Extreme Value Based Adaptive Operator Selection

Álvaro Fialho[1], Luís Da Costa[2], Marc Schoenauer[1,2], and Michèle Sebag[1,2]

[1] Microsoft Research-INRIA Joint Centre
28, rue Jean Rostand, 91893 Orsay Cedex, France
[2] Team TAO, INRIA Saclay - Île-de-France & LRI (UMR CNRS 8623)
Bât. 490, Université Paris-Sud, 91405 Orsay Cedex, France
FirstName.LastName@inria.fr

Abstract. Credit Assignment is an important ingredient of several proposals that have been made for Adaptive Operator Selection. Instead of the average fitness improvement of newborn offspring, this paper proposes to use some empirical order statistics of those improvements, arguing that rare but highly beneficial jumps matter as much or more than frequent but small improvements. An extreme value based Credit Assignment is thus proposed, rewarding each operator with the best fitness improvement observed in a sliding window for this operator. This mechanism, combined with existing Adaptive Operator Selection rules, is investigated in an EC-like setting. First results show that the proposed method allows both the *Adaptive Pursuit* and the *Dynamic Multi-Armed Bandit* selection rules to actually track the best operators along evolution.

1 Introduction

Evolutionary Algorithms (EAs) have demonstrated their ability to solve challenging optimization problems that resisted the standard optimization methods, thanks to their flexibility: EAs can handle structured and mixed search spaces, irregular, noisy, or highly constrained objective functions. However, EAs are still a long way from being part of the standard optimization toolboxes; paradoxical as it may seem, the main reason for that is their high flexibility. Indeed, most EAs provide the user with quite a few levers to tackle problem difficulties; although knowledgeable users can benefit from this diversity and take the most out of the Evolutionary approach, the naive user will generally fail to appropriately tune the EA in a reasonable amount of time. Therefore, a mandatory step for EAs to "cross the chasm"and make it out of the research labs is to offer some automatic parameter tuning capabilities.

Parameter setting was and remains one of the most active research directions in EC (see e.g. [1]). Statistical methods derived from Design Of Experiments have been adapted to the off-line setting of EA parameters [2,3,4,5]; while they are more efficient than classical ANOVA, these methods however require extensive experiments. Online tuning seems another promising way of tackling the

G. Rudolph et al. (Eds.): PPSN X, LNCS 5199, pp. 175–184, 2008.

EA parameter control, and in particular, handling the selection of the variation operators. When addressing a new problem, the user can usually define a variety of crossover and mutation operators, fulfilling different roles in the exploitation/exploration dilemma; how to find a strategy for their combined usage is a prominent part of the user's burden. This strategy however most often boils down to a set of static user-defined probabilities, or relative weights, that depend on the user's experience and intuition. After [6], the only dynamic parameter setting strategy widely used in practice concerns the continuous mutation step size adaptation in Evolution Strategies (see references in [6]).

The work presented in this paper is concerned with on-line tuning of operator selection in an EA. Designing an Adaptive Operator Selection (AOS) method involves two main ingredients: the credit assignment mechanism, which associates to each operator a reward, modelling its impact on the progress of evolution; the selection rule, which determines the operator to be used at each time step, depending on the operator rewards. The credit assignment mechanism and the selection rule must be geared to each other to achieve some exploration/exploitation tradeoff in the operator landscape; typically, if the reward provides an instant feedback, modelling the immediate benefits of applying the operator, then the selection rule must ensure that operators with low current benefits can still be explored at a later stage of evolution. Section 2 will present a brief survey of the state of the art, summarizing the credit assignment mechanisms presented in the literature and detailing the selection rules. In particular, the *Probability Matching* (PM) [7] and *Adaptive Pursuit* (AP) [8] will be described, together with the *Dynamic Multi-Armed Bandit* (D-MAB) proposed in [9]. In [9], these three AOS methods have been compared within an artificial setting originally proposed by [8], involving pre-defined rewards whose dynamics are independent from any fitness landscape.

