Applied Soft Computing 35 (2015) 359–372

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

An experimental study of adaptive control for evolutionary algorithms

Giacomo di Tollo [a,*], Frédéric Lardeux [b], Jorge Maturana [c], Frédéric Saubion [b]

[a] *Dipartimento di Management, Universitá Ca Foscari, Venezia, Italy*
[b] *LERIA, University of Angers, France*
[c] *Instituto de Informática, Universidad Austral de Chile, Chile*

ARTICLE INFO

Article history:
Received 17 June 2013
Received in revised form 4 March 2015
Accepted 16 June 2015
Available online 2 July 2015

Keywords:
Algorithms
Design experimentation
Measurement
Performance
Evolutionary algorithms
Adaptive operator selection

ABSTRACT

In this paper, we investigate how adaptive operator selection techniques are able to efficiently manage the balance between exploration and exploitation in an evolutionary algorithm, when solving combinatorial optimization problems. We introduce new high level reactive search strategies based on a generic algorithm's controller that is able to schedule the basic variation operators of the evolutionary algorithm, according to the observed state of the search. Our experiments on SAT instances show that reactive search strategies improve the performance of the solving algorithm.

1. Introduction

During the past decades, Evolutionary Algorithms (EAs) [14,22,27] have been successfully applied to many optimization problems. From a high level point of view, EAs manage a set of potential solutions of a problem – a population of individuals according to the evolutionary metaphor. The population is progressively modified by variation operators in order to converge to an optimal solution with regards to a fitness function, which evaluates the quality of the individuals. Two well-known concepts are commonly used to describe the behavior of a EA: *exploitation* – which reflects the ability of the algorithm to converge to an optimum – and *exploration* – which ensures that the algorithm is able to visit sufficiently sparse areas of the search space. The balance between exploration and exploitation (referred to as EvE) is widely recognized as a key issue of the overall search performance. This balance often relies on the adjustment of several parameters, such as the size of the population and the application rates of the different operators.

Significant progress has been achieved in parameter setting [33]. Following the taxonomy proposed by [15], *tuning* techniques adjust the parameters of the algorithm before the run, and *control* techniques modify the behavior of the algorithm during the search process. Efficient *tuning* methods use statistical tools such as racing techniques [4] or meta-algorithms that explore the parameters' space (e.g., ParamILS [29] or Revac [42]). *Control* techniques have also been proposed in order to provide adaptive or self-adaptive EAs [13].

In this paper, in the context of control, we focus on the operator selection problem, i.e., given a set of available operators, how to select the operator to apply for the next iteration of the evolutionary process. To this aim, we use an Adaptive Operator Selection (AOS) approach [36] using a *control* point of view in order to dynamically adjust the EvE balance and improve search efficiency. The control of the EvE balance has been only partially investigated so far: most works focus on exploitation and use the quality of the population as a unique criterion to guide the search [17,24,52], and the few works that use several criteria [39] keep the EvE balance fixed. Since it has been shown that an efficient algorithm requires different parameter values during the search for achieving better results [32], the EvE balance should be dynamically controlled.

The purpose of our work is twofold. Firstly, we investigate the management of dynamic control strategies by using the framework proposed by [38] to implement a generic *controller*.[1] This controller must thus identify the suitable operators at each step of the search in order to achieve the required EvE balance, which may change

* Corresponding author. Tel.: +39 041 2346927.
E-mail address: giacomo.ditollo@unive.it (G. di Tollo).

[1] In this paper, we call *controller*, the complete architecture that allows us to perform adaptive operator selection.

http://dx.doi.org/10.1016/j.asoc.2015.06.016
1568-4946/

dynamically according to a given control strategy. Then we want to assess the impact of dynamic control on the performance of the algorithm. Our experimental methodology is organized as follows:

1. **Evaluating the operators management of the controller**:
   - by assessing whether the controller is able to identify the required operators in presence of non-efficient operators, i.e., in presence of noisy operators;
   - by checking whether the controller is able to manage a high level search policy that modifies the desired EvE balance during the search.
2. **Evaluating the solving performances**:
   - by checking whether the controlled EA is able to solve problems efficiently with regards to existing algorithms on a sufficiently large set of problems.

We want to point out that Maturana et al. [38] have proposed a controller that maintains a fixed desired compromise amongst criteria, to check whether the operators application fits the desired compromise. In our work we extend this approach by implementing a controller in which the desired compromise may change over time, and by designing high level search strategies that adjust this compromise: these new strategies allow us to improve the EA's performances. Furthermore we have devised a new reactive strategy referred to as *REACTIVEMOVING* in Section 5.2 that achieves very good performances in terms of solution quality, which are comparable –when not better– than those obtained by a specific state-of-art solver on large instances of the satisfiability problem.

Organization of the paper: we recall the main literature on the topic in Section 2 before describing the controller in Section 3. Then, we introduce the experimental setting in Section 4 before discussing results obtained through the experimental phase: Section 5 focuses on the management of the operators, and solving performance is investigated in Section 6.

