Experimental Study of Automated Parameter Tuning on the Example of irace and the Traveling Salesman Problem

Daniil Chivilikhin
ITMO University
Computer Technologies Laboratory
Saint Petersburg, Russia
chivdan@rain.ifmo.ru

ABSTRACT

Performance of optimization algorithms (OA) greatly depends on their parameter values. Automated (offline) parameter tuning methods allow to select good parameter values for an OA. Though such methods are widely used, there is little to none information on how their performance depends on essential parameters such as the tuning time limit (TL). In this paper we report results of an experimental study aimed at covering this gap to some extent. We use `irace`, a popular algorithm configuration tool, for tuning ACOTSP – an ant colony optimization (ACO) software for solving the traveling salesman problem (TSP). The dependencies of `irace` performance on such parameters as `irace` and ACOTSP time limits are studied.

1. INTRODUCTION

One of the main issues that arise with the use of OAs is selecting appropriate algorithm parameter values – so-called *configuration*. This led to the emergence of several automated algorithm configuration methods such as `irace` [2] and SMAC [1], designed for solving the *algorithm configuration problem*: given an OA and a set of problem instances (training set), find algorithm parameter values that optimize a certain performance metric (e.g. running time of the algorithm).

All algorithm configurators allow to place a limit on the time allotted for tuning. Naturally, these parameters should have a great effect on the algorithm configuration method performance, which can be estimated by running the parameterized OA on a set of test instances. Surprisingly, at least to the best of our knowledge, there have yet been no works on the effect these essential parameters have on the configuration performance. In this paper we focus on one of the most popular algorithm configurators, `irace` [2], and study its performance on the example of tuning ACOTSP[1], an ACO based tool for solving the classical TSP.

[1]http://iridia.ulb.ac.be/~mdorigo/ACO/downloads/ACOTSP-1.03.tgz

GECCO '16 July 20-24, 2016, Denver, CO, USA

ACM ISBN 978-1-4503-4323-7/16/07.

DOI: http://dx.doi.org/10.1145/2908961.2908978

2. TUNING ACOTSP WITH IRACE

Given a list of cities and distances d_{ij} between them, the TSP problem consists in finding the shortest route visiting each city only once and returning to the origin city. In symmetric TSP, $d_{ij} = d_{ji}, \forall i, j$.

ACOTSP has 11 tunable parameters and also a TL t, after which the current best TSP solution is returned.

Given a set of optimization problem instances, `irace` can tune an OA for two goals: minimizing running time or maximizing solution quality. The user can specify the total time T that may be used for tuning.

Mentioned facts give rise to the following two research questions (RQ).

RQ1: Given fixed TLs for both ACOTSP (t) and `irace` (T), how stable is the performance of `irace` in terms of the efficiency of the tuned ACOTSP on the test set?

RQ2: How does the performance of `irace` depend on t and T? For example, for a fixed value of t and fixed training and test sets, will increasing T improve the performance of ACOTSP on the test set?

3. EXPERIMENTS

We used the `tsp-rue-1000-3000` TSP instance set from ACLib[2] and divided it into training and test sets 150 instances each. The exact solver Concorde[3] was used to obtain the optimal solutions for test set instances. Experiments were performed on a machine with a 24-core AMD Opteron™ processor 6234 @ 2.4 GHz with 132 Gb of RAM. The `parallel` option for `irace` was used enabling 16 cores. While an `irace` run may result in more than one configuration, we always select the last one as the final result. Since `irace` has a randomized behavior, M independent runs were performed for each combination of t and T values.

During the *testing phase* ACOTSP is parameterized with each found configuration C^i_{tT} obtained by the i-th `irace` run with TLs t and T and executed K times on each test instance j with a TL of $t_{\text{test}} = 30$ seconds. For each such execution the resulting *tour length ratio* r^{ijk}_{tT} is calculated as the ratio of the optimal tour length found using Concorde and the tour length found with ACOTSP.

3.1 RQ1

Tuning of ACOTSP with `irace` was independently performed $M = 20$ times with $t = 30$ seconds and $T = 36$ hours, each configuration was tested $K = 20$ times. For each

[2]http://www.aclib.net

[3]http://www.math.uwaterloo.ca/tsp/concorde.html

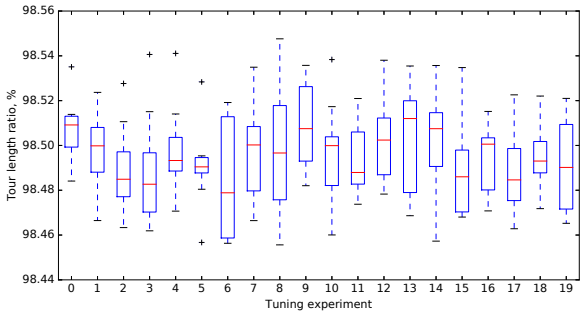

Figure 1: Boxplots of average tour length ratios r^i_{tT} for $t = 30$ s and $T = 36$ h.

