PACE Solver Description: LetsJustCHILS

Adil Chhabra
Heidelberg University, Germany

Marlon Dittes
Heidelberg University, Germany

Ernestine Großmann
Heidelberg University, Germany

Kenneth Langedal
Heidelberg University, Germany

Henrik Reinstädtler
Heidelberg University, Germany

Christian Schulz
Heidelberg University, Germany

Darren Strash
Hamilton College, Clinton, NY, US

Henning Woydt
Heidelberg University, Germany

Abstract

This is a short description of our exact solver and heuristic, which was submitted to the PACE 2025 challenge on Dominating Set and Hitting Set. Our solvers reduce the Dominating Set to the Hitting Set problem, making both problems equivalent to our solvers. Then, we use known data reduction rules for the Hitting Set problem to simplify the instances for both the exact solver and the heuristic. For the exact track, we pass the reduced instance to a MaxSAT solver that tries to compute an optimal solution. For the heuristic, we further reduce the problem to the Vertex Cover problem, where we apply known reductions for that problem. On this reduced instance, we run the CHILS heuristic to quickly compute a high-quality solution. Since the reduction to Vertex Cover can significantly increase the graph size, we also utilize two simple heuristics for the Hitting Set problem, which we use as a backup for particularly problematic instances.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Hitting Set, Dominating Set, Maximum-Weight Independent Set, Data Reduction, Local Search, MaxSAT

Supplementary Material *Software*: `https://github.com/KennethLangedal/PACE2025` [8]
archived at `swh:1:dir:c834fc32fda6f61d5e845c626395b5ab077a3f4f`

1 Introduction

This document presents LetsJustCHILS, an exact algorithm and heuristic submitted to all tracks of the 2025 iteration of the Parameterized Algorithms and Computational Experiments (PACE) challenge on the Dominating Set and Hitting Set problems. Our solvers employ a combination of effective data reduction techniques, exact solving via a Partial MaxSAT formulation, and heuristics based on a reduction to the well-studied Maximum Weight Independent Set problem.

In the following, we first define the problems and notation used in Section 2. Then, we present the data reduction rules used across all tracks in Section 3. Finally, we provide detailed descriptions of both our exact solver and heuristic in sections 4 and 5.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set $E \subseteq V \times V$. A *dominating set* is a subset $D \subseteq V$ such that every vertex $u \in V$ is either in D or adjacent to a vertex in D. The DOMINATING SET problem is that of finding a dominating set with the smallest cardinality. A *hypergraph* is defined as $G = (V, \mathcal{S})$ where V is a set of vertices and $\mathcal{S}$ is a collection of sets, where each $S \in \mathcal{S}$ is a subset of the vertices in the graph, later referred to as a *hyperedge*. For a vertex u in a hypergraph, let $\mathcal{S}(u) = \{S \in \mathcal{S} \mid u \in S\}$ be the hyperedges containing u as an endpoint. A *hitting set* for a hypergraph is a subset $H \subseteq V$ such that every set $S \in \mathcal{S}$ contains at least one element from H. The HITTING SET problem asks for a hitting set with the smallest cardinality.

3 Reductions

Our algorithm makes use of reductions between problems, as well as data reductions. We detail each reduction here.

From Dominating Set to Hitting Set. It is well known that the DOMINATING SET problem can be reduced to the HITTING SET problem by considering every induced neighborhood as a hyperedge. Specifically, for a given graph G, we construct a hypergraph H with the same vertex set V and for each vertex $v \in V$, we add a hyperedge corresponding to the *closed neighborhood* of v, defined as $N[v] = \{v\} \cup \{u \in V \mid \{u, v\} \in E\}$. A dominating set in G then corresponds to a hitting set in H that intersects every such hyperedge. As a first step in our exact and heuristic DOMINATING SET solvers, we perform this reduction to HITTING SET.

Hitting Set Data Reductions. We exhaustively apply the following data reductions as described by Bläsius et al. [1] to the hitting set instance. The domination rules were first introduced by Weihe [11].

