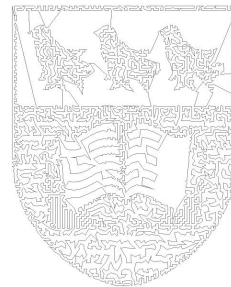


# **Solving Combinatorial Optimization Problems using Statistical Learning**

**Application on the Vehicle Routing and Bin Packing Problems**

**Andrew Naguib**



**Electrical and Computer Engineering  
University of Victoria  
Victoria, BC  
Canada**

**12 April, 2023.**

# Contents

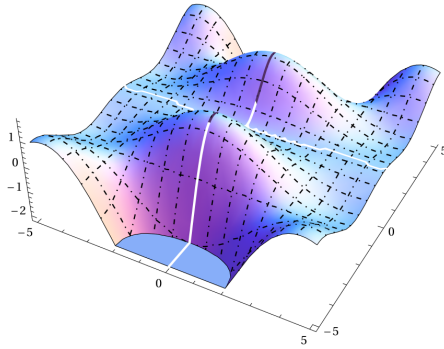
<b>Preliminaries</b>	<b>2</b>
combinatorial optimization . . . . .	2
integer program . . . . .	3
Integer program relaxation . . . . .	3
primal-dual method . . . . .	4
branch-and-bound . . . . .	4
strong branching (SB) . . . . .	4
reliable pseudo-cost (RPC) . . . . .	4
capacitated vehicle routing problem (CVRP) . . . . .	5
bin packing problem (BPP) . . . . .	6
mathematical solvers . . . . .	6
<b>Method</b>	<b>7</b>
graph convolutional neural network . . . . .	7
GraphSAGE . . . . .	7
graph attention neural network . . . . .	7
<b>Evaluation</b>	<b>8</b>
criteria . . . . .	8
visualization . . . . .	9
results . . . . .	10
findings . . . . .	14
<b>Discussion</b>	<b>15</b>
	<b>17</b>

# Preliminaries

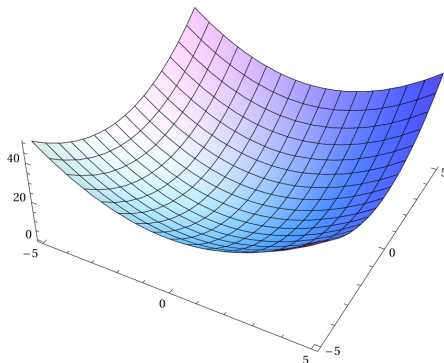
## combinatorial optimization

Let  $x \in \mathbb{R}^n, f, g, h : \mathbb{R}^n \rightarrow \mathbb{R}$ . An optimization problem has the following form:

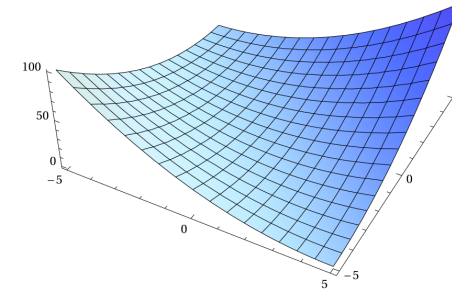
$$\begin{aligned} & \underset{x}{\text{minimize}} && f(x) \\ & \text{subject to} && g_i(x) \leq 0 \quad i = 1, \dots, m, \\ & && h_j(x) = 0 \quad j = 1, \dots, p \end{aligned} \quad (1)$$



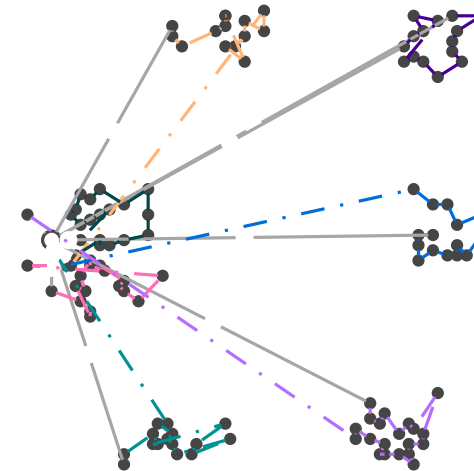
if,  $f_i(x)$  is strictly convex, then a unique global optimum exist



if  $f_i(x), g_i(x)$  are restricted to be convex functions and  $h_j(x)$  to be an affine one, then we have what we call a convex optimization problem



if  $x$  is a discrete variable and is an enumeration, a combination, or a permutation of a set of elements, then we have a combinatorial optimization problem



## integer program

Let  $c, l, u \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m \times n}$ ,  $b \in \mathbb{R}^m$ ,  $I \subseteq \{1, 2, \dots, n\}$ , and  $l, u \in \mathbb{R}^n$ . An IP can be defined as follows:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^T x & (2a) \\ \text{subject to} \quad & Ax \leq b, & (2b) \\ & l \leq x \leq u, & (2c) \\ & x_i \in \mathbb{Z} \quad i \in I. & (2d) \end{aligned}$$

A feasible assignment  $x_i = z_i$  would satisfy the above constraints (2b)-(2d), and an optimal one  $z^*$ , called the minimizer, is both feasible and minimizes the objective (2a). For a shorthand notation, let  $\delta_i$  refer to  $A_i x_i \leq b_i$  in (2b).

