Test-Time Search in Neural Graph Coarsening for the Capacitated Vehicle Routing Problem

Anonymous Author(s)
Affiliation
Address
email

Abstract

The identification of valid inequalities, such as Rounded capacity inequalities (RCIs), is a key component of cutting plane methods for the Capacitated Vehicle Routing Problem (CVRP). While a neural separation method can learn to find high-quality cuts, improving the learned model further often requires costly retraining with diminishing returns. This paper proposes an alternative: enhancing the performance of a trained model at inference time through two test-time search techniques. First, we introduce stochastic edge selection into the graph coarsening procedure, replacing the previously proposed greedy approach. Second, we propose the Graph Coarsening History-based Partitioning (GraphCHiP) algorithm, which leverages coarsening history to identify not only RCIs but also, for the first time, the Framed capacity inequalities (FCIs). Experiments on randomly generated CVRP instances demonstrate the effectiveness of our approach in reducing the dual gap compared to the existing neural separation method. Additionally, our method discovers effective FCIs on a specific instance, despite the challenging nature of identifying such cuts.

1 Introduction

The Capacitated Vehicle Routing Problem (CVRP) is a fundamental combinatorial optimization problem in logistics and operations research. The goal of the CVRP is to design a set of minimum-cost routes for a fleet of vehicles, each with a limited capacity, to serve a group of customers with specific demands. Since this problem is $\mathcal{NP}$-hard, finding optimal solutions for large-scale instances poses a significant computational challenge. To find optimal solutions, exact methods such as branch-and-cut [13, 11] or branch-and-price-and-cut algorithms [14, 4, 19] are commonly employed. These methods rely on the *cutting plane method*, which iteratively refines the linear programming relaxation of the CVRP by adding valid inequalities, or *cuts*, to eliminate infeasible solutions. The task of identifying violated inequalities for a given fractional solution is known as the separation problem.

For the CVRP, one of the most important families of cuts is the Rounded capacity inequalities (RCIs), which leverage vehicle capacity constraints. However, the separation of effective RCIs is not trivial. Traditionally, this is addressed by either exact algorithms [6], which guarantee finding the most violated cuts but are relatively slow, or heuristic methods [11], which are faster but tend to produce cuts with relatively minimal violation. To bridge this gap, Kim et al. [9] recently introduced NeuralSEP, a neural separation algorithm designed for generating RCIs. By learning to identify promising customer subsets for cut generation via a *neural graph coarsening* procedure, NeuralSEP achieves good performance on large-scale problems.

While NeuralSEP shows promising results, achieving further performance gains is often subject to diminishing returns, as strategies like architectural modifications or hyperparameter tuning require costly retraining for only marginal gains. We turn to *test-time search*, a technique that improves

Submitted to 39th Workshop on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

model performance at inference time without any retraining. This approach has been widely adopted in neural combinatorial optimization (NCO). For example, common techniques include random sampling and beam search, which focus on generating multiple candidate solutions [2, 10, 3]. Other strategies adapt the model itself, such as active search, which refines model parameters for a specific test instance [2, 7]. More recently, methods inspired by evolutionary algorithms, such as neural genetic search [8], have also been proposed. Building on this established paradigm, our work introduces novel test-time search methods specifically designed to enhance the cut generation capabilities of NeuralSEP. In this work, we adapt this paradigm to introduce novel test-time search methods specifically designed to enhance the cut generation capabilities of NeuralSEP.

We propose two methods that improve its performance on finding existing cuts and extend its capabilities to a more complex class of inequalities for the first time. Our first method introduces a stochastic edge selection strategy into NeuralSEP's neural graph coarsening procedure, enhancing the generation of RCIs and leading to improved dual gap reduction. Our second, the novel GraphCHiP algorithm, leverages the model's coarsening history to identify additional RCIs and to enable the training-free generation of Framed capacity inequalities (FCIs). Experimental results demonstrate that these methods collectively enhance the performance of NeuralSEP in terms of dual gap reduction. Additionally, we showcase the ability of GraphCHiP to identify effective FCIs on a specific instance, despite the challenging nature of finding such cuts.

2 Preliminaries and background

The Capacitated Vehicle Routing Problem (CVRP) is typically defined on a complete graph $G = (V, E)$, where $V = \{v_0\} \cup V_C$ is the set of vertices, consisting of a depot (v_0) and a set of customers (V_C). Each edge $e \in E$ is associated with a binary decision variable x_e, where $x_e = 1$ if the edge is used in the solution and $x_e = 0$ otherwise. For any subset of customers $S \subseteq V_C$, let $\delta(S)$ denote the set of edges with exactly one endpoint in S (the cut-set). The term $x(\delta(S))$ represents the sum of decision variables for all edges in this cut-set, i.e., $x(\delta(S)) = \sum_{e \in \delta(S)} x_e$. Each customer $i \in V_C$ has a demand q_i, and each vehicle has a uniform capacity Q.

