Analysis of Adaptive Operator Selection Techniques on the Royal Road and Long K-Path Problems

Álvaro Fialho, Marc Schoenauer, Michèle Sebag

HAL Id: inria-00377449

https://hal.inria.fr/inria-00377449v2

Submitted on 16 Jul 2009

Analysis of Adaptive Operator Selection Techniques on the Royal Road and Long K-Path Problems

Álvaro Fialho[1], Marc Schoenauer[1,2], Michèle Sebag[1,2]

[1]Microsoft Research–INRIA Joint Centre
Parc Orsay Université
91893 Orsay Cedex, France

[2]Project-Team TAO, INRIA Saclay - Île-de-France
LRI, Bât. 490, Université Paris-Sud
91405 Orsay Cedex, France

FirstName.LastName@inria.fr

ABSTRACT

One of the choices that most affect the performance of Evolutionary Algorithms is the selection of the variation operators that are efficient to solve the problem at hand. This work presents an empirical analysis of different Adaptive Operator Selection (AOS) methods, i.e., techniques that automatically select the operator to be applied among the available ones, while searching for the solution. Four previously published operator selection rules are combined to four different credit assignment mechanisms. These 16 AOS combinations are analyzed and compared in the light of two well-known benchmark problems in Evolutionary Computation, the Royal Road and the Long K-Path.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence: Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Genetic Algorithms, Parameter Control, Adaptive Operator Selection

1. INTRODUCTION

Evolutionary Algorithms (EAs) constitute efficient solvers for general optimization problems and their performances have been assessed on a wide range of applications. Algorithmically, EAs proceed by selecting and applying transformation, a.k.a. variation, operators on sets of possible configurations of the problem to be solved. The EAs efficiency relies on making quite a few appropriate algorithmic and parametric decisions. On the so-called genotypic level are the suitable encoding of the search space and the variation operators, e.g., mutation and crossover; on the phenotypic level are the selection procedures and the size of the population.

This paper focuses on parameter setting in EAs, which usually requires an extensive expertise in either the problem to be solved, or EAs, or both. Parameter setting has been and still is acknowledged a most critical aspect of Evolutionary Computation [23]. Interestingly, the search for algorithmic technologies enabling the (naive) end-user to benefit from good performances through autonomous parameter setting is considered a priority in neighbor fields such as operation research or constraint programming [18, 24]; these fields likewise involve sophisticated solver platforms, requiring an extensive expertise to be used to their fullest extent.

Parameter setting involves two main components: acquiring some knowledge about the fitness landscape of the problem at hand, referred to as *learning*; and exploiting the acquired knowledge to appropriately shape the algorithm, referred to as *adaptation*. While parameter setting actually regards the selection of any solver components, this paper will only consider the *on-line selection rate* of the variation operators, such as mutation and crossover.

On-line learning and adaptation raise two specific issues compared to their offline equivalent. Firstly, on-line learning associates a value or *reward* to each operator, depending on its current effects, while offline learning observes *a posteriori* the contribution of the operator to the overall performance. On-line learning requires a *Credit Assignment* mechanism to be designed, typically considering the average effects of the operator and/or its peak (extreme) effects. Secondly, the past rewards attached to each operator are used to define a selection rule in charge of actually selecting the current operator (*Operator Selection*). Lastly, the *Credit Assignment* and *Operator Selection* together must account for the fact that the genetic population follows some trajectory in the fitness landscape; the instant reward attached to each operator, measuring e.g., the fitness gain of the offspring w.r.t. the parent, not only is a random variable; furthermore, the underlying distribution of this random variable is a dynamic one, changing as evolution goes on.

A short survey of autonomous parameter setting, *Credit Assignment* and *Operator Selection* in EAs is presented in Section 2, focussing on four *Operator Selection* methods, namely Probability Matching and Adaptive Pursuit [30, 31] on the one hand, and the static and dynamic versions of the Multi-Armed Bandit [5, 10] on the other hand.

The first contribution of this paper is an extensive experimental study of the above four *Operator Selection* methods, combined with four *Credit Assignment* mechanisms, namely

the extreme [33] and the average fitness improvements, either in absolute value, or normalized w.r.t. the current ones. While the above-mentioned methods have been studied on merely artificial settings [30, 31, 5] or on the OneMax problem [10], the present paper considers the Royal Road [16] and the Long K-Path [17] problems, which have been extensively investigated in the Evolutionary Computation (EC) literature. For the sake of efficiency, the 16 parameter setting frameworks are experimented at their best; a racing mechanism [4] is used to find the best hyper-parameters for these frameworks in a tractable way. The experimental results are presented and discussed in Section 4, and the paper concludes with some perspectives for further research.

2. BACKGROUND AND STATE OF THE ART

After a brief introduction to Evolutionary Parameter Setting, this section focuses on *Credit Assignment* and *Operator Selection*.

2.1 Evolutionary Parameter Setting

After [8, 9], Evolutionary Parameter Setting proceeds along two main modes. Offline or external tuning, referred to as **Parameter Tuning**, determines *a priori* the appropriate parameter values. Parameter tuning thus takes place before the run, e.g., exploiting the lessons learned from previous runs. Standard approaches from experimental studies such as *ANOVA* or *Design Of Experiments* have been used for *Parameter Tuning*, e.g., modeling the impact of parameter values on the overall performance and accordingly determining the optimal values [4, 34, 3, 27]. These methods however are very computationally expensive as each observation corresponds to the average of a few evolutionary runs; furthermore, only static settings are considered (the parameter value is fixed along the run), whereas the optimal setting likely depends on the local landscape explored by the genetic population.

