

Title: Heuristic solution of the Elementary Shortest Path Problem with Resource Constrained
Group: Algorithms and Optimization
Prerequisite: Videregående Algoritmik
Contact: David Pisinger pisinger@diku.dk

In the Elementary Shortest Path Problem with Resource Constrained (ESPPRC) a shortest path in a graph G with possible negative edges and cycles is to be found. The path must obey a certain set of resource constraints. Some examples of resource constraints are: A capacity resource where each node is assigned a weight and the sum of the chosen nodes may not exceed a global capacity. A time window resource where each node can only be visited in a certain time interval. An example of an ESPPRC instance can be seen in figure 1. Dror [5] has shown that ESPPRC is NP-hard in the strong sense. A description

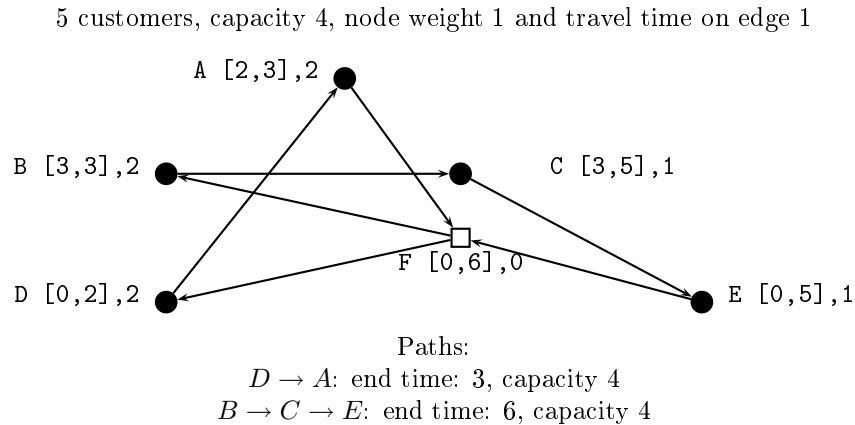


Figure 1: A ESPPRC instance example

of the ESPPRC problem and an exact algorithm can be found in Feillet et al. [1]

Solution Method

Adaptive Large Scale Neighborhood Search (ALNS) has successfully been used to solve several variants of the Vehicle Routing Problem, see Pisinger and Røpke [7]. The method is iterative and starts with a feasible solution. By using a random combination of several fast destruction and construction heuristics it destroys and rebuilds the solution differently in each iteration.

Research approach

Your task is to describe the theoretic results regarding the ESPPRC as well as the general ALNS framework. Finally you should construct and implement your own ALNS algorithm for the ESPPRC.

There are several instances of the ESPPRC with a single capacity resource, as well as instances with a single capacity and a single time window available. So you may restrict your implementation to these. So you may restrict your implementation to these, but the theory should be as general as possible.

Title: Exact solution of the Elementary Shortest Path Problem with Resource Constrained
Group: Algorithms and Optimization
Prerequisite: Videregående Algoritmik
Contact: David Pisinger pisinger@diku.dk

In the Elementary Shortest Path Problem with Resource Constrained (ESPPRC) a shortest path in a graph G with possible negative edges and cycles is to be found. The path must obey a certain set of resource constraints. Some examples of resource constraints are: A capacity resource where each node is assigned a weight and the sum of the chosen nodes may not exceed a global capacity. A time window resource where each node can only be visited in a certain time interval. An example of an ESPPRC instance can be seen in figure 2.

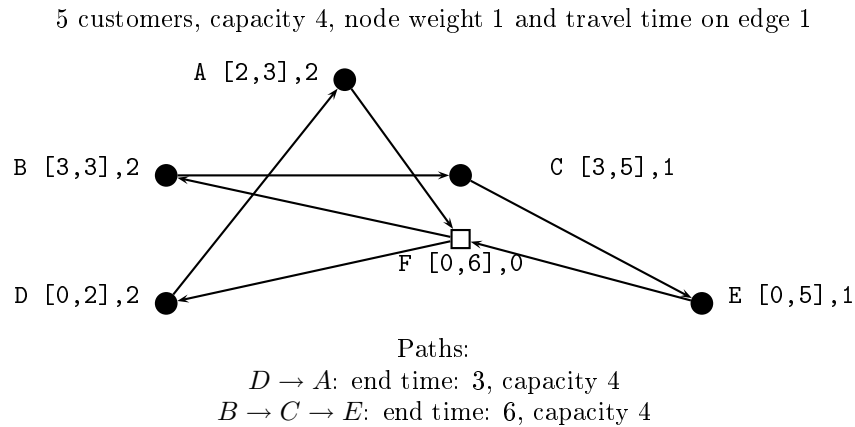


Figure 2: A ESPPRC instance example

Solution Method

One solution method for the ESPPRC is a so called “label setting algorithms”. A label setting algorithm for ESPPRC basically enumerates all possible paths. During the enumeration the different partial path (Labels) are compared and if it can be proven some of these are removed. The step of removing labels is called dominance and is very time consuming.

Feillet et al.[1] describe how to construct a label setting algorithm and Righini and Salani [8] proposed a bi-directional algorithm based on this label setting algorithm.

Research approach

Your task is to describe and implement either the standard label setting algorithm or the bi-directional algorithm. Secondly you are to consider different ways of speeding up the dominance. To speed up dominance you may consider different ways of storing the labels and/or consider various ways of calculating upper and lower bounds for the value of any path that may include the partial path the label represent.