2.1 Capacity inequalities

Rounded capacity inequalities (RCIs) are the most widely used capacity inequalities in the CVRP. The form of RCIs is given by:

$$x(\delta(S)) \geq 2 \left\lceil \frac{\sum_{i \in S} q_i}{Q} \right\rceil \qquad \forall S \subseteq V_C, |S| \geq 2. \tag{1}$$

This inequality ensures that the number of vehicles entering and leaving any subset of customers S is sufficient to meet the total demand of that subset, given the vehicle capacity Q. The exact separation problem for RCIs is $\mathcal{NP}$-hard [5], and solving it for large-scale problems is computationally expensive. Alternatives to the exact separation algorithm range from heuristic methods, exemplified by the well-known CVRPSEP library Lysgaard et al. [11], to the learning-based algorithm NeuralSEP, which is trained using optimal solutions to the exact separation problem as labels.

RCIs are effective, but they do not consider demands outside the set S [13]. To address this issue, Augerat et al. [1] introduced the framed capacity inequalities (FCIs). FCIs extend RCIs by defining a structure composed of a larger customer subset, which contains a set of smaller, mutually disjoint components. The total demand of each component is then treated as an individual item in a bin-packing problem. By solving this problem, the minimum number of vehicles needed to serve all the components is determined collectively. This approach allows FCIs to account for capacity constraints across multiple subsets simultaneously, resulting in tighter bounds and the generation of stronger inequalities compared to RCIs. The inequality is formulated as:

$$x(\delta(H)) + \sum_{i \in I} x(\delta(S_i)) \geq 2r(\Omega) + 2 \sum_{i \in I} \left\lceil \frac{d(S_i)}{Q} \right\rceil. \tag{2}$$

The identification of these FCIs is computationally expensive, since it requires solving the bin-packing problem, which is known to be $\mathcal{NP}$-hard. The separation of FCIs is addressed in a few studies [1, 11]. As far as we know, our work is the first to propose a learning-based approach for separating FCIs.

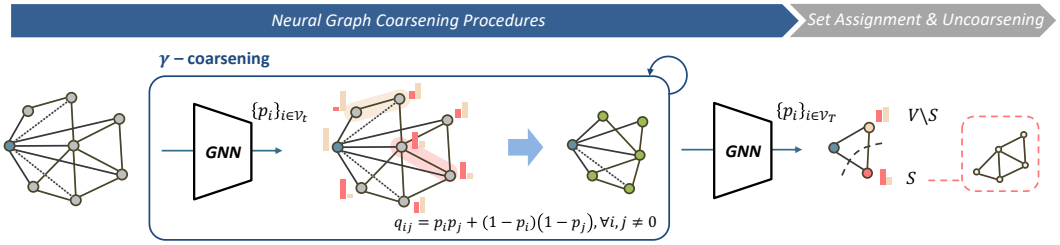


Figure 1: The details of NeuralSEP framework. p_i is the predicted probability that vertex $i \in V_C$ is included in the subset S. q_{ij} is the contraction probability that vertices $i \in V_C$ and $j \in V_C$ are contracted into a single vertex.

2.2 Revisiting NeuralSEP

NeuralSEP, proposed by Kim et al. [9], is a neural separation algorithm designed to generate RCIs for the CVRP. In essence, NeuralSEP is a learned function that maps an input support graph to an output customer subset, S, which defines a violated inequality. The parameters of this function, embodied by a neural network, are trained in a supervised manner using labels derived from optimal solutions to the exact separation problem for instances with 50 to 100 customers. The overall framework of NeuralSEP consists of four parts, which are graph embedding with graph neural networks (GNNs), message passing GNNs, neural graph coarsening, and set assignment and graph uncoarsening. As illustrated in Figure 1, a key part is the *neural graph coarsening* procedure, inspired by the shrinking heuristic of Ralphs et al. [16]. It works by iteratively merging vertices that the model predicts are likely to be on the same side of the cut—that is, either both inside the violated subset S or both outside of it. Experimental results show that NeuralSEP efficiently generates high-quality cuts and outperforms CVRPSEP on large-scale instances with over 400 customers under fixed iterations. Further evaluation on benchmark datasets demonstrates scalability and effective generalization to out-of-distribution problems.

