



# PACE Solver Description: Weighting-based Local Search Heuristic for the Hitting Set Problem

Canhui Luo 


Huazhong University of Science and Technology, Wuhan, China

Qingyun Zhang 

Huazhong University of Science and Technology, Wuhan, China

Zhouxing Su<sup>1</sup> 

Huazhong University of Science and Technology, Wuhan, China

Zhipeng Lü 

Huazhong University of Science and Technology, Wuhan, China

## Abstract

We present a unified heuristic solver for the PACE 2025 challenge, addressing both the dominating set and hitting set problems by reducing them to the unicost set covering problem. Our solver applies standard reduction rules, a multi-round frequency-based greedy initializer, and a local search guided by adaptive element weights. Additional techniques, such as component-level exact solving and swap restriction, further enhance performance. In the final official evaluation, our proposed solver achieved second place in the heuristic track for the dominating set problem of the PACE 2025 challenge, while securing first place in the heuristic track for the hitting set problem.

**2012 ACM Subject Classification** Mathematics of computing → Combinatorial optimization; Computing methodologies → Search methodologies

**Keywords and phrases** PACE 2025, Dominating Set, Hitting Set, Heuristic Optimization, Weighted Local Search

**Digital Object Identifier** 10.4230/LIPIcs.IPEC.2025.40

**Category** PACE Solver Description

**Supplementary Material** *Software (Source Code)*: <https://github.com/lxily/PACE2025.DS-HS>

**Funding** This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62402191, the Interdisciplinary Research Program of Hust under Grant 5003300129, and Innovation Program for Quantum Science and Technology under Grant No. 2024ZD0300500.

**Acknowledgements** We want to thank the organizers of PACE 2025 challenge and all other participants for creating such an engaging challenge.

## 1 Challenge Problem and Transformation

The PACE 2025 challenge involves two fundamental NP-hard problems: the dominating set problem and the hitting set problem. The dominating set problem seeks to select as few vertices as possible in a given graph such that every other vertex is adjacent to at least one selected vertex. The hitting set problem requires selecting as few elements as possible from a set system such that each set contains at least one selected element. In fact, the former can be regarded as a special case of the latter. We unify these two problems by transforming them into a unicost set covering problem, as follows:

<sup>1</sup> Corresponding author: Zhouxing Su



© Canhui Luo, Qingyun Zhang, Zhouxing Su, and Zhipeng Lü;  
licensed under Creative Commons License CC-BY 4.0

20th International Symposium on Parameterized and Exact Computation (IPEC 2025).

Editors: Akanksha Agrawal and Erik Jan van Leeuwen; Article No. 40; pp. 40:1–40:4

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- 40 ■ In the dominating set problem, each vertex  $v_i$  is represented as both a set  $s_i$  and an
- 41 element  $e_i$ . Each set  $s_i$  covers all elements corresponding to its neighboring vertices.
- 42 ■ In the hitting set problem, each original set  $s_i^o$  is mapped to an element  $e_i$ , and each
- 43 original element  $e_j^o \in s_i^o$  is mapped to a set  $s_j$  covering  $e_i$ .

44 Subsequently, the optimization objectives of the two original problems are unified as  
 45 minimizing the number of selected sets that cover all elements, although such a transformation  
 46 inevitably loses certain characteristic information of the original problems.

## 47      **2      Solver Methodology**

48 In this section, we present our heuristic solver for the set covering problem. It integrates  
 49 instance reduction, initialization, element-weighted local search, and two simple but effective  
 50 optimization strategies that further enhance the search process.

### 51      **2.1      Preprocessing via Reduction Rules**

52 Considering that the datasets in the PACE 2025 Challenge may contain millions of sets and  
 53 elements, applying possible reductions to them is highly beneficial for the subsequent search.  
 54 We employ three classical reductions without sacrificing optimality [2]:

- 55 ■ **Element Dominance:** If all sets that cover element  $e_i$  also cover element  $e_j$  (where  
 56  $i \neq j$ ), then  $e_j$  can be safely removed.
- 57 ■ **Set Dominance:** If all elements covered by set  $s_i$  are also covered by set  $s_j$  (where  
 58  $i \neq j$ ), then  $s_i$  can be safely removed.
- 59 ■ **Mandatory Coverage:** If an element  $e_i$  is uniquely covered by one set  $s_i$ , then  $s_i$  must  
 60 be selected, and all elements covered by  $s_i$  can be removed.

61 These reduction rules preserve the existence and structure of optimal solutions. By  
 62 iteratively applying them, the problem size can typically be reduced significantly. We solve  
 63 the reduced instance and merge the optimized result with the sets fixed during the reduction  
 64 process to form the final solution.