You should implement and test the new storing methods and discuss the pros and cons caused by these.

There are several instances of the ESPPRC with a single capacity resource, as well as instances with a single capacity and a single time window available. So you may restrict your implementation to these.

Title: The cutting stock problem
Group: Algorithms and Optimization
Prerequisite: Videregående Algoritmik og Introduktion til optimering
Contact: David Pisinger pisinger@diku.dk

Several materials such as paper, textiles and metallic are manufactured in rolls of large widths called raws. The raws are later cut into smaller pieces (finals) which are used to manufacture a given item. In the cutting stock problem a demand for several finals is given and the task is to find a set of cuttings that minimizes the total number of whole raws. On figure 3 a example of a cutting is seen.

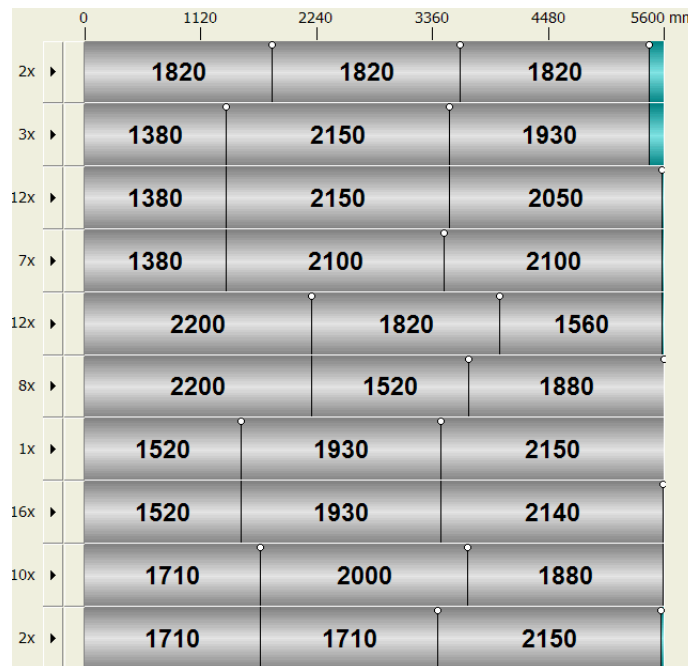


Figure 3: A cutting stock example

Solution method

Branch and Cut and Price (BCP) is a solution method used for solving large scale optimization problems. The BCP method embeds the Danzig-Wolfe decomposition principle (see [2, 9]) in a branch and bound framework. The Danzig-Wolfe decomposition principle splits an optimization problem into a master and

a pricing problem. For the cutting-stock problem the master problem becomes an integer optimization problem and the pricing problem becomes a knapsack problem.

Research Approach

Your task is to describe the general Branch and Cut and Price theory and the decomposition of the cutting stock problem. Make an implementation of a BCP algorithm for the cutting stock problem. Experimentally investigate if it is possible to use the reoptimization strategy for the knapsack problem suggested by Pisinger and Saidi [4] in the BCP algorithm.

Alternative to the reoptimization strategy you may also consider how cuts in the master problem can be handle in the pricing problem. You task is then to transfer the work of Petersen, Pisinger and Spoorendonk [6] to the cutting stock problem.

An implementation of the knapsack algorithm with reoptimization is available as well as the open source Branch and Price framework COIN.

Title: Advanced algorithms for solving the Shortest Path Problem
Group: Algorithms and Optimization
Prerequisite: Algoritmer og datastrukturer Videregående Algoritmik
Contact: David Pisinger pisinger@diku.dk

Given a graph $G(V,E)$ the well known shortest path problem can be solved in $O(E \log V)$ (see CLRS [3]) when the graph contains no negative edges. In the general case when negative edges exist the problem can be solved in $O(VE)$ with the Bellman-Ford algorithm.

Research approach

There exist several theoretic results regarding the shortest path problem with negative edges that improves the running time of the Bellman Ford algorithm. And there exist several approaches that improves the practical running time of the Dijkstra algorithm. Your task is to make a survey of both the theoretical and practical algorithms. You may choose to implement some of the algorithms but this is not required. Some inspiration can be found in Andrew V. Goldbergs lecture slides from the summer school Path05 (<http://www.avglab.com/andrew/pub/path05-slides.pdf>)

References

- [1] An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Netw.*, 44(3):216–229, 2004.
- [2] V. Chvatal. *Linear Programming*. W. H. Freeman and Company, 1983.
- [3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. Addison, 2001.
- [4] Alima Saidi David Pisinger. Tolerance analysis for pseudo-polynomial problems. Technical report, DIKU, 2008. Working paper.
- [5] M. Dror. Note on the complexity of the shortest path models for column generation in vrptw. *Operations Research* 42, pp. 977-979, 1994.
- [6] B. Petersen, D. Pisinger, and S. Spoorendonk. Chvátal-gomory rank-1 cuts used in a dantzig-wolfe decomposition of the vehicle routing problem with time windows. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, 2007. forthcoming.

- [7] David Pisinger and Stefan Ropke. A general heuristic for vehicle routing problems. *Comput. Oper. Res.*, 34(8):2403–2435, 2007.
- [8] G. Righini and M. Salani. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Elsevier Science (submitted), 2004.
- [9] L. A. Wolsey. *Integer Programming*. A-Wiley-Interscience Publication, 1998.