3 Test-time search in neural graph coarsening

3.1 Stochastic edge selection in graph coarsening

The original NeuralSEP employs a greedy strategy in its neural graph coarsening procedure, where edges with the highest contraction probabilities are selected for merging. This deterministic approach, however, can lead to a lack of diversity in the generated cuts, as the same edges are consistently chosen for contraction across different runs. One effective approach to addressing the challenge is to introduce a stochastic element into the procedure. We modify the contraction probability by introducing a stochastic parameter π_{ij}, resulting in a perturbed contraction probability $\tilde{q}_{ij}$:

$$\tilde{q}_{ij} = \begin{cases} p_i p_j + (1-p_i)(1-p_j) + \pi_{ij} & \text{if } i,j \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

where π_{ij} is a small random value drawn from a uniform distribution $\mathcal{U}(0, 0.001)$. During the coarsening procedure, an edge e whose perturbed probability $\tilde{q}_{ij}$ is the highest among the edges in the support graph $\bar{G}$ is selected for contraction, i.e., $e = \arg\max_{(i,j)\in\bar{E}} \tilde{q}_{ij}$. This approach is termed *π-greedy selection*, and Figure 2 illustrates this process. Adding randomness can alter which edges are chosen for contraction and create chain effects throughout the procedure. Since the coarsening procedure iterates and generates predictions several times, early changes in contraction order spread through the entire process. As a result, the algorithm produces different cuts despite starting from the same initial conditions.

3.2 Graph coarsening history-based partitioning algorithm

The Graph Coarsening History-based Partitioning (GraphCHiP) algorithm is a novel test-time search method designed to identify both RCIs and FCIs using the trained NeuralSEP model. The key idea of GraphCHiP is to leverage the intermediate steps of the neural graph coarsening procedure from NeuralSEP to identify promising candidate *subsets* and *partitions* for generating violated inequalities.

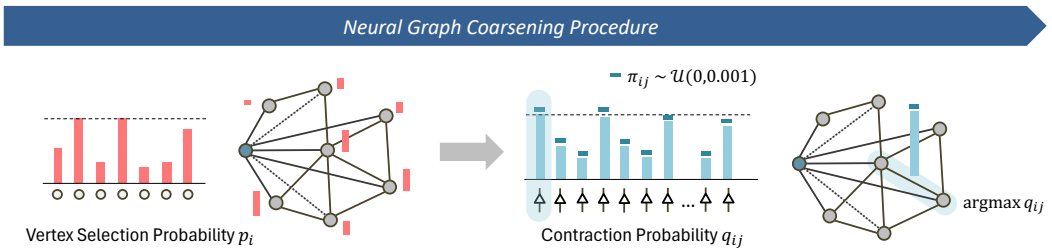


Figure 2: Illustration of π-greedy selection method in the *neural graph coarsening* procedure

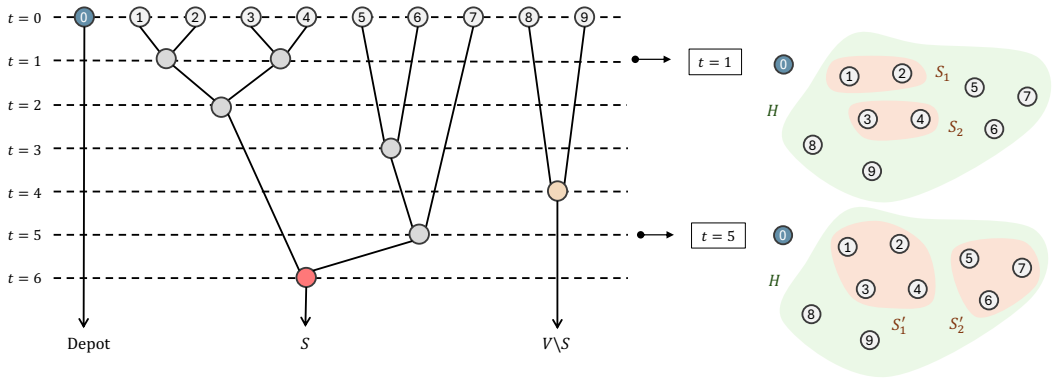


Figure 3: Illustration of how the GraphCHiP algorithm makes the subsets and partitions from the node map. The left diagram shows the neural graph coarsening process from $t = 0$ (original graph with vertices 0-9) to $t = 6$ (final coarsened graph with three supernodes: Depot, S, and $V \setminus S$). The right diagrams demonstrate partition generation at steps $t = 1$ and $t = 5$ by backtracking the coarsening history. At $t = 5$, merged vertex groups (e.g., $S'_1 = \{1, 2, 3, 4\}$, $S'_2 = \{5, 6, 7\}$) and individual vertices form partition Ω of subset H.

In neural graph coarsening, vertices are iteratively merged based on their contraction probabilities. This process is controlled by the coarsening ratio γ, which dictates the fraction of vertices to be coarsened at each step. Thus, for a given support graph $\bar{G} = (V, \bar{E})$, the coarsening process generates a sequence of graphs $\{\mathcal{G}_t\}_{t=0}^{T}$, where V_t is the set of vertices (supernodes) at coarsening step $t \in [0, T]$. At each step t, vertices in V_{t-1} are merged into disjoint supernodes to form the next step's vertex set, V_t. The process is captured by a collection of node maps $\{\mathcal{M}_t\}_{t=1}^{T}$, where $\mathcal{M}_t : V_t \to 2^V$ provides, for any supernode $u \in V_t$, the complete set of initial vertices from V. Thus, these node maps provide a complete record of how the original vertices are hierarchically clustered. Figure 3 illustrates this core mechanism. The left side shows a complete coarsening process, while the right side demonstrates how the node map can be used to backtrack and reconstruct the specific vertex groupings at any intermediate step. This ability to query the historical structure of the graph is the foundation for our cut separation procedures. The supernodes formed at each step serve as candidate subsets for RCIs, and collections of these supernodes can be organized into candidate partitions for FCIs.

