Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Computing and Information Systems

School of Computing and Information Systems

4-2022

Algorithm selection for the team orienteering problem

Mustafa MISIR

Aldy GUNAWAN
Singapore Management University, aldygunawan@smu.edu.sg

Pieter VANSTEENWEGEN

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Artificial Intelligence and Robotics Commons, and the Theory and Algorithms Commons

Citation

MISIR, Mustafa; GUNAWAN, Aldy; and VANSTEENWEGEN, Pieter. Algorithm selection for the team orienteering problem. (2022). *Proceedings of the 22nd European Conference, EvoCOP 2022 Held as Part of EvoStar 2022, Madrid, Spain, April 20-22*. 13222, 33-45.
Available at: https://ink.library.smu.edu.sg/sis_research/7159

This Conference Proceeding Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

Algorithm Selection for the Team Orienteering Problem

Mustafa Mısır[1,2](✉), Aldy Gunawan[3], and Pieter Vansteenwegen[4]

[1] Duke Kunshan University, Suzhou, China
mustafa.misir@dukekunshan.edu.cn
[2] Istinye University, Istanbul, Turkey
[3] Singapore Management University, Singapore, Singapore
aldygunawan@smu.edu.sg
[4] KU Leuven, Leuven, Belgium
pieter.vansteenwegen@kuleuven.be

Abstract. This work utilizes Algorithm Selection for solving the Team Orienteering Problem (TOP). The TOP is an NP-hard combinatorial optimization problem in the routing domain. This problem has been modelled with various extensions to address different real-world problems like tourist trip planning. The complexity of the problem motivated to devise new algorithms. However, none of the existing algorithms came with the best performance across all the widely used benchmark instances. This fact suggests that there is a performance gap to fill. This gap can be targeted by developing more new algorithms as attempted by many researchers before. An alternative strategy is performing Algorithm Selection that will automatically choose the most appropriate algorithm for a given problem instance. This study considers the existing algorithms for the Team Orienteering Problem as the candidate method set. For matching the best algorithm with each problem instance, the specific instance characteristics are used as the instance features. An algorithm Selection approach, namely ALORS, is used to conduct the selection mission. The computational analysis based on 157 instances showed that Algorithm Selection outperforms the state-of-the-art algorithms despite the simplicity of the Algorithm Selection setting. Further analysis illustrates the match between certain algorithms and certain instances. Additional analysis showed that the time budget significantly affects the algorithms' performance.

1 Introduction

Orienteering is essentially a type of sports, concerned with moving form a starting location to an end location while visiting a number of intermediate points. The goal is to maximize the total score which is collected through those visited points, within a given time limit. Orienteering is approached as an optimization problem in a general context beyond sports as the Orienteering Problem (OP) [12,15]. From this perspective, OP is modeled as a routing problem.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

Unlike pure routing attached to the Traveling Salesman Problem (TSP), the OP also includes the Knapsack Problem (KP) due to score maximization under a time budget. When the goal is to determine multiple paths, achieved by multiple people or vehicles, between the given starting and end points, the problem is denoted as the Team OP (TOP) [4]. Tourist trip planning is an example, real-world use case of the TOP [46]. The NP-hard nature of the TOP [36] has been attracting many researchers and practitioners to devise effective algorithmic solutions, mostly in the form of algorithms without optimality guarantee. Despite these development efforts, there is not a single algorithm outperforming all other algorithms for the TOP, as is the case for many other search and optimization problems [48].

One way to deal with this need for developing "the best" TOP algorithm is to specify the most suitable algorithm for each TOP instance. Algorithm Selection (AS) [22,40] is a systematic and automated way of achieving this task. AS is a high-level, meta-algorithmic strategy, traditionally achieving the selection tasks by deriving performance prediction models. These models basically map a set of features, $\mathcal{F}$, characterizing the problem instances, $\mathcal{I}$, to the algorithms, $\mathcal{A}$, performance, $\mathcal{P}_{|\mathcal{I}|\times|\mathcal{A}|}$. The resulting models are used to predict the performance of the candidate algorithms on a given problem instance. Regarding the candidate algorithms, Algorithm Portfolios (AP) [14] focus on having diverse and complementary algorithm sets to choose from. Such sets can also be used through scheduling [18] without AS. The schedules assign time periods to the candidate algorithms to run on one or more processing units. With sufficient computational resources, each AP member can run in parallel, returning the overall best solution(s). Going back to the AS models, the algorithm(s) with the best expected performance can be utilized to solve the target instance. Unlike this traditional use of AS, there have been AS methods with additional components such as pre-solvers [49,50]. While AS can be used to specify the best possible per-instance algorithm, there is not a single or an ultimate AS strategy. Considering that AS can be designed by utilizing different sub-components such as using a specific machine learning method as well as setting different parameter values for those accommodated sub-components, its automation was also studied, i.e. designing the best AS system based on the given AS tasks [25].

