ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume 29, Number 1, 2026, 89–99
doi: 10.59277/ROMJIST.2026.1.08

Learning-Guided Adaptive Search Optimization for the Weighted Independent Set Problem

Caixia CHEN[1], Jie WU[1], Jie CHEN[2,*], Yu XIA[3], and Radu-Emil PRECUP[4,5]

[1]Department of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi, China
[2]Department of Electronic Engineering, National University of Defense Technology, Hefei, China
[3]State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai, China
[4]Department of Automation and Applied Informatics, Politehnica University of Timisoara, Timisoara, Romania
[5]Center for Fundamental and Advanced Technical Research, Romanian Academy – Timisoara Branch, Timisoara, Romania

6233115001@stu.jiangnan.edu.cn, jiewu20@jiangnan.edu.cn, 706945180@qq.com*, rainyx@sjtu.edu.cn, radu.precup@upt.ro

* Corresponding author

Abstract. The weighted independent set problem is a classic combinatorial optimization problem with broad applications. However, since practical applications require high solution quality, effectively balancing solution efficiency and quality remains a significant challenge. This paper proposes a new hybrid solution method that combines neural network heuristics with an exact search, using heuristics to reduce the search space effectively. Based on the predicted probability, the solving process is divided into two stages. First, a graph neural network is used to predict the marginal probability of vertices belonging to the solution, and deep reinforcement learning trains a policy to construct a high-quality initial solution. Second, an adaptive search strategy is proposed, which dynamically defines a search range based on the credibility of the predicted probability. By reducing the size of the space that the exact solver needs to search, the solution time can be significantly decreased. The experimental results show that the proposed hybrid method improves the average solving speed by 86% while maintaining the same solution quality as the exact solver. Meanwhile, it demonstrates good scalability for large-scale graphs.

Key-words: Combinatorial optimization; exact search; graph neural network; weight independent set; machine learning.

1. Introduction

The weighted independent set (WIS) problem isa widely studied problem with significant applications in diverse fields, including multi-agent task allocation [1], wireless network [2–4], and social networking [5]. A WIS is a subset of vertices that are not connected to each other in the weighted graph. In wireless communication, when multiple devices transmit signals simultaneously, the interfering devices cannot transmit simultaneously. By representing the potential interference relationships between devices or links as edges of the graph, it is natural to model the search for maximum revenue in concurrent transmissions as a WIS problem [6]. In large cellular interference graphs, which may contain tens of thousands of vertices, significant challenges arise for the hardware and efficiency of traditional integer linear programming (ILP) solvers. The branch-and-bound method is the main framework for exact algorithms, which systematically searches for all feasible solutions and combines branching and pruning to accurately solve them [7]. However, as the scale of the graph increases, the solving time increases exponentially.

To quickly obtain high-quality solutions, a graph kernel technique was proposed in [8] to recursively approximate the diagram. In [9], a set of reductions were proposed to reduce the scale of the problem. Moreover, numerous heuristics have been proposed, which apply heuristic rules to rapidly produce high-quality solutions. For example, in [10], a fast local search algorithm was proposed, which improves both efficiency and solution quality by iteratively breaking the current solution and greedily reconstructing it using two low-complexity heuristics to escape local optima. In [11], a supervised learning framework was proposed that combined tree search methods to generate multiple solutions. In [12], a reinforcement learning framework was proposed to incrementally construct solutions by adding vertices to the solution greedily. In [13], the authors introduced a delay action to reduce the number of solving stages.

In addition to using neural networks to discover heuristics, the reduction in problem size can effectively lower the computational complexity. In [14], a post-hoc method based on neural diving was proposed to solve ILP problems, which uses a trust threshold method to fix high-confidence variables to reduce the size of the problem. In [15], a predict-and-search framework was proposed that constructs partial solutions and then searches for feasible solutions within the trust region of the partial solutions. Compared to the trust threshold method, trust region-based methods retain more flexibility and reduce the infeasible risk caused by fixed error variables. However, current methods cannot adaptively define trust regions based on the problem instances.

To effectively balance the quality and time of the solution, this paper proposes a two-stage solution method based on the predicted probability. First, deep reinforcement learning (DRL) is used to develop a policy that utilizes the marginal probability of vertices belonging to the solution to construct an initial solution. Then, based on the predicted probability of the class label in the initial solution, we adaptively define a search range centered around the solution to refine it.