▶ **Reduction 1** (Degree one hyperedge). *Let $S \in \mathcal{S}$ such that $|S| = 1$ and $u \in S$ be its single element. Then, u can be included in the hitting set, and all hyperedges containing u can be removed.*

▶ **Reduction 2** (Domination vertex). *Let $u, v \in V$ such that $\mathcal{S}(u) \subseteq \mathcal{S}(v)$. Then, we can remove u from the hypergraph since any optimal solution either includes v, or can substitute u with v without loss of optimality.*

▶ **Reduction 3** (Domination hyperedge). *Let $S_1, S_2 \in \mathcal{S}$ such that $S_1 \subseteq S_2$. Then, we can safely delete S_2 from $\mathcal{S}$ since any hitting set that hits S_1 will always hit S_2 as well.*

From Hitting Set to Vertex Cover. After the HITTING SET reductions, our heuristics continue by reducing the HITTING SET problem to the VERTEX COVER problem. This reduction works by encoding each hyperedge as a clique [6]. The general idea is to add one vertex v_u for every vertex $u \in V$, and for each hyperedge $S \in \mathcal{S}$, add $|S|$ new vertices corresponding to the elements in S. That is, for each $u \in S$, add a new vertex v_{uS}. We then connect all the vertices originating from the same set $S \in \mathcal{S}$ to form a clique. Finally, for every vertex originating from a set, we add the edge $\{v_{uS}, v_u\}$. The intuition for this reduction is that each added clique ensures that at least one vertex from the corresponding hyperedge must be included in the VERTEX COVER instance. That is because each clique needs at least $|S| - 1$ vertices in the vertex cover. For the one v_{uS} that we do not include, we must include

v_u instead. An optimal solution could include all the vertices in a clique. However, in that case, we can greedily swap any vertex v_{uS} in that clique for the corresponding v_u and still have an optimal solution. Lifting the solution back to the Hitting Set problem is done by simply reading off the configuration of the vertices corresponding to the hitting set vertices.

The motivation for using the Vertex Cover problem instead of Hitting Set is the ample amount of experimental work already done for this problem. Considering the Minimum Vertex Cover problem and its complementary problems, Maximum Independent Set and Maximum Clique, as well as their weighted generalizations, more than thirty experimental publications have been made in the last twenty years [4]. There is also an increasing focus on data reduction in this area. The Minimum Vertex Cover problem was also the focus for the 2019 iteration of the PACE challenge [2].

Vertex Cover Data Reductions. Once we have an equivalent Vertex Cover instance, we use a wide range of known reductions for that problem. We actually use a library designed for reducing the Maximum Weight Independent Set problem. However, we can set all the weights to one and read off the complement of the resulting independent set to get our vertex cover. For an overview of these reductions, see the survey by Großmann et al. [4] that also comes with a reference implementation[1]. One especially important reduction for our heuristic is the weighted struction by Gellner et al. [3].

4 Exact

For the exact track, our approach first reduces the size of the Hitting Set instance using reduction rules and solving connected components individually. Then, it solves it exactly using a Partial MaxSAT formulation.

A boolean satisfiability formula consists of boolean variables $\{x_1, x_2, ..., x_n\}$, and a set of *clauses* where a clause could look like $c = \neg x_1 \vee x_2 \vee \neg x_3 \vee ... \vee x_k$. Notice that each variable can appear negated. The MaxSAT problem is to find an assignment that maximizes the number of satisfied clauses. Partial MaxSAT is a generalization of MaxSAT that divides clauses into *hard* clauses that must be satisfied in every feasible assignment and *soft* clauses that can be unsatisfied. A solution to the Partial MaxSAT problem is an assignment that satisfies all hard clauses and maximizes the number of soft clauses satisfied.

