WHICH ARE BETTER, PROBABALISTIC MODEL-BUILDING GENETIC ALGORITHMS (PMBGAS) OR SIMPLE GENETIC ALGORITHMS (SGAS)? A COMPARISON FOR AN OPTIMAL GROUNDWATER REMEDIATION DESIGN PROBLEM

BY

RACHEL EILEEN ARST

B.S., Cornell University, 2000

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Environmental Engineering in Civil Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2002

Urbana, Illinois

ABSTRACT

Simple genetic algorithms (SGAs) have been used to solve many water resources optimization problems. SGAs are appealing because they can solve non-convex, discrete, or discontinuous problems of any form. However, they can be very slow and are not guaranteed to converge to the optimal solution. In this paper, two new advanced genetic algorithms, the Extended Compact Genetic Algorithm (ECGA) and the Bayesian Optimization Algorithm (BOA), are tested on an optimal groundwater remediation design problem to determine whether they show promise for reducing computational time and improving reliability relative to the SGA.

These algorithms have been shown to be more effective than the SGA at solving synthetic problems because they identify the links between the building blocks (short, highly fit chromosome sequences), called linkage learning. Both BOA and ECGA use linkage learning, but in slightly different ways. The ECGA uses probabilistic modeling of the population to learn linkage. BOA improves on that approach by using Bayesian networks to estimate joint distributions that are used to generate new candidate solutions. The performances of these algorithms are compared with simple genetic algorithms to demonstrate their computational power for several different types of remediation design problems with different levels of complexities.

It was found that BOA was able to solve problems of different complexities 30-2400% faster than the ECGA or SGA, but for the complex problems, BOA had a slightly lower solution quality (less than 1.7% higher cost than SGA). ECGA's performance was more erratic, with improvements in solution quality and time in some cases, but deterioration of both in others. This inconsistent performance likely stems from the need for larger population sizes with advanced GAs as compared to the SGA, since their model-building capabilities have been shown to require large populations to solve test problems effectively. However, the reductions in solution quality are relatively minor compared with the substantial gains in the computational performance, demonstrating significant overall promise for these algorithms.

ACKNOWLEDGMENTS

First, I would like to thank my advisor, Dr. Barbara Minsker, for the opportunity to work on this exciting research project. I am thankful for her support, guidance, and insight throughout the past two years.

I would like to thank several members of my research group. Dr. Patrick Reed provided many helpful discussions early on during my project. Felipe Espinoza and Meghna Babbar contributed not only their helpful advice but also their excellent debugging skills. Yong Liu and Jun Lee were both very ready to lend a hand with their extensive programming language knowledge. I could not have finished this project in a timely manner without the help of Shengquan Yan and his inter-language programming expertise.

I would also like to thank Dr. David Goldberg and his laboratory for their contributions to this work, as well as the consultants at the National Center for Supercomputing Applications (NCSA).

Finally, I would like to thank my friends and family for their support during this process.

This material is based upon work supported by, or in part by, the U. S. Army Research Office under grant numbers DAAD19-00-1-0389 and DAAD19-001-1-0025. The computational resources of NCSA were used in the early stages of this project.

TABLE OF CONTENTS

LIST OF FIGURES

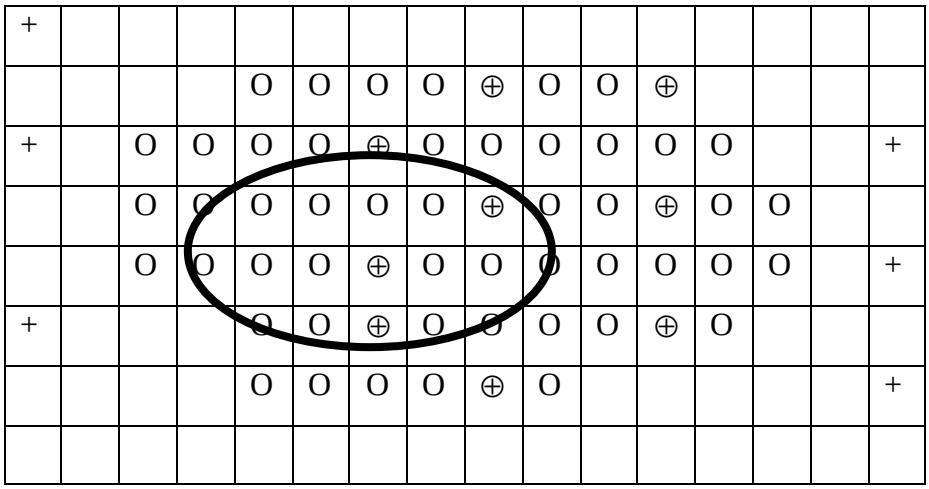
LIST OF TABLES

1. INTRODUCTION

Simple genetic algorithms have been used for many water resources applications, including groundwater remediation design, optimal reservoir system operation, calibrating rainfall-runoff models, water pipe replacement, and optimal remediation policy selection (e.g., Wang 1991; Wang and Zheng 1997; Wardlaw and Sharif 1999; Aksoy and Culver 2000; Smalley et al. 2000; Dandy and Engelhardt, 2001). The primary reason that genetic algorithms (GAs) have been chosen to solve such problems is that they are able to solve any type of function, including non-convex, discrete, and discontinuous problems (Goldberg 1989). However, they can also be slow and are not guaranteed to converge to the optimal solution. The purpose of this study is to investigate whether new advanced GAs, called probabilistic model-building GAs, or PMBGAs, can solve groundwater remediation design problems of different complexities faster and more reliably than the simple GA.

