PARCO:

Parallel AutoRegressive Models for Multi-Agent Combinatorial Optimization

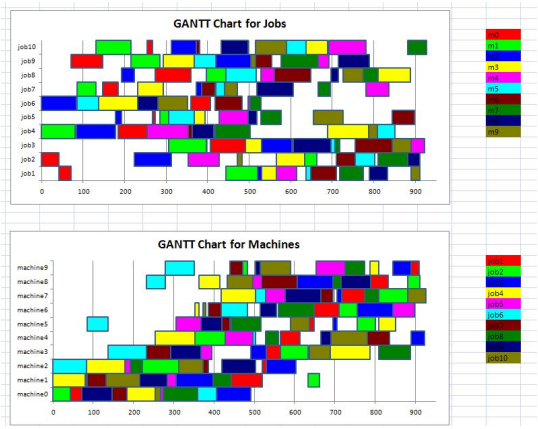
Federico Berto, Chuanbo Hua*, Laurin Luttmann*,*
Jiwoo Son, Junyoung Park, Kyuree Ahn,
Changhyun Kwon, Lin Xie, Jinkyoo Park

Combinatorial Optimization (CO)

AI4CO

Goal 🎯: finding an optimal set of actions from a finite set of discrete objects

Routing Problems



Scheduling Problems

The logistics industry is worth over 10 Trillion USD! (Statista, 2025)

Problem: CO is NP-hard!

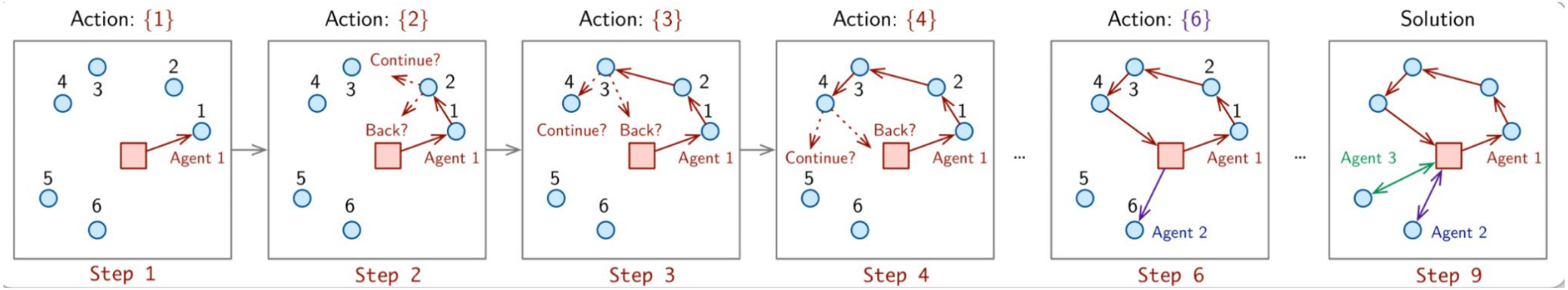
Solution: Reinforcement Learning (RL)

➔ Better solutions than traditional methods
➔ Fast and scalable solvers
➔ Less or no reliance on manual design

NEURAL INFORMATION PROCESSING SYSTEMS

Problem: Multi-Agent CO

Most current approaches to Multi-Agent CO decode solutions AutoRegressively (AR), one agent at a time:



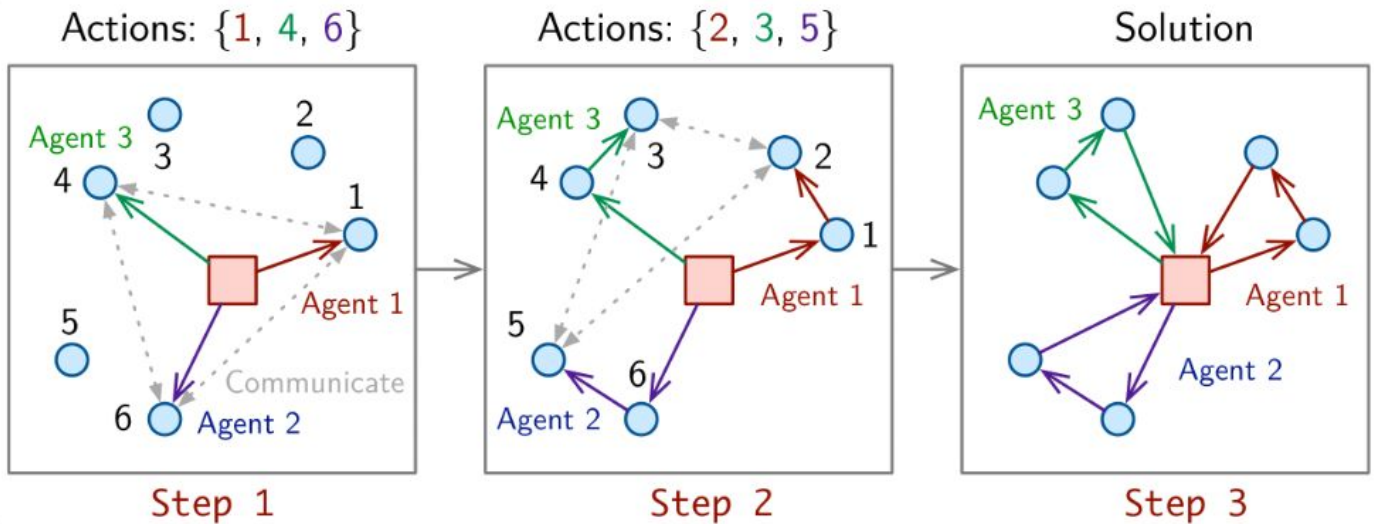
This results in:

1. Ineffective coordination :(
2. Slow decoding speed :(

Solution: PARCO

AI4CO

We propose Parallel AutoRegressive Combinatorial Optimization. We can decode in parallel for the different agents and have them communicate for each step:



This results in:

★ Effective coordination

★ Fast decoding speed

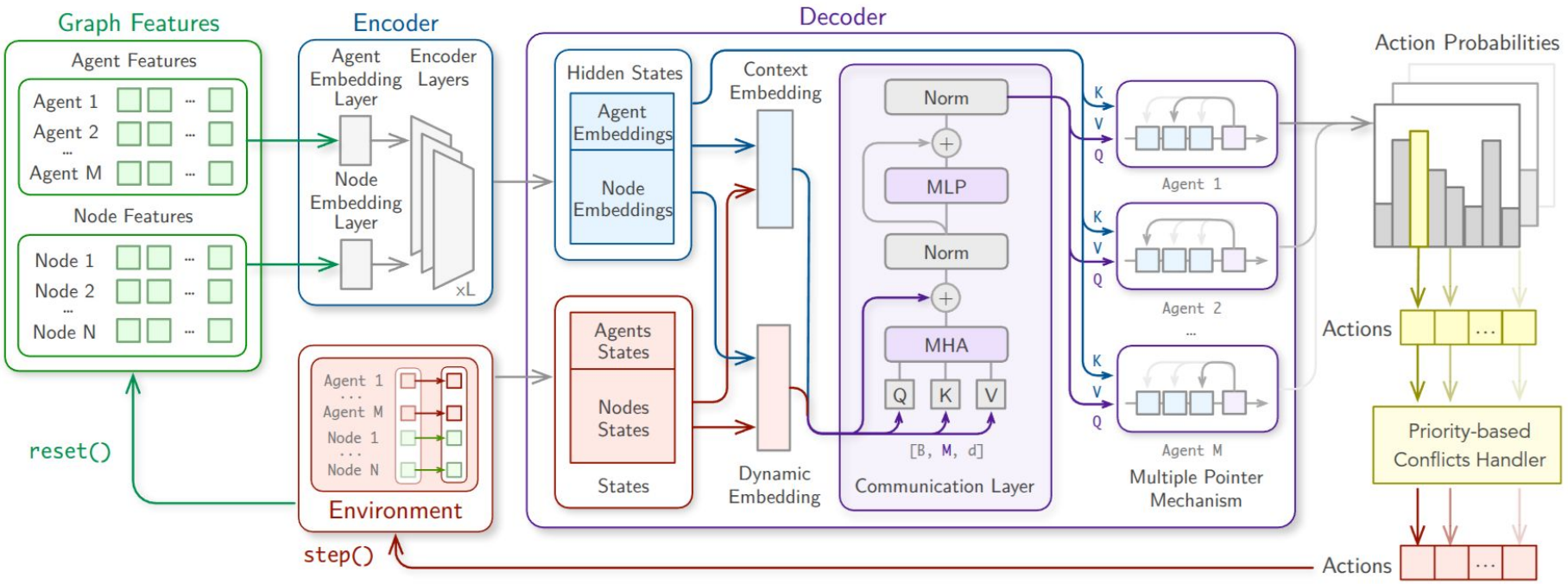
NEURAL INFORMATION PROCESSING SYSTEMS

Trivia: PARCO also simply means “park/공원/公园” in Italian 🏞️

PARCO Model

We propose:

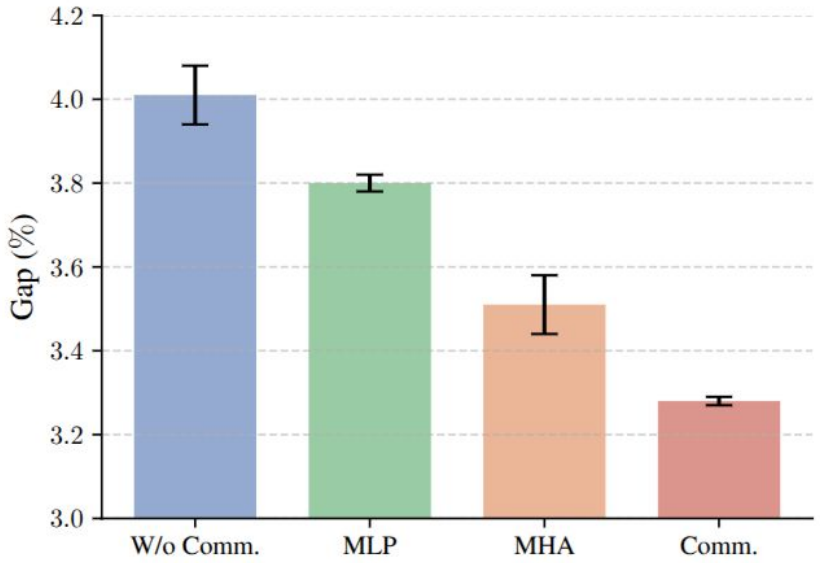
1. Communication Layers for effective coordination
2. Multiple Pointer Mechanism for parallel decoding
   a. What if agents select the same action? → We propose a Priority-based Conflict Handler based on confidence for tie-breaking