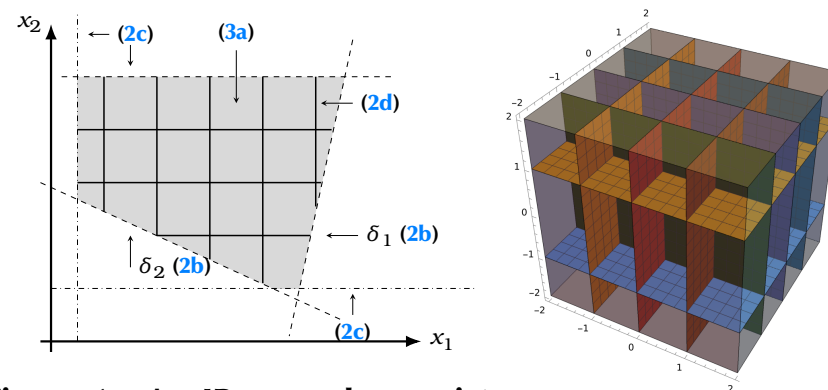


Figure 1: An IP example consisting of two variables and three constraints.

## Integer program relaxation

The linear programming relaxation (LP-relaxation) for the IP defined above would be:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^T x & (3a) \\ \text{subject to} \quad & Ax \leq b, & (3b) \\ & l \leq x \leq u. & (3c) \end{aligned}$$

That is, the integrality constraints are relaxed, which also makes the problem convex. Clearly, the optimal solution to (3a) is also optimal to the original problem (2a) if and only if it satisfies (2d).

Figure 1 is an example of an integer program consisting of two variables,  $\{x_1, x_2\}$ , and two constraints,  $\{\delta_1, \delta_2\}$ . The gray shading represents the relaxed problem (3a), where the solid grid lines are the integrality constraints (2d), and the dashed lines, (2c), are the box constraints.

## primal-dual method

The quality of each feasible solution found to (2a) is assessed using the primal-dual method. The primal (lower) bounds  $p^*$  are provided by feasible solutions, and the dual (upper) bounds  $d^*$  by relaxation or duality. The Lagrangian dual problem [1] to (3a) is:

$$\underset{\lambda}{\text{maximize}} \quad -b^T\lambda \tag{4a}$$

$$\text{subject to} \quad A^T\lambda + c \geq 0, \tag{4b}$$

where  $\lambda$  is the Lagrange multiplier or the dual variable for the inequality constraints (3b). The dual problem acts as a certificate on the limit of the performance, i.e., the upper bound that declares optimality of  $p^*$ . The duality gap  $f$  is given by  $p^* - d^*$ . A gap  $f > 0$  or  $f = 0$  indicates a weak- or strong-duality, respectively. We define the relative gap to be:

$$\Delta f = \frac{|p^* - d^*|}{\min\{|p^*|, |d^*|\}}.$$

## branch-and-bound

The two building blocks in the algorithm are:

- **branching:** the problem is divided into several smaller and less constrained ones
- **bounding:** selects which subproblems to solve next.

### strong branching (SB)

carefully branches the tree to guarantee the smallest B&B tree size by performing a one-step look-ahead before deciding to branch. The solver starts by choosing a set of integer variables,  $S$ , that are fractional in the LP-relaxation (when  $S$  represents all of the integer variables, it is the Full Strong Branching policy).

### reliable pseudo-cost (RPC)

assigns an estimated cost to each variable based on the results of the previous subproblems, and occasionally uses SB on the unreliable pseudo-costs according to some predefined reliability constant.

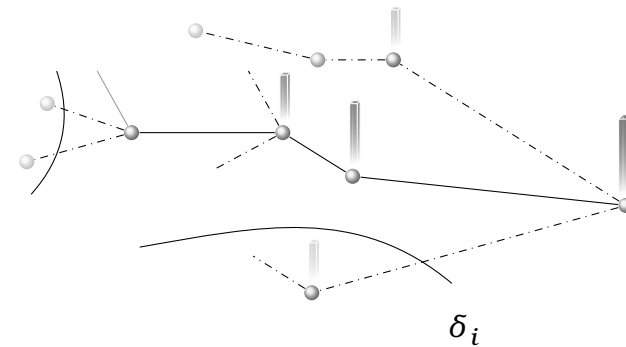


Figure 2: The arcs on the dashed branches are equivalent to the  $\delta_i$ . The bars hovering over each node is a visual representation of the upper bound.

## capacitated vehicle routing problem (CVRP)



**Figure 3: A CVRP example with  $n = 40$  and  $k = 5$ . Given the dispersed points shown in (a), CVRP asks to find the routes presented in (b).**

The goal is to construct a tour for each vehicle where

- (i) Each customer is visited once and by a single vehicle;
- (ii) The demand served by vehicle does not exceed the capacity;
- (iii) Each tour starts and finishes at the depot (to eliminate sub-tours);
- (iv) The tours jointly minimize the cost and serve the total demand.

