Achieving Replicability: Is there life for our experiments after publication?

Jose Antonio Parejo, Sergio Segura Rueda, and Antonio Ruiz-Cortés*

Department of Computing Languages and Systems. Univeristy of Sevilla

Abstract Metaheuristics are algorithmic schemes that ease the derivation of novel algorithms to solve optimization problems. These algorithms are typically approximated and stochastic, leading to the preeminence of experimentation as the mean of supporting claims in research and applications. However, the huge number of variants and parameters of most metaheuristics, the ambiguity of natural language used in papers, and the lack of widely accepted reporting standards threatens the replicability of those experiments. This problem, that has been identified in the literature by several authors, significantly hinders the construction of a complete and cohesive body of knowledge on the behavior of metaheuristics. This paper proposes a set of minimum information guidelines for reporting metaheuristic experiments, and an experiment description language that supports the meeting of those guidelines. By using this language, metaheuristic optimization experiments are described in a tool-independent and unambiguous way, while maintaining readability and succinctness. Those contributions pave the way for replication using different problem instances and parameters, bringing a new life to metaheuristic experiments after publication.

1 Introduction

An experiment is replicable when its results can be verified and/or clarified through conducting another experiment. This process can be preformed either following the same experimental procedure in similar conditions (named repetitions when the conditions are exactly the same and replications when they are slightly different), or through a different procedure aimed to verify similar hypotheses about the same phenomenon (named conceptual replications).

The quote of [1]: "*The use of precise, repeatable experiments is the hallmark of a mature scientific or engineering discipline*", clashes violently with the statements of some authors in the area of metaheuristic optimization[2]: "*Verifying results found in the literature is in practice almost impossible*", "*running a reportedly good algorithm on your own data is an extremely difficult task*", "the details presented in a typical paper are insufficient to ensure that one would implement the same algorithm".

* This work was partially supported by the EU Commission (FEDER), the Spanish and the Andalusian R&D&I programmes grants SETI (TIN2009-07366), TAPAS (TIN2012-32273), ISABEL (TIC-2533) and THEOS (TIC-5906).

The replicability of experiments using metaheurisitcs is therefore a problem. Since most of the research on metaheuristics relies on empirical validation and experimentation[3], this problem becomes of paramount importance in our area. It is therfore and important problem that significantly hinders the construction of a solid and cohesive body of knowledge on the behavior of metaheuristics.

Experimental replicability is a problem due huge number of variants, parameters and possible customizations of the algorithms, its own stochastic nature, and the lack of an widely accepted scheme of experimental reporting (in contrast to those used in the natural sciences [4]).

Minimum information guidelines have enabled the reuse of existing work and reproducibility of the experimental process in different research areas such as biology and simulation. However, guidelines are not enough. Several set of guidelines has been published by different authors (see for instance [5,6,7,8]), but standard experimental reporting in the area has not improved to the levels of other scientific disciplines [9]. The proposal of this paper for this issue is the creation of: i) experiment description languages (EDLs) that vertebrate the description and reporting of experiments according to the guidelines; and ii) tools that automatically check that the experiment descriptions expressed on those languages are following the guidelines actually.

The remainder of this paper is structured as follows: Section 2 describes our guidelines of minimum information to be reported about metaheuritic optimiztation experiments. Next, 3 describes an experiment description language whose use guarantees the meeting of most of the guidelines defined in previous section. Section 4 presents the tools developed to ease the creation and manipulation of experimental descriptions with the language. Section 5 reports some validation performed on our proposal to ensuer its suitability. Finally, 7 provides some conclusions and points out future work.

2 Minimum Information about Metaheuristic Optimization Experiments

Minimum Information guidelines have enabled the reuse of existing work and reproducibility of the experimental process in different research areas such as biology and simulation. Authors propose a set of guidelines about Minimum Information about Metaheuristic Optimization Experiments (MIaMOE). In so doing, the guidelines defined for simulation experiments in [10] have been adapted and extended to the specific context and characteristics of metaheuristic optimization experiments.

