A Meta-Learning Approach to Select Meta-Heuristics for the Traveling Salesman Problem Using MLP-Based Label Ranking

Jorge Kanda[1,2], Carlos Soares[3], Eduardo Hruschka[1], and Andre de Carvalho[1]

[1] Instituto de Ciencias Matematicas e de Computacao, Universidade de Sao Paulo, Avenida Trabalhador Sao-Carlense, 400 - Centro, 13566-590 Sao Carlos - SP, Brazil
{kanda,erh,andre}@icmc.usp.br
[2] Instituto de Ciencias Exatas e Tecnologias, Universidade Federal do Amazonas, Rua Nossa Senhora do Rosario, 3863 - Tiradentes, 69103-128 Itacoatiara - AM, Brazil
[3] INESC TEC Porto LA/Faculdade de Economia, Universidade do Porto, Rua Dr. Roberto Frias, 4200-464 Porto, Portugal
csoares@fep.up.pt

Abstract. Different meta-heuristics (MHs) may find the best solutions for different traveling salesman problem (TSP) instances. The *a priori* selection of the best MH for a given instance is a difficult task. We address this task by using a meta-learning based approach, which ranks different MHs according to their expected performance. Our approach uses Multilayer Perceptrons (MLPs) for label ranking. It is tested on two different TSP scenarios, namely: *re-visiting customers* and *visiting prospects*. The experimental results show that: 1) MLPs can accurately predict MH rankings for TSP, 2) better TSP solutions can be obtained from a label ranking compared to multilabel classification approach, and 3) it is important to consider different TSP application scenarios when using meta-learning for MH selection.

Keywords: meta-learning, label ranking, multilayer perceptron, traveling salesman problem.

1 Introduction

The Traveling Salesman Problem (TSP) is a classic optimization problem, which is formally defined by means of a weighted graph $G = (V, E)$, in which $V = \{v_1, v_2, ..., v_n\}$ is a set of vertices and $E = \{\langle v_i, v_j \rangle : v_i, v_j \in V\}$ is a set of edges. Each vertex $v_i \in V$ represents a city and each edge $\langle v_i, v_j \rangle \in E$ connects the vertices v_i and v_j. The cost of travel from v_i to v_j is given by the weight value of the edge $\langle v_i, v_j \rangle$. The best solution for a TSP instance involves finding the minimal cost tour visiting each of n cities only once and returning to the starting city [1].

It is difficult to find the best solution for several TSP instances, since this problem belongs to the class of problems known as NP-complete [16]. The TSP complexity is factorial with the number of cities, thus exhaustive search methods

T. Huang et al. (Eds.): ICONIP 2012, Part III, LNCS 7665, pp. 488–495, 2012.

present a high computational cost even for small TSP instances. For example, there are approximately 1.22×10^{17} feasible solutions for a TSP with 20 cities.

Good solutions for TSP can be quickly found by different meta-heuristics (MHs) — *e.g.*, Genetic Algorithms [13], and Ant Colony [5]. MHs are search methods that try to escape from local optima through of interaction between local improvement procedures and higher level strategies [8]. Each MH has its own bias which makes it more suitable for a particular class of instances [25]. Thus, given the large number of available MHs, there can be a MH that is the best for a new TSP instance.

Recently, a meta-learning approach addressed the problem of recommending MHs for new TSP instances as a multilabel classification task [14]. However, when multiple MHs are recommended, no guidance is provided concerning the order in which they should be executed. In this work, we address this problem by using a label ranking approach [4] to predict a ranking of MHs, according to their expected performance. Additionally, previous approaches do not distinguish between different TSP scenarios. Here we separately investigate two important scenarios: when the salesperson (re-)visits current customers and when the prospects are visited for the first time.

The remainder of this paper is organized as follows. Section 2 provides a brief background on meta-learning for algorithm selection and on label ranking. The adaptation of MLPs to learn label rankings is discussed in Section 3. Practical application scenarios of interest are described in Section 4. Based on such scenarios, the experimental setting is detailed in Section 5, and the results are reported in Section 6. Finally, the conclusions are presented in Section 7.

2 Meta-Learning and Label Ranking

The selection of the best algorithm for a given problem has been dealt with in Machine Learning (ML) with meta-learning [2]. Meta-learning studies how learning systems can increase in efficiency through experience and how learning itself can become flexible according to the domain or task under study [24].