On-line or internal tuning, referred to as **Parameter Control**. determines the appropriate parameter values at each time step during the evolutionary run. One further distinguishes *Deterministic* (parameter values are predefined functions of time), *Self-Adaptive* (parameters are part of the genotypic information and optimized by evolution itself), and *Adaptive* (parameter values are predefined functions of the whole history of the run) Parameter Control.

Deterministic Parameter Control essentially raises the same difficulties as Parameter Tuning: while the parameter values depend on the time step, these functions must still be defined *a priori*. Self-Adaptive Parameter Control is acknowledged one of the most effective approaches to evolutionary parameter setting, specifically in the framework of continuous parameter optimization (see the discussion in [7] and references therein). In the general case however, self-adaptive approaches often significantly increase the size of the search space, and/or the complexity of the optimization problem (not only should a successful individual have good genes; it should also bear parameter values enforcing some effective transmission of its genes).

Adaptive Parameter Control, also referred to as Feedback-Based control, use information from the history of evolution to modify the parameter values while solving the problem. *Adaptive Operator Selection* (AOS), a particular case of adaptive parameter control, aims to defining an on-line strategy for selecting the most appropriate variation operators. As shown on Fig. 1, AOS involves two subproblems, detailed in next subsections: i) assessing the impact of each operator upon the progress of evolution (*Credit Assignment*); ii) using these assessments to actually select the operator with best impact expectation (*Operator Selection*).
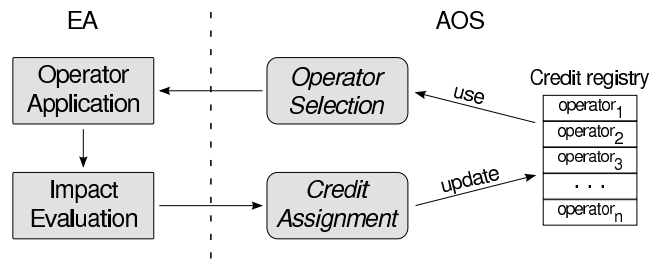

Figure 1: The Adaptive Operator Selection scheme.

2.2 Credit Assignment

Several *Credit Assignment* mechanisms have been proposed in the literature, following Davis' seminal paper [6]. These mechanisms mostly differ by the measure used to compute the credit, and the genetic individuals taken into consideration.

Most approaches consider the new offspring and use its fitness improvement as credit measure. The fitness improvement is assessed by comparison with i) the current best individual [6]; ii) the median fitness [20]; or iii) the parent fitness [22, 32, 2]. When there is no improvement, the offspring is simply discarded.

In the case of multi-modal optimization, another relevant measure concerns the population diversity, which must be enforced to avoid premature convergence. Along this line, [25] proposed another credit measure called *Compass*, defined as a weighted sum of fitness improvement (intensification) and offspring diversity (diversification).

While the credit measure usually considers instantaneous or average improvement (the average being taken over the last n applications of the operator), [33] proposes instead to consider extreme improvements, using a statistical measure aimed at outlier detection. The experimental results presented show that the method significantly outperforms its competitors on a set of continuous benchmark problems.

Independently, some authors consider that the operator impact should be measured after the genealogy of the outstanding offspring, e.g., rewarding the operators producing the ancestors of a good offspring according to a bucket brigade algorithm [6, 20]. No clear indication however about the benefits of this approach is found in the literature to the best of our knowledge.

2.3 Operator Selection Rules

Most *Operator Selection* rules attach a probability of success to each operator[1]. These probabilities can be used for selection along a roulette wheel-like process, like Probability Matching (PM) and Adaptive Pursuit (AP) (section 2.3.1); another possibility is based on the so-called Multi-Armed Bandit framework [1] (section 2.3.2).

---

[1]Methods that recompute those probabilities from scratch from the most recent rewards [20, 32] will not be considered here.

2.3.1 Probability Matching and Adaptive Pursuit

Let K denote the number of variation operators. PM and AP both 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:

i) the i-th operator is selected with probability $s_{i,t}$, and gets an instant reward r computed after the credit assignment at hand

ii) the reward estimate $\hat{p}_{i,t}$ of the i-th operator is updated using an additive relaxation mechanism with adaptation rate α ($0 < \alpha \le 1$, the memory span decreases as α increases):

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

Probability Matching mostly selects the i-the operator proportionally to $\hat{p}_{i,t}$, except for the fact that a minimum amount of exploration can be enforced. If the selection probability of an operator would become too low at some point, it would never be used again, thus precluding AOS from discovering that it becomes the optimal one in further stages of evolution.

Formally, letting p_{min} denote the minimal selection probability, then the selection probability of the i-th operator is defined as:

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

After Eq (2), any ineffective operator (not getting any reward) would be selected with probability p_{min}, while the best operator (getting maximal rewards) would be selected with probability $1 - K * p_{min}$. In practice, all mildly relevant operators keep being selected, hindering the *Probability Matching* performance (all the more so as the number of operators increases) [30].

Adaptive Pursuit, originally proposed for learning automata, has been used in AOS to address the above *Probability Matching* shortcoming; a winner-take-all strategy is used to push forward the best current operator noted i_t^* as follows, where $p_{max} = (1 - (K-1)p_{min})$:

$$\begin{cases} i_t^* & = \arg\max_{i=1...K}\{\hat{p}_{i,t}\} \\ s_{i,t+1} & = \begin{cases} s_{i,t} + \beta\,(p_{max} - s_{i,t}) & if\ i = i_t^* \\ s_{i,t} + \beta\,(p_{min} - s_{i,t}) & \text{otherwise} \end{cases} \end{cases} \qquad (3)$$

Finally, both PM and AP are controlled from the p_{min} parameter (enforcing the exploration of the operators) and adaptation rate α (ruling the memory span of the AOS). AP additionally involves learning rate β, ruling the greediness of the winner-take-all strategy.

