Chance-Constrained Genetic Algorithms

Daniel H. Loughlin and S. Ranji Ranjithan

Department of Civil Engineering
NC State University
Raleigh, NC 27695-7908

Abstract

The applicability of classical mathematical optimization approaches is often limited by the necessity for strict mathematical formulations. As a result, powerful tools, such as chance-constrained optimization, have seen limited use. Genetic algorithms (GAs) are much more flexible than traditional mathematical programming, allowing many problems that were previously deemed intractable to be revisited. In this paper, we explore an approach called a chance-constrained GA (CCGA) that uses a Monte Carlo (MC) simulation in the GA fitness function. Previous applications of CCGA in the literature have required several orders of magnitude of increased computational demands over conventional GAs. To reduce these demands, we test and compare alternative MC sampling approaches and identify an approach that works efficiently with little loss in solution quality. This approach, which uses Latin Hypercube Sampling to generate a new set of only 10 to 50 realizations each generation, is demonstrated for solving an air quality management problem.

1 INTRODUCTION

The majority of applications of optimization in practice have involved deterministic model formulations. In deterministic models, single point-estimates are used for the parameters of the objective function and constraint equations. The actual values for many of these parameters, however, may not be known. Differences between the actual values and the point-estimates may be attributed to variability, uncertainty, or both. Variability is caused by natural stochastic behavior or fluctuations. Uncertainty may originate from measurement error, sampling bias, lack of data, etc.

Consider a problem in which optimization is used to identify the stock portfolio that maximizes total return over an investment time period. The exact return for each potential investment would not be known with certainty. A better way to characterize each potential investment's return would be with a statistical distribution that reflects the range and probability of returns that may be expected. Given these distributions, the optimization model could be modified to identify the portfolio that would maximize return, while at the same time providing a 90% likelihood (or some other confidence level) that some minimum return would be achieved. This kind of formulation is referred to as a chance-constrained optimization model, and fits into the class of stochastic optimization approaches.

While stochastic optimization approaches are powerful in their ability to explicitly consider estimates on variability and uncertainty in model parameters, their application has been limited by the strict mathematical structures required by solution procedures such as linear and nonlinear programming. Many realistic problems that exhibit highly complex behavior cannot be readily modeled within these frameworks.

When compared to traditional optimization solution procedures, GAs offer much more flexibility in problem formulation. Sophisticated analytical and numerical models can even be incorporated into the GA search process. In this paper, we explore the use of GAs for chance-constrained optimization. The approach investigated here, called a chance-constrained GA (CCGA), uses a Monte Carlo simulation in the GA fitness function. If the solution does not meet the chance-constraint at a target frequency it is penalized. Since the approach makes use of as many as several hundred Monte Carlo (MC) simulations in each fitness evaluation, its computational burden can be roughly two orders of magnitude greater than a conventional GA. To address the issue of computational efficiency, we explore a number of alternative MC sampling approaches for a case study in air quality management problem. The results of these runs suggest that the multi-generational search that is characteristic of GAs allows us to use MC sample sizes that would normally be considered insufficient.

This paper is organized as follows: Section 2 provides a cursory background of a classical chanced-constrained optmization solution procedure called chance-constrained programming (CCP); Section 3 describes a framework through which GAs can be used for chance-constrained optimization; the remaining sections describe the case

Deterministic Formulation	Chance-ConstrainedJoint Formulation	Chance-Constrained Formulation
Minimize $c(\underline{x})$ s.t. $A\,\underline{x} \le B$	Minimize $c(\underline{x})$ s.t. $P(\,A\,\underline{x} \le B\,) \ge \alpha$	Minimize $c(\underline{x})$ s.t. $P(\,A_1\,\underline{x} \le B_1\,) \ge a_1$ $P(\,A_2\,\underline{x} \ge B_2\,) \ge a_2$ $P(\,A_1\,\underline{x} + A_2\,\underline{x} \le B_3\,) \ge \alpha$

Where:
$\underline{x}$ is an array of decision variables
A, B, A_1, A_2, B_1, B_2 are coefficients
P(Y) is the probility of Y
α is the probability constraint supplied by the modeler
a_1 and a_2 are decision variables specifying the probability constraints for each constraint equation

Figure 1. Examples of deterministic, chance-constrained, and joint chance-constrained formulations.

study, discuss the results, and identify several important conclusions.