For RCI separation Each supernode generated during the coarsening process is a natural candidate subset for RCIs. The procedure is straightforward: GraphCHiP iterates backward through the node map, from step $t = T - 1$ to $t = 1$. At each step t, it considers every supernode $u \in V_t$ that is also present in the subset S identified by NeuralSEP. Using the node map $\mathcal{M}_t$, it identifies the subsets which consists of the original vertices constituting the supernode u. It then directly checks if these subsets violate the RCI. For efficiency, this search terminates as soon as any subset that violates the RCI is found. By examining the entire coarsening history, this method can find a violating subset even when the final subset S itself is not violated.

For FCI separation While NeuralSEP is originally designed to identify RCIs, GraphCHiP extends its capability to find FCIs at test time. One of the main challenges in separating FCIs is to find

promising partitions of the vertex set, since searching all possible partitions is computationally intractable. Our algorithm leverages the coarsening history to generate candidate partitions. In detail, after NeuralSEP identifies a final subset S for a potential RCI, we trace its formation history backward. At each coarsening step t, we construct a candidate partition Ω of the customer set V_C by three steps. First, for each supernode $u \in V_t$ in the set S, we use the node map $\mathcal{M}_t(u)$ to identify its constituent set of original vertices. Second, we determine the set of all customer vertices in V_C that are not included in any of the sets resolved in the previous step. Third, we complete the partition by adding each unassigned vertex as a distinct singleton set. This process guarantees that Ω is a complete partition of all customer vertices in V_C. We then apply a filtering heuristic that discards any partition Ω if one of its subsets in Ω already violates an RCI. This makes the search focus on cuts arising from the partition's structure. For the remaining partitions, we employ a two-stage evaluation to reduce computational cost. First, we use a fast, approximate calculation of the bin-packing value $r(\Omega)$ to quickly screen out unpromising candidates. For partitions that pass this check, we then compute the tight dual bound of $r(\Omega)$ using the algorithm in Martello and Toth [12].

4 Experimental results

In this section, we present a comprehensive evaluation of our proposed test-time search methods for NeuralSEP. Our evaluation focuses on the performance of a cutting plane method at the root node of a branch-and-cut algorithm. Our approach is used as the primary engine for generating cuts.

The experiments are designed to address two primary objectives. First, we evaluate our *stochastic edge selection method* by measuring its impact on the dual gap. Second, we assess *GraphCHiP algorithm*'s effectiveness in leveraging the coarsening history to find additional RCIs, measured by improvements in the dual gap, and its novel capability to identify FCIs. Regarding the FCI evaluation, it is important to note that violated inequalities are not present in every problem instance, as their existence is highly contingent on the specific support graph structure and customer demands. Accordingly, our analysis of FCIs focuses on a detailed examination of a specific instance where our algorithm successfully identified them.

Our evaluation methodology follows the approach from Kim et al. [9]. We measure performance using the optimality gap defined as $GAP = \frac{UB-LB}{UB} \times 100(\%)$, where UB is the upper bound provided by the hybrid genetic search (HGS) algorithm from Vidal [18] and LB is the lower bound obtained from the cutting plane method at the root node of a branch-and-cut algorithm. We also use the same set of test instances, which are publicly available at `https://github.com/hyeonahkimm/neuralsep/tree/main/data/instances`. These randomly generated CVRP instances were created following the guidelines of Uchoa et al. [17] and Queiroga et al. [15]. The test set includes instances with a varying number of customers, $|V_C| \in \{50, 75, 100, 200, 300, 400, 500, 750, 1000\}$. For a fair comparison, we run all experiments under the same conditions. The experiments are conducted on a single machine with 62 GB DDR4 RAM @ 3200 MT/s, an AMD Ryzen 9 5900X CPU, and an NVIDIA GeForce RTX 4070 GPU.

