Solving a multi objective shortest path problem

Antoine Giret, Yannick Kergosien, Emmanuel Neron

▶ **To cite this version:**

Antoine Giret, Yannick Kergosien, Emmanuel Neron. Solving a multi objective shortest path problem. 7th International Workshop on Freight Transportation and Logistics (ODYSSEUS 18), Jun 2018, cagliari, Italy. hal-01932625

HAL Id: hal-01932625
https://hal.science/hal-01932625

Submitted on 17 Feb 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Solving a multi objective shortest path problem

Antoine Giret
Université François Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305, Tours, France

Emmanuel Néron
Université François Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305, Tours, France

Yannick Kergosien
Université François Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305, Tours, France
Email: yannick.kergosien@univ-tours.fr

1 Introduction

The shortest path problem is a well-known problem in combinatorial optimization [8]. Since the first studies [9] and [11] – conducted more than 60 years ago – several variants of this problem are still being studied. The Multi-Objective Shortest Path (MOSP) problem is defined as follows: let $G = (\mathcal{V}, \mathcal{A})$ be a directed graph where $\mathcal{V}$ is the set of vertices and $\mathcal{A}$ the set of arcs. c_{ij}^k denotes the cost associated with the arc $(i, j) \in \mathcal{A}$ for the criterion $k \in \mathcal{K}$, with $\mathcal{K}$ the set of criteria. We suppose that $c_{ij}^k \geq 0$. The problem consists in finding a set of paths $\mathcal{P}$ from a source node s to a target node t minimizing several sum-type objectives functions. The result of the MOSP problem is a set of strictly non-dominated paths also called Pareto front. An efficient label-correcting algorithm – called Label-Correcting with Dynamic update of Pareto Front (LCDPF) – is proposed to solve the MOSP problem. This work is an extension of [10]. Computational experiments show that the LCDPF algorithm matchs or outperforms the recent algorithms in the literature.

2 Recent advances in MOSP with sum-type objectives

In 2010, [13] proposes a label-correcting and label-setting algorithm (bLSET) using an acceleration technique improving the efficiency of these types of algorithm. He showed that it is not always necessary to propagate a label to the target node to confirm that it is dominated. According to [16], bLSET algorithm presented the best computational times among labelling approaches proposed during this period. Recently, [14] proposed a new

exact method, called Pulse algorithm, for the bi-objective shortest path problem and large-scale road networks. Pulse algorithm is based on recursive method using pruning strategies that speed up the graph exploration. The results show that the proposed algorithm outperforms the bLSET algorithm on very-large scale instances from the DIMACS dataset. In [15], the authors proposed a Dijkstra-like method generalizing the original method to only determine all extreme supported solutions of the bi-objective shortest path problem. The authors proved that the running times of the proposed method are $\mathcal{O}$(N(m+n log n)) with n is the number of nodes, m is the number of arcs and N is the number of extreme supported points in the outcome space. Finally, [17] proposed a lower bound set calculation method for the bi-objective shortest path problem. The proposed lower bounds aim to improve the $NAMOA*$ method [18] which is an exact generalization of an $A*$ method to the multi-objective one-to-one problem.

3 The LCDPF algorithm

The LCDPF algorithm is composed of two phases, similarly to the general two-phase method introduced by [12] for solving bi-objective problem. The first phase consists in two steps. The goal of the first step is to determine some initial solutions belonging to the final Pareto front (the supported solutions). The second step computes partial paths, from pertinent nodes of the graph to the target node, in order to deduce some upper and lower bounds. This step also allows to remove useless nodes from the graph (nodes for which none non-dominated path in the final Pareto front solution passes through them). Both steps are based on a set of resolutions of reversed mono-objective shortest path problems using dijkstra's algorithm of [11]. The second phase consists in finding all non-dominated solutions by exploiting informations obtained in the previous phase. To do this, the proposed algorithm is based on a classical label-correcting algorithm which has been introduced in [19] and for which we developed additional improvements to make the algorithm more efficient. The main specificity of the LCDPF algorithm is that the final Pareto front is dynamically built all along the process and not necessary when the target node is reached. This dynamic update allows to obtain a better estimation of the final Pareto front during the search and so the classic techniques to prune labels (partial solutions) are more efficient.

