Information Sciences 258 (2014) 122–139

Contents lists available at ScienceDirect

Information Sciences

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

# Assisting in search heuristics selection through multidimensional supervised classification: A case study on software testing

Ramón Sagarna [*], Alexander Mendiburu, Iñaki Inza, Jose A. Lozano

*Intelligent Systems Group, University of the Basque Country, Paseo Manuel de Lardizábal 1, 20018 San Sebastián, Spain*

ARTICLE INFO

*Article history:*
Received 1 December 2012
Received in revised form 17 September 2013
Accepted 22 September 2013
Available online 17 October 2013

*Keywords:*
Algorithm selection
Multidimensional classification
Dominance relation
Search based software testing

ABSTRACT

A fundamental question in the field of approximation algorithms, for a given problem instance, is the selection of the best (or a suitable) algorithm with regard to some performance criteria. A practical strategy for facing this problem is the application of machine learning techniques. However, limited support has been given in the literature to the case of more than one performance criteria, which is the natural scenario for approximation algorithms. We propose multidimensional Bayesian network (mBN) classifiers as a relatively simple, yet well-principled, approach for helping to solve this problem. Precisely, we relax the algorithm selection decision problem into the elucidation of the nondominated subset of algorithms, which contains the best. This formulation can be used in different ways to elucidate the main problem, each of which can be tackled with an mBN classifier. Namely, we deal with two of them: the prediction of the whole nondominated set and whether an algorithm is nondominated or not. We illustrate the feasibility of the approach for real-life scenarios with a case study in the context of Search Based Software Test Data Generation (SBSTDG). A set of five SBSTDG generators is considered and the aim is to assist a hypothetical test engineer in elucidating good generators to fulfil the branch testing of a given programme.

## 1. Introduction

One of the all-pervasive and, from a pragmatical standpoint, fundamental questions in algorithmic problem solving is the identification of a suitable algorithm with regard to some performance criteria, for a given problem instance [46,50]. Specifically, for randomised search heuristics, many research efforts have been addressed towards this concern and two other challenging but required issues: the discovery of relevant problem characteristics and the measurement of algorithm behaviour.

The typical directions for problem characterisation (e.g. the study of phase transitions [12]) often demand the understanding of the characteristics and their relationship with algorithm behaviour. Also, the cost of their assessment should be considered, as they might be computationally inefficient [29]. As for the behaviour of randomised search heuristics, their stochasticity and sophistication often make it difficult to analyse measures of interest exactly [43,8], such as the worst-case expected runtime. Besides, an approximation algorithm is usually deemed good if it guarantees near-optimal solutions (in a well-defined sense) within polynomial time complexity, which implies the assessment of two measurements.

[*] Corresponding author. Tel.: +34 943 018070; fax: +34 943 015590.
*E-mail addresses:* ramon.sagarna@ehu.es (R. Sagarna), alexander.mendiburu@ehu.es (A. Mendiburu), inaki.inza@ehu.es (I. Inza), ja.lozano@ehu.es (J.A. Lozano).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.09.050

All these difficulties often become even more apparent when one needs to face a real-life domain, where any sensible analysis is likely to be constrained by resources (e.g. time, budget, expertise) or reliable problem information may be unavailable.

A strategy which is increasingly emerging as a practical response to algorithm selection is the application of machine learning techniques to induce sound models from empirical data [50,55,35,32]. These so-called *metalearning* approaches could be thought of as a compromise solution between classical empirical studies and theoretical analyses.