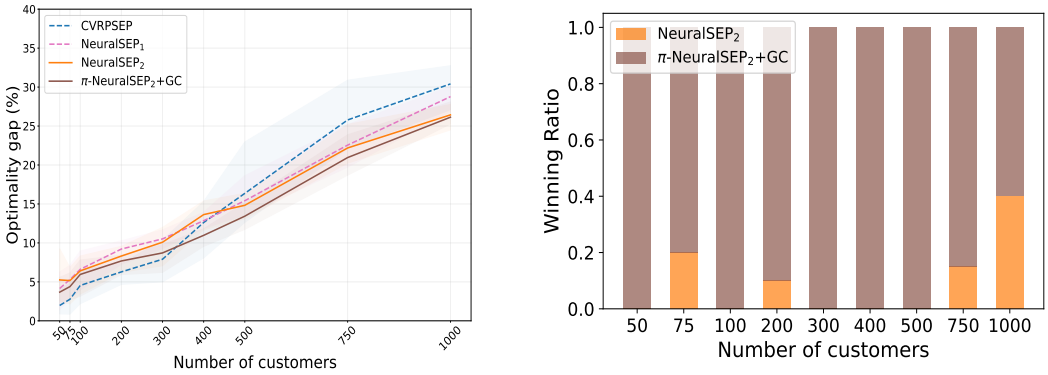
4.1 Results on RCI separation

For our comparative analysis, we use several benchmarks: CVRPSEP, a library of traditional separation heuristics, serves as a non-learning-based benchmark, while the original NeuralSEP$_1$ acts as our primary learning-based benchmark implemented by Deep Graph Library (DGL). We also developed NeuralSEP$_2$, our re-implementation of NeuralSEP in PyTorch Geometric (PyG), to provide a controlled baseline for our enhancements. Our proposed methods are π-NeuralSEP$_2$, which integrates a stochastic edge selection algorithm, and our full approach, π-NeuralSEP$_2$ + GC, which further incorporates the GraphCHiP algorithm. This setup allows us to measure the performance gains from each of our algorithmic contributions.

We evaluate the performance of the proposed RCI separation methods on the benchmark instances, imposing a uniform computational time limit of 3600 seconds for each run. The performance of the RCI separation algorithms on the CVRP test set is summarized in Table 1 and visually represented in Figure 4. The combination of the more efficient PyG implementation and our test-time search methods allows our approach to improve the dual gap within the given time limit.

Table 1: Summary of the performance of RCI separation algorithms

Size	CVRPSEP		NeuralSEP_1		NeuralSEP_2		π-NeuralSEP_2 + GC	
	Gap	Time/Iter	Gap	Time/Iter	Gap	Time/Iter	Gap	Time/Iter
50	**1.970%**	0.009	4.151%	0.830	5.250%	0.120	3.679%	0.133
75	**2.769%**	0.054	5.305%	1.066	5.164%	0.209	4.393%	0.246
100	**4.539%**	0.145	6.611%	1.440	6.410%	0.378	5.953%	0.394
200	**6.280%**	2.001	9.214%	3.411	8.314%	1.293	7.683%	1.594
300	**7.903%**	10.431	10.515%	12.006	10.087%	4.607	8.714%	7.482
400	12.618%	16.936	12.848%	26.714	13.632%	13.518	**10.970%**	19.850
500	16.357%	16.947	15.413%	41.227	14.826%	26.705	**13.429%**	39.125
750	25.783%	16.603	22.553%	102.623	22.187%	90.436	**20.956%**	111.835
1,000	30.408%	23.321	28.777%	161.183	26.434%	139.826	**26.136%**	159.042



(a) Comparison of optimality gap

(b) The winning ratio out of 10 instances

Figure 4: Comparison of the performance of RCI separation algorithms

4.2 Experimental results of FCI separation

We now present the results of the experiments on FCI separation using our proposed method, GraphCHiP. Our separation routine is designed to add FCI cuts only when the violation found during RCI separation is less than 1.0. As this condition typically occurs in the later stages of the iterative process, the FCI experiments are conducted without a time limit or other early stopping criteria.

Given the known rarity of FCIs, we present the results on the `X-n153-k22` instance from Uchoa et al. [17]. The purpose is to demonstrate and validate the capability of our algorithm in the challenging task of identifying FCIs. Table 2 summarizes the detailed results on the instance. Combining GraphCHiP with π-NeuralSEP_2 yields the best overall performance. This combination improves the optimality gap by an additional 1.38%p compared to using π-NeuralSEP_2 alone.

Table 2: Results on `X-n153-k22` instance

Method	Algorithm	Lowerbound	Gap	FCI cuts
RCI	CVRPSEP RCI	19983.51	5.83%	-
	π-NeuralSEP_2 RCI	19861.37	6.40%	-
RCI+FCI	CVRPSEP RCI + CVRPSEP FCI	19984.29	5.82%	4
	π-NeuralSEP_2 RCI + GraphCHiP FCI	**20153.95**	**5.02%**	324

Optimal Value (opt): 21220.0

References

[1] Augerat, P., Naddef, D., Belenguer, J., Benavent, E., Corberan, A., and Rinaldi, G. (1995). Computational results with a branch and cut code for the capacitated vehicle routing problem.

[2] Bello, I., Pham, H., Le, Q. V., Norouzi, M., and Bengio, S. (2016). Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*.