The main contribution of the paper is an original credit assignment mechanism termed *EXtreme value-based Adaptive Operator Selection* (*ExAOS*, described in Section 3), based on empirical order-statistics of the fitness improvement. This mechanism is combined with the above mentioned selection rules and experimentally investigated in an EC-like setting, where the operator rewards computed by *ExAOS* actually follow the dynamics of the evolution trajectory in the fitness landscape (Section 4). This setting considers the eternal OneMax problem; such a simple setting enables to compare the experimental behavior of the online adaptation scheme with the optimal behavior, and to understand the interaction between the dynamics of the fitness landscape, the variance in the operator reward, and the exploration strength in the selection rules. The paper reports on the empirical results in Section 5, and Section 6 concludes with a discussion of the perspectives for further research.

2 Credit Assignment and Adaptive Operator Selection

Credit Assignment. Starting back in the late 80s [10], several methods to assign credit (or reward) to variation operators have been proposed in the literature.

They differ in how they compute the credit of an operator for each newborn offspring. Most methods only use the fitness of the new individual, compared with a reference fitness value: that of the individual parents [11,12,13], of the current best [10] or median [14] individuals. Individuals which do not improve on the reference fitness result in a null credit for the operator. Admittedly, no clear conclusive result can be gathered from those works. Some recent work [15] proposes to use a more sophisticated statistical measure that aims at detecting outliers in the fitness distribution. Reported comparative results with other credit assignment techniques are conclusive, indicating the superiority of this approach over a set of continuous benchmark problems. Though calling to another measure, the method proposed here borrows the idea of detecting beneficial but rare events.

Another distinguishing feature is whether the credit assignment mechanism rewards the operators used to generate the ancestors of the current individual, e.g. using some bucket brigade algorithm [10,14]; creating efficient parents is indeed as important as creating improved offspring. Some authors however do not consider ancestors [11,12] and some even suggest that it sometimes degrades the results [13]. In the rest of the paper, the genealogy of the fit individuals will not be considered. Instant operator credit is computed for each generated offspring, and aggregated through an operator selection rule.

Operator Selection Rules. Most Operator Selection Rules attach a probability to each operator and use a roulette wheel-like process to select the operator to be applied, based on these probabilities[1]. Two such selection rules, namely *Probability Matching* (PM) and *Adaptive Pursuit* (AP), are detailed below.

Let K denote the number of variation operators. Both PM and AP maintain a probability vector $(s_{i,t})_{i=1,K}$, and an estimate of the current operator reward noted $\hat{p}_{i,t}$. At each time t:

- Operator i is selected with probability $s_{i,t}$
- The corresponding reward r_t is computed using the credit assignment at hand
- The reward estimate $\hat{p}_{i,t}$ of the selected operator is updated after r_t, using an additive relaxation mechanism with learning rate α ($0 < \alpha \leq 1$). It controls the memory of the reward estimate (forgettingness increases with α):

$$\hat{p}_{i,t+1} = (1 - \alpha)\hat{p}_{i,t} + \alpha\, r_t \tag{1}$$

Probability Matching, a very popular AOS method [7,11,13], aims to making $s_{i,t}$ proportional to $\hat{p}_{i,t}$, while enforcing a minimal amount of Exploration. More formally, let p_{min} denote the minimal probability of selection of any operator, then:

$$s_{i,t+1} = p_{min} + (1 - K * p_{min}) \frac{\hat{p}_{i,t+1}}{\sum_{j=1}^{K} \hat{p}_{j,t+1}} \tag{2}$$

Note that if some operator gets no reward (respectively the maximal reward) for some time, its expected reward will go to p_{min} (resp. $1 - K * p_{min}$). However,

[1] Methods that recompute those probabilities from scratch from the most recent rewards [14,12] will not be considered here.

this convergence is very slow; experimentally, all mildly relevant operators keep being selected, thus hindering the performance of *Probability Matching* [8].

This drawback is partly addressed by the ***Adaptive Pursuit***: Originally proposed for learning automata, this method follows a winner-take-all strategy, selecting at each time step the operator i_t^* with maximal reward, and accordingly increasing its selection probability:

$$\begin{cases} i^* & = argmax\{\hat{p}_{i,t}\,,\, i = 1 \dots K\} \\ s_{i^*,t+1} & = s_{i^*,t} + \beta\,(1 - (K-1)p_{min} - s_{i^*,t})\,,\ (\beta > 0), \\ s_{i,t+1} & = s_{i,t} + \beta\,(p_{min} - s_{i,t})\,,\ \text{for } i \neq i^* \end{cases} \tag{3}$$

Both PM and AP thus involve the p_{min} parameter to guarantee a sufficient exploration of the operators; AP additionally involves the learning rate β, controlling the greediness of the winner-take-all strategy.