2.3.2 Static and Dynamic Multi-Armed Bandit

Operator selection can be framed as another *Exploration vs. Exploitation* (EvE) dilemma, where Exploitation aims at selecting the best rewarded operators in the last stages of evolution whereas Exploration is concerned with checking whether other operators might in fact become the best ones at some later stages. The EvE dilemma has been intensively studied in *Game Theory*, more specifically in the so-called Multi-Armed Bandit (MAB) framework [21, 1].

The MAB framework considers a set of K independent arms, each one of which having some unknown probability of getting a (boolean) reward. The optimal selection strategy is one maximizing the cumulative reward along time. The *Upper Confidence Bound* (UCB) selection strategy proposed by Auer et al. [1], providing asymptotic optimality guarantees, can be phrased as *Optimism in front of the Unknown*. Formally, to the i-th arm is associated i) its empirical reward $\hat{p}_i$ (the average reward obtained) and ii) a confidence interval, depending on the number of times n_i the i-th arm has been tried. UCB selects in each time step the arm with best upper bound of the confidence interval:

$$\text{Select } \arg\max_{i=1...K}\left(\hat{p}_{i,t} + C\sqrt{\frac{\log\sum_k n_{k,t}}{n_{i,t}}}\right) \qquad (4)$$

The C parameter, referred to as *scaling* factor, controls the tradeoff between exploitation (left term in Eq. (4), favoring the arms with best empirical reward) and exploration (right term, favoring the infrequently tried arms). The efficiency of the UCB rule follows from the fact that, although every arm is selected exponentially often, the lapse of time between two selections of some under-optimal arm increases exponentially.

The standard MAB framework and the UCB algorithm however consider a static environment (the unknown reward probability of any arm being fixed along time), whereas the AOS framework is intrinsically dynamic (the quality of any operator is bound to vary along evolution). Even though every operator keeps being selected, enabling UCB to ultimately realize that some new operator has become the best one, in practice UCB would need to wait way too long before switching to the best operator. A Dynamic Multi-Armed Bandit (DMAB) strategy [14] has thus been used in [5] to address this limitation, coupling UCB with a change detection test, the statistical Page-Hinkley (PH) test [15]. Basically, the PH test is in charge of checking whether the operator reward distribution has changed; upon its triggering, UCB is restarted (i.e., the empirical rewards and confidence intervals are re-initialized) in order to quickly identify the new best operators without being slowed down by now irrelevant information.

Formally, the PH test works as follows, where $\bar{r}_t$ denotes the average reward over the last t steps, and e_t the difference between the instant and average reward, plus some small tolerance δ. Considering the random variable $m_t = \sum_1^t e_i$, the PH test is triggered when the difference between $M_t = \max_{i\le t}|m_i|$ and $|m_t|$ is greater than some user-specified threshold γ:

$$\bar{r}_t = \tfrac{1}{t}\sum_{i=1}^{t} r_i \qquad m_t = \sum_{i=1}^{t}(r_i - \bar{r}_i + \delta)$$
$$\text{Return } (\max_{i=1...t}\{|m_i|\} - |m_t| > \gamma) \qquad (5)$$

The PH test thus is parametrized by γ (controlling the test sensitivity and the rate of false alarms) and δ (enforcing the test robustness w.r.t slowly varying environments). Following early experiments, δ has been kept fixed to 0.15 throughout this work.

3. GOALS OF THE STUDY

As mentioned earlier on, the above AOS settings have been mostly considered outside of any evolutionary environment [30, 31, 5], assuming the operator reward to be neatly defined as a (continuous or boolean) random variable following a periodically changing (uniform or Bernoulli) distribution. The first embedding of the above AOS schemes within an actual evolutionary algorithm has been investigated in

[10]; the operator reward was based on the fitness improvement of the offspring compared to the parent, and computed as either the average (AverageReward) or the maximum (ExtremeReward) fitness improvement observed the last $\mathcal{W}$ times the operator had been applied. The fitness landscape considered in [10], the OneMax landscape, however is devoid of any deceptivity or discontinuity, two among the main difficulties faced by EC on real-world problems. Deceptive landscapes, intensively investigated in the EC literature [13], involve a (concatenation of) sub-optimal, OneMax like regions, besides an external optimal peak. Discontinuous landscapes involves short-cuts, where significantly higher fitness regions can be discovered through the lucky application of a given variation operator.

The goal of the present paper thus is to study the AOS behavior w.r.t deceptive and discontinuous landscapes; the deceptive Royal Road [16, 28] and Long K-Path problems (briefly described in section 4.1 for the sake of self containedness) are considered as well-studied representatives of both types of difficulties. Specifically, Royal Road and Long K-Path problems raise an additional challenging issue for AOS compared to the OneMax problem. Fitness improvements are no longer homogeneous along evolution; whereas beneficial mutations all increase the fitness by the same amount in the OneMax problem, a beneficial crossover in Royal Road, or a beneficial mutation in Long K-Path, improve the fitness by an order of magnitude more than the standard fitness improvement. Furthermore, upon such a beneficial event, evolution migrates toward another region, potentially causing the reward distribution of all operators to change abruptly.