[3] Choo, J., Kwon, Y.-D., Kim, J., Jae, J., Hottung, A., Tierney, K., and Gwon, Y. (2022). Simulation-guided beam search for neural combinatorial optimization. *Advances in Neural Information Processing Systems*, 35:8760–8772.

[4] Costa, L., Contardo, C., and Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985.

[5] Diarrassouba, I. (2017). On the complexity of the separation problem for rounded capacity inequalities. *Discrete Optimization*, 25:86–104.

[6] Fukasawa, R., Longo, H., Lysgaard, J., Aragão, M. P. d., Reis, M., Uchoa, E., and Werneck, R. F. (2006). Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming*, 106:491–511.

[7] Hottung, A., Kwon, Y.-D., and Tierney, K. (2021). Efficient active search for combinatorial optimization problems. *arXiv preprint arXiv:2106.05126*.

[8] Kim, H., Choi, S., Son, J., Park, J., and Kwon, C. (2025). Neural genetic search in discrete spaces. *arXiv preprint arXiv:2502.10433*.

[9] Kim, H., Park, J., and Kwon, C. (2024). A neural separation algorithm for the rounded capacity inequalities. *INFORMS Journal on Computing*.

[10] Kool, W., Van Hoof, H., and Welling, M. (2018). Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*.

[11] Lysgaard, J., Letchford, A. N., and Eglese, R. W. (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical programming*, 100:423–445.

[12] Martello, S. and Toth, P. (1990). *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc.

[13] Naddef, D. and Rinaldi, G. (2002). Branch-and-cut algorithms for the capacitated vrp. In *The vehicle routing problem*, pages 53–84. SIAM.

[14] Pecin, D., Pessoa, A., Poggi, M., and Uchoa, E. (2017). Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, 9(1):61–100.

[15] Queiroga, E., Sadykov, R., Uchoa, E., and Vidal, T. (2021). 10,000 optimal cvrp solutions for testing machine learning based heuristics. In *AAAI-22 workshop on machine learning for operations research (ML4OR)*.

[16] Ralphs, T. K., Kopman, L., Pulleyblank, W. R., and Trotter, L. E. (2003). On the capacitated vehicle routing problem. *Mathematical programming*, 94:343–359.

[17] Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., and Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858.

[18] Vidal, T. (2022). Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. *Computers & Operations Research*, 140:105643.

[19] You, Z. and Yang, Y. (2025). RouteOpt: An open-source modular exact solver for vehicle routing problems. *Available at SSRN 5314242*.

A Pseudo Code for GraphCHiP

We present two algorithms that leverage NeuralSEP's coarsening history to find new cuts. Algorithm 1 details our procedure for identifying additional RCIs, while Algorithm 2 introduces our GraphCHiP method for separating FCIs.

Algorithm 1 GraphCHiP for RCI

Input: Support Graph $\mathcal{G}$,
Subset $S \subseteq V_C$,
A node map $\{\mathcal{M}_t\}_{t=1}^{T}$
Output: The collection $\mathcal{R}$ of the subset h and RHS of RCI

$\mathcal{R} \leftarrow \emptyset, t \leftarrow T-1$ ▷ Initialize. Iterate backwards from the coarsest graph
while $t > 0$ **do**
  for each $u \in V_t \cap S$ **do**
    $h \leftarrow \mathcal{M}_t(u)$ ▷ Get initial nodes using the map for the current level t
    LHS $\leftarrow$ Calculate_LHS$(\mathcal{G}, h)$
    RHS $\leftarrow 2\left\lceil \frac{d(h)}{Q} \right\rceil$
    if RHS $-$ LHS > 0 **then** ▷ Check if the violation exists
      Add (h, RHS) to $\mathcal{R}$
    end if
  end for
  if $\mathcal{R} \neq \emptyset$ **then**
    return $\mathcal{R}$
  end if
  $t \leftarrow t-1$ ▷ Backtrack to the previous coarsening step
end while
return $\mathcal{R}$

Table 3: Comparison of the performance with and without π-greedy selection

Size	**NeuralSEP$_2$**	**π-NeuralSEP$_2$**	**Difference**		**Winning Ratio**
	RCI Cuts	RCI Cuts	% Cut Increase	Δ Gap	π-NeuralSEP$_2$ Win
50	130.9	165.2	26.2%	↓1.337%p	1.0
75	395.0	391.5	-0.9%	↓0.075%p	0.4
100	724.3	731.1	0.9%	↓0.117%p	0.5
200	1425.4	1481.1	3.9%	↓0.193%p	0.7
300	2336.1	3188.8	36.5%	↓1.045%p	0.9
400	2362.1	3363.8	42.4%	↓2.396%p	1.0
500	2163.9	2612.0	20.7%	↓1.307%p	1.0
750	1649.4	1735.8	5.2%	↓1.095%p	0.8
1000	1295.2	1332.7	2.9%	↓0.239%p	0.5

B Ablation Study on Test-time Search for RCIs

B.1 Results on π-greedy selection

To investigate the impact of the π-greedy selection method, we analyze its effectiveness through direct comparison. Table 3 presents a detailed comparison between the original NeuralSEP and π-NeuralSEP under the same PyG implementation. We evaluate the π-greedy selection method by examining the average total number of cuts generated, differences in optimality gaps, number of RCIs. The results demonstrate reduced average optimality gaps across all instance sizes. Particularly noteworthy improvements are observed in medium to large instances ranging from 300 to 750 customers. The number of RCI cuts increases for all sizes except 75.