Multi-Armed Bandit Methods. Another approach is inspired from the Multi-Armed Bandit framework, first introduced in the context of Operator Selection by the authors [9]. Multi-Armed Bandit algorithms have been initially proposed as decision making algorithms in uncertain environments.

The so-called *Upper Confidence Bound* (UCB) algorithm devised by Auer et al. [16] achieves the optimal cumulative reward through an Exploration vs Exploitation-based criterion: Let $n_{i,t}$ denote the number of times the i-th arm has been played up to time t, and let $\hat{p}_{i,t}$ denote the average corresponding reward. UCB1 selects in each time step t the arm maximizing:

$$\hat{p}_{j,t} + C\sqrt{\frac{\log \sum_k n_{k,t}}{n_{j,t}}} \tag{4}$$

where C is the *Scaling* factor, controlling the exploration/exploitation tradeoff: the left term in Eq. (4) favors the option with best reward (exploitation) while the right term ensures that each arm is selected infinitely often (exploration). The efficiency of this rule follows from the fact that the lapse of time between two selections of under-optimal arms increases exponentially.

Unfortunately, MABs are not suited to dynamic environments: if the current best option becomes less efficient at some later stage, and happens to be outperformed by another one, it will take a long time before the latter option catches up. A *Dynamic Multi-Armed Bandit* algorithm (D-MAB) was thus proposed in [9], that combines MAB ideas with a specific statistical test known as Page-Hinkley (PH) [17], which is used to detect the changes in the reward distribution, and, upon such a detection, restart the MAB.

More precisely, let $\bar{r}_\ell$ denote the average of $r_1, \dots r_\ell$ and let e_ℓ denote the difference $r_\ell - \bar{r}_\ell + \delta$, where δ is a tolerance parameter. The PH test considers the random variable $m_t = \sum_1^t e_i$. When the difference between $M_t = \max_{i \leq t} m_i$ and m_t is greater than some user-specified threshold γ, the PH test is triggered.

The PH test involves two parameters. Parameter γ controls the trade-off between false alarms and un-noticed changes. Parameter δ enforces the robustness

of the test when dealing with slowly varying environments. Following initial experiments in [9], δ was set to 0.15 in all experiments here.

3 *EXtreme Value-Based Adaptive Operator Selection*

This section presents a proposal for Credit Assignment, to be combined with a selection rule to achieve an Adaptive Operator Selection. Let $\mathcal{F}$, o and x respectively denote the fitness function (to be maximized), a variation operator, and an element of the current population. As discussed in Section 2, the proposed credit assignment will only take into account the non-negative fitness differences $(\mathcal{F}(o(x)) - \mathcal{F}(x))_+$. The proposed mechanism is inspired from the following remark. Let us consider an operator bringing frequent small improvements, and compare it with an operator bringing rare large improvements. The latter one will hardly be considered if the reward reflects the *average* fitness improvement, for the average estimated after a few trials is likely to be 0, implying that very few further trials will take place. Hence, in agreement with [15], attention should be payed to extreme, rather than average, events. Incidentally, the role of extreme events in design has long been acknowledged in numerical engineering (e.g. taking into account rogue waves when dimensioning an oil rig); it receives an ever growing attention in the domain of complex systems, as extreme events govern diffusion-based processes ranging from epidemy propagation to financial markets.

