbf{x}) \ge h_{opt}\right)$ can be expressed as a product of a sequence of conditional probabilities. During Subset Simulation a series of intermediate threshold values $\left\{h_i : i = 1,2,\cdots\right\}$ are generated that correspond to boundaries consistent with the specified conditional probabilities $\left\{p_k : k = 1,2\ldots\right\}$

$$p_1 = P(h(\mathbf{x}) \ge h_1) = P\left(F_1\right) \tag{3}$$

$$p_i = P(h(\mathbf{x}) \geq h_i \,|\, h(\mathbf{x}) \geq h_{i-1}) = P(F_i \,|\, F_{i-1}), i > 1 \tag{4}$$

where F_i is the intermediate event determined by the objective function value h_i. By the definition of conditional probability, we have

$$P_F = \prod_{i=1} p_i \tag{5}$$

We can generate samples (solutions) using the modified Metropolis-Hasting algorithm (Au & Beck, 2001) that progressively towards the maximum, while the rare event region is gradually explored. For an optimization problem with at least one global point, one can expect that $h_i \to h_{opt}$ as $P_F \to 0$.

3.1. CONSTRAINT HANDLING

One central problem for applying stochastic optimization techniques to the constrained optimization problem is how to handle different types of constraints. A lot of research has been devoted to handling constraints in Genetic Algorithm (GA) as an important step in the design process (Coello Coello, 2002a; Michalewicz, 1996). Simulated Annealing (SA), on the other hand, has not yet gained similar popularity because it requires special strategies to maintain diversity (Hedar & Fukushima, 2006). That is, when the feasible domain consists of several discontinuous sub-feasible domains, SA may encounter difficulty in exploring the whole feasible domain. Similar to SA, Subset Simulation is also built on Markov Chain Monte Carlo to generate candidate samples (solutions). However, SA is a typically point-to-point algorithm, while Subset Simulation operates a population of samples. To a certain extent, Subset Simulation may be regard as a population-based SA with a simple evolutionary strategy for reliability problem. In this work, we borrow constraint handling techniques in population-based stochastic optimization algorithms to develop appropriate strategies for Subset Simulation.

The existing constraint handling techniques in the literature can be roughly classified into four groups: (1) rejection method; (2) penalty function method; (3) multi-objective function method; (4) specified strategy. In the rejection method, only the feasible solutions are kept in the search process and infeasible solutions are discarded. It is difficult to approach the feasible region if the feasible region in the search space is comparatively small. By construction, the rejection method cannot explore the infeasible region. In contrast, the penalty function method exploits also the infeasible solutions in the searching process by transforming a constrained problem into an unconstrained one through the incorporation of a penalty function into the objective function. The main disadvantage of this technique is the difficulty in the selection of the penalty function because it is problem- and scale-dependent (Coello Coello, 2002b). Multi-objective optimization concept has recently been adapted to handle constraints in GA (Mezura-Montes & Coello Coello, 2006), which converts a single objective optimization problem into a multi-objective one and then applies multi-objective techniques to solve it. This strategy by-passes the selection of penalty function. There is, however, one major difference in the optimization aim between multi-objective optimization and handling constraints using multi-objective optimization concept. The former aims at finding a Pareto optimal set in the searching space, while the later requires the associated multi-objective optimization problem to degenerate into a single-objective one in the feasible region again because the optimal value is gained at one point instead of a point set. In the forth group, the specified strategies for different heuristic algorithms have been developed, such as ‘gene repair’ in GA and ‘fly-back’ in PSO. However, these methods lack generality.

We adopt the constraint fitness priority based ranking method first proposed by Dong et al (2005) to solve constraints in PSO. The method automatically takes care of the constraints during samples generation and so it by-passes the need to modify the objective function. In the context of the proposed algorithm, a new constraint fitness function for handling constraints is proposed in this work.

A 'constraint violation function' is first defined to handle constraints and ranking samples. For an inequality constraint $g_i(\mathbf{x}) \le 0$, it is defined as

$$\nu_i(\mathbf{x}) = \begin{cases} 0 & \text{if } g_i(\mathbf{x}) \le 0 \\ g_i(\mathbf{x}) & \text{if } g_i(\mathbf{x}) > 0 \end{cases} \tag{6}$$

The overall constraint fitness function that accounts for all constraints is defined as:

$$F_{con}(\mathbf{x}) = -\max_i \ \nu_i \tag{7}$$

It is clear that $F_{con}(\mathbf{x}) = 0$ if and only if the trial solution $\mathbf{x}$ belongs to the feasible domain. The expression in Eq.(7) reveals that all solutions in the feasible domain have the same value of $F_{con}(\mathbf{x})$ (equal to zero) while the worst violated constraint dominates the violation level for an infeasible solution.