2 CLASSICAL APPROACHES FOR CHANCE-CONSTRAINED OPTIMIZATION

Chance-constrained programming (CCP) techniques (Charnes and Cooper, 1962; Cooper et al., 1997) are the most commonly used stochastic optimization approaches. In a CCP formulation, one or more constraints are treated as probabilistic. CCP techniques convert constraints from this probabilistic form to a deterministic form that can be used in a nonlinear programming model.

A varient of CCP is joint-CCP. In joint-CCP models, a total system reliability target is specified, considering the probabilities on any individual constraints to be decision variables. The optimization model selects the probabilities on each chance constraint so that a system-wide reliability is met. Examples of deterministic, CCP, and joint CCP models are shown in Figure 1. To simplify the discussion presented in this paper, joint-CCP models are not discussed further. However, CCGA could also be applied to problems with joint chance-constraints.

CCP methods for converting probabilistic constraints are relatively easy to apply if the uncertain parameters can be isolated on the right side of the chance constraint and the decision variables isolated on the left side. This formulation is called a right-sided chance constraint. The independence of uncertain parameters from one constraint to another, called row-independence, is another factor that makes solving these problems easier.

CCP can also be used for left-sided chance constraints and for problems without row independence, but correlations between various random variables may make conversion of the chance constraints to deterministic constraints exceedingly cumbersome.

3 CCGA: AN ALTERNATIVE TO CCP

Many realistic phenomena are modeled using functions that are nonlinear, discontinuous, discrete, or that exhibit a combination of these characteristics, making them difficult to model with traditional optimization approaches. Newer, nongradient-based search approaches, such as GA and simulated annealing (SA), provide a compelling alternative because of the flexibility in how the optimization objectives and constraints may be represented. GA and SA can easily be extended to solve stochastic optimization problems. SA was applied to stochastic optimization by Painton and Diwekar (1995). Their approach, called stochastic annealing, tested the performance of potential optimization solutions over many MC realizations of the uncertain parameters. The fitness of a solution was calculated as the mean fitness, or expected value, for the realizations. Painton was also involved in the application of GA to a stochastic optimization problem in which system reliability for a network of computers was evaluated (Painton and Cambell, 1995).

Ritzel et al. (1994) and Cieniawski et al. (1995) applied CCGAs to groundwater problems. In contrast to Painton and Diwekar's expectation-based approach, their GAs used a risk-averse methodology that can be classed as chance-constrained optimization. Potential solutions were evaluated based on both their cost and their ability to meet a constraint for a specified fraction of MC realizations. The authors used 200 MC realizations to adequately represent the potential range of realizations that could occur. All 200 realizations were used in each fitness evaluation of each generation, resulting in an increased

