Comparison of GAs in Black-Box Scenarios*

Use-Case Specific Analysis†

Cheng-Hao Yang
Taiwan Evolutionary Intelligence Laboratory
Department of Electrical Engineering
National Taiwan University
b03901018@ntu.edu.tw

Hou-Teng Cheong
Taiwan Evolutionary Intelligence Laboratory
Department of Computer Science and Information Engineering
National Taiwan University
r07922143@ntu.edu.tw

Tian-Li Yu
Taiwan Evolutionary Intelligence Laboratory
Department of Electrical Engineering
National Taiwan University
tianliyu@ntu.edu.tw

ABSTRACT

In black-box optimization scenarios, researchers have no control over the fitness functions, and hence genetic algorithms (GAs) are usually compared by the number of function evaluations. Commonly used statistics include arithmetic mean, median, and standard deviation. However, these statistics can be misleading. For example, when there exist unsolvable instances within limited time, median simply ignores those instances, and arithmetic mean is not applicable at all. In this paper, we propose comparison methods from a practical point of view. Specifically, we propose three use cases which cover most of the situations that GA practitioners may encounter. Among these three use cases, two of them are matchups, which requires a pair of GAs to be compared with each other, while the other provides as a standalone performance indicator of GAs.

CCS CONCEPTS

• **Computing methodologies → Genetic algorithms**;

KEYWORDS

Genetic algorithms, Performance measures

ACM Reference Format:
Cheng-Hao Yang, Hou-Teng Cheong, and Tian-Li Yu. 2019. Comparison of GAs in Black-Box Scenarios: Use-Case Specific Analysis. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3321963

1 INTRODUCTION

Genetic algorithms (GA) [3, 6] are one of the most important optimization techniques since many real-world problems fit into the black-box optimization scenarios. Evaluating the performance of GAs has always been crucial when GA researchers develop a new algorithm. Since users of GAs usually have no control over the black-box function, the evaluation time of the function is considered long. As a result, the number of function evaluation (NFE) is the most commonly used performance metric when comparing different GAs [2, 4, 7]. However, NFE statistics, such as its average, median and standard deviation (stdev), cannot fully represent the performance of a GA. When GAs fail in solving some instances, GA researchers usually consider the NFE required on failed instances as a large value or infinity. In such cases, the NFE value of unsolved instances becomes an obstacle in creating an indicative metric. A formulated, fair comparison based on NFE between parameterized algorithms is proposed in [1]. However, the work does not solve the problem unsolved instances create. Also, it does not cover parameterless GAs, whose usages are more common in real-world applications.

In this regard, use cases of GAs need to be considered in comparing the performance of GAs. We conclude different scenarios users may encounter and compare GAs based on them. Each of the use cases focuses a certain aspect of GAs and should be chosen to best fit user needs.

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321963

2 PROPOSED USE CASES

2.1 Win-rate

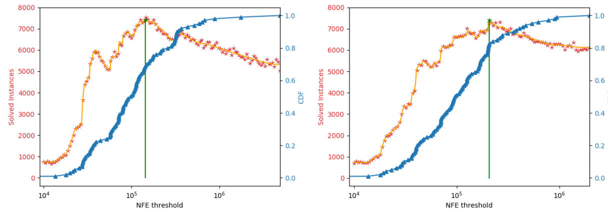
In scenarios that users have plenty of computational resources and are required to obtain the optimal solutions of tasks, users can run two GAs to be compared simultaneously. In this case, users compare GAs based on the proportion of tasks solved most quickly by them respectively. Based off these scenarios, the Win-rate use case is designed. Through experiments, we find that comparing P3 [4] and Fast-Efficient-P3 [5] on solving 100 instances of MAX-SAT in the Win-rate use case gives an opposite result to the result given by averages of NFE shown in Table 1.

NFE Statistics	P3	Fast-Efficient-P3
Average NFE	176k	158k
Stdev NFE	284k	181k
Number of Solved Tasks	55	45

Table 1: Comparing P3 and Fast-Efficient-P3 on solving MAX-SAT problem in the Win-rate use case

2.2 Online Algorithm

When computational resources ($totalNFE$) are limited and users are required to obtain the optimal solutions of tasks, users sometimes



(a) P3 (b) Fast-Efficient-P3

Figure 1: CDF and Expected Solved Instances of P3 and Fast-Efficient-3

need to give up some difficult tasks and focus on others. A reasonable strategy is to decide a give-up threshold of NFE to optimize the number of solved tasks.

With the CDF model, we can find the optimal threshold θ^* of NFE that maximizes the expected number of instances using one NFE solves, denoted by *ind*:

$$\theta^* = \arg\max_{\theta} ind = \arg\max_{\theta} \frac{F(\theta)}{(1 - F(\theta)) \times \theta + \int_0^{\theta} x f(x) \mathrm{d}x} \quad (1)$$

By definition, the expected number of solved tasks equals to $ind \times totalNFE$.