2. CASE STUDY APPLICATION

A groundwater remediation system for a heterogeneous, isotropic, and confined hypothetical BTEX-contaminated aquifer was used to compare the efficiency of ECGA, BOA, and SGA. This system has been studied in previous works (Smalley et al. 2000; Gopalakrishnan et al. 2002 in press) and is modeled on the Borden aquifer.



+ Observation wells (15)

O Possible remediation wells (58)

Fig. 1. Plan view of the case study aquifer

Pump-and–treat technology was used with a goal of minimizing the pumping cost to meet a target risk level at the 15 observation well locations shown in **Fig. 1**. There are 58 possible well locations for 3 possible remediation wells to be placed as shown in **Fig. 1**. The extracted water is treated by air stripping.

The dimensions of the modeled area shown in **Fig. 1** are 240 by 480 meters. This area was modeled using two grid sizes, 16 by 8 (shown in Fig. 1) and 48 by 24, with MODFLOW (McDonald and Harbaugh 1988) and RT3D (Clement 1997; Clement et al. 1998) finite difference models to predict flow and contaminant transport. The transport was modeled with

advection, dispersion and linear adsorption. An analytical model (Smalley et al. 2000) was then used to predict risk at an exposure point 200 meters downgradient from the site shown in **Fig. 1**. The aquifer characteristics are presented in **Table 1**.

Table 1. The case study aquifer characteristics

Parameter	**Value**
Average hydraulic conductivity K (m/day)	6.12
Porosity (n)	0.3
Longitudinal dispersivity α_L (m)	15.00
Transversal dispersivity α_T (m)	3.00
Aquifer thickness b (m)	20.00
Retardation coefficient (R)	1.41

There are four decision variables in this problem: pumping well location, pumping rate (u_i), injection or extraction (Y_i), and installation (X_i) where:

$$X_i = \begin{cases} 1 & \textit{if a pumping well is installed at location } i \\ 0 & \textit{Otherwise} \end{cases}$$

$$Y_i = \begin{cases} 1 & \textit{for injection from well } i \\ 0 & \textit{for extraction from well } i \end{cases}$$

The objective function that was used to determine the cost is as follows:

$$\text{Min } C_{TOT} = C_{REM}(u_i) + C_{MON}(X_i) + C_{SYST}(u_i)$$

where:

C_{REM} = the capital and operating costs for the wells

C_{MON} = the cost of on-site monitoring

C_{SYST} = the additional capital and operating costs for the ex-situ treatment system

The details of the monitoring and remediation costs (C_{REM} and C_{MON}) are given by Smalley et al. (2000), while the details of the C_{SYST} evaluation, done using RACER (1999), a parametric cost modeling system, are presented by Gopalakrishnan et al. (in press 2002).

The constraints, which Smalley et al. (2000) present in more detail, are applied to the human health risk, the pumping rates, and the hydraulic drawdowns. The constraints are as follows:

$$Risk_{t,k}^{TOTAL} = Risk_{t,k}^{w} + Risk_{t,k}^{shw} + Risk_{t,k}^{nc} \leq \quad TR \,\forall t, \forall k$$

where:

$Risk_{t,k}^{w}$ = the cancer risk from ingestion of contaminated drinking water

$Risk_{t,k}^{shw}$ = the cancer risk from inhalation of volatiles from contaminated water in the shower

$Risk_{t,k}^{nc}$ = the cancer risk from inhalation of volatiles from contaminated water during non-consumptive uses (except from showering)

$Risk_{t,k}^{TOTAL}$ = the total lifetime carcinogenic risk

TR = the target risk level

t = time

k = location

This human health risk constraint ensures that the lifetime risk that is evaluated at locations k and at times t is less than the target level, TR. The constraints on the pumping rates and hydraulic heads are as follows:

$$u_{min,i} \leq \left| u_i \right| \leq u_{max,i} \qquad \forall i$$

$$h_{min,l} \leq h_{i,l}(u_i) \leq h_{max,l} \qquad \forall i, \forall l$$

where:

$u_{min,i}$ = minimum pumping rate for remediation well i

$u_{max,i}$ = maximum pumping rate for remediation well i

$h_{i,l}$, = computed hydraulic head (meters) for remediation well i at location l

$h_{min,l}$ = minimum hydraulic head (meters) for remediation well i at location l

$h_{max,l}$ = maximum hydraulic head (meters) for remediation well i at location l

The fitness of the design can then be evaluated as follows:

$$\text{Fitness} = C_{TOT} + \omega_1 \cdot \text{Risk violation} + \omega_2 \cdot \text{Head violation}$$

where:

ω_1 = the penalty weight for the risk constraint

ω_2 = the penalty weight for the head constraint

Both penalty weights are set to 1000 in this case study. A penalty was not needed for the pumping rate since the binary representation of this decision variable in the GAs will limit it directly. In this case, lower fitness is better, so the function is minimized.

The three different cases that were used to test the GAs in this study are summarized in **Table 2**. The relative complexity is based on the time each algorithm takes to calculate the fitness for each individual. The pumping rates for the remediation wells were either fixed at an optimal solution found in previous runs or left variable.

Table 2. Case descriptions

Case	Relative Complexity	Grid Size	Population Size	Pumping Rate
1	Easy	16 by 8	240	Fixed
2	Moderate	16 by 8	240	Variable
3	Hard	48 by 24	200	Variable

3. METHODOLOGY

This section describes our methodology, including an overview of the SGA and a description of the PMBGAs used in this study.