It should be noted that the constraint fitness function used here is different from that used in Dong et al (2005). The real violation is used to reflect the magnitude of violation in this work, while a relative measure was adopted in Dong et al (2005). Another difference is that the most serious violation among all constraints is used to construct constraint fitness function instead of a sum of relative ratio with random weighted factors.

In the proposed algorithm for optimization problems, samples with favorable values are selected from the population at the current level to provide "seeds" for the next simulation level. The selection process needs to consider the constraint fitness function and objective function simultaneously. A double-criterion ranking method is used for this purpose (Dong etal, 2005). Specifically, the samples are first sorted according to their constraint fitness function values

$$F_{con}(\mathbf{x}_1) \le F_{con}(\mathbf{x}_2) \le \cdots \le F_{con}(\mathbf{x}_N) \tag{8}$$

Those samples that satisfy the constraints will appear at the top of the list with the same value of $F_{con}(\mathbf{x})$ equal to zero. These samples are then sorted again according to their objective function value. The first ranking based on the constraint fitness function is designed to search the feasible domain, while the second ranking based on objective function looks for the optimal solution. In this manner the searching processes for the feasible region and the optimal solution proceed together. This method preserves the advantage that the constraint fitness function value of a feasible solution is always better than that of an infeasible one and it by-passes the need to trade off between the objective and penalty function (Dong et al, 2005), which can be nontrivial due to different scaling.

3.2. PROCEDURE

The proposed algorithm is described as follows:

Step 0

Select the distribution parameters of input variables. In the original problem, each input variable is

deterministic parameter but it is augmented as a random variable in the presented algorithm with an artificial probability density functions (PDF) $f(x)$.

Step 1

Generate N independent and identically distributed samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ by direct Monte Carlo Simulation according to the artificial distributions. Each sample $\mathbf{x}_i (i = 1, \ldots, N)$ has n components, i.e. $\mathbf{x}_i = \left\{x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right\}^T$ where $x_i^{(j)} (j = 1, \ldots, n)$ are generated from $f_j(x_j)(j = 1, \ldots, n)$. Calculate the constraint fitness function values and the objective function values of the samples, and then sort them according to the double-criterion ranking algorithm described in Section 3.1. Refer $\mathbf{x}_N$ as the best solution and $\mathbf{x}_1$ as the worst. From the sequence, obtain the $N(1-p_1)$ th sample $\mathbf{x}_{N(1-p_1)}$ with corresponding $h_{1,N(1-p_1)}$ and $F_{con}\left(\mathbf{x}_{N(1-p_1)}\right)$, assuming that p_1 and N are chosen such that $N(1-p_1)$ is an integer. The subscript '1' here denotes simulation level 1. Note that the sample estimate of $P\left\{F_1 : h \geq h_{1,N(1-p_1)} \cap F_{con} \geq F_{con}\left(\mathbf{x}_{N(1-p_1)}\right)\right\}$ is by definition p_1. In this manner, the event F_1 is adaptively chosen. Due to the choice of $\mathbf{x}_{N(1-p_1)}$, there are Np_1 samples whose objective function values are large than $h_{1,N(1-p_1)}$ and constraint fitness function values are large than $F_{con}\left(\mathbf{x}_{N(1-p_1)}\right)$ among all samples, and the corresponding samples belong to the event F_1. These samples provide the "seeds" samples for next simulation level.

Step 2

The modified Metropolis-Hasting algorithm (Au & Beck, 2001; Au & Beck, 2003) is employed for generating conditional samples. In the *k*-th simulation level, starting from each of samples in F_{k-1}, a Markov chain can be generated with the same conditioning. Since the initial samples obey the conditional distribution, all these Markov chains are automatically in stationary state and samples in these Markov chains distribute according to conditional distribution. Note that the length of each Markov Chain is $1/p_{k-1}(k = 2, \ldots)$ where p_{k-1} is the level probability in last simulation level. Evaluate and sort the samples again, determine the $N(1-p_k)$ th sample $\mathbf{x}_{N(1-p_k)}$ from the sequence. Again, select the samples whose objective function values are larger than $h_{k,N(1-p_k)}$ and constraint fitness function values are larger than $F_{con}\left(\mathbf{x}_{N(1-p_k)}\right)$ to provide "seeds" for sampling in the next simulation level.