The corresponding integer program is:

$$\text{minimize}_x \quad \sum_{(i,j) \in \vec{A}} c_{ij} x_{ij} \quad (5a)$$

$$\text{subject to} \quad \sum_{j \in \text{deg}^+(i)} x_{ij} = 1 \quad \forall i \in N, \quad (5b)$$

$$\sum_{i \in \text{deg}^-(i)} x_{ij} = 1 \quad \forall j \in N, \quad (5c)$$

$$\sum_{j \in \text{deg}^+(0)} x_{0j} = k, \quad (5d)$$

$$\sum_{j \in \text{deg}^-(0)} x_{0j} = k, \quad (5e)$$

$$u_i - u_j + Qx_{ij} \leq Q - q_j \quad \forall (i,j) \in \vec{T}, \quad (5f)$$

$$q_i \leq u_i \leq Q \quad \forall i \in N, \quad (5g)$$

$$x_{ij} \in \{0, 1\} \quad \forall (i,j) \in \vec{A}, \quad (5h)$$

where  $x \in \mathbb{Z}^{|\vec{V}| \times |\vec{V}|}$  and  $u \in \mathbb{R}^{|\vec{T}|}$  are the decision variables representing the tours and a constructed tour  $\vec{T}$ . Both (5b) and (5c) ensure (i). The sub-tour elimination and capacity constraints (SEC) (ii) and (iii) are guaranteed by (5f) and (5g), which are the Miller-Tucker-Zemlin (MTZ)-formulation [2]. The integrality constraints are enforced by (5h).

## bin packing problem (BPP)

A formulation of the IP for BPP in terms of CVRP<sup>a</sup> can be written by removing constraints (5f)-(5g) and through a simple algebraic manipulation to minimize the number of vehicles, referred to as bins in this context, in lieu of (5a). This is achieved by expanding the value  $k$  to the bins variable  $K = \{1, 2, \dots, k\}$ , and conditioning that the demands  $q_j$  for  $j \in N$ , represented as weighted items, are all distributed across the available bins without exceeding the bin capacity  $Q$ .

<sup>a</sup>Both problems are NP-hard

$$\underset{k}{\text{minimize}} \quad \sum_i^k k_i \quad (6a)$$

$$\text{subject to} \quad \sum_i^k q_j x_{ij} \leq Q k_i \quad \forall j \in N, \quad (6b)$$

$$\sum_i^k x_{ij} = 1 \quad \forall j \in N, \quad (6c)$$

$$k_i \in \{0, 1\} \quad \forall i \in K, \quad (6d)$$





$$x_{ij} \in \{0, 1\} \quad \forall i \in K, j \in N. \quad (6e)$$

The decision variable  $x$  represents the allocation of customers' demands or items to vehicles or bins, with each entry  $x_{i,j}$  representing the assignment of the  $i$ -th vehicle or bin to the  $j$ -th demand or item.

We reformat equation (6b) as  $\sum_i^k q_j x_{ij} - Q k_i \leq 0$  to match the format required by a SCIP constraint. Using this integer program, we can determine the value of  $k_{\min}$ , which is the minimum between the optimal solution for (6a) and the strict lower bound obtained by  $\lceil \frac{\sum_i^n q_i}{Q} \rceil$ . Equation (6b) represents the bin capacity constraint, (6c) ensures that each item is assigned to a single bin, and (6d)-(6e) are the integrality constraints.

## mathematical solvers

A piece of software that implements several algorithms (either heuristics or optimality) to solve linear, quadratic, or nonlinear optimization problems. Examples include:

-  , SCIP[3] (the one we use as an evaluation environment);
-  GUROBI OPTIMIZATION, Gurobi;
-  , CPLEX;
-  LocalSolver, LocalSolver.

# Method

- We model each branching step, as described in Section , using a bipartite node with two disjoint sets. These sets are denoted as  $x = \{x_1, x_2, \dots, x_n\}$  and  $\delta = \{\delta_1, \delta_2, \dots, \delta_m\}$  (see Figure 4), representing the variables and constraints in (2b), respectively.
- The sensitivity of a variable to the objective function (either (5a) for the CVRP or (6a) for the BPP) is represented by an edge weight, or coefficient,  $a_{ij} \in A$ , where  $i$  belongs to Set  $x$  and  $j$  belongs to Set  $\delta$ . Each element in either set has a feature vector that describes its attributes during the solving process<sup>1</sup>.

## graph convolutional neural network

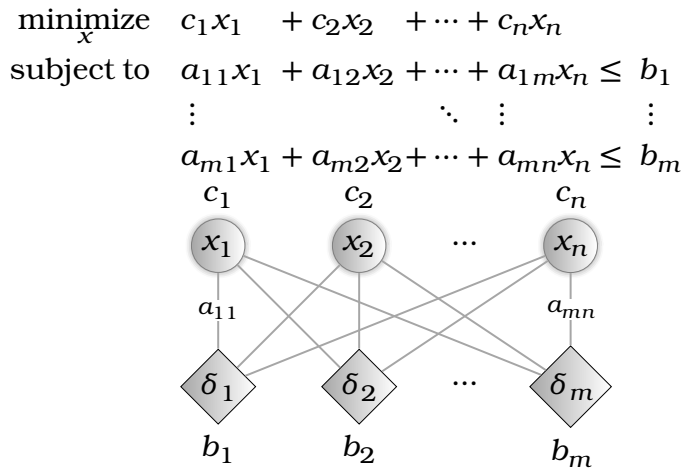


Figure 4: The corresponding integer program would then be minimize  $\sum c_i x_i$ ; subject to  $\sum a_{ij} x_j \leq b_i$

## GraphSAGE

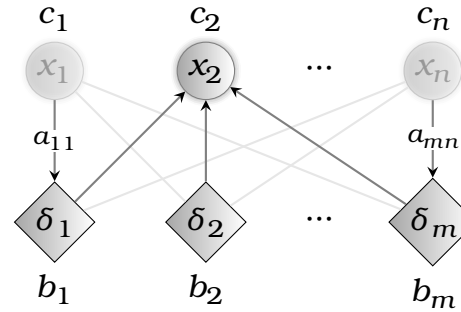


Figure 5: GraphSAGE accumulates information from a selection of neighboring nodes, with more distant nodes becoming increasingly influential in the aggregated information as the process continues.