The proposed credit assignment mechanism, referred to as *EXtreme value-based Adaptive Operator Selection* (*ExAOS*), proceeds as follows. When operator o is selected after the selection rule under examination (PM, AP or D-MAB), o is applied on the current individual x; the fitness of the offspring is computed and the current improvement is added to the window (*FIFO* order, with window of size W); lastly, the operator reward is set to the maximal fitness improvement in this time window. Formally, let t be the current time step, and t_1 (respectively t_k) denote the time step where operator o was used for the last time (resp., the last time before t_{k-1}). If $\delta(t)$ denotes the fitness improvement observed at time t, then the expected reward for operator o is computed as:

$$\hat{p}_t = argmax\{\delta(t_i), i = 1 \dots W\} \tag{5}$$

Hence, the *EXtreme value-based Adaptive Operator Selection* mechanism involves a single parameter W, the window size. This parameter W is meant to reflect the time scale of the process; if too large, operators will be applied after their optimal epoch and the switch from the previous best operator to the next best one will be delayed. If W is too small, operators causing large but infrequent jumps will be ignored (as successful events will not be observed at all in the first place) or too rapidly forgotten.

4 Experimental Setting

The artificial setting first proposed in [8] and used in [9] to compare PM, AP (and D-MAB) involved two main simplifications. Firstly, the reward associated

to each operator is assumed to be uniform in a given interval. Secondly, the average reward of every operator is subject to abrupt periodic modifications, jumping from one given interval to another.

The experiments below consider a more realistic environment, embedding the *ExAOS* and the adaptive operator selection rules in an actual EA; rewards are computed after the *ExAOS* mechanism, and their dynamics depends on the evolution trajectory and the fitness landscape. It involves the OneMax problem (the "Drosophila of EC"), with $N = 10,000$ bits. Only mutation operators are considered, ranging from the standard bit-flip operator (every bit is flipped with probability $1/N$) to the b-bit mutations (flipping exactly b randomly chosen bits) with $b = 1, 3, 5$. A standard $(1 + \lambda)$-EA is used (λ offspring are created from the current parent; next parent is the best among the current offspring and parent). One main advantage of this setting is to enable the assessment of the approach by comparison with the known optimal behavior.

In many respects, the considered setting is still far from being realistic evolutionarily speaking (applying a $(1 + \lambda)$-EA, $\lambda > 1$ with b-bit mutations is meaningless on the OneMax problem – though it might make more sense on multi-core architectures). It nevertheless confronts the proposed approach with the actual difficulties of taming a dynamic system, where the decisions made govern the expected benefits of further decisions (the selected operators determine the position of the population and hence the improvement expectation of the operators at further stages), as opposed to [8,9]. The considered setting is thus meant to be a "sterile EC-like" environment.

The *ExAOS* mechanism is independently investigated in combination with the three selection rules, AP, PM and D-MAB. The goal of the experiments is to assess the relevance of *ExAOS* in interaction with the three selection rules. The main criterion of performance clearly is the average time-to-solution, though the ability of the adaptive scheme to track the best operator is also considered. In all reported experiments, the initial individual is set to $(0, \ldots, 0)$. However, as PM was found significantly outperformed in all pairwise tests, its results are not presented here. Every selection rule is used with its optimal setting, determined after a preliminary DOE campaign [9]. All results are validated using 11 independent runs and followed by a one-way ANOVA with $\alpha = 0.05$, eventually followed by pairwise Scheffé tests.

5 Experimental Validation

The optimal baseline is provided by the optimal behavior of all operators (computed by a Monte-Carlo simulation). Fig. 1 depicts the operator landscape from the perspective of a $(1 + 50)$-EA; for each fitness we report the fitness gain for the best out of 50 offsprings generated respectively with the 1-,3-,5-bit or bit-flip mutation (averaged on 100 runs).

The trajectory of evolution involves distinct phases. In *stable* phases, the optimal operator remains the same (though its performance might decrease). For instance, while the 5-bit mutation dominates all other operators while $\mathcal{F}(x) <$

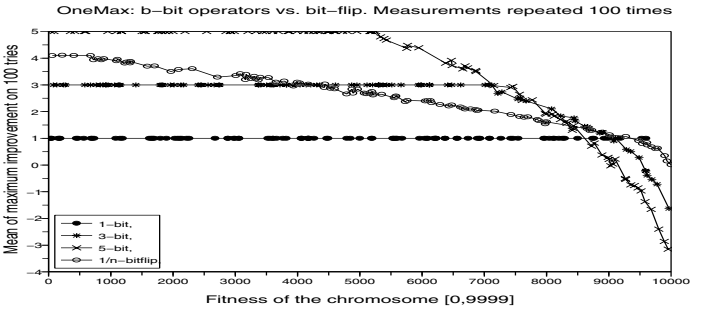


Fig. 1. Average fitness gain with 1-bit, 3-bit, 5-bit and bit-flip mutations within $(1 + 50)$-EA vs fitness of parent. The best operators are: 5-bit mutation in $[0, 6579]$; 3-bit mutation in $[6580, 8400]$; bit-flip for $[8401, 8600]$; and 1-bit for fitness > 8600.

