LMask: Learn to Solve Constrained Routing Problems with Lazy Masking

Tianyou Li [1] Haijun Zou [2] Jiayuan Wu [3] Zaiwen Wen [1]

[1]Beijing International Center for Mathematical Research, Peking University [2]Academy of Mathematics and Systems Science, Chinese Academy of Sciences
[3]Department of Statistics and Data Science, University of Pennsylvania

Problem Formulation

Let $V := \{0, 1, \ldots, n\}$ denote the set of nodes and $\Pi := V^T$ denote the sequence space of all candidate solutions of length T. A wide range of constrained routing problems can be written as

$$\min_{\pi \in \Pi} \quad f(\pi; \mathcal{P}), \qquad \text{s.t.} \quad c(\pi; \mathcal{P}) \le 0, \quad d(\pi; \mathcal{P}) = 0. \tag{1}$$

Here $d(\pi; \mathcal{P}) = 0$ enforces validity constraints such as visiting each node exactly once, and $c(\pi; \mathcal{P}) \le 0$ represents complex hard constraints such as time windows or draft limits.

The feasible set is

$$C := \{\pi \in \Pi : \ c(\pi; \mathcal{P}) \le 0, \ d(\pi; \mathcal{P}) = 0\}.$$

In this paper, we focus on two representative problems:

- **TSPTW**: traveling salesman problem with time windows.
- **TSPDL**: traveling salesman problem with draft limits.

A Distribution Approximation View

Let Π^* be the optimal solution set and $f^*(\mathcal{P})$ be the optimal objective value. The target distribution over optimal solutions is

$$q^*(\pi; \mathcal{P}) := \frac{1}{|\Pi^*|} \mathbb{1}_{\Pi^*}(\pi).$$

Since q^* is inaccessible, we approximate it by the constrained Gibbs distribution

$$q_\lambda(\pi; \mathcal{P}) := \frac{1}{Z_\lambda} \exp\left(-\frac{f(\pi; \mathcal{P}) - f^*(\mathcal{P})}{\lambda}\right) \mathbb{1}_C(\pi),$$

where $Z_\lambda := \sum_{\pi \in C} \exp(-(f(\pi; \mathcal{P}) - f^*(\mathcal{P}))/\lambda)$. Moreover,

$$q_\lambda(\pi; \mathcal{P}) \to q^*(\pi; \mathcal{P}), \qquad \lambda \to 0.$$

We then learn a parameterized probabilistic model $p_\theta(\pi; \mathcal{P})$ by minimizing the KL divergence to q_λ:

$$\mathrm{KL}(p_\theta \| q_\lambda) = \mathbb{E}_{p_\theta}[\log p_\theta] + \frac{1}{\lambda} \mathbb{E}_{p_\theta}[f(\pi; \mathcal{P})] + \log Z_\lambda - f^*(\mathcal{P}).$$

Discarding θ-independent terms gives the equivalent optimization problem

$$\min_\theta \quad L(\theta; \mathcal{P}) := \mathbb{E}_{p_\theta(\cdot; \mathcal{P})}[f(\pi; \mathcal{P})] + \lambda \mathbb{E}_{p_\theta(\cdot; \mathcal{P})}[\log p_\theta(\pi; \mathcal{P})].$$

Why Existing Neural Solvers Are Insufficient

Existing neural constructive solvers employ an auto-regressive policy

$$p_\theta(\pi; \mathcal{P}) = \prod_{t=1}^{T-1} p_\theta(\pi_{t+1} \mid \pi_{1:t}; \mathcal{P}),$$

and handle constraints by masking infeasible actions in a one-pass forward decoding framework.

For complex hard constraints, this framework is inherently limited:

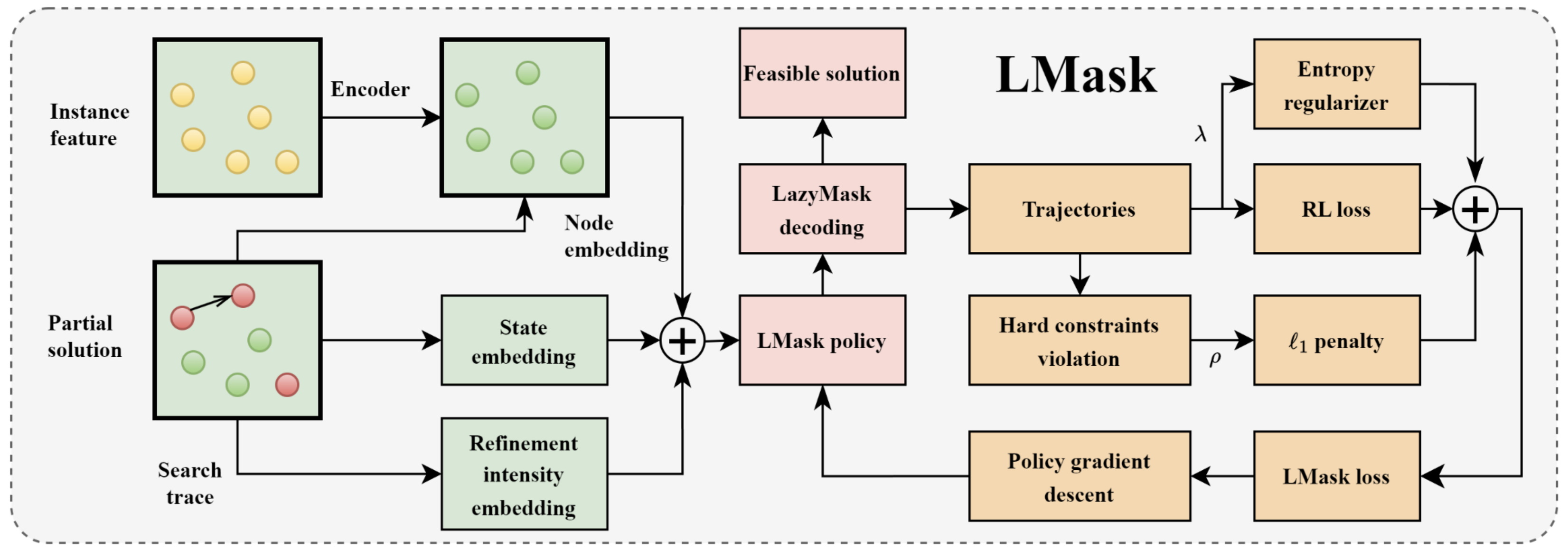
- **Local feasibility only.** Masking certifies the admissibility of the current partial solution, but not whether the residual subproblem still admits a feasible completion.
- **Irreversibility demands lookahead.** In one-pass forward decoding, each decision is irrevocable, so preserving future feasibility may require substantial lookahead.
- **Lookahead remains insufficient.** Deeper lookahead is computationally expensive and may still fail to certify a feasible completion.

Therefore, the main challenge is to learn a probabilistic model that can generate **high-quality feasible solutions** while explicitly handling feasibility failures during decoding.

Our Contributions

1. **Algorithm innovation.** We propose the **LazyMask** decoding algorithm, which lazily refines feasibility masks by backtracking, together with the **refinement intensity embedding** (RIE) to encode the search trace into the decoder.
2. **Theoretical guarantee.** We provide a systematic formulation of masking through the potential set, prove the validity of LazyMask, and establish probabilistic optimality guarantees for the learned model with entropy regularization.
3. **Experimental results.** On TSPTW and TSPDL, LMask substantially improves feasibility and solution quality over existing neural constructive baselines, while remaining fast and robust on benchmark and large-scale instances.

LMask Overview



LazyMask Decoding

The ideal potential set is

$$S(\pi_{1:t}) := \left\{\pi_{t+1} : \exists \pi_{t+1:T} \in V^{T-t}, [\pi_{1:t}, \pi_{t+1:T}] \in C\right\}.$$

It induces the constrained conditional probability

$$p_\theta(\pi_{t+1} \mid \pi_{1:t}; \mathcal{P}) = \frac{e^{\phi_\theta(\pi_{t+1} | \pi_{1:t}; \mathcal{P})} \mathbb{1}_{S(\pi_{1:t})}(\pi_{t+1})}{\sum_{k=0}^{n} e^{\phi_\theta(k | \pi_{1:t}; \mathcal{P})} \mathbb{1}_{S(\pi_{1:t})}(k)}.$$

Since $S(\pi_{1:t})$ is often computationally inaccessible, LazyMask replaces it by an overestimation set $\hat{S}(\pi_{1:t}) \supseteq S(\pi_{1:t})$, initialized by SSL or TSL and refined by backtracking.

