

# **A New Genetic Algorithm for Multi-objective Optimization in Water Resource Management**

V. Rao Vemuri and Walter Cedeño  
Department of Applied Science  
University of California, Davis and  
Lawrence Livermore National Laboratory  
7000 East Avenue, Livermore, CA 94550  
(vemuri1@llnl.gov and wcedeno@voicenet.com)

## **Abstract**

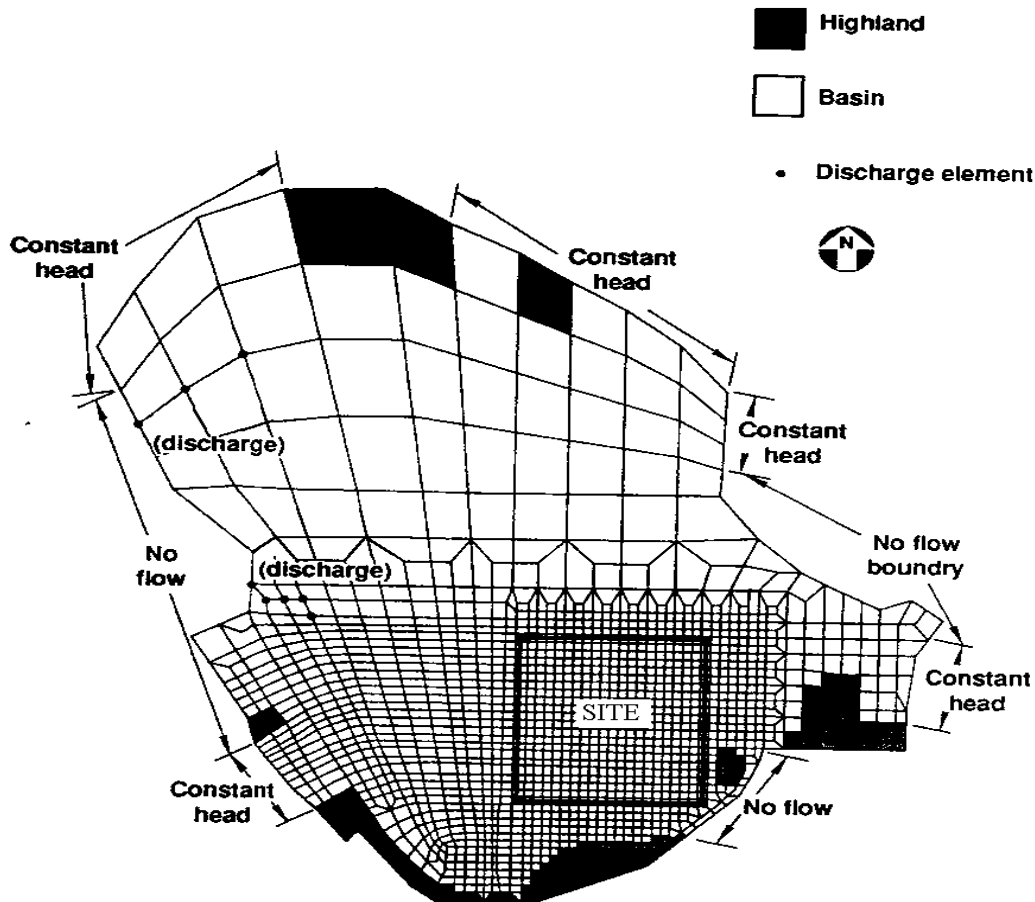
A genetic algorithm (GA) based on multi niche crowding (MNC) is capable of locating all the peaks of a multi-modal function. By associating these peaks with the utility accrued from different sets of decision variables, it is possible to extend the use of GAs to multi criteria decision making. This concept is applied to the remediation of a contaminated aquifer. The MNC GA is used to decide the optimal location of pumping wells. Aquifer dynamics are simulated by solving the partial differential equations describing the flow of water using SUTRA code. Output of this simulation constitutes the input to the GA.

## **1. Introduction**

In this paper we demonstrate the suitability of a new genetic algorithm (GA) to address decision making problems characterized by multiple objectives [Reid and Vemuri, 1971]. The crux of the method lies in identifying the objectives with the peaks of a function and solving for the locations and heights of these peaks with the new GA, called the multi-niche crowding genetic algorithm (MNC GA) [Cedeño and Vemuri, 1994]. Specifically, we determine the optimum placement of pumping (and recharge) wells and optimum pumping schedules during the remediation of contaminated groundwater aquifers in order to achieve a set of objectives [Rogers and Dowla, 1994; Horn and Nafpliotis, 1993].

