Raising Mutation Rate in the Context of Hybrid Genetic Algorithms

Yu-Min Son
School of Computer Science & Engineering
Seoul National University
599 Gwanak-ro, Gwanak-gu
Seoul, 151-744 Korea
eclipse726@snu.ac.kr

Byung-Ro Moon
School of Computer Science & Engineering
Seoul National University
599 Gwanak-ro, Gwanak-gu
Seoul, 151-744 Korea
moon@snu.ac.kr

ABSTRACT

A typical genetic algorithm uses a constant mutation rate or reduces the mutation rate over the generation. Generally, the degree of perturbation by crossover operators gets weaker as the generations go by. Hybrid GAs, which use a local optimization heuristic, strongly drive the offspring to a chromosome similar to or the same as one of the parents. We suspect that one needs to raise the degree of mutation in the late stages of a hybrid GA, contrary to the practice. The experimental results supported our suspection, by showing performance improvement over two philosophically representative mutations: the traditional fixed-rate mutation and the non-uniform mutation. We used two representative NP-hard problems in the experiments: the graph bisection problem and the traveling salesman problem.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Global optimization; G.2.2 [**Graph Theory**]: Graph algorithms

General Terms

Algorithms, Experimentation

Keywords

Hybrid genetic algorithm, mutation rate, perturbation, convergence of population, traveling salesman problem, graph bisection problem

1. INTRODUCTION

In Genetic Algorithms (GAs), mutation is one of the representative operators and numerous studies have attempted to find and explore the desirable mutation rates. All these studies, however, were for classical GAs that do not combine a local search algorithm. Without a local search algorithm, it is desirable not to perturb high-quality solutions, which mostly appear in the late stages of a GA, for fine-tuning around local optima.

On the Contrary, when it comes to hybrid GAs, we suspect that higher mutation rates are appropriate in the late stages. In a traditional GA, keeping the characteristics of high-quality parents is crucial. In a hybrid GA, however, a weak perturbation makes offspring return to one of the parents with a high probability, by the power of local search engines. Thus a hybrid GA seems to need a higher degree of perturbation not to be overwhelmed by the power of local optimization.

We suggest a new direction that increases the mutation rate in accordance with the population convergence in the context of hybrid GAs. We took experiments on two NP-hard problems: the traveling salesman problem and the graph bisection problem. Overall, the new mutation strategies showed improvement over the fixed-rate mutation and the non-uniform mutation within 5% statistical risk. Note that the fixed-rate mutation and the non-uniform mutation are two philosophically representative mutations in the community of GA.

GECCO'11, July 12–16, 2011, Dublin, Ireland.
ACM 978-1-4503-0690-4/11/07.

2. MUTATION STRATEGIES FOR HYBRID GENETIC ALGORITHMS

We compare three mutation strategies: traditional fixed-rate mutation, non-uniform mutation, and rate-increasing mutation. The former two are two representative mutations in philosophy in the GA community. The rate-increasing mutation is the new one that we propose.

Fixed-rate Mutation

This uses a fixed mutation rate all through the generations. We apply the rate 0.005 to this type of mutation.

Non-Uniform Mutation

This decreases the degree of mutation rate over the generations, which was suggested by Michalewicz [1]. So we use the formula and give new meaning to some parameters to be better suited to TSP and graph bisection problem as follows:

$$y = y_0 \cdot (1 - r_2^{(1-t/T)^b})$$

where y is the current mutation rate, y_0 is the initial mutation rate, r_2 is a random number on the range [0,1), t and T are the current generation and the maximum generation number, respectively, and b is a parameter to control the dependence on the time.

Rate-Increasing Mutation

This increases the mutation rate over the generations. The rate increases in proportion to the convergence rate. When

the population converges over a threshold, say 50%, we start to raise the mutation rate. We keep changing mutation rates unless the convergence rate falls down to 50% or less. When the convergence rate drops below 50%, we set the mutation rate to the initial rate. We used two versions as in the following.

- **linear-increasing mutation**

$$r = (1 + k) \cdot r_0, \quad k = max\{0, \lfloor 1 + \frac{c - c_0}{10} \rfloor\} \qquad (1)$$

where r is the current mutation rate, r_0 is the initial mutation rate, c is the current convergence rate, and c_0 is the threshold.

- **exponential-increasing mutation**

$$r = 2^k \cdot r_0, \quad k = max\{0, \lfloor 1 + \frac{c - c_0}{10} \rfloor\} \qquad (2)$$

3. EXPERIMENTAL RESULTS

Table 1 shows the experimental results of TSP instances. The column "Strategy," "Fixed," "Non-Uni," "Lin-Inc," and "Exp-Inc" mean fixed-rate mutation, non-uniform mutation, linear-increase mutation, and exponential-increase mutation, respectively.

