Learning to Handle Complex Constraints for Vehicle Routing Problems

Jieyi Bi[1], Yining Ma[1,†], Jianan Zhou[1],
Wen Song[2], Zhiguang Cao[3], Yaoxin Wu[4], Jie Zhang[1]

[1]Nanyang Technological University
[2]Shandong University
[3]Singapore Management University
[4]Eindhoven University of Technology

jieyi001@e.ntu.edu.sg, yiningma@u.nus.edu,
jianan004@e.ntu.edu.sg, wensong@email.sdu.edu.cn,
zgcao@smu.edu.sg, y.wu2@tue.nl, zhangj@ntu.edu.sg

总结

摘要

- 以往的神经网络方法在基于可行性掩码构建解决方案方面表现出色，但在处理复杂约束时却力不从心，尤其是在获取掩码本身即为 NP 难问题的情况下。
- 本文提出了 **主动预防不可行性（Proactive Infeasibility Prevention, PIP）** 框架来提升神经网络方法解决复杂VRPs问题的能力。PIP以**拉格朗日乘子**为基础来增强对约束的感知，并引入**预防性不可行性掩码**来主动引导解的构建过程。
- 进一步的，本文还提出了PIP-D，应用 **辅助解码器** 和两种 **自适应策略** 来学习这些掩码，希望在提升性能的同时显著较少训练中的计算开销。
- 主要在带时间窗的旅行商问题（Traveling Salesman Problem with Time Window, TSPTW）和吃水限制的旅行商问题（Traveling Salesman Problem with Draft Limit, TSPDL）上针对简单、中等、困难三种难度的约束开展实验验证效果。
- PIP对多种神经网络方法通用，显著减少不适用率和提升解质量。
- Github：https://github.com/jieyibi/PIP-constraint

研究问题

TSP、CVRP问题的掩码机制假设：1）整个解的可行性可以被合理地分解为每个节点选择步骤的可行性；2）每一步的真实掩码容易获取。
但在决策变量间具有复杂的相互制约约束的VRPs问题上（如TSPTW、TSPDL），这样的假设将失效，考虑节点选取的局部可行性不能确保构造出来的解的全局可行性。

- **时间窗约束 (TSPTW)**：到达节点 v_i 的时间记为 t_i ，必须落在客户节点特定的时间窗 $[l_i, u_i]$ 内，如果车辆提前到达（$t_i < l_i$），则必须等待至 l_i 。
- **吃水限制约束 (TSPDL)**：每个节点 v_i 代表一个港口，具有非负需求 δ_i 和最大吃水 d_i 。在给定解中，记货轮在港口 v_i的当前累计载货量为 α_i， 它不能超过当前港口对应的最大吃水深度 d_i 。

方法

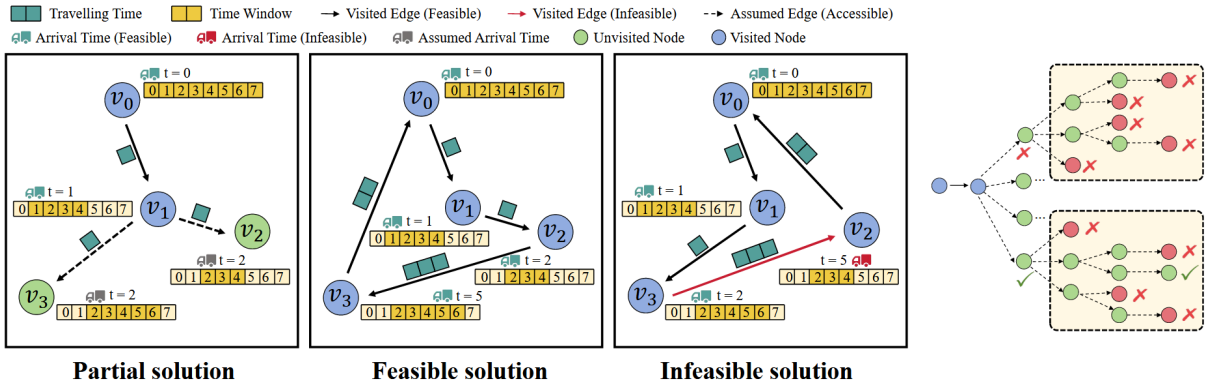

图1. 可行性掩码的困境。仅考虑局部可行性不能确保全局可行性。

如图，对于局部解 $v_0\rightarrow v_1$ ， v_2 和 v_3 是局部可行的，但如果下一步选择 v_3 ，整条路径就不可逆转地编程不可行了。
如果去计算涵盖未来所有可能性的全局可行性掩码（图1右侧），又将使掩码的计算本身成为一个NP难问题。
以往的方法尝试通过将约束条件从硬性条件转变为软性条件，或通过补充更多与可行性相关的特征来缓解这一问题，但前者在应用于更复杂场景时容易失效，后者则需要问题特定的特征以及大量的有监督学习数据集，限制了这些方法的适用性。

基于PIP的引导式策略搜索

首先，VRP问题的解构造过程可以建模成带约束马尔可夫决策过程（Constrained MDP, CMDP），用一个元组 $(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R},\mathcal{C})$表示CMDP。