## **2. The Simulation Environment**

The region studied is a hypothetical facility, polluted with hydrocarbons, occupying a one square mile area. The pollutants seep into the ground and contaminate the



**Figure 1:** SUTRA nodes and elements in mesh.

aquifer, located, say, 90 to 180 feet below the ground. The goal is the containment of the pollutant from seeping outside the boundaries of the facility. A useful strategy to contain the groundwater plume would be to pump the water, treat it, and then recharge the basin. In the simulation we assumed that three "pump, treat and recharge" facilities already exist on the western edge of the site and one on the eastern edge. Three more "pump, treat and recharge" facilities are planned. The task is to find optimum locations for these three new facilities.

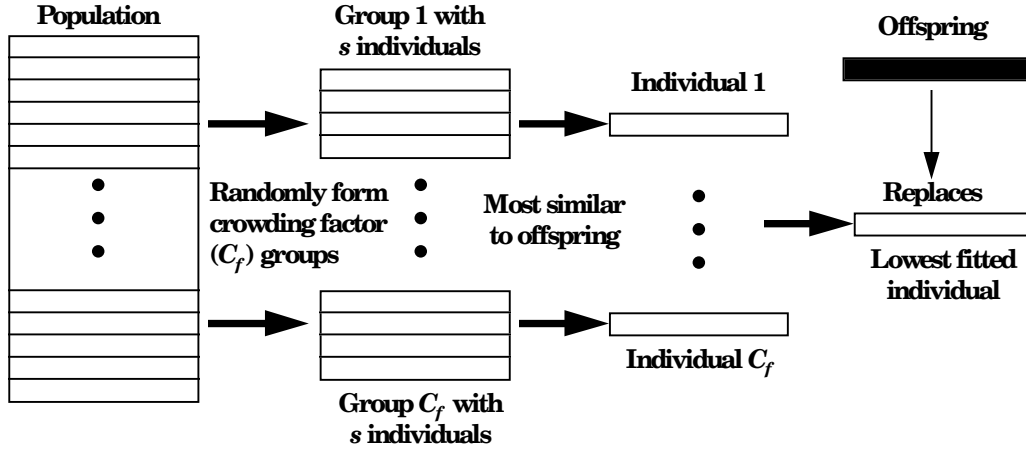
We used the flow and transport simulation code, SUTRA, of the U. S. Geological Survey to solve the partial differential equations (PDEs) describing the flow. SUTRA is a 2-D hybrid finite-element/ finite-difference model aimed at solving the governing PDEs for confined areal groundwater flow and areal solute transport. Boundary conditions suitable for the region are applied. A hydraulic conductivity of about 10 feet/day is assumed.

The mesh used has 2436 nodes and 2385 elements. It covers the entire areal extent of the aquifer. Figure 1 shows the mesh covering not only the facility but also the surrounding area. Out of this, we have chosen a subregion of size 20 x 21 nodes, covering an area of 5200 ft x 4950 ft. Each node in this subregion is a potential pump location. The distance between nodes is 260 feet, and there are 420 possible pumping locations. SUTRA took about 6.5 minutes of elapsed time per fitness evaluation. The output of SUTRA constitutes an input to the GA. The algorithm's goal is to find sets of well locations that best meets the objectives, subject to the constraint that no more than 10 wells are allowed.

### 3. MNC GA for Multi-modal Search

Any function of several variables exhibiting more than one peak is a *multi-modal function*. When a search technique, proven to be useful for uni-modal functions, is applied to multi-modal functions, the method tends to converge to an optimum in the local neighborhood of the first guess. There are many applications where the location of "k best extrema" are of interest. Then it is advisable to invoke both exploitation and exploration of the search space. The MNC GA is one such hybrid strategy. Experimental evidence indicates that the MNC GA has the ability to converge to multiple solutions. It achieves this result by encouraging competition between individuals within the same locally optimal group while limiting global competition. To achieve this goal, in MNC GA both the selection and replacement steps of the Simple GA (SGA) are modified with the introduction of some form of *crowding*.