The effects of such fitness leaps will be empirically investigated through introducing an additional normalization mechanism as follows. In the rest of the paper, the instant operator value refers to the fitness gain of the offspring compared to its parent (mutation case) or its best parent (crossover case). The value is either set to the fitness gain (AbsoluteValue), or to the fitness gain *divided by the best fitness gain gathered by an operator* in the last $\mathcal{W}$ time steps (NormalizedValue). The operator reward is finally set to either the instant value averaged over the last $\mathcal{W}$ times the operator has been applied (AverageReward), or to the maximal (extreme) instant value observed during the last $\mathcal{W}$ times the operator has been applied (ExtremeReward). Overall, four types of reward will thus be considered: ExtremeAbsoluteReward(XAbs), ExtremeNormalizedReward(XNorm), AverageAbsoluteReward (AvgAbs) and AverageNormalizedReward (AvgNorm). All four *Credit Assignment* involve the time window $\mathcal{W}$ as single hyper-parameter. Note that $\mathcal{W}$ relates to the time scale of evolution; if too large, operators will be applied after their optimal epoch and the switch from the previous best operator to the new best one will be delayed. If $\mathcal{W}$ is too small, operators causing large but infrequent jumps will be ignored (as lucky events will not be observed in the first place), or rapidly forgotten.

These four *Credit Assignment* will be independently combined with the four *Operator Selection* (PM, AP, Multi-Armed Bandit and Dynamic Multi-Armed Bandit) described in section 2.3. For the sake of the lessons learned, each one of these 16 AOS settings will be launched with the best hyper-parameters. A secondary goal of the study is to assess the stability and robustness of the AOS settings with respect to their hyper-parameters.

4. EXPERIMENTAL SETTING

This section first describes the Royal Road and Long K-Path problems used for the empirical validation of the AOS schemes. AOS hyper parameters are thereafter summarized, and the racing procedure used to select appropriate values for these hyper-parameters in a tractable way is last described.

4.1 The Royal Road

Royal Road problems were purposely devised as difficult optimization problems for hill-climbing algorithms, while being easy for GAs as they are made of "building blocks" [26]. Due to unexpected difficulties (the so-called hitch-hiking phenomenon), revised Royal Road problems were proposed [16] and analyzed [19].

In the revised Royal Road landscape, each bit-string is divided in 2^k regions, referred to as first-level *schemata*; each schema is made of a *block* and a *gap* string, of respective length b and g. Higher level schemata are formed by combining lower-level ones. Formally, 2^{k-L} schemata of level L are defined, each one being made of 2^L 1st-level ones, supposedly defining a crossover-friendly landscape.

The fitness function only considers the block region of each low level schema, ignoring the gap region. The PART function computes the number z of correct bits in the b-length block; the associated fitness is $z \times v$ if $z < m$ and $(b - z) \times v$ for $m < z < b$, thus corresponding to a locally deceptive fitness function. If the block is complete ($z = b$), and the block is the first one to be formed in the individual, the PART function is replaced by a BONUS one, and the individual fitness scores a bonus of u^*; any further complete block is worth an additional u fitness score.

After [28], uniform crossover (respectively 1-point crossover) allegedly is the best operator during the first (resp. the last) evolution stages; the 4-point crossover is the best operator in-between. The goal of the experiments with the Royal Road problem thus is whether the AOS approaches actually select the appropriate operators at different stages of evolution.

4.2 The Long K-Path Problem

Originally proposed by Horn et al. [17], *Long paths* problems are unimodal optimization problems defined on $\{0,1\}^\ell$, designed to challenge local search algorithms. The optimum can be found by climbing a path, the length of which increases exponentially with the dimension ℓ of the search space; efficient optimization thus relies on finding short-cuts on the path.

The so-called Long k-paths problems, introduced by [29], generalizes the long paths problems; k is the minimal number of bits that must be simultaneously flipped in order to take a shortcut on the path. Formally the Long K-Path can be described as follows [12]:

- The path starts at point $0, \ldots, 0$, with fitness ℓ; the fitness of any point not on the path is the number of its 0 bits;

- Any point on the path has exactly 2 neighbors at Hamming distance 1 that are on the path;

- Mutating $i < k$ bits of a point on the path leads to a point which is either off the path (hence with a very low fitness), or on the path but only i positions away from the parent point;

- A shortcut is found by mutating the correct k bits (or more), thus with probability $p^k(1-p)^{l-k}$.

Long K-Path problems are defined by recurrence on ℓ. The path associated to problem $P(k, \ell + k)$ is built as the sequence of $(x_i, 0_k)$ where x_i belongs to $P(k, \ell)$ and 0_k is the k-length vector made of 0s, followed by a "bridge", followed by the sequence $(x_{L-i}, 1_k)$, where x_{L-i} ranges in inverse order in $P(k, \ell)$ and 1_k is the k-length vector made of 1s. The bridge is the sequence of (x_L, y_z) where x_L is the last point of path $P(k, \ell)$ and y_z is the k-length vector made of z 0s followed by $k-z$ 1s. It turns out that the path length decreases as k increases (the original long path corresponds to $k = 2$). The probability of finding a shortcut however exponentially decreases with k, and the fastest strategy for $k > \sqrt{\ell}$ is to simply follow the path [12]. Otherwise ($k \leq \sqrt{l}$), optimization should provably strive to find the shortcuts; in such cases, *exceptional properties of operators are more relevant to EAs behavior than their average properties.*

Overall, the choice of the Long K-Path problem is meant to investigate whether (and which) AOS approaches manage to use operators which rarely but very significantly contribute to the progress of optimization. This problem was also considered in [11] to investigate the relevance of extreme-value based (as opposed to average-value based) rewards. Finally, artificial Long K-Path problems make it feasible to identify the optimal operator at each point of the path (e.g., through intensive Monte-Carlo simulations), and to assess the AOS approaches by comparison with the optimal strategy.