The adaptive search method rests on two key observations. First, solutions produced by neural heuristics lie only a small Manhattan distance from the known optimum. By exploring every feasible assignment within a fixed radius around the initial solution, we can locate higher-quality solutions; when the radius becomes large enough, the optimal solution will certainly be reached. A temperature scaling parameter regulates the balance between solution quality and running time. Second, the expected distance between the initial and optimal solutions depends on the predicted inclusion probability of each vertex. A transformation from the predicted probability to the calibrated confidence values is performed by means of a calibration function [16]. The confidence values define a spherical trust region that assigns larger radii to vertices with low confidence and smaller radii to those with high confidence, directing computational effort where it is most

effective. The main contributions of this paper are as follows

- The policy of constructing a solution using probability maps is learned through DRL. The initial solution is then used as the center of the exact search.
- Based on the prediction accuracy, a confidence-guided adaptive search method is proposed. This method dynamically defines the search space and prunes the exact search.

The remainder of this paper is organized as follows: Section 2 describes the research problem. Section 3 presents the proposed method. Section 4 illustrates the experiments results and conducts a detailed analysis. In Section 5, the entire paper is summarized.

2. Problem Description

Let $G = \{V, E\}$ denote a weighted undirected graph, where $V = \{v_1, \cdots, v_n\}$ is the set of vertices, each vertex $v_i \in V$ having a weight $w_i > 0$. $E = \{e_{i,j}\}$is the set of edges, $e_{i,j} = (v_i, v_j)$ is the edge between v_i and v_j. The neighbor set of vertex v_i is denoted by $N(v_i) = \{v_j | (v_i, v_j) \in E, \forall v_j \in V\}$. A weighted independent set, denoted by $V_{\mathrm{WIS}}, V_{\mathrm{WIS}} \subseteq V$, is a subset of vertices that for any edge $e_{i,j}$ in the graph, there is only one endpoint in the V_{WIS}.

The state of vertex v_i is denote as x_{i}, $x_{\mathrm{i}} \in \{0, 1\}$. If $x_{\mathrm{i}} = 1$, then vertex v_{i} belongs to the V_{WIS}; otherwise, it does not. A WIS problem can be defined as

$$\begin{aligned} \max_{x_v} \quad & \sum_{v \in V} w_v x_v, \\ \text{s. t.} \quad & x_i + x_j \leq 1, \forall (v_i, v_j) \in E, \\ & x_v \in \{0, 1\}, \forall v \in V. \end{aligned} \tag{1}$$

$\{V_{\mathrm{WIS}}\}$ is the set of V_{WIS} for a graph. If the state profile $x = (x_1, \cdots, x_n)$ satisfies $\{i | x_i = 1, \forall i \in V\} \in \{V_{\mathrm{WIS}}\}$, then x is a solution to the WIS problem. Among them, the V_{WIS} with the maximum sum of weights is the optimal solution to the problem.

3. Methods

In this section, we propose a hybrid solution method that integrates neural network heuristics with an exact search. This method comprises two stages: initial solution generation and an adaptive search. This method utilizes a graph convolutional network (GCN) model to predict the marginal probability of vertices belonging to the solution and constructs an initial solution using a policy trained by DRL. The adaptive search method constructs a trust region based on the confidence of vertices in the initial solution as the search space.

3.1. Probability prediction learning

In this subsection, we employ supervised learning to train a GCN for the probability prediction of vertex classification. Vertices are classified as either included in V_{WIS} or excluded from V_{WIS}. The GCN is trained using known optimal solutions as labels and learns to map vertices

and their local graph information to the classification probabilities [17]. The input feature is the weight of the vertex, represented by $\mathcal{V} \in \mathbb{R}^{|V|\times 1}$. The output is the classification probability, denoted by $P \in [0,1]^{N\times 2}$. Let $\mathcal{D} = (G_i, l_i)$ represent the training set, where G_i represents an individual graph, and $l_i \in \{0,1\}^{N\times 1}$ represents an optimal solution for the WIS problem. The binary vector l_i indicates which vertices belong to V_{WIS}.

