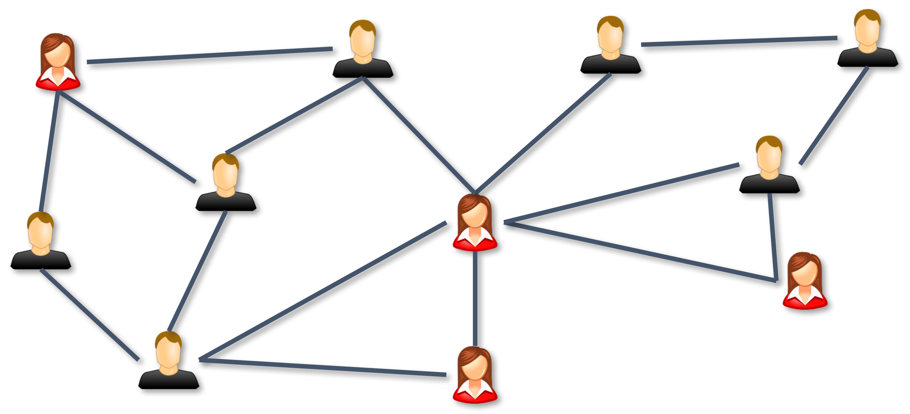
Learning Combinatorial Optimization Algorithms over Graphs

Hanjun Dai*, Elias B. Khalil*, Yuyu Zhang, Bistra Dilkina, Le Song (*equal contribution)

Georgia Institute of Technology

Background

Minimum Vertex Cover: NP-Complete Problems on Graphs

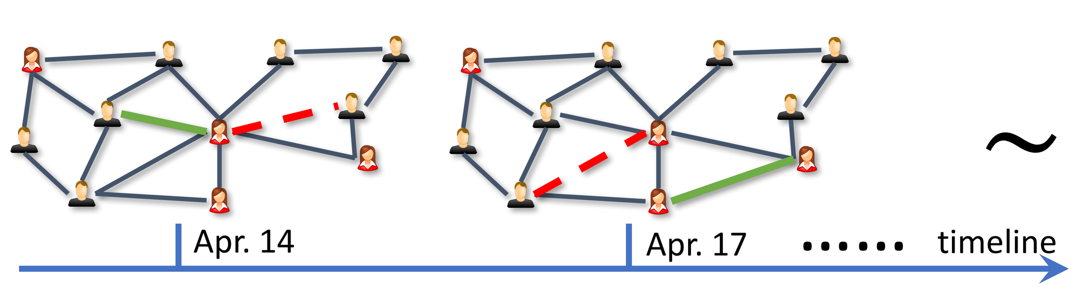


Definition:
Find smallest vertex subset S **s.t.**, S covers all the edges

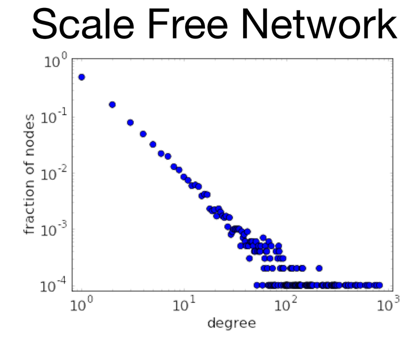
Application:
Advertising optimization in social networks

Tackling NPC problems	Design rationale	Example
Exact algorithms	Tight formulations, IP solvers	CPLEX
Approximation algorithms	Worst-case guarantees	Edge-picking
Heuristics	Empirical performance	Degree-greedy

Real-world: Same problem is solved repeatedly with slightly different data



Scale Free Network

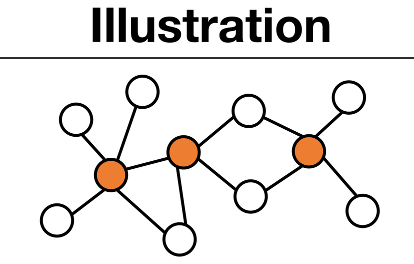
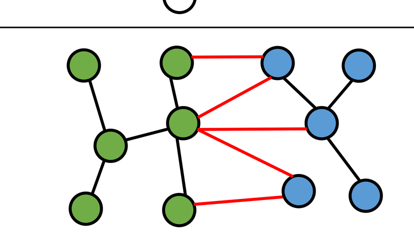
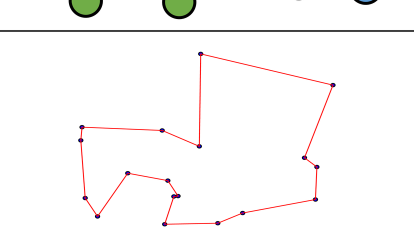


Can classical algorithms exploit the common distribution of instances?