The present study applies AS to the TOP for the first time. The goal is to benefit from the existing algorithms by placing an AS layer on top of them. The algorithm set consists of 27 algorithms while the instance set used for both training and testing has 157 instances. For performing AS, the most obvious benchmark specifying elements are employed as $\mathcal{F}$. An existing AS system designed as an algorithm recommender system, i.e. ALORS [27], is used for the automated selection duty. Its problem instance and algorithm analysis capabilities besides its success in automatically choosing algorithms or their components is further shown on varying problems [28,29,31,32,39]. The experimental results revealed that AS is able to outperform all the constituent algorithms with almost no additional effort. Besides the pure selection results, a dis-/similarity analysis is reported both on the algorithm and instance space. All these analysis comes from the results of ALORS.

In the remainder of the paper, Sect. 2 further explains ALORS and how it is exactly used for this TOP setting. Section 3 reports the AS performance results while delivering the aforementioned analysis on the algorithms and the instances. The paper is summarized and the outcomes are briefly discussed together with listing the future research plans in Sect. 4.

2 Method

ALORS is a per-instance Algorithm Selection (AS) system based on recommender systems. ALORS specifically accommodates Collaborative Filtering (CF) [42]. Unlike the majority of the AS methods, CF allows ALORS to be able to work with incomplete performance data, $\mathcal{P}$. Yet, the present study works with the complete performance data. Considering the varying performance criteria, ALORS use $\mathcal{P}$ with the algorithms' ranks on each instance for general applicability. Another difference of ALORS is the way of building the performance prediction models. The mapping of instance features, $\mathcal{F}$, to $\mathcal{P}$ is indirectly achieved.

For this purpose, ALORS follows an intermediate step of feature to feature mapping. The initial, source features are those problem features, $\mathcal{F}$. The features to be mapped, i.e. latent (hidden) features $\mathcal{F}_h$, are automatically extracted from $\mathcal{P}$. As these features are directly driven from $\mathcal{P}$, they are expected to fairly represent the algorithms' performance. In that sense, ALORS takes the initial feature space to a new and more reliable feature space. This feature extraction process is handled by a well-known Matrix Factorization (MF) approach from linear algebra, i.e. Singular Value Decomposition (SVD) [13]. SVD is commonly used in CF or for dimensionality reduction as a data preprocessing step.

SVD applied to $\mathcal{P}_{|\mathcal{I}| \times |\mathcal{A}|}$ returns two main matrices besides a diagonal matrix of singular values, Σ. Referring to the components of $\mathcal{P}$, the initial matrix, U, consists of the latent features representing the problem instances. The other matrix, V, represents the candidate algorithms. They have been earlier used effectively on different problem domains [26,30]. Those matrices together approximates to $\mathcal{P}$ with the specified matrix ranks, $k \leq min(|\mathcal{I}|, |\mathcal{A}|)$, as follows. In our setting, r denotes the number of latent features. Since the singular values are sorted, from larger to smaller, earlier dimensions are more important than the latter dimensions, so the features. Thus, r can be picked as a small value while having strong approximation to $\mathcal{P}$ with possible noise elimination.

$$\mathcal{P} = U \Sigma V^T \approx U_k \Sigma V_k^T$$

When the mapping model is used for a new problem instance, it simply predicts a new row for U. This row is multiplied with Σ and V as denoted above. The outcome will be the rank predictions for all the candidate algorithms. Then, the algorithm with the best predicted rank is utilized for the target problem instance. The complete pseudo-code is provided in Algorithm 1.

Algorithm 1: ALORS: AS functionality [27]

```
Input
    Performance matrix in ranks $\mathcal{P} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{A}|}$
    Initial representation $\mathcal{F} \in \mathbb{R}^{|\mathcal{I}| \times d}$ of the problem instances
    Features $f$ of the target problem instance
AS Model Generation
    Build matrices $U$ and $V$
    Build $\mathcal{E} = \{(f_i, U_i), i = 1 \dots |\mathcal{I}|\}$.
    Learn $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^k$ from $\mathcal{E}$ (for a given matrix rank $k$)
AS Prediction
    Compute $U_f = \Phi(f)$
    Return $\underset{j=1\dots|\mathcal{A}|}{\text{argmin}} \langle U_f, V_j \rangle$
```

3 Computational Results

The Team Orienteering Problem (TOP) benchmarks are originated from [4]. The benchmark set consists of 387 instances. As the present performance data is collected from [16], only a subset of the instances are utilized, ignoring the ones where all the tested algorithms return the solutions of the same quality. This exclusion leads to 157 instances. The algorithm set has 27 approaches to be chosen as listed in Table 1.