Algorithm 1 LazyMask algorithm

Input: routing problem instance $\mathcal{P}$, neural network p_θ, backtracking budget R.
Initialize $\pi_1 := 0$, $t := 1$, $r := 0$ and the overestimation set $\hat{S}(\pi_1)$ either by SSL or TSL.
while $t \le T - 1$ **do**
  if $\hat{S}(\pi_{1:t}) = \emptyset$ and $r \le R$ **then**
    Update $\hat{S}(\pi_{1:t-1}) := \hat{S}(\pi_{1:t-1}) \backslash \{\pi_t\}$.
    Set $t := t - 1$, $r := r + 1$.
  else
    if $\hat{S}(\pi_{1:t}) = \emptyset$ **then**
      $\hat{S}(\pi_{1:t}) := V \setminus \{\pi_1, \ldots, \pi_t\}$.
    end if
    Calculate the probability $p_\theta(\cdot | \pi_{1:t}; \mathcal{P})$ using $\hat{S}(\pi_{1:t})$.
    Sample $\pi_{t+1} \sim p_\theta(\cdot | \pi_{1:t}; \mathcal{P})$.
    Set $t := t + 1$ and initialize $\hat{S}(\pi_{1:t})$ either by SSL or TSL.
  end if
end while
Output: route π.

Refinement Intensity Embedding

Standard dynamic features ignore the search trace induced by backtracking. RIE injects two refinement signals into the decoder:

- **Local feature:** the refinement count c_t of $\hat{S}(\pi_{1:t})$, encoded as a capped N-dimensional one-hot vector with active index $\min(c_t + 1, N)$.
- **Global feature:** a two-dimensional one-hot vector marking whether the total number of backtracks has reached the budget R.

These features are concatenated and then projected to form the final RIE. In this way, RIE makes the decoder explicitly aware of the refinement history induced by backtracking.

Training problem with ℓ_1 Penalty

With a finite backtracking budget, LMask is trained by

$$\min_\theta \quad \mathbb{E}_{\pi \sim p_\theta(\cdot; \mathcal{P})} \left[\Psi_\rho(\pi; \mathcal{P}) + \lambda \log p_\theta(\pi; \mathcal{P})\right],$$

where

$$\Psi_\rho(\pi; \mathcal{P}) := f(\pi; \mathcal{P}) + \rho \sum_{j=1}^{J} \max\bigl(c_j(\pi; \mathcal{P}), 0\bigr).$$

Here $\rho > 0$ is the penalty parameter and $c_j(\pi; \mathcal{P})$ denotes the violation of the j-th complex constraint.

Theoretical Results

Proposition (Validity of LazyMask). Suppose that problem (1) is feasible and the backtracking budget in LazyMask is $R = +\infty$. Then every solution π generated by LazyMask is feasible, and LazyMask assigns a non-zero probability to generate any feasible solution π.

Assumption. The approximation error of the auto-regressive model satisfies

$$\delta(\lambda) := \min_\theta \max_{\mathcal{P} \in \mathcal{D}} \mathrm{KL}\bigl(p_\theta(\cdot; \mathcal{P}) \parallel q_\lambda(\cdot; \mathcal{P})\bigr) \le \frac{c}{\lambda}, \qquad \lambda > 0,$$

where c is a small constant and $p_{\theta^*(\lambda)}$ denotes the corresponding optimal distribution.

Theorem (Probabilistic optimality). Define

$$\Delta(\mathcal{P}) := \min_{\pi \in C \setminus \Pi^*} f(\pi; \mathcal{P}) - f^*(\mathcal{P}).$$

Suppose the above assumption holds. Then, for any $\epsilon > 0$ and $\Delta(\mathcal{P}) \ge \lambda > 0$,

$$\mathbb{P}_{p_{\theta^*(\lambda)}}(f(\pi; \mathcal{P}) \ge f^*(\mathcal{P}) + \epsilon) \le \frac{|C| \, \Delta(\mathcal{P}) e^{-\Delta(\mathcal{P})/\lambda}}{|\Pi^*| \max\{\epsilon, \Delta(\mathcal{P})\}} + \sqrt{\frac{c}{2\lambda}}.$$