Studies that relate ML and optimization problems are recent [20]. Concerning the TSP, a meta-learning approach to recommend MHs [14] classifies TSP instances according to the solutions obtained by a set of MHs. As the best solution for a given TSP instance may be achieved by more than one MH, multilabel classification techniques are applied. In [19], MLP-based models are induced to predict the search effort that each algorithm will need to find the best solution.

The induction of a meta-learning model to select MHs for the TSP is illustrated in Figure 1. TSP properties (meta-features) are calculated to a set of TSP instances. Each instance corresponds to one meta-example in the meta-data. A meta-example is labeled by the performance of different MHs when applied to TSP instance. The meta-data is used by a ML technique to induce a meta-model.

In this work, meta-learning is addressed as a label ranking task [7]. In label ranking, the learning problem is to map the instances x from a dataset X to rankings $\succ_x$ (total strict orders) over a finite set of labels $\mathcal{L} = \{\lambda_1, ..., \lambda_m\}$,

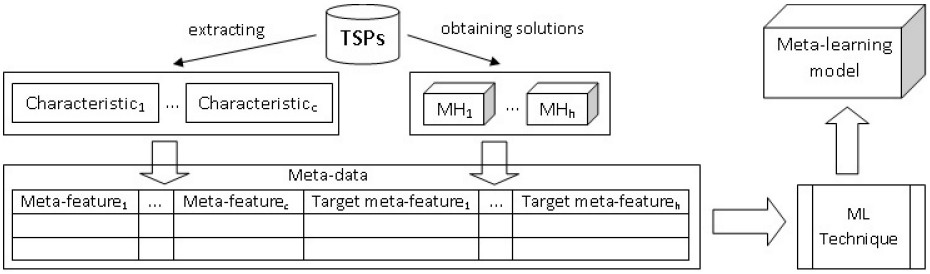

Fig. 1. Meta-learning approach to select meta-heuristics for the TSP

where $\lambda_i \succ_x \lambda_j$ means that, for instance x, label λ_i is preferred to λ_j. A ranking over $\mathcal{L}$ can be represented by a permutation as there exists a unique permutation τ such that $\lambda_i \succ_x \lambda_j$ iff $\tau(\lambda_i) < \tau(\lambda_j)$, where $\tau(\lambda_i)$ denotes the position of the label λ_i in the ranking. A survey on label ranking is presented in [23].

3 Training MLPs for Label Ranking

Since MLPs presented a good performance on a similar problem [19], we use them to rank MHs in this study. In a meta-learning context to rank labels, the input values of the MLP [18] are meta-feature values for a TSP instance. The output layer of the MLP produces a ranking of MHs for this TSP instance. The MH identified in the top position (i.e., rank 1) is the most promising one for this instance.

It is worth noting that the back-propagation algorithm is guided by a regression error measure (e.g., mean squared error) rather than a ranking accuracy measure (e.g., Spearman's correlation coefficient). However, by using a single network to learn the ranks of all labels, the weights to the output layer represent patterns that are specific to the corresponding label. On the other hand, given that there is a single set of weights to the hidden layer, they represent patterns in the data that are common to all the labels and act as latent features.

4 Recommendation Scenarios

Previous approaches [19,14] have considered a single scenario: the recommendation of MHs for instances in which the salesperson revisits current customers. We consider an additional scenario in which the meta-learning approach is used to recommend MHs for instances that contain new customers. To illustrate these scenarios, consider that a company visits clients in different cities and, for simplicity, that there exists only one client in each city. Thus, the recommendation scenarios investigated in our experiments are as follows:

***Revisiting customers* scenario.** The clients in the new instance are a subset of the ones that have been previously visited. Given instances concerning different subsets of the set of cities (e.g., {New York, Washington DC, Boston,

Philadelphia}), we would like to know the most promising MH in order to define a route to visit another subset of those cities (e.g., {New York, Boston, Philadelphia}). The intersection between the cities in different instances is non-empty.

***Prospect visits* scenario.** All clients on the new route have never been visited in previous routes. Given instances concerning different subsets of the set of cities (e.g., {New York, Washington DC, Boston, Philadelphia}), we would like to know the most promising MH in order to define a route to visit a different set of cities (e.g., {Edinburgh, London, Liverpool, Bristol}). The intersection between the sets of cities in new and old instances is empty.