The procedure is repeated for higher simulation levels until a convergence criterion is met. Note that the total number of samples is equal to $N + \sum_{k=2}^{m}(1-p_k)N$ where *m* is the total simulation levels for a problem.

4. Computational issues

4.1. ARTFICIAL PDFS FOR INPUT VARIABLES

The choice of the distribution for the input variables directly affects the domain where feasible solutions are searched. A truncated Gaussian distribution may be used to handle simple bound constraints on individual design variable

$$f\left(x;\mu,\sigma,x_u,x_l\right)=\frac{\phi\left(\frac{x-\mu}{\sigma}\right)}{\Phi\left(\frac{x_u-\mu}{\sigma}\right)-\Phi\left(\frac{x_l-\mu}{\sigma}\right)} \tag{9}$$

where $\phi(\cdot)$ is the probability density function of the standard Gaussian distribution, $\Phi(\cdot)$ is the cumulative distribution function of the standard Gaussian distribution and the definition domain $\Omega=\left\{x: x_l \le x \le x_u\right\}$. The mean μ should be chosen close to the global optimum. If no prior information on the problem is available, one may locate it at the center of the definition domain.

The standard deviation of the artificial distribution controls the range to be explored and it has an influence on the efficiency. If it is too small, most of samples will cluster in a small region and then the sequence of objective function will increase slowly. If it is too large, the samples will scatter over a large region and it would require a longer process to converge to the global optimum. One strategy is to use the three sigma limits in reliability engineering, setting the distance from sampling center to the upper or lower bound equal to three standard deviations, i.e.

$$\sigma_i=\frac{L_i}{6} \tag{10}$$

where L_i is the interval length of definition domain of the ith input variable x_i, and σ_i is the corresponding artificially standard deviation.

4.2. CONVERGENCE CRITERION

In this paper, we use the standard deviation of samples in each simulation level to check the convergence of the searching process. In order to eliminate the effects of different scaling of design variables, the interval length of definition domain is used as a reference. The convergence criterion is

$$\left|\frac{\hat{\sigma}_k}{x_u-x_l}\right| \le \varepsilon \tag{11}$$

where $\hat{\sigma}_k$ is the estimator of standard deviation of samples in k-th simulation level, and ε is a specified tolerance. The idea of this criterion stems from the fact that when the searching procedure approaches the global optimum, more repeated samples would be found in samples and then the estimator of standard deviation of samples will tend to zero. Here, we adopt ε =10^{-5}~10^{-7}.

When the convergence criterion is satisfied, the largest value in the objective function sequence is taken as the optimal value, and the corresponding sample as the optimal solution.

4.3. LEVEL PROBABILITY

The level probability p_k is a parameter that regulates the convergence of the optimization process. If a small value is used, the algorithm would have a low probability of reaching a global optimum. The level probabilities must be high enough to permit the locally developed Markov Chain samples to move out of a local optimum in favor of finding a global optimal, especially in early simulation levels. However, high level probabilities would increase the number of simulation levels. Here, we adopt a decreasing strategy to handle this difficulty. First, we set $p_1 = 0.5$ at initial simulation level and then reduce to 0.2 in the (i+1)th simulation level when the largest estimator of all design variables $\hat{\sigma}_i$ is less than 0.1, and further reduce to 0.1 when $\hat{\sigma}_j < 0.01$ in order to accumulate the convergent speed.

5. Application examples

The proposed algorithm is applied to four well-studied engineering design problems with different kinds of constraints. In each example, 30 independent runs are carried out to investigate the performance of the algorithm in a statistical sense.

5.1. WELDED BEAM DESIGN

The first problem considers the optimal design of a welded beam, which is taken from Coello Coello (2000). The objective is to minimize the total cost function $h(\mathbf{x})$ including setup cost, welding labor cost and material cost with constraints on shear stress $\tau(\mathbf{x})$, bending stress in the beam $\sigma(\mathbf{x})$, buckling load on the bar $P_c(\mathbf{x})$, end deflection of the beam $\delta(\mathbf{x})$, side constraints $\{g_3(\mathbf{x}), g_4(\mathbf{x}), g_5(\mathbf{x})\}$, and limits on design variables. There are four design variables as shown in Figure 1: $[x_1, x_2, x_3, x_4] = [h, l, t, b]$.

