Standard Steady State Genetic Algorithms Can Hillclimb Faster than Evolutionary Algorithms using Standard Bit Mutation

Dogan Corus
Department of Computer Science
University of Sheffield
Sheffield
d.corus@sheffield.ac.uk

Pietro S. Oliveto
Department of Computer Science
University of Sheffield
Sheffield
p.oliveto@sheffield.ac.uk

ABSTRACT

Genetic Algorithms (GAs) use principles of natural selection to evolve a population of candidate solutions obtained by the recombination of individuals of the current generation. Albeit their huge popularity, providing natural examples where standard GAs provably outperform more traditional mutation-based heuristics has turned out to be a tedious task. In this paper we rigorously prove that a standard steady state (μ+1) GA outperforms any evolutionary algorithm, that relies only on standard bit mutation (SBM) as variation operator, for hillclimbing the classical OneMax benchmark function. In particular, we show that the GA is 25% faster by providing an upper bound of $(3/4)en \ln n$ on its expected runtime versus the $en \ln n$ expected function evaluations required by any algorithm using only SBM. To achieve the result, we devise a mathematical framework which extends the classical artificial fitness levels method by coupling each level with a Markov chain. This Markov chain allows to bound the improvement probabilities of the current population based on its diversity. In turn it can be appreciated how larger populations sustain diversity for a longer time, effectively giving crossover more chances of finding improved solutions. Since diversity is created via mutation, higher rates than the standard 1/n mutation rate, lead to better upper bounds on the expected runtime. This paper summarises the work that appeared in [1][1].

CCS CONCEPTS

• **Theory of computation → Theory of randomized search heuristics;** *Optimization with randomized search heuristics*;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference Format:
Dogan Corus and Pietro S. Oliveto. 2018. Standard Steady State Genetic Algorithms Can Hillclimb Faster than Evolutionary Algorithms using Standard Bit Mutation. In *Proceedings of the Genetic and Evolutionary Computation Conference 2018 (GECCO '18),* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 2 pages. https://doi.org/10.1145/3205651.3208214

[1] The research leading to these results has received funding from the EPSRC under grant no. EP/M004252/1.

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07…$15.00
https://doi.org/10.1145/3205651.3208214

Algorithm 1: (μ+1) GA

$P \leftarrow \mu$ individuals, uniformly at random from $\{0,1\}^n$;
repeat
  Select $x, y \in P$ uniformly at random;
  $z \leftarrow$ Uniform crossover with probability $1/2$ (x, y);
  Flip each bit in z with probability c/n;
  $P \leftarrow P \cup \{z\}$;
  Choose one element from P with lowest fitness and remove it from P, breaking ties uniformly at random;
until *termination condition satisfied*;

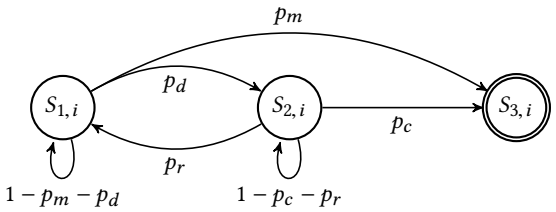
1 INTRODUCTION

Genetic algorithms (GAs) recombine promising solutions to generate new solutions. This distinguishes them from mutation based evolutionary algorithms which concentrate their search in the immediate vicinity of candidate solutions. The fact that standard GAs themselves use the mutation operator to avoid losing diversity in the population poses one of the longest lasting open problems in evolutionary computation: "To what extent can the optimisation capabilities of GAs be attributed to the crossover operator?". To this day a rigorous result which shows that a standard genetic algorithm outperforms a standard bit mutation-only algorithm on a natural benchmark problem is lacking.

The most prominent results addressing this problem rely on one of two leeways; either additional mechanisms are introduced to standard GAs to simplify the analysis, or an artificial function designed to highlight the capabilities of the crossover operator is analysed. Several such examples are reported in the original paper [1]. By adding a diversity preserving mechanism to the steady-state (μ+1) GA, an important recent work showed that its expected runtime on the classical OneMax function is half of the (1 + 1) EA's runtime, which is the fastest standard bit mutation only evolutionary algorithm for OneMax [4]. Examplifying the second approach, Dang et al. showed that the (μ+1) GA without any modification optimises the artificially designed Jump function at least a linear factor faster than the (1 + 1) EA [2].

In this paper we finally prove that the standard steady-state (μ+1) GA (see Alg. 1) in its simplest form can optimise the fundamental benchmark function OneMax in 25% less function evaluations (runtime) in expectation than any standard bit mutation based algorithm.

Figure 1: Markov Chain for fitness level i.



2 MAIN RESULT

Since no SBM based algorithm can optimise OneMax in less than $en \ln n$ expected function evaluations [4], the following theorem establishes the main result when a mutation rate of $1/n$ is used by the GA.

Theorem 2.1. *The expected runtime of the $(\mu+1)$ GA with mutation rate c/n for any constant c, $\mu \geq 3$ and $\mu = o(\log n / \log\log n)$ on* OneMax *is:* $E[T] \leq \frac{3(1+o(1))}{c(3+c)} e^c n \ln n$.