MIaMOE describes the minimum information that should be provided in order to properly the describe metaheuristic optimization experiments. The goal of MIaMOE is to enable experimental replicability. In order to achieve this goal, MIaMOE defines a set of requirements of the reported information. Thus, MIaMOE does not define neither operation procedures, nor specific experimental description formats, but a set of rules for describing experiments. The description of an experiment is MIaMOE-compliant if it meets all the rules defined by

MIaMOE. Specifically, MIaMOE defines the following rule: *For an experiement witn metaheuristic optimization techniques*

1. **Each optimization algorithm used in the experiment must be identified, fully described and accessible.**
   (a) Either the description of each optimization algorithm is complete and unambiguous, or their implementation is provided.
   (b) All parameter values, configuration settings, and governing conditions of the optimization algorithms used in the experiment are unambiguously described. Specifically, it should be described:
      i. The set of parameters, their domain, and the specific value used for experimentation.
      ii. Specific operators and variants of the optimization technique used.
      iii. The optimization process termination criteria
      iv. The random number generation algorithms used and the seeds used for experimentation.
   (c) If an algorithm is not implemented using a common implementation language and runtime, then the implementation should be provided and the implementation language and required run-times used must be clearly specified.
   (d) Any preprocessing step required before the execution of the optimization algorithm must be fully and unambiguously described.
2. **All information used to obtain the results and draw conclusions of the experiment must be provided.**
   (a) All the information used as input by the optimization algorithms must be provided, including problem instances data. If standard benchmarks of problem instances are used, the specific reference where the benchmark was proposed, the specific version (or year) of the benchmark used, and a download address must be provided.
   (b) The variables of each algorithm used to obtain results reported must be identified, including the specific step of algorithms in which results data are collected.
   (c) All the results datasets (raw numerical data and post-processing results) generated by the experiment must be described unambiguously, and a copy of the actual results should be provided.
   (d) If conclusions drawn from the experiment depend on the relation between different results datasets, the specific results datasets to be compared and the equations of the relation must be described unambiguously.
3. **A precise description of the experimentation procedure, its execution context, and other procedures used in the experiment must be provided.**
   (a) All the steps of the experimentation procedure must be clearly described, including:
      i. The total number and order of execution of algorithms on problem instances, or the criteria used to determine such executions and its order (for instance the experimental design).

ii. Information used by algorithms that is produced during the own experimentation process and its processing.
iii. All post-processing steps applied on the raw numerical data used to obtain each result dataset must be described in detail.

(b) The physical and computational equipment used during the experimentation process must be described. Those elements that are required for a replication should be specified explicitly, defining the minimum equipment requirements for replication.

Evaluating if the description of an experiment is MIaMOE-compliant just from a paper is a time-consuming task that must be performed manually. For instance, in a optimization technique comparison experiment, if authors do not provide the source code of the implementation of the techniques and the experimental procedure followed for conducting the experiment, then the description of the experimental procedure, the algorithms used in the paper, and the parametrization used in each experiment must be carefully scrutinized in order to determine MIaMOE-compliance. Moreover, if some optimization techniques compared in the experiment are described using references, the MIaMOE-compliance of the descriptions of those techniques in the referenced papers should be evaluated. Laboratory packages (also named lab-pack), are detailed reports with attached materials used by experimenters to provide all the relevant details about an study. Providing the source code, binaries and the input/output files of the experiments, along with the description of the computation environment in a lab-pack is the most straightforward way to achieve MIaMOE-compliance. However, providing source code and input files does imply being MIaMOE-compliant. For instance, if the algorithms are implemented in JAVA using the standard random number generator provided by the language and the seed, but the specific version of the Java Virtual Machine used for experiment conduction is not specified, then rule 1(b)iv is violated, since the random number generation algorithm is undetermined (it depends on the version of JAVA). Given the previous definitions, a MIaMOE-compliant experiment should be reproducible. However depending on the way researchers follow the guidelines, the investment and effort required for reproducing the experiments can vary dramatically. For instance, we suppose a MIaMOE-compliant a metaheuristics comparison experiment where researchers provide just the pseudo-code of their proposals and references to the publications where the alternative approaches are defined. The reproduction of such an experiment requires to implement each optimization technique based on the descriptions provided in the corresponding papers, which implies a significant amount of effort. Even when researchers provide the full source code, input files and specification of the environmental requirements of their experiments (such as libraries, run-times, specific platform for execution, etc.), the effort required to replicate an environment that meets those requirements and launch the experiment may be non-negligible. For instance, in [11] this effort is described as "considerable agony" even when performed by the own researchers in their lab. On the other hand, using the source code as a description of the experiments has some drawbacks. First, it makes experiment descriptions dependent on the