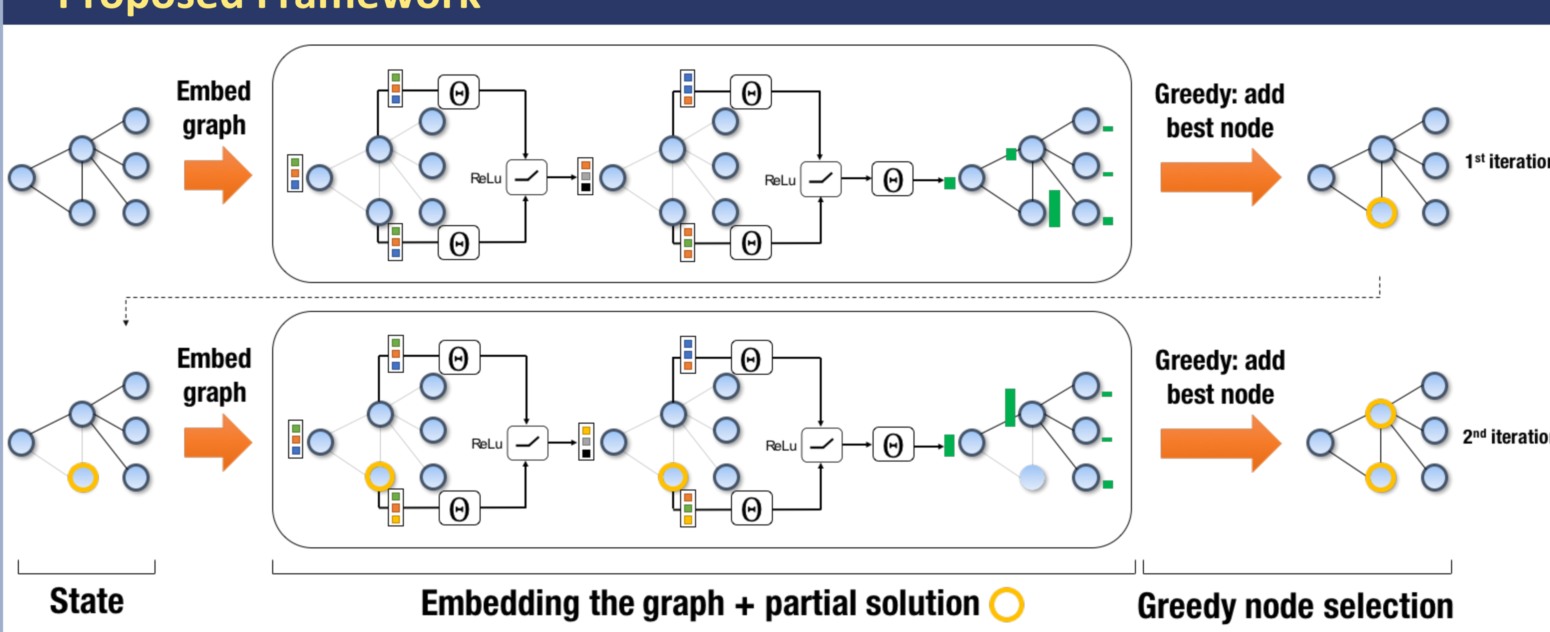
Not automatically! Typical approach: customize b.n.b./approx./heuristic

Problem Statement

Given a **graph optimization problem** G and a **distribution** $\mathcal{D}$ of problem instances, can we **learn better greedy heuristics** that generalize to unseen instances from $\mathcal{D}$?

Graph Opt. Prob.	Greedy Procedures	Illustration
Minimum Vertex Cover	Insert nodes into cover	
Maximum Cut	Insert nodes into subset	
Traveling Salesman Prob.	Insert nodes into sub-tour	

Proposed Framework



Reinforcement Learning Formulation

Minimum Vertex Cover

$$\min_{x_i \in \{0,1\}} \sum_{i \in \mathcal{V}} x_i$$

$$s.t.\ x_i + x_j \geq 1, \forall (i,j) \in \mathcal{E}$$

Reward: $r^t = -1$

State S: current selected nodes

Repeat until all edges are covered:

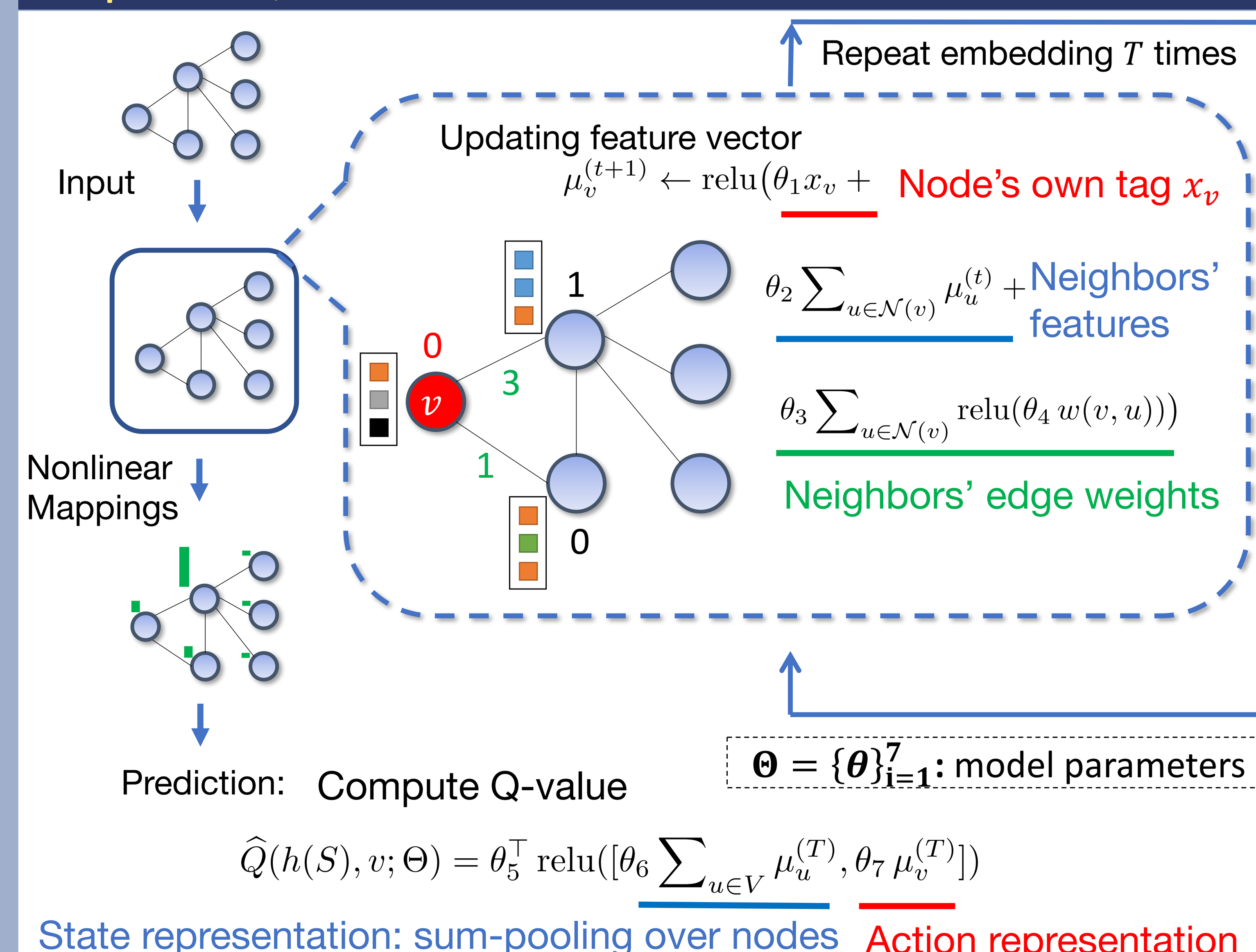
Action value function: $\widehat{Q}(S, v)$

1. Compute score for each vertex
2. Select vertex with largest score
3. Add best vertex to cover

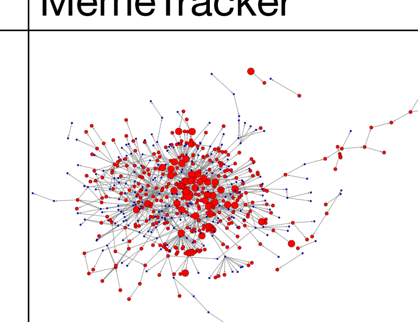
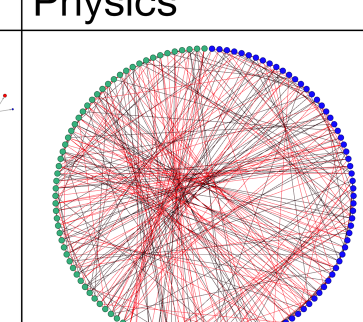
