Crossy: An Exact Solver for One-Sided Crossing Minimization

Tobias Röhr ✉
Hasso Plattner Institute, University of Potsdam, Germany

Kirill Simonov ✉
Hasso Plattner Institute, University of Potsdam, Germany

Abstract

We describe Crossy, an exact solver for the One-Sided Crossing Minimization (OSCM) problem, submitted to the Parameterized Algorithms and Computational Experiments (PACE) Challenge 2024. Crossy applies a series of reductions and subsequently transforms the instance to a Weighted Directed Feedback Arc Set (WDFAS) instance formulated as incremental MaxSAT. We use the recently introduced concept of User Propagators for CDCL SAT solvers to implicitly add cycle constraints.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases One-Sided Crossing Minimization, Exact Algorithms, Graph Drawing, Incremental MaxSAT

Supplementary Material *Software (Source Code)*: `https://doi.org/10.5281/zenodo.12082773`
Software (Source Code): `https://github.com/roehrt/crossy`

1 Preliminaries

Given a bipartite graph $G = (A, B, E)$ and a linear ordering of the vertices in A, the One-Sided Crossing Minimization problem asks for a linear ordering $\prec$ of the vertices in B that minimizes the number of crossings of a straight-line drawing when placing the vertices in A and B on two parallel lines. To enable some of our reduction rules, it is convenient to relax the problem and allow the input to be a multigraph.

For $u, v \in B$, define $c(u, v)$ to be the number of crossings between the edges incident to u and v when $u \prec v$. Moreover, we call $u \prec v$ the *natural order* of u and v if and only if $c(u, v) < c(v, u)$. Since either $u \prec v$ or $v \prec u$, we get a simple lower bound on the number of crossings: $\sum_{u \in B} \sum_{v \in B} \min(c(u, v), c(v, u))$.

The penalty graph of an OSCM-instance is a directed graph on the vertices in B. In order to penalize pairs of vertices that do not appear in the natural order, we add an edge $u \to v$ carrying weight $c(u, v) - c(v, u)$ for any pair $u, v \in B$ with $c(u, v) > c(v, u)$. Note that the weight of a Minimum Weight Feedback Arc Set in the penalty graph equals the minimum number of crossings in the corresponding OSCM-instance above the lower bound [6].

We say that we *commit* $u \prec v$ if we only look for solutions where u appears before v. To model this knowledge in the penalty graph, we insert an edge $u \to v$ with infinite weight in this case.

2 Reduction rules

After merging twins in B and removing isolated vertices, we proceed to apply two sets of reduction rules to the instance. The first set of rules is OSCM-specific and the second set contains general-purpose rules for the Weighted Directed Feedback Arc Set problem.

2.1 Rules for OSCM

We apply two well-known rules for OSCM that we call Planar Ordering and Transitivity, and introduce a new rule, Dominance.

Planar Ordering If there is a pair of vertices $u, v \in B$ such that $c(u, v) = 0$, commit $u \prec v$ [2].
Transitivity If $u \prec v$ and $v \prec w$, commit $u \prec w$.

For Dominance, consider the following argument showing that the natural order $u \prec v$ is optimal, in a certain case. Assume by contradiction that $u \prec v$ is not optimal, and consider an optimal ordering where v appears before u instead. Let S be the set of vertices between v and u, i.e., $S = \{w \in B \mid v \prec w \prec u\}$.

In order for $u \prec v$ not to be optimal, the number of crossings inflicted by $v \prec S \prec u$ must be strictly less than the number of crossings inflicted by $u \prec S \prec v$ and $u \prec v \prec S$ as well as $S \prec u \prec v$. If we can derive a contradiction for each possible set S, we have proven that $u \prec v$ is optimal and are able to commit $u \prec v$.

Inspired by the tabular analysis technique of [1], let us categorize the neighbors of S into disjoint sets based on their relative position to the neighbors of u and v, so that the neighbors in the same set are effectively indistinguishable with respect to the condition above.

By introducing variables describing these sets, we can express the number of crossings for each configuration as a linear combination of the variables. Let $L(u, v, w)$ denote this linear combination for some configuration $u \prec v \prec w$. Now we can derive a system of linear inequalities that must hold for $u \prec v$ not to be optimal:

$$
\begin{aligned}
&L(v, S, u) < L(S, u, v) \\
&L(v, S, u) < L(u, S, v) \\
&L(v, S, u) < L(u, v, S).
\end{aligned}
$$

Each variable can also be bounded by counting the number of edges outgoing from each category of neighbors of S. To check the feasibility of this system of inequalities, we can use an LP solver.

Dominance If there is a pair of vertices $u, v \in B$ such that the LP derived from the above analysis is infeasible, commit $u \prec v$.

Note that the Planar Ordering rule is thus a special case of the Dominance rule. While still running in polynomial time, the Dominance rule is computationally quite expensive as we need to solve a linear program for each pair of vertices. To mitigate this, we apply a weaker version of the rule in our implementation. We drop the upper bound constraints on the variables and only check the feasibility of the pairwise inequalities.