3.1 Simple Genetic Algorithm Basics

GAs are search algorithms based on the survival of the fittest theory. The search is usually done with a group or population of binary strings, which are encoded forms of the decision variables. Each string is called a chromosome and each binary bit within the string is called a gene. There are three operators used in the simple GA to model the process of natural selection: selection, crossover, and mutation. First, the objective function, also called the fitness function, is evaluated for each string. Selection occurs when the individuals with higher fitness values are assigned higher probabilities of producing offspring for the next generation. Therefore, the highly fit chromosomes will have a larger number of copies in the succeeding generation. There are several different types of selection used in GAs. Tournament selection was used in this work, which allows only the fittest individual from a randomly selected subset of the population to be put into a mating pool. Mu plus lambda selection was used in the SGA to ensure proper convergence. Mu refers to the number of strings (the parents) from which the new population will be generated and lambda refers to the number of offspring generated by mu parents. The parents and offspring are then combined and the best mu are selected from the pool of parents and offspring.

After the mating pool is established, crossover takes place. Individuals in the mating pool are mated by randomly selecting place(s) to divide the two chromosomes and then exchanging the pieces. After crossover, mutation randomly switches a bit or bits in the chromosome with a user-specified probability. Using these operations, more fit members of the population are created over time and the population evolves to optimal or near-optimal solutions.

The way the GA assembles optimal solutions can be described by building block theory. Building blocks are short, low order, and highly fit groups of binary digits in the chromosome

(Goldberg 1989). Certain combinations of building blocks are more fit than others and will ultimately take over through the survival-of-the-fittest operations.

The SGA parameters were set using the 3-step method developed by Reed et al. (2000) and are shown in **Table 3**.

Table 3. SGA parameters

Parameter	Value
Population size: 16 by 8 cases	240
Population size: 48 by 24 case	200
Tournament size	4
Crossover probability	0.75
Mutation probability: 16 by 8 cases	0.004
Mutation probability: 48 by 24 case	0.005
Convergence criteria	100%

The convergence was determined by calculating the percentage of fitness values that reached the minimum fitness value. When 100% of the individuals reached the minimum fitness value, the run was terminated.

3.2 Probabilistic Model Building GAs (PMBGAs)

A class of GAs called competent GAs have recently been introduced (Goldberg 1999). These competent GAs are able to solve hard problems (ones that earlier GAs could not) quickly, accurately and reliably (Sastry and Xiao 2001). Linkage learning GAs are competent GAs that identify the important building blocks and conserve them during crossover. Crossover can disrupt building blocks and impede the GA from finding good solutions, so models of the relationships among building blocks can help to ensure that correlated genes stay together for better algorithm performance. Linkage learning GAs use model-building instead of genetic recombination to guide the search (Harik 1999), building an explicit model of promising chromosomes as opposed to the implicit model that is created by genetic recombination (Pelikan and Goldberg 2000). The ECGA and the BOA are linkage learning GAs that use probabilistic

chromosome modeling techniques and are termed probabilistic model-building genetic algorithms. This type of algorithm, specifically the ECGA, has been used successfully by Sastry and Goldberg (Sastry and Goldberg 2000) to optimize a binary fluid power cycle. These algorithms are described in detail in the following sections.

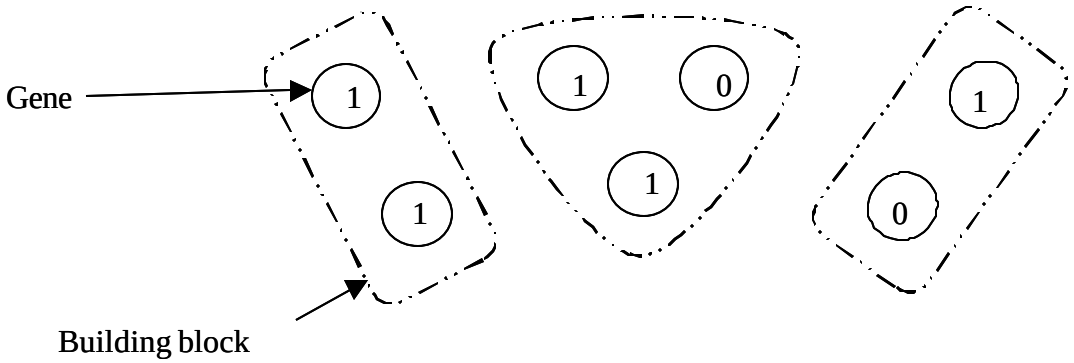


Fig. 2. ECGA chromosome

3.2.1 The Extended Compact Genetic Algorithm (ECGA)

The class of probability distributions used in the ECGA is marginal product models (MPMs). MPMs represent relationships among genes in the chromosomes through products of marginal distributions. MPMs are implemented in ECGA by clustering genes into building blocks; each cluster is treated as a group that cannot be disrupted by crossover. This is represented in **Fig. 2**, which is adapted from Pelikan and Goldberg (2000). Each of the circles represents a gene, while the dashed lines around the genes represent the building blocks. Crossover can only be performed between the building blocks.

The MPM in the ECGA uses a MDL (minimum descriptions length) model. These models are built on the premise that when all other things are equal, the simpler distributions are better than the complex ones. In each generation of the ECGA, a greedy search is performed to identify a probability distribution that best models the correlations among chromosomes within good members of the population, in the most compact fashion. The ECGA is based on the fact that learning probability distributions over multi-variate spaces is equivalent to linkage learning (Sastry and Goldberg 2000) and that linkage learning is equivalent to building block identification (Harik 1999). The ECGA is essentially identifying the building blocks by learning

the probability distributions. The clustering approach then allows these building blocks to be preserved during crossover, which should improve performance.