B.2 Results on GraphCHiP for RCIs

To assess the effectiveness of the GraphCHiP algorithm for RCI separation, we conduct a comparative analysis between our full approach (π-NeuralSEP$_2$ + GC) and the version without GraphCHiP (π-NeuralSEP$_2$). As summarized in Table 4, adding GraphCHiP to our π-NeuralSEP$_2$ approach consistently identifies more RCI cuts and leads to a reduction in the average dual gap across all problem sizes. This enhancement is particularly effective for small to medium-scale instances with 75 to 400 customers. We hypothesize that the smaller effect observed on instances with 500 or more

Algorithm 2 GraphCHiP for FCI

Input: Support Graph $\mathcal{G}$,
Subset $S \subseteq V_C$,
A node map $\{\mathcal{M}_t\}_{t=1}^{T}$
Output: The collection $\mathcal{F}$ of the partition Ω of V_C and RHS of FCI

Initialize $\mathcal{F} \leftarrow \emptyset$, $t \leftarrow T-1$ ▷ Iterate backwards from the coarsest graph
while $t > 0$ **do**
  $\Omega \leftarrow \emptyset$ ▷ Initialize the partition of V_c
  for each $u \in V_t \cap S$ **do**
    $h \leftarrow \mathcal{M}_t(u)$ ▷ Get initial nodes using the map for the current level t
    Add h to Ω ▷ Add the subset to the partition
  end for
  for each $v \in V_C \setminus \bigcup \Omega$ **do**
    $h \leftarrow \{v\}$
    Add h to Ω ▷ Add each remaining individual node as a subset
  end for
  if $\exists h \in \Omega$ such that $2\lceil d(h)/Q \rceil >$ Calculate_LHS$(\mathcal{G}, h)$ **then**
    $t \leftarrow t-1$ ▷ Filter out subsets which violate RCI
    continue ▷ Proceed to the next iteration of the outer loop
  end if
  LHS $\leftarrow$ Calculate_LHS$(\mathcal{G}, \Omega)$
  RHS $\leftarrow 2\left\lceil \frac{d(V_C)}{Q} \right\rceil + 2\sum_{h\in\Omega} \left\lceil \frac{d(h)}{Q} \right\rceil$
  if RHS $-$ LHS > -2 **then** ▷ Check tight RHS for the potential cuts
    RHS $\leftarrow 2r(\Omega) + 2\sum_{h\in\Omega} \left\lceil \frac{d(h)}{Q} \right\rceil$ ▷ Update RHS by calculating $r(\Omega)$
    if RHS $-$ LHS > 0 **then** ▷ Check if the violation exists
      Add (Ω, RHS) to $\mathcal{F}$
    end if
  end if
  $t \leftarrow t-1$ ▷ Backtrack to the previous coarsening step
end while
return $\mathcal{F}$

Table 4: Comparison of the performance with GraphCHiP algorithm for RCIs

Size	**π-NeuralSEP$_2$**	**+ GC**	**Difference**		**Winning Ratio**
	RCI Cuts	RCI Cuts	% Cut Increase	Δ Gap	π-NeuralSEP$_2$+GC Win
50	165.2	185.8	12.5%	↓0.234%p	0.5
75	391.5	465.1	18.8%	↓0.696%p	0.9
100	731.1	806.0	10.2%	↓0.340%p	0.9
200	1481.1	1734.8	17.1%	↓0.438%p	1.0
300	3188.8	3503.6	9.9%	↓0.328%p	0.9
400	3363.8	3669.1	9.1%	↓0.266%p	0.9
500	2612.0	2751.8	5.4%	↓0.090%p	0.7
750	1735.8	1775.4	2.3%	↓0.136%p	0.7
1000	1332.7	1352.2	1.5%	↓0.059%p	0.5

customers may be partly due to the 3600-second time limit being insufficient for the full benefits of the search to materialize.