## graph attention neural network

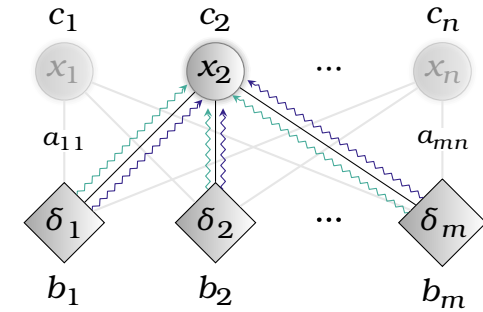


Figure 6: GAT assigns attention scores to the relationships between each node and its neighboring nodes. This is an example of a three-head attention mechanism.

<sup>1</sup>For example, the feature vector for a variable  $i \in x$  may include the objective value, variable type (such as binary, integer, or continuous), and lower and upper bounds. Similarly, the feature vector for a constraint  $j \in \delta$  may include the tightness, dual problem solution value (4a), scaled age, and bias. In total, there are 19 features collected for each variable and 5 for each constraint.



## Evaluation

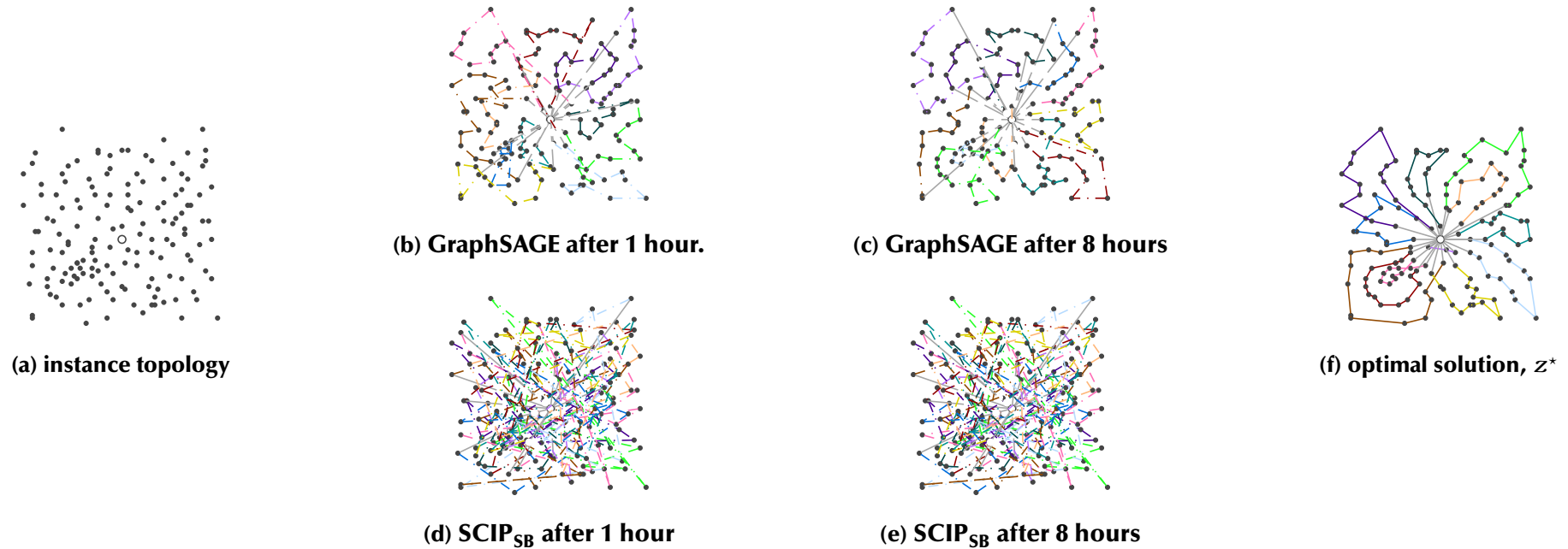


Figure 7: Example on solving the CVRP instance M-n151-k12 using both GraphSAGE and SCIP with allowed runtime of 8 hours.

### criteria

- The SB decision samples were drawn using École for the CVRP and BPP integer programs.
  - The CVRP training and evaluation instances were drawn from the 12 benchmark sets in the CVRPLIB [4], with six instances from sets A and P [5] for training and eight instances from sets P, B, and M for evaluation.
  - The BPP training instances were taken from [6], specifically sets U and T, with two instances for training (u100\_00 and u80\_00) and five instances for evaluation (t249\_00, t120\_00, t60\_00, u250\_00, and u500\_00)
- We evaluated five branching strategies: SB, RPC (which are the inherently built branching strategies), GCNN, GraphSAGE, and GAT (which are the learned strategies) using SCIP with the time limit set to 1 hour, 2 hours, 4 hours, or 8 hours.