6579, its performance decreases as the fitness increases after $\mathcal{F}(x) = 5300$. In *transition* phases, the established best operator becomes dominated by another one; the 3-bit mutation outperforms the 5-bit after $\mathcal{F}(x) = 6579$ and the 1-bit mutation outperforms the 3-bit after $\mathcal{F}(x) = 8601$. The last phase is a *desert*, where hardly any operator brings any improvement.

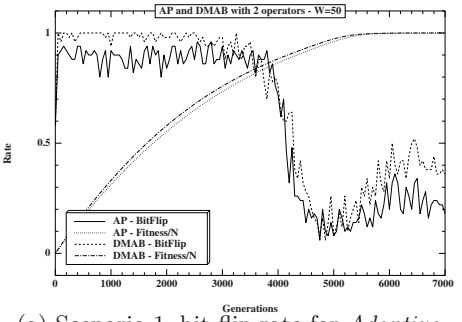
Such an operator landscape enables to assess the basic skills of an Adaptive Operator Selection mechanism: the ability to pick up the best operator and stick to it in stability phases; to swiftly switch to the next best operator in transition phases; and to remain efficient during the desert phases.

Scenario 1. The first experiment considers only the 1-bit and bit-flip mutations, examining the operator rates adapted by *ExAOS* ($W = 50$), AP and D-MAB selection rules, comparatively to the optimal decisions.

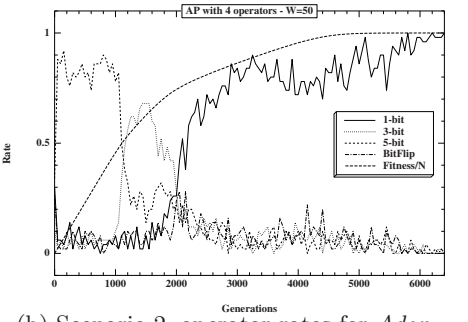
At the beginning of the trajectory (from $(0, \ldots, 0)$), the 1-bit mutation brings a constant improvement of 1 (independently of λ) whereas the average expected reward of bit-flip increases with λ, but with a high variance. This intuition is confirmed by simulations with $\lambda = 1, 5$, and 10. When $\lambda = 1$, bit-flip outperforms 1-bit only until fitness=7, but until fitness=4753 when $\lambda = 5$, fitness=6469 when $\lambda = 10$, and until fitness=8722 when $\lambda = 50$. Therefore, the optimal decision in a $(1 + \lambda)$-EA would be to *always* start with the bit-flip mutation, and to switch to 1-bit afterwards (e.g. at fitness $= 8722$ for $\lambda = 50$).

The experimental results (Fig. 2.a) demonstrate a good agreement with the optimal rates; the bit-flip rate is close to 1 in the early stages of evolution and switches to p_{min} shortly after the transition point. In the desert phase where rewards are extremely rare, the selection rule consistently selects the 1-bit mutation in the majority of cases, although a high level of exploration is still performed (and would allow any beneficial operator to eventually catch up).

Principled investigations varying λ and W are reported in Table 1, using the best naive strategy (among different fixed mixtures of operators, including using each operator alone) as basline. Firstly. AP and D-MAB obtain comparable



(a) Scenario 1, bit-flip rate for *Adaptive Pursuit* and D-MAB.



(b) Scenario 2. operator rates for *Adaptive Pursuit*

Fig. 2. Adaptive Operator Selection: Operator Rates for $\lambda = W = 50$ (avg. / 11 runs)

Table 1. Comparative results on both scenarios for AP and D-MAB. The figures are the mean number of **generations** (std. dev.) to reach the global optimum. The *best naive* strategy is chosen among 1-bit alone, bit-flip alone or a uniform mixture of both.