In the ECGA, the initial random population is generated, the fitness of the chromosomes is evaluated, and tournament selection is performed in the same manner as in the SGA. The MPM model is then built from the mating population using the MDL. Crossover is then performed as in the SGA, but is only allowed to occur between clusters in the MPM. This modified form of crossover preserves the critical relationships among the genes of highly fit chromosomes. This procedure, which is shown in **Fig. 3**, is followed until a specified percent convergence is reached.

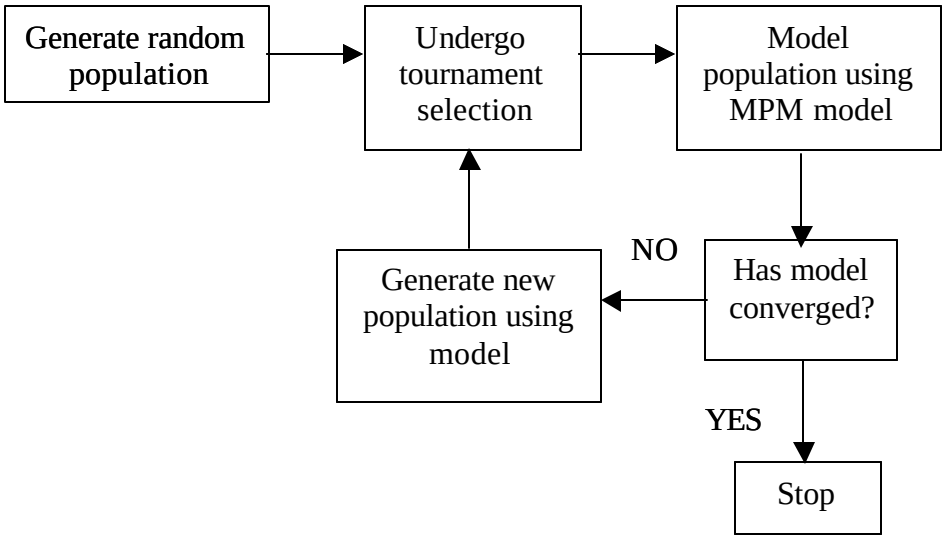


Fig. 3. ECGA flow chart

The ECGA and SGA share all of the parameters that are listed in **Table 2** except for mutation probability. To compare the ECGA and the SGA using the same computational resources, the ECGA parameters were set to the values found to be optimal for the SGA (listed in **Table 2**). ECGA does not use mutation, so it was not necessary to set the mutation probability.

3.2.2 The Bayesian Optimization Algorithm (BOA)

BOA builds Bayesian networks as its probabilistic model (Pelikan, Goldberg, Cantú-Paz 2000 and Pelikan, Goldberg, and Sastry 2000). A Bayesian network is a directed acyclic graph, where

each node represents a gene in the chromosome and the edges of the graphs represent the critical correlations among the genes that create highly fit chromosomes. **Fig. 4** (adapted from Pelikan and Goldberg 2000) shows the correlations between each gene in a sample chromosome. The value of Gene A depends on the values of Genes B and C, which is shown by the arrows that point from Genes B and C to Gene A.

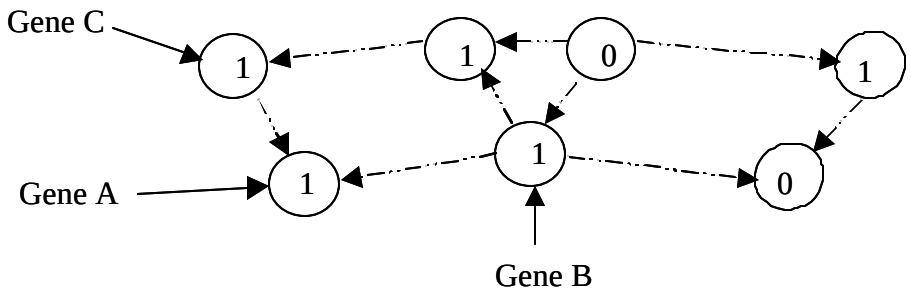


Fig. 4. BOA chromosome

Like the simple GA, the initial population is generated randomly. Any selection method can be used to select the best chromosomes from the initial population, but in this work tournament selection was used. A Bayesian network is then created using a greedy search in which edges between the most highly correlated genes are added to the network first. New chromosomes are generated using the joint distribution represented by the network, starting from the root node. The new chromosomes are then added to the old population, replacing a specified percentage of the old population (for this application, 75%). This process continues until convergence is reached. This sequence is shown in **Fig. 5**. BOA is able to work independently of the building block length and can identify the building blocks on the fly by using the information from the selected chromosomes. For more details on the algorithm, see Pelikan, et al. (1999).