In MNC GA the Fitness Proportionate Reproduction (FPR) step of SGA is replaced by *crowding selection* where each individual in the population has the same chance for mating in every generation. This takes place in two steps. First, an individual A is selected for mating. Second, its mate M is selected, not from the entire population, but from a group of individuals of size  $C_s$ , picked at random from the population. The mate M thus chosen must be the one who is the most "similar" to A. The similarity metric used here is not *genotypic* but a suitably defined *phenotypic distance metric*. That is, similarity was not measured using the encoded individuals but the solution represented by them. Crowding Selection promotes mating between individuals from the same niche while allowing some mating between individuals from different niches.



**Figure 2:** Schematic showing crowding factor groups created during replacement.

During the replacement step, MNC GA uses a policy called *worst among the most similar*. In a sense, this is a minmax strategy. Implementation of this policy follows these steps. First,  $C_f$  groups are created by randomly picking  $s$  individuals per group from the population. These groups are called *crowding factor groups*. Second, one individual from each group that is most similar to the offspring is identified. This gives  $C_f$  individuals that are candidates for replacement by virtue of their similarity to the offspring that will replace them. From this group of most similar individuals, we pick the one with the lowest (minimum) *fitness* to die and that slot is filled with the offspring (see Figure 2). The following pseudo-code summarizes the method. A generation is every  $N$  mating operations, where  $N$  is the population size:

1. Generate initial population of  $N$  individuals.
2. For gen = 1 to MAX\_GEN
3.   For i = 1 to  $N$
4.     Use *crowding selection* to find mate for individual i.
5.     Mate and mutate.
6.     Insert offspring in the population using *worst among most similar* replacement.

If we use FPR in Step 4 and replace the lowest fitness individual in the population (Step 6) with the newly generated offspring, this model corresponds to a steady-state GA.

#### 4. Interfacing with the Genetic Algorithm

A solution to the problem is represented by a variable length chromosome. Each chromosome is comprised of information on up to 10 wells, where the well locations appear in ascending order. The well locations are identified by node numbers of the

grid. The chromosome ( 41 107 224 635 972) denotes a five-well pumping scenario with the wells located at the nodes 41, 107, 224, 635, and 972. Only 420 of the total 2436 nodes in the mesh are valid well locations. This representation is equivalent to a binary string with a bit for each possible well location in the grid.

During the recombination, crossover (or, mating) and mutation are applied. The mating operator used here is equivalent to uniform crossover. A child is formed by first passing to it the genes that are common to both the parents and then selecting with a 0.5 probability the non common genes in both parents. Let individual I = (41 107 224 635 972) and its mate M = (22 107 635 700). The child resulting from mating I and M will have wells at nodes 107 and 635. The other well locations in the child will come from some subset of the remaining five non-common nodes, {22, 41, 224, 700, 972}. These well locations are selected with probability 0.5. A resulting offspring can be (22 107 635 972). Three mutation operators, namely, add well, delete well, and move well are applied independently with the same mutation probability. During an "add well" mutation a well from one of the 420 nodes is selected and added to the offspring while taking care to see that no duplicate nodes are allowed. The "delete well" and "add well" are similarly defined.

The replacement step in the MNC GA selects the individual in the crowding factor group with the lowest fitness. In a multiobjective problem a single fitness value can only be obtained if the objectives are grouped into a single utility function. This is not appropriate since a specific set of weight values drives the GA towards specific solutions. Here we determine the worst individual in the group by ranking the individuals for each objective and selecting the individual with the worst total ranking. This is a type of minmax strategy (see Figure 3, below). The individuals in the crowding factor group, indicated by A, B, C, and D, are ranked for each objective. Here a lower rank number indicates a better value for the objective being considered. Same rank numbers are assigned to individuals with equal or similar objective values. The total ranking for each individual is then determined by the sum of all the ranking numbers. The individual with the highest total rank, here B, is replaced by the offspring in the population. Using ranking as a way to determine the worst individual in a group allows good individuals for specific objectives to evolve as well as individuals that are average in two or more objectives.

Individual	Objective 1	Objective 2	Objective 3	Total Ranking
A	2	1	3	6
B	2	4	2	8
C	1	2	3	6
D	3	3	1	7

**Figure 3:** Use of ranking in Worst Among Most Similar Replacement.

Similarity between individuals is used during the selection and replacement. This allows the formation of species within the population and maintains diversity throughout the run. To calculate the distance between individual A = ( a1 a2 a3 a4)

and individual B = ( b1 b2 b3 b4 b5 b6) with 4 and 6 wells respectively, the following three steps are: (1) the squared difference in the number of wells between the individuals is calculated, namely,  $(4 - 6)^2 = 4$ . (2) the individual with the lower number of wells is selected, namely A. (3) For each well in A, the *closest* well in B is determined, say (a1, b4), (a2, b1), (a3, b6), and (a4, b3). *Similarity* between individuals A and B is now given by the sum of squared difference and the distance between all pairs of wells. That is,

$$\text{similarity} = \text{sqrDiff} + \text{dist}(a1, b4) + \text{dist}(a2, b1) + \text{dist}(a3, b6) + \text{dist}(a4, b3).$$

The smaller this number, the more similar the individuals are.