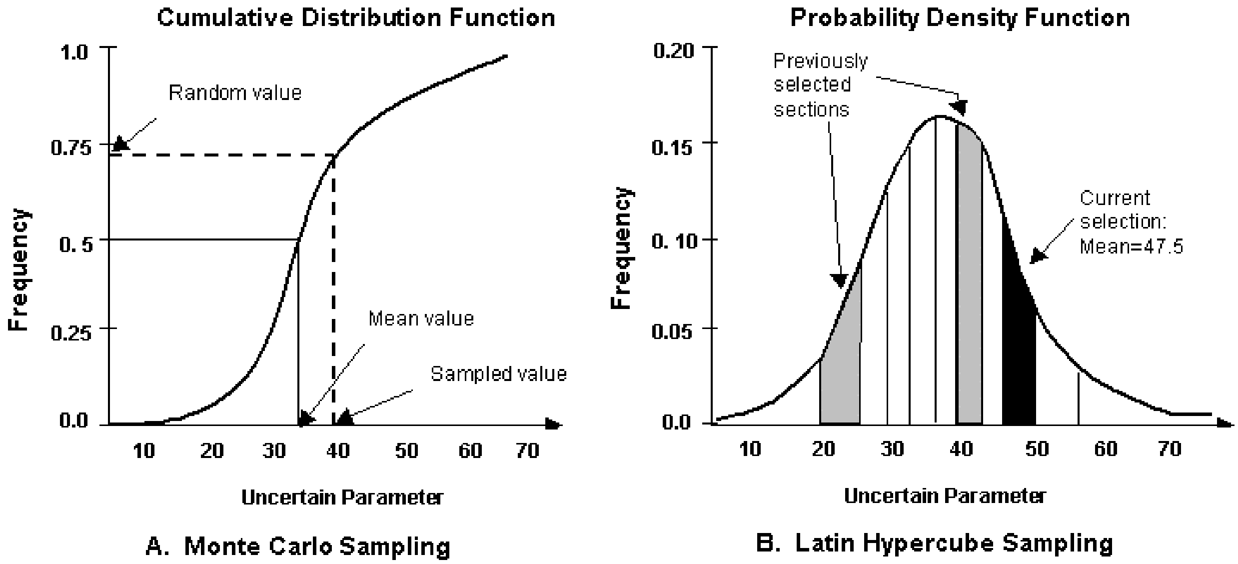

Figure 2. Illustrations of the Monte Carlo sampling and Latin Hypercube Sampling methodologies.

computational burden of a factor of roughly 200 over a deterministic run.

4 CCGA FRAMEWORK

A CCGA is very similar to a conventional GA, but differs in the manner in which solutions are evaluated. In CCGA, each solution is evaluated based on its performance over a number of MC realizations; the reliability of meeting one or more designated constraints is estimated for these realizations; if the reliability is less than that specified in the chance-constraint, the solution is penalized.

To perform CCGA, two parameters must be specified. The first parameter is the number of MC realizations, n, used in each strategy evaluation. The second parameter is the target reliability, defined as the fraction of the MC realizations in which the strategy should meet a constraint.

Consider an optimization problem for which the user has specified these parameters to be 100 and 0.7, respectively. In each fitness evaluation, a solution is tested for 100 realizations of the uncertain and variable parameters. The optimal strategy should meet the constraint for at least 70 of 100 of the MC realizations. If the fraction is less than the target reliability, 0.7, a penalty is exacted, causing the strategy's fitness to be reduced.

There are a number of additional implementation details that must be considered when implementing CCGA. These include:

- the choice of a sampling mechanism with which to obtain the MC realizations: traditional MC sampling or Latin Hypercube Sampling (LHS);
- the choice of when MC realizations are generated: at the onset of the GA run, before each generation, or for every strategy evaluation; and,
- the manner in which the chance-constraint is encoded in the GA fitness function.

Each of these options is discussed below.

4.1 CHOICE OF SAMPLING MECHANISM

For each variable or uncertain parameter considered in CCGA, a statistical distribution must be obtained or assumed (e.g., estimates of the mean and standard deviation are required for a parameter described by a normal distribution). To create a MC realization, values are sampled simultaneously from the distributions of each of the uncertain or variable parameters. Sampling is carried out using procedures such as MC sampling and LHS, depicted in Figure 2.

In MC sampling, a value between 0.0 and 1.0 is selected at random for each uncertain parameter. The corresponding parameter value is then determined from the parameter's cumulative distribution function. For each subsequent sample, this methodology is repeated. The sampled values will conform to the original distribution if the sample size is sufficiently large.

LHS differs from MC sampling by discretizing the probability density function of the uncertain and variable parameters into n sections of equal area, where n is the total number of samples that will be taken from the distribution. To obtain a random sample using LHS, a section is selected at random. The mean value of the parameter in the section is used. No section is sampled twice. Since the sections are of equal area, the likelihood that a value will be sampled from any particular section is equal. LHS is considered to provide a more accurate representation of parameter distributions with small sample sizes.

4.2 WHEN TO GENERATE NEW REALIZATIONS

An important detail of CCGA is the timing of when new realizations are generated and used. Three approaches were considered.