> 其中，$\mathcal{S}$：状态空间；$\mathcal{A}$：动作空间；$\mathcal{P}$：转移概率函数；$\mathcal{R}$：奖励函数；$\mathcal{C}$：违反约束惩罚函数

该CMDP问题的目标函数为：

$$ \begin{equation} \begin{aligned} \max_{\theta} \mathcal{J}(\pi_{\theta}) &= \mathbb{E}{\tau \sim \pi{\theta}} \left[\sum_{e(v_i, v_j) \in \tau} \mathcal{R}(e(v_i, v_j)) \right], \ \text{s.t.} \quad \pi_{\theta} &\in \Pi_F, \quad \Pi_F = {\pi \in \Pi \mid \mathcal{J}_{\mathcal{C}_m}(\pi) \leq \kappa_m, , \forall m \in [1, M]}, \end{aligned} \end{equation} $$

> 其中，Π_F：所有可行策略的集合；κ_m：不等式约束 $\mathcal{C}*m$ 的边界，本文中固定为0；$\mathcal{J}*{\mathcal{C}_m}(\pi)$：可行性策略 π 的违反约束期望值；M：约束数量

应用传统可行性掩码的神经网络求解器仅专注于上述目标函数，没有考虑约束感知和违反约束的情况。当可行性掩码失效时，神经网络求解器将在大型的不可行区域中进行效率低下的搜索。

- **一、拉格朗日乘子法辅助的约束感知** 基于拉格朗日乘数的思想，PIP将约束 $\mathcal{C}$ 纳入奖励函数 R 中，CMDP的目标函数转化为： $$ \min_{\lambda \geq 0} \max_{\theta} \mathcal{L}(\lambda, \theta) = \min_{\lambda \geq 0} \max_{\theta} -\mathbb{E}{\tau \sim \pi{\theta}} \left[\sum_{e(v_i, v_j) \in \tau} |v_i - v_j|2 + \sum{m=1}^{M} \lambda_m \mathcal{J}{C_m}(\tau) + \mathcal{J}{\text{IN}} \right] $$

> 其中，$\mathcal{L}$：拉格朗日函数；λ_m：非负拉格朗日乘子 约束违反项通过计算所有约束的违规值计算得到的总和。 $\mathcal{J}{\text{TW}}(\tau) = \sum{i=0}^{n} \max(t_i - u_i, 0)$（TSPTW）

> $\mathcal{J}{\text{DL}}(\tau) = \sum{i=0}^{n} \max(\alpha_i - d_i, 0)$（TSPDL） $\mathcal{J}_{IN}$：解τ中不可行节点的数量

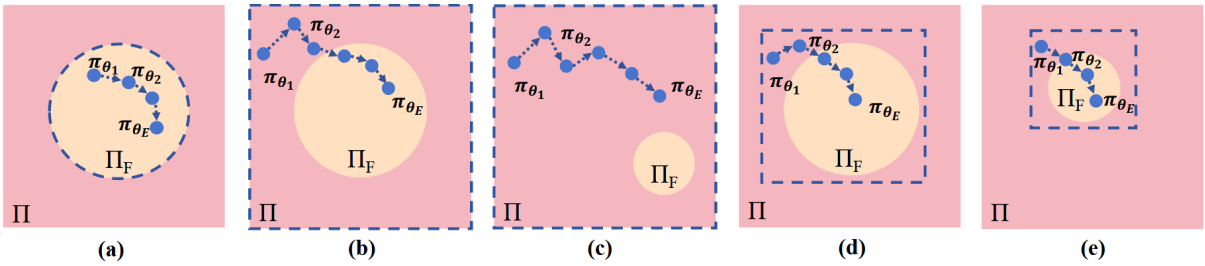

图2. 不同难度（容易：a、b、d；困难：c、e）和不同约束处理方案（可行性掩码：a；拉格朗日乘子：b、c；PIP：d、e）在VRP问题上的策略优化轨迹示意图。

- **二、预防性不可行掩码** 为进一步提高训练效率和解的可行性，本文提出预防性不可行性掩码，以在解构造过程中主动避免选取不可行节点。
  如图3左侧所示，若选择某一候选节点将导致下一步中任一剩余候选节点因违反约束而不可行时，将该候选节点进行掩码。本文仅采用一步PI掩码。
  结合拉格朗日乘子和一步PI掩码，PIP主动将搜索空间缩减至近乎可行的域$\Pi_{\tilde{F}}$中。