There is no explicit fitness function in this problem. Fitness of individuals is implicitly given by their objective values and their rank against others. Survival of an individual from generation to generation is determined by competition among small groups during replacement. Specifically, an individual will survive for many generations if its total ranking given by the objective values is lower than those in the crowding factor group.

The objectives used are: (1) minimization of remediation cost, (2) maximization of the contaminant removed, and (3) minimization of the concentration of contaminant leaving the site. Cost of pumping, estimated at \$70 per running foot, is calculated using the distance of each well to a hypothetical pump and treat facility. The goal is to find a set of wells whose cost stays close to an established budget of \$2.5M. The amount of contaminant removed is taken from the estimates obtained from SUTRA runs. This value is output in kilograms and is dependent on the well locations and the amount of water being pumped. In this implementation we used a fixed pumping rate of 0.0223 cubic feet per second. To monitor the attainment of the third objective, a set of wells are located on the west side of the site. The amount of contaminant in parts per billion (ppb) is estimated using SUTRA. We assumed that the total amount of contaminant reaching the monitor wells should be no more than 20 ppb. The values calculated for each objective are used by MNC GA only during the replacement step. Each objective value is used to determine the ranking of the individual for that particular objective. The sum of the rank values is then used to select the individual to be replaced by the offspring. An objective which ranks the individual based on the similarity to the individual can also be used to increase the diversity in the population.

## 5. Results

Results were obtained on a Sparc 2 workstation running Sun OS 4.1.3. Parameter values used for the MNC GA are:  $N = 25$ ; Crossover probability = 1.0; Mutation probability = 0.1;  $C_s = 3$ ;  $C_f = 3$ ;  $s = 3$ ; Number of generations = 20 or 25. In addition to restricting the potential pumping sites to a subregion of the total grid, small population sizes and a small number of generations were selected primarily to reduce

the computational burden. The crowding parameters were selected by trial and error. Crossover is always applied to mates. Initial population was generated at random. For each individual the number of wells in the chromosome was chosen at random. Well locations were selected at random from the 420 nodes in the 20 x 21 grid. The initial population never contained a solution meeting the 20.0 ppb regulatory limit.

At first, the MNC GA was implemented to solve a two objective optimization problem. The first objective is to restrict the amount of contaminant leaving the site to a presumed regulatory limit of 20 ppb. The second objective is to maximize the contaminant removed from the pumped water at the treatment facilities. Although it may appear at first sight that achieving the second objective automatically results in the attainment of the first objective, it is not always necessarily so. Contaminant transport in ground water basins is very slow. If the pumping wells are located at some distance from the site boundary, it is conceivable that the effects of pumping will reach a far off location (say, the boundary of the site) only after several years. This in turn would mean that the SUTRA simulation should be conducted for long durations of time. This is one aspect of the computational bottleneck that prompted several simplifying assumptions in the numerical simulations.

Table 1: Sample solutions with two objectives

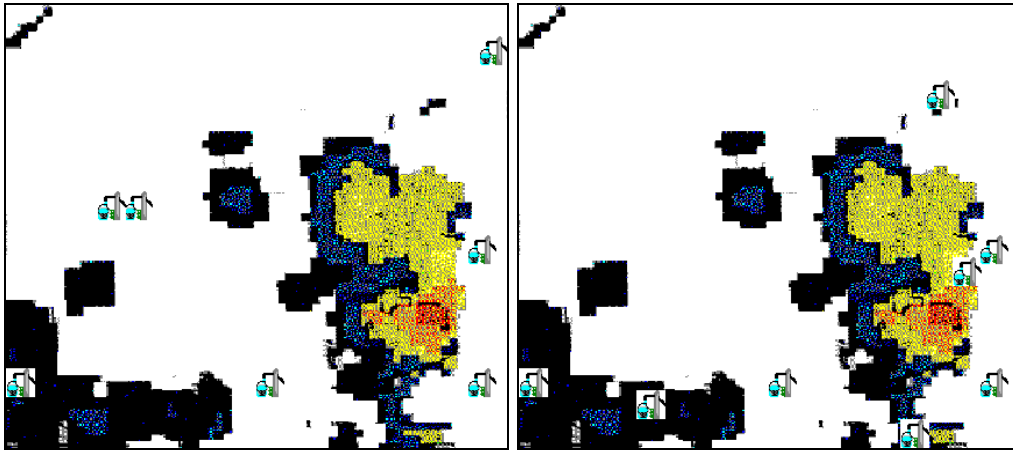
Gen.	No. of Wells	Contaminant leaving site (in ppb)	Contaminant Removed (in kg)
4	7	19.23	4.52
5	9	1.90	25.87
7	8	3.80	35.25
12	7	1.29	35.38
13	8	13.36	75.62
15	10	0.92	51.94
20	9	0.31	46.82