		Scenario 1		Scenario 2		Best
λ	W	AP	D-MAB	AP	D-MAB	naive
1	1	93720 (5158)	95296 (6224)	91221 (7738)	$> 100k$	1-bit
	50	93890 (7363)	98871 (3704)	92700 (6800)	98467 (4067)	94928 (4776)
2	1	51629 (2910)	54880 (6379)	49609 (4159)	67085 (7911)	1-bit
	50	52905 (5667)	58514 (6087)	48950 (6352)	59496 (9780)	51817 (3760)
5	1	25284 (1128)	26230 (3096)	21536 (1640)	27421 (2500)	1-bit
	50	24668 (1954)	25683 (1180)	21225 (1776)	24966 (5161)	25715 (1392)
10	1	16558 (980)	16437 (1174)	12769 (1035)	15068 (1230)	1-bit
	50	14521 (1165)	15265 (1011)	12517 (967)	14256 (1748)	16740 (597)
25	1	10285 (326)	10343 (720)	7937 (501)	7778 (591)	Uniform
	50	8830 (493)	8733 (529)	7393 (614)	7728 (768)	10752 (309)
50	1	7882 (245)	7547 (318)	5715 (212)	5786 (364)	Uniform
	50	6619 (285)	6460 (285)	5476 (248)	5513 (431)	7329 (147)

performances on this scenario, not significantly better than the best of the naive strategies: ANOVA rejects the null hypothesis, but all pairwise Scheffé tests fail, even though the means of AP and D-MAB are slightly better than the others.

Secondly, the improvement of using memory ($W = 50$) is found significant for $\lambda \geq 10$ (with decreasing p-values for increasing λ). Indeed, the instant reward ($W = 1$) gives little information on the expected fitness gain out of λ offspring (or even $\lambda/2$ offspring during the exploration phase).

Scenario 2. Scenario 2 presents the AOS mechanism with two additional difficulties. Firstly, it considers all four mutation operators (1,3,5-bit and bit-flip),

and thus expectedly requires a higher amount of exploration. Secondly, the operator landscape involves several transition points (Fig. 1): the 5-bit mutation is the best one until $\mathcal{F}(x) = 6579$, the 3-bit mutation until $\mathcal{F}(x) = 8400$, then the bit-flip until $\mathcal{F}(x) = 8722$, and finally the 1-bit dominates until the end. Qualitatively, the experimental results of *Adaptive Pursuit* (Fig. 2.b) and D-MAB (not shown) closely match the above optimal behavior for $W = 50$.

Again, Table 1 reports on the performances of AP and *Dynamic Multi-Armed Bandit* for different values of λ and W. The results are statistically similar to those of scenario 1: ANOVA rejects the null hypothesis, but no pair-wise difference can be found significant between AP, D-MAB and the best naive strategy, even though AP and D-MAB have larger means for large values of λ (AP slightly outperforming here D-MAB). Regarding the effect of memory, setting $W = 50$ does bring, here again, some improvements of the mean performances, but these differences are found significant only in the case of AP with $\lambda = 50$.

6 Discussion and Perspectives

Compared to earlier work related to Adaptive Operator Selection [8,9], this paper presents two extensions. The first one is a new credit assignment method, *EXtreme value-based Adaptive Operator Selection*, translating the fitness gains brought by an operator into actionable rewards, to be exploited by selection rules such as *Adaptive Pursuit* or *Dynamic Multi-Armed Bandit*. Along the same lines as [15], *ExAOS* is driven by the extreme fitness gains brought by an operator, as opposed to the average fitness gain. The *rationale* is that EC must be able to explore "risky" operators, providing rare and large jumps, while average gain-based rewards are strongly biased toward conservative strategies.

The second contribution of the paper is an experimental setting enabling to investigate the AOS behavior *in situ*; although it is still a long way from being evolutionarily challenging, this setting definitely improves on the one considered in [8,9], as it actually couples AOS with an evolutionary-driven system. Within this setting, experiments demonstrate that the adapted operator rates satisfactorily match the optimal ones. However, some problems remain, regarding the (meta-)parameter setting of those methods: The best results of PM and AP were obtained using p_{min}=0, contradicting its definition; Similarly, the performances of D-MAB using *ExAOS* could be improved by a better understanding of the interaction between the scaling factor C (Eq. 4), the PH threshold γ (see Section 2, or [9]), λ and W. Hopefully, the proposed experimental setting will help us in that respect, using other well-known benchmark functions.