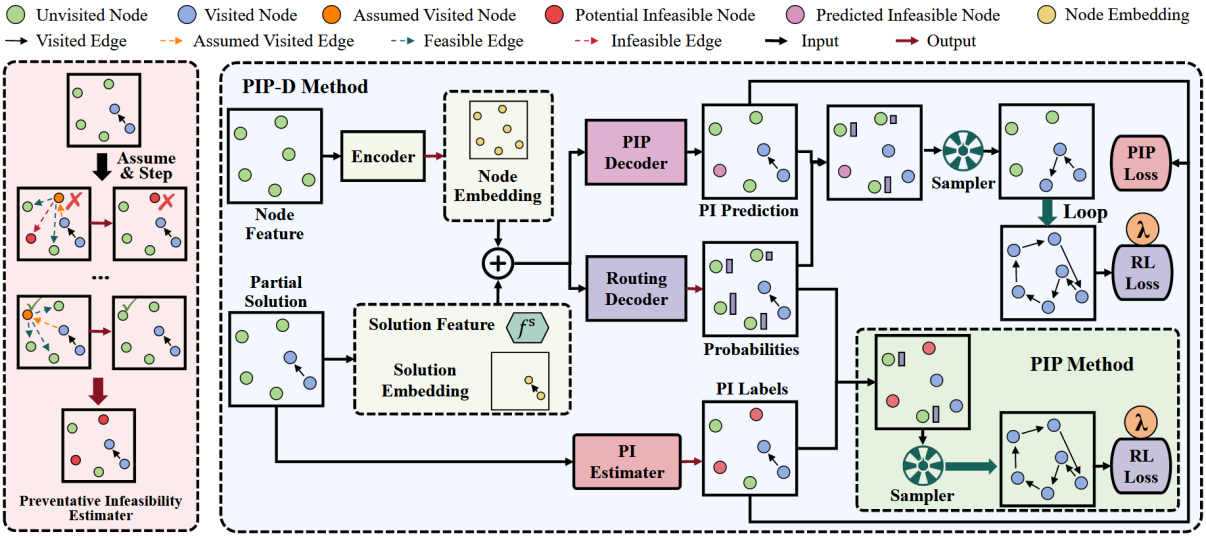

图3. 左边：预防性不可行估计器（Preventative infeasibility (PI) estimator）
右边：PIP框架（绿色）和PIP-D框架（蓝色）

辅助解码器学习预防不可行性

获取PI信息引入额外的计算开销，本文提出使用一个辅助解码器网络来学习和预测这些掩码，用一种速度更快的 PIP 解码器前向处理过程来取代生成 PI 信息这一耗时的过程。

- **辅助PIP解码器** PIP-D同时训练一个最大化解的期望奖励的路由解码器，以及使用加权二进制交叉熵损失训练一个最小化PI掩码预测误差的PIP解码器。 联合损失函数为：$\mathcal{L}=\alpha \mathcal{L}_{RL}+\beta \mathcal{L}_{PIP}$

- **PIP-D通过自适应策略训练**

  - 采用周期性更新策略更新PIP解码器。具体来说，首先使用 E_{init} 个$epochs$对PIP解码器初始训练，然后每 E_p 个$epoch$更新 E_u 轮，最后再进行E_l个$epochs$更新。

  - 由于不同难度不同VRP变体间的可行和不可行节点PI信号比例相差很大，本文采用加权平衡策略来减轻标签不平衡的影响，即：

$$ \nabla \mathcal{L}_{\text{PIP}}(\theta|\mathcal{G}) = -\frac{1}{T} \sum_{t=0}^{T} \left(\omega_{\text{infsb}} \cdot g_t \cdot \nabla \log \left(p_{\theta}(g_t) \right) + \omega_{\text{fsb}} \cdot (1 - g_t) \cdot \nabla \log \left(1 - p_{\theta}(g_t) \right) \right) $$

> 其中，T是构造解的总解码次数；$\omega_{\text{infsb}} = \frac{N_{\text{infsb}} + N_{\text{fsb}}}{2N_{\text{misb}}}$, $\omega_{\text{fsb}} = \frac{N_{\text{misb}} + N_{\text{fsb}}}{2N_{\text{fsb}}}$，$N_{\text{infsb}}$和$N_{\text{fsb}}$分别是第 t 步解码时通过PI掩码识别的不可行节点和可行节点数量。

实验

实验设备：NVIDIA GeForce RTX 3090 GPUs and Intel(R) Xeon(R) Gold 6326 CPU at 2.90GHz。

实验设置：

- 自回归方法：AM、POMO，节点数：50、100
- 非自回归方法：GFACS，节点数：500
- TSPTW：通过调整时间窗对的宽度和重叠度生成容易、中等、困难三种难度实例
- TSPDL：生成中等、困难两种难度实例
- 基线：
  - 启发式方法：LKH3、OR-Tools、贪心启发式方法
  - 神经网络方法：AM、POMO、GFAS、JAMPR、MUSLA
- 评估指标：不可行解比例、平均最佳差距、平均路径长度、推理时间