2. Related works

Using an EA requires to define its basic structural parameters (components) and to set the values of its behavioral parameters. Parameter setting is thus an important challenge for building efficient and robust EAs; more details can be found in [33,25]. Concerning structural parameters, automated tuning techniques [28] can be used as tools for selecting the initial configuration of the algorithm. The configuration and the discovery of new heuristics from building blocks is also addressed by the concept of hyper-heuristics [6]. We may also mention self-adaptive operators that mainly consists in encoding directly the parameters of the operator in the individuals. This approach also allows the algorithm to dynamically manage the EvE balance and has been successfully applied for solving combinatorial and continuous optimization problems [43,49,50,56]. An adaptive management of the operators, which dynamically adds and discards operators during the search, has been proposed by Maturana et al. [38].

As mentioned in introduction, we focus on Adaptive Operator Selection, i.e., the choice of the best policy for selecting the operators during the search and we recall now more precisely the different methods that have been proposed to this aim.

Let us consider n operators: the probability of selecting operator op_i at time t is $s_i(t)$. In a static setting, the probability of selecting op_i (for each i) is fixed over time (i.e., $s_i(t)=s_t(t')$, for any t and $t' \in [1, t_{max}]$), and can be determined by an automated tuning process. Contrarily to a static tuning of the operator application rates, adaptive operator selection consists in selecting the next operator to apply at time $t+1$ by adapting the selection probability during the search according to the performance of the operators. Let us consider an estimated utility $u_i(t)$ of operator op_i at time t. This utility of the operators has to be re-evaluated at each time, classically using a formula $u_i(t+1)=(1-\alpha)u_i(t)+\alpha r_i$ where r_i is the reward associated to the application of operator op_i (immediate performance) and α is a coefficient that controls the balance between past and immediate performance, as done in classic reinforcement learning techniques [47]. Note that α can be set to $\frac{1}{t+1}$ in order to compute the mean value. A classic selection mechanism is the probability matching selection rule (PM) and can be formulated as:

$$s_i(t+1) = p_{\min} + (1 - n \times p_{\min})\frac{u_i(t+1)}{\Sigma_{k=1}^{n} u_k(t+1)} \tag{1}$$

where a non negative p_{min} ensures a non zero selection probability for all operators [23,34].

Thierens [51,52] has explored a *winner-take-all* strategy for AOS, based on the quality (or fitness) of the population:

$$\begin{cases} s_{i^*}(t+1) = s_{i^*}(t) + \beta(p_{\max} - s_{i^*}(t)) \\ s_i(t+1) = s_i(t) + \beta(p_{\min} - s_i(t)) \end{cases} \tag{2}$$

where $i^* = \text{argmax}\{u_i(t),\ i=1..n\}$, $p_{max}=1-(n-1)p_{min}$ and β is a parameter to adjust balance of this winner-take-all strategy.

Alternatively, AOS can also be considered as a multi-armed bandit problem. The initial multi-armed bandit problem was introduced in the context of the experiment design by Robbins [45]. It was formulated as the maximization of the total gain of a gambler who could make n tosses with two coins A and B with a gain of 1 for each head but nothing for tails. The biases of the coins are unknown. This problem is known as the *Two-armed Bandit* and has been extended to multi-armed bandit by Rodman [46]. Later, Auer [2] has proposed to use this problem to manage the compromise between exploration and exploitation in optimization algorithms. The *MAB* (Multi-Armed Bandit) algorithms that uses an UCB (Upper Confidence Bound) in order to approximate the expected benefit of an operator op_i at time t have been firstly extended to AOS by Da Costa et al. [11]: the operator that maximizes $Mab_i(t)$ in the following formula is selected:

$$Mab_i(t) = u_i(t) + C\sqrt{\frac{\log \sum_{j\in 1..n} n_j(t)}{n_i(t)}}, \tag{3}$$

where $r_i(t)$ is the reward of operator op_i at time t, $n_i(t)$ is the number of times operator op_i has been applied so far, and C is the scaling factor used to properly balance rewards and application frequency. In the initial multi-armed bandit problem, the expected gain of each possible action is supposed to be fixed over time. Therefore, in Da Costa et al. [11], the authors propose to use a Page-Hinkley test in order to detect a change of the operators' behavior, and thus to reset $r_i(t)$ and $n_i(t)$. In Fialho et al. [19], an improved technique has been proposed for comparing the respective performance of the operators.

Note that Eq. (3) uses $n_i(t)$ as a way to avoid forgetting less favorable operators, supposing that all operators were included from the start of the search. Indeed, if one of them were introduced to the eligible set in the middle of the search, it would be necessary to apply the operator several times to catch up with respect to the other ones. This would imply a waste of time and an eventual degradation of the search if the new operator would not be suited to the current search requirements. In order to deal with this situation, a variation of the AOS was proposed in Maturana et al. [38] that considers *idle time* instead of the number of times an operator has been applied.

Focusing on the performance measures, Whitacre et al. [55] consider extreme values over a few applications of the operators, based on the idea that highly beneficial but rare events might be more beneficial than regular but smaller improvements.

Most works rely on quality as the only criterion used for control. Nevertheless, EA literature has constantly been concerned with