Our exact solver first checks the input hypergraph for connectivity. If it contains multiple connected components, each component is solved independently. Solving a component involves applying a series of standard Hitting Set reduction rules, as described in Section 3. If the instance becomes disconnected after reduction, we split it into connected components again and solve each one separately. Finally, each reduced and connected component is transformed into a Partial MaxSAT instance. For each vertex v in the hypergraph, we have a soft clause $\neg x_v$, and for each hyperedge $S \in \mathcal{S}$ we add one hard clause $\vee_{v \in S} x_v$ that ensures S is satisfied. This Partial MaxSAT instance is then solved using `UWrMaxSat` [10] to compute an optimal solution.

Correctness follows from the fact that each reduction rule preserves optimality and that connected components can be solved independently since there are no hyperedges between them. The Partial MaxSAT encoding guarantees optimality by enforcing all hitting constraints as hard clauses while minimizing the number of selected vertices through soft clauses.

[1] `https://github.com/KarlsruheMIS/DataReductions`

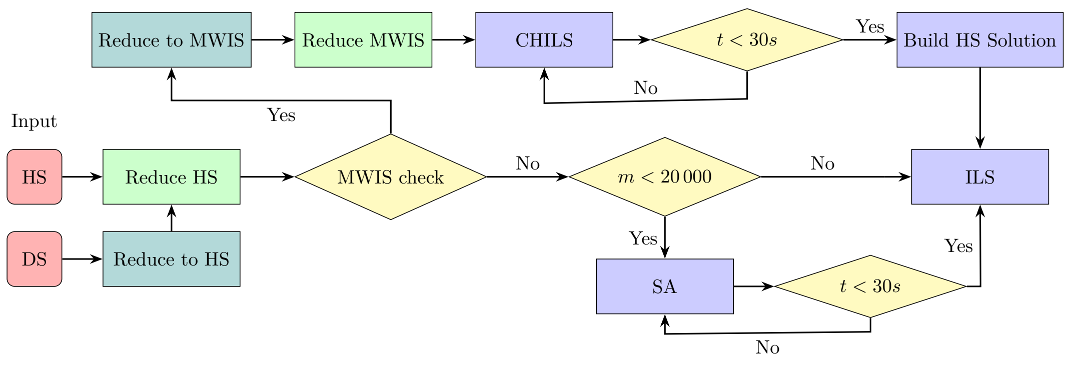

Figure 1 Illustration of our heuristic approach for the DOMINATING SET (DS) and HITTING SET (HS) problem. Dark green are reductions to another problem, while light green is applying data reductions to reduce the problem instance. Our checks involve the remaining time t and number of edges m. The MAXIMUM WEIGHTED INDEPENDENT SET (MWIS) check consists of checking the number of vertices ($n > 5\,000$), whether the largest hyperedge size is small ($|S_{max}| \leq 32$), and if we have enough memory for the reduction to MWIS.

5 Heuristic

For the heuristic track, we again apply the same HITTING SET reductions as the exact solver. Then, we reduce the HITTING SET instance to an equivalent instance of VERTEX COVER, where we again apply known data reductions. The whole reduction procedure is described in Section 3.

After preprocessing using data reductions, we use the Concurrent Hybrid Iterated Local Search (CHILS) heuristic for the MAXIMUM WEIGHTED INDEPENDENT SET problem by Großmann et al. [5]. An illustration of the heuristic is shown in Figure 1.

In practice, the reduction from HITTING SET to VERTEX COVER is effective only when the hyperedges are relatively small in size. If not, the resulting VERTEX COVER instance becomes too large to process effectively. Therefore, we also developed two simple HITTING SET heuristics to deal with problematic instances.

Simulated Annealing. Simulated annealing is a well-known probabilistic metaheuristic that is inspired by crystallization in physics [7]. Our implementation works by iteratively making changes to a feasible hitting set H. At every step, we pick a random vertex $u \in V$ from the hypergraph. Then, we compute a cost c based on the change in the solution size resulting from removing or including u in the hitting set. If $u \notin H$, then $c = 1$ since the solution size would increase by one. If $u \in H$, we approximate the cost by counting the number of hyperedges that the vertex covers alone, that is, $c = |\{S \in \mathcal{S} \mid S \cap H = \{u\}\}| - 1$. Finally, we change the configuration of the vertex with probability $e^{-c/T}$, where T is the current temperature. In the case where a vertex is removed, we greedily make the solution feasible again by random additions. The temperature starts at 0.25 and is gradually decreased to 0.08 over the course of 200 million iterations. We repeat multiple cooling cycles as long as time allows, keeping track of the best solution discovered.