	Method	$n = 50$ Infeasible% Sol.↓	Inst.↓	Obj.↓	Gap↓	Time↓	$n = 100$ Infeasible% Sol.↓	Inst.↓	Obj.↓	Gap↓	Time↓
Easy	LKH3	0.00%	0.00%	7.31	0.00%	4.6h	0.00%	0.00%	10.21	0.00%	8.5h
	ORTools	0.00%	0.00%	7.34	0.96%	7h	0.00%	0.00%	10.41	1.97%	14h
	Greedy-L	100.00%	100.00%	/	/	13.8s	100.00%	100.00%	/	/	1.3m
	Greedy-C	0.00%	0.00%	26.08	257.27%	4.5s	0.00%	0.00%	52.14	411.13%	12s
	JAMPR #	/	0.00%	/	249.03%	1.2m	/	100.00%	/	/	1.6m
	OSLA #	/	11.80%	/	8.15%	15.6s	/	/	/	/	/
	MUSLA #	/	8.20%	/	7.32%	1.3m	/	18.60%	/	14.6%	9.8m
	MUSLA adapt #	/	0.10%	/	5.63%	7.7m	/	0.60%	/	12.01%	1.1h
	AM	100.00%	100.00%	/	/	5m	100.00%	100.00%	/	/	21m
	AM*	3.46%	0.22%	8.02	9.82%	5.2m	7.87%	1.49%	11.84	16.07%	21m
	AM*+PIP	0.55%	0.00%	7.87	7.67%	10.7m	0.45%	0.00%	11.42	11.86%	1h
	AM*+PIP-D	0.51%	0.00%	7.91	8.19%	11m	0.25%	0.00%	11.53	13.02%	1h
	POMO	100.00%	100.00%	/	/	13s	100.00%	100.00%	/	/	21s
	POMO*	1.75%	0.00%	7.54	3.08%	13s	2.11%	0.00%	10.83	6.07%	21s
	POMO* + PIP	0.32%	0.00%	7.50	2.65%	15s	0.15%	0.00%	10.57	3.53%	48s
	POMO* + PIP-D	0.28%	0.00%	7.49	2.51%	15s	0.06%	0.00%	10.66	4.39%	48s
Medium	LKH3	0.00%	0.00%	13.02	0.00%	7h	0.00%	0.00%	18.74	0.00%	10.8h
	ORTools	15.77%	15.77%	13.02	0.30%	5.9h	0.52%	0.52%	19.34	3.23%	13.8h
	Greedy-L	100.00%	100.00%	/	/	15s	100.00%	100.00%	/	/	1m
	Greedy-C	47.52%	47.52%	25.33	96.43%	4.2s	20.34%	20.34%	51.62	176.07%	11.4s
	AM	100.00%	100.00%	/	/	5m	100.00%	100.00%	/	/	21m
	AM*	24.84%	0.27%	13.81	6.11%	5m	50.19%	0.09%	21.42	14.34%	21m
	AM*+PIP	7.62%	0.35%	13.68	5.06%	11m	12.73%	0.04%	20.57	9.82%	1h
	AM*+PIP-D	11.96%	0.33%	13.65	4.87%	11m	8.80%	0.02%	20.80	11.03%	1h
	POMO	100.00%	100.00%	/	/	13s	100.00%	100.00%	/	/	21s
	POMO*	14.92%	3.77%	13.68	5.23%	13s	18.77%	0.12%	20.78	10.93%	21s
	POMO* + PIP	4.53%	0.90%	13.40	2.91%	15s	3.88%	0.19%	19.61	4.65%	48s
	POMO* + PIP-D	3.83%	0.65%	13.45	3.32%	15s	3.34%	0.03%	19.79	5.64%	48s
Hard	LKH3	0.12%	0.12%	25.61	0.00%	7h	0.07%	0.07%	51.24	0.00%	1.4d
	ORTools	65.72%	65.72%	25.76	-0.00%	2.4h	89.07%	89.07%	51.61	0.00%	1.6h
	Greedy-L	100.00%	100.00%	/	/	21.8s	100.00%	100.00%	/	/	1.3m
	Greedy-C	72.55%	72.55%	26.39	1.53%	4.5s	93.38%	93.38%	52.95	1.43%	11.1s
	AM	100.00%	100.00%	/	/	5m	100.00%	100.00%	/	/	21m
	AM*	39.87%	18.88%	26.08	1.425%	5m	100.00%	100.00%	/	/	21m
	AM*+PIP	18.07%	1.98%	25.71	0.38%	11m	41.92%	16.46%	51.49	0.47%	1h
	AM*+PIP-D	30.39%	4.40%	25.80	0.67%	11m	53.09%	5.33%	51.55	0.57%	1h
	POMO	100.00%	100.00%	/	/	13s	100.00%	100.00%	/	/	21s
	POMO*	39.26%	35.25%	26.22	1.61%	13s	100.00%	100.00%	/	/	21s
	POMO* + PIP	5.54%	2.67%	25.66	0.18%	15s	31.49%	16.27%	51.42	0.37%	48s
	POMO* + PIP-D	6.76%	3.07%	25.69	0.28%	15s	13.18%	6.48%	51.39	0.31%	48s

