HyMeHeu - A Hybrid Metaheuristic Solver for the Directed Feedback Vertex Set Problem

Maria Bresich
Algorithms and Complexity Group, TU Wien, Vienna, Austria

Günther Raidl
Algorithms and Complexity Group, TU Wien, Vienna, Austria

Johannes Varga
Algorithms and Complexity Group, TU Wien, Vienna, Austria

Abstract

The HyMeHeu-solver is a hybrid metaheuristic solver for the directed feedback vertex set problem. It uses a construction heuristic based on a greedy function to generate an initial solution which is further improved by applying a simple local search procedure to it. A large neighborhood search with a pair of a destroy and repair method is employed as general metaheuristic framework. The hybridization is achieved by using a mixed integer programming solver in the repair operator.

2012 ACM Subject Classification Mathematics of computing → Combinatorial optimization; Mathematics of computing → Solvers; Mathematics of computing → Graph algorithms

Keywords and phrases Directed feedback vertex set, local search, large neighborhood search, mixed integer programming

Supplementary Material The source code for the described solver is available on GitHub and Zenodo:
Software (Source Code): `https://github.com/NunuNoName/dfvsp-solver`
Software (Source Code): `https://doi.org/10.5281/zenodo.6643236` [1]

1 Introduction

The HyMeHeu-solver is a **Hy**brid **Me**ta **Heu**ristic solver for the *directed feedback vertex set problem* (DFVSP). The DFVSP is defined on a directed graph and its aim is to find a minimum cardinality subset of vertices whose removal from the graph results in a *directed acyclic graph* (DAG). The provided solver employs a hybrid metaheuristic based on a large neighborhood search (LNS), which repeatedly applies a pair of a destroy and a repair operator. First, the repair operator changes a part of a given solution and then, the destroy operator fixes this partial solution to recover a valid solution. The hybridization is achieved by using a mixed integer programming (MIP) solver in the repair operator. A construction heuristic based on a greedy function is used to generate an initial solution. This initial solution is improved by applying a simple local search procedure and the resulting solution then serves as the starting point for the LNS procedure.

2 Graph Preprocessing

The first step of our solver is the preprocessing of the input graph by the application of five reduction rules, which reduce the size of the graph and make it easier to handle. Four of the rules come from Levy and Low [5], the fifth reduction rule is inspired by Park and Akers [7]. The reduction rules are as follows:

1. If a vertex v has an indegree or outdegree of zero, remove v and all incident edges.
   - Levy and Low [5] formulated this operation as two separate contraction rules, one for the indegree and one for the outdegree.

2. Find a vertex v with an indegree of one and let u be the unique predecessor of v. Merge v into u by adding edges from u to all successors of v and then removing v and its incident edges from the graph.
3. Find a vertex v with an outdegree of one and let u be the unique successor of v. Merge v into u by adding edges from u to all predecessors of v and then removing v and its incident edges from the graph.
4. If a vertex v has a self-loop, add v to the DFVS and remove v and all incident edges.
5. Partition the graph into its strongly connected components (SCCs). If an SCC contains a single vertex v, add v to the DFVS and remove v and its incident edges from the graph. If an SCC consists of two vertices, randomly pick one of them and add it to the DFVS. Then remove both vertices and their incident edges from the graph. The other SCCs remain unchanged.

These or similar rules are also used by Lin and Jou [6] and by Fleischer et al. [2] when tackling the DFVSP. In our solver, they are repeatedly applied until none of them leads to a reduction of the graph anymore. The preliminary DFVS is stored and the remaining graph is partitioned into its SCCs, which are sorted by increasing size. If their number exceeds 1000, we merge smaller ones into subgraphs of up to 100 vertices to reduce memory consumption. The resulting subproblems are processed separately and sequentially, each starting with another application of the reduction rules. Components with exactly three vertices are a special case that is dealt with by randomly selecting two vertices for the DFVS. Subgraphs or SCCs with up to 100 vertices are also handled differently as they are directly encoded with the mixed integer programming model that is introduced in Section 4. The remaining bigger SCCs are passed to the large neighborhood search algorithm which is described in Section 3.

3 Large Neighborhood Search

Large neighborhood search (LNS) is a prominent category of local search (LS) metaheuristics that was proposed by Shaw [8]. The key idea of LNS is to consider much larger neighborhoods than in normal LS but to investigate these neighborhoods by more effective techniques than naive enumeration. This is often achieved by using a pair of a *destroy* and a *repair* method. A destroy method changes a part of the current solution and the repair method can then fix and also modify the resulting solution in order to achieve again a valid solution. The concrete operators we use for the DFVS and the neighborhood they create are described in Section 3.3. The LNS procedure also needs an initial solution which is generated by the construction heuristic that is introduced in Section 3.1. As a better starting point is beneficial for the LNS, we improve the initial solution by employing a simple local search procedure as described in Section 3.2.

This LNS algorithm is used to solve to each subproblem with more than 100 vertices by repeatedly applying the destroy and repair operators. It terminates when either a fixed time limit is exceeded or 50 iterations without improvement of the solution have passed.

3.1 Construction Heuristic

Our construction heuristic is based on the following greedy function as presented by Tang et al. [9]: $h(v) = \deg^-(v) + \deg^+(v) - \lambda \times |\deg^-(v) - \deg^+(v)|$, where v denotes a vertex in the graph, $\deg^-(v)$ is its indegree and $\deg^+(v)$ its outdegree. The parameter λ controls the influence of each term and is set to $\lambda = 0.3$ [9]. The vertices are sorted in increasing order of their h-value and then iterated over while keeping track of visited vertices. Each

vertex which has an already visited outneighbor, that is not part of the DFVS, is added to the DFVS itself.