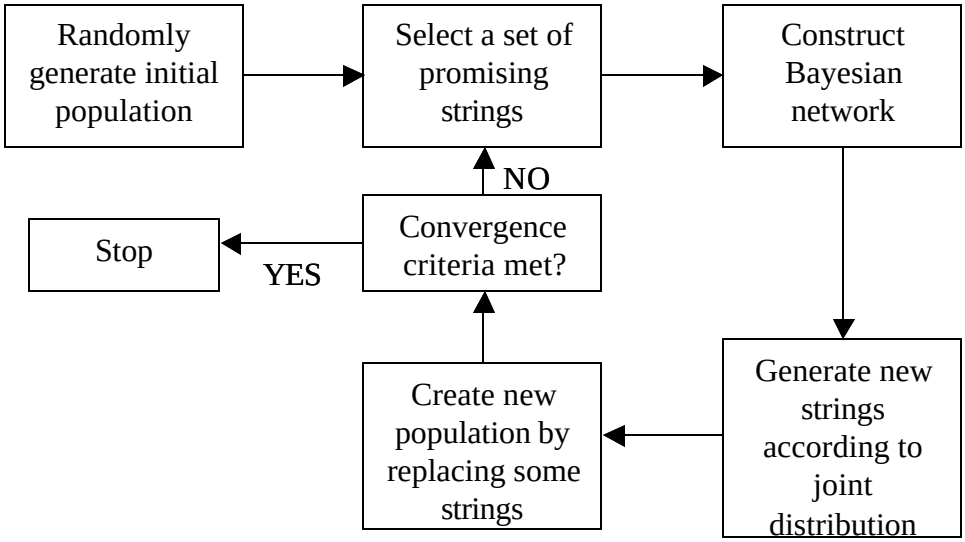


Fig. 5. BOA flow chart

As with ECGA, the BOA parameters were also set to the optimal SGA parameters for the algorithm comparison (see **Table 2**). The BOA parameters that are not used in the SGA are presented in **Table 4** The offspring percentage is the number of offspring to generate expressed as a percentage of the population size. The maximum incoming edges are the maximum number of edges that may join a single node. This sets a limit on the number of interactions that can be covered by the model. Since the metric controls the simplicity of the model, this parameter should be set to a high value to ensure convergence (Pelikan 2000).

Table 4. BOA parameters

Parameter	Value
Offspring percentage	75
Maximum incoming edges	20

4. RESULTS

This section presents the results for the three test cases, starting with the simplest problem and increasing in complexity.

4.1 Case 1: 16 by 8 Grid Size, Fixed Pumping Rates

In Case 1, BOA was able to find the best solution in the least amount of time. These results can be seen in **Fig. 6** and **Fig. 7**, which show the average and minimum costs for each generation. **Fig. 7** shows that BOA found a very good solution quickly, and, as shown in **Fig. 6**, converged to that solution more quickly than either ECGA or SGA. BOA solved the problem 34% faster than the ECGA and 116% faster than SGA. Its solution quality was 0.1% better than ECGA and 0.2% better than SGA.

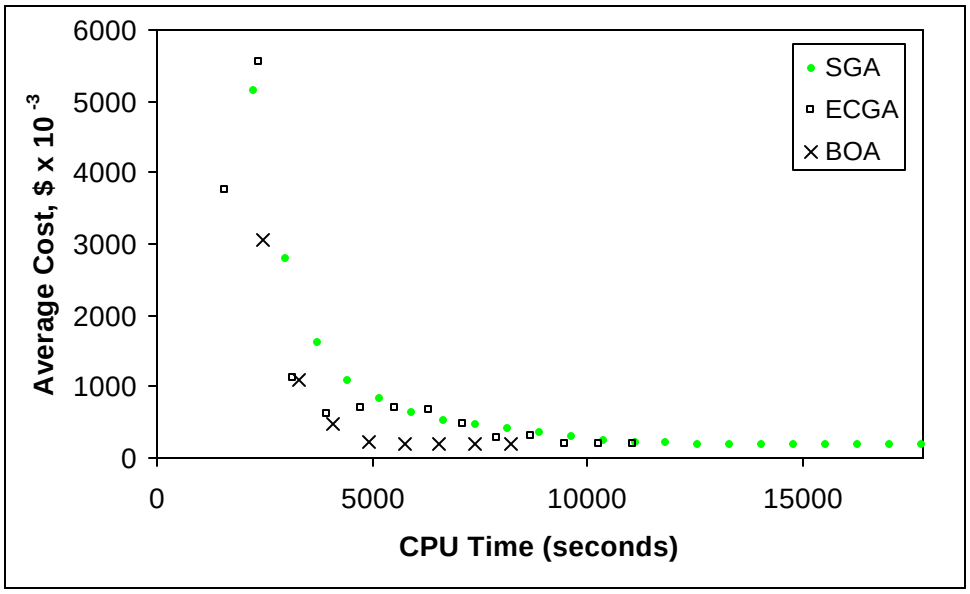


Fig. 6. Case 1: Comparison of SGA, ECGA, and BOA average costs for a 16 by 8 grid and fixed pumping rates

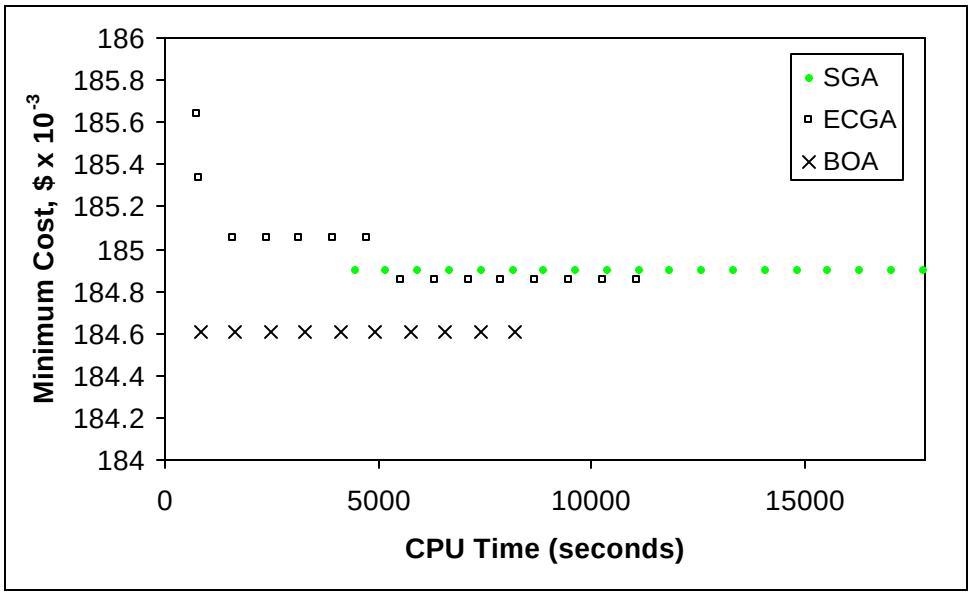


Fig. 7. Case 1: Comparison of SGA, ECGA, and BOA minimum costs for a 16 by 8 grid and fixed pumping rates