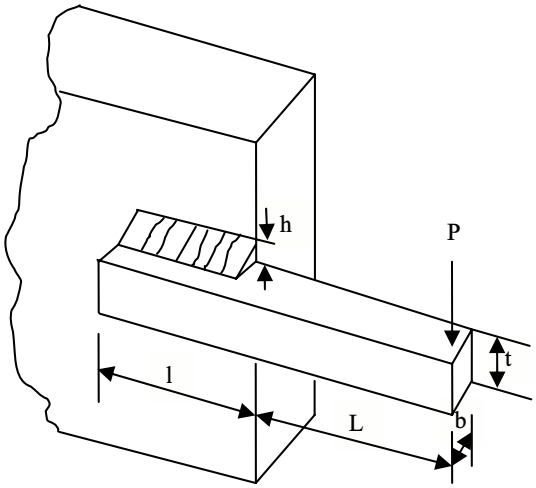

Figure 1 The welded beam design problem

The problem can be formulated as

$$\min \quad h(\mathbf{x}) = 1.1047 x_1^2 x_2 + 0.0481 x_3 x_4 (14.0 + x_2) \tag{12}$$

Subject to

$$g_1(\mathbf{x}) = \tau(\mathbf{x}) - \tau_{\max} \le 0 \tag{13}$$

$$g_2(\mathbf{x}) = \sigma(\mathbf{x}) - \sigma_{\max} \le 0 \tag{14}$$

$$g_3(\mathbf{x}) = x_1 - x_4 \le 0 \tag{15}$$

$$g_4(\mathbf{x}) = 0.1047 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \le 0 \tag{16}$$

$$g_5(\mathbf{x}) = 0.125 - x_1 \le 0 \tag{17}$$

$$g_6(\mathbf{x}) = \delta(\mathbf{x}) - \delta_{\max} \le 0 \tag{18}$$

$$g_7(\mathbf{x}) = P - P_c(\mathbf{x}) \le 0 \tag{19}$$

$$0.1 \le x_1 \le 2,\ 0.1 \le x_2 \le 10,\ 0.1 \le x_3 \le 10,\ 0.1 \le x_4 \le 2 \tag{20}$$

where

$$\tau(\mathbf{x}) = \sqrt{(\tau')^2 + 2\tau'\tau'' + (\tau'')^2} \tag{21}$$

$$\tau' = \frac{P}{\sqrt{2} x_1 x_2}, \quad \tau'' = \frac{MR}{J}, \quad M = P\left(L + \frac{x_2}{2}\right) \tag{22}$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2} \tag{23}$$

$$J = 2\left\{\sqrt{2} x_1 x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \tag{24}$$

$$\sigma(\mathbf{x}) = \frac{6PL}{x_4 x_3^2}, \quad \delta(\mathbf{x}) = \frac{4PL^3}{E x_3^3 x_4} \tag{25}$$

$$P_c(\mathbf{x}) = \frac{4.013E\sqrt{x_3^2 x_4^6/36}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right) \tag{26}$$

$$P = 6000\text{lb}, \quad L = 14\text{in}, \quad E = 30\times10^6\,\text{psi}, \quad G = 12\times10^6\,\text{psi} \tag{27}$$

$$\tau_{\max} = 13600\text{psi}, \quad \sigma_{\max} = 30000\text{psi}, \quad \delta_{\max} = 0.25\text{in} \tag{28}$$