5 Experimental Setup

A predictive ability of a learning model depends on the significant amount of instances used to train it [22]. We generated several TSP subproblems from benchmark instances extracted from the TSPLIB library [17].

Let $P = \{p_1, ..., p_k\}$ be the set of real TSP instances extracted from the TSPLIB library. A set of subproblems $S_i = \{s_{i,1}, ..., s_{i,z}\}$ can be generated from $p_i \in P$. Thus, the meta-data is a set of meta-examples $X^P = \{x_{1,1}, ..., x_{1,z}, ..., x_{k,1}, ..., x_{k,z}\}$, where each $x_{i,j}$ corresponds to $s_{i,j}$ that represents the j-th subproblem generated from the i-th instance of the real TSP, p_i. For each scenario, the TSP subproblems were generated as follows.

Revisiting customers scenario: 1000 TSP instances were generated from 10 TSP files, $P = \{d1655, fl1400, fnl4461, nrw1379, pcb3038, pr2392, rat783, rl1889, u1817, vm1748\}$. From each $p_i \in P$, 10 subproblems were generated for each of ten different quantities of cities (10, 20, ..., 100), resulting in $S_i = \{s_{i,1}, ..., s_{i,100}\}$.

Prospect visits scenario: 300 TSP instances were generated from 30 TSP files, $P = \{a280, berlin52, bier127, ch130, d1655, d15112, eil101, fl417, fl3795, fnl4461, kroA200, kroB100, kroC100, kroD100, kroE100, linhp318, lin318, nrw1379, p654, pcb3038, pr2392, rat783, rd400, rl1889, rl11849, ts225, tsp225, u1817, usa13509, vm1748\}$. The cities of each $p_i \in P$ were randomly distributed into 10 equal-sized sets. Each set of cities was used to generate a subproblem $s_{i,j}$ that belongs to $S_i = \{s_{i,1}, ..., s_{i,10}\}$.

Five MHs have been used in our experiments: Tabu Search (TS) [9], GRASP (GR) [6], Simulated Annealing (SA) [15], Genetic Algorithms (GA) [13], and Ant Colony (AC) [5]. The following parameter settings were used: TS: tabu list size = 2; number of iterations with no improvement of the current solution = 2; GR: number of iterations = 10; level of randomness and greedy search = 0.5; SA: initial temperature = 1; acceptance rate of neighbor solution = 0.9; cooling rate = 0.01; GA: PMX [10] as the crossover operator; population size = 20; mutation rate = 5%; elitism selection; AC: number of ants = 5; pheromone evaporation rate = 0.5; pheromone influence = 1; heuristic information influence = 1.

These parameter values were chosen after performing some preliminary experiments — just to ensure that every MH could find a reasonable solution for the

TSP instances in hand. Our goal was not to optimize the performance of each MH or promote any particular MH. Instead, we focus on the prediction of the ranking of MHs, with particular emphasis on how the user can take advantage of that ranking to get better solutions for TSP instances.

As these MHs are stochastic, every MH was run 30 times (with same processing time and different initial seeds) for each TSP instance. The average cost of the route of the 30 solutions was used as performance of each MH to compose the ranking of MHs.

Our MLP-based meta-learning models were trained with the standard *back-propagation* algorithm [1] and ten-fold *cross-validation* methodology [12]. We used 14 neurons in the input layer which correspond to 14 meta-features based on the measurements of edges and vertices proposed in [14]. The output layer has five neurons that identify the ranks of the five MHs for the TSP instance provided in the MLP input. The best configuration of the hidden layer is problem-dependent [12]. Therefore, we used the default number of hidden neurons proposed in [11]. For the multilabel classification, we use the binary classification method that has also been successfully applied to classify instances of TSP [14].

6 Experimental Evaluation

We compute the Spearman coefficient (r_S) [21] for every pair of $\langle predicted, ideal \rangle$ (vectors of) ranking and then we average the results. These results are compared to a baseline (the average ranking over the whole dataset [3]). In order to analyze if the performance difference between the proposed approach and baseline is significant, results of the statistical t-test are presented.

Top-N strategy [3] was used to compare the results of our ranking-based approach with the multilabel approach. This strategy evaluates the ranking of MHs by assessing the compromise between the quality of the solutions (cost of the routes) and the cost to obtain them (run time). The quality of the solutions provided by the top-N MHs is given by the best solution among those generated by all MHs. The cost is computed as the sum of the run times of those MHs. As the multilabel classification model does not suggest a ranking of MHs, its average performance is identified by a single point in figures 2a and 2b.