Further research is concerned with assessing the respective roles of the time window and the maximum improvement. Furthermore, it has been emphasized that one of EC strengths is to be a rank-based optimization method [18,19]. Accordingly, the reward associated to an operator might consider the top rank of the last fitness gains, as opposed to, its extreme value. Another perspective is to use the extreme value statistics to online adapt the λ parameter in $(1+\lambda)$-EA;

exceptional offspring (wrt extreme gains) can immediately replace the parent without waiting for all λ offspring to be generated.

References

1. Lobo, F., Lima, C., Michalewicz, Z. (eds.): Parameter Setting in Evolutionary Algorithms. Springer, Heidelberg (2007)
2. Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K.: A racing algorithm for configuring metaheuristics. In: Langdon, W.B., et al. (eds.) Proc. GECCO 2002, pp. 11–18. Morgan Kaufmann, San Francisco (2002)
3. Yuan, B., Gallagher, M.: Statistical racing techniques for improved empirical evaluation of evolutionary algorithms. In: Yao, X., et al. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 172–181. Springer, Heidelberg (2004)
4. Bartz-Beielstein, T., Lasarczyk, C., Preuss, M.: Sequential parameter optimization. In: Proc. CEC 2005, IEEE Press, pp. 773–780. IEEE Press, Los Alamitos (2005)
5. Nannen, V., Eiben, A.E.: Relevance estimation and value calibration of evolutionary algorithm parameters. In: Proc. IJCAI 2007, Hyderabad, India, pp. 975–980 (2007)
6. De Jong, K.: Parameter Setting in EAs: a 30 Year Perspective. In: [1], pp. 1–18
7. Goldberg, D.: Probability Matching, the Magnitude of Reinforcement, and Classifier System Bidding. Machine Learning 5(4), 407–426 (1990)
8. Thierens, D.: An adaptive pursuit strategy for allocating operator probabilities. In: Beyer, H.G. (ed.) Proc. GECCO 2005, pp. 1539–1546. ACM Press, New York (2005)
9. Da Costa, L., Fialho, A., Schoenauer, M., Sebag, M.: Adaptive operator selection with dynamic multi-armed bandits. In: Keijzer, M., et al. (eds.) Proc. GECCO 2008, ACM Press, New York (to appear, 2008)
10. Davis, L.: Adapting operator probabilities in genetic algorithms. In: Schaffer, J.D. (ed.) Proc. ICGA 1989, pp. 61–69. Morgan Kaufmann, San Francisco (1989)
11. Lobo, F., Goldberg, D.: Decision making in a hybrid genetic algorithm. In: Proc. ICEC 1997, pp. 121–125. IEEE Press, Los Alamitos (1997)
12. Tuson, A., Ross, P.: Adapting operator settings in genetic algorithms. Evolutionary Computation 6(2), 161–184 (1998)
13. Barbosa, H.J.C., e Sá, A.M.: On adaptive operator probabilities in real coded genetic algorithms. In: XX Intl. Conf. of the Chilean Computer Science Society (2000)
14. Julstrom, B.A.: What have you done for me lately? adapting operator probabilities in a steady-state genetic algorithm on genetic algorithms. In: Eshelman, L.J. (ed.) Proc. ICGA 1995, pp. 81–87. Morgan Kaufmann, San Francisco (1995)
15. Whitacre, J.M., Pham, T.Q., Sarker, R.A.: Use of statistical outlier detection method in adaptive evolutionary algorithms. In: Cattolico, M. (ed.) Proc. GECCO 2006, pp. 1345–1352. ACM, New York (2006)
16. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2/3), 235–256 (2002)
17. Page, E.: Continuous inspection schemes. Biometrika 41, 100–115 (1954)
18. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation 9(2), 159–195 (2001)
19. Gelly, S., Ruette, S., Teytaud, O.: Comparison-based algorithms are robust and randomized algorithms are anytime. Evolutionary Comp. 15(4), 411–434 (2007)