4.3 Hyper-parameters and F-Racing

Heuristic	H-P	Range	Comments
XAbs, XNorm, Avg{Abs,Norm}	$\mathcal{W}$	$\{50, 500\}$	Time window
AP, PM	p_{min}	$\{0; .05; .1; .2\}$	Min. select. prob.
AP, PM	α	$\{.1, .3, .6, .9\}$	Adaptation rate
AP	β	$\{.1, .3, .6, .9\}$	Learning rate
MAB, DMAB	$\mathcal{C}$	$\{\{1, 5\}.10^{\{-4 \leq i \leq 1\}}, 25, 100\}$	Scaling factor
DMAB (PH)	γ	$Range(\mathcal{C}) \cup \{250, 500, 1000\}$	PH threshold

Table 1: AOS Hyper-parameters and value range

The hyper-parameters involved in the presented AOS schemes are summarized in Table 1. In the absence of any theoretical guidance, how to tune the hyper-parameter values depending on the optimization problem at hand defines yet another optimization problem. An exhaustive exploration of the (discretized) search space, a.k.a complete factorial design of experiment, is quite computationally expensive; it requires one to compute the performance associated to every possible hyper-parameter setting, averaged over some independent M runs for the sake of significance.

Therefore a racing method is used for discarding as early as possible the unpromising settings. Formally, the Friedman's two-way Analysis of Variances by the ranks is used as statistical test [4] to determine whether a setting is significantly worse than the current best one. F-racing proceeds by repeating an "execution/comparison/elimination" cycle until there is a single candidate setting left, or all remaining candidates have been run M times. The criterion used for Friedman's test is the number of generations needed to reach the optimal solution, or 25000 if the optimum is not found before that many generations. The number M of runs is set to 50, due to the high variance of the results (e.g., as compared to [4]); for the same reason, a minimum number

of 11 runs for each setting was launched before elimination starts; elimination is based on Friedman's test with 95% confidence. Overall, F-racing reduces the computational cost by 50% compared to the complete factorial DoE, bringing a significantly lesser gain than in [4].

Table 2 displays the result of the F-race for each AOS setting, providing some insight into their sensitivity and robustness. Interestingly, the Dynamic Multi-Armed Bandit AOS retains many good hyper-parameter settings until the end; although this increases the computational load as less settings are eliminated, it suggests that Dynamic Multi-Armed Bandit is actually more robust w.r.t. hyper-parameter values than other AOS settings. Quite the opposite, AP shows a fast convergence toward a unique hyper-parameter setting in most of the cases, finding the optimal p_{min} value as $= .2$ − which actually corresponds to a uniform selection among the five variation operators considered.

4.4 Experimental Conditions

In the Long K-Path problems, k is set to 3; the bit-string length ℓ ranges in $\{43, 49, 55, 61\}$. Most results will however be presented on the instance of size 49, an arbitrary choice as all instances seem to give approximately the same results with respect to AOS performance (except the largest one that proved too difficult). Five mutation operators are considered: $1, 3$ and 5-bit mutations (mutating exactly 1, 3 or 5 bits uniformly selected in the bitstring); the $1/\ell$ bit-flip (every bit is flipped with probability $1/\ell$, ℓ being the bit-string length), and lastly the k/ℓ bit-flip, supposedly the optimal single operator on Long K-Path problems [12].

All AOS schemes are embedded into a $(1+50)$-EA [2]; the hyper-parameters are set to their optimal values according to the racing results, that can be found in Table 3. Their results are assessed comparatively to two reference strategies: the *Naive* strategy uniformly selects one operator among the available ones; and the *Optimal* strategy only selects the optimal operator at each point of the path (section 4.2).

The Royal Road problems involve the default values proposed by Holland [16], i.e., $k = 4$, $b = 8$, $g = 7$, $m = 4$, $v = 0.02$, $u^* = 1.0$ and $u = 0.3$. Each one of the 2^k regions involving $(b + g)$ bits, the dimension of the search space is 240 bits[3].

The AOS schemes are embedded into a $(100, 100)$-EA with weak elitism (the best individual is never lost) with a tournament size 2. Five variation operators are considered: 1-point, 2-point, 4-point and uniform crossover, plus a disruptive mutation operator that flips each bit with a probability of $1/30$, thus flipping 8 bits (and hence possibly one block) on average. Each crossover is followed by a $1/100$ bit-flip mutation. Contrasting with Long K-Path, the optimal operator cannot be easily accessed, as the fitness landscape includes many paths toward the optimal solution. The AOS schemes (likewise parametrized after the optimal setting, see the results in Table 4) are assessed comparatively to the *Naive* reference strategy, uniformly selecting one operator among the available ones.

---

[2]50 offspring are created from the single parent; next parent is the best out of the offspring and the current parent.

[3]Notably, the fully deceptive ($m = 1$) and not deceptive ($m = 7$) Royal Road were also investigated. The former problem was however found too difficult to be solved within 100,000 generations whereas the latter was too easy.

Problem	Reward	DMAB		MAB		AP		PM	
		Exps.	Configs.	Exps.	Configs.	Exps.	Configs.	Exps.	Configs.
RR's $m^*=4$	XAbs	9352/23800	40/476	1197/1400	21/28	2304/6400	17/128	1156/1600	20/32
	XNorm	11623/23800	108/476	1124/1400	20/28	2518/6400	18/128	1236/1600	11/32
	AvgAbs	11650/23800	89/476	1092/1400	14/28	1478/6400	1/128	898/1600	14/32
	AvgNorm	8790/23800	51/476	821/1400	11/28	1478/6400	1/128	400/1600	1/32
3-Path(49)	XAbs	6572/23800	16/476	1088/1400	20/28	1819/6400	1/128	841/1600	11/32
	XNorm	10397/23800	72/476	437/1400	1/28	1770/6400	1/128	840/1600	10/32
	AvgAbs	17878/23800	261/476	971/1400	17/28	2666/6400	1/128	1564/1600	29/32
	AvgNorm	12449/23800	78/476	1088/1400	20/28	1608/6400	1/128	1600/1600	32/32