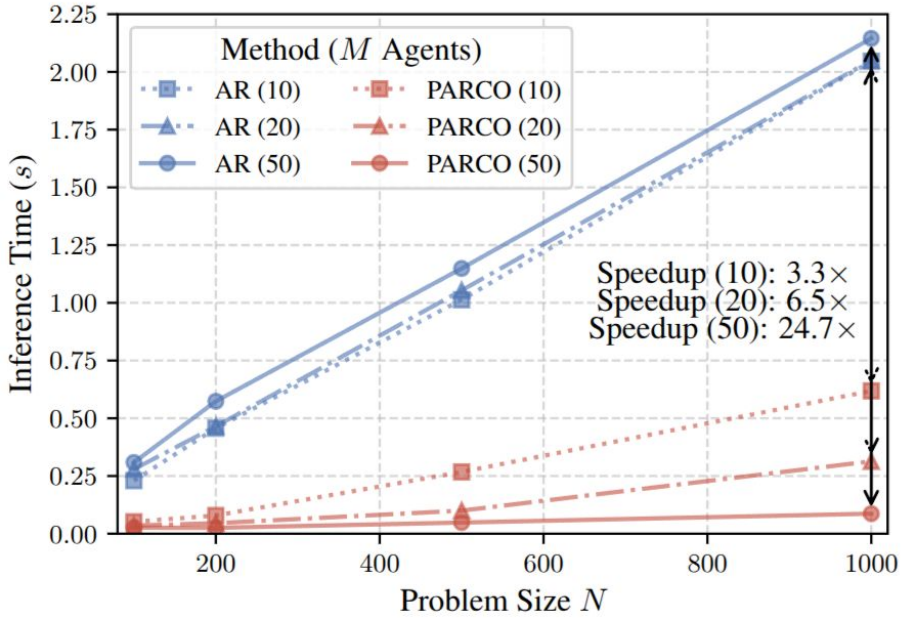
Results

AI4CO

★ SOTA results among neural solvers and vs traditional methods on ⅔ tasks (2x routing: HCVRP, OMDCPDP; 1x scheduling: FFSP)



★ Communication Layers enables much better performance



★ Parallel enables much better performance (more agents -> more speedup!)

NEURAL INFORMATION PROCESSING SYSTEMS

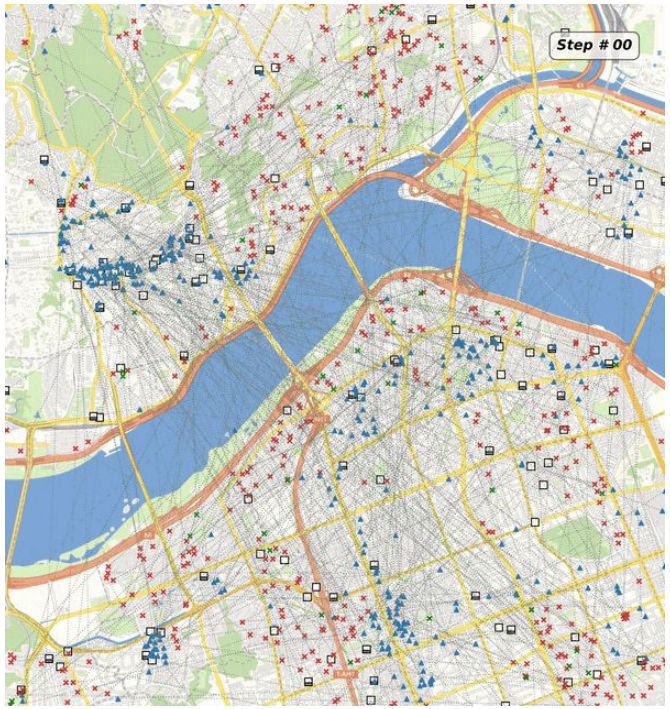
Results: scaling & generalization

AI4CO

★ PARCO can generalize to large-scale pickup-and-delivery problems with 50× larger number of nodes and agents than seen during training

Table 7: Large-scale generalization results for OMDCPDP with $N = 5000$.

	$M = 500$			$M = 750$			$M = 1,000$		
	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
OR-Tools	5575.73	134.06%	3600s	5127.46	115.24%	3600s	4974.81	188.10%	3600s
HAM	4813.99	102.08%	17.4s	3732.06	97.33%	19.5s	3258.26	88.69%	22.3s
PARCO	**2382.22**	**0.0%**	**0.21s**	**1891.28**	**0.0%**	**0.21s**	**1726.78**	**0.0%**	**0.22s**



→ SOTA results at 10,000x speedup compared to Google's OR-Tools!

NEURAL INFORMATION PROCESSING SYSTEMS

Thank you!

AI4CO

Star us on Github⭐

Join AI4CO 🤝

NEURAL INFORMATION PROCESSING SYSTEMS