6.1 Experimental Results

Considering r_S as measure to evaluate the predictive models performance, our meta-learning based approach provided good ranking predictions. In particular, average scores $\bar{r_S} = 0.96$ and $\bar{r_S} = 0.93$ were obtained for the scenarios: *revisiting customers* and *prospect visits*, respectively, whereas the baseline model obtained $\bar{r_S} = 0.89$ and $\bar{r_S} = 0.83$, respectively. By applying the t-test to compare the $\bar{r_S}$ values, p-values of 7.15×10^{-42} and 2.61×10^{-12} were obtained for the respective scenarios. These results show that at the 95% confidence level, the performance of the proposed model is significantly better than the baseline model.

[1] Using the default values of the nnet package (R programming language).

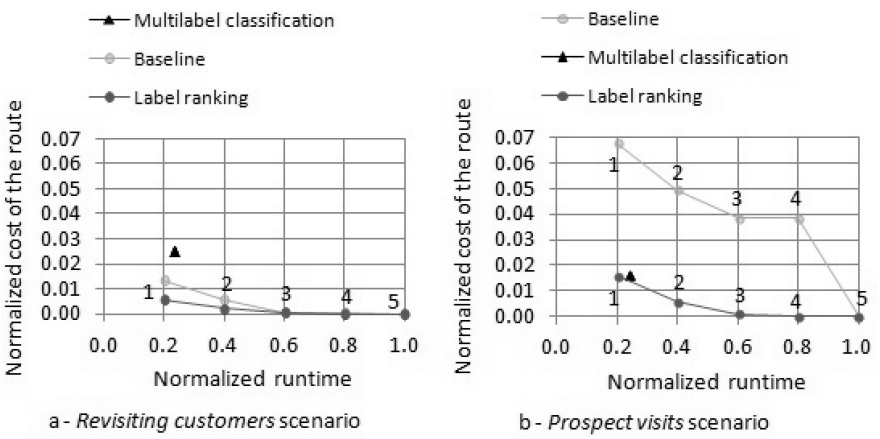

Fig. 2. Normalized average cost of the route versus normalized average runtime for the strategy of running the Top-N MHs for two real-world scenarios

It is hard for the meta-learning approach to achieve significantly higher accuracy than the baseline when predicting the most frequent rankings of MHs. These rankings are usually the most similar to the average ranking. The gain of meta-learning becomes clear in the least frequent rankings. In the *revisiting customers* scenario, the label ranking approach was better than the baseline on six of the seven rankings that were observed only once. For all the different rankings of MHs observed in the *prospect visits* scenario, the label ranking approach presented $\bar{r_S} > 0.6$, while the baseline model achieved this performance in 43% of those rankings.

Figure 2 shows the results for the Top-N strategy. In both scenarios, the Top-1 MH recommended by label ranking model provided a better solution than the Top-1 MH suggested by baseline model. The results for the *revisiting customers* scenario (Figure 2a) show that it is necessary run the Top-2 MHs indicated by the baseline to obtain a solution as good as those provided only by the Top-1 MH of the label ranking. The main advantage of using meta-learning model is the time required to obtain the solution. The time to run the MH, which is in the top position recommended by the meta-learning model, is 50% lower than the time to run the MHs in the first two positions of the baseline ranking. The average solution of the MHs recommended by the multilabel classification is worse than the solution generated by the Top-1 MH of our model. This is due to the fact that the MHs classified for some instances are not the best ones.

For the *prospect visits* scenario, the strategy of running the Top-1 MH suggested by the proposed approach provided a better solution compared to that obtained after processing the four most promising (Top-4) MHs from the baseline ranking — see Figure 2b. The average solution of the MHs ranked by the baseline model is worst for this scenario, in relation to the previous scenario, due to the increase in the number of different rankings observed in the

meta-data. The multilabel classification model recommends MHs whose solutions are as good as those provided by Top-1 MH suggested by the label ranking. However, every MH recommended by multilabel classification must be performed to indicate the solution for a given instance, requiring a longer processing time.

The model based on label ranking allows the user to obtain a good solution by running only the Top-1 MH. In practical situations where the user has enough time to run the other recommended MHs, even better solutions can be obtained.