表1. 三种不同难度TSPTW问题实验结果

	Method	$n = 50$ Infeasible% Sol.↓	Inst.↓	Obj.↓	Gap↓	Time↓	$n = 100$ Infeasible% Sol.↓	Inst.↓	Obj.↓	Gap↓	Time↓
Medium	LKH3	0.00%	0.00%	10.87	0.00%	5.1h	0.00%	0.00%	16.39	0.00%	14h
	ORTools	100.00%	100.00%	/	/	10.9s	100.00%	100.00%	/	/	56.9s
	Greedy-L	100.00%	100.00%	/	/	2.4m	100.00%	100.00%	/	/	9.5m
	Greedy-C	0.00%	0.00%	26.09	144.24%	9.1s	0.00%	0.00%	52.16	222.71%	27s
	POMO*	17.72%	12.52%	10.98	3.80%	6.9s	49.39%	32.19%	17.11	9.15%	18s
	POMO* + PIP	2.21%	0.43%	11.22	3.41%	8.5s	2.88%	0.38%	17.71	8.08%	31s
	POMO* + PIP-D	2.64%	0.37%	11.26	3.78%	8.4s	2.14%	0.23%	17.84	8.86%	31s
Hard	LKH3	0.00%	0.00%	13.30	0.00%	6.8h	0.00%	0.00%	20.70	0.00%	1.2d
	ORTools	100.00%	100.00%	/	/	10.6s	100.00%	100.00%	/	/	56.8s
	Greedy-L	100.00%	100.00%	/	/	2.4m	100.00%	100.00%	/	/	9.4m
	Greedy-C	0.00%	0.00%	26.07	99.73%	10.9s	0.00%	0.00%	52.17	156.37%	25s
	POMO*	37.01%	29.25%	13.03	4.11%	6.8s	99.98%	99.85%	20.95	15.87%	18s
	POMO* + PIP	4.53%	2.10%	13.66	3.13%	8.5s	28.55%	20.66%	22.30	12.67%	31s
	POMO* + PIP-D	3.89%	0.82%	13.80	3.95%	8.5s	12.84%	7.91%	22.84	12.32%	31s

表2. 两种不同难度TSPDL问题实验结果

- 原始的 AM 和 POMO 即便在最简单的层面上也无法解决这些问题。通过引入拉格朗日乘数，这些模型开始能够生成一些可行的解决方案。然而，在更复杂的约束条件下，这一优势会逐渐减弱。
- 与 ORTools、Greedy-L 和 Greedy-C 等传统启发式算法相比，PIP-D 算法始终表现更优，并且在大规模问题中，其效果优于 JAMPR 和 MUSLA 算法。

- 与 PIP 相比，PIP-D 在提供具有竞争力甚至更优的解和最优性差距的同时，显著提高了训练效率。

Method	Infeasible%		Gap↓	Time↓
	Sol.↓	Inst.↓		
LKH3	0.00%	0.00%	0.00%	26m
Greedy-L	100.00%	100.00%	/	3.2m
Greedy-C	100.00%	100.00%	/	4.1s
GFACS*	58.20%	57.81%	21.32%	6.4m
GFACS* + PIP	4.72%	1.56%	15.04%	6.5m
GFACS* + PIP-D	0.03%	0.00%	11.95%	6.5m

表3. 中等难度TSPTW-500上的实验结果

- 为 GFACS 配备 PIP 可显著降低不可行率，同时提高了解质量。

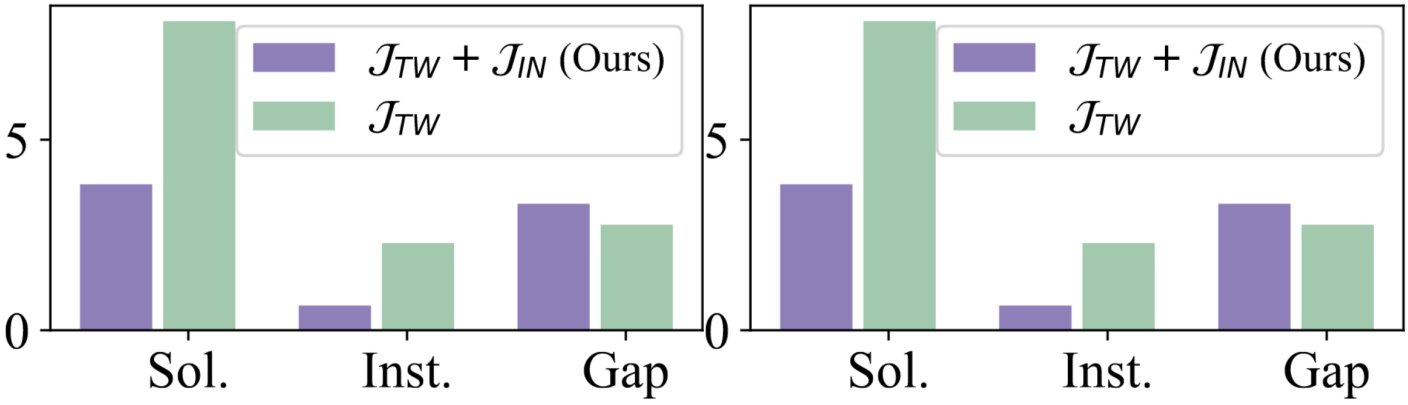

Table 4: Results of PIP steps on Medium TSPTW-50.

Method	PIP Step	Sol. Infsb%↓	Inst. Infsb%↓	Gap↓	Time↓
POMO*+PIP	0	76.92%	47.28%	4.24%	13s
POMO*+PIP	1	4.53%	0.90%	2.91%	15s
POMO*+PIP	2	2.90%	0.50%	2.93%	4.2m
POMO*+PIP-D	0	47.86%	20.86%	3.49%	13s
POMO*+PIP-D	1	3.83%	0.65%	3.32%	15s
POMO*+PIP-D	2	2.59%	0.35%	3.34%	4.2m

Table 5: Results of PIP steps on Hard TSPTW-100.