**visualization**

<https://isotlaboratory.github.com/ml4vrp>

## results

The following figures represent 144 (CVRP) + 120 (BPP) experiments out of the 800 presented in the manuscript.

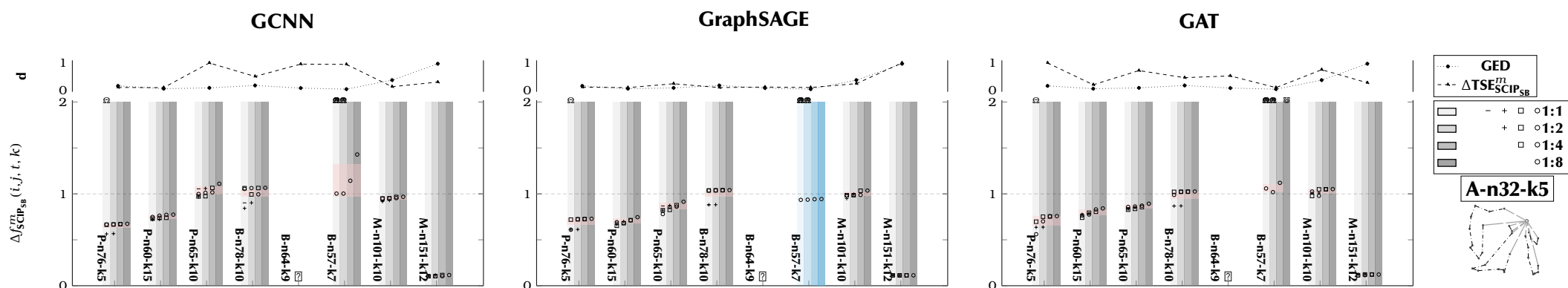


Figure 8: The performance of the three CVRP classifiers against  $SCIP_{SB}$  calculated by (??), across the training instances.

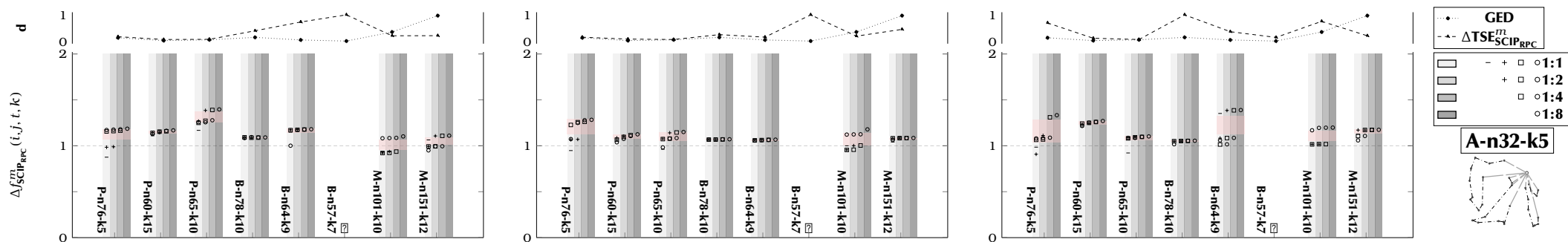


Figure 9: The performance of the three CVRP classifiers against  $SCIP_{RPC}$  calculated by (??), across the training instances.

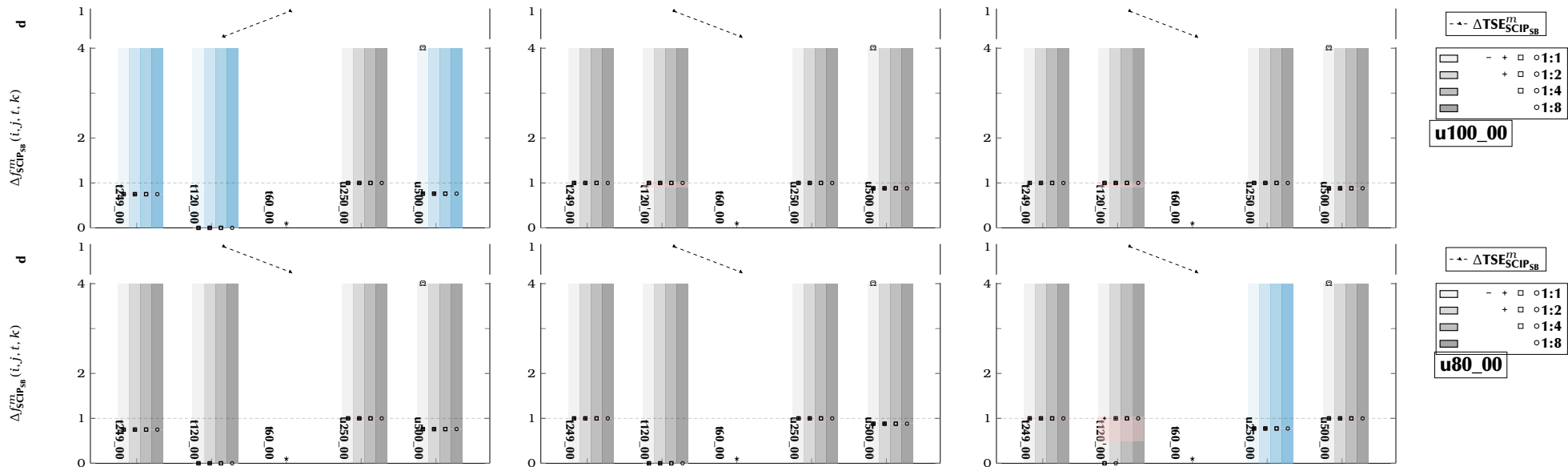


Figure 10: The performance of the three BPP classifiers against  $SCIP_{SB}$  calculated by (?), across the training instances.