7 Final Remarks

In this study, we addressed the problem of choosing the best MH for a given instance of the TSP. We use a meta-learning approach, which consists of learning a model that relates the properties of the TSP instances with the performance of MHs. We use an adaptation of the MLP for label ranking. Our results show that it is possible to predict the ranking of MHs and that, by following the recommendations in the rankings, it is possible to obtain good quality solutions when compared to simpler selection strategies. In particular, the comparison with a multilabel classification approach to the same problem additionally shows the advantage of addressing the problem as a label ranking task.

We consider two different scenarios: *re-visiting customers*, in which the new instances which we want to select the algorithms for share cities with the instances in the training meta-data; and *prospect customers*, in which the cities in new instances are new. Our results indicate that the latter type of scenario is harder to learn. This is expected because, in the *re-visiting customers* scenario, we cannot really say that the test instances are independent from the training instances because they share parts of their structure. However, since both scenarios may be true in practice, our work shows that it is important to investigate them separately. As future work, we will investigate new meta-features, following different approaches, such as adapting subsampling landmarkers [2].

Acknowledgment. The authors acknowledge CAPES, CNPq, FAPESP, FAPEAM and FCT.

References

1. Applegate, D., Bixby, R., Cook, W.: The Traveling Salesman Problem: A Computational Study. Princeton University Press, New Jersey (2006)
2. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning: Applications to Data Mining. Springer, Berlin (2009)
3. Brazdil, P., Soares, C., Costa, J.: Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. Machine Learning 50, 251–257 (2003)
4. Dekel, O., Manning, C.D., Singer, Y.: Log-Linear Models for Label Ranking. In: Advances in Neural Information Processing Systems. MIT Press (2003)
5. Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation 1(1), 53–66 (1997)

6. Feo, T., Resende, M.: Greedy randomized adaptive search procedures. Journal of Global Optimization 6, 109–133 (1995)
7. Fürnkranz, J., Hüllermeier, E., Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Mach. Learn. 73, 133–153 (2008)
8. Gendreau, M., Potvin, J.Y.: Handbook of Metaheuristics, 2nd edn. Springer Publishing Company, Incorporated (2010)
9. Glover, F., Taillard, E., Taillard, E.: A user's guide to tabu search. Annals of Operations Research 41, 1–28 (1993)
10. Goldberg, D., Lingle Jr., R.: Alleles, loci, and the traveling salesman problem. In: International Conference on Genetic Algorithms and Their Applications, pp. 154–159 (1985)
11. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11(1), 10–18 (2009)
12. Haykin, S.: Neural networks and learning machines, 3rd edn. Pearson Education Inc., New York (2009)
13. Holland, J.: Genetic algorithms and the optimal allocations of trial. SIAM J. Comp. 2, 88–105 (1973)
14. Kanda, J., Carvalho, A., Hruschka, E., Soares, C.: Selection of algorithms to solve traveling salesman problems using meta-learning. International Journal of Hybrid Intelligent Systems 8(3), 117–128 (2011)
15. Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by simulated annealing. Science 220, 671–680 (1983)
16. Papadimitriou, C.H.: The euclidean traveling salesman problem is np-complete. Theoretical Computer Science 4(3), 237–244 (1977)
17. Reinelt, G.: TSPLIB - a traveling salesman problem library. ORSA Journal on Computing 3, 376–384 (1991)
18. Rumelhart, D., Hinton, G., Williams, R.: Learning internal representations by error propagation. In: Rumelhart, D., McClelland, J. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
19. Smith-Miles, K., van Hemert, J., Lim, X.Y.: Understanding TSP Difficulty by Learning from Evolved Instances. In: Blum, C., Battiti, R. (eds.) LION 4. LNCS, vol. 6073, pp. 266–280. Springer, Heidelberg (2010)
20. Smith-Miles, K., Lopes, L.: Review: Measuring instance dificulty for combinatorial optimization problems. Comput. Oper. Res. 39(5), 875–889 (2012)
21. Spearman, C.: The proof and measurement of association between two things. American Journal of Psychology 15, 72–101 (1904)
22. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education, Inc., Boston (2006)
23. Vembu, S., Gärtner, T.: Label ranking algorithms: A survey. In: Fürnkranz, J., Hüllermeier, E. (eds.) Preference Learning, pp. 45–64. Springer, Heidelberg (2011)
24. Vilalta, R., Drissi, Y.: A perspective view and survey of meta-learning. Artificial Intelligence Review 18, 77–95 (2002)
25. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1, 67–82 (1997)