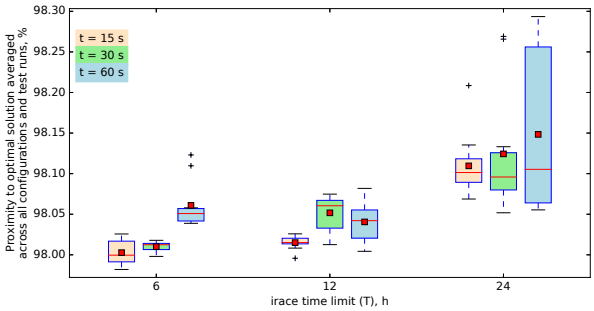

Figure 2: Boxplots of tour length ratios averaged across all test runs for $t \in \{15, 30, 60\}$ seconds and $T \in \{6, 12, 24\}$ hours

configuration C^i_{tT} and each test run k the resulting r^{ijk}_{tT} is averaged across all instances: $R^{ik}_{tT} = \frac{1}{|I_{\text{test}}|} \sum_{j=0}^{|I_{\text{test}}|-1} r^{ijk}_{tT}$. Boxplots of average tour length ratios R^{ik}_{tT} for each configuration i are depicted in Fig. 1.

Plots for all configurations are very similar, no single configuration yields significantly better or worse testing results. Tour length ratios for all configurations are well-situated near their mean values and do not have significant outliers. This points to an observation: *irace*, given sufficient training time, results in different configurations that on average perform similarly (and quite well) on unseen test instances.

3.2 RQ2

In the training phase $M = 10$ independent `irace` runs were performed for each combination of $t \in \{15, 30, 60\}$ seconds and $T \in \{6, 12, 24\}$ hours. For each configuration i, each value of t and each value of T we calculated average tour length ratios R^{ik}_{tT}. We then further averaged these values across all configurations for each combination of t and T. Boxplots of these average tour length ratios are shown in Fig. 2.

Let us first consider fixed values of T. For both $T = 6$ h and $T = 12$ h, results for $t = 60$ s are significantly better than for $t = 15$ s and $t = 30$ s (the paired Wilcoxon signed-rank test [3] was used to check statistical significance of differences in tour length ratios). However, for $T = 24$ h results for all three values of t are not significantly different from each other, though mean values (depicted as squares) seem to show an increasing trend.

On the other hand, if t is fixed, larger values of T yield significantly better performance than smaller ones.

4. DISCUSSION

First of all, we can give a clear and simple answer to RQ1 "How stable is the performance of `irace` given fixed TLs?". It is indeed quite stable. Results from Section 3.1 suggest that given a fair amount of training time, `irace` generates configurations that yield very similar results on the test set. Furthermore, boxplots from Fig. 2 in Section 3.2 also indicate that this statement can be extended to smaller `irace` TL values than the ones considered in Section 3.1.

Answering RQ2 is more challenging. First, it can be concluded that the time limit t of the OA (ACOTSP) during training should be at least not larger than the TL used during testing. Second, for a fixed value of t the performance on test data has a non-decreasing behavior with an increase of training time T.

Third, from Fig. 2 we can derive that when T is small, it is better to have a larger value of ACOTSP time limit t. However, when T is large, it seems to be more preferrable to have more ACOTSP runs with a smaller t.

5. CONCLUSION

Performance evaluation of `irace` applied to ACOTSP has been conducted. First, it was found that `irace` has stable performance (in terms of accuracy on the training set) across several independent tuning runs. Second, we analyzed the dependency of `irace` performance from `irace` and ACOTSP time limits using fixed training and test sets. It was found that, for a fixed ACOTSP time limit, the test set performance has a non-decreasing behavior with the increase of `irace` TL. Results also suggest that if one has a large amount of available resources for tuning their OA, it is more preferrable to set a fairly low TL on the OA to allow `irace` use more algorithm executions.

We believe that our results somewhat cover the gap in understanding how algorithm configuration tools work. Future work in this direction includes studying other algorithm configuration tools and optimization problems.

6. ACKNOWLEDGEMENTS

This work was financially supported by the Government of Russian Federation, Grant 074-U01, and also partially by RFBR, research project No. 14-01-00551 a.

7. REFERENCES

[1] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, LION'05, pages 507–523, Berlin, Heidelberg, 2011. Springer-Verlag.

[2] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Birattari. The `irace` package, iterated race for automatic algorithm configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium, 2011.

[3] F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945.