The GCN model consists of L graph convolution layers and an output layer, implemented as a multi-layer perceptron. The graph convolutional layers use the rectified linear unit as the non-linear activation function, and the output layer employs the softmax function. Cross-entropy loss is used as the loss function for vertex classification. For each training sample (G_i, l_i), the cross-entropy loss is

$$\varphi = \sum_{j=1}^{n} \{l_{i,j}\log(p_{i,j}^1) + (1-l_{i,j})\log(p_{i,j}^0)\}, \tag{2}$$

where $l_{i,j}$ is the j-th element of l_i, $p_{i,j}^1$ is the probability that vertex v_j belongs to V_{WIS}, and $p_{i,j}^0$ is the probability that vertex v_j does not belong to V_{WIS}.

3.2. Strategy learning for constructing solutions

We can model the WIS problem as a Markov decision process and train an agent to learn the optimal policy. The goal of policy training is to optimize the objective value while ensuring that the constraint of the independent set problem is satisfied.

The state vector s represents the current states of each vertex $s = \{x_i\}^n, i \in V, x_i \in \{0, 1, *\}$, where $*$ represents the undecided state, 0 indicates that the vertex does not belong to the V_{WIS}, and 1 indicates otherwise. Initially, the state of each vertex is undecided, $s_0 = \{*\}^n$. The action vector a represents the decisions for the states of the remaining undecided vertices, $a = [x_i \colon i \in V_*] \in \{0,1\}^{|V_*|}$, where V_* represents a subset of undecided vertices. If $a_i = 1$, vertex i is added to V_{WIS}. If $a_i = 0$, the vertex remains in an undecided state. To ensure that the independent set constraint is satisfied at each step, a dynamic masking mechanism is used to identify adjacent vertices that are both $a = 1$. The states of these vertices are then reset to 0. The state transition is deterministic. The vertex states are updated according to actions, and the state of the neighboring vertices of vertices with $a = 1$ is set to 0. The sum of the vertex weights selected at each step is the reward, $r = \sum_{i=1}^{n} w_i a_i$.

For each state, the trained model is first used to obtain the predicted probability P, which serves as the vertex feature to determine the action using policy π. Proximal Policy Optimization (PPO) is adopted for training within the actor-critic framework [18]. The actor network updates policy π by maximizing the cumulative reward. The critic network, denoted as $V^\pi(s)$, estimates the expected cumulative reward for a given state s_t as follows

$$V^\pi(s_t) = \mathbb{E}_{a_\tau \sim \pi}\left[\sum_{\tau=0}^{T-t} \gamma^\tau R^{t+\tau} \,\middle|\, s_0 = s_t\right], \tag{3}$$

where γ is the discount factor.

The advantage function at step t, denoted as $\hat{A}_t$, measures the difference between the actual cumulative reward and the estimated value of state s_t, and is computed as

$$\hat{A}_t = \sum_{\tau=0}^{T-t} \gamma^\tau R^{t+\tau} - V^\pi(s_t). \tag{4}$$

The optimization objective function, $L(\theta)$ is expressed as

$$L(\theta) = E(min(r(\theta)\hat{A}, clip(r(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A})), \tag{5}$$

$$r(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}, \tag{6}$$

where ε is the clipping factor, and $r(\theta)$ is the ratio of the new policy to the old policy.

3.3. Adaptive search

Next, we refine the solution by conducting a targeted search within a defined trust region around the initial solution. The proposed adaptive search is based on two key observations. First, the initial solution, denoted as $\mathbf{x}^0$, is hypothesized to be close to the optimal solution, as measured by the ℓ_1-norm. Therefore, the search is restricted to a trust region of limited size, centered around $\mathbf{x}^0$. This trust region encompasses all feasible solutions within a specified search radius of $\mathbf{x}^0$, where the search radius is defined by the ℓ_1 distance from $\mathbf{x}^0$.

