The Executable Experimental Template Pattern for the Systematic Comparison of Metaheuristics

[Extended Abstract]

Geoffrey Neumann
Computing Science and Mathematics
University of Stirling
FK9 4LA Scotland UK
gkn@cs.stir.ac.uk

Jerry Swan
Computing Science and Mathematics
University of Stirling
FK9 4LA Scotland UK
jsw@cs.stir.ac.uk

Mark Harman
Department of Computer Science
University College London,
London, WC1E 6BT, UK.
mark.harman@ucl.ac.uk

John A. Clark
Department of Computer Science and YYCSA
University of York
York, YO10 5GH, UK.
john.clark@cs.york.ac.uk

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence; I.2.8 [**Problem Solving, Control Methods, and Search**]: [Heuristic Methods]

Keywords

Statistical Significance, Hypothesis Testing, Effect Size

1. PROBLEM STATEMENT

In order to establish the effectiveness of a new metaheuristic it is necessary to compare it against a base case (e.g. some version of the metaheuristic without the new functionality under consideration or a metaheuristic representing the state-of-the-art).

This is an area that is fraught with difficulty for several reasons. Firstly, for some metaheuristics there are many parameters that may require manual tuning (e.g. mutation or crossover rate [8]). As there is a tendency to tune these parameters to obtain the best possible performance for a new metaheuristic it is difficult to differentiate between a genuine advantage and the effect of tuning.

Secondly, since metaheuristics are stochastic, it is important to distinguish genuine improvements in the performance of a metaheuristic from apparent improvements arising from chance. There are a wide variety of statistical tests for this purpose, many of which have certain preconditions regarding the nature of the data (e.g. assumptions of normality). It is desirable that a statistical test is chosen which is appropriate for the results, and yet metaheuristic researchers are often not sufficiently knowledgeable to make this decision [2] [9].

This paper focusses primarily on a solution to the second of these problems, but also describes how this solution may be incorporated into a wider framework for testing a new metaheuristic.

2. THE SOLUTION

We are developing an automated system for comparing metaheuristics. We claim that this would be of benefit, both in ensuring good practice in the assessment of new metaheuristics and as part of the wider goal of automating the metaheuristic design process. A framework for automatically choosing and carrying out statistical tests, based on the nature of the data, forms a key part of such a system.

The approach proposed in this paper is an application for systematic comparison of experimental results. For simplicity, we will restrict discussion to comparison of a pair of experiments, since the nontransitive effects that may result from comparing multiple experiments [12] would detract from the essential points we wish to convey. We will also initially focus solely on real valued data, although nomimal data may be considered at a later date. This will ultimately form the 'Core Layer' of a framework for the systematic comparison of metaheuristics. We term this framework as an 'Executable Experimental Template' (EET) which is an instance of a Template Method Pattern whereby a skeleton structure is defined which defers to user-specified callbacks which are invoked at key points in the skeleton. Each statistical test, for example, constitutes one subclass [7].

The EET framework consists of three layers. Although the 'Core Layer' is the focus of this paper, all three layers are briefly outlined below:

1. 'Core Layer': Responsible for determining whether two datasets are different to a statistically significant extent and what the effect size of this difference is.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2881-4/14/07 ...$15.00.
http://dx.doi.org/10.1145/2598394.2609850.

2. ‘Experimental Layer’: Responsible for planning and conducting experiments. In the case of metaheuristics, it should be able to conduct the experiments necessary to determine whether there is a difference in performance between different techniques. Given two datapoint-generating procedures f_a and f_b (potentially paired), this layer will initially generate k datapoints, producing two sets of fitness scores **a** and **b**. Although we are working in the domain of metaheuristic development, without loss of generality, f_a and f_b can be any stochastic functions able to generate data points. The functionality of the ‘Core Layer’ is invoked to determine whether **a** and **b** are significantly different and the size of the effect observed. If the ‘Core Layer’ is unable to find a statistically significant difference between the two sets of results then the ‘Experimental Layer’ may decide to run further tests to obtain more results. This increases the probability that a significant p value will be obtained but, in doing so, increases the risk that two techniques will be judged to be significantly different even when the difference is minimal. This concern is addressed by ensuring that the ‘Core Layer’ carries out effect size tests in addition to p value tests.