The referenced study [16] reports the best solution delivered among 20 runs by each algorithm on each instance besides the average spent CPU time. The rank data is derived both based on the performance and the computational consumption. The spent time is used to break ties when two algorithms return the solution of the same quality. In other words, faster algorithms have better ranks than the slower algorithms with the same performance in terms of the solution quality. The features are simply the TOP benchmark specifications including n: the number of nodes, m: the number of vehicles and t_{max}: the maximum duration of each route.

For mapping, ALORS uses Random Forests (RF) [3]. SVD is performed with the rank of $k = 3$. The selection performance evaluation is achieved through 10-fold cross validation (10-cv). In other words, the TOP instance set of 157 instances is partitioned into 10 mutually exclusive, equally-sized[1] subsets. Each time, 9 subsets are used for training while the remaining subset involving unseen instances during training is used for testing.

Figure 1 illustrates the rank variations of each candidate TOP algorithm. While HALNS happens to be the overall best algorithm, MSA, AuLNS, PSOMA and MS-LS follow it. GLS, SiACO and DACO are the worst performing algorithms. Additionally, SkVNS and FPR are the least robust options as their rank performances significantly differ across the TOP instances. The detailed performance results can be found in the aforementioned study where the performance data is taken from [16].

[1] Almost equally-sized as 157 is not integer divisible by 10.

Table 1. 27 TOP algorithms

Algorithm	**Variant**
Tabu Search (TS) [9–11]	TMH [43], GTP [1], GTF [1]
Variable Neighborhood Search (VNS) [17,33]	FVNS [1], SVNS [1], SkVNS [45]
Ant Colony Optimization (ACO) [7]	SACO [19], DACO [19], RACO [19], SiACO [19],
Particle Swarm Optimization (PSO) [21]	PSOiA [6]
Local Search (LS)	GLS [45], MS-ILS [47], MS-LS [47]
Simulated Annealing (SA)	MSA [24]
GRASP [37] + Path Relinking (PR) [38]	FPR [41], SPR [41]
Large Neighborhood Search (LNS) [35]	AuLNS [23], LNS [34], HALNS [16]
Evolutionary Algorithms (EAs)	Memetic Algorithm (MA) [2], UHGS [47], UHGS-f [47]
Harmony Search (HS) [8]	SHHS [44], SHHS2 [44]
Hybrid	PSOMA [5]
Others	Pareto Mimic Algorithm (PMA) [20]

Table 2 reports the selection performance. Oracle, a.k.a. Virtual Best Solver (VBS), denotes the optimal performance by choosing the best algorithm for each instance. Single Best (SB) which is HALNS, represents overall best algorithm as one algorithm is used for all the instances. The results show that AS achieved by ALORS reaches to the overall average rank of 3.93 while the best algorithm out of 27 candidates come with the rank of 4.77. Take note that the lower the rank, the better the performance. Additionally, when the performance is evaluated with respect to each instance set, ALORS delivers the average rank of 4.18. The average rank of HALNS gets much worse, i.e. 4.77. These values indicate the clear advantage of ALORS. Being said that Oracle achieves the average rank of 1.12 and 1.10 for the respective evaluations. This means that there is still a room for improvement, likely by extending the instance feature space.

Figure 2 reveals the contribution of each feature on the AS prediction model of ALORS. RF used in ALORS, is a decision-tree based ensemble strategy providing importance on each single feature in terms of Gini importance/index. The corresponding importance evaluation shows that t_{max} happens to be the significantly most critical feature contributing to the AS recommendation model. Yet, although m and n are illustrated as substantially less important features than t_{max}, they still affect the algorithms' performance ranks. Being said that the use of t_{max} in the selection decisions may not be dominant if the benchmark instances are further diversified in terms of t_{max}.