3.2 Local Search

As the solution generated by the construction heuristic can be far from optimal, we apply a classical local search to it. The intention behind this is to find or at least come closer to a local optimum because a smaller initial solution is beneficial for the following LNS. Our local search iteratively removes a single vertex from the DFVS and accepts the resulting solution if the remaining graph is still acyclic. As this procedure might be time-consuming for some instances, we limit the time to 60 seconds and try to search promising parts first by sorting the vertices according to their h-value (cf. Section 3.1).

3.3 LNS Neighborhood Structure

The neighborhoods of the LNS procedure are induced by the following employed destroy and repair operators.

Destroy Operator. The destroy operator moves vertices from the current solution to the induced DAG which can render the solution invalid and destroy the DAG by introducing cycles. The number of selected vertices depends on the size of the DFVS but is at most 1000. The vertices are chosen by tournament selection with the tournament size $k = 3$. In each tournament, the vertex with the smallest h-value (cf. Section 3.1) is selected.

Repair Operator. The repair operator receives the partial solution and the enlarged DAG from the destroy operator and has the goal of restoring a valid solution. This is done by solving the DFVSP on the smaller subproblem that is now posed by the enlarged DAG. First, this subgraph is simplified by applying the reduction rules as introduced in Section 2 while storing the resulting preliminary DFVS. A mixed integer programming formulation of the DFVSP is used to encode the remaining subproblem, which is solved by a MIP solver. The new valid solution is then composed of the solution from the MIP solver, the vertices selected by the reduction rules and the remaining part of the destroyed original solution.

4 Mixed Integer Programming

We solve a restricted mixed integer programming model of the DFVSP in the repair operator of the LNS procedure to achieve a hybridization of the metaheuristic. It uses a formulation based on cycle elimination constraints (CEC): For each simple cycle C, a constraint of the form $\sum_{v \in C} y_v \leq |C| - 1$ is added, where y_v denotes the binary decision variable for a vertex which is set to zero if the vertex is part of the DFVS and to one otherwise. These constraints ensure that at least one vertex of each simple cycle is selected for the DFVS. As the number of these constraints can grow exponentially with the size of the graph, we add them dynamically. To further speed up the solving process, we already add such constraints for all cycles of length two in the beginning and strengthen them by identifying cliques in the graph that has an edge for each cycle of length two. This MIP model is then solved with the solver SCIP[1] [3, 4].

[1] `https://www.scipopt.org/`

References

1 Maria Bresich, Günther Raidl, and Johannes Varga. HyMeHeu-solver - A Hybrid Metaheuristic Solver for the Directed Feedback Vertex Set Problem, June 2022. doi:10.5281/zenodo.6643236.

2 Rudolf Fleischer, Xi Wu, and Liwei Yuan. *Experimental Study of FPT Algorithms for the Directed Feedback Vertex Set Problem*, volume 5757, page 611–622. Springer Berlin Heidelberg, 2009. doi:10.1007/978-3-642-04128-0_55.

3 Gerald Gamrath, Daniel Anderson, Ksenia Bestuzheva, Wei-Kun Chen, Leon Eifler, Maxime Gasse, Patrick Gemander, Ambros Gleixner, Leona Gottwald, Katrin Halbig, Gregor Hendel, Christopher Hojny, Thorsten Koch, Pierre Le Bodic, Stephen J. Maher, Frederic Matter, Matthias Miltenberger, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Christine Tawfik, Stefan Vigerske, Fabian Wegscheider, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 7.0. Technical report, Optimization Online, March 2020. URL: http://www.optimization-online.org/DB_HTML/2020/03/7705.html.

4 Gerald Gamrath, Daniel Anderson, Ksenia Bestuzheva, Wei-Kun Chen, Leon Eifler, Maxime Gasse, Patrick Gemander, Ambros Gleixner, Leona Gottwald, Katrin Halbig, Gregor Hendel, Christopher Hojny, Thorsten Koch, Pierre Le Bodic, Stephen J. Maher, Frederic Matter, Matthias Miltenberger, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Christine Tawfik, Stefan Vigerske, Fabian Wegscheider, Dieter Weninger, and Jakob Witzig. The SCIP Optimization Suite 7.0. ZIB-Report 20-10, Zuse Institute Berlin, March 2020. URL: http://nbn-resolving.de/urn:nbn:de:0297-zib-78023.

5 Hanoch Levy and David W. Low. A contraction algorithm for finding small cycle cutsets. *Journal of Algorithms*, 9(4):470–493, 1988. doi:https://doi.org/10.1016/0196-6774(88)90013-2.

6 Hen-Ming Lin and Jing-Yang Jou. On computing the minimum feedback vertex set of a directed graph by contraction operations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(3):295–307, 2000. doi:10.1109/43.833199.

7 Sungju Park and Sheldon B. Akers. An efficient method for finding a minimal feedback arc set in directed graphs. In *[Proceedings] 1992 IEEE International Symposium on Circuits and Systems*, volume 4, pages 1863–1866 vol.4, May 1992. doi:10.1109/ISCAS.1992.230449.

8 Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In Michael Maher and Jean-Francois Puget, editors, *Principles and Practice of Constraint Programming — CP98*, pages 417–431, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. doi:https://doi.org/10.1007/3-540-49481-2_30.

9 Zhipeng Tang, Qilong Feng, and Ping Zhong. Nonuniform neighborhood sampling based simulated annealing for the directed feedback vertex set problem. *IEEE Access*, 5:12353–12363, 2017. doi:10.1109/ACCESS.2017.2724065.