4.2.1 Approach 1. Pre-Specified Realizations.

In this approach, a test set of realizations is generated before the GA search process begins. All strategies throughout the run are tested with the entire test set. An advantage of this approach, which was used by Ritzel et al. (1994) and Cianawaski et al. (1995), is that the test set needs to be generated only once. A potential disadvantage is that the CCGA solutions will be evaluated with only this test set. If the set does not adequately represent the range of possible realizations, CCGA solutions may not perform well for realizations not considered in the set. Increasing the sample size increases the likelihood of adequately representing the potential realizations, but also imparts a computational burden since each individual in the GA population is tested with every realization in the set for every generation.

4.2.2 Approach 2. Random Selection Each Generation

This approach differs from Approach 1 because a new test set of realizations is created prior to every generation. All strategy evaluations in that generation use this test set. An advantage of this approach over Approach 1 is that it has the potential to consider a greater number of realizations in the GA search process without requiring that there be a larger sample size. A potential disadvantage is the need to generate a new set of realizations in each generation. This process can be computationally expensive if there are a large number of uncertain or variable parameters.

4.2.3 Approach 3. Random Sampling

In Approach 3, every strategy is tested with a new set of realizations. This approach uses more realizations in the search process than do Approaches 1 and 2, which may be considered to be an advantage. Many more realizations must be generated, however. For example, if the sample size is 50, the GA terminates after 200 generations, and the population size is 80, then 50 realizations must be generated using Approach 1; 10,000 must be generated using Approach 2; and 800,000 must be generated using Approach 3. Another disadvantage of Approach 3 is that no two individuals in the same generation are evaluated with the same set of realizations. Good individuals may drop out of the population because they were evaluated with a particularly difficult set of realizations, while poor solutions may survive an easy set of realizations.

4.3 PENALTY FACTOR

A penalty factor is used to decrease the fitness of strategies that meet a constraint for less than the desired frequency of realizations. A penalty factor can be implemented using many different approaches. For the work presented here, the following methodology was used.

To incorporate the chance constraint into the GA, an additional term was added to the fitness calculation. This term was defined as follows:

$$CC = \tau * \left[1 - \left(\frac{R - r_n}{R}\right)\right]^2 \qquad if\ (r_n < R) \qquad (1)$$

or

$$CC = 1 \qquad if\ (r_n \geq R) \qquad (2)$$

where:
τ = *a constant*
CC = a term of the fitness function representing the chance constraint
R = the target reliability (as a fraction)
r_n = *the fraction of realizations for which the constraint is met*

If the target reliability is met, the value of CC is 1.0. If the constraint is not met, the value of CC is determined using Equation 1. Several values for τ were tested. From those trials, a value of 0.7 seemed to perform well.

In the fitness function, CC is weighted to specify its relative importance compared to the other fitness terms. For instance, the fitness function may be represented as:

$$F = w_1 * F' + w_2 * CC \qquad (3)$$

and

$$w_2 >> w_1 \qquad (4)$$

where:
F = the fitness of the strategy after considering the chance-constraint
F'=the fitness of the strategy before consideration of the chance-constraint
w_1, w_2, = *the weights on the terms of the fitness function*

w_2 is chosen to be much greater than w_1, resulting in the chance-constraint being the primary constraint of the GA optimization process. Values for w_2 and w_1 of 1000 and 1, respectively, have been used successfully for these parameters.

5 COMPARISON OF THE SAMPLING ALTERNATIVES FOR A CASE STUDY PROBLEM

5.1 PROBLEM DESCRIPTION

To demonstrate the use of CCGA and compare the alternative sampling approaches, we selected a realistic case study in the field of air quality management. Consider the following scenario:

There are 575 point sources that release nitrous oxide (NOx) emissions into an urban-sized airshed. Based on modeling studies, the regional environmental authority has mandated that a 25% reduction of NOx point source emissions will be necessary to meet the legally-enforced air quality standards.

Two hundred seventy-six of the sources, including all of the largest sources, have discrete control options that may be used to reduce their respective emissions. For each option, there is an estimate of the associated emissions reducion and cost, which were obtained from a USEPA control database.