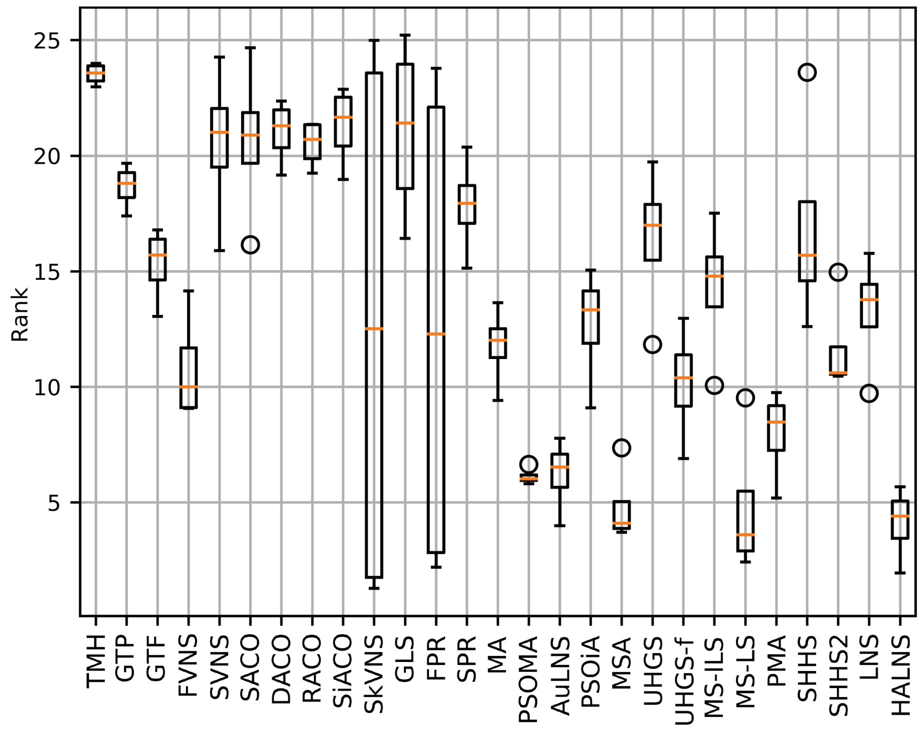

Fig. 1. The ranks of the TOP algorithms across all the instances

Table 2. The average ranks of each constituent algorithm besides ALORS where the best per-benchmark performances are in bold (#: the size of the instance set; SB: the Single Best solver; AVG: the average rank considering the average performance on each benchmark function; O-AVG: the overall average rank across all the instances)

Inst. Set (#)	**Oracle**	**SB (HALNS)**	**ALORS**
4 (54)	*1.19*	3.12	**3.02**
5 (45)	*1.07*	**5.18**	5.49
6 (15)	*1.00*	7.57	**5.23**
7 (43)	*1.13*	3.20	**2.99**
AVG	*1.10*	4.77	**4.18**
O-AVG	*1.12*	4.16	**3.93**

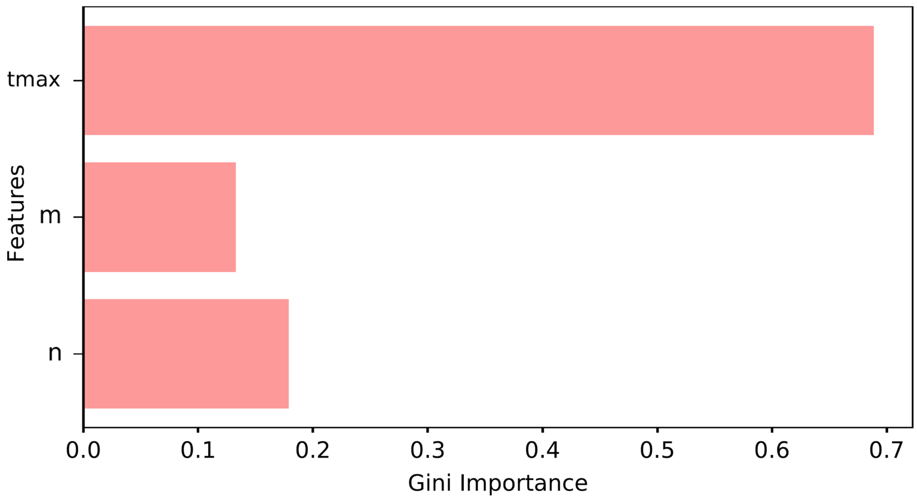

Fig. 2. Importance of the initial, hand-picked TOP instance features, using Gini Index/Importance

Further on the instance features, Fig. 3 visualizes the characterization of the instances through k-means clustering both considering those initial 3 basic features and 3 SVD originated latent features. The number of clusters, i.e. k, is determined by the best mean Silhouette score regarding the latent features. The features spaces are degraded into 2 via Multi-dimensional scaling (MDS). While the initial features yield rather clear instance distribution, the instance space looks more dispersed with the latent features. This is anticipated as the initial feature space is limited, only using 3 straightforward features. Instance by instance resemblance is depicted in Fig. 4.