implementation technologies. Second it penalizes descriptions readability and understandability by non-programmers. Third it lowers the abstraction level at which the experiment is described, forcing to read and understand the specific implementation details of the experiment source in order to gather the information about the experiment. The need of language at an intermediate level of abstraction between natural-language descriptions found in papers and implementations could mitigate such problems is proposed in this paper. Such a language should be both human readable and machine processable. Having an experiment described using such a language in companion with the source code or binaries used in metaheuristic optimization experiments, could enable and ease experiment reproducitiblity, and even automate its replication with proper tools.

3 Metaheuristic Optimization Experiments Description Language (MOEDL)

MOEDL defines a set of elements for describing metaheuristic experiments that are of major relevance for ensuring MIaMOE compliance, such as the objective functions and problem instances used in the experiment, termination criteria used by each technique, the random number generation algorithm and seeds used, etc.

The structure of MOEDL is the result of an extensive analysis of a variety of experiments developed by authors [12,13,14], a careful study of the related literature and a process of successive refinements of the meta-model after applying it to different scenarios. Specifically, we have taken [4] as the main reference for general experiment descriptions and [9,8] for the specific details of metaheuristic optimization experiments. Additionally, we have evaluated other approaches (Section 6) for experiment description and the proprietary formats and classes used by the set of metaheuristic optimization frameworks assessed in [13].

MOEDL supports the succinct description of some common types of experiments in the area. Those experiment types are described below.

Figure 1 depicts the general structure of a MOEDL document. The schema depicted in Fig. 1 is informal and only provides a snapshot of the general stucture of MOEDL documents. In order to accurately describe the abstract syntax of MOEDL, its meta-model is described in [15]. That description is independent of the specific concrete syntax and serialization used for such as xml schemas and documents, plain text files, or graphical notations. For the purpose of this paper, due to space limitations the general structure of such documents in described, without diving deep in the details.

A `MetaheuristicOptimizationExperiment` in MOEDL contains the description of one or more problem types (with their corresponding set of problem instances), and one or more optimization techniques. Each problem instance and optimization technique, has a unique identifier in the context of the document, and their descriptions contains all the details about its parameters and