Table 2: Results of F-racing procedure, comparing all selection schemes over 5 variation operators, on the Royal Road and Long K-Path problems. *Exps* columns indicate the number of runs which were not pruned (not significantly worse than the best one) out of the complete factorial DoE. Columns *Configs* give the number of optimal hyper-parameter settings after the F-racing.
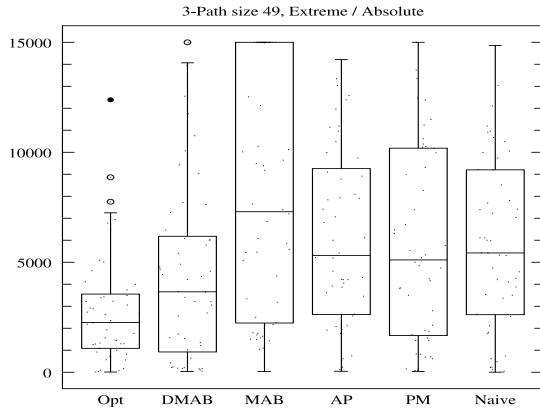

Figure 2: AOS performances on Long K-Path, using the Extreme/Absolute *Credit Assignment*. From left to right: optimal strategy, Dynamic Multi-Armed Bandit, Multi-Armed Bandit, AP, PM and the naive uniform strategy.

Figure 3: AOS performances on Long K-Path, using the Dynamic Multi-Armed Bandit *Operator Selection*. From left to right: optimal strategy, ExtremeAbsolute, ExtremeNormalized, AverageAbsolute and AverageNormalized *Credit Assignment*, and the naive uniform strategy.

Such different settings corresponds to different characteristics of the operators under scrutiny here: the Long K-Path problem is better solved by mutations, and crossover is of poor help: a few trials with the (100,100)-EA scheme gave very poor results indeed. The situation is even more constrained with the crossover operators, as they do not make any sense with a population of size 1, as in the (1+50)-EA.

5. RESULTS AND DISCUSSION

For each problem and AOS scheme, the results will be assessed through the number of generations needed to reach the optimal solution. Because of the high dispersion of the results, averages and standard deviations are not meaningful, and best and median results will be presented, grouped in Tables 3 and 4. However, those numbers are clearly insufficient, and some empirical distributions of the results will also be presented as boxplots. Furthermore, all differences between AOS schemes will be validated using both unsigned Wilcoxon rank sum, and Kolmogorov-Smirnov nonparametric tests, (termed W and KS in the following).

Regarding the Long K-Path problem, the best AOS scheme is Dynamic Multi-Armed Bandit with ExtremeAbsoluteReward which outperforms all other selection rules with Ex-
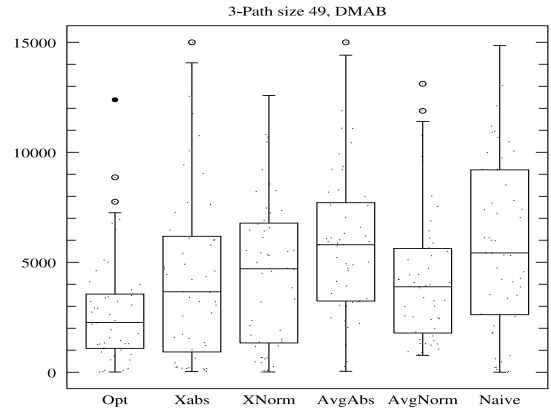
tremeAbsoluteReward (Fig. 2; the difference is significant w.r.t. MAB for W 99% and KS 95%, AP for W 95%, and the naive approach for W 90%), as well as all other rewards with Dynamic Multi-Armed Bandit selection (Fig. 3; the difference is significant w.r.t. AverageAbsoluteReward for both tests at 95% confidence level). Moreover, this best AOS scheme is not statistically different from the optimal Long K-Path strategy for both tests at 99% confidence level. The high dispersion of the results is explained as some runs happen to discover shortcuts at the beginning of evolution, after the initial bump in the performance distribution noted in [12]. This high dispersion also justifies *a posteriori* the choice of a large number of runs ($M = 50$) in the F-Race (section 4.3).

The situation is similar in the Royal Road problem: Dynamic Multi-Armed Bandit with ExtremeAbsoluteReward is better than all other selection rules with the same *Credit Assignment* (Fig. 4; the difference is significant w.r.t. AP for W 90%, PM for both tests at 90%, and the naive approach for both tests at 99%). The extreme value-based rewards, either absolute or normalized (XAbs and XNorm) significantly
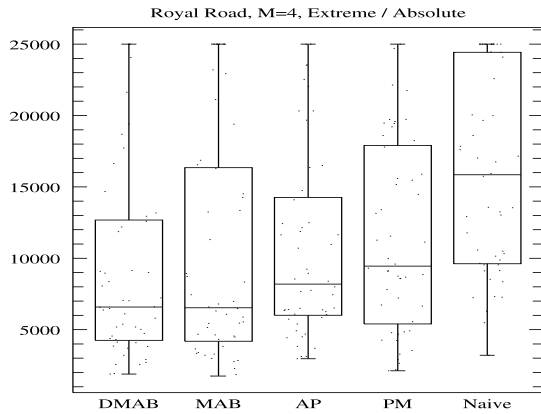
Figure 4: AOS performances on Royal Road, using the Extreme/Absolute *Credit Assignment*. From left to right: Dynamic Multi-Armed Bandit, Multi-Armed Bandit, AP, PM and the naive strategy.
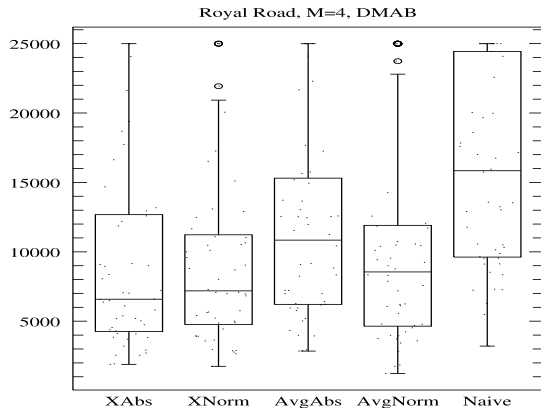

