Empirical Analysis and Random Respectful Recombination of Crossover and Mutation in Genetic Algorithms

V.Kapoor
Institute of Engineering & Technology
Devi Ahilya University,
Indore, India.

S.Dey
Indian Institute of Management
Indore

A.P.Khurana
School of Computer Science
Devi Ahilya University,
Indore, India.

ABSTRACT
Genetic algorithms (GAs) are multi-dimensional, blind & heuristic search methods which involves complex interactions among parameters (such as population size, number of generations, various type of GA operators, operator probabilities, representation of decision variables etc.). Our belief is that GA is robust with respect to design changes. The question is whether the results obtained by GA depend upon the values given to these parameters is a matter of research interest. This paper studies the problem of how changes in the four GA parameters (population size, number of generations, crossover & mutation probabilities) have an effect on GA's performance from a practical stand point. To examine the robustness of GA to control parameters, we have tested two groups of parameters & the interaction inside the group (a) Crossover & mutation alone (b) Crossover combined with mutation . Based on calculations and simulation results it is seen that for simple problems mutation plays an momentous role. For complex problems crossover is the key search operator. Based on our study complementary crossover & mutation probabilities is a reliable approach.

Keywords: Genetic algorithm, control parameters, crossover, mutation, population sizing.

1. INTRODUCTION
Genetic algorithms (GAs) are multidimensional, blind & heuristic search methods which involves complex interactions among its parameters. GA imitates the biological principles of natural evolution and genetics to solve the complex optimization problems in various fields. Genetic algorithm was initiated by J H Holland, which shows that how evolutionary process is based on the Darwin concept of survival of fittest, is applied to the artificial systems. J H Holland studied the phenomenon of evolution in nature and initiated the mechanism of natural adoption in computer systems. A GA is iterative and parallel procedure that consists of a population of individuals (representing the solutions to the problems) which are improved in each generation by use of various operators such crossover & mutation generating new individuals (solutions) which are then tested by fitness function to know how better they are, and from which best ones are selected and worse ones are eliminated by means of selection operator. See Goldberg (1989).

A GA is designed to search for global optima but it cannot assure that the best solution will be found, sometimes solution converges to local rather than global optima. This problem can be avoided by making use of appropriate design parameters choices. GA parameters (such as population choice, choice of GA operators, operator probabilities etc) interacts in a complex way. Given a time frame to obtain a solution of problem, it is better to know that what GA parameter settings must be used to get good solution? Since overall time to run a GA is proportional to the number of function evaluations used. Thus any advance knowledge of interaction among GA parameters will lead to better global solution in lesser time & making GA more robust. It is seen that lack of robustness in the design choices always lead to local optima and lower levels of performance.

There exists a large number of studies which explore the interaction among different GA parameters for its successful application (see section 2 for references). In this paper we investigate how changes in design of GA parameters effect the GA performance from practical standpoint. Our investigation is that results obtained by GA are vigorous depending upon GA parameters choices. Vigorousness of GA with respect to design parameters cannot be confirm, as it is impossible to test all parameters combinations for all GA applications. Thus we have carried out a large number of experimental tests. Based on these tests we believe that our theory of vigorousness remains reasonable in context of functions being solved.

There are three conditions which are relevant in GAs design: (i) Encoding (ii) operators & (iii) Control Parameters. Our study of vigorousness & experimental observations is limited to the third topic i.e control parameters. For encoding we use binary coding. For operators we have chosen roulette wheel selection, one point crossover & bit level representation mutation. Since there are number of other selection schemes such as Bolzman selection, Tournament Selection, Rank Selection, Steady state selection etc. We do not have any reason to justify why we choose roulette wheel selection. We have chosen roulette wheel selection due to its simplicity and wide use of it in GA literature. Our interest is only centered on the analysis of control parameters: Crossover & mutation probabilities, number of generations & size of population, which are driving reproduction.

To examine the robustness of the GA to control parameters we test three groups of parameters & the interactions inside each group: (a) Crossover & mutation alone (b) Crossover combined with mutation (c) Population size & number of generations.