4 Computational Experiments

In order to compare our results with benchmark algorithms solving the MOSP problem, computational experiments have been performed on instances from the 9^{th} DIMACS challenge. It consists of 3 graphs that represent New York City (264,346 nodes, 733,846 arcs), San Francisco Bay Area (321,270 nodes, 800,172 arcs) and the state of Florida (1,070,376

nodes, 2,712,798 arcs). For each comparison, we used the same pairs of source and target nodes. We developed our algorithm in C++ unsing the Boost Graph Library. Experiments are performed on a Linux machine with an Intel Xeon 2.67GHz, 8 cores and 8GB of RAM. First, we compared the LCDPF algorithm to the some adaptations of the NAMOA* algorithm proposed by [17] (KDLS) and for which they used the NYC graph. Table 1 compares the better execution times among the algorithms proposed by [17] (columns KDLS) and the execution times of LCDPF algorithm (column LCDPF). The $|S|$ column indicates the number of non-dominated solutions on the Pareto front for each instance.

#	KDLS	LSDPF	$\|S\|$	#	KDLS	LSDPF	$\|S\|$	#	KDLS	LSDPF	$\|S\|$
1	499.2	**0.1**	1,089	8	-	**2.1**	7,391	15	0.7	**0.4**	1
2	21,363.7	**0.6**	1,469	9	3,379.2	**1.4**	919	16	5,201.4	**4.4**	2,034
3	4.2	**0.4**	16	10	722.7	**2.0**	774	17	7,668.2	**0.9**	1,724
4	-	**3.9**	5,121	11	346.5	**0.5**	631	18	1,135.3	**0.4**	1,276
5	15,710.4	**0.6**	2,451	12	14,272.4	**2.0**	1,573	19	-	**1.5**	4,224
6	5,697.4	**1.2**	1,502	13	21,782.2	**0.6**	3,046	20	-	**1.9**	3,262
7	129.7	**0.4**	272	14	18,693.8	**0.4**	2,957				

Table 1: Comparison to [17]

Table 1 shows that execution times of algorithms proposed by [17] (with a time limit equals to 12h) are bigger than LCDPF algorithm. LCDPF algorithm may have an execution time of 12000 times faster than the best others algorithms. Computation times of LCDPF algorithm remain below the 5 second which is reasonable.

We also compared LCDPF algorithm to the bounded label-setting algorithm (blSET) proposed by [13] and the Pulse algorithm proposed by [14]. As presented in [14], the 30 instances of each graph has been clustered into the same three equal sized groups, denoted S (small), M (medium) and L (large), based on the number of non-dominated solutions found in the Pareto front. Table 2 compares average execution times between blSET, Pulse and LCDPF algorithms. Table 2 shows that execution times of blSET algorithm are in average bigger than other algorithms. Regarding the Pulse and LCDPF algorithms, the Pulse algorithm is particularly efficient on instances with a small number of solutions. For these instances, LCDPF is in average 1 to 4 times less effective to find the entire Pareto front, but execution times remain reasonable (< 2 seconds). LCDPF algorithm is much more efficient than the two others on medium to large instances. Indeed, it is in average 20 times faster than the Pulse algorithm and 120 faster than the blSET algorithm.

Finally, we compared our algorithm to the Ratio-Labeling BSP (RLBSP) algorithm proposed by [15] to find the supported extreme non-dominated solutions only. The authors also proposed a modified bi-objective label-setting algorithm for finding this type of solutions (called SLSET). Table 3 compares execution times between RLBSP, SLSET

and LCDPF* algorithms. LCDPF* algorithm is based on LCDPF algorithm that has been modified to find only the supported extreme non-dominated solutions. The results of Table 3 shows that LCDPF* algorithm has a better execution times in average than the others algorithms.

	blSET	pulse	LCDPF	$\|S\|$
NY S	62,392.4	**315.2**	378	34.1
NY M	513,945.8	131,304	**1,131**	163
NY L	8,812,59.6	1,367,664	**2,849.5**	422.7
BAY S	6,777.7	**160.1**	381	8.8
BAY M	61,950	9,326.7	**484**	57.2
BAY L	317,432.3	105,549.2	**2,302**	171.8
FLA S	330,122.3	**349.8**	1,389	14.7
FLA M	562,195.9	348,114.7	**1,647**	116.1
FLA L	2,627,432.9	888,586	**3,045**	552.3

Table 2: Comparison to [13] and [14]