This problem has been attempted previously by several stochastic optimization algorithms: GA based on a co-evolution model (GA1) (Coello Coello, 2000), GA through the use of dominance-based tournament selection (GA2) (Coello Coello & Montes, 2002), evolutionary programming with a cultural algorithm (EP) (Coello Coello & Becerra, 2004), co-evolutionary particle swarm optimization (CPSO) (He & Wang, 2007a), hybrid particle swarm optimization (HPSO) with a feasibility-based rule (He & Wang, 2007b) and hybrid Nelder–Mead simplex search method and particle swarm optimization (NM–PSO) (Zahara & Kao, 2009). Their best solutions are compared with that obtained by the proposed algorithm, and are listed in Table I. It should be noted that the result produced by NM-PSO (Zahara & Kao, 2009) is an infeasible solution because the third constraint $g_3(\mathbf{x})$ had been slightly violated. From Table I, the best feasible solution obtained by Subset Simulation is competitive to the results produced by EP (Coello Coello & Becerra, 2004) and HPSO (He & Wang, 2007b) and is better than the results obtained by other optimization techniques. Table II summaries the average optimum-locating performance and computational effort spent by different methods over 30 independent runs. It can be seen that the mean of the objective function yielded by Subset Simulation is better that of other algorithms although the worst case is some what average among others.

Based on 30 independent runs, the average iteration number was 122 and the average number of function evaluations was 83,703. Comparing with the function evaluations required by other methods the proposed algorithm can be an efficient choice for this problem.

Table I Comparison of the best solution for welded beam design problem by different methods

	Best solution						
	GA1	GA2	EP	CPSO	HPSO	NM-PSO	SS
x_1	0.208800	0.205986	0.205700	0.202369	0.205730	0.205830	0.20572964
x_2	3.420500	3.471328	3.470500	3.544214	3.470489	3.468338	3.470488668
x_3	8.997500	9.020224	9.036600	9.048210	9.036624	9.036624	9.03662391
x_4	0.210000	0.206480	0.205700	0.205723	0.205730	0.205730	0.20572964
x_1	-0.337812	-0.10305	1.9886769	-13.65555	-0.0254	-0.025251	-0.000021
g_2	-353.9026	-0.231748	4.4815489	-75.81408	-0.053122	-0.053122	-0.000029
g_3	-0.0012	-0.000494	0.000000	-0.003354	0.000000	0.000100	0.000000
g_4	-3.411865	-3.430044	-3.433213	-3.424572	-3.432981	-3.433169	-3.432984
g_5	-0.083800	-0.080986	-0.080700	-0.077369	-0.08073	-0.08083	-0.080730
g_6	-0.235649	-0.235514	-0.235538	-0.235595	-0.235540	-0.235540	-0.235540
g_7	-363.2324	-58.64689	2.6033472	-4.472858	-0.031556	-0.031556	-0.000019
h	1.748309	1.728226	1.724852	1.728024	1.724852	1.724717	1.724852

Table II Statistical performance of different methods for welded beam design problem					
Methods	Best	Mean	Worst	Std.	Function evaluations
GA1	1.748309	1.771973	1.785835	0.011220	900,000
GA2	1.728226	1.792654	1.993408	0.074713	80,000
EP	1.724852	1.971809	3.179709	0.443131	N/A
CPSO	1.728024	1.748831	1.782143	0.012926	200,000
HPSO	1.724852	1.749040	1.814295	0.040049	81,000
SS	1.724852	1.747429	1.928811	0.046668	83,703

5.2. TENSION-COMPRESSION STRING DESIGN

Consider the tension-compression string design problem taken again from Coello Coello (2000), as shown in Figure 2. The objective is to minimize the string weight under constraints on deflection, shear stress, surge frequency, limits on outside diameter and on design variables. There are three design variables in this problem: the wire diameter d, the mean coil diameter D and the number of active coils P, i.e. $\left[x_1, x_2, x_3\right] = \left[d, D, P\right]$.

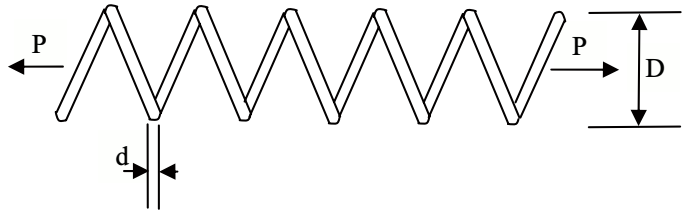

Figure 2 The tension-compression string design problem

The problem can be formulated as

$$\min \; h(\mathbf{x}) = (x_3 + 2)\, x_2 x_1^2 \tag{29}$$

Subject to

$$g_1(\mathbf{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0 \tag{30}$$

$$g_2(\mathbf{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} - 1 \le 0 \tag{31}$$

$$g_3(\mathbf{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0 \tag{32}$$

$$g_4(\mathbf{x}) = \frac{x_2 + x_1}{1.5} - 1 \le 0 \tag{33}$$

$$0.05 \le x_1 \le 2.0,\; 0.25 \le x_2 \le 1.3,\; 2.0 \le x_3 \le 15.0 \tag{34}$$

