Reinforcement Learning for Multi-Neighborhood Metaheuristics

Roberto Maria Rosati[1 [0000−0001−9560−6301]]

DPIA, University of Udine, via delle Scienze 206, 33100 Udine, Italy
robertomaria.rosati@uniud.it

Abstract. This paper discusses the use of reinforcement learning for the adaptive tuning of neighborhood probabilities in randomized multi-neighborhood search. In detail, we consider the potential application of Q-learning, a model-free reinforcement learning technique, to multi-neighborhood Simulated Annealing. We discuss the possibility to extend this approach to other metaheuristics, providing the example of the application to CMSA (Construct, Merge, Solve and Adapt).

Keywords: learning, metaheuristics, local search, multi-neighborhood

1 Motivation

Machine learning applied to metaheuristics for adaptive tuning of parameters is a growing research field [11, 19]. This is motivated by the fact that offline tuning remains a time-consuming activity, despite the availability of many black-box automatic algorithm configuration tools, that are designed to simplify the task of designing the experiments for researchers, and to look for the best configurations in the parameters space through an automated statistically-principled procedure. Additionally, details on the design of experiments and on the time spent for tuning are rarely shared [6], and the quest for competitive results leads to the practice of overfitting the parameters on the benchmark instances [14]. Furthermore, tuning procedures struggle to find a single best configuration when instances differ consistently, either in size or in problem-specific features. This scenarios are said to be heterogeneous [17] and some classical approaches to face them are instance clustering [10], instance space analysis [18] or feature-based tuning [4]. However, they can be expensive or require problem-specific knowledge. The need for best practices is still felt as a priority by the community [2].

An advantage of parameter learning lies in the reduction in time spent for the tuning and its related complexity. Additionally, the potential on heterogeneous scenarios is higher, because an adaptive algorithm can behave well on instances with different features, also if they differ substantially from the training ones. We believe that automated learning deserves special consideration if the algorithm is designed for real-world applications, because it is likely that not all practitioners have the statistical and programming background to conduct a tuning procedure by themselves.

More specifically, we are considering the application of Q-learning, a reinforcement learning algorithm, to neighborhood probabilities in metaheuristics based on local search. At every iteration of those algorithms, a local search neighborhood is applied to the current solution, the new solution is evaluated and it is accepted or discarded, according to a specific acceptance criterion. In many complex combinatorial problems a single neighborhood is not enough for good performance and a union of a set of neighborhoods, which is called *multi-neighborhood*, is employed.

In the case of Simulated Annealing, a move is chosen at random at every iteration. Different probabilities are assigned to the neighborhoods. Traditionally, a tuning procedure is used to find those values, such that a higher probability of choice is given to those neighborhoods that give the highest contribution to the improvement of the solution. Other metaheuristics, such as Variable Neighborhood Search or Iterated Local Search use sequential exploration, that is, neighborhoods are exploited one after the other, with some shaking criteria to escape from local minima.

The idea of modifying dynamically the probability of the neighborhoods during the search has been explored recently also in [1, 16, 20], with applications to Variable Neighborhood Search (VNS), Iterated Local Search (ILS) and Late Acceptance Hill-Climbing (LAHC). We consider hereafter the application of a similar methodology to multi-neighborhood Simulated Annealing.

2 Model-free Q-learning

Q-learning is a reinforcement learning algorithm that was initially proposed in [21]. It does not require a complete model of the environment, thus it is a model-free technique.

Given a real parameter $\alpha \in [0, 1]$, called *learning rate*, we calculate the new probability of neighborhood i at iteration j as:

$$p_{i,j} = (1 - \alpha) \cdot p_{i,j-1} + \alpha \cdot r_{i,j}$$

where $r_{i,j}$ is the reward obtained by neighborhood i at iteration j and $p_{i,j-1}$ is the previous probability of neighborhood i. If $\sum_i p_{i,0} = 1$, at iteration 0, and $\sum_i r_{i,j} = 1$, at every iteration j, then this learning criterion guarantees that the probabilities of the neighborhoods continue to sum exactly 1 along the whole process.

Note that if we choose $\alpha = 0$, we have that the neighborhood probabilities remain equal to the initial values during the full execution, because we have no learning. This is what happens in the case of offline tuning, when multi-neighborhood probabilities are pre-determined. On the other hand, if we choose $\alpha = 1$, we have that at every iteration the probabilities depend just on the result of the last iteration, and the memory from previous results is not maintained.

Q-learning can quickly knock out the neighborhoods that contribute less to the solution, especially if learning rate α is high. Indeed, if a neighborhood

reaches a probability very near to 0, it will have little or no chances to be chosen again and thus to get a reward. Some solutions to mitigate such effects are ϵ-greedy technique or the application of a threshold τ. Both guarantee that a certain amount of exploration is performed whatsoever. In ϵ-greedy Q-learning, at every iteration, neighborhoods are chosen uniformly with probability ϵ, while with probability $1 - \epsilon$ the choice is done according to the learned probabilities. Learning is applied as usual, but ϵ-greedy Q-learning guarantees that those neighborhoods that are contributing less to the solution do not disappear from the portfolio, and that a certain degree of exploration is assured.