Table 1 shows a set of solutions for a scenario that considered only Objectives 2 and 3. Table 2 shows the results for a scenario that included the cost objective also. The cost objective is to minimize the difference between the cost of installation and an established budget. No major changes in the quality of solutions were observed from Scenario 1 to 2 except that the cost objective added some pressure towards smaller number of wells. At the same time the best solutions removed a greater amount of contaminant and had lower amounts of contaminant leaving the site. The last column, in Table 2, ranks the solutions according to their cost. Lower rank values indicate lower cost solution. As expected, improvements in one objective had a negative impact in other objectives during the initial generations.

Table 2: Sample solutions with three objectives

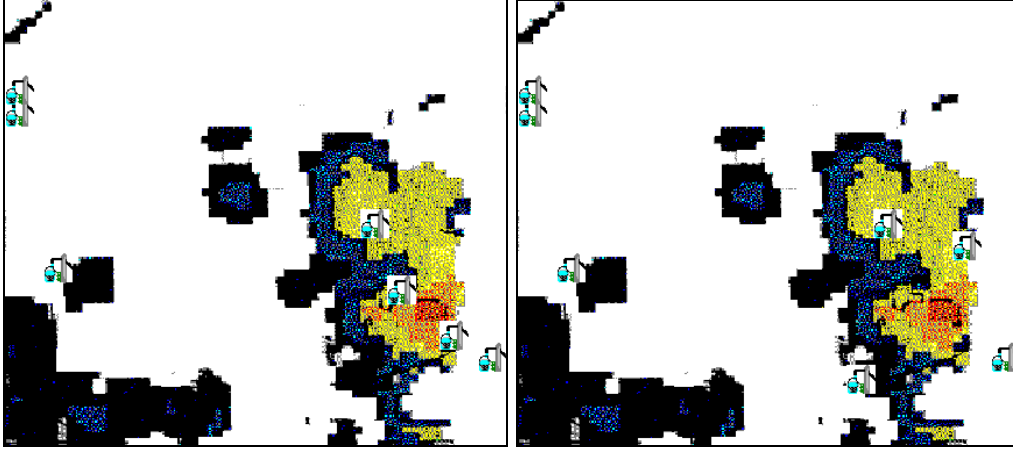
Gen.	No. of Wells	Contaminant leaving site (in ppb)	Contaminant Removed (in kg)	Cost Rank
10	7	6.30	36.71	6
10	6	7.79	39.62	5
18	7	11.15	43.37	4
19	7	13.27	59.02	3
20	7	2.44	64.68	1
21	7	1.21	81.98	2
23	9	6.59	98.89	7

Figure 4 shows the solution represented by the fourth and fifth generations in Table 1. These solutions have four wells in common although they have different objectives values. In general small changes in well locations may cause large changes in the objective values. The MNC GA alleviates this behavior by promoting recombination between similar solutions. In both solutions the wells are mostly in the southern part of the site where the pollutant concentration is greater.

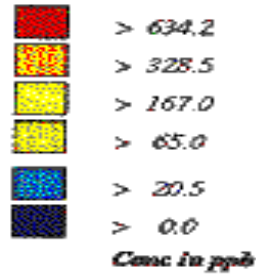


**Figure 4:** Well configurations for the fourth (left) and fifth (right) solutions in Table 1.

Figure 5 shows the placement of the wells for solutions shown on the fifth and sixth rows of Table 2. Both solutions have seven wells with five of them common between them. Both solutions have very low values for the amount of contaminant leaving the site. They also have high rankings in cost and contaminant removed. Prior to what we had seen using only two objectives, improvements in more than one objective was more common in later generations.