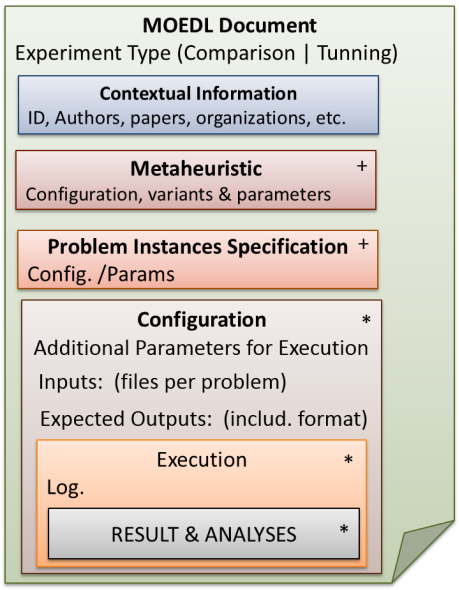

Figure 1. General Structure of MOEDL experimental descriptions

configuration (such as the specific objective function or its concrete parameter values in problem instances, and the specific variant used, the source file where it is implemented, and the parameter values used in the experiment for each metaheuristic). The detailed description of each optimization technique is performed through its `Configuration`, this element supports the description of simple parameters (such as the population size, or the probability of application of the mutation operator in Evolutionary Algorithms) and complex algorithmic variants. However, given a `Technique` and a `Configuration`, there is not guarantee that the technique configuration describes all the relevant parameter of the technique. As a consequence, in order to meet MIaMOE rules 1a and 1b, the provision of a lab-pack containing the source code of the technique and experimental procedure is mandatory. The single exception to this rule is the case when a software framework (such as ECJ or HeuristicLab) is used and the exact replications of the executions of the experiment can be performed automatically using the experiment description. Under those circumstances, the framework is able to enact the optimization technique based on the description provided, and consequently, the source code of the own framework and the experiment description fully describe the optimization techniques applied[1].

Additionally, a MOEDL description can contain a set of global configurations, that are used to specify configuration settings for the experiment as a whole.

[1] If custom variants are described in the framework-specific configuration of the technique and its binaries are provided in the experiment lab-pack. Then the source code of such custom variants should be provided also in the lab-pack

For instance, the termination criterion used for optimization can be specified for the whole experiment or for an specific optimization technique. In order to support the automation of the experiment, and ease replication using different termination criteria, this element defines an extension point in MOEDL, since its description could depend on the software used for implementing the metaheuristic. Either a global termination criterion is specified for the experiment, or each optimization technique should define its own termination criterion. It is worth noting that the use of different termination criteria in this kind of experiments could lead to bias in the comparison and wrong conclusions in the experiments.

The random number generation algorithm, seeds used for the experiment, and termination criterions can also be specified either for the whole experiment or for each specific optimization technique in particular in the configuration. This constraint enforces the fulfillment of the points in MIaMOE rule 1b.

In a techniques comparison experiment, the objective functions values of solutions obtained with several optimization techniques are compared for an specific set of problem instances.

In a Tuning experiment, the objective functions values of solutions obtained with several configurations of one single technique are compared for an specific set of problem instances.

Thus a `TechniqueParametrizationExperiment` contains a single metaheuristic optimization technique and a set of `ComplexParameter` definitions. These definitions specify the space of possible values of the parameter using a `Domain`, since finding its optimal value is the purpose of the experiment. This `Domain` is specified either by extension, i.e. providing the the whole set of possible values (e.g. 100, 200, and 300), or by intension, i.e. specifying a range for its the value and an algorithm or iteration mechanism to generate successive values of the parameter.

4 Supporting Tools & Libraries

An XML-schema encoding the language structure described in the previous section has been created. This schema provides an xml-based syntax for the language, which support the objectives of while maintaining human readability.[2] Moreover, a JAVA library for manipulating this kind of documents has been created. This library is named `JLibMOEDL`, and provides features such as the validation of MOEDL documents against the XML-schema and the semantic constraints exposed above, proving a detailed validation report, the load and saving of experiental descriptons as XML-files, and the representation of all the elements of the experimental description as Java objects. In this sense, `JLibMOEDL` eases the manipulation and automated generation of experimental descriptions with MOEDL.

[2] The XML schemas are available at `http://moses.us.es/schemas/moedl/v1/`.

5 Validation of the proposals

The suitabilty of the proposed language for describing metaheuristic optimization experiments has been validated using a case study. Specifically, two technique comparison experiments, and two parameter tuning experiments supporting the main conclusions of [16] has been described using MOEDL with its xml-based syntax. The language has been expressive enough to accurately describe all those experiments[3].

6 Related Work

Related work of the proposals made in this chapter present in the literature can be divided into two categories: (i) guidelines for reporting and support of the experimental replication, and (ii) languages for describing experiments.