4.2 Case 2: 16 by 8 Grid Size, Variable Pumping Rates

In Case 2, SGA found the best solution but BOA was able to solve the problem in the least amount of time. As shown in **Fig. 8** and **Fig. 9**, BOA solved the problem 250% faster than ECGA and 240% faster than SGA. SGA found a solution that was 0.4% better than the one ECGA found and a solution that was 0.7% better than the one BOA found. Unlike Case 1, ECGA did not perform as well as the SGA for this case, requiring 4% more computational time to find the same solution. As in Case 1, BOA identified the good solutions early on (see **Fig. 9**) and was able to converge to that solution faster than the other algorithms (see **Fig. 8**). Please note that since the different cases were run on different PCs, the CPU times cannot be compared between cases 1, 2, and 3.

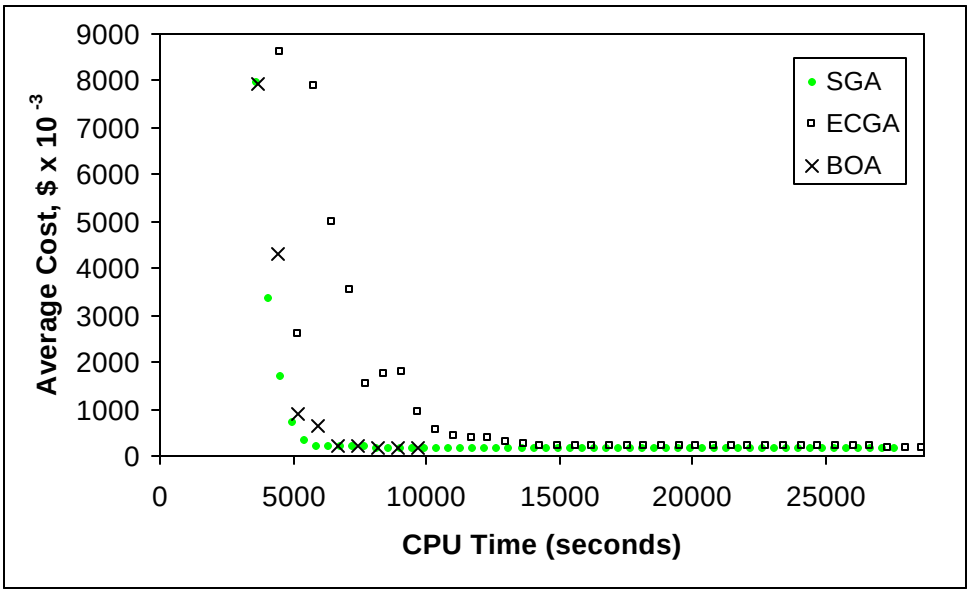


Fig. 8. Case 2: Comparison of SGA, ECGA, and BOA average costs for a 16 by 8 grid and variable pumping rates

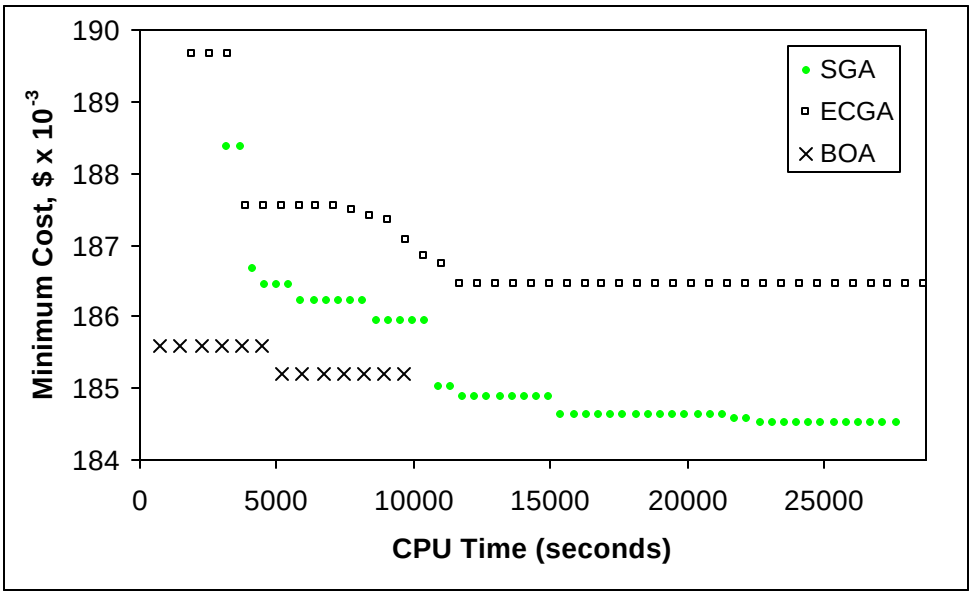


Fig. 9. Case 2: Comparison of SGA, ECGA, and BOA minimum costs for a 16 by 8 grid and variable pumping rates

4.3 Case 3: 48 by 24 Grid Size, Variable Pumping Rates

In Case 3, BOA solved the problem 900% faster than ECGA and 240% faster than SGA. However, the ECGA had the best solution quality for this case, improving on BOA's solution by 1.7% and SGA's solution by 0.3%. The results can be seen in **Fig. 10** and **Fig. 11**.