Model	PIP Step	Sol. Infsb%↓	Inst. Infsb%↓	Gap↓	Time↓
POMO*+PIP	0	100.00%	100.00%	/	21s
POMO*+PIP	1	31.49%	16.27%	0.37%	48s
POMO*+PIP	2	26.87%	12.88%	0.37%	35m
POMO*+PIP-D	0	79.73%	63.29%	0.31%	21s
POMO*+PIP-D	1	13.18%	6.48%	0.31%	48s
POMO*+PIP-D	2	11.62%	5.63%	0.31%	35m

Table 6: Results on LKH3 with the similar instance inference time limit as POMO*+PIP(-D).

	Method	$n = 50$			$n = 100$		
		Inst. Time	Obj.	Inst. Infsb%	Inst. Time	Obj.	Inst. Infsb%
Easy	LKH3 (Default)	27s	7.31	0.00%	49s	10.21	0.00%
	LKH3	0.37s	7.35	0.00%	0.9s	10.37	0.00%
	POMO*+PIP	0.38s	7.50	0.00%	0.9s	10.57	0.00%
	POMO*+PIP-D	0.38s	7.49	0.00%	0.9s	10.66	0.00%

Medium	LKH3 (Default)	40s	13.02	0.00%	1.0m	18.74	0.00%
	LKH3	0.37s	13.06	0.00%	0.9s	19.00	0.00%
	POMO*+PIP	0.38s	13.40	0.90%	0.9s	19.61	0.19%
	POMO*+PIP-D	0.38s	13.45	0.65%	0.9s	19.79	0.03%
Hard	LKH3 (Default)	40s	25.61	0.12%	3.2m	51.24	0.07%
	LKH3	0.37s	25.43	30.60%	0.9s	49.94	97.28%
	POMO*+PIP	0.38s	25.66	2.67%	0.9s	51.42	16.27%
	POMO*+PIP-D	0.38s	25.69	3.07%	0.9s	51.39	6.48%

Table 7: Results on LKH3 with the similar total inference time limit as POMO*+PIP(-D).

	Method	$n = 50$			$n = 100$		
		Total Time	Obj.	Inst. Infsb%	Total Time	Obj.	Inst. Infsb%
Easy	LKH3 (Default)	4.6h	7.31	0.00%	8.5h	10.21	0.00%
	LKH3	26s	8.81	99.29%	58s	/	100.00%
	POMO*+PIP	21s	7.50	0.00%	48s	10.57	0.00%
	POMO*+PIP-D	21s	7.49	0.00%	48s	10.66	0.00%
Medium	LKH3 (Default)	7h	13.02	0.00%	10.8h	18.74	0.00%
	LKH3	25s	13.05	39.91%	63s	/	100.00%
	POMO*+PIP	21s	13.40	0.90%	48s	19.61	0.19%
	POMO*+PIP-D	21s	13.45	0.65%	48s	19.79	0.03%
Hard	LKH3 (Default)	7h	25.61	0.12%	1.4d	51.24	0.07%
	LKH3	22s	/	100.00%	54s	/	100.00%
	POMO*+PIP	21s	25.66	2.67%	48s	51.42	16.27%
	POMO*+PIP-D	21s	25.69	3.07%	48s	51.39	6.48%

结论

本文提出了主动不可行预防（PIP）框架，以提升处理复杂约束VRP的能力。通过引入拉格朗日乘子法和预防性不可行掩码，主动引导解构造过程。进一步整合辅助解码器，PIP框架显著提升训练效率，并在复杂数据集上展现卓越性能。 未来研究方向包括：

1. 探索降低计算复杂度的新策略（如用可训练热力图缩小PI掩码计算空间）；
2. 将PIP应用于更大规模的神经方法；
3. 将PIP扩展至神经迭代式求解器；
4. 应用于更多复杂约束VRP变体（包括掩码非NP难但最优性差距大的场景）；
5. 探索PIP在其他领域的应用（如作业车间调度——需顺序执行操作且可主动预防不可行性）；
6. 发展PIP的理论证明。

附录

TSPTW实例生成方法

> 每个实例包含：1个仓库节点 + n个客户节点 每个节点特征：欧式空间二维坐标 (x_i, y_i)；时间窗下界 l_i；时间窗上界 u_i

- $(x_i, y_i) ~ $\mathcal{U}$[0,100]$，在 $[0,100]×[0,100]$ 正方形内**均匀分布**
- **时间窗生成的三种方法对比**

方法	核心原理	优点	缺点	代表文献
1. 基于近优TSP解生成	1. 先求TSP近优解 2. 按路径相邻节点距离设置时间窗	保证可行解存在	1. 计算开销大 2. 削弱时间窗约束强度	[74, 75]

方法	核心原理	优点	缺点	代表文献
2. 基于随机路径生成	1. 随机排列节点 2. 按随机路径相邻距离设置时间窗	保证可行解存在	时间窗约束强度不足	[30, 67, 76]
3. 均匀随机生成	直接对(l_i,u_i)均匀采样，无路径先验	通用性强/更接近现实场景	不保证可行解存在	[7, 9]