To achieve the 25% emissions reduction as cost-effectively as possible was a goal of the local governtment in this case study. A deterministic integer programming model was used to identify the least cost set of control options to achieve the target reduction level. As an afterthought, the local government used a Monte Carlo simulation to test the reliability that the strategy would actually meet the reduction target. Uncertainties were considered in emissions estimates, and in control costs and removal efficiencies. Each was modeled with a normal distribution and a coefficient of variance of 10%. The simulation predicted that the least cost strategy only achieved the target reduction level for 47% of the 200 MC realizations.

This was unacceptable to the local government, which demanded a reliability of at least 75%. Another option was to use a chance-constrained optimization to solve the problem. Since many of the uncertainties were left-sided, the problem was difficult to model with traditional chance-constrained programming techniques. Instead, CCGA was used.

This problem can be represented mathematically as:

$$\text{minimize} \quad C = \sum_{i=1}^{N} \sum_{t=1}^{T_i} \left((u_i * e_{i,t}) * c_{i,t} * x_{i,t} \right) \qquad (5)$$

subject to:

$$P\left(\frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \left((u_i * e_{i,t}) * x_{i,t} \right)}{\sum_{i=1}^{N} u_i} \geq E \right) \geq \alpha \qquad (6)$$

$$x_{i,t} \in \{0,1\} \quad \forall i = 1, N; t = 1, T_i \qquad (7)$$

$$\sum_{t=1}^{T_i} x_{i,t} = 1 \quad \forall i = 1, N \qquad (8)$$

where:
C = the total cost of the strategy
i = source index
N = the number of sources
t = control technology index
T_i = the number of control options at source i
u_i = the uncontrolled emissions from source i
$e_{i,t}$ = the control efficiency of technology t at source i
$c_{i,t}$ = the cost per unit of emissions removed using control t at source i
$x_{i,t}$ = a binary variable that is 1 if control t at source i is being used or 0 if it is not
E = a target emissions reduction level. For this example, 0.25 was used.
P = the probability that the emissions reduction constraint will be met
α = the target reliability

Equation 5 states the objective of the formulation: minimization of cost. Equation 6 specifies that the emissions constraint should be met with a probability of α or greater. Equations 7 and 8 require that controls are mutually exclusive and discrete, i.e., a single control option must be used at each source, and the option is applied to the entire waste stream. "No control" is assumed to be one of the control options.

In this formulation, there were a total of 1127 stochastic parameters. The distributions for these parameters were assumed to be uncorrelated. To obtain a MC realization, a value was sampled independently from the assumed distribution of each stochastic parameter.

5.2 METHODOLOGY

A real-value GA was adapted to solve the case study problem. Potential solutions were represented as an array of 276 genes: one gene for each controllable source. The value of each gene ranged from 0.0 to 1.0. This value was then translated to specify which discrete control option to use at the corresponding source. While other representations may have been superior, this representation was used to maintain compatability with concurrent research.

GA search parameters used in these trials included a population size of 80, a mutation rate of 0.005, tournament selection with a tournament size of 2, a

Table 1. Results of various CCGA configurations. For each trial, the clock time, total number realizations required, and the performance of the solutions in a Monte Carlo simulation is listed.