In our study we evaluate the changes in the performance of GA with respect to the changes in the control parameters. This paper is organized as follows Section 2 introduces with literature survey, Section 3 describes the basics of GAs, Section 4 describes the experimental details employed to carry out sensitivity analysis of control parameters, Section 5 reports the results obtained, analysis & observations, Conclusion & future work appear in Section 6 & Section 7.

2. LITERATURE SURVEY

Based on previous studies following salient observations are being made:

Analysis of various selection schemes used in modern GA such as Roulette wheel, rank, tournament & steady state is done by Goldberg & Deb (1991). Schemes are compared & verified according to convergence time and growth ratio. Harrik & Lobo (1999) proposed a parameter less Genetic Algorithm which is one step closer in the direction of making GA more robust. It is seen by BOYABATLI & SABUNCUOGLU (2007) that in case of dominant set of decision variable the crossover does not have a significant effect on the performance measures, where as high mutation rates are more suitable. The problem of finding optimal parameter is studied by many. Optimization of control parameters of GA is often time consuming. Cirirello & Smith (2000) approach of having meta level GA for control parameter optimization is a good approach. To study dynamics of these interactions more sophisticated stochastic models using Markov chain have also been developed and analyzed (Cao & Wu, 1999).

Eiben, Michalewicz, Schoenaver & Smith (1999) found that parameters values adjusted during evaluation gives better results than set in advance. This has potential of adjusting the algorithm to the problem while solving the problem. Crossover operator is largely dependent on the coding used to represent decision variables as shown by Radcliffe (1991). The success of Genetic algorithm depends on how well the crossover operator respects the underlying coding of the problem as shown by Kargupta Deb & Goldberg (1992). Culberson (1994) has shown that effect of crossover and mutation can be interchange by using a suitable coding transformation. It does not help in terms of deciding to which operator we should give importance. Crossover is useful in problem where preservation of building block is necessary. Mutation may destroy already obtained good information as illustrated by Goldberg, Deb & Clark 1992, Spears (1993). With this in mind it is suggested that GAs will work well with high crossover & low mutation probability (Goldberg, 1989). A crossover hill climbing algorithm is presented by Jones(1995) that illustrates the power of mechanics of crossover. Comparison between normal GA & a GA that uses random crossover has been made. Rana (1999) questioned the merits of crossover for genetic research. Exploratory power of crossover depends on the differences between its parents. As population converges its exploratory power diminishes. Recent work has extended the theoretical analysis of n-point and uniform crossover with respect to random sampling distributions as described by Spears & De Jong (1990). Spears (1995) describes an adapting mechanism for controlling the use of crossover in a Evolutionary algorithm. Spears (1994) describes an adaptive genetic algorithm that describes the optimal crossover probability as it runs.

Muhlenbein (1991) suggested that optimal mutation rate is proportional to the length of chromosomes. For deceptive functions an evolutionary algorithm with a good hill climbing strategy & reasonable mutation rate performs the best. Tate & Smith (1993) has shown that optimal mutation probability is dependent on the representation being used. Adaptive crossover and mutation probabilities helps in locating global optimum in a multimodal landscape (Srinivas & Patnaik, 1994). Hinterding, Gielewski & Peachey (1995) has shown that mutation can be an independent operator. Number of bits in variable representation effects the performance of mutation. Aqiirre & Tnaka (2002) proposed a model of GA that applies varying mutations parallel to crossover & background mutation by using extinctive selection to enhance the effectiveness of GA. This paper studies the problem of how changes in the four GA parameters (crossover & mutation probabilities) have an effect on GA's performance from a practical stand point.

3. BASIS OF GENETIC ALGORITHM

A GA is flexible optimization procedure stimulated by evolutionary concept & natural selection. Genetic algorithm is based on Darwinian evolutionary processes and naturally occurring genetic operations on chromosomes as illustrated by Koza. GA is highly parallel mathematical algorithm; however its implementation is serialized and is executed on serial machines most of the time. It transforms population of individuals (representing solutions to the problem coded into fixed length binary strings), each with an associated fitness value, into new population of individuals (i.e next generation) using operators modeled after Darwinian principle of reproduction and survival of fittest.