Figure 5: AOS performances on Royal Road, using the Dynamic Multi-Armed Bandit *Operator Selection*. From left to right: ExtremeAbsolute, ExtremeNormalized, AverageAbsolute and AverageNormalized *Credit Assignment*, and the naive uniform strategy.

outperform the average-value based ones for both Dynamic Multi-Armed Bandit (Fig. 5) and Multi-Armed Bandit (not shown), at 95% confidence level.

6. DISCUSSION AND PERSPECTIVES

The systematic assessment of diverse Adaptive Operator Selection schemes on two challenging although artificial fitness landscapes supports some former claims, while bringing some unexpected results. A first claim, fully supported by the empirical evidence in the limit of the considered problem instances, concerns the relevance of extreme-value based rewards, significantly outperforming average-based value rewards on all considered problem instances. This result confirms the importance of extreme events for the success of evolution.

A second experimental finding concerns the good general performance of *Dynamic Multi-Armed Bandit* (although slightly outperformed by Multi-Armed Bandit on the Royal

Reward	DMAB	MAB	AP	PM
XAbs	38 - 3641 $\mathcal{C}50\gamma10*$	33 - 7201 $\mathcal{C}100$	43 - 5207 $P_m.2*$	38 - 5003 $P_m.1\alpha.1*$
XNorm	14 - 4680 $\mathcal{C}10\gamma.1$	23 - 4851 $\mathcal{C}100$	43 - 5207 $P_m.2*$	43 - 5198 $P_m.2$
AvgAbs	45 - 5794 $\mathcal{C}.5\gamma500$	736 - 3851 $\mathcal{C}10$	43 - 5207 $P_m.2*$	65 - 4592 $P_m0\alpha.3$
AvgNorm	770 - 3840 $\mathcal{C}.005\gamma10$	736 - 3304 $\mathcal{C}.001$	43 - 5207 $P_m.2*$	4 - 4720 $P_m.05\alpha.3$

Table 3: Long K-Path, $k = 3, \ell = 49$. For each AOS scheme is reported the best − median result; the optimal hyper-parameter setting after F-Racing is indicated below, where $*$ stands for $\mathcal{W} = 500$ (50 otherwise).

DMAB	MAB	AP	PM
1889 - 6572 $\mathcal{C}.0005\gamma.0001$	1751 - 6472 $\mathcal{C}.1$	2963 - 7994 $P_m.05\alpha.9\beta.3$	2121 - 9304 $P_m0\alpha.1$
1751 - 7086 $\mathcal{C}.001\gamma25$	2555 - 7659 $\mathcal{C}5$	1293 - 6823 $P_m0\alpha.1\beta.9$	2583 - 9561 $P_m0\alpha.9$
2845 - 10429 $\mathcal{C}.0001\gamma.1$	2845 - 11252 $\mathcal{C}.0001$	2948 - 13508 $P_m.2*$	3081 - 9668 $P_m0\alpha.1$
1230 - 8338 $\mathcal{C}.0001\gamma100$	1230 - 8338 $\mathcal{C}.0001$	2948 - 13508 $P_m.2*$	2594 - 12441 $P_m.1\alpha.1$

Table 4: Royal Road (5 operators). For each AOS is reported the best − median result; the optimal hyper-parameter setting after F-Racing is indicated below, where $*$ stands for $\mathcal{W} = 500$ (50 otherwise). Rows are as in Table 3.

Road), together with its stability w.r.t. the hyper-parameters of AOS (Table 2) and its lesser variance compared to Multi-Armed Bandit (see for instance Fig. 2, and, to a lesser extent, Fig. 4); the same behavior is observed in many others settings (not shown). These results contrast with *Adaptive Pursuit*, whose best setting in fact implement a uniform search ($p_{min} = .2$ for the selection among 5 operators).

Overall, the relevance of AOS schemes can be argued from the fact that they significantly improve on fixed operator selection strategies[4]. While one might object that AOS schemes involve hyper-parameters, and thus also require some preliminary parameter tuning phase, it must be observed that the number of hyper-parameters does not depend on the number of variation operators. Furthermore, as suggested by Table 2, the question of finding good hyper-parameters might be less peaked and thus less critical than the question of finding good parameters.

Further research is concerned with devising artificial problems with known optimal AOS strategies, in order to identify more precisely the strengths and limitations of the AOS schemes under investigation. Another issue is to self-adapt the change detection threshold γ in *Dynamic Multi-Armed Bandit*, to account for the fact that the fitness improvements vary along the different evolution stages. Specifically, the bounding of the total number of restarts allowed along evolution will be investigated.

---

[4]With however one exception: the 4-point crossover (followed by a 1% bit-flip mutation) is shown to be the best single operator on Royal Road; it needs 6,500 generations in average to find the optimum, which is not statistically different from the ExtremeAbsoluteReward Dynamic Multi-Armed Bandit performance for both tests at 95%.

7. REFERENCES

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2/3):235–256, 2002.

[2] H. J. C. Barbosa and A. M. Sá. On adaptive operator probabilities in real coded genetic algorithms. In *Proc. Intl. Conf. Chilean Computer Science Society*, 2000.