Figure 5 visualizes the algorithms when they are hierarchically clustered, using the SVD ($k = 3$) driven features. For example, the leading single algorithm, i.e. HALNS, resembles to AuLNS and PMA. Essentially, in terms of performance HALNS and AuLNS similar as were discussed on Fig. 1. Unlike those comments, PMA is relatively poor compared both. Being said that the similarity comes from rank variation across the instances, not specifically the rank values. This aspect is reflected in Fig. 1 as the boxplot shapes are substantially similar.

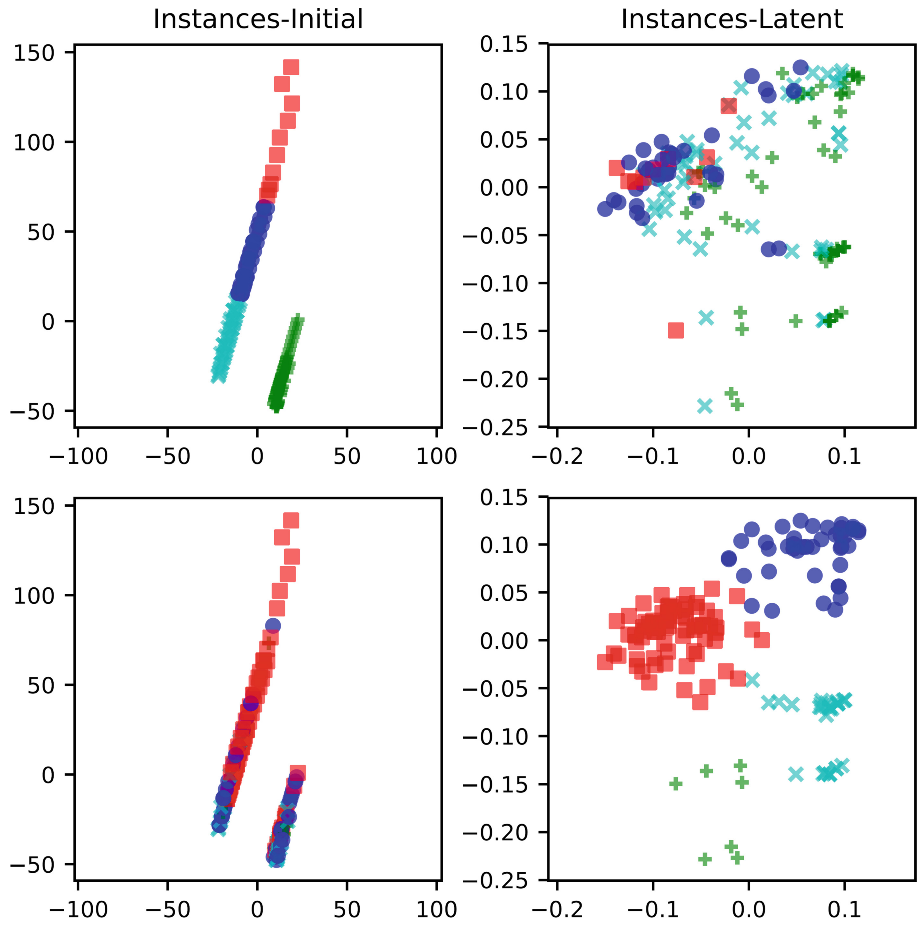

Fig. 3. Instances in 2-dimensional space using the initial and latent features (coloring is achieved by the initial features (top) and the latent features (bottom))

Besides these algorithms, PSOiA and LNS are determined as the most similar algorithms. Additionally, earlier mentioned least robust algorithms, i.e. SkVNS and FPR, are off the chart as they are cleary different than the rest, as shown in Fig. 1.