An alternative to ϵ-greedy is the application of a minimum threshold τ. At each iteration, the new neighborhood probability is computed as:

$$p_{i,j} = \max((1 - \alpha) \cdot p_{i,j-1} + \alpha \cdot r_{i,j}, \tau)$$

where τ is the minimum probability threshold for the neighborhoods. The probability of the other neighborhoods needs to be re-balanced so that the probabilities sum one.

3 Applications to Simulated Annealing

Simulated Annealing involves several parameters: start temperature T_0, final temperature T_f, cooling rate α, neighbors sampled per iteration N_s. In addition to them, multi-neighborhood Simulated Annealing requires that a vector of probabilities $P = (p_1, p_2...p_n)$, with $\sum_{i=1}^{|P|} p_i = 1$ is provided, so that at each iteration, the algorithm selects, at random, the neighborhood to be applied according to its probability in the vector. Successful applications of this technique are found, among others, in [8, 15].

Traditionally, the probabilities of the neighborhoods are determined through an offline tuning procedure. Nevertheless, neighborhood effectiveness might be influenced by different features of the instance, as well as by different stages of the search process: for example, some neighborhoods may be more useful than others to escape from local minima, while others are to be preferred in the exploitation. This adaptive mechanism can help to reach a better balance between exploration and exploitation during the search.

We consider the Q-learning algorithm proposed in Section 2. Values $p_{i,j}$ are associated with the probabilities of the neighborhoods. The choice of the reward function $r_{i,j}$ is more complex and it is a relevant and critical design choice. Some aspects need to be considered:

- The length of the learning batch, that is the number of iterations between sequential updates of the probabilities.
- The reward function may be based either on the improvement that neighborhoods give in terms of objective function, or on their acceptance rate in the last learning batch. It can also be a mixture of both.
- Parameters α, and ϵ or τ, still require to be tuned. They might have a correlation with other parameters, especially initial temperature T_0 and final temperature T_f.

Two potential applications are the ITC2021 Sports Timetabling problem [15] and the Minimum Interference Frequency Assignment Problem [8], that we solved in the past with Simulated Annealing. In the former, we understood that the feature *phased* was a relevant one and we clustered instances into phased and non-phased instances. We applied independent tuning procedures for the neighborhood probabilities, and we showed that the neighborhood *PartialSwapTeamsPhased* was more useful on phased instances, thus resulting in a higher assigned probability. Nevertheless, other authors demonstrated that another important feature was the presence of constraint BR2 [12], which may influence relevantly neighborhoods behavior. We plan to study if an online learning procedure for the neighborhood probabilities has a positive effect on the algorithm. In the latter, two slightly different formulations of the problem were tackled by the algorithm. The same multi-neighborhood composed by six neighborhoods was used in both formulations. Furthermore, instances of the second formulations were characterized by different costs, thus those instances were partitioned into three clusters for tuning purposes. The result is a total of four different sets of instances, thus four distinct tuning procedures for the probabilities of the multi-neighborhood. Again, we consider that online learning of the neighborhood probabilities will result in a more generalized algorithm and in a simplified tuning procedure.

Other works we have in mind for Simulated Annealing concern Examination Timetabling [3] and Home Healthcare Routing and Scheduling [13].

4 Other applications

Extended applications concern hyperheuristics [7], that are metaheuristics where low-level heuristics replace neighborhoods, or matheuristics [9], where exact solvers are combined with metaheuristic procedures.

An example comes from Construct, Merge, Solve and Adapt (CMSA) [5]. In the construct phase of this matheuristic, a randomized greedy algorithm generates a number of solutions in a probabilistic way. Traditionally, just one greedy has been employed, but nothing impedes making use of a portfolio of algorithms, combined in a multi-neighborhood fashion. Adaptive probabilities can be employed according to the methodologies defined in Section 2. The reward function is computed after the solve phase, when an exact solver finds a possibly optimal solution in the sub-instance, obtained by merging the solution components generated in the construct phase. The idea for learning is that every greedy is rewarded proportionally to the number of generated solutions components that are chosen by the exact solver. If the same component was generated by multiple greedy procedures, all of those that generated it are rewarded. Please note that we are not considering rewarding the algorithms that provide the best solutions in terms of costs, but rather those that are more useful to the exact solver. As in the case of Simulated Annealing, the ϵ-greedy variant or the threshold τ can be employed, in order to avoid that a greedy permanently disappears from the portfolio.