Config-uration	Sampling Technique	Appr-oach	Sample Size	# of Trials	Run-time	Evaluations until stopping criterion met		Reliability from Monte Carlo simulation	
					(min)	Mean	Stdev	Mean (%)	Stdev (%)
1	MC	1	10	23	5.7	95440	19840	65.62	13.30
2	MC	1	20	19	10.0	201440	56000	67.85	9.87
3	MC	1	50	19	23.0	447600	89600	71.05	6.89
4	MC	1	100	22	44.8	882180	176880	70.68	4.30
5	MC	2	10	20	5.8	90880	16720	67.96	10.72
6	MC	2	20	19	10.2	183520	31170	67.71	9.91
7	MC	2	50	16	23.7	493200	96960	71.63	6.77
8	MC	2	100	20	47.2	876800	173620	71.39	4.61
9	MC	3	10	20	12.4	90080	17590	68.56	11.44
10	MC	3	20	19	24.4	173390	36890	67.81	9.66
11	MC	3	50	11	80.8	469450	94850	71.38	7.63
12	MC	3	100	12	240.0	862000	156250	70.43	4.91
13	LHS	1	10	20	5.8	91600	16820	72.17	4.34
14	LHS	1	20	20	10.0	179120	38620	71.47	2.33
15	LHS	1	50	20	23.0	425400	85970	74.53	0.82
16	LHS	1	100	24	45.5	875600	232690	74.53	0.81
17	LHS	2	10	20	5.8	81680	10820	74.67	1.77
18	LHS	2	20	19	10.2	177680	38740	72.51	0.53
19	LHS	2	50	20	23.7	413200	86550	74.77	0.64
20	LHS	2	100	20	48.0	778800	135220	74.07	0.37
21	LHS	3	10	10	12.4	93840	11840	75.56	0.69
22	LHS	3	20	10	24.4	190880	26770	72.88	0.62
23	LHS	3	50	6	81.0	428000	123050	75.47	1.06
24	LHS	3	100	11	244.3	872000	123050	74.34	0.71

crossover rate of 1.0 (i.e., pairs of selected individuals always mate), and uniform crossover at a rate of 0.2 (i.e., 20% of genes crossed during mating). In addition, of the genes that crossed, 35% underwent linear recombination.

The stopping criterion was defined as follows:

- For each generation, g, after generation 20, the average fitness in the population is compared with the mean average fitness of the previous 20 generations.
- If the average fitness of the current generation is better, continue. Otherwise, terminate.

All combinations of sampling methodologies (LHS and MC sampling) and choices for when to generate realizations (Approaches 1, 2, and 3) were evaluated for sample sizes of 10, 20, 50, and 100. Because GAs are stochastic search techniques, the quality of the solutions that they identify can be dependent on the random number seed. To estimate the sensitivity of each configuration to the random number seed, each configuration was tested with approximately twenty seeds. Some configurations were tested with several fewer than twenty seeds because of resource limitations. These runs were performed on a low-end 100-MHz Unix workstation with 64 megabytes of memory. After each run terminated, the best solution identified in that run was tested with a new set of MC realizations to estimate to what extent the 75% reliability target was met. The MC simulation used a new set of 350 realizations obtained with LHS.

5.3 RESULTS AND DISCUSSION

Table 1 summarizes the results of the trials of each of the 24 configurations. Figure 3 compares how the solutions from each configuration performed with respect to reliability (from post-screening) and the total number of realizations that were tested before the stopping criterion was met. Observations from these results are grouped into two categories: observations on solution reliability and on the total number of realizations evaluated.

5.3.1 Observations on Solution Reliability

The results show that CCGA-derived solutions are generally much closer to the reliability target when LHS is used than with MC sampling. This observation is particularly the case for smaller sample sizes. LHS performed well with a sample size of 50, and appeared to perform adequately with sample sizes as low as 20 or 10. In contrast, MC solutions were inferior to all LHS solutions, even for sample sizes of 100.

The results also support the premise that Approaches 2 and 3 provide better solutions given small sample sizes