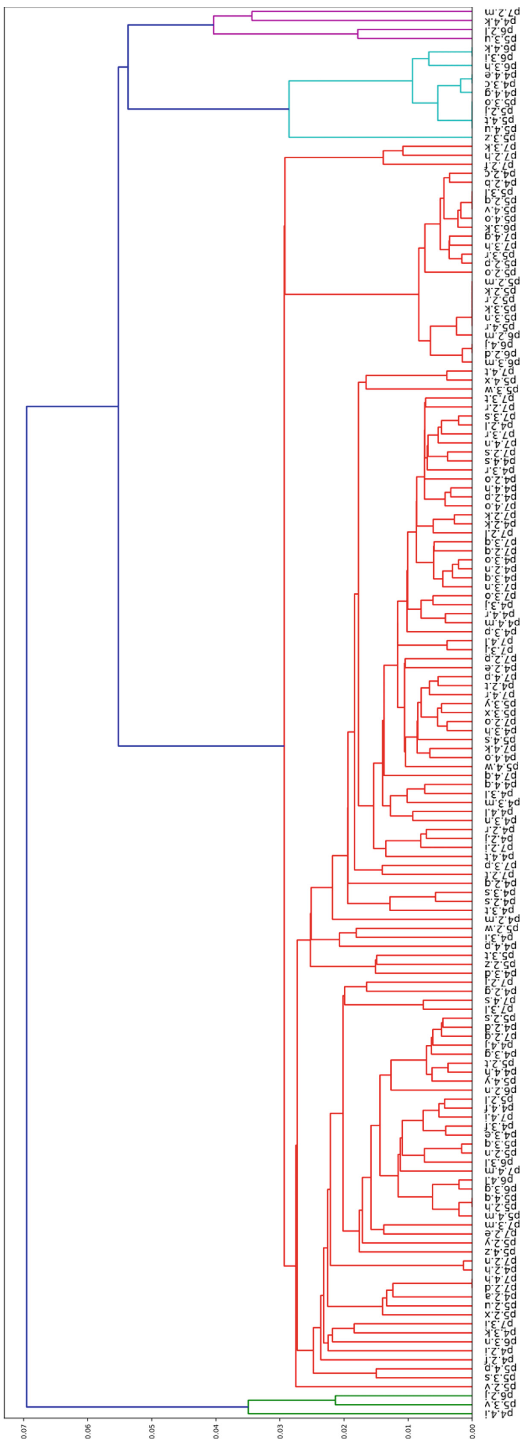

Fig. 4. Hierarchical clusters of instances using the latent features extracted from the performance data by SVD ($k = 3$)

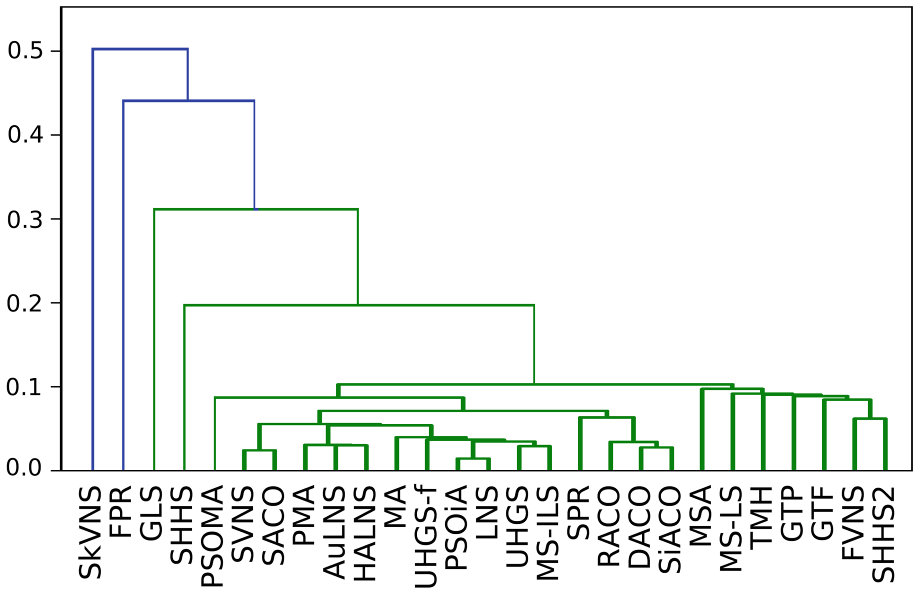

Fig. 5. Hierarchical clusters of algorithms using the latent features extracted from the performance data by SVD ($k = 3$)

4 Conclusion

Algorithm Selection (AS) is a meta-level approach allowing to benefit from multiple algorithms for solving a given target problem. Traditionally, AS delivers performance prediction models for a group of algorithms on an instance set, per-instance basis. This study applies AS to the Team Orienteering Problem (TOP), using an existing AS method named ALORS [27]. The task is to automatically determine the expectedly best algorithm among 27 candidate algorithms for any given TOP instance. Using the basic TOP benchmark specifications as the instance features, ALORS showed that all the constituent TOP algorithms are outperformed. Considering the diversity of the algorithms and the relatively large instance space, the reported dis-/similarity analysis provides a view both on the nature and characteristics of the algorithms and the instances.

This paper raises a series of research questions to be investigated. The reported research will be further extended by incorporating new features for strengthening the AS prediction quality, for example, whether nodes are clustered, nodes with higher scores are further from the start/end point, nodes with higher scores are clustered, and so on.

The next step will be achieved again on the feature space, yet targeting automated feature extraction. This idea will be realized by constructing images representing the TOP instances. The starting point will be using specific heuristics to deliver solution graphs. These graphs will then be given to Convolutional Neural Networks (CNNs) for feature engineering. Additionally, common graph based features will also be utilized. Finally, the computational results will be enriched by the other existing AS systems.