Second, the true distance between $\mathbf{x}^0$ and the known optimal solution is related to the predicted probability of the vertex. To evaluate the true distance, the correlation between the prediction probability and accuracy is first obtained. Confidence calibration is a strategy for adjusting the original output of the model to reflect the true likelihood of correctness [19]. We use temperature scaling to calibrate the prediction probability to obtain confidence. This confidence reflects the reliability of the vertex state in $\mathbf{x}^0$ and serves as an estimate of the prediction accuracy. Based on the prediction accuracy, the difference between the total number of vertices and the number of correct predictions is approximately the true distance, which is also the search radius. To minimize the evaluation error, the vertices are divided into multiple groups according to confidence. The average confidence is used as the prediction accuracy to calculate the search radius for each group. Let D_s represent the true distance between $\mathbf{x}^0$ and the optimal solution of group s, and r_s is the search radius. When the r_s of each segment can cover the D_s, the optimal solution is included in the search space.

This adaptive search dynamically adjusts the search range based on the average confidence of the vertices, obviating the difficulty of pre-setting the parameters. Specifically, given a graph $G = (V, E)$ with $|V| = n$ vertices, the number of groups, denoted by S, is adaptively determined based on the size of the graph

$$S = \lfloor \log n \rfloor + 1. \tag{7}$$

Based on the predicted probability of vertex class in $\mathbf{x}^0$, the vertices are sorted in descending order and then partitioned into S groups, denoted as $\{V_1, ..., V_S\}$. Each group contains $n_s = \lceil n/S \rceil$ vertices. For each group, the r_s is calculated by the average confidence in V_s

$$\begin{aligned} r_s &= (1 - \lambda\mu_s)n_s, \\ \mu_s &= \frac{1}{n_s}\sum_{v \in V_s} p_v, \end{aligned} \tag{8}$$

where λ is the scaling factor, p_v is the predicted probability for vertex v. The λ is used to calibrate the probabilities.

Table 1. The comparison results of the average maximum objective value and the time to find it within the time limit of 1000s on synthetic and real graphs of KAMIS, SCIP, PS-SCIP, Gurobi, and our proposed ASS-Gurobi and ASS-SCIP

Graph	n	KAMIS		SCIP		PS-SCIP		ASS-SCIP		Gurobi		ASS-Gurobi	
		Obj	Time	Obj	Time	Obj	Time	Obj	Time	Obj	Time	Obj	Time
BA $< 6 >$	500	200.42	0.9	**200.44**	26.0	**200.44**	15.0	**200.44**	6.3	**200.44**	2.0	**200.44**	0.7
BA $< 6 >$	2000	443.73	0.5	448.34	708.0	**448.98**	684.1	**448.98**	18.6	**448.98**	62.0	**448.98**	1.8
BA $< 6 >$	5000	1117.62	0.8	1121.7	837.0	1125.02	220.1	**1125.72**	101.2	**1125.72**	546.0	**1125.72**	2.5
ER $< 4 >$	500	99.83	0.4	**100.83**	149.0	**100.83**	24.0	**100.83**	12.2	**100.83**	4.0	**100.83**	2.3
ER $< 4 >$	1500	307.00	0.4	308.76	570.0	309.10	722.1	309.06	242.5	**309.11**	102.0	309.09	20.5
ER $< 4 >$	5000	1009.56	0.6	989.11	286.0	1012.48	897.1	1016.96	1003.8	1026.92	533.0	**1027.47**	360.8
Ego-facebook	4039	1084.19	2.7	1084.40	849.0	1083.13	983.0	1084.88	682.0	**1086.21**	35.0	**1086.21**	12.0
Athletes	13866	23018.34	6.4	22071.01	150.0	22414.69	37.0	22571.03	30	23109.60	89.0	**23213.4**	931.0
HU	47538	19588.30	5.1	18197.64	12.0	19047.63	11.0	19146.03	9.0	19708.44	867.0	**19908.45**	930.0
HR	54573	19796.70	4.3	18432.69	26.0	18758.61	24.0	18837.08	715.0	19838.73	113.0	**19978.34**	547.0

Note: Bold values indicate the best objective values obtained. Time measured in seconds.

A constraint is introduced to allow only r_s vertices to change their state relative to $\mathbf{x}^0$ for group s. An ILP model is then constructed as follows to refine the solution

$$\begin{aligned} \max_{x_v} \quad & \sum_{v \in V} w_v x_v, \\ \text{subject to} \quad & x_i + x_j \leq 1, \forall (v_i, v_j) \in E, \\ & \sum_{v \in V_s} \|x_v - x_v^0\|_1 \leq r_s, s = 1, ..., S \\ & x_v \in \{0, 1\}, \forall v \in V. \end{aligned} \tag{9}$$

The resulting ILP model is then solved using a ILP solver. Algorithm 1 presents the pseudocode for adaptive search method.

Adaptive search constructs a high-confidence search region through confidence calibration. When calibrated accurately, the resulting trust region encompasses the global optimum solution. The search radius r_s for each segment limits the number of vertices that can be modified within that segment compared to x^0. Consequently, the search space size across the entire trust region is constrained to $\prod 2^{r_s}$. Within this finite space, the ILP solver performs an exact search to explore all feasible solutions and accelerates the search through bound pruning. Consequently, the adaptive search ultimately converges to the optimal solution within the trust region and converges to the global optimum when calibration is accurate.

4. Results and Discussion

This section provides a detailed presentation of the experimental process, results, and analysis. The experiments are conducted on synthetic and four real graphs [20], [21]. This paper uses networkX [22] to generate graphs of varying scales, specifically Erdős-Rényi (ER) graphs with an average degree of 4 and Barabási-Albert (BA) graphs with an average degree of 6. Each vertex

Algorithm 1 The Proposed Adaptive Search Method

1. Obtain probability of vertex: $p_v \leftarrow P(v, \mathbf{x}_0(v)), \forall v \in V$
2. Sort vertices: $\mathcal{V}_{\text{sorted}} \leftarrow \text{sort}(V, p_v, \text{descending})$
3. Compute group number by Eqs. (7)
4. Partition the vertices evenly into S groups: $\{V_1, \ldots, V_S\} \leftarrow \text{split}(V_{\text{sorted}}, S)$
5. Compute search radius by Eqs. (8) to obtain $r_s, \forall s \in 1, \ldots, S$
6. Build ILP model $\mathcal{M}$ by Eqs. (9) and the partial details are as follows
7. Set $s \leftarrow 1$
8. If $s > S$, go to Step 16
9. Create binary variables $\delta_u \in \{0, 1\},\ \forall u \in V_s$
10. Set $index \leftarrow 1$
11. If $index > |V_s|$, go to Step 14
12. Let u be the $index$-th vertex in V_s
    If $x_0(u) = 0$:
    Add constraint: $x_u \leq \delta_u$
    Else:
    Add constraint: $1 - x_u \leq \delta_u$
13. Set $index \leftarrow index + 1$ and go to Step 11
14. Add constraint: $\sum_{u \in V_s} \delta_u \leq r_s$
15. Set $s \leftarrow s + 1$ and go to Step 8
16. $\mathbf{x}^* \leftarrow \text{Solver}(\mathcal{M})$
17. Return $\mathbf{x}^*$

is assigned a weight drawn from a normal distribution. The open-source solver SCIP 5.5.0 [23] and the commercial solver Gurobi 11.0.2 (Academic Edition) are employed as ILP solvers.

For supervised learning label acquisition, Gurobi is employed to obtain the optimal solutions on BA graphs with 500 vertices and ER graphs with 300 vertices. The obtained models is directly applied to larger-scale graphs. The evaluation metrics included the maximum objective value (OBJ) identified within the specified time limit and the time required to obtain that solution. The average optimality gap is calculated to quantify the difference between the current solution and the best-known solution (BKS), $gap = |BKS - OBJ|/(BKS + 10^{-9})$.

Table 1 presents a comparison of traditional ILP solvers (SCIP and Gurobi), the independent-set solver KAMIS [8], the PS-SCIP scheme [15], and our adaptive search strategy ASS. ASS-

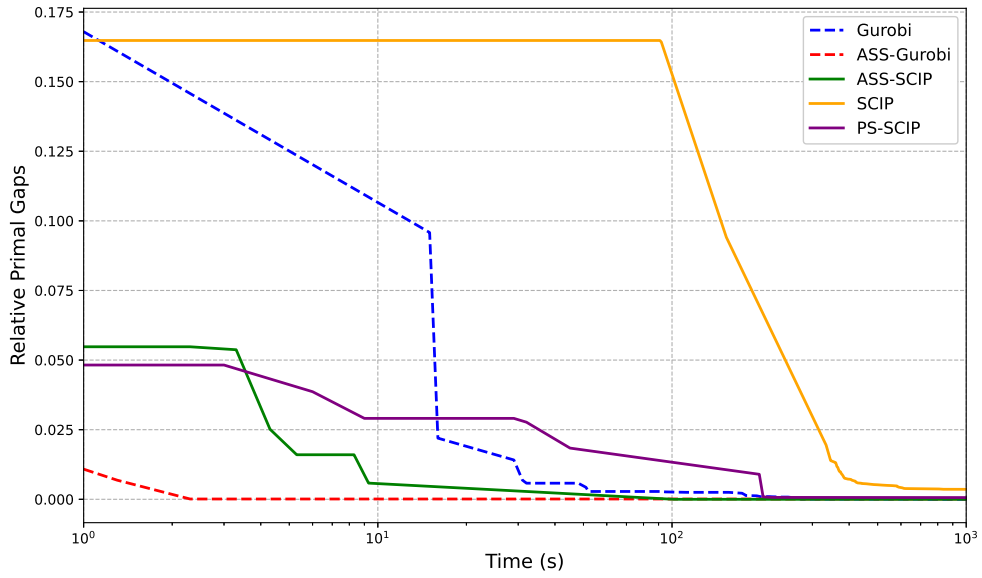

Fig. 1. Average relative primal gap versus solving time of Gurobi, SCIP, PS-SCIP, and our proposed ASS-Gurobi and ASS-SCIP.

SCIP indicates that SCIP solves the reduced ILP model produced by ASS. In ASS, vertices are ranked by prediction confidence and partitioned into groups, and each group may flip at most a prescribed number of variables. This group-wise radius assignment significantly reduces the feasible space and lowers solver effort. On BA graphs, ASS-SCIP and ASS-Gurobi match vanilla Gurobi's objective values exactly, confirming that ASS preserves optimality. At the same time, ASS-Gurobi cuts average solve time by 86% compared with Gurobi. On the largest ER graphs with $n = 5000$, ASS-Gurobi even improves upon Gurobi's best-known solution, demonstrating scalability to large graphs.

By contrast, PS-SCIP [15] fixes the k_1 variables with the highest marginal probability to one and the k_0 variables with the lowest probability to zero. Then, PS-SCIP enforces a global Hamming distance bound Δ around the partial assignment before solving the resulting ILP. We follow the original tuning, the parameters (k_0, k_1) are set to $(700, 0)$ at $n = 1500$ with other sizes scaled proportionally. Δ is varied from 0 to 20, and each method's best outcome is reported. Under these settings, ASS-SCIP outperforms PS-SCIP in both runtime and objective quality. The result may be attributed to PS method's use of a single threshold for high-confidence variables and a fixed search radius. ASS, however, adjusts the radius per confidence level and prunes low-uncertainty vertices more aggressively, which leads to a substantially smaller search space. KAMIS applies reduction rules and local heuristics to reduce the graph before branching. It quickly finds high-quality bounds but often stalls during exact search due to a greedy branching rule and the absence of learned heuristics. In contrast, ASS-guided solvers navigate directly to the optimal region and consistently finish within the time limit.

Fig. 1 illustrates the optimality gap as a function of solving time on the BA graphs with $n = 5000$ for various methods. Note that the reported times for ASS-Gurobi and ASS-SCIP include both model prediction and exact search. ASS-Gurobi achieves the lowest gap from the

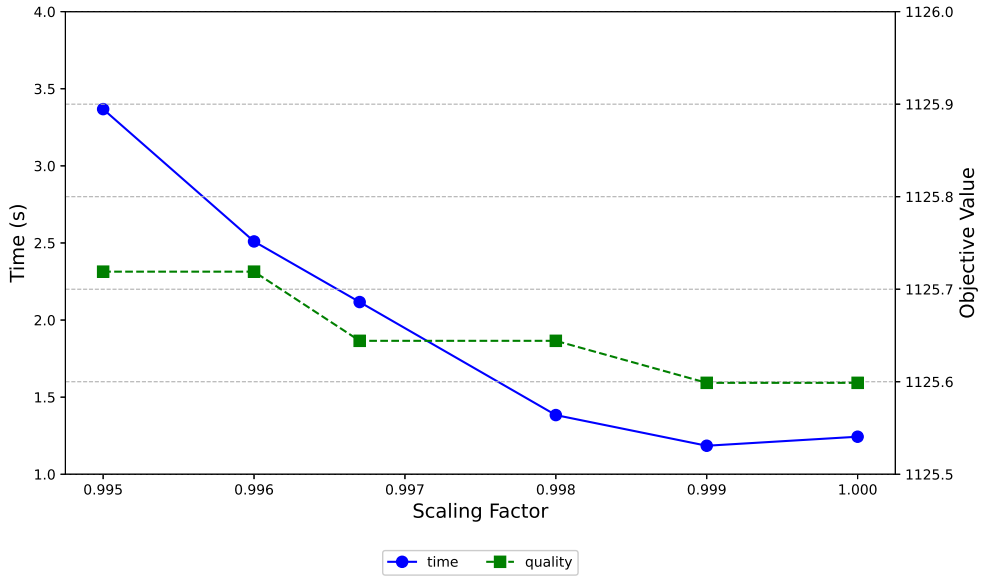

Fig. 2. The effect of different scaling factors for BA with $N = 5000$.

outset, with negligible further improvement after approximately 1 s. ASS-SCIP closes the gap within 10 s. These results demonstrate the efficiency advantage of ASS.

Fig. 2 shows the effect of varying the scaling factor λ on ASS-Gurobi's performance. The parameter λ directly controls the trade off between objective quality and runtime, and larger λ values place more trust in the neural predictions which yields a smaller search radius and thus faster solve times at the risk of slight objective degradation. We select λ post-hoc by validating over a range of candidate values. Overall, the experiments confirm that ASS, especially when paired with Gurobi, offers superior scalability and efficiency without compromising solution quality.

5. Conclusions

In this paper, a hybrid solution method that combines neural network heuristics and exact search is proposed to solve the weighted independent set problem. The network model is used to predict the marginal probability of vertices belonging to solutions, and the policy for constructing solutions is obtained through deep reinforcement learning. Then, an adaptive search method is used to conduct exact search within the search range centered around the initial solution.

This method adaptively divides the vertices into different groups based on their confidence and calculates the search radius. The experimental results on graphs of different scales demonstrate the efficiency of the method, which can greatly improve the solving efficiency while ensuring the quality of the solution.

Acknowledgements. This work was partly supported in part by Youth Fund of the National Natural Science Foundation of China under Grant 62103166, the Jiangsu Shuangchuang (Innovation and Entrepreneurship) Talent Program [No.JSSCBS20210840], in part by the National Natural Science Foundation of China under Grant 62073345, and a grant of the Romanian Ministry of Research, Innovation and Digitization, CNCS/CCCDI - UEFISCDI, project number ERANET-ENUAC-e-MATS, within PNCDI IV.

References

[1] J. CHEN, Y. GUO, Z. QIU, B. XIN, Q.-S. JIA and W. GUI, *Multiagent dynamic task assignment based on forest fire point model*, IEEE Transactions on Automation Science and Engineering **19**(2), 2021, pp. 833–849.

[2] A. MATEESCU, D.-C. POPESCU, I.-L. STEFAN, I.-M. VLASCEANU, A.-C. PETRESCU-NITA, I. SACALA and I. DUMITRACHE, *Machine learning control for assistive humanoid robots using blackbox optimization of PID loops through digital twins*, Romanian Journal of Information Science and Technology **28**(1), 2025, pp. 63–76.

[3] E.-K. ZAVADSKAS, H. DINCER, S. YUKSEL and S. ETI, *Intelligent expert systems using molecular fuzzy genetic algorithms and multi-objective particle swarm optimization for circular-oriented project investments*, Romanian Journal of Information Science and Technology **28**(3), 2025, pp. 260–273.

[4] R.-E. PRECUP, C.-A. BOJAN-DRAGOS, K. GAO and S. CUI, *Problem setting for trajectory planning and cruise control of a connected autonomous electric bus in intersection scenarios with human-driven vehicles to optimize energy, comfort and tracking*, Romanian Journal of Information Science and Technology **28**(3), 2025, pp. 299–312.

[5] C.-C. CHANG, M.-Y. CHANG, J.-Y. JHANG, L.-Y. YEH and C.-Y. SHEN, *Learning to extract expert teams in social networks*, IEEE Transactions on Computational Social Systems **9**(5), 2022, pp. 1552–1562.

[6] Y. JIANG, Z. ZHOU, J. ZHOU, C. QIN, W. TANG, Z. CAI, D. ZHAI and R. ZHANG, *Coverage maximization for air-and-ground cooperative networks*, IEEE Wireless Communications Letters **12**(12), 2023, pp. 2113–2117.

[7] M. ZHENG, J.-K. HAO and Q. WU, *Exact and heuristic solution approaches for the generalized independent set problem*, Computers & Operations Research **164**, 2024, paper 106561.

[8] S. LAMM, C. SCHULZ, D. STRASH, R. WILLIGER and H. ZHANG, *Exactly solving the maximum weight independent set problem on large real-world graphs*, Proceedings of the 2019 SIAM Workshop on Algorithm Engineering and Experiments, San Diego, CA, USA, 2019, pp. 144–158.

[9] J. GU, W. ZHENG, Y. CAI and P. PENG, *Towards computing a near-maximum weighted independent set on massive graphs*, Proceedings of the 27$^{\text{th}}$ ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Singapore, Singapore, 2021, pp. 467–477.

[10] R. LI, S. HU, S. CAI, J. GAO, Y. WANG and M.-Y. AND, *NuMWVC: A novel local search for minimum weighted vertex cover problem*, Journal of the Operational Research Society **71**(9), 2020, pp. 1498–1509.

[11] Z. LI, Q. CHEN and V. KOLTUN, *Combinatorial optimization with graph convolutional networks and guided tree search*, Advances in Neural Information Processing Systems **31**, 2018, pp. 537–546.

[12] E. KHALIL, H. DAI, Y. ZHANG, B. DILKINA and L. SONG, *Learning combinatorial optimization algorithms over graphs*, Advances in Neural Information Processing Systems **30**, 2017, pp. 6351–6361.

[13] S. AHN, Y. SEO and J. SHIN, *Learning what to defer for maximum independent sets*, Proceedings of the 2020 International Conference on Machine Learning, Vienna, Austria, 2020, pp. 134–144.

[14] T. YOON, *Confidence threshold neural diving*, arXiv preprint arXiv:2202.07506, 2022.

[15] Q. HAN, L. YANG, Q. CHEN, X. ZHOU, D. ZHANG, A. WANG, R. SUN and X. LUO, *A GNN-guided predict-and-search framework for mixed-integer linear programming*, arXiv preprint arXiv:2302.05636, 2023.

[16] H. YANG, M. WANG, Q. WANG, M. LAO and Y. ZHOU, *Balanced confidence calibration for graph neural networks*, Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Barcelona, Spain, 2024, pp. 3747–3757.

[17] T.-N. KIPF and M. WELLING, *Semi-supervised classification with graph convolutional networks*, arXiv preprint arXiv:1609.02907, 2016.

[18] Y. GU, Y. CHENG, C.-L.-P. CHEN and X. WANG, *Proximal policy optimization with policy feedback*, IEEE Transactions on Systems, Man, and Cybernetics: Systems **52**(7), 2022, pp. 4600–4610.

[19] C. GUO, G. PLEISS, Y. SUN and K.-Q. WEINBERGER, *On calibration of modern neural networks*, Proceedings of the 2017 International Conference on Machine Learning, Sydney, Australia, 2017, pp. 1321–1330.

[20] B. ROZEMBERCZKI, R. DAVIES, R. SARKAR and C. SUTTON, *GEMSEC: graph embedding with self clustering*, Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Vancouver, BC, Canada, 2019, pp. 65–72.

[21] R.-A. ROSSI and N.-K. AHMED, *The network data repository with interactive graph analytics and visualization*, Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, TX, USA, 2015, pp. 4292–4293.

[22] A. HAGBERG, P.-J. SWART and D.-A. SCHULT, *Exploring network structure, dynamics, and function using networkx*, Technical Report, Los Alamos National Laboratory, Los Alamos, NM, USA, 2008.

[23] K. BESTUZHEVA, M. BESANÇON, W.-K. CHEN, A. CHMIELA, T. DONKIEWICZ, J. VAN DOORNMALEN, L. EIFLER, O. GAUL, G. GAMRATH, A. GLEIXNER et al., *The SCIP optimization suite 8.0*, arXiv preprint arXiv:2112.08872, 2021.