GA begins by selecting a random sample of potential solutions to the problem to be solved. Previously solutions to the problem have to be coded into fixed length binary string chromosomes notation. In second step fitness value of every string is calculated according to the objective function defined. In third step selection operator is applied to the initial set of potential solutions, in which individuals with higher fitness values are largely selected. In the forth step crossover & mutation operators are applied where binary bits of chromosomes are exchange and mutated to generate a new set of population (Solutions). Thus process of generation completes. After these iterative steps from second to forth take place until a fixed number of times or until population converges (What is called number of generations). There are two essential events in the GA process (i) Creation of new solutions or concept to solve the problem through crossover (recombination) and mutation. (ii) Elimination of bad solutions by selection operator. Apart from this there are some other things that have to be implemented i.e genetic representation of the solutions to the problem (often binary chromosomes codification) and calculation of objective function from which fitness function is derived. In case of operators crossover serves as an accelerator and is expected to propagate existing building block to next generation i.e its role is of exploitation. Mutation provides higher levels of disruption and exploration, but at the expense of preserving alleles. Mutation is expected to add random diversity in the population.

4. EXPERIMENTAL DETAILS

Number of functions evaluations (S) that is to be assigned for a application is product of number of generations (T) & population size (N) i.e $S = T \times N$. The minimum number of function evaluations assigned for an application depends on the function being solved. It is understanding that if function is difficult than number of function evaluations assigned must be increased. Following are the major difficulties that may be present in an arbitrary problem: 1. Multi Modality 2. Deception 3. Isolation 4. Collatoral noise. Here we have chosen one variable unimodal function, two variable function from De Jong test function bed, two variable unimodal function and a two variable four peaked function. Following test functions are chosen.

Test Functions

4.1 One variable Unimodal function

This function has only one optimum solution. We evaluate the function $f_1(x) = x / \text{Coeff}$. The actual value of coeff. is chosen to normalize x parameter when a bit string of length 30 bit is chosen. Thus Coeff.= $2^{30} - 1$ which is equal to 1073741823.0. Since the x value has been normalized, the maximum value of function $f_1(x)$ will be 1.0. $f_1(x)$ is one variable function and it is simplest of all. Variable x is represented as 30 bit binary string. Thus total search space is 2^{30} .

4.2 Two variable De Jong Function

Our second function is $f_2(x_1,x_2)$ is a two variable function from De Jong five function test bed. The function has a convex character tics. It is a two variable unimodal function.

$f_2(x_1,x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$

The search space is considered in the range of $-2 \leq x_1,x_2 \leq 2$. It has a single maximum point at $(-2, -2)$ with a function value equal to 3609. Variables x_1 & x_2 are represented as 10 bit binary strings. Hence total search space is 1024 × 1024.

4.3 Two variable Unimodal function

Our third function $f_3(x_1,x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$ is a two variable unimodal function (Himmelblau function) often used in optimization literature (Deb, 1995). The search space is of range $0 \leq x_1,x_2 \leq 6$ in which the above function has a single minimum point a (3,2), with a function value equal to zero. Here we converted minimization problem into maximization problem by using the formula:

$$f_3(x_1,x_2) = 1/(1+ f_3(x_1,x_2))$$

x_1,x_2 are represented as 10 bit binary strings & total binary search space is 1024 ×1024.

4.4 Four peaked function

This function is same as the previous one, but the ranges for x_1,x_2 are extended to $-6 \leq x_1,x_2 \leq 6$. The function has a total of four minima, one in each quadrant. All minima have function values equal to zero. In order to make one of them global minimum, we add a term to the above function.

$$f_4(x_1,x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 + 0.1(x_1 - 3)^2 (x_2 - 2)^2$$

x_1, x_2 are represented as 10 bit binary strings & total binary search space is 1024 ×1024.