This problem has been attempted previously by GA1, GA2, EP, CPSO, HPSO, and NM–PSO. A comparison of best solutions obtained by these methods and Subset Simulation is shown in Table III. The result produced by NM-PSO is infeasible because the first and second constraints are violated by a relative

magnitude of 10^{-3}. As shown in Table III, HPSO gives the most favorable solution, although Subset Simulation is still competitive compared to other methods. The statistical results of 30 independent runs are listed in Table IV. It can be seen that Subset Simulation also produced competitive results to other compared methods.

Table III Comparison of tension-compression string design problem by different methods

Methods	x_1	x_2	x_3	h
GA1	0.051480	0.351661	11.632201	0.0127048
GA2	0.051989	0.363965	10.890522	0.0126810
EP	0.050000	0.317395	14.031795	0.0127210
CPSO	0.051728	0.357644	11.244543	0.0126747
HPSO	0.051706	0.357126	11.265083	0.0126652
NM-PSO	0.051620	0.355498	11.333272	0.0126302
SS	0.051672	0.356304	11.313358	0.0126654

On the other hand, the mean of the iteration number for Subset Simulation was 69 and the average number of function evaluations was only 40,207. Considering both qualities of solution and computational effort spent, the proposed method is quite favorably.

Table IV Statistical performance of different methods for tension-compression string design problem

Methods	Best	Mean	Worst	Std.	Function evaluations
GA1	0.0127048	0.0127690	0.012822	3.9390e-005	900,000
GA2	0.0126810	0.0127420	0.012973	5.9000e-005	80,000
EP	0.0127210	0.0135681	0.015116	8.4152e-004	N/A
CPSO	0.0126747	0.0127300	0.012924	5.1985e-004	200,000
HPSO	0.0126652	0.0127072	0.012719	1.5824e-005	81,000
SS	0.0126654	0.0127510	0.012981	7.6753e-005	40,207

5.3. AN OPTIMIZATION PROBLEM WITH DISJOINT FEASIBLE REGION

The search space of this problem has 9^3 disjoint spheres, and it can be stated as follows (Coello Coello & Montes, 2002):

$$\max\ h(\mathbf{x}) = 1 - 0.01\left[(x_1 - 5)^2 + (x_2 - 5)^2 + (x_3 - 5)^2\right] \tag{35}$$

Subject to

$$g(\mathbf{x}) = (x_1 - i)^2 + (x_2 - j)^2 + (x_3 - k)^2 \tag{36}$$

where $0 \le x_1, x_2, x_3 \le 10$ and $i, j, k = 1, 2, \ldots, 9$.

The optimal solution of this problem is $f(\mathbf{x}) = 1.000000$ with $\mathbf{x} = \{5.000000, 5.000000, 5.000000\}$. Table V shows the statistical results obtained by EP, GA2, HPSO, NM-PSO, Subset Simulation and the homomorphous mapping (HM) (Koziel & Michalewicz, 1999), the stochastic ranking (SR) (Runarsson & Yao, 2000). GA2, HPSO, NM-PSO, SR and Subset Simulation are very robust in reaching the optimal solution for this problem, showing low variability of results. With respect to efficiency, NM-SPO is the clear winner as it needs only 923 function evaluations. The function evaluations required by other methods

are also summarized in Table V. For this problem, the average iteration number for Subset Simulation was 34 and the average number of function evaluations was 12,093. This is better than other methods except NM-PSO. Note, however, that NM-PSO requires the explicit gradient information of constraints to repair infeasible solutions, which may not be available in realistic applications.

Table V Statistical performance of different methods for optimization problem with disjoint feasible region

Methods	Best	Mean	Worst	Std.	Function evaluations
HM	0.999999	0.999135	0.991950	N/A	140,000
SR	1.000000	1.000000	1.000000	0.0e+000	350,000
GA2	1.000000	1.000000	1.000000	0.0e-006	80,000
EP	1.000000	0.996375	0.996375	9.7e-003	N/A
HPSO	1.000000	1.000000	1.000000	1.6e-015	81,000
NM-PSO	1.000000	1.000000	1.000000	0.0e+000	923
SS	1.000000	1.000000	1.000000	7.7e-016	12,093

5.4. TEN-BAR PLANE TRUSS DESIGN

The ten-bar plane truss shown in Figure 3 is taken from Coello Coello & Montes (2002) as an example of structural optimization with implicit constraints. The modulus of elasticity E is 1.0×104ksi (68965.5MPa) while the mass density ρ is 0.101lb/in^3 (2768.096kg/m^3). The structure is designed for a single loading condition: 100kips (45351.47kg) applying in the negative y-direction at nodes 2 and 4.