[3] T. Bartz-Beielstein, C. Lasarczyk, and M. Preuss. Sequential parameter optimization. In *Proc. CEC*, pages 773– 780. IEEE Press, 2005.

[4] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In *Proc. GECCO*, pages 11–18. Morgan Kaufmann, 2002.

[5] L. Da Costa, A. Fialho, M. Schoenauer, and M. Sebag. Adaptive operator selection with dynamic multi-armed bandits. In *Proc. GECCO*, pages 913–920. ACM Press, 2008.

[6] L. Davis. Adapting operator probabilities in genetic algorithms. In *Proc. ICGA*, pages 61–69. Morgan Kaufmann, 1989.

[7] K. De Jong. Parameter Setting in EAs: a 30 Year Perspective. In F. Lobo, C. Lima, and Z. Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms*, chapter 1, pages 1–18. Springer Verlag, 2007.

[8] A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in Evolutionary Algorithms. *IEEE Trans. on Evolutionary Computation*, 3(2):124, 1999.

[9] A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J. E. Smith. Parameter Control in Evolutionary Algorithms. In F. Lobo, C. Lima, and Z. Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms*, chapter 2, pages 19–46. Springer Verlag, 2007.

[10] A. Fialho, L. Da Costa, M. Schoenauer, and M. Sebag. Extreme value based adaptive operator selection. In *Proc. PPSN X*, pages 175–184. Springer, 2008.

[11] A. Fialho, L. DaCosta, M. Schoenauer, and M. Sebag. Dynamic multi-armed bandits and extreme value-based rewards for adaptive operator selection. In *Proc. LION-3*. Springer-Verlag, 2009.

[12] J. Garnier and L. Kallel. Statistical distribution of the convergence time of evolutionary algorithms for long-path problems. *IEEE Trans. Evolutionary Computation*, 4(1):16–30, 2000.

[13] J. J. Grefenstette. Deception considered harmful. In *Foundations of Genetic Algorithms 2*, pages 75–91. Morgan Kaufmann, 1993.

[14] C. Hartland, N. Baskiotis, S. Gelly, O. Teytaud, and M. Sebag. Change point detection and meta-bandits for online learning in dynamic environments. In *Proc. CAp'07*, July 2007.

[15] D. Hinkley. Inference about the change point from cumulative sum-tests. *Biometrika*, 58(3):509–523, 1970.

[16] J. H. Holland. Royal road functions. In *Internet Genetic Algorithms Digest 7:22*. Massachusetts Institute of Technology, 1993.

[17] J. Horn, D. E. Goldberg, and K. Deb. Long path problems. In *Proc. PPSN III*, pages 149–158. Springer-Verlag, 1994.

[18] F. Hutter, Y. Hamadi, H. H. Hoos, and K. Leyton-Brown. Performance prediction and automated tuning of randomized and parametric algorithms. In *Proc. CP 2006*, number 4204 in LNCS, pages 213–228. Springer Verlag, 2006.

[19] T. Jones. A description of Holland's Royal Road. *Evolutionary Computation*, 2(4):409–415, 1994.

[20] B. A. Julstrom. What have you done for me lately? Adapting Operator Probabilities in a Steady-State Genetic Algorithms. In *Proc. ICGA*, pages 81–87. Morgan Kaufmann, 1995.

[21] T. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6:4–22, 1985.

[22] F. Lobo and D. Goldberg. Decision making in a hybrid genetic algorithm. In *Proc. ICEC'97*, pages 121–125. IEEE Press, 1997.

[23] F. Lobo, C. Lima, and Z. Michalewicz. *Parameter Setting in Evolutionary Algorithms*. IEEE Press, Piscataway, NJ, 2005.

[24] V. Maniezzo, R. Battiti, and J.-P. Watson, editors. *Learning and Intelligent Optimization*, Foundations of Computing. Springer Verlag, 2008.

[25] J. Maturana and F. Saubion. A compass to guide genetic algorithms. In *Proc. PPSN X*, pages 256–265. Springer, 2008.

[26] M. Mitchell, S. Forrest, and J. H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proc. ECAL*, pages 245–254, Cambridge, MA, 1992.

[27] V. Nannen and A. E. Eiben. Relevance estimation and value calibration of evolutionary algorithm parameters. In *Proc. IJCAI'07*, pages 975–980, Hyderabad, India, 2007.

[28] R. J. Quick, V. J. Rayward-Smith, and G. D. Smith. The royal road functions: Description, intent and experimentation. In *Selected Papers from AISB Workshop on Evolutionary Computing*, pages 223–235. Springer-Verlag, 1996.

[29] G. Rudolph. *Convergence Properties of Evolutionary Algorithms*. Verlag Dr. Kovac, 1997.

[30] D. Thierens. An adaptive pursuit strategy for allocating operator probabilities. In H.-G. Beyer, editor, *Proc. GECCO'05*, pages 1539–1546. ACM Press, 2005.

[31] D. Thierens. Adaptive Strategies for Operator Allocation. In F. Lobo, C. Lima, and Z. Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms*, pages 77–90. Springer Verlag, 2007.

[32] A. Tuson and P. Ross. Adapting operator settings in genetic algorithms. *Evolutionary Computation*, 6(2):161–184, 1998.

[33] J. M. Whitacre, T. Q. Pham, and R. A. Sarker. Use of statistical outlier detection method in adaptive evolutionary algorithms. In M. Cattolico, editor, *Proc. GECCO'06*, pages 1345–1352. ACM, 2006.

[34] B. Yuan and M. Gallagher. Statistical racing techniques for improved empirical evaluation of evolutionary algorithms. In Xin Yao et al., editor, *PPSN VIII*, pages 172–181. LNCS 3242, Springer Verlag, 2004.