Convergence Properties of $(\mu + \lambda)$ Evolutionary Algorithms

Aram Ter-Sarkisov
School of Computer Science
Massey University
Wellington, New Zealand
a.ter-sarkisov@massey.ac.nz

Stephen Marsland
School of Computer Science
Massey University
Palmerston North, New Zealand
s.r.marsland@massey.ac.nz

Introduction

Evolutionary Algorithms (EA) are a branch of heuristic population-based optimization tools that is growing in popularity (especially for combinatorial and other problems with poorly understood landscapes). Despite their many uses, there are no proofs that an EA will always converge to the global optimum for any general problem. Indeed, only for a set of trivial functions there are any proofs at all.

In common with other research in this area (recent advances include proofs of convergence for $(1 + 1)$EA and Randomized Local Search (RLS) and other interesting properties in Doerr, Johannsen, and Winzen (2011), Doerr, Johannsen, and Winzen (2010), Doerr, Fouz, and Witt (2010), He and Yao (2004) focusing on OneMax (Counting Ones), Binary Values and combinatorial test functions) we denote algorithm $(\mu + \lambda)$EA that has a population of size μ and recombination pool of size λ and analyze some instances thereof (see Algorithms 1-3).

We apply a new EA operator called k-Bit-Swap (kBS) that we introduced in Ter-Sarkisov, Marsland, and Holland (2010). It randomly recombines information from two species in the pool. It can be used instead of and together with mainstream EA operators improving the convergence speed of EAs on several problems. We analyze the working of $(\mu + \lambda)\text{EA}_{kBS}$ on test functions, such as OneMax and Royal Roads to establish its convergence properties and compare it to other EAs.

Test Functions

We use two test functions, OneMax (or Counting Ones) and Royal Roads. The second one is a problem designed to test EAs on schemata recombination. A precise description of it can be found in Mitchell (1996).

$(\mu + \lambda)$ EAs

Here we present the pseudocode of analyzed algorithms. μ and λ are as defined above, n is the length of the chromosome. More about the construction of EAs can be found in Mitchell (1996), Goldberg(1989). We do not use any form of crossover.

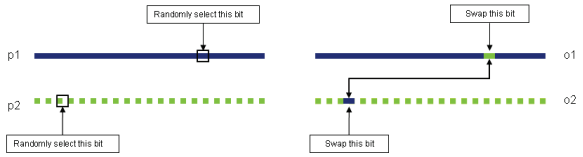

Figure 1: 1-Bit-Swap operator

Algorithm 1: $(\mu + \lambda)\text{EA}_{\frac{k}{n}}$	
1	Initialize population size μ
2	repeat until condition fulfilled:
3	select λ species from the population using Tournament selection
4	flip each bit with probability $\frac{k}{n}$
5	keep α best species in the population, replace the rest with the best species from the pool

Algorithm 2:$(\mu + \lambda)$RLS	
1	Initialize population size μ
2	repeat until condition fulfilled:
3	select λ species from the population using Tournament selection
4	flip exactly one bit per chromosome
5	keep α best species in the population, replace the rest with the best species from the pool

Algorithm 3:$(\mu + \lambda)\text{EA}_{kBS}$	
1	Initialize population size μ
2	repeat until condition fulfilled:
3	select λ species from the population using Tournament selection
4	apply kBS operator to each pair in the recombination pool
5	keep α best species in the population, replace the rest with the best species from the pool

We use a local kBS, that is, k is fixed for each run by the user. An alternative is global kBS, that is, for each pair the number of swapped bits is selected randomly with k being the expectation, exactly like with $(\mu + \lambda)\text{EA}_{\frac{k}{n}}$. We also set $\lambda = \mu$ and 100% mutation and swap rates, i.e., these operators are applied to all species (or pairs of species) in the pool. Tournament selection closely mimics natural selection: two species are selected randomly and the best

one enters the pool.

Analysis basics

EA convergence can be modeled with probabilistic tools, e.g. martingales that are also known as drift function (see Doerr, Johannsen, and Winzen [2010], He and Yao [2004]) in EA theory. The value of the best species in the population is a stochastic process $X_{t,t\geq 1}$. Among other properties, we are interested in the expectation of the least time X_t takes to hit a specific value (or set of values) A. In MC analysis this is denoted by $\tau_A = \min t : X_t \in A$. Specifically, we want it to be at least finite.

$$\mathbf{E}\tau_{1A} = \sum_{t=1}^{\infty} t\mathbf{P}[\tau = t] \tag{1}$$

which under certain assumptions is a version of coupon-collector problem, or the sum of Geometric random variables with a changing parameter. So far in Doerr, Johannsen, and Winzen (2011), Doerr, Fouz, and Witt (2010) the upper and lower bounds for $(1+1)\text{EA}_{\frac{1}{n}}$ optimizing linear functions were found to be $(1+o(1))1.39e\ln n$ and $(1-o(1))en\ln n$ respectively, n being the length of the chromosome (assuming starting fitness value was $\frac{n}{2}$).