Although recognized as an issue of paramount importance for progress of science and engineering, replication of experiments has been an extremely difficult task in our context [2,11,17].

In the context of experimentation with metaheuristics, specific guidelines has been proposed [18,6,19,7], and the didactic effort of the research community is still going on [17]. However, those proposals focus mainly on aiding the correct conduction of the experiments, on the use of appropriate of experimental designs for the hypothesis and populations addressed [8,20,21], and on the use of the correct statistical analysis tools for such designs [22]. These efforts have been successful, given the current widespread use of statistical tests for validating conclusions and of standard experiment designs in the area. Most of those proposals focus on specifying how researchers must do experimental research, but not how they must report and publish the materials in order to improve experimental replicability. Only a small subset those proposals described guidelines about reporting of experiments, with the aim of enabling experimental replicability (see for instance [6,7]). The main difference between the proposals in this paper and previous work is that our proposal of reporting guidelines (MIaMOE), focuses on *what* should be described (the minimum information required for replication) from an abstract perspective, and it delegates the tasks of specifying *how* to describe such information in the experimental description language (MOEDL), while providing means for checking that all required information is provided through software tooling.

Regarding experiment description languages, we can classify previous proposals in two categories. On the one hand, *Descriptive languages* support the description of the experiment but do not specify how to use such description, delegating the tasks of interpreting the description and using it for replication on researchers. An example of this kind of proposals is EDL [23], that provides a basic xml syntax to organize the description of the experiment. On the other hand, operational languages provide an description of the experiments at such an

[3] The XML files describing the experiments described using MOEDL are available at `http://moses.us.es/?page_id=16`

abstraction level that enables its automated replication, an example of this kind of experimental description language SED-ML [24]. The problem of operational languages is that they need to support very specific structures in order support the required details for replicability, as a consequence, the roposals for operational languages are specific of certains areas, for instance SED-ML is specific for simulation experiments. Another consequence of this need for specificity is that some languages have been created for support the specification of specific experimental artifacts and experimental protocols that of the whole experiment for ceitan areas, such as PEBL [25] that is intended for the creation of experiments in psychology.

Our proposal is operational, in the sense that enables the replication of experiment based on the description of experiments, but also contains some elements whose purpose is merely descriptive, such as the `Context`. Our proposal follows the general approach of SED-ML [24], but it adapts most of its concepts and introduce elements specific for describing experiments with metaheuristics that are MIaMOE-compliant.

7 Conclusions

In this paper, the problem of experimental replicability in the specific context of metaheuristic optimization experiments has been pointed out. Two contributions has been proposed to tacke this problem: (i) a set of guidelines about the information to, and (ii) an experiment description language that supports the meeting of those guidelines. Our proposals provide a balance, where metaheuristic optimization experiments are described in a tool-independent and unambiguous way, while maintaining readability and succinctness. Those contributions pave the way for replication using different problem instances and parameters, bringing a new life to metaheuristic experiments after publication.

References

1. Lewis, J.A., Henry, S.M., Kafura, D.G., Schulman, R.S.: On the relationship between the object-oriented paradigm and software reuse: An empirical investigation. Technical report, Blacksburg, VA, USA (1992)
2. Eiben, A., Jelasity, M.: A critical note on experimental research methodology in ec. Computational Intelligence, Proceedings of the World on Congress on **1** (2002) 582–587
3. Floudas, C.A., Pardalos, P.M.: Encyclopedia of Optimization. 2nd edn. (October 2008) ISBN 978-0-387-74760-6.
4. Gliner, J.A., Morgan, G.A., Leech, N.L.: Research methods in applied settings: an integrated approach to design and analysis (second edition). Volume 2nd. Tylor and Francis Group (2009)
5. Florian, M., Fox, B., Crowder, H., Dembo, R., Mulvey, J.: Reporting computational experience in operations research. Operantions Research **27** (1979) 7–10
6. Barr, R., Golden, B., Kelly, J., Resende, M., Stewart, W.: Designing and reporting on computational experiments with heuristic methods. Journal of Heuristics **1**(1) (1995) 9–32