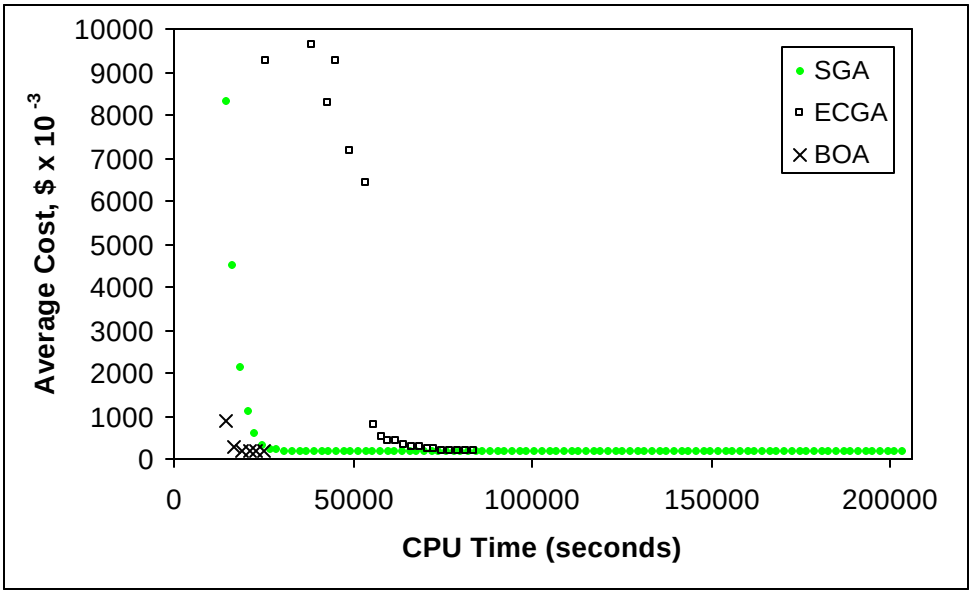


Fig. 10. Case 3: Comparison of SGA, ECGA, and BOA average costs for a 48 by 24 grid and variable pumping rates

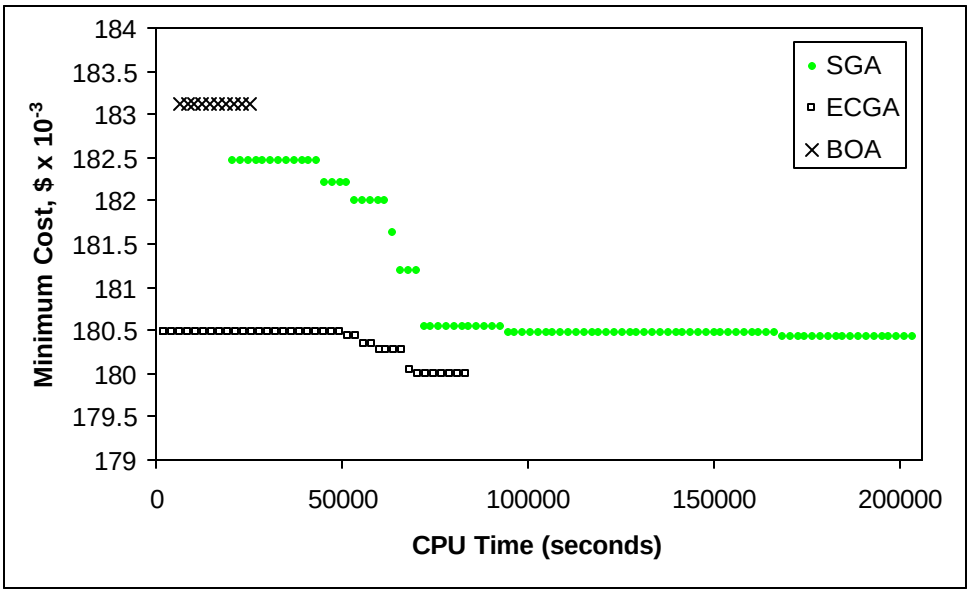


Fig. 11. Case 3: Comparison of SGA, ECGA, and BOA minimum costs for a 48 by 24 grid and variable pumping rates

For these three algorithms, there is a tradeoff between solution quality and time. BOA solved the problem 30-2400% faster than the other algorithms, but for the more complex cases (Case 2 and Case 3), the solution quality was 0.4-1.7% worse. Moreover, the solution quality for BOA became worse as the problem became more complex. It is likely that the small population sizes used, which were optimal for the SGA, caused the degradation in solution quality. Sastry and Goldberg (2000) and Pelikan et al. (2000) found that larger population sizes were needed for the ECGA and BOA, respectively, to reliably identify the best solutions for test functions. The larger population is needed to ensure that the probabilistic models used to generate the solutions are representative.

Table 5 shows a comparison of solution quality, time, and generations to convergence for each of the GAs and each case. It is interesting to note that the generations to convergence do not necessarily correspond with the time to convergence. For instance, in Case 2, the ECGA solves the problem in fewer generations than the SGA, but in slightly more time. This stems from the fact that ECGA, as well as BOA, often needs more time in addition to the normal GA operation time to build the probabilistic model in each generation.