For $(\mu+\lambda)EA_{1BS}$ the increments each generation is a random variable ξ_t with distribution

$$\xi_t = \begin{cases} 0 & \text{with probability } 1-p_t \\ 1 & \text{with probability } p_t \end{cases} \tag{2}$$

therefore, the best chromosome in the population is the sum of these increments over time $t : X_t = \frac{n}{2} + \sum_{k=1}^{t} \xi_t$. Distribution in Equation 2 changes over time due to the proportion of 1's in the recombination pool, so $\mathbf{E}[\xi_{t+1}|\xi_t] \neq \mathbf{E}[\xi_{t+1}]$. The roughest way of going about this issue is to find upper and lower bounds on the success probability. A better way is to apply the above-mentioned coupon-collector problem that for $(1+2)\text{EA}_{1BS}$ yields a very sharp upper bound:

$$\mathbf{E}\tau \leq .98n + .5n\log n + 1 \tag{3}$$

which was obtained using the probability of successful swap in the two offsprings in the pool: $p_{swap} = \frac{1}{2} - 2(\frac{k}{n})^2$ where k denotes the number of 1's in each parent.

We need to take into consideration that population μ consists of two disjoint subsets $\alpha_t \cup \beta_t$, where α_t is the set of all best species and β_t is the rest, so the expectation of increment ξ_t and runtime $\mathbf{E}\tau$ depend on the proportion α_t in the population. Since kBS crucially depends on pairing elite species, the probability to pair them using Tournament selection is

$$p_{sel} = \frac{\alpha^2(\alpha+2\beta)^2}{\mu^4} \tag{4}$$

Since we do not know the actual proportion of elite species in the population, we need to account for each case: $1 \leq \alpha_t \leq \mu$ and apply Law of total probability:

$$P(S) = \sum_{j=1}^{\frac{\lambda}{2}} P(S|H_j) \sum_{k=1}^{\mu} P(H_j|A_k)P(A_k) \tag{5}$$

where S is at least 1 new elite species in th next population, H_j is the number of elite pairs in the recombination pool and A_k is the number of elite species in the current population. Next thing we will do is a continuous-time approximation. We are interested in the expected number of improvements occurring in the interval $[s,t)$: $\mathbf{E}[N(s,t)] = m(s,t) = \sum_{k=1}^{n} kp_k(s,t)$ where $p_k(s,t)$ is the probability of occurrence of k events in this interval.

In case these arrivals are homogeneous (do not depend on time), the number of improvements over period $[t-s]$ follows Poisson distribution with intensity $\lambda(t-s)$. This implies interarrival times T_n are exponentially distributed. We will also analyze cases without this assumption (Polya process).

Expected Outcomes

We will derive expectation of runtime for the three population-based elitist EAs applying different operators on two test functions and compare to the existing findings to prove the benefit of population and efficiency of the operators, especially k-Bit-Swap. We will look at other stochastic properties (mean of the population, nonelitist algorithms, continuous approximation, etc.).

Conclusions

In this abstract we have outlined certain ways of analyzing the evolution of an elitist $(\mu+\lambda)$EA solving some test problems. We presented some important notions, such as first hitting time, selection probability, etc used to derive these expressions. Analysis in the area of population-based algorithms is fairly limited and this research will be a worthy addition.

References

Doerr, B.; Fouz, M.; and Witt, C. 2010. Quasirandom Evolutionary Algorithm. In *Genetic and Evolutionary Computing Conference (GECCO) 2010*, 1457–1464.

Doerr, B.; Johannsen, D.; and Winzen, C. 2010. Multiplicative Drift Analysis. In *Genetic and Evolutionary Computing Conference (GECCO) 2010*, 1449–1456.

Doerr, B.; Johannsen, D.; and Winzen, C. 2011. Multiplicative Drift Analysis. In press.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, Massachussetts: Addison-Wesley.

He, J., and Yao, X. 2004. A study of drift analysis for estimating computation time of evolutionary algorithm. *Natural Computing* 3:21–35.

Mitchell, M. 1996. *An Introduction to Genetic Algorithms*. Cambridge, Massachussetts: MIT Press.

Ter-Sarkisov, A.; Marsland, S.; and Holland, B. 2010. The k-Bit-Swap: A New Genetic Algorithm Operator. In *Genetic and Evolutionary Computing Conference (GECCO) 2010*, 815–816.