3. ‘Sensitivity Layer’: Responsible for establishing how senstive a datapoint generator is to its choice of parameters. This layer is provided with a *parameter schema* that describes the set of permissible parameter values, (e.g. mutation or crossover probabilities when the datapoint generator is a genetic algorithm). This layer samples the search space of parameters by generating datapoints from different parameter settings. These will be compared using the ‘Experimental Layer’. The underlying intention is to allow fair comparision of the ‘tunability’ of metaheuristics in terms of: a) the robustness of metaheuristics w.r.t. their parameter settings b) the number of samples required to achieve equivalent results. The process of automated parameter tuning to ensure the fair comparison of algorithms has precedent in the work of Wang et al [18].

The EET framework has the following design criteria:

1. Portability: It should be widely portable and easily deployed. Conforming to these requirements will ensure maximum adoption and enable EETs to become a standard.

2. Familiarity: It should use a widely-accepted implementation for statistical tests.

3. Recordability: Every decision made by an EET (as well as the outcome of every test) should be recorded and made available to the user. This will enable the user to publish a complete account of all testing carried out and so will help to ensure complete transparency for all published results.

4. Controllability: Although the system should be able to function without user input, it should also be possible for the user to take manual control of any part of the process. Where a decision cannot be made automatically the system should inform the user and prompt for further action.

5. Plugability: The user should be able to modify EETs to reflect their own preferences in areas such as the choice of statistical test. They should be able to add new tests or new preconditions, to remove tests or to change the way in which tests are selected. Although EETs will have a default configuration suitable for most users, those who have expertise in a particular area should be able to construct versions of EETs that are most suitable for their requirements.

In order to satisfy the portability and familiarity requirements, the EETs are implemented in the Java programming language. All statistical tests are carried out using R, a popular statistical language providing standard implementations of many statistical tests [15]. This is a natural implementation as Java provides RJava, an interface to R. Java itself has the advantage of being a widely used and platform independent language.

2.1 The ‘Core Layer’

This layer takes the form of a rule-based system for hypothesis testing. It takes as input two datasets (viz. ‘Dataset A’ and ‘Dataset B’). It may be used to compare a new metaheuristic (Metaheuristic A) against a baseline metaheuristic, (Metaheuristic B). In this case Dataset A will consist of a vector of fitnesses achieved by Metaheuristic A and Dataset B will be the corresponding vector for Metaheuristic B. The purpose of this layer is to find a statistical test which is able to correctly reject the null hypothesis if and only if there is a significant difference between the two datasets. The null hypothesis in this case is that there is no such difference and that both datasets are drawn from the same distribution. This test should have sufficient power to avoid making type II errors [14] without making inaccurate assumptions about the data, increasing the risk of type I errors.

In order to identify the appropriate test, the datasets are examined for the following characteristics:

- Whether both datasets follow a normal distribution.
- Whether both datasets have a similar variance or whether this varies between the two datasets, in which case the two datasets are said to *heteroscedastic*.
- If one or both datasets are skewed, whether the extent and direction of skew is similar in both datasets.
- Whether the level of kurtosis is similar in both datasets.
- Whether either dataset contains missing data.
- Whether the size of both datasets is the same or whether there is a difference. If there is a difference then the two datasets are said to be *unbalanced*.

These characteristics are used in a decision tree to choose the appropriate statistical test (i.e. a test which does not make an assumption which contradicts an observed characteristic).

We do not claim this list to be complete: rather it will be revised according to feedback from the empirical methods community. It should be noted that, although some research has already been carried out on automating the process of statistical testing [10] we are not aware of any complete solutions. That is to say, a solution that takes into account all of these considerations and that can be integrated into a wider automated experimental framework.

Having decided on a test, the EET performs the test and outputs the corresponding p value. If this p value is below the level of confidence specified by the user, then this is interpreted as a rejection of the null hypothesis. Although the p value on its own is useful for determining statistical significance, it can be misleading when the number of runs is large. In this situation the p value will likely be low even when two experimental treatments differ to only a very slight degree. For this reason it is necessary to supplement a statistically significant p value with a measure of effect size in order to give an indication of the magnitude of difference between the two metaheuristics [1]. The 'Core Layer' of the EET framework therefore also chooses and carries out an appropriate effect size test and outputs the results of this test too.