Table 5. Comparing results for BOA, ECGA, and SGA

Case No.	Case Description	Genetic Algorithm	Generations to Convergence	Time to Convergence, seconds	Cost, $ x 10^{-3}
1	16 by 8 grid size, Fixed pumping rates	BOA	9	8,216	184,600
		ECGA	14	11,046	184,900
		SGA	23	17,736	184,900
2	16 by 8 grid size, Variable pumping rates	BOA	13	9,665	185,200
		ECGA	43	28,651	186,400
		SGA	60	27,576	184,500
3	48 by 24 grid size, Variable pumping rates	BOA	12	25,169	183,100
		ECGA	39	83,538	180,000
		SGA	98	203,200	180,400

5. SUMMARY AND CONCLUSIONS

The ECGA and the BOA are competent GAs that use linkage learning to improve GA performance and computational efficiency. These algorithms, which are part of a class of GAs known as probabilistic model building GAs, build different types of models of the correlations among the genes in the best current solutions. The models are then used to generate new solutions with similar statistical properties to the best current solutions, enabling a more informed search than would be possible with the traditional operations of crossover and mutation in the SGA. Their performance in solving a groundwater remediation design problem reflects these improved capabilities. BOA solved the case study 30-2400% faster than the ECGA or the SGA, yet its solution quality degraded for the more complex problems by 0.4-1.7%. Most likely the solution quality degraded because of a need for larger population sizes when using probabilistic model-building GAs to ensure that the model used to generate new solutions is sufficiently representative. Future research is needed to optimize the parameter settings for these algorithms, which could improve performance. Overall, though, the results indicate that PMBGAs show substantial promise for improving computational performance with little or no degradation in solution quality.

REFERENCES

Aksoy, A., and Culver, T. B. (2000). "Effect of sorption assumptions on aquifer remediation designs." *Groundwater*, 38(2), 200-208.

Clement, T. P, Sun, Y., Hooker, B. S., and Petersen, J. N., (1998). "Modeling multi-species reactive transport in groundwater." *Ground Water Monitoring and Remediation*, 18(2), 79-92.

Clement, T. P., (1997). "RT3D - A modular computer code for simulating reactive multi-species transport in 3-Dimensional groundwater aquifers." Battelle Pacific Northwest National Laboratory Research Report, PNNL-SA-28967. http://bioprocesses.pnl.gov/rt3d.htm>.

Dandy, G. C., and Engelhardt, M. (2001). "Optimal Scheduling of Water Pipe Replacement Using Genetic Algorithms." *Journal of Water Resources Planning and Management*, 127(4), 214-223.

Goldberg, David E. (1989). *Genetic algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, New York, NY 1989.

Goldberg, David E. (1999). *Evolutionary design by computers* (Chapter 4. The Race, the Hurdle, and the Sweet Spot: Lessons from Genetic Algorithms for the Automation of Design Innovation and Creativity, pp. 105-118). San Mateo, CA: Morgan Kaufmann.

Gopalakrishnan, G., Minsker, B., and Goldberg D. E. (in press 2002) "Optimal Sampling in a Noisy Genetic Algorithm for Risk-Based Remediation Design." *Journal of Hydroinformatics*.

Harik, G., (1999). "Linkage Learning via Probabilistic Modeling in the ECGA." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Technical Report No. 99010, January.

McDonald, M.G., and Harbaugh, A.W. (1988). "A modular three-dimensional finite-difference ground-water flow model." Techniques of Water Resources Investigations 06-A1, United States Geological Survey.

Pelikan, M. (2000). "A C++ Implementation of the Bayesian Optimization Algorithm (BOA) with Decision Graphs." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Report No. 2000025, May.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (1999). "BOA: The Bayesian Optimization Algorithm" Joint meeting; 8th – July: Orlando, FL, International conference on genetic algorithms, GECCO-99, p.525-532.

Pelikan, M., and Goldberg, D. E., (2000). "Research on the Bayesian Optimization Algorithm." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Report No. 2000010, February.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (2000). "Bayesian Optimization Algorithm, Population Sizing, and Time to Convergence." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Report No. 2000001, January.

Pelikan, M., Goldberg, D. E., and Sastry, K. (2000). "Bayesian Optimization Algorithm, Decision Graphs, and Occam's Razor." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Report No. 2000020, May.

Reed, P., Minsker, B., and Goldberg, D. E. (2000). "Designing a Competent Single Genetic Algorithm for Search and Optimization." *Water Resources Research*, 36(12), 3757-3761.

Sastry, K., and Goldberg, D. E. (2000). "On Extended Compact Genetic Algorithm." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Report No., 2000026, April.

Sastry, K., and Xiao, G., (2001). "Cluster Optimization Using Extended Compact Genetic Algorithm." Illinois Genetic Algorithms Laboratory, Department of General Engineering. http://www-illigal.ge.uiuc.edu>. Technical Report No. 2001016, January.

Smalley, J. B., Minsker, B. S., and Goldberg, D. E. (2000). "Risk-based in situ bioremediation design using a noisy genetic algorithm." *Water Resources Research*, 36(20), 3043-3052.

Wang, M., and Zheng, C. (1997). "Optimal remediation policy selection under general conditions." *Groundwater*, 35(5) 757-764.

Wang, Q.J. (1991). "The genetic algorithm and its application to calibrating conceptual rainfall-runoff models." *Water Resources Research*, 27(9), 2467-2471.

Wardlaw, R., and Sharif, M. (1999)."Evaluation of Genetic Algorithms for Optimal Reservoir System Operation." *Journal of Water Resources Planning and Management*, 125(1), 25-33.