Notes on the Estimation of the Expected Performance of Automatic Methods for the Design of Control Software for Robot Swarms

Mauro Birattari

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2020-010

June 2020

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:
IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2020-010

Notes on the estimation of the expected performance of automatic methods for the design of control software for robot swarms

Mauro Birattari
IRIDIA, Université libre de Bruxelles, Belgium

June 2020

The goal of this short document is to sketch some ideas that I plan on developing formally in a forthcoming publication. I focus here on the automatic design of robot swarms and, specifically, on the empirical assessment of automatic design methods. For an introduction to swarm robotics, I would refer the reader to Dorigo, Birattari, and Brambilla (2014); for a review of the literature, to Brambilla et al. (2013) and to Garattoni and Birattari (2016); for a discussion on the automatic design of robot swarms and for a sketch of the domain literature, to Francesca and Birattari (2016) and to Birattari et al. (2019).

In this document, I consider the case in which a mission is given and one wishes to estimate the expected performance of an automatic method for the optimization-based design of the control software of a robot swarm that is deemed to perform the given mission. The performance that one would observe is a stochastic variable and therefore estimating its expectation is a reasonable goal. The expectation should be computed with respect to all the sources of randomness involved in the process. The sources of randomness are the following:

The realization of the design process: the design process is stochastic in nature. If it is performed multiple times, it will (likely) produce different instances of control software.

The realization of the execution: the performance of a given instance of control software is clearly a stochastic quantity. If the same instance of control software is executed multiple times, the performance observed will (likely) vary.

Having stated this, it is clear that, if one runs the automatic design method once on the given mission, and then executes once the control software generated by the automatic design method, the performance observed is an unbiased estimation of the expected performance of the design method at hand on the given mission. It is also clear that such an estimation based on a single run of the design process and a single execution will have a (possibly) high variance. It is legitimate to consider such an estimation as poor and unsatisfactory.

The widely accepted practice is to observe the performance over multiple runs (of the design process and/or of the generated control software) and report the average. Other statistics can be considered and might be even more appropriate under some circumstances. In the following, we will assume that the expected value of the performance is the quantity that one wishes to estimate, and we will therefore focus on the average as a relevant statistic.

Taking for granted that multiple runs are needed to reduce the variance, the questions that arise are: How many design processes should one run on the given mission? How many times should one then execute each of the instances of control software produced? The obvious answer would be: the more the better! Indeed, by increasing indefinitely the number of design processes performed and the number of evaluations of each instance of control software generated, the variance of the estimation would converge to zero. In practice, one has typically (if not always) to face practical constraints that limit the number of experiments that can be performed. Indeed, running experiments with robots is time consuming and could demand a large amount of resources.

It is realistic to assume that an upper bound N is given on the number of execution, that is, the number of experiments that one can run with the robots. It is also realistic to assume that the number of execution is the real bottleneck in terms of resources demanded. Indeed, running experiments with the robots is a labor-intensive activity and the really expensive and time-consuming part in the research on the automatic design of control software for robot swarms. On the other hand, the design process is fully automatic and multiple processes can run in parallel on a high-performance computing cluster. We can assume that the cost (in abstract terms: time and resources) of running a design process is negligible compared to the one of running robot experiments. We also assume that, before running a design process on the given mission, we do not have any prior information 1. on how well the control software we can generate automatically will perform and 2. on what will be the variance of the performance. An experimental design for estimating the expected performance of a design method on a given mission, with the constraint that a maximum number N of executions can be performed, can be formally described by a pair $\langle d, n \rangle$, with $d \cdot n \leq N$: the expected performance is estimated on the basis of d design processes (to generate d instances of control software) and n executions of each of the d instances of control software generated.

It has to be noticed that any pair $\langle d, n \rangle$ yields an unbiased estimate of the expected performance. Yet, different pairs might differ for what concerns the variance of the estimate they yield.

THEOREM. Under the assumptions made above, given that a maximum number N of executions can be performed, the experimental design described by the pair $\langle d, n \rangle$, with $d = N$, and $n = 1$, is the one that minimizes the variance of the estimate.

A proof will be provided in a future publication. It will be formally identical to the one I previously provided for a very similar (formally identical) estimation problem that emerges in the assessment of heuristic algorithms for combinatorial optimization (Birattari, 2004; Birattari, 2009). Also in that case, the expected value of a quantity has to be estimated empirically and the expectation has to be computed with respect to two probability measures, which describe two sources of uncertainties. For the time being, I refer the interested reader to the aforementioned original publications to gain an insight in the structure of the proof.