7. Michalewicz, Z., Fogel, D.B.: How to Solve It: Modern Heuristics. Springer (2004)
8. Bartz-Beielstein, T.: Experimental Research in Evolutionary Computation: The New Experimentalism (Natural Computing Series). 1 edn. Springer (April 2006)
9. Bartz-Beielstein, T., Preuss, M.: Tuning and experimental analysis in evolutionary computation: what we still have wrong. In: GECCO (Companion). (2010) 2625–2646
10. Waltemath, D., Adams, R., Beard, D., Bergmann, F., Bhalla, U., Britten, R., Chelliah, V., Cooling, M., Cooper, J., Crampin, E., et al.: Minimum information about a simulation experiment (miase). PLoS computational biology **7**(4) (2011) e1001122
11. Schwab, M., Karrenbach, N., Claerbout, J.: Making scientific computations reproducible. Computing in Science Engineering **2**(6) (nov/dec 2000) 61 –67
12. Parejo, J.A., Fernández, P., Ruiz-Cortés, A.: Qos-aware services composition using tabu search and hybrid genetic algorithms. In: Talleres de las JISBD. IADIS'08. (2008)
13. Parejo, J.A., Lozano, S., Ruiz-Cortès, A., Fernandez, P.: Metaheuristic optimization frameworks: A survey and benchmarking. Soft Computing (2011)
14. Segura, S., Parejo, J.A., Hierons, R.M., Benavides, D., Ruiz-Cortés, A.: Ethom: An evolutionary algorithm for optimized feature models generation (v. 1.2). Technical report (July 2012)
15. Parejo, J.A.: MOSES: A Software Ecosystem for Metaheuristic Optimization. PhD thesis, University of Sevilla (2013) Early Draft Version available at `https://dl.dropboxusercontent.com/u/1279737/tesis/Prelectura/Dissertation.pdf`.
16. Parejo, J.A., Fernández, P., Ruiz-Cortés, A.: On parameter selection and problem instances generation for qos-aware binding using grasp and path-relinking. Research Report 2011-4, ETSII. Av. Reina Mercedes s/n. 41012. Sevilla. Spain (2011)
17. Bartz-Beielstein, T., Preuß, M., Zaefferer, M.: Statistical analysis of optimization algorithms with r. In Soule, T., Moore, J.H., eds.: GECCO (Companion), ACM (2012) 1259–1286
18. Hooker, J.: Testing heuristics: We have it all wrong. Journal of Heuristics **1**(1) (1995) 33–42
19. Coffin, M., Saltzman, M.: Statistical analysis of computational tests of algorithms and heuristics. INFORMS Journal on Computing **12**(1) (2000) 24–44
20. Ridge, E., Kudenko, D.: Tuning the performance of the mmas heuristic. In Stützle, T., Birattari, M., H. Hoos, H., eds.: Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics. Volume 4638 of LNCS. Springer (2007) 46–60
21. Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K.: A racing algorithm for configuring metaheuristics. In: Proc. of the Genetic and Evolutionary Computation Conference. GECCO '02, Morgan Kaufmann (2002) 11–18
22. Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence alg. Swarm and Evolutionary Computation **1**(1) (2011) 3–18
23. Sinnema, R.: Experiment description language (edl) (2003)
24. Köhn, D., Novère, N.: Sed-ml — an xml format for the implementation of the miase guidelines. In: Proc. of the 6th Inte. Conf. on Computational Methods in Systems Biology. CMSB '08, Springer-Verlag (2008) 176–190
25. Shane, T.: A partial implementation of the bica cognitive decathlon using the psychology experiment building language (pebl). International Journal of Machine Consciousness **2**(02) (2010) 273–288