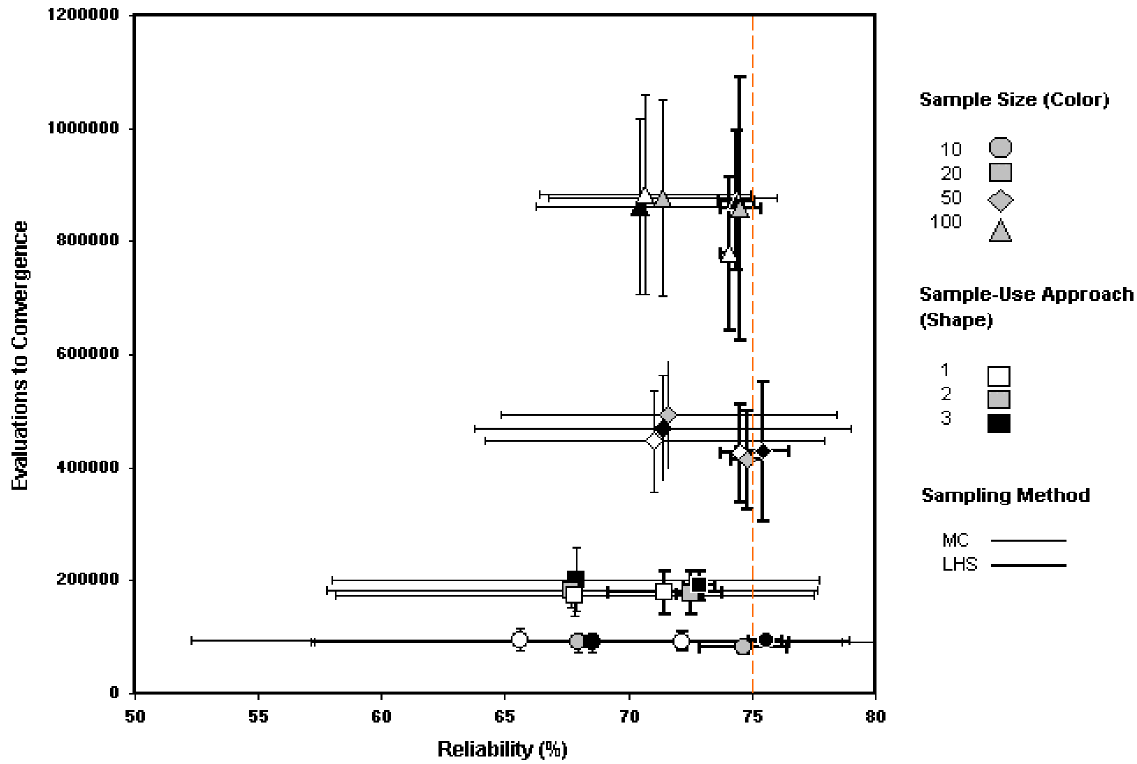

Figure 3. Comparison of the reliability and evaluations until convergence for each configuration. Error bars represent one standard deviation.

since many more realizations are considered during the course of each run. The disruptiveness of changing the trial set during the run does not appear to outweigh this benefit. Also, while creating new realizations for each evaluation does not appear to be considerably more disruptive than generating realizations each generation, there appears to be little if any benefit. The computational time needed to generate the additional realizations is apparent in the clock times reported in Table 1, making Approach 2 a better choice of the two.

It is intriguing to note that solutions identified using LHS and a sample size of 10 generally were closer to the 75% target than those obtained using a sample size of 20. Examining the relationship between the reliability target and the sample size provides an explanation for this trend. For example, with a sample size of 10, a solution that meets the reduction target for eight realizations is not penalized. A solution that meets the target for only seven realizations is penalized. But a solution cannot meet the target for 7.5 of the 10 realizations; therefore, for sample sizes of 10, CCGA used an effective reliability target of 80%, not 75%. The CCGA solutions were approximately 5% below the effective 80% target. Interpreted in this manner, the results using a sample size of 10 are consistent with the expected trend of solution quality improving as sample sizes increases.

5.3.2 Observations on the Number of Evaluations Required

Sample size had a direct and significant impact on the number of realizations that were evaluated for each configuration. For example, if two configurations required 100 generations each to converge, but one used a sample size of 50 and the other 100, the configuration using 100 would require roughly twice as many realizations to be evaluated.

5.4 RECOMMENDATIONS

The results presented in this section indicate that the best performance of the alternative configurations, considering solution quality and computational requirements, are:

- Latin Hypercube Sampling- LHS greatly improved the quality of the CCGA solutions
- Approach 2 - Both Approaches 2 and 3 outperformed Approach 1. In general, it is suggested that Approach 2 be used instead of Approach 3 since it requires the generation of far fewer realizations.
- Small sample sizes of as little as 10 to 50 - Sample sizes within this range produced results of nearly the same quality as larger sample sizes, but with considerably less computations.

6 CONCLUSIONS

A chance-constrained GA (CCGA) differs from a general GA formulation because the fitness of an individual is evaluated based on its performance in a MC simulation. In this paper, several CCGA approaches using different sampling schemes were compared for an ELC air quality management problem. The results suggest that CCGA is a viable approach for complex problems that are not readily solved with traditional chance-constrained approaches in mathematical programming, such as chance-constrained programming (CCP).