Main Benchmark Results

- Results on TSPTW synthetic datasets

	Nodes	$n = 50$					$n = 100$				
	Method	Infeasible Sol.	Infeasible Inst.	Obj.	Gap	Time	Infeasible Sol.	Infeasible Inst.	Obj.	Gap	Time
Medium	PyVRP	-	0.00%	13.03	*	1.7h	-	0.00%	18.72	*	4.3h
Medium	LKH3	-	0.00%	13.02	0.00%	2.9h	-	0.01%	18.74	0.16%	10.3h
Medium	VSR-LKH	-	0.00%	13.03	0.01%	8.2h	-	0.00%	18.72	0.00%	8.7h
Medium	OR-Tools	-	15.12%	13.01	0.12%	1.5h	-	0.52%	18.98	1.40%	4.3h
Medium	Random-L	98.31%	32.18%	18.91	47.04%	1.6m	100.00%	100.00%	-	-	5.8m
Medium	Random-C	91.17%	8.18%	21.04	61.91%	1.6m	100.00%	100.00%	-	-	5.9m
Medium	PIP	4.82%	1.07%	13.41	2.93%	10s	4.35%	0.39%	19.61	4.79%	29s
Medium	PIP-D	4.14%	0.90%	13.46	3.31%	9s	3.46%	0.03%	19.80	5.76%	31s
Medium	LMask	**0.04%**	**0.00%**	**13.25**	**1.68%**	6s	**0.05%**	**0.00%**	**19.51**	**4.23%**	18s
Hard	PyVRP	-	0.00%	25.61	*	1.7h	-	0.01%	51.27	*	4.3h
Hard	LKH3	-	0.52%	25.61	0.00%	2.3h	-	0.95%	51.27	0.00%	1d8h
Hard	VSR-LKH	-	0.52%	25.68	0.00%	4.4h	-	0.91%	51.27	0.00%	8.9h
Hard	OR-Tools	-	65.11%	25.92	0.00%	0.6h	-	89.25%	51.72	0.00%	0.5h
Hard	Random-L	100.00%	100.00%	-	-	1.6m	100.00%	100.00%	-	-	5.6m
Hard	Random-C	100.00%	99.82%	25.98	1.22%	1.6m	100.00%	100.00%	-	-	5.8m
Hard	PIP	5.65%	2.85%	25.73	0.18%	9s	31.74%	16.68%	51.48	0.37%	28s
Hard	PIP-D	6.44%	3.03%	25.75	0.27%	9s	13.59%	6.60%	51.43	0.32%	31s
Hard	LMask	**0.00%**	**0.00%**	**25.71**	**0.10%**	6s	**0.00%**	**0.00%**	**51.38**	**0.21%**	18s