难度	时间窗生成方法	关键参数	约束特性	可行性保证
Easy	独立均匀采样	$l_i ~ $\mathcal{U}$[0,T_N]$ $u_i = l_i + T_N·$\mathcal{U}$[0.5,0.75]$	时间窗宽（50%-75%T_N）	无保证
Medium	独立均匀采样	$l_i ~ $\mathcal{U}$[0,T_N]$ $u_i = l_i + T_N·$\mathcal{U}$[0.1,0.2]$	时间窗窄（10%-20%T_N）	无保证
Hard	基于随机路径的累计距离	$l_i ~ $\mathcal{U}$[ψ_i-50,ψ_i]$ $u_i ~ $\mathcal{U}$[ψ_i,ψ_i+50]$	极窄窗（固定宽度50）	有保证

> **参数说明**：
>
> - T_N：规模N的问题的期望路径长度（如$T_{20}≈10.9$）
> - $ψ_i$：从仓库到节点i的随机路径累计距离
> - **Hard难度**：将原文献[67]的$η=500$收紧至$η=50$，大幅提升难度

归一化处理：

- 坐标归一化：$(x_i,y_i)/100 → [0,1]$
- 时间窗归一化：$(l_i,u_i)/u_0 → [0,1]$，其中 $u_0 = max(u_i + ‖v_i-v_0‖_2)$

TSPDL实例生成方法

参考文献(后续阅读)

> Jingxiao Chen, Ziqin Gong, Minghuan Liu, Jun Wang, Yong Yu, and Weinan Zhang. Looking ahead to avoid being late: Solving hard-constrained traveling salesman problem. arXiv preprint arXiv:2403.05318, 2024.

VRPs神经网络求解器

- 提升性能

> Liang Xin, Wen Song, Zhiguang Cao, and Jie Zhang. Multi-decoder attention model with embedding glimpse for solving vehicle routing problems. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 12042–12049, 2021.
> Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural

combinatorial optimization. In Advances in Neural Information Processing Systems, 2022.
André Hottung, Yeong-Dae Kwon, and Kevin Tierney. Efficient active search for combinatorial optimization problems. In International Conference on Learning Representations, 2022.
Jinho Choo, Yeong-Dae Kwon, Jihoon Kim, Jeongwoo Jae, André Hottung, Kevin Tierney, and Youngjune Gwon. Simulation-guided beam search for neural combinatorial optimization. In Advances in Neural Information Processing Systems, volume 35, pages 8760–8772, 2022.
Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for generalizable neural combinatorial optimization. In Advances in Neural Information Processing Systems, 2023.
Felix Chalumeau, Shikha Surana, Clément Bonnet, Nathan Grinsztajn, Arnu Pretorius, Alexandre Laterre, and Thomas D Barrett. Combinatorial optimization with policy adaptation using latent space search. In Advances in Neural Information Processing Systems, 2023.
Nathan Grinsztajn, Daniel Furelos-Blanco, Shikha Surana, Clément Bonnet, and Thomas D Barrett. Winner takes it all: Training performant RL populations for combinatorial optimization. In Advances in Neural Information Processing Systems, 2023.
Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In Advances in Neural Information Processing Systems, 2023.
André Hottung, Mridul Mahajan, and Kevin Tierney. PolyNet: Learning diverse solution strategies for neural combinatorial optimization. arXiv preprint arXiv:2402.14048, 2024.
Fu Luo, Xi Lin, Zhenkun Wang, Tong Xialiang, Mingxuan Yuan, and Qingfu Zhang. Self-improved learning for scalable neural combinatorial optimization. arXiv preprint arXiv:2403.19561, 2024.

- 提升功能性

Yeong-Dae Kwon, Jinho Choo, Iljoo Yoon, Minah Park, Duwon Park, and Youngjune Gwon. Matrix encoding networks for neural combinatorial optimization. In Advances in Neural Information Processing Systems, volume 34, 2021.
Jingwen Li, Yining Ma, Ruize Gao, Zhiguang Cao, Andrew Lim, Wen Song, and Jie Zhang. Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem. IEEE Transactions on Cybernetics, 2021.
Federico Berto, Chuanbo Hua, Junyoung Park, Laurin Luttmann, Yining Ma, Fanchen Bu, Jiarui Wang, Haoran Ye, Minsu Kim, Sanghyeok Choi, Nayeli Gast Zepeda, André Hottung, Jianan Zhou, Jieyi Bi, Yu Hu, Fei Liu, Hyeonah Kim, Jiwoo Son, Haeyeon Kim, Davide Angioni, Wouter Kool, Zhiguang Cao, Jie Zhang, Kijung Shin, Cathy Wu, Sungsoo Ahn, Guojie Song, Changhyun Kwon, Lin Xie, and Jinkyoo Park. RL4CO: an extensive reinforcement learning for combinatorial optimization benchmark. arXiv preprint arXiv:2306.17100, 2023.
Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Xu Chi. MVMoE: Multitask vehicle routing solver with mixture-of-experts. In International Conference on Machine Learning, 2024.