Although a number of metalearning approaches have already been developed (see [50]), the case of algorithm selection when more than one performance criteria is of interest has scarcely been addressed. We note this is a more realistic situation in practice, as approximation algorithms naturally demand their evaluation on solution quality as well as runtime. Moreover, it is not difficult to think of situations where additional behavioural aspects need to be considered; e.g. if computing in a shared machine, memory consumption might be a concern. In the present work, we approximate to the multicriteria case by means of a classifier which returns a recommendation of *good* algorithms and, so, it serves to assist the final practitioner in the decision. The classifier is built upon a reformulation of the problem in two stages. Firstly, the multicriteria decision problem implied by algorithm selection is relaxed into the elucidation of the nondominated set of algorithms. This set is given by a dominance relation which accounts for particular preference information of the decision maker. Secondly, a supervised classification problem is posed. Several variants may be of interest for the latter. We concentrate on two of them: predicting the whole nondominated set and whether a particular algorithm is nondominated or not. In order to face the corresponding classification problem we propose recently developed multidimensional Bayesian network classifiers [21,47,9]. On the one hand, our problem formulation fits naturally into the multidimensional classification scenario by associating one class variable to each algorithm. On the other hand, Bayesian networks offer two interesting aspects to this kind of problem: they provide an explicit factorisation of the joint probability distribution of features and class variables which facilitates the required inference, and, depending on the model structure, they may account for correlations among class variables.

In order to demonstrate the feasibility of our approach for real-life situations, we present a case study in the domain of software testing. This element from the life cycle of software is the main tool used in practice for discovering programme faults [16]. Thus, it plays a key role in the assessment and improvement of software reliability [6,16]. One of the principal concerns is the automatic generation of test inputs that fulfil an adequacy criterion. Precisely, we focus on branch testing, which demands a set of inputs exercising every branch in the source code of the programme. To accomplish this task, the field of Search Based Software Test Data Generation (SBSTDG) calls on the use of search heuristics during the selection of the appropriate test inputs [39,38,54]. The potential as an important real-life application domain and the encouraging results achieved have contributed to an increasing interest in SBSTDG over the last decade (see [39,28]). In spite of this engagement, there is a general lack of meaningful knowledge about the macroscopic behaviour of SBSTDG generators and their relations with the software testing problem. Besides the fact that most of the work to date has focused on the proposal of new methods [28], a SBSTDG generator is usually an advanced randomised heuristic for which a complete analysis is nontrivial. In the case study, we consider a set of five SBSTDG generators following a popular strategy, and aim at assisting the final practitioner in elucidating suitable generators from the set. We explain the application of our approach in the particular domain of numerical functions in C [44]. When the amount of data is relatively contained, it is sensible to choose classifiers with the popular naive Bayes-like model structure. We present an empirical comparison of classifiers based on multidimensional and unidimensional versions of this structure on the two problems pointed out above: predicting the nondominated set of generators and whether or not a particular generator is nondominated. The results indicate a superiority of the multidimensional classifier version. Finally, we discuss the utility of the approach upon a set of seven C functions. Apart from the straight outcome of the classifier, we illustrate other clues the approach can offer to the practitioner. Namely, it can serve to spot programmes for which a generator performs worse than others (e.g. random testing), or it can point at the programme features (software complexity metrics, in our case) which affect the relative performance differences among generators.

The rest of the paper is organised as follows.

In the next section, we introduce the algorithm selection problem and related work. Our aim is to highlight the lack of a definitive and practical procedure for this problem. Also, we discuss the motivation for our work. In Section 3, we introduce multidimensional classifiers and explain our formulation proposal for algorithm selection. Regarding the former, we sketch the multidimensional classification problem and different solution strategies. We emphasise Bayesian network classifiers following a naive Bayes-like model. As for our approach, we explain how the algorithm selection problem with multiple criteria is relaxed and connected to the classifier. In Section 4, we outline the branch testing problem and the SBSTDG solution approach we have adopted in this work. We show the application of our proposal in this real-world problem through the case study in Section 5. There, we first describe the considered scenario and the steps to apply the approach. Then, we present the empirical comparison of the classifiers and, finally, we demonstrate their operation. We finish with concluding remarks and suggestions for future work.

## 2. Scope and motivation

We first outline the algorithm selection problem, in terms of randomised search heuristics, and then we overview different solution strategies. Building upon these, we can discuss the motivation for our proposal.