- Results on TSPDL synthetic datasets

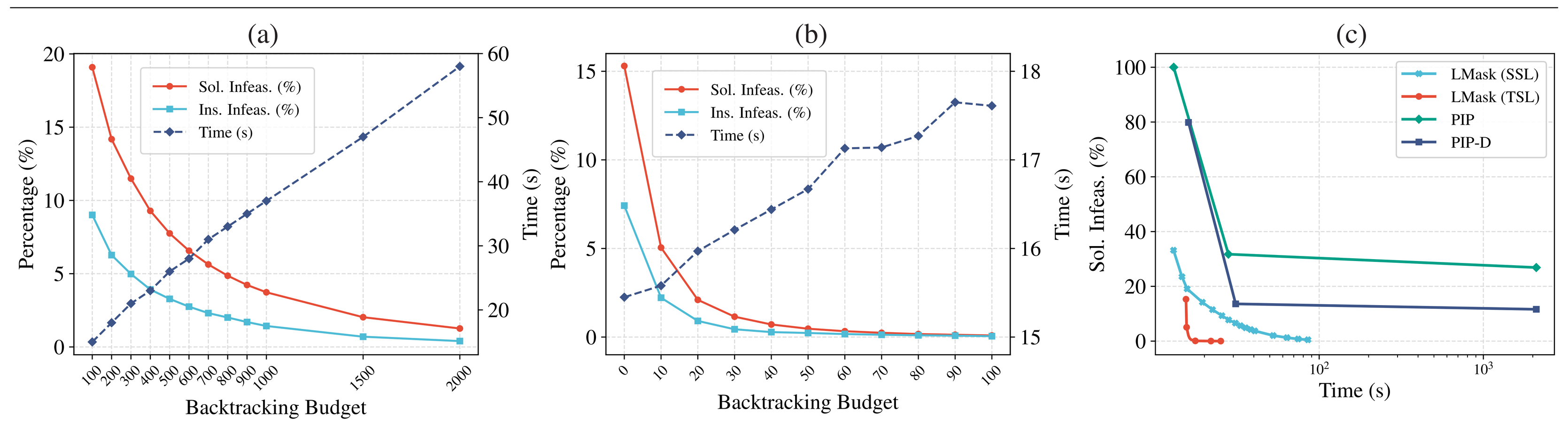
	Nodes	$n = 50$					$n = 100$				
	Method	Infeasible Sol.	Infeasible Inst.	Obj.	Gap	Time	Infeasible Sol.	Infeasible Inst.	Obj.	Gap	Time
Medium	LKH3	-	0.00%	10.85	*	2.3h	-	0.00%	16.36	*	10.2h
Medium	VSR-LKH	-	0.00%	10.85	0.08%	3.8h	-	0.00%	16.35	- 0.07%	11.0h
Medium	OR-Tools	-	100.00%	-	-	10.9s	-	100.00%	-	-	56.9s
Medium	Random-L	99.96%	97.28%	21.02	138.56%	38s	100.00%	100.00%	-	-	2.0m
Medium	Random-C	96.89%	47.39%	24.71	145.85%	37s	100.00%	99.98%	50.48	319.97%	2.0m
Medium	PIP	1.75%	0.17%	11.23	3.59%	8s	2.50%	0.16%	17.68	8.10%	21s
Medium	PIP-D	2.29%	0.22%	11.26	3.96%	8s	1.83%	0.23%	17.80	8.84%	23s
Medium	LMask	**0.03%**	**0.01%**	**11.14**	**2.75%**	6s	**0.20%**	**0.05%**	**17.04**	**4.24%**	15s
Hard	LKH3	-	0.00%	13.25	*	2.6h	0.00%	0.00%	20.76	*	15.8h
Hard	VSR-LKH	-	0.00%	13.25	0.05%	6.0h	-	0.00%	20.75	- 0.05%	17.2h
Hard	OR-Tools	-	100.00%	-	-	10.6s	-	100.00%	-	-	56.8s
Hard	Random-L	100.00%	99.96%	22.2	132.40%	37s	100.00%	100.00%	-	-	2.0m
Hard	Random-C	99.90%	94.05%	25.55	135.68%	37s	100.00%	100.00%	-	-	2.0m
Hard	PIP	4.83%	2.39%	13.63	3.42%	8s	29.34%	21.65%	22.35	12.87%	20s
Hard	PIP-D	4.16%	0.82%	13.79	4.28%	8s	13.51%	8.43%	22.90	12.53%	23s
Hard	LMask	**0.19%**	**0.04%**	**13.57**	**2.52%**	6s	**0.80%**	**0.26%**	**21.63**	**4.34%**	15s

- Results on the TSPTW benchmark

Nodes	$n = 20$			$n = 40$			$n = 60$			$n = 80$		
Method	**Infeas.**	**Obj.**	**Gap**	**Infeas.**	**Obj.**	**Gap**	**Infeas.**	**Obj.**	**Gap**	**Infeas.**	**Obj.**	**Gap**
PIP	5.0%	337.00	5.2%	45.0%	428.09	4.6%	20.0%	580.25	11.5%	22.2%	644.43	8.7%
PIP-D	5.0%	336.63	5.2%	25.0%	460.27	6.3%	40.0%	608.67	13.1%	66.7%	662.67	12.0%
LMask	0.0%	326.55	0.7%	0.0%	445.50	1.0%	0.0%	530.60	4.3%	0.0%	615.11	3.5%

- **Algorithmic Efficiency.** LMask is substantially faster than traditional solvers at inference time.
- **Solution Quality.** LMask achieves lower infeasibility and smaller gaps than competing neural methods.

Backtracking Analysis



Runtime grows nearly linearly with R, infeasibility drops sharply at small budgets, and TSL reaches zero solution infeasibility on hard TSPTW-100 within about 30 seconds.

Take-Home Message

- **LazyMask** replaces irreversible one-pass masking with a *forward-backward* decoding process.
- **RIE** is essential for making the decoder aware of the refinement history induced by backtracking.
- LMask provides a principled bridge between **constrained Gibbs approximation**, **feasibility-aware decoding**, and **strong empirical performance**.
- Across TSPTW and TSPDL, LMask achieves a better trade-off among feasibility, solution quality, and inference time than existing neural constructive methods.