**Figure 5:** Well configurations for the fifth (left) and sixth (right) solutions in Table 2.



Legend for Figures 4 and 5.

## 6. Discussion and Conclusions

Several issues were addressed in this paper. First, the scope of this paper is limited to the issue of demonstrating the ability of the new GA, namely the multi-niche GA, in finding an acceptable solution to the multi-objective optimization problem arising in a practical context. Performance of MNC GA relative to alternate approaches is being investigated. Some preliminary results obtained when MNC GA is applied to artificially created, multi-modal test functions have already been reported [Cedeño and Vemuri, 1994].

This paper clearly demonstrated the suitability of MNC GA in meeting this restricted scope. It is shown that the MNC GA indeed has the ability to maintain different solutions satisfying multiple (and perhaps, conflicting objectives). The stratagem of replacing a single utility function to measure fitness by a rank score

assigned to various objectives appears to have allowed non-inferior solutions (i. e., solutions with favorable rankings in all objectives) to evolve. Given the limits imposed by us on the population size and the maximum number of generations in the evolutionary process, the quality of solutions obtained appear to be satisfactory. The policy of maintaining diversity throughout the search (enforced by MNC GA), clearly paid off in the form of better solutions in later generations.

Second, there are several issues that remain to be addressed. For example, could an equally acceptable or even a better result be obtained by some simple search strategy? Not likely, for two reasons. First, the problem is computationally complex. On a 20x21 grid there are 420 potential well positions. According to the formulation used in the paper, each pumping/recharge well configuration can contain anywhere from one to ten wells. This gives us

$$\sum_{i=1}^{10} \binom{420}{i}$$

combinations of well positions to explore. This is a huge number. Secondly, most of the classical optimization methods locate the peaks of a multi-modal function one at a time, in a sequential manner. The merit of the MNC GA is in finding all the peaks simultaneously in one sweep. Perhaps this one sweep takes more effort than  $n$  successive sweeps made by a classical method, where  $n$  is the number of peaks. This question can be addressed by making an estimate of the computational effort of both approaches using the "big O" notation.

What about the influence of the SUTRA simulation code on the performance of MNC GA? The SUTRA code only solves the partial differential equation simulating the flow of water and the resulting solution is used only in evaluating the fitness function used in the MNC GA. It has a major influence on the time taken to get a solution, but it should have no influence on the performance of the GA itself.

Finally, more research is also needed on the GA front. What are the benefits of bigger population size and more generations? The issue here practical. To adequately address this issue, we need another way of obtaining the values for the SUTRA dependent objectives to minimize the running time. We will be looking into other ways to calculate contaminant removed and regulatory limits that although may not be as precise will give us the ranking of the solutions for each objective. Additionally we can include heuristics in the mutation operators to add, move, or delete wells that will improve the overall ranking of the individual. Finally the chromosome can be extended to include other well parameters like pumping rate to evolve solutions with the complete pumping schedule. Adding well location constraints can easily be handle in the genetic operators also.

## Acknowledgments

The authors wish to acknowledge the assistance received from Virginia Johnson. She gave us copies of SUTRA code and helped us understand the intricacies of using it. Some comments made by Dr. Leah Rogers on the first draft of the manuscript were also helpful. Finally, the authors wish to thank one anonymous reviewer for identifying areas where some clarification was warranted. Work reported here is supported in part by a grant from the Institute of Scientific Computing Research of the Lawrence Livermore National Laboratory.

## References

### [Cedeño and Vemuri, 1994]

Cedeño, W. and V. Vemuri, Assembly of DNA restriction-fragments using genetic algorithms, *Evolutionary Computation*, 2:4, MIT Press. Winter 1994.

### [Horn and Nafpliotis, 1993]

Horn, J. and N. Nafpliotis, "Multi-objective optimization using the niched Pareto genetic algorithm," IlliGAL Report No. 93005, Department of Computer Science, University of Illinois, Urbana-Champaign, July 1993.

### [Rogers and Dowla, 1994]

Rogers, L. L., and F. U. Dowla, "Optimal groundwater remediation using neural networks and the genetic algorithm", *Water Resources Research*, 30:2, 458-481, 1994.

### [Reid and Vemuri, 1971]

Reid, R. W. and V. Vemuri, "On the Noninferior Index Approach to Large-Scale Multi-Criteria systems", *Journal of Franklin Institute* 291:241-254, 1971.