### 65      **2.2      Frequency-Guided Initialization**

66 We define the *importance score*  $IS(e_i) = \frac{1}{\text{freq}(e_i)}$  of an element as the reciprocal of its  
 67 frequency (i.e., the number of sets that can cover it). Then, the score of a set  $IS(s_i) =$   
 68  $\sum_{e_j \in \text{uncovered}(s_i)} IS(e_j)$  is the sum of the scores of all currently uncovered elements it covers.  
 69 The greedy algorithm iteratively selects the set with the highest score, thereby prioritizing  
 70 the coverage of hard-to-cover elements. After each selection, scores of related sets are updated  
 71 to reflect the new uncovered element set.

72 Since the number of sets that can cover an element differs from the number of sets that  
 73 actually cover it in a solution, the original importance score may be biased. To correct this,  
 74 after obtaining a feasible solution, we refine the score of each element  $e_i$  by multiplying it  
 75 with a scaling factor, i.e.,  $IS'(e_i) = IS(e_i) \cdot \frac{c_{\max}}{c_i}$ , where  $c_i$  is the number of times element  
 76  $e_i$  is actually covered in the solution, and  $c_{\max}$  is the maximum coverage count among all  
 77 elements. Through multiple rounds of score refinement and reconstruction, higher-quality  
 78 initial solutions can typically be obtained, albeit with increased construction time.

## 2.3 Element-Weighted Local Search

Starting from a feasible initial cover, we iteratively remove a randomly selected set and attempt to reconstruct a feasible cover without increasing the number of sets used. Once such a reconstruction is successful, we obtain an improved solution. Thus, our optimization focuses on how to achieve full coverage using a fixed number of sets.

### 2.3.1 Weighting Technique

Weighting techniques have demonstrated strong effectiveness in various set covering-related problems [2, 3, 4]. Our solver adopts a similar strategy: we assign weights to currently uncovered elements, and seek to minimize the total weight of uncovered elements during local search. Compared to minimizing just the count of uncovered elements, the weighted objective yields a smoother search landscape and better optimization performance.

### 2.3.2 Neighborhood Search

The neighborhood is defined by a pairwise swap operation: removing one set from the current solution and adding another. In each iteration:

- Randomly select an uncovered element  $e$  such that the subsequent swap operation ensures it will be covered. This random selection strategy enhances the diversification of the search while simultaneously reducing the evaluation time complexity.
- For each element  $e$ , we consider adding a set that covers it and simultaneously removing one set from the current solution. We evaluate all swap pairs and select the one that minimizes the total weight of uncovered elements after the move.

When the search reaches a local optimum, we increase the weight of a random uncovered element, encouraging its coverage in future iterations. Tabu search is a well-known metaheuristic for combinatorial optimization [1]. We integrate a one-iteration recency-based tabu mechanism that temporarily forbids recently involved sets from participating in swaps, thus encouraging search diversification and preventing cycling. Upon finding a new feasible solution (i.e., all elements are covered), we proactively remove a random set, and the optimization moves to the new bottleneck of one fewer set. The search procedure is repeated until the time limit is reached and the best solution found so far is returned.

## 2.4 Additional Enhancements

For instances that contain multiple connected components after reduction, we introduce two dedicated optimization strategies:

- Reducing the number of connected components: We apply a simple branch-and-bound algorithm to exactly solve connected components that involve fewer than  $k$  sets (with  $k = 23$  in our implementation).
- Restricting active components: Initially, the removal of a set introduces uncovered elements in only one connected component. In subsequent swap iterations, if the added set fails to restore full coverage within this component while the removed set belongs to a different component, we revert the current component to its last feasible state.

117      **References**      118

- 119      1      Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers*  
120           *& operations research*, 13(5):533–549, 1986.
- 121      2      Zhouxing Su, Qingyun Zhang, Zhipeng Lü, Chu-Min Li, Weibo Lin, and Fuda Ma. Weighting-  
122           based variable neighborhood search for optimal camera placement. In *Proceedings of the AAAI*  
123           *Conference on Artificial Intelligence*, volume 35, pages 12400–12408, 2021.
- 124      3      Qingyun Zhang, Zhipeng Lü, Zhouxing Su, and Chumin Li. A vertex weighting-based double-  
125           tabu search algorithm for the classical p-center problem. *Computers & Operations Research*,  
126           160:106373, 2023.
- 127      4      Qingyun Zhang, Zhipeng Lü, Zhouxing Su, Chumin Li, Yuan Fang, and Fuda Ma. Vertex  
128           weighting-based tabu search for p-center problem. In *Proceedings of the Twenty-Ninth In-*  
129           *ternational Conference on International Joint Conferences on Artificial Intelligence*, pages  
1481–1487, 2021.