This can be done in linear time, resulting in the overall running time of $\mathcal{O}(|B||E|)$ for all OSCM-reductions.

2.2 Rules for WDFAS

Crossy proceeds to apply very general rules for the Weighted Directed Feedback Arc Set problem on the penalty graph.

Strongly Connected Components Find an optimal ordering for each strongly connected component, then combine the solutions following a topological ordering of the condensation graph.

Minimum Cut For each edge $u \to v$, commit $u \prec v$, if the weighted minimum cut separating v from u does not exceed the weight of $u \to v$.

Even though the Minimum Cut Rule is able to commit some pairs, it turns out to be not worth the extra computational cost in our experiments. We therefore disable it in our implementation, resulting in the overall running time of $\mathcal{O}(|B||E|)$ for all preprocessing steps.

3 Incremental MaxSAT formulation

Following the work of the winning team of the PACE Challenge 2022 [4], we formulate the Weighted Directed Feedback Arc Set problem as incremental MaxSAT, aiming to hit all cycles in the penalty graph.

3.1 Encoding

For each arc $u \to v$ in the penalty graph, we introduce a variable $x_{u \to v}$ representing the arc's inclusion in the solution. We set the weight of each variable to the negated weight of the corresponding arc. If the weight of an arc is infinite, then the corresponding variable will be fixed as false. To encode a cycle constraint, we add a disjunction of the variables corresponding to the arcs in the cycle.

Crossy starts by adding short cycles of length at most 4 to the MaxSAT instance explicitly supplying the MaxSAT solver with some initial cores. All longer cycles will later be added implicitly by the user propagator.

3.2 User Propagator

We modify UWrMaxSAT [5] to allow us to connect a user propagator to its underlying SAT solver, CaDiCal. The recently introduced IPASIR-UP interface [3] enables us to add user-defined propagators to CaDiCal without modifying the solver itself.

Our user propagator employs the concept of Cycle Propagation as introduced by [4]. It follows the steps of the SAT solver and adds a cycle constraint whenever a new arc is assigned that closes a cycle.

Specifically, we approach the problem of incremental cycle detection with rollbacks by maintaining the depth of each vertex eagerly. On each arc insertion, we recursively update the depth of affected vertices and check if we have found a cycle.

As we have a considerable amount of edges with infinite weight that always remain in the graph, we initially compute the transitive reduction of this subgraph to reduce the amount of work our cycle detection has to do.

4 Discussion

Despite not using any user-defined heuristics, Crossy successfully solves a wide range of OSCM instances. However, OSCM allows for the application of various effective heuristics, which can enhance ILP-based approaches but are not applicable to MaxSAT-based solvers like Crossy. This inability to integrate user heuristics appears to be Crossy's most significant limitation.

The performance of UWrMaxSAT in MaxSAT competitions relies on running SCIP beforehand. This reliance presents a notable disadvantage for Crossy, as SCIP cannot utilize the user propagator.

References

1. Vida Dujmovic, Henning Fernau, and Michael Kaufmann. Fixed parameter algorithms for one-sided crossing minimization revisited. *J. Discrete Algorithms*, 6(2):313–323, 2008. URL: https://doi.org/10.1016/j.jda.2006.12.008, doi:10.1016/J.JDA.2006.12.008.
2. Vida Dujmovic and Sue Whitesides. An efficient fixed parameter tractable algorithm for 1-sided crossing minimization. *Algorithmica*, 40(1):15–31, 2004. URL: https://doi.org/10.1007/s00453-004-1093-2, doi:10.1007/S00453-004-1093-2.
3. Katalin Fazekas, Aina Niemetz, Mathias Preiner, Markus Kirchweger, Stefan Szeider, and Armin Biere. IPASIR-UP: user propagators for CDCL. In Meena Mahajan and Friedrich Slivovsky, editors, *26th International Conference on Theory and Applications of Satisfiability Testing, SAT 2023, July 4-8, 2023, Alghero, Italy*, volume 271 of *LIPIcs*, pages 8:1–8:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. URL: https://doi.org/10.4230/LIPIcs.SAT.2023.8, doi:10.4230/LIPICS.SAT.2023.8.
4. Rafael Kiesel and André Schidler. PACE solver description: Dager - cutting out cycles with maxsat. In Holger Dell and Jesper Nederlof, editors, *17th International Symposium on Parameterized and Exact Computation, IPEC 2022, September 7-9, 2022, Potsdam, Germany*, volume 249 of *LIPIcs*, pages 32:1–32:4. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. URL: https://doi.org/10.4230/LIPIcs.IPEC.2022.32, doi:10.4230/LIPICS.IPEC.2022.32.
5. Marek Piotrów. Uwrmaxsat: Efficient solver for maxsat and pseudo-boolean problems. In *32nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2020, Baltimore, MD, USA, November 9-11, 2020*, pages 132–136. IEEE, 2020. doi:10.1109/ICTAI50040.2020.00031.
6. Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.