- 非自回归

Minsu Kim, Sanghyeok Choi, Jiwoo Son, Hyeonah Kim, Jinkyoo Park, and Yoshua Bengio. Ant colony sampling with gflownets for combinatorial optimization. arXiv preprint arXiv:2403.07041, 2024.
Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. arXiv preprint arXiv:1906.01227, 2019.
Zhang-Hua Fu, Kai-Bin Qiu, and Hongyuan Zha. Generalize a small pre-trained model to arbitrarily

large tsp instances. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 7474–7482, 2021.
Wouter Kool, Herke van Hoof, Joaquim Gromicho, and Max Welling. Deep policy dynamic programming for vehicle routing problems. In International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research, pages 190–213. Springer, 2022.
Ruizhong Qiu, Zhiqing Sun, and Yiming Yang. Dimes: A differentiable meta solver for combinatorial optimization problems. Advances in Neural Information Processing Systems, 35:25531–25546, 2022.
Zhiqing Sun and Yiming Yang. Difusco: Graph-based diffusion solvers for combinatorial optimization. Advances in Neural Information Processing Systems, 2023.
Yimeng Min, Yiwei Bai, and Carla P Gomes. Unsupervised learning for solving the travelling salesman problem. Advances in Neural Information Processing Systems, 2023.
Haoran Ye, Jiarui Wang, Zhiguang Cao, Helan Liang, and Yong Li. DeepACO: Neural-enhanced ant systems for combinatorial optimization. In Advances in Neural Information Processing Systems, 2023.

- 可扩展性

Sirui Li, Zhongxia Yan, and Cathy Wu. Learning to delegate for large-scale vehicle routing. Advances in Neural Information Processing Systems, 34:26198–26211, 2021.
Zefang Zong, Hansen Wang, Jingwei Wang, Meng Zheng, and Yong Li. Rbg: Hierarchically solving large-scale routing problems in logistic systems via reinforcement learning. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pages 4648–4658, 2022.
Hanni Cheng, Haosi Zheng, Ya Cong, Weihao Jiang, and Shiliang Pu. Select and optimize: Learning to aolve large-scale tsp instances. In International Conference on Artificial Intelligence and Statistics, pages 1219–1231. PMLR, 2023.
Yan Jin, Yuandong Ding, Xuanhao Pan, Kun He, Li Zhao, Tao Qin, Lei Song, and Jiang Bian. Pointerformer: Deep reinforced multi-pointer transformer for the traveling salesman problem. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, pages 8132–8140, 2023.
Xuanhao Pan, Yan Jin, Yuandong Ding, Mingxiao Feng, Li Zhao, Lei Song, and Jiang Bian. H-tsp: Hierarchically solving the large-scale traveling salesman problem. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, pages 9345–9353, 2023.
Qingchun Hou, Jingwei Yang, Yiqiang Su, Xiaoqing Wang, and Yuming Deng. Generalize learned heuristics to solve large-scale vehicle routing problems in real-time. In International Conference on Learning Representations, 2023.
Haoran Ye, Jiarui Wang, Helan Liang, Zhiguang Cao, Yong Li, and Fanzhang Li. Glop: Learning global partition and local construction for solving large-scale routing problems in real-time. In Proceedings of the AAAI Conference on Artificial Intelligence, 2024.

- 泛化性

Chaitanya K Joshi, Quentin Cappart, Louis-Martin Rousseau, and Thomas Laurent. Learning tsp requires rethinking generalization. In International Conference on Principles and Practice of Constraint Programming, 2021.
Zeyang Zhang, Ziwei Zhang, Xin Wang, and Wenwu Zhu. Learning to solve travelling salesman problem with hardness-adaptive curriculum. In Proceedings of the AAAI Conference on Artificial Intelligence, 2022.
Jieyi Bi, Yining Ma, Jiahai Wang, Zhiguang Cao, Jinbiao Chen, Yuan Sun, and Yeow Meng Chee. Learning generalizable models for vehicle routing problems via knowledge distillation. In Advances in Neural Information Processing Systems, 2022.

> Jiwoo Son, Minsu Kim, Hyeonah Kim, and Jinkyoo Park. Meta-SAGE: Scale meta-learning scheduled adaptation with guided exploration for mitigating scale shift on combinatorial optimization. In International Conference on Machine Learning, 2023.
> Jianan Zhou, Yaoxin Wu, Wen Song, Zhiguang Cao, and Jie Zhang. Towards omni-generalizable neural methods for vehicle routing problems. In International Conference on Machine Learning, pages 4276942789. PMLR, 2023.
> Chenguang Wang, Zhouliang Yu, Stephen McAleer, Tianshu Yu, and Yaodong Yang. ASP: Learn a universal neural solver! IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024.

- 鲁棒性

> Simon Geisler, Johanna Sommer, Jan Schuchardt, Aleksandar Bojchevski, and Stephan Günnemann. Generalization of neural combinatorial solvers through the lens of adversarial robustness. In International Conference on Learning Representations, 2022.
> Han Lu, Zenan Li, Runzhong Wang, Qibing Ren, Xijun Li, Mingxuan Yuan, Jia Zeng, Xiaokang Yang, and Junchi Yan. ROCO: A general framework for evaluating robustness of combinatorial optimization solvers on graphs. In International Conference on Learning Representations, 2023.