Acknowledgement. This study was supported by a Reintegration Grant project (119C013) of Scientific and Technological Research Council of Turkey (TUBITAK 2232).

References

1. Archetti, C., Hertz, A., Speranza, M.G.: Metaheuristics for the team orienteering problem. J. Heuristics **13**(1), 49–76 (2007). https://doi.org/10.1007/s10732-006-9004-0
2. Bouly, H., Dang, D.C., Moukrim, A.: A memetic algorithm for the team orienteering problem. 4OR **8**(1), 49–70 (2010). https://doi.org/10.1007/s10288-008-0094-4
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
4. Chao, I.M., Golden, B.L., Wasil, E.A.: The team orienteering problem. Eur. J. Oper. Res. **88**(3), 464–474 (1996)
5. Dang, D.-C., Guibadj, R.N., Moukrim, A.: A PSO-based memetic algorithm for the team orienteering problem. In: Di Chio, C., et al. (eds.) EvoApplications 2011. LNCS, vol. 6625, pp. 471–480. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20520-0_48
6. Dang, D.C., Guibadj, R.N., Moukrim, A.: An effective PSO-inspired algorithm for the team orienteering problem. Eur. J. Oper. Res. **229**(2), 332–344 (2013)
7. Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. IEEE Comput. Intell. Mag. **1**(4), 28–39 (2006)
8. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. SIMULATION **76**(2), 60–68 (2001)
9. Gendreau, M., Potvin, J.Y.: Tabu search. In: Burke, E.K., Kendall, G. (eds.) Search Methodologies, pp. 165–186. Springer, Boston (2005). https://doi.org/10.1007/0-387-28356-0_6
10. Glover, F.: Tabu search-part I. ORSA J. Comput. **1**(3), 190–206 (1989)
11. Glover, F.: Tabu search-part II. ORSA J. Comput. **2**(1), 4–32 (1990)
12. Golden, B.L., Levy, L., Vohra, R.: The orienteering problem. Nav. Res. Logist. (NRL) **34**(3), 307–318 (1987)
13. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solutions. Numer. Math. **14**(5), 403–420 (1970). https://doi.org/10.1007/BF02163027
14. Gomes, C., Selman, B.: Algorithm portfolio design: theory vs. practice. In: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI), Providence/Rhode Island, USA, 1–3 August 1997, pp. 190–197 (1997)
15. Gunawan, A., Lau, H.C., Vansteenwegen, P.: Orienteering problem: a survey of recent variants, solution approaches and applications. Eur. J. Oper. Res. **255**(2), 315–332 (2016)
16. Hammami, F., Rekik, M., Coelho, L.C.: A hybrid adaptive large neighborhood search heuristic for the team orienteering problem. Comput. Oper. Res. **123**, 105034 (2020)
17. Hansen, P., Mladenović, N., Todosijević, R., Hanafi, S.: Variable neighborhood search: basics and variants. EURO J. Comput. Optim. **5**(3), 423–454 (2016). https://doi.org/10.1007/s13675-016-0075-x