Because of the number of characteristics to be taken into consideration and the fact that many of them require their own tests, the process of choosing an appropriate statistical test may become complex and time consuming. For this reason, it is highly desirable that a simple and efficient test should occupy the root of the decision tree and that further tests should only take place when the results of this test are unsatisfactory. The Cliff test was chosen for this purpose due to its simplicity and because it makes no assumptions about the data [5]. This test produces an effect size (d). d is simply the number of possible comparisons between the two datasets that return in Dataset A's favour subtracted from the number of comparisons that return in Dataset B's favour. This is then divided by the total number of possible comparisons. d will be in a range -1.0 to +1.0, with values close to 0 indicating little difference between the two samples. A d value close to -1.0 or +1.0 indicates both a high probability that the two datasets are from different distributions and provides a measure of effect size similar to the Vargha-Delaney $\hat{A}_{12}$ measure recommended elsewhere [2]. Because of this, by initially using the Cliff test, EETs do not necessarily need to identify and carry out either an ideal test for statistical significance or an ideal test for effect size.

3. AN EXAMPLE: CHOOSING AND CARRYING OUT STATISTICAL TESTS

This section illustrates the behaviour of the 'Core Layer'. As previously mentioned, the schematic provided here is simply an example produced from the rules existing in our current, standard implementation of the EET framework. These rules may change due to further feedback from the empirical methods community. It is also a key requirement that the specific tests used and the decisions made are plugable modules. This is so that the user is able to adapt the default EET configuration to their needs.

We will assume that this layer is presented with two datasets, Dataset A and Dataset B. Let us assume that these two datasets have the attributes specified in Table 1.

To establish whether there is a significant difference between these two datasets, the Cliff test is first employed. As discussed above, this is the EET framework's default test as it has no preconditions and so is suitable for any two datasets. The result of this test will be returned to the user (who may be either a human researcher or the 'Experimental Layer' of the framework). If the user is not satisfied with the outcome of the Cliff test then the following steps will be completed. The outcome of these steps will be a p value indicating the statistical significance of the difference between the two sets and a measure of effect size, indicating the magnitude of the difference between the two sets.

Table 1: Datasets

Attribute	Dataset A	Dataset B
Dataset Size	35	60
Distribution Type	Normal	Not Normal
Heteroscedastic	Yes	Yes
Missing Data	No	No

1. First of all, the number of data points in each dataset will be counted. If either dataset contains fewer data points than a pre specified threshold then a warning will be returned to the user. For example: with fewer than 20 data points determining whether the data is normally distributed can be difficult [16]. As normality is a prerequisite for many statistical tests, this may lead to an inappropriate test being chosen. A low number of datapoints may also reduce the probability that a statistically significant difference can be found [1].

2. It will then be determined whether or not the data is normally distributed. A decision tree within EETs will be used to choose an appropriate normality test:

    (a) For Dataset A the Shapiro-Wilk test will be used. This has been shown to perform better than several well known alternatives for a dataset size of between 20 and 40 [16].

    (b) For Dataset B the Shapiro-Francia test will be used as it has been shown to perform better than the Shapiro-Wilk test for dataset sizes above 40 [17].

3. Once it has been established that Dataset A does follow a normal distribution but Dataset B does not, a test will need to be carried out to determine whether the data is heteroscedastic or not. This will be the case if the two datasets have a different variance. Again, a decision tree within EETs is used to determine an appropriate check for heteroscedasticity.

    (a) For this decision the main consideration is whether the data is normally distributed. As in this case it is not, a common check for heteroscedasticity, the Bartlett test, is not suitable as it relies on both datasets following a normal distribution. Brown and Forsythe discuss a number of alternatives [3], including several variations of the Levene test, which may be used instead.

4. In this case, a reliable test for heteroscedasticity should show that the data that we are using is heteroscedastic.

5. It is now known that two preconditions that some statistical tests rely on, i.e. that both datasets are normally distributed and that there is no heteroscedasticity, are not met. This information will be used by the main decision tree that is used to decide on a statistical test:

    (a) In the absence of normality, a nonparametric test is likely to be more reliable than a parametric test. The Mann-Whitney U test, a popular non-parametric test, will be considered [13].

(b) The U test will be rejected when heteroscedasticity is considered as it is sensitive to heteroscedasticity. A more robust test, such as the Brunner-Munzel test [4] will be chosen.

6. Having chosen a test, the p value will be obtained.

7. If the p value is below the threshold, the datasets will be judged to be different to a statistically significant extent. An effect size test will now be chosen in a similar manner to how the significance test was chosen.