The same conclusion that the pair $\langle N, 1\rangle$ is the one that minimizes the variance is relevant also in the case one wishes to compare the expected performance of two design methods—the reasoning can be generalized to more than two design methods, as well. When two methods are considered, the reasoning presented above applies to the estimation of the expected value of the difference between the performance of the control software produced by the two methods under analysis.

It should be noticed that, in the setting described above, the naive approach that is often adopted and that consists in running multiple executions of the same instance of control software hides some catches that could lead to misleading results. In particular, it could lead to wrong conclusions when two (or more) design methods are compared. By taking $n \gg 1$, one runs the risk of undersampling the space of the realizations of the design processes and oversampling the one of the executions. Let us consider the comparison of two design methods, A and B. Let us make the hypothesis that, on the mission at hand, the expected performance of A is better than the one of B. Let us also make the hypothesis that some realizations of the design process of B produce an instance of control software that performs better than the one produced by some realizations of the design process of A. This is perfectly possible, and typically very likely, as the variance involved in an automatic design process is often quite large. If the experimental design adopted undersamples the space of the realizations of the design processes so as to allow multiple execution of the same instances of control software, the risk exists that the sample over-represents the realizations of the design process that are favorable to B. If this happens, as $n \gg 1$, the risk exists that the observed performance difference, which will be wrongly in favor of B, is eventually statistically significant. By undersampling the space of the realizations of the design processes and oversampling the one of the executions, the confidence level imposed does not apply anymore to the overall estimation of the differences but rather to the performance of the specific instances of control software produced by the few realizations of the design processes that have been sampled.

The above reasoning could possibly appear clearer if we push things to the extreme. Let us sample a single realization of the design processes ($d = 1$) and use all the N evaluations available to test the single pair of instance of control software obtained—one for A and one for B. The confidence level will refer to the performance difference of the specific instances of control software produced by the single realization of the design processes sampled for A and B—rather than to the expected performance difference between A and B across all possible realizations of their respective design processes, as we intend. If we happen to sample a pair of realizations of the design processes (one for A and one for B) for which the instance of control software produced by B performs better than the one produced by A, we will observe a performance difference that is in favor of B. If N is sufficiently large, the difference will be statistically significant and we will wrongly conclude that B is better than A. Clearly, this does not extend to the whole set of realizations of the design processes. The result obtained will be wrong (at least with respect to the estimation we intend to perform), even if statistical significance is detected. If $d = N$ (and consequently $n = 1$), the issue does not arise and the confidence level applies indeed to the significance of the difference across the whole set of possible realizations of the design processes of A and B. In this case, if statistical significance is detected, the conclusion to which we get is correct—within the margins of the confidence level, as usual when we perform a statistical test of significance.

To conclude, under the assumptions stated in the document, the experimental design that minimizes the variance of the estimation is the one in which the largest possible number of missions is considered, and in which therefore each instance of control software produced is tested only once. Future work will be devoted to elaborating the ideas sketched here and to providing formal proofs.

Acknowledgements

I have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 681872). I also acknowledges support from the Belgian *Fonds de la Recherche Scientifique*–FNRS, of which I am a Research Director.

References

Birattari, Mauro (2004). *On the estimation of the expected performance of a metaheuristic on a class of instances. How many instances, how many runs?* Tech. rep. TR/IRIDIA/2004-01. Belgium: IRIDIA, Université libre de Bruxelles.

Birattari, Mauro (2009). *Tuning Metaheuristics: A Machine Learning Perspective.* Berlin, Germany: Springer.

Birattari, Mauro, Antoine Ligot, Darko Bozhinoski, Manuele Brambilla, Gianpiero Francesca, Lorenzo Garattoni, David Garzón Ramos, Ken Hasselmann, Miquel Kegeleirs, Jonas Kuckling, Federico Pagnozzi, Andrea Roli, Muhammad Salman, and Thomas Stützle (2019). "Automatic off-line design of robot swarms: a manifesto". In: *Front. Robot. AI* 6.59, pp. 1–6.

Brambilla, Manuele, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo (2013). "Swarm robotics: a review from the swarm engineering perspective". In: *Swarm Intell.* 7.1, pp. 1–41.

Dorigo, Marco, Mauro Birattari, and Manuele Brambilla (2014). "Swarm robotics". In: *Scholarpedia* 9.1, p. 1463.

Francesca, Gianpiero and Mauro Birattari (2016). "Automatic design of robot swarms: achievements and challenges". In: *Front. Robot. AI* 3.29, pp. 1–9.

Garattoni, Lorenzo and Mauro Birattari (2016). "Swarm robotics". In: *Wiley Encyclopedia of Electrical and Electronics Engineering.* Ed. by J.G. Webster. Hoboken NJ: John Wiley & Sons.