C Detailed Results of `X-n153-k22` Instance

We present the detailed analysis of the `X-n153-k22` instance, which shows the capability of our GraphCHiP algorithm to identify a violated FCI. The `X-n153-k22` instance is characterized by a skewed demand distribution, with a large number of low-demand customers and a smaller group of high-demand customers. Figure 5 illustrates the specific FCI found by GraphCHiP, with a violation of 0.31. The partition Ω consists of two non-singleton subsets, S_1 and S_2, along with several singleton

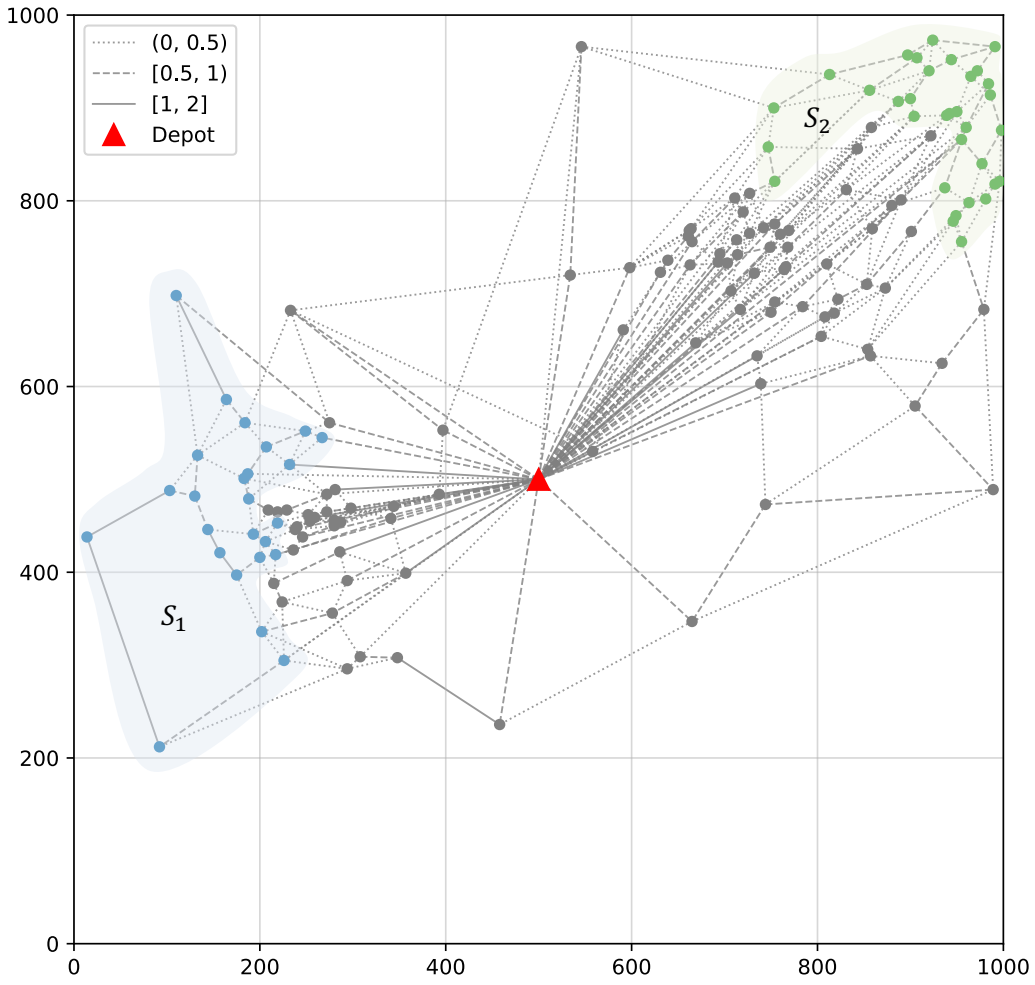


Figure 5: An example of a FCI found by GraphCHiP on `X-n153-k22` instance. The blue and green nodes represent the two non-singleton subsets in the partition Ω, respectively, while the remaining nodes are singleton subsets. The sum of demands for customers in S_1 and S_2 is 602 and 662, respectively. (total customer demand: 3068, vehicle capacity: 144). The set H for the partition is the entire set of customers V_C, and the corresponding cut values are $\bar{x}(\delta(H)) = 44.0$, $\bar{x}(\delta(S_1)) = 10.5$, $\bar{x}(\delta(S_2)) = 11.19$. The bin-packing value for the partition is $r(\Omega) = 23$, larger than $\lceil \frac{d(V_C)}{Q} \rceil = 22$.

subsets. We then verify the violation of the corresponding FCI. The left-hand side (LHS) of the inequality, which is the sum of the relevant cut values, is $\bar{x}(\delta(V_C)) + \bar{x}(\delta(S_1)) + \bar{x}(\delta(S_2)) = 65.69$. The right-hand side (RHS) is $2r(\Omega) + \lceil \frac{d(S_1)}{Q} \rceil + 2\lceil \frac{d(S_2)}{Q} \rceil = 46 + 10 + 10 = 66$. Since the LHS is less than the RHS, the inequality is violated by 0.31.