5 Conclusions

Learning neighborhood probabilities during search has the advantage of reducing the time spent for the tuning procedure. It can also help to create a more general algorithm in the case of heterogeneous scenarios, characterized by the absence of a single dominating configuration. We considered the application of Q-learning, a model-free reinforcement learning technique. In Q-learning, the probabilities of the neighborhood for the next iteration depend solely on the current values and on the obtained reward. For a better trade-off between exploration and exploitation, a ϵ-greedy criterion or a minimum threshold τ can be considered. Future and ongoing applications concern the extension of multi-neighborhood Simulated Annealing algorithms. To this regard, Sports Timetabling and Minimum Interference Frequency Assignment are considered. The model can be extended also to other metaheuristics that are not based on local search, but that have an exploration component that can be assimilated to a multi-neighborhood.

References

1. Alicastro, M., Ferone, D., Festa, P., Fugaro, S., Pastore, T.: A reinforcement learning iterated local search for makespan minimization in additive manufacturing machine scheduling problems. Computers & Operations Research **131**, 105272 (2021)
2. Bartz-Beielstein, T., Doerr, C., Berg, D.v.d., Bossek, J., Chandrasekaran, S., Eftimov, T., Fischbach, A., Kerschke, P., La Cava, W., Lopez-Ibanez, M., et al.: Benchmarking in optimization: Best practice and open issues. arXiv preprint arXiv:2007.03488 (2020)
3. Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A.: Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. Computers & Operations Research **132**, 105300 (2021)
4. Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A., Urli, T.: Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem. Computers & Operations Research **65**, 83–92 (2016)
5. Blum, C., Pinacho, P., López-Ibáñez, M., Lozano, J.A.: Construct, merge, solve & adapt a new general algorithm for combinatorial optimization. Computers & Operations Research **68**, 75–88 (2016)
6. Brownlee, J., et al.: A note on research methodology and benchmarking optimization algorithms. Complex Intelligent Systems Laboratory (CIS), Centre for Information Technology Research (CITR), Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology, Victoria, Australia, Technical Report ID **70125** (2007)
7. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Qu, R.: Hyper-heuristics: A survey of the state of the art. Journal of the Operational Research Society **64**(12), 1695–1724 (2013)
8. Ceschia, S., Di Gaspero, L., Rosati, R.M., Schaerf, A.: Multi-neighborhood simulated annealing for the minimum interference frequency assignment problem. EURO Journal on Computational Optimization **10**, 100024 (2022)
9. Fischetti, M., Fischetti, M.: Matheuristics. In: Handbook of heuristics, pp. 121–153. Springer (2018)

10. Kadioglu, S., Malitsky, Y., Sellmann, M., Tierney, K.: Isac–instance-specific algorithm configuration. In: ECAI 2010, pp. 751–756. IOS Press (2010)
11. Karimi-Mamaghan, M., Mohammadi, M., Meyer, P., Karimi-Mamaghan, A.M., Talbi, E.G.: Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. European Journal of Operational Research **296**(2), 393–422 (2022)
12. Lamas-Fernandez, C., Martinez-Sykora, A., Potts, C.N.: Scheduling double round-robin sports tournaments. In: Proceedings of the 13th International Conference on the Practice and Theory of Automated Timetabling-PATAT. vol. 2 (2021)
13. Mankowska, D.S., Meisel, F., Bierwirth, C.: The home health care routing and scheduling problem with interdependent services. Health care management science **17**(1), 15–30 (2014)
14. Rabanal, P., Rodríguez, I., Rubio, F.: Assessing metaheuristics by means of random benchmarks. Procedia Computer Science **80**, 289–300 (2016)
15. Rosati, R.M., Petris, M., Di Gaspero, L., Schaerf, A.: Multi-neighborhood simulated annealing for the sports timetabling competition itc2021. Journal of Scheduling **25**(3), 301–319 (2022)
16. dos Santos, J.P.Q., de Melo, J.D., Neto, A.D.D., Aloise, D.: Reactive search strategies using reinforcement learning, local search algorithms and variable neighborhood search. Expert Systems with Applications **41**(10), 4939–4949 (2014)
17. Schneider, M., Hoos, H.H.: Quantifying homogeneity of instance sets for algorithm configuration. In: International Conference on Learning and Intelligent Optimization. pp. 190–204. Springer (2012)
18. Smith-Miles, K., Baatar, D., Wreford, B., Lewis, R.: Towards objective measures of algorithm performance across instance space. Computers & Operations Research **45**, 12–24 (2014)
19. Talbi, E.G.: Machine learning into metaheuristics: A survey and taxonomy. ACM Computing Surveys (CSUR) **54**(6), 1–32 (2021)
20. Toffolo, T.A., Christiaens, J., Van Malderen, S., Wauters, T., Berghe, G.V.: Stochastic local search with learning automaton for the swap-body vehicle routing problem. Computers & Operations Research **89**, 68–81 (2018)
21. Watkins, C.J.C.H.: Learning from delayed rewards (1989)