18. Kadioglu, S., Malitsky, Y., Sabharwal, A., Samulowitz, H., Sellmann, M.: Algorithm selection and scheduling. In: Lee, J. (ed.) CP 2011. LNCS, vol. 6876, pp. 454–469. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23786-7_35
19. Ke, L., Archetti, C., Feng, Z.: Ants can solve the team orienteering problem. Comput. Ind. Eng. **54**(3), 648–665 (2008)
20. Ke, L., Zhai, L., Li, J., Chan, F.T.: Pareto mimic algorithm: an approach to the team orienteering problem. Omega **61**, 155–166 (2016)
21. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN 1995-International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE (1995)
22. Kerschke, P., Hoos, H.H., Neumann, F., Trautmann, H.: Automated algorithm selection: survey and perspectives. Evol. Comput. **27**(1), 3–45 (2019)
23. Kim, B.I., Li, H., Johnson, A.L.: An augmented large neighborhood search method for solving the team orienteering problem. Expert Syst. Appl. **40**(8), 3065–3072 (2013)
24. Lin, S.W.: Solving the team orienteering problem using effective multi-start simulated annealing. Appl. Soft Comput. **13**(2), 1064–1073 (2013)
25. Lindauer, M., Hoos, H.H., Hutter, F., Schaub, T.: AutoFolio: an automatically configured algorithm selector. J. Artif. Intelli. Res. **53**, 745–778 (2015)
26. Mısır, M.: Matrix factorization based benchmark set analysis: a case study on HyFlex. In: Shi, Y., et al. (eds.) SEAL 2017. LNCS, vol. 10593, pp. 184–195. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68759-9_16
27. Mısır, M., Sebag, M.: ALORS: an algorithm recommender system. Artif. Intell. **244**, 291–314 (2017)
28. Mısır, M.: Algorithm selection across selection hyper-heuristics. In: The Data Science for Optimization (DSO)@ IJCAI Workshop at the 29th International Joint Conference on Artificial Intelligence (IJCAI) (2021)
29. Mısır, M.: Algorithm selection on adaptive operator selection: a case study on genetic algorithms. In: Simos, D.E., Pardalos, P.M., Kotsireas, I.S. (eds.) LION 2021. LNCS, vol. 12931, pp. 237–251. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92121-7_20
30. Mısır, M.: Benchmark set reduction for cheap empirical algorithmic studies. In: IEEE Congress on Evolutionary Computation (CEC), pp. 871–877. IEEE (2021)
31. Misir, M.: Generalized automated energy function selection for protein structure prediction on 2D and 3D HP models. In: IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–6. IEEE (2021)
32. Mısır, M.: Selection-based per-instance heuristic generation for protein structure prediction of 2D HP model. In: IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–6. IEEE (2021)
33. Mladenovic, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. **24**(11), 1097–1100 (1997)
34. Orlis, C., Bianchessi, N., Roberti, R., Dullaert, W.: The team orienteering problem with overlaps: an application in cash logistics. Transp. Sci. **54**(2), 470–487 (2020)
35. Pisinger, D., Ropke, S.: Large neighborhood search. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics. ISOR, vol. 146, pp. 399–419. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-1665-5_13
36. Poggi, M., Viana, H., Uchoa, E.: The team orienteering problem: formulations and branch-cut and price. In: 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2010). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2010)

37. Resende, M.G., Ribeiro, C.: Greedy randomized adaptive search procedures (GRASP). AT&T Labs Research Technical Report **98**(1), 1–11 (1998)
38. Resendel, M.G., Ribeiro, C.C.: Grasp with path-relinking: recent advances and applications. In: Ibaraki, T., Nonobe, K., Yagiura, M. (eds.) Metaheuristics: Progress as Real Problem Solvers. ORCS, vol. 32, pp. 29–63. Springer, Boston (2005). https://doi.org/10.1007/0-387-25383-1_2
39. Ribeiro, J., Carmona, J., Mısır, M., Sebag, M.: A recommender system for process discovery. In: Sadiq, S., Soffer, P., Völzer, H. (eds.) BPM 2014. LNCS, vol. 8659, pp. 67–83. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10172-9_5
40. Rice, J.: The algorithm selection problem. Adv. Comput. **15**, 65–118 (1976)
41. Souffriau, W., Vansteenwegen, P., Berghe, G.V., Van Oudheusden, D.: A path relinking approach for the team orienteering problem. Comput. Oper. Res. **37**(11), 1853–1859 (2010)
42. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009**, 4 (2009)
43. Tang, H., Miller-Hooks, E.: A tabu search heuristic for the team orienteering problem. Comput. Oper. Res. **32**(6), 1379–1407 (2005)
44. Tsakirakis, E., Marinaki, M., Marinakis, Y., Matsatsinis, N.: A similarity hybrid harmony search algorithm for the team orienteering problem. Appl. Soft Comput. **80**, 776–796 (2019)
45. Vansteenwegen, P., Souffriau, W., Berghe, G.V., Van Oudheusden, D.: A guided local search metaheuristic for the team orienteering problem. Eur. J. Oper. Res. **196**(1), 118–127 (2009)
46. Vansteenwegen, P., Souffriau, W., Berghe, G.V., Van Oudheusden, D.: Iterated local search for the team orienteering problem with time windows. Comput. Oper. Res. **36**(12), 3281–3290 (2009)
47. Vidal, T., Maculan, N., Ochi, L.S., Vaz Penna, P.H.: Large neighborhoods with implicit customer selection for vehicle routing problems with profits. Transp. Sci. **50**(2), 720–734 (2016)
48. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**, 67–82 (1997)
49. Xu, L., Hutter, F., Hoos, H., Leyton-Brown, K.: SATzilla: portfolio-based algorithm selection for SAT. J. Artif. Intell. Res. **32**(1), 565–606 (2008)
50. Xu, L., Hutter, F., Shen, J., Hoos, H., Leyton-Brown, K.: SATzilla 2012: improved algorithm selection based on cost-sensitive classification models. In: Proceedings of SAT Challenge 2012: Solver and Benchmark Descriptions, pp. 57–58 (2012)