Through experiments, we find that comparing P3 and Fast-Efficient-P3 on solving 100 instances of MAX-SAT based on the Online Algorithm use case shows the opposite result to that given by comparing with average NFE.

For both algorithms, the CDF models are shown in Figure 1a and Figure 1b as the blue triangular curves. The orange star curves show the expected numbers of problems solved in a fixed number of NFE using the threshold on the x-axis. The optimal threshold and the re-simulated number of solved tasks are marked as the green vertical line.

NFE Statistics	P3	Fast-Efficient-P3
Average NFE	176k	158k
Stdev NFE	284k	181k
Optimal Threshold θ^*	143k	209k
Expected number of solved tasks ($1/ind$)	7428	7392

Table 2: Comparing P3 and Fast-Efficient-P3 on solving MAX-SAT problem in the Online Algorithm use case, where total NFE limit is 10^9

2.3 Deadline & Baseline

In scenarios that users have a deadline, which is represented by a limited number of NFE, the optimal solution may not be necessary and only a tolerable solution is required. In such cases, the probability of reaching the baseline before the deadline is all that users care about. We generate loose, moderate, and strict requirements based on the averages and stdevs of NFE required to solve it with the two GAs. Based off the three criteria, the Deadline & Baseline use case is designed.

In practice, we consider a situation where the two versions of DSMGA-II, DSMGA-II and DSMGA-II-TwoEdge [2, 7], are solving 100 instances NK landscape problem [8] with step size 3. In the three different deadline and baseline settings, two of them gives an inverse result with respect to the average and stdev indicators.

Since the deadlines and baselines are determined by each instance, we show only the NFE and fitness function distributions of the first instance of NK landscape problem with step size 3 as below.

Prob of reaching baseline	DSMGA-II Two Edge	DSMGA-II
Average NFE	556k	548k
Stdev NFE	250k	242k
Loose Requirement	89.15%	89.51%
Moderate Requirement	78.89%	76.64%
Strict Requirement	57.49%	50.77%

Table 3: Comparing DSMGA-II and DSMGA-II-TwoEdge on solving NK-3, and DSMGA-II-TwoEdge is preferred in two of the three requirements

We rerun all 100 instances, each with the three requirements, and in Table 3, DSMGA-II-TwoEdge is preferred in Moderate and Strict requirements.

3 CONCLUSION

In this work, we proposed new frameworks for comparing GAs based on different settings to mitigate the bias of directly using NFE statistics for comparison. This gives a more meaningful result in the practical sense.

When choosing the most suitable GAs, users should consider their real use cases and find the best NFE distributions to fit their needs instead of choosing GAs with best NFE statistics. To the extent of our knowledge, this is the first ever work that formally addresses the distributions of NFEs and discusses different aspects of them under different scenarios. Three use cases were considered and simulated in this paper: Win-rate, Online Algorithm, and Deadline & Baseline.

REFERENCES

[1] Chia-Sheng Chen, Hung-Wei Hsu, and Tian-Li Yu. 2018. Fast Algorithm for Fair Comparison of Genetic Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '18)*. ACM, New York, NY, USA, 913–920. https://doi.org/10.1145/3205455.3205511

[2] Ping-Lin Chen, Chun-Jen Peng, Chang-Yi Lu, and Tian-Li Yu. 2017. Two-edge Graphical Linkage Model for DSMGA-II. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17)*. ACM, New York, NY, USA, 745–752. https://doi.org/10.1145/3071178.3071236

[3] David E. Goldberg and John H. Holland. 1988. Genetic Algorithms and Machine Learning. *Machine Learning* 3, 2 (01 Oct 1988), 95–99. https://doi.org/10.1023/A:1022602019183

[4] Brian W. Goldman and William F. Punch. 2014. Parameter-less Population Pyramid. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. ACM, New York, NY, USA, 785–792. https://doi.org/10.1145/2576768.2598350

[5] B. W. Goldman and W. F. Punch. 2015. Fast and Efficient Black Box Optimization Using the Parameter-less Population Pyramid. *Evol. Comput.* 23, 3 (Sept. 2015), 451–479. https://doi.org/10.1162/EVCO_a_00148

[6] John H. Holland. 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence.* University of Michigan Press, Ann Arbor, MI.

[7] Shih-Huan Hsu and Tian-Li Yu. 2018. Optimization by Pairwise Linkage Detection, Incremental Linkage Set, and Restricted / Back Mixing: DSMGA-II. *CoRR* abs/1807.11669 (2018). arXiv:1807.11669 http://arxiv.org/abs/1807.11669

[8] Krzysztof L. Sadowski, Peter A.N. Bosman, and Dirk Thierens. 2013. On the usefulness of linkage processing for solving MAX-SAT. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*, 853–860. https://doi.org/10.1145/2463372.2463474