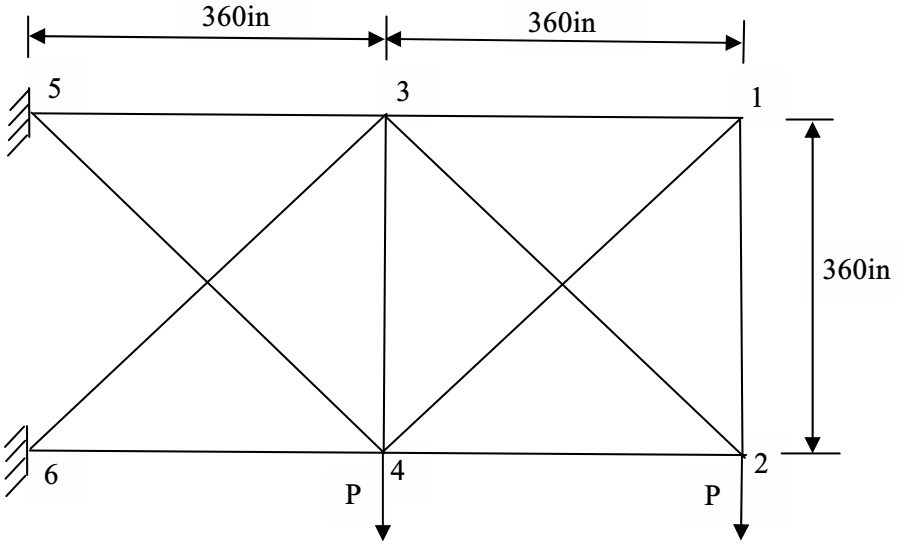

Figure 3 The ten-bar plane truss design problem

The objective is to minimize the weight of the truss considering stress and displacement related constraints. The objective function is given by

$$h(\mathbf{x}) = \sum_{j=1}^{10} \rho A_j L_j \tag{37}$$

where $\mathbf{x}$ is the candidate solution, A_j is the cross-sectional area of the j-th bar ($A_j = \sqrt{I_j}$, where I_j is the second moment of area of bar *j*), L_j is the length of j-th bar. The second moment of area $I_j, j = 1,2,\ldots,10$ are taken as design variables with a range of $0.1 \le I_j \le 999.0\,\text{in}^4$. There are 10 stress-related constraints and 8 nodal displacement-related constraints. The allowable stress of each bar is ± 25ksi (172.41MPa). The displacement of free nodes 1-4 in both direction x and y must be less than 2in (5.08cm).

This problem has been attempted by GA2 (Coello Coello & Montes, 2002), and its best solutions is compared against those obtained by Subset Simulation in Table VI. From Table VI, it is can be seen that a better optimal solution of $f = 5142.53684$ lb has been found by Subset Simulation. Table VII shows the statistical results based on 30 independent runs. From Table VII, it can be seen that the average searching quality of Subset Simulation is also superior to those of GA2 for this problem. Moreover, GA2 needed 80,000 function evaluations for this problem, while Subset Simulation only needs 58,741 function evaluations.

Table VI Comparison of best solutions obtained by GA2 and Subset Simulation

	Best solution	
	GA2	SS
x_1	985.808351	937.801131
x_2	0.105877	0.100657
x_3	519.966658	568.391128
x_4	188.576078	221.001915
x_5	0.102124	0.100015
x_6	0.137725	0.105266
x_7	690.171450	66.1667993
x_8	495.366009	465.329860
x_9	467.438050	453.048813
x_{10}	0.135133	0.100284
h	5157.685516	5142.536834

Table VII Statistical performance of different methods for ten-bar structure

Methods	Best	Mean	Worst	Std.	Function evaluations
GA2	5157.685516	5198.085918	5274.693439	29.496938	80,000
SS	5142.536834	5147.932963	5181.67382	7.961660	58,741