In all the above test functions, the global optimal solutions is already known. It is seen that the out come of a GA depends on number of function evaluations (S). To compare different GAs reasonable amount of function evaluations (S) must be chosen. The success of GA is measured whether the global solution is found within pre-specified number of function evaluations (S). Outcome of results found by GA depends on the choice of S. To compare various GAs moderate amount of S must be chosen. We chose N = 100 such that GAs have moderate chance of finding a global optimum solution. Our choice of S = 5000 is well with our estimates. In order to reduce biases in the initial population, we simulate GAs from 50 different initial populations & for 100 generations. Four program in C + + were implemented for this study (one for each function) for a fixed number of generations (T) = 100.

5. EXPERIMENTAL RESULTS, ANALYSIS & OBSERVATIONS

Our research assess the functioning of the GA for different combinations of control parameters. Previous studies do not provide clear answer to this question of dependency of results obtained by GA, on value assigned to these parameters. Second the process to obtain value of these parameters can take longer than the process to built and run the GA for a specific problem.

We have carried a large number of experiments, but they are limited as number of possible test we can do to analyze GA functioning with respect to control parameters is infinitive. Analyzing the experimental data from large number of test we have carried out. It is seen that validity of expectations described in section 1 & section 3 is difficult to measure from these observed outcomes alone. Here we examine the effects of changes in GA parameters on various function described above.

5.1 Mutation alone

Mutation plays a secondary role in the operation of GA. Mutation adds random diversity in the population which helps in avoiding to get trap in local optima. Here each bit is selected probabilistically and then flipped (we are assuming a bit level representation). Mutation zaps a 0 to a 1 and vice versa. We do not change an individual with in the hyper plane if mutation rate is 0.0. If mutation rate is 1.0 we always change the individual & produce the complement of the individual. Mutation is used because it has a high explorative power. In summary we can control the amount of exploration that mutation performs by adjusting the mutation rate. Mutation is the function of one individual and is not affected by mutation of another. Traditional view is that mutation is destructive and must be used sparingly. Mutation is needed to add some potentially useful genetic material that is not represented in the population in each generation. It helps in recovering desirable genes that has been deleted or not included in the population during successive generations. It is a random walk through string space. Here we examine performance of GA with selection & mutation operators alone. Based on our experiments following investigations are made:

- Figure 1 & 2 shows the performance of GA with different mutation probabilities for function f_1 & f_3. It find near optimal solution in most of the cases, but performance degrades as mutation probability decreases. There is a significant increase in solution quality for mutation probability (P_m = 0.09) & above. This is due to the fact that high mutation rates sometimes helps in deletion of lowly fit schemata & introduction of new desirable schemata.
- It is seen that there is similarity in performance of GA for function f_2, f_3 & f_4 for mutation alone based GA. It is clear from the results that if a function is difficult to solve or is more complex, the effect of mutation alone based GA is deleterious. Figure 2 shows that mutation based GA did not find true optimum solution in any run for different parameter setting, though it reaches nearer to the optimal point due to the hill climbing nature of mutation based GA. Thus mutation based GA fails miserably for complex problems.

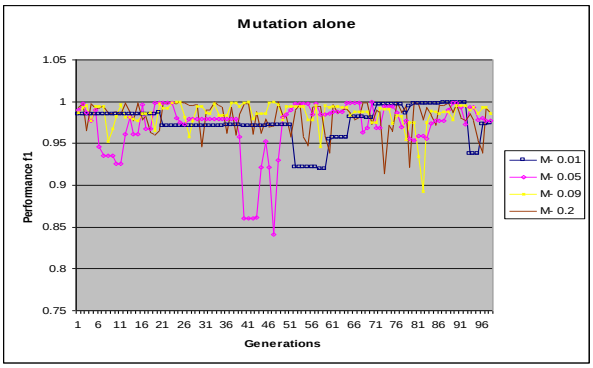

Figure 1. Performance of GA for various mutation probabilities for function f1.