	SLSET	RLBSP	LSDPF*
NY	16,450	2,300	**791.9**
BAY	10,720	1,980	**536.9**
FLA	119,170	12,800	**1,878.4**

Table 3: Comparison to [15]

5 Conclusion

An efficient label-correcting algorithm – called Label-Correcting with Dynamic update of Pareto Front (LCDPF) – is proposed to solve the MOSP problem. Even if the graph is large, the LCDPF algorithm is able to find all non-dominated solutions in a short computing time, compared to best benchmark algorithms for real road networks. It composes of best improvement techniques that we can find in the literature about this problem and it integrates new ones referred later as: (1) quickly update of the Pareto front during the search that allows to efficiently prune partial paths of the search space and (2) remove useless nodes from the graph in a preprocessing phase.

References

[1] E. Prescott-Gagnon, G. Desaulniers, M. Drexl and L.-M. Rousseau, "European driver rules in vehicle routing with time windows", *Transportation Science* 44(4), 455-473, 2010.

[2] B.L. Golden, S. Raghavan and E.A. Wasil, *The Vehicle Routing Problem: Latest Advances and New Challenges*, Springer, 2008.

[3] J.-F. Cordeau, G. Laporte, J.-Y. Potvin and M.W.P. Savelsbergh, "Transportation on demand", in *Handbooks in Operations Research and Management Science*, C. Barnhart and G. Laporte (eds), vol. 14, 429-466, Elsevier, Amsterdam, 2007.

[4] J.-M. Belenguer, E. Benavent, C. Prins, C. Prodhon and R. Wolfler-Calvo, "A branch and cut method for the capacitated location-routing problem", *Proceedings of the International Conference on Service Systems and Service Management*, 1541-46, 2007.

[5] M.S. Parragh, *Ambulance Routing Problems with Rich Constraints and Multiple Objectives*, Ph.D. thesis, University of Vienna, 2009.

[6] C. Archetti, L. Bertazzi, A. Hertz and M.G. Speranza, "A hybrid heuristic for an inventory-routing problem", Technical Report 317, Department of Quantitative Methods, University of Brescia, 2009.

[7] H.L. Petersen, A. Larsen, O.B.G. Madsen and S. Røpke, "The simultaneous vehicle scheduling and passenger service problem", Tristan VII, Tromsø, June 2010.

[8] N. Deo and C. Pang, "Shortest-path algorithms: Taxonomy and annotation", *Networks* 14 (2), 275-323, 1984.

[9] R. Bellman, "On a routing problem", *Quarterly of applied mathematics* 16 (1), 87-90,1958.

[10] G. Sauvanet and E. Neron, "Search for the best compromise solution on Multiobjective shortest path problem", *Electronic Notes in Discrete Mathematics* 36, 615-622, 2010.

[11] E.W. Dijkstra, "A note on two problems in connection with graphs", *Numerische Mathematik* 1, 269-271, 1959.

[12] E.L. Ulungu and J. Teghem, "The two phases method : An efficient procedure to solve biobjective combinatorial optimization problems", *Foundations of Computing and Decision Sciences* 20 (2), 149-165,1995.

[13] R. Andrea, "Speed-up of labelling algorithms for biobjective shortest path problems", in *Proceedings of the 45th annual conference of the ORSNZ. Auckland, New Zealand*, 313-322, 2010.

[14] D. Duque, L. Lozano and A.L. Medaglia, "An exact method for the biobjective shorstest path problem for large-scale road networks", *European Journal of Operational Research* 242, 788-797, 2015.

[15] S.N. Antonio and R. Andrea, "A Dijkstra-like method computing all extreme supported non-dominated solutions of the biobjective shortest path problem", *Computers & Operations Research* 57, 83-94, 2015.

[16] S. Demeyer, J. Goedgebeur, P. Audenaert, M. Pickavet and P. Demeester, "Speeding up Martins? algorithm for multiple objective shortest path problems", *4OR* 11 (4), 323-348, 2013.

[17] E. Machuca and L. Mandow, "Lower bound sets for biobjective shortest path problems", *Journal of Global Optimization* 64 (1), 63-77, 2016.

[18] L. Mandow and J.L.P. De La Cruz, "Multiobjective A* search with consistent heuristics", *Journal of the ACM (JACM)* 57 (5), 27, 2010.

[19] E.Q.V. Martins, "On a multicriteria shortest path problem", *European Journal of Operational Research* 16, 236-245, 1984.