4. CONSEQUENCES

- **Decreased human effort** EETs allow further automation of the time-consuming process of parameter tuning and choosing an appropriate statistical test can now be achieved automatically.
- **Less dependence on the knowledge of the researcher** The necessary statistical knowledge is embedded within EETs.
- **A step towards fully-automated metaheuristic development** If a generative hyperheuristic [6] is integrated into EETs then the complete process of developing and testing metaheuristics may be automated.
- **Helps to ensure complete transparency in experimental methods** EETs will generate a report detailing every decision that was made and the grounds on which it was made. For example, suppose that a normality test is carried out in order to choose the most appropriate statistical test. Details of which normality test was used and how it was chosen, the resulting normality score and how this affected subsequent decisions will all be included in the report. This report will be in latex so that a complete, standardized and unambiguous account of statistical testing can be easily incorporated into publications for peer review.
- **Helps to ensure reproducibility** By making experimental conditions explicit it will be easier to recreate these experiments.
- **Helps to ensure a standard practise in the analysis of results** In current research, experimentation is often carried out in an informal and *ad hoc* manner [11]. A recent study by Arcuri found a lack of consistency in the quality of statistical testing [1]. In many cases the number of data points was insufficient and sometimes statistical testing was omitted altogether. By automating the process of statistical testing EETs will bring much needed standardization to this area.
- **Standardization and transparency in parameter tuning** The 'Sensitivity Layer' also encourages standardization and full transparancy in the process of parameter tuning. EETs produce a report which will enable researchers to provide complete, detailed and standardized information on how a new metaheuristic performs under different parameter settings. This will both confirm that the performance has been fairly assessed and provide information on metaheuristic robustness and the amount of parameter tuning that is required to produce good performance.

5. REFERENCES

[1] A. Arcuri and L. Briand. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In *ACM/IEEE Int. Conf. on Software Engineering (ICSE)*, pages 1–10, 2011.

[2] Andrea Arcuri and Lionel Briand. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability*, 2012.

[3] Morton B Brown and Alan B Forsythe. Robust tests for the equality of variances. *Journ. of the American Statistical Association*, 69(346):364–367, 1974.

[4] Edgar Brunner and Ullrich Munzel. The nonparametric behrens-fisher problem: Asymptotic theory and a small-sample approximation. *Biometrical Journ.*, 42(1):17–25, 2000.

[5] Norman Cliff. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological Bulletin*, 114(3):494, 1993.

[6] G. Kendall G. Ochoa E. Özcan E.K. Burke, M. Hyde and J. R. Woodward. *A Classification of Hyper-heuristic Approaches*, chapter Handbook of Meta-Heuristics, pages 449–468. Kluwer, 2010.

[7] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns: elements of reusable object-oriented software.* Pearson Education, 1994.

[8] Greffenstette. Optimization of control parameters for genetic algorithms. In *IEEE Trans. Syst. Man Cybern. SMC-16*, volume 1, pages 122–128, Jan/Feb 1986.

[9] Phil Souza Jerffeson Teixeira de Harman, Mark McMinn and Shin Yoo. Search based software engineering: Techniques, taxonomy, tutorial. In Bertrand Meyer and Martin Nordio, editors, *Empirical software engineering and verification: LASER 2009-2010*, pages 1–59. Springer, 2012. LNCS 7007.

[10] W.M. Hathaway. Automated hypothesis testing, February 5 2013. US Patent 8,370,107.

[11] John N Hooker. Needed: An empirical science of algorithms. *Operations Research*, 42(2):201–212, 1994.

[12] Sean Luke. *Essentials of Metaheuristics.* Lulu, second edition, 2013. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/.

[13] Henry B. Mann and Donald R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, 18(1):50–60, 1947.

[14] J. Neyman and E. S. Pearson. On the Use and Interpretation of Certain Test Criteria for Purposes of Statistical Inference: Part I. *Biometrika*, 20A(1/2):175–240, July 1928.

[15] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2013.

[16] Martin B Shapiro, Samuel S Wilk and Hwei J Chen. A comparative study of various tests for normality. *Journ. of the American Statistical Association*, 63(324):1343–1372, 1968.

[17] Samuel S Shapiro and RS Francia. An approximate analysis of variance test for normality. *Journ. of the American Statistical Association*, 67(337):215–216, 1972.

[18] Tiantian Wang, Mark Harman, Yue Jia, and Jens Krinke. Searching for better configurations: a rigorous approach to clone evaluation. In *European Software Engineering Conf. and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE'13*, Saint Petersburg, Russian Federation, August 2013. ACM.