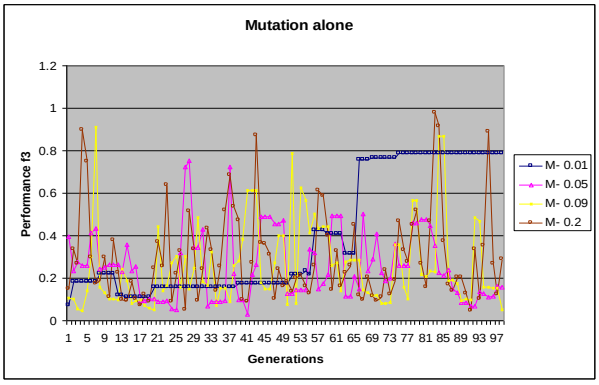

Figure 2. Performance of GA for various mutation probabilities for function f_3

5.2 Crossover alone

In GA literature crossover is considered to be a powerful operator. Role of crossover is of constructive in nature. It is important to know that how often a sample will be constructed & how it is disrupted. Crossover is a function of two individuals. Crossover maintains the existing state of alleles which are common to the individuals i.e its main role is of preservation. Preservation prevents exploration & it becomes more acute as the population looses diversity, since the number of common alleles will increase. Crossover is used where qualities of construction & survival are required for good performance. This happens when population is diverse and it consists of appropriate building blocks. From the experiment results it is seen that there is a similarity in the performance of GA for all the above four function in case of crossover alone based GA. Following investigations were made on seeing the results:

- Figure 3 & 4 shows the performance measure for various GA parameters for function f_1 & f_3. Crossover alone based GA with different probabilities (P_c) shows that $P_c = 0.75$ works better that other values. Increase in crossover probability increases the performance, but there is a significant decrease in performance at $P_c = 0.95$ for all functions tested.
- It is seen that as generation passes by, performance becomes stagnant. This is due to the fact that crossover looses its power as population looses its diversity or number of common alleles increases & it limits the type of exploration that crossover can perform.
- The performance of crossover (alone) based GA with selection operator finds close to optimal solution. It also performs better than mutation based GA for complex problems as shown in figure 3 & 4.

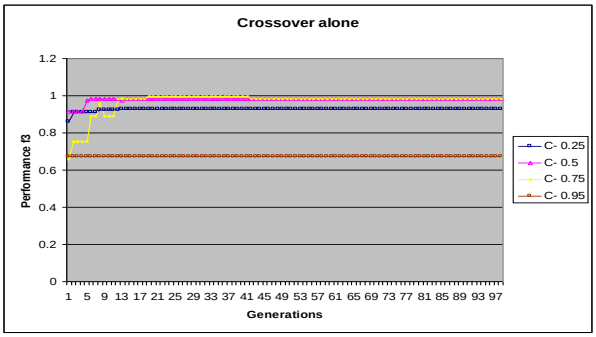

Figure 3. Performance of GA for various crossover probabilities for function f_1

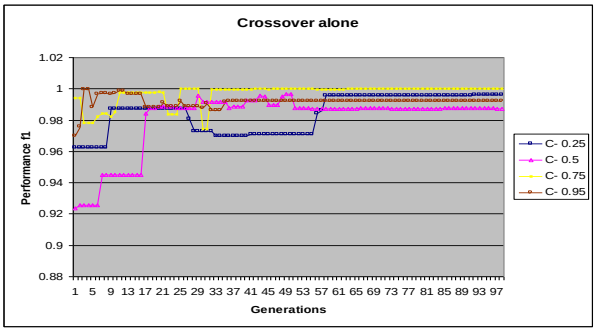

Figure 4. Performance of GA for various crossover probabilities for function f_3

5.3 Crossover & Mutation combined

It has been discussed earlier that role of mutation is to include diversity in the population thus preventing the population to get trapped in local optima. Role of crossover is to be construct & preserve good building blocks. It is of the view that both operators are necessary & performs different roles in the GA search. We had perform experiment with different parameters with these two operators, since one without the other will not be useful. Based on our results we had found similarity in the performance for all the above functions described in section IV. Following investigations are made:

- It is seen from figure 5 & 6 that GA with all three operators (Crossover, Mutation & Selection) performs the best. As compare to crossover alone and mutation alone based GA, it performs with high efficiency thus utilizing the total number of function evaluations allocated for all the functions.
- As shown in figure that high crossover rate ($P_c = 0.9$) combined with low mutation rate ($P_m = 0.01$) gives us the best results in terms of solution quality. Thus it allows us to pose hypothesis of complementary crossover & mutation probabilities i.e combining high crossover with low mutation rate.
- Is also seen from the figure that as mutation rate is increased to a value of $P_m = 0.1$, performance starts to resemble like a random noisy search, as mutation destroys already found good solution or building blocks in the population.

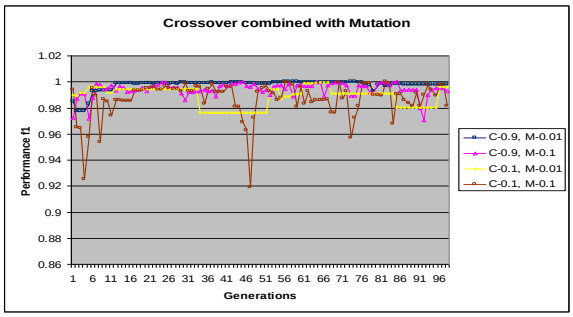

Figure 5. Performance of GA for various crossover & mutation probabilities for function f_1

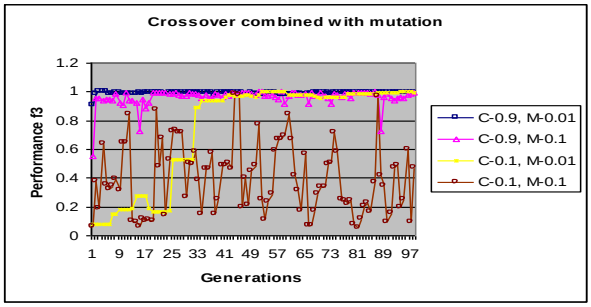

Figure 6. Performance of GA for various crossover & mutation probabilities for function f_3

- By examining the results of interaction of various population sizes with generations we cannot falsify our theory of robustness of GA to control parameters.

6. CONCLUSION

In this paper we have studied the effect of changes in GA parameters from a view of fixed function evaluations. Our aim is not to find the best parameter combination for a particular problem, but to come up with general conclusions. We have applied GA for four different functions (as discussed in section IV). We made following conclusions from our study:

- Results has shown that for simple problems mutation based approaches have performed better. As complexities of problem increases, tendency of GA to achieve global optima decrease. Mutation based GA failed miserably for complex problems.
- Crossover based GA has performed better than mutation based GA for all the complexities we had tested. But it has a tendency to get stuck at local optimum.
- It is concluded that high crossover rate combined with low mutation rate is a reliable approach. Based on this a simple tripartite GA (Roulette wheel selection, single point crossover & bit wise mutation operator) is recommended.
- Examining the experimental data from large number of test carried out, we have failed to falsify the theory of GA vigorousness to parameter implementation.

7. FUTURE WORK

Our research is limited as there are many other possible tests that can be carried out. Many questions that are open can be left for future research. Effect of different selection, recombination & mutation operators (such as ranking, tournament, two point crossover etc.) should be investigated from a view of solution quality. Analysis of other complex functions may be an interesting feature. It is seen that population size plays an important role for successful GA application. We strongly feel that more energy is needed to be spent in finding correct population sizing estimates. So that they limit the intensive use of computational recourses made by GA & also reduces time to find optimal solution.

8. REFERENCES

[1] Aguirre, H.E., Tanaka, K., 2002. Parallel varying mutation genetic algorithms. IEEE transactions.