Iterated Local Search. Iterated local search is another popular metaheuristic that has been successfully used in previous iterations of the PACE challenge [9]. Unlike simulated annealing, iterated local search only accepts changes that do not increase the size of the current hitting set. The general idea is to randomize the solution in a small area, greedily search for local improvements where the solution changes, and then check if the new solution is equal to or better than the old one. If the solution worsens, the changes are undone and the process is repeated.

6 Conclusion

In summary, LetsJustCHILS reduces Dominating Set instances to Hitting Set instances, allowing for the same efficient reduction rules to be applied across all tracks. Our exact solver utilizes a Partial MaxSAT formulation to compute an optimal solution. The heuristic further reduces the problem to Vertex Cover and Maximum Weight Independent Set, leveraging existing work on these problems. Additionally, two lightweight Hitting Set heuristics ensure that we can still handle instances that are not easily reduced.

References

1 Thomas Bläsius, Tobias Friedrich, David Stangl, and Christopher Weyand. An efficient branch-and-bound solver for hitting set. In *Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, pages 209–220. SIAM, 2022. `doi:10.1137/1.9781611977042.17`.

2 M. Ayaz Dzulfikar, Johannes K. Fichte, and Markus Hecher. The PACE 2019 parameterized algorithms and computational experiments challenge: the fourth iteration. In *Proceedings of the 14th International Symposium on Parameterized and Exact Computation (IPEC 2019)*, pages 25–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019. `doi:10.4230/LIPICS.IPEC.2019.25`.

3 Alexander Gellner, Sebastian Lamm, Christian Schulz, Darren Strash, and Bogdán Zaválnij. Boosting data reduction for the maximum weight independent set problem using increasing transformations. In *2021 Proceedings of the Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 128–142. SIAM, 2021. `doi:10.1137/1.9781611976472.10`.

4 Ernestine Großmann, Kenneth Langedal, and Christian Schulz. A comprehensive survey of data reduction rules for the maximum weighted independent set problem. 2024. URL: `https://arxiv.org/abs/2412.09303`, `arXiv:2412.09303`.

5 Ernestine Großmann, Kenneth Langedal, and Christian Schulz. Accelerating reductions using graph neural networks and a new concurrent local search for the maximum weight independent set problem, 2025. URL: `https://arxiv.org/abs/2412.14198`, `arXiv:2412.14198`.

6 Ashwin Jacob, Fahad Panolan, Venkatesh Raman, and Vibha Sahlot. Structural parameterizations with modulator oblivion. *Algorithmica*, 84(8):2335–2357, 2022. `doi:10.1007/S00453-022-00971-7`.

7 Scott Kirkpatrick, C. Daniel Gelatt Jr, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. `doi:10.1126/science.220.4598.671`.

8 Kenneth Langedal, Marlon Dittes, Henning Woydt, and Ernestine Großmann. Kenneth-Langedal/PACE2025: PACE-2025, June 2025. `doi:10.5281/zenodo.15767385`.

9 Helena R. Lourenço, Olivier C. Martin, and Thomas Stützle. Iterated local search. In *Handbook of metaheuristics*, pages 320–353. Springer, 2003. `doi:10.1007/0-306-48056-5_11`.

10 Marek Piotrów. UWrMaxSat: Efficient solver for MaxSAT and pseudo-boolean problems. In *Proceedings of the 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 132–136, 2020. `doi:10.1109/ICTAI50040.2020.00031`.

11 Karsten Weihe. Covering trains by stations or the power of data reduction. *Proceedings of Algorithms and Experiments, ALEX*, pages 1–8, 1998.