CCGA applications in the literature have used a large number of Monte Carlo realizations in each fitness evalutation. As a result, the computational burden of these applications was several orders of magnitude above that of deterministic runs. We tested a number of alternative configurations, with the goal of reducing CCGA's computational burden and improving the quality of solutions. Our results suggest that using LHS as a sampling approach will improve the solution quality substantially. This improvement allows smaller sample sizes to be used. We obtained results of reasonable quality with sample sizes of as little as 10 to 50. There also appeared to be a benefit associated with generating a new set of realizations each generation and testing each individual in the generation with that set.

The observation that a small number (10-50) of realizations can be used in the CCGA algorithm is important. To a statistician, a MC simulation with 10 to 20 realizations would be woefully inadequate for a problem such as the emissions reduction case study, a problem with over 1000 uncertain and variable parameters. Yet, in the trials presented in this chapter, runs with these numbers of samples performed well. How was CCGA able to use such small sample sizes?

A possible explanation can be found in schema theorem (Goldberg, 1989), which describes GA search as a process of identifying and matching components of solutions, or building blocks. Schema theorem would seem to suggest that CCGA is using the realizations in each evaluation to assist in its selection and combination of the appropriate solution building blocks. These blocks are then carried and re-evaluated and recombined from generation to generation. The final solution, therefore, is not a function of the performance of the population's solutions for the last set of realizations, but a set of building blocks assembled over a much larger set of realizations – those of all the previous generations. We hope to investigate this hypothesis in future work.

Acknowledgements

We would like to thank the other researchers who have contribued to this work, including: Downey Brill and John Baugh of NC State University and Steve Fine of MCNC. Funding was provided by the United States Environmental Protection Agency's High Performance Computing and Communications Program, MCNC, and the NC State Industrial Extension Service.

Disclaimer

Although this work has been funded by the USEPA MCNC, and NC State University, it has not been subjected to the internal review processes of these organizations. The viewpoints and ideas expressed are those of the authors and do not necessarily represent those of the funding organizations.

References

Charnes, A., and Cooper, W. W. (1963). Deterministic equivalents for optimizing and satisficing under chance constraints. *Operations Research*, 11, 18-39.

Cieniawski, S. E., Eheart, J. W., and Ranjithan, S. (1995). Using genetic algorithms to solve a multiobjective groundwater monitoring problem. *Water Resources Research*, 31(2): 399-409.

Cooper, W. W., Hemphill, H., and Sullivan, D. (1997). Survey of mathematical programming models in air pollution management. *European Journal of Operations Research*, 96.

Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA., 1989.

Loughlin, D.H. Genetic algorithm-based optimization in the development of tropospheric ozone control strategies: least cost, multiobjective, alternative generation, and chance-constrained applications. Doctoral Dissertation, Civil Engineering Dept., North Carolina State University, Raleigh, NC, 1998.

Loughlin, D. H., Neal, J. K., Ranjithan S., Brill, E. D., Baugh, J. W., Fine, S. S., Vukovich, J. M., and Eyth A. (1996). Design of oxidant air quality management strategies using genetic algorithm-based optimization in a decision support system. In *Proc. of Computing in Environmental Resoure Management*, A&WMA Spec. Conf.

Morgan, M. G., and Henrion, M. Uncertainty: A guide to dealing with uncertainty in quantitative risk and policy analysis. Cambridge Press, New York, NY, 1995.

Painton, L., and Cambell, J. (1995). Genetic algorithms in optimization of system reliability. *IEEE Transactions on Reliability*, 44(2), 172-178.

Painton, L., Diwekar, U. (1995). Stochastic annealing for synthesis under uncertainty. *European Journal of Operations Research*, 83, 489-502.

Ritzel, B. J., Eheart, J. W., and Ranjithan, S. (1994). Using genetic algorithms to solve a multiple objective groundwater pollution containment problem. *Water Resources Research*, 30(5), 1589-1603.