[2] BOYABALTI, O., SABUNCUOGLU, I., 2007. Parameter Selection in Genetic Algorithms. System, Cybernatics & Informatics. Volume 2-Number 4, pp. 78-83.

[3] Cao, Y.J., Wu, Q.H., 1999. Optimization of control parameters in genetic algorithms: a stochastic approach. International journal of systems science, volume 30, number 2, pp. 551-559.

[4] Cicirello, V.A., Smith, S.F., 2000. Modelling GA Performance for Control Parameter Optimization. Proceedings of Genetic & Evolutionary Computing Conference. GEECO-2000.

[5] Culberson, J.C., 1994. Mutation-Crossover Isomorphisms and the construction of discriminating functions. Evolutionary Computation 2(3). 279-311.

[6] De Jong, K. A., Spears, W. M. 1990. An analysis of the interacting roles of population size and crossover in genetic algorithms. Proceedings of the International Conference on parallel problem solving from nature. Springer. Pp. 38-47.

[7] Eiben, A.E., Michalewich, Z., Schoenaur, M., Smith, J.E., 1999. parameter control in evolutionary algorithms. Proceedings of Genetic & Evolutionary Computing Conference.

[8] Goldberg, D.E., 1989a. Genetic algorithm in search, optimization & machine learning. New York: Addison Wisley.

[9] Goldberg, D.E., 1989b. Sizing populations for serial and parallel genetic algorithms. In: Schaffer, J.D. (Ed.), Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann, Los Altos, CA, pp. 70–79.

[10] Goldberg, D.E., Deb, K., 1991. A comparative analysis of selection schemes used in genetic algorithms. In: Rawlins, Gregory J.E. (Ed.), Foundations of Genetic

[11] Goldberg, D.E., Deb, K., 1991. A comparitive analysis of selection schemes used in genetic algorithms. In : Rawlins, Gregory J.E. (Ed.), Foundations of Genetic algorithms. Morgan Kaufmann Publishers, Inc., pp. 69–93.

[12] Harik, R.G., Lobo.F.G., 1999. A parameter-less genetic algorithm. IEEE transactions on evolutionary computation.

[13] Hinterding, R., Gielewski, H., Peachey, T.C., 1995. Proceedings of the 5th International Conference on Genetic Algorithms.

[14] Jones, T., 1995. Crossover, macromutation, and population-based search. Proceedings of 6th International conference on genetic algorithms.

[15] Kargupta, H., Deb, K., Goldberg, D.E., 1999. Ordering genetic algorithms and deception. Parallel problem solving from nature 2. pp. 47-53.

[16] Muhlenbein, H., 1992. How genetic algorithms really work I. Mutation and Hillclimbing. Foundation of genetic algorithms II pp. 15-25.

[17] Radcliffe, N., 2002. Forma analysis and random respectful recombination. Proceedings of 4th International conference on genetic algorithms.

[18] Rana, S., 1999. The distributional baises of crossover operators. Proceedings of Genetic & Evolutionary Computing Conference.

[19] Spears, W. M. & De Jong, K. A. 1991. An analysis of multi-point crossover. Proceedings of the Fourth International Conference on Genetic Algorithms, 230-236. La Jolla, CA: Morgan Kaufmann.

[20] Spears, W. M. & De Jong, K. A. 1991. On the virtues of uniform crossover. Proceedings of the Fourth International Conference on Genetic Algorithms, 230-236. La Jolla, CA: Morgan Kaufmann.

[21] Spears, W. M., 1995. Adapting crossover in evolutionary algorithms. Proceedings of the Fourth International Conference on Evolutionary programming.

[22] Spears, W. M., 1994. Adapting crossover in a genetic algorithm. Artificial intelligence center internal report # AIC-94-019.

[23] Srinivas, M. & Patnaik, L.M., 1994. Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE transactions on Systems, Man & Cybernatics, Vol. 24, No. 4.

[24] Tate, D.M., Smith, A.E., 1993. Expected allele coverage and role of mutation in genetic algorithms. Proceedings of the 5th International Conference on Genetic Algorithms. pp. 31- 37.