6. Conclusions

This paper describes a new application of Subset Simulation for optimizing engineering design problems under general constraints. The proposed algorithm is based on the idea that extreme events (optimization problems) can be considered rare events (reliability problems). A feasibility-based rule is used to guarantee that feasible solutions are always better than infeasible solutions in term of constraint fitness function values. The rule employs a modified constraint fitness function to evaluate the constraint fitness of a solution and a double-criterion ranking method to select the best solutions according to both constraint

fitness function values and objective function values. Four benchmarks of engineering design are shown to demonstrate the robustness and efficiency of the proposed algorithm compared with existing ones. The proposed algorithm is found to be competitive in exploiting the feasible regions and providing optimal designs in complex problems. Current research focuses on further testing its performance on structural optimization design and improving its efficiency by combining with local search strategies.

Acknowledgements

The work presented in this paper is supported by the Hong Kong Research Grant Council (HKRGC) through General Research Fund (GRF) (Project No. 9041484). The support is gratefully acknowledged.

References

Au, S. K. Reliability-based design sensitivity by efficient simulation. *Computers and Structures,* 83:1048–1061, 2005.

Au, S. K., and J. L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4), 263-277, 2001.

Au, S. K., and J. L. Beck. Subset simulation and its application to seismic risk based on dynamic analysis. *Journal of Engineering Mechanics*, 129(8), 901-917, 2003.

Au, S. K., J. Ching, and J. L. Beck. Application of subset simulation methods to reliability benchmark problems. *Structural Safety*, 29(3), 183-193, 2007.

Coello Coello, C. A. Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, 41(2), 113-127. 2000.

Coello Coello, C. A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12), 1245-1287, 2002a.

Coello Coello, C. A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12), 1245-1287. 2002b.

Coello Coello, C. A., and R. L. Becerra. Efficient evolutionary optimization through the use of a cultural algorithm. *Engineering Optimization*, 36(2), 219-236, 2004.

Coello Coello, C. A., and E. M. Montes. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16(3), 193-203, 2002.

Dong, Y., et al. An application of swarm optimization to nonlinear programming. *Computers and Mathematics with Applications*, 49(11-12), 1655-1668, 2005.

Eberhart, R. C., and J. Kennedy. A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995.

He, Q., and L. Wang. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence*, 20(1), 89-99, 2007a.

He, Q., and L. Wang. A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Applied Mathematics and Computation*, 186(2), 1407-1422, 2007b.

Hedar, A. R, and M. Fukushima. Derivative-free filter simulated annealing method for constrained continuous global optimization. *Journal of Global Optimization*, 35(4), 521-549, 2006.

Holland, J. H. Adaptation in natural and artificial systems. Ann Arbor, Michigan: University of Michigan Press, 1975.

Kennedy, J., and R. C. Eberhart. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 4, 1942-1948, 1995.

Kirkpatrick, S., C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220(4598), 671-680, 1983.

Koziel, S., and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7(1), 19-44, 1999.

Maniezzo, V., M. Dorigo, and N. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, 26(1), 29-41, 1996.

Mezura-Montes, E., and C. A. Coello Coello. Use of multiobjective optimization concepts to handle constraints in genetic Algorithms. In A. Abraham, L. Jain & R. Goldberg (Eds.), *Evolutionary multiobjective optimization*, Springer, 2006.

Michalewicz, Z. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1), 1-32,1996.

Ravindran, A., K. M. Ragsdell, and G. V. Reklaitis. *Engineering optimization: Methods and applications (2nd Edition ed)*. New Jersey: John Wiley & Sons, 2006.

Robert, C. P., and G. Casella. *Monte carlo statistical merhods* (Second Edition ed.). New York: Springer, 2004.

Runarsson, T. P., and X. Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 4(3), 284-294, 2000.

Schuëller, G. I. Efficient Monte Carlo simulation procedures in structural uncertainty and reliability analysis - recent advances. *Structural Engineering and Mechanics*, 32(1), 1-20, 2009.

Spall, J. C. *Introduction to stochastic search and optimization: Estimation, simulation, and control*. New York: Wiley, 2003

Zahara, E., and Y. T. Kao. Hybrid Nelder-mead simplex search and particle swarm optimization for constrained engineering design problems. *Expert Systems with Applications*, 36(2 PART 2), 3880-3886, 2009.