The experimental results of Table 1 show that the suggested mutations were overall better than fixed-rate mutation and non-uniform mutation. Non-uniform mutation showed the worst in the context of hybrid GAs. The suggested mutations showed improvement for all the instances when combined with 5-pt, uniform, and distance preserving crossover. But when combined with cycle crossover we could observe little improvement in statistical sense with 5% risk. Although we used 5% risk for comparison, most cases satisfied superiority with 1% risk for both problems.

The improvement was consistent in the experiments with the graph bisection problem, as shown in Tables 2. The suggested mutations overall showed improvement over the fixed-rate (traditional) mutation and non-uniform mutation.

We should also note that the GAs found superior best solutions with the suggested mutation, than with the fixed-rate mutation and the non-uniform mutation.

4. CONCLUSIONS

In this paper, we suggested rate-increasing mutations and investigated the effects in the context of hybrid GAs. We suspect that one does not have to worry about somewhat strong mutation in the late stages of hybrid GAs, while the traditional GAs maintain or decrease the mutation rates not to destroy attractive characteristics of high-quality solutions. Local optimization heuristics, in hybrid GAs, may recover most perturbed characteristics and even find new ones by their moderately wide space search capability. The experimental results backed up our suspection and showed that rate-increasing mutation outperforms two philosophically representative mutations.

5. ACKNOWLEDGMENTS

This work was supported by the Engineering Research Center of Excellence Program (Grant 2011-0000966) and Mid-career Researcher Program (Grant 2010-0014218) of Korea Ministry of Education, Science and Technology(MEST) / National Research Foundation of Korea(NRF). The ICT at Seoul National University provided research facilities for this study.

Table 1: TSP Test Results for the Instance PCB3038(137694)

Strategy	Crossover	Best	Average*	$\sigma/\sqrt{n}$	Time†
Fixed	5-pt	137694	137778.95	4.99	257.81
	Uniform	137698	137757.23	4.51	1720.73
	Natural	137694	137754.00	4.53	613.15
	Cycle	137698	137735.08	3.23	600.43
	DPX	1376988	137744.23	4.17	694.32
	EAX	137698	137808.48	6.59	159.97
Non-Uni	5-pt	137694	137779.24	5.11	223.39
	Uniform	137698	137772.43	4.96	342.98
	Natural	137694	137768.62	5.37	634.89
	Cycle	137698	137779.59	5.86	621.11
	DPX	137698	137760.54	4.18	704.25
	EAX	137694	137804.64	6.81	280.43
Lin-Inc	5-pt	137694	**137726.34**	5.79	466.58
	Uniform	137698	**137726.08**	6.67	1866.99
	Natural	137698	**137736.52**	5.51	1102.15
	Cycle	137698	**137723.47**	4.21	899.91
	DPX	137701	**137734.07**	4.16	1235.67
	EAX	137694	137800.36	6.02	489.81
Exp-Inc	5-pt	137694	**137722.76**	4.66	505.88
	Uniform	137698	**137741.18**	6.51	1992.67
	Natural	137698	**137731.92**	4.02	1335.14
	Cycle	137698	137734.34	3.95	967.68
	DPX	137698	**137720.10**	4.23	1447.69
	EAX	137698	137799.34	6.42	496.30

∗ Over 100 runs.
† CPU seconds on Pentium PC 2.66GHz.

Table 2: GB Test Results for the Instance U5000.10

Strategy	Crossover	Best	Average*	$\sigma/\sqrt{n}$	Time†
Fixed	5-pt	85	126.73	0.988	22.13
	Uniform	85	127.36	1.076	23.42
	Natural	85	126.95	0.988	22.32
	Cycle	85	126.04	0.996	22.02
Non-Uni	5-pt	89	142.00	0.982	19.37
	Uniform	89	141.17	1.050	20.20
	Natural	89	140.88	0.988	20.04
	Cycle	89	139.12	1.072	19.90
Lin-Inc	5-pt	79	**112.52**	0.996	24.09
	Uniform	79	**113.32**	0.996	24.99
	Natural	79	**112.44**	1.072	24.31
	Cycle	79	**111.92**	1.054	24.13
Exp-Inc	5-pt	79	**114.23**	1.070	24.65
	Uniform	79	**115.33**	0.988	26.99
	Natural	79	**113.10**	1.072	24.98
	Cycle	79	**113.27**	0.996	24.58

∗ Over 1000 runs.
† CPU seconds on Pentium PC 2.66GHz.

6. REFERENCES

[1] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolutionary Programs.* Berlin, Germany: Springer-Verlag, 1996.