The mathematical framework devised for the analysis and leading to our main result also highlights: (1) why the GA is faster; (2) that large populations make the GA more efficient; (3) that higher mutation rates than the standard $1/n$ are beneficial to the GA.

2.1 Mathematical Framework

The primary shortcoming of the classical artificial fitness level (AFL) method [3] is that the improvement probability at a fitness level is determined according to the worst-case configuration of the population. Crossover, on the other hand, requires a diverse population to be effective implying that its contribution cannot be captured by the standard AFL method.

We overcome this obstacle by modeling the population at each level i with a Markov chain (MC_i) (see Fig. 1). Each MC_i has two transient states representing populations with and without diversity. Each state has a different transition probability to an absorbing state where absorption denotes the first improvement to the next level. The framework reflects that crossover improves with high probability $p_c = \Theta(1)$ when diversity is present in the population. The improvement probability changes to $p_m = \Theta(i/n)$ when the population has no diversity, hence the GA behaves exactly like an SBM-only algorithm since crossover is ineffective.

2.2 Larger population sizes are beneficial

Diversity is more likely to be lost (i.e., p_r is greater) for $\mu = 2$ since each genotype can takeover the population in a single generation which is not the case for larger μ. Thus, crossover has more chances to be effective when $\mu \geq 3$ (we get a worse upper bound for $\mu = 2$). Fig. 2 provides experimental evidence that $\mu \geq 2$ performs better.

2.3 Higher mutation rates are beneficial

The mathematical framework highlights that diversity is necessary for the crossover to be effective. Since diversity in generated via mutation (with probability p_d), higher mutation rates than the standard $1/n$ rate lead to better performances. The best theoretical

Figure 2: Average runtime gain of the $(\mu+1)$ GA versus the (2+1) GA for different population sizes.

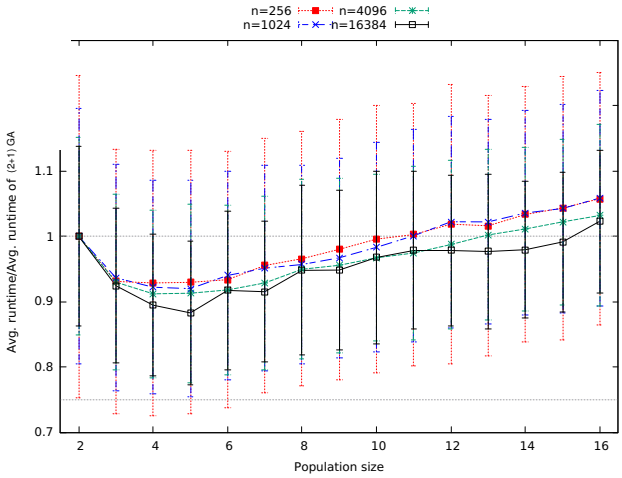
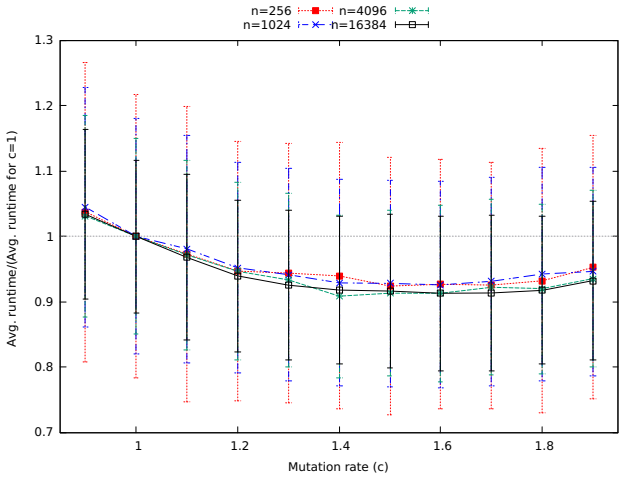


Figure 3: Average runtime gain of the (5+1) GA for various mutation rates versus the standard $1/n$ mutation rate.



upper bound of $0.72en \log n + O(n)$ is achieved for c/n with $c = \frac{1}{2}\left(\sqrt{13}-1\right) \approx 1.3$. Fig. 3 provides experimental evidence that even larger mutation rates up to $c = 1.6$ lead to better performance.

REFERENCES

[1] D. Corus and P. S. Oliveto. 2017. Standard Steady State Genetic Algorithms Can Hillclimb Faster than Mutation-only Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation* (2017). https://doi.org/10.1109/TEVC.2017.2745715

[2] D.-C. Dang, T. Friedrich, T. Kötzing, M. S. Krejca, P. K. Lehre, P. S. Oliveto, D. Sudholt, and A. M. Sutton. 2017. Escaping Local Optima Using Crossover with Emergent Diversity. *IEEE Transactions on Evolutionary Computation* (2017), –. https://doi.org/10.1109/TEVC.2017.2724201

[3] P. S. Oliveto and X. Yao. 2011. Runtime analysis of evolutionary algorithms for discrete optimization. In *Theory of Randomized Search Heuristics: Foundations and Recent Developments*, Benjamin Doerr and Anne Auger (Eds.). World Scientific, 21–52.

[4] D. Sudholt. 2017. How Crossover Speeds Up Building-Block Assembly in Genetic Algorithms. *Evolutionary Computation* 25, 2 (2017), 237–274.