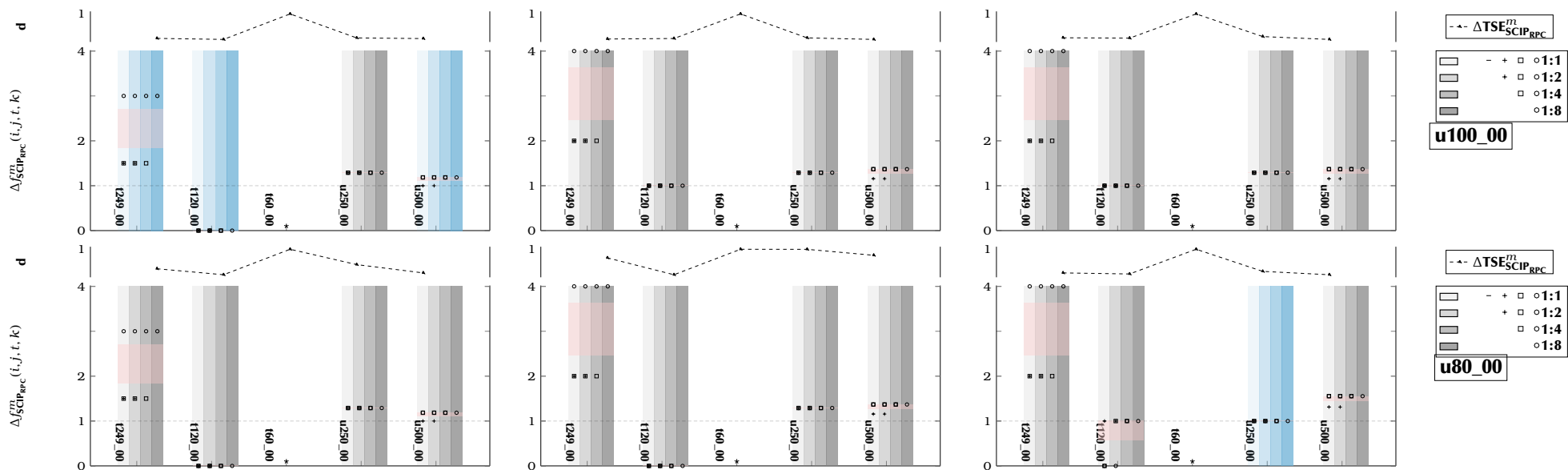


Figure 11: The performance of the three BPP classifiers against  $SCIP_{RPC}$  calculated by (?), across the training instances.

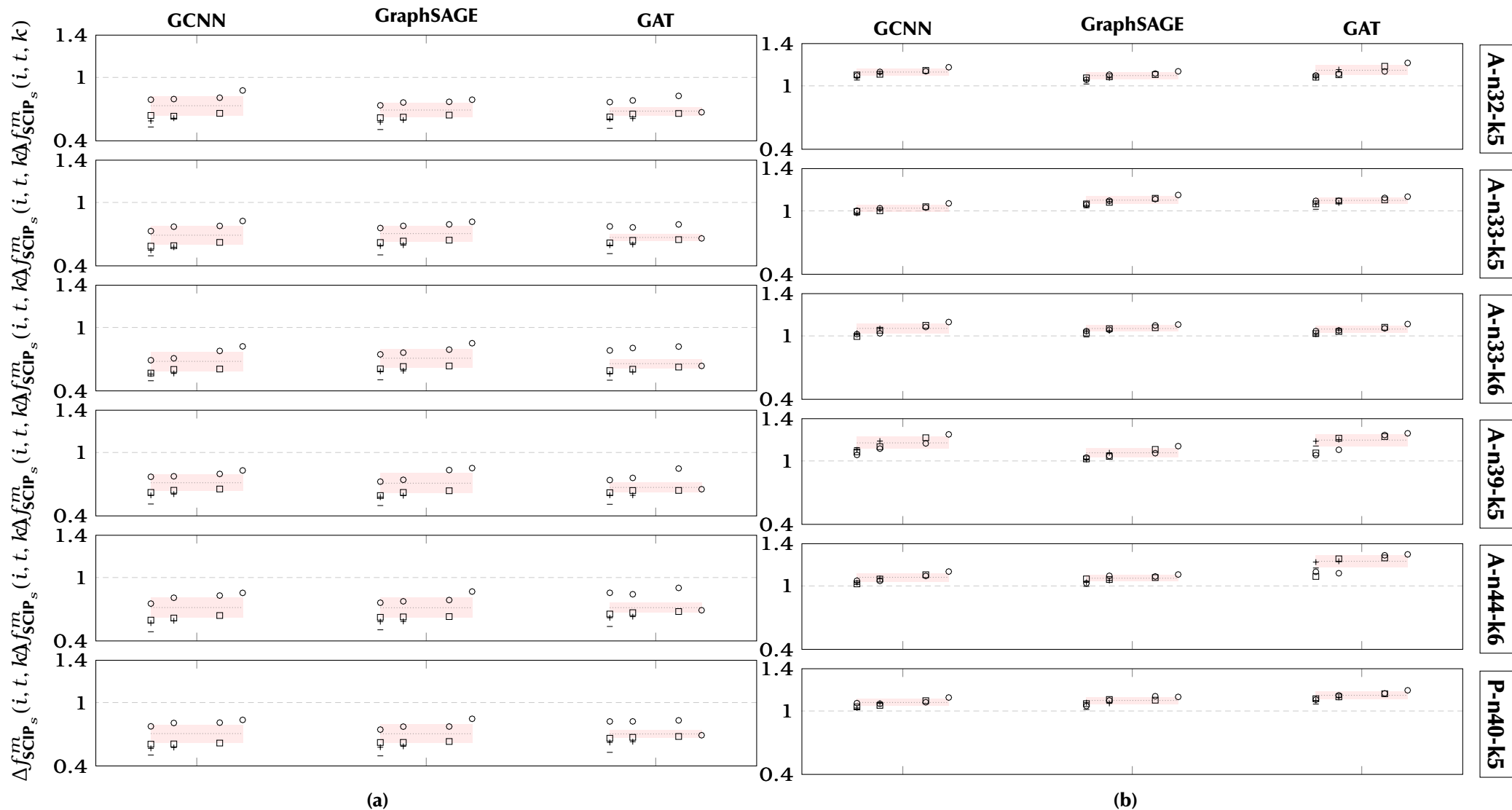


Figure 12: The average performance of each classifier across the CVRP evaluation instances; (a) compares against  $\text{SCIP}_{\text{SB}}$  and (b) compares against  $\text{SCIP}_{\text{RPC}}$ .

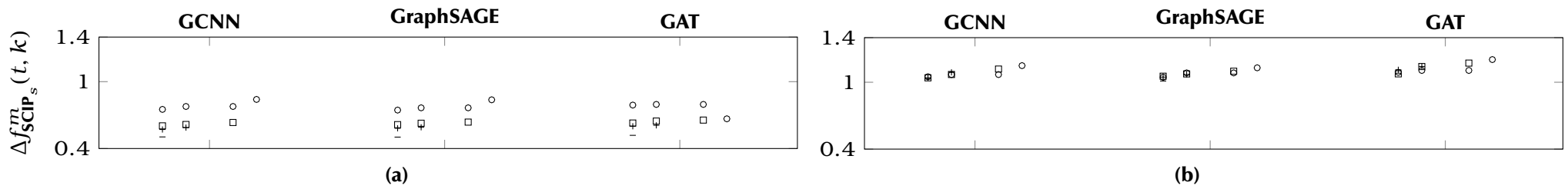


Figure 13: The performance of each classifier averaged by both of the CVRP training and evaluation instances against (a)  $\text{SCIP}_{\text{SB}}$  and (b)  $\text{SCIP}_{\text{RPC}}$ .

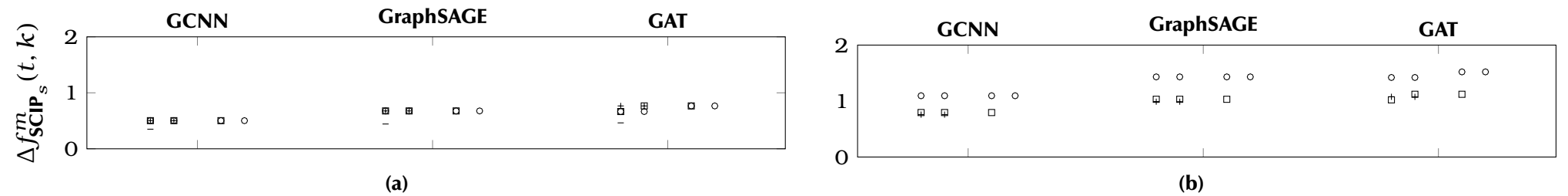


Figure 15: The performance of each classifier averaged by both of the BPP training and evaluation instances against (a)  $\text{SCIP}_{\text{SB}}$  and (b)  $\text{SCIP}_{\text{RPC}}$ .

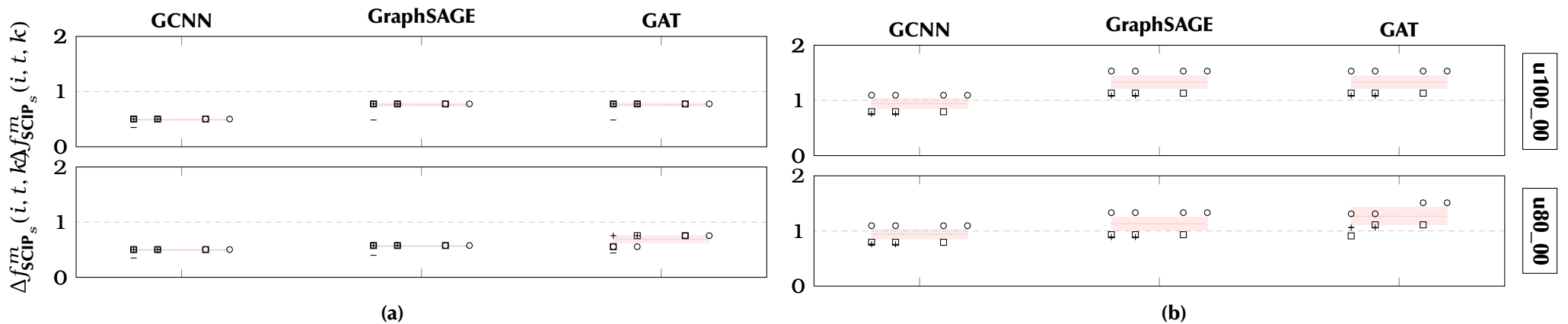



Figure 14: The average performance of each classifier across the BPP evaluation instances; (a) compares against  $\text{SCIP}_{\text{SB}}$  and (b) compares against  $\text{SCIP}_{\text{RPC}}$ .

## findings

- In Figure 8, which compares against the performance of  $SCIP_{SB}$ , the trained classifiers perform better than  $SCIP_{SB}$  in the majority of the experiments
- $SCIP_{SB}$  was unable to find a feasible solution within the first hour for Instance P-n76-k5 or within the first, second, and fourth hour for Instance B-n57-k7. This trend continues for Instance B-n64-k9 across all time windows.
- the three classifiers are able to find significantly lower gaps on Instance M-n151-k12 even when given the maximum time allowance to  $SCIP_{SB}$  (eight hours).
- the GED and  $\Delta TSE_{SCIP_{SB}}^m$  showed fluctuations with no apparent correlation to the classifiers performance, making it challenging to affirm generalization capabilities.
- When comparing the performance of the individual classifiers, GCNN reaches the lowest gap on four out of eight instances, although the surplus is modest when compared to GraphSAGE.
- GAT is able to outperform the other two classifiers at the eighth hour;
- Using RPC clearly ameliorate SCIP's performance. Nonetheless, at least one of the classifiers achieves equivalent performance or with slight improvement/decline in at least four hours less.
- The fourth bar  depicts the performance of the classifiers against a time allowance of eightfold increase for  $SCIP_S$ . The potential gains when using either branching strategies are minute, and in some cases, there may be a net loss,
- On average, the classifiers are consistently able to find lower gaps than  $SCIP_{SB}$  in significantly less time.

## Discussion

While the results were impressive, there are some challenges that may arise in larger instances with hundreds of thousands of customers. These include:

- difficulties in generating decision samples for the SB to train a custom classifier;
- limitations in the current approach's ability to generate more efficient strategies than those they were trained on.

We suggest two directions for these two problems:

- The first option is the UG [7], which is a parallelized implementation of SCIP, UG is a promising direction for enriching our sampling tools.
- The second method, which is our current focus, is formulating the CVRP as a zero-sum game [8, 9] between two reinforcement-learning agents, with the reward set to maximizing set negative relative primal-dual gap until a Nash equilibrium is approximated.



## References

- [1] S. P. Boyd, *Convex optimization*. Cambridge, UK: Cambridge University Press, 2004 - 2004.
- [2] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *J. ACM*, vol. 7, no. 4, p. 326–329, oct 1960. [Online]. Available: <https://doi.org/10.1145/321043.321046>
- [3] K. Bestuzheva, M. Besançon, W.-K. Chen, A. Chmiela, T. Donkiewicz, J. van Doornmalen, L. Eifler, O. Gaul, G. Gamrath, A. Gleixner, L. Gottwald, C. Graczyk, K. Halbig, A. Hoen, C. Hojny, R. van der Hulst, T. Koch, M. Lübbecke, S. J. Maher, F. Matter, E. Mühmer, B. Müller, M. E. Pfetsch, D. Rehfeldt, S. Schlein, F. Schlösser, F. Serrano, Y. Shinano, B. Sofranac, M. Turner, S. Vigerske, F. Wegscheider, P. Wellner, D. Weninger, and J. Witzig, "The SCIP Optimization Suite 8.0," Zuse Institute Berlin, ZIB-Report 21-41, December 2021. [Online]. Available: <http://nbn-resolving.de/urn:nbn:de:0297-zib-85309>
- [4] I. Lima, E. Uchoaa, D. Pecinb, A. Pessoaa, M. Poggib, T. Vidal, A. Subramanian, D. Oliveria, and E. Queiroga. (2014) Capacitated vehicle routing problem library. [Online]. Available: <http://vrp.galgos.inf.puc-rio.br/>
- [5] A. P. N. D, J. M. Belenguer, E. Benavent, A. Corberan, and G. Rinaldi, "Computational results with a branch and cut code for the capacitated vehicle routing problem," Sep 1995.
- [6] J. E. Beasley. (2004) Operation research library. [Online]. Available: <http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/>
- [7] Y. Shinano, T. Achterberg, T. Berthold, S. Heinz, and T. Koch, "Parascip: A parallel extension of scip," in *Competence in High Performance Computing 2010*, C. Bischof, H.-G. Hegering, W. E. Nagel, and G. Wittum, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 135–148.
- [8] J. von Neumann and O. Morgenstern, *ZERO-SUM GAMES : THEORY*. Princeton University Press, 1947, p. 85–165.
- [9] J. von Neumann, 1. A Certain Zero-sum Two-person Game Equivalent to the Optimal Assignment Problem. Princeton: Princeton University Press, 1953, pp. 5–12. [Online]. Available: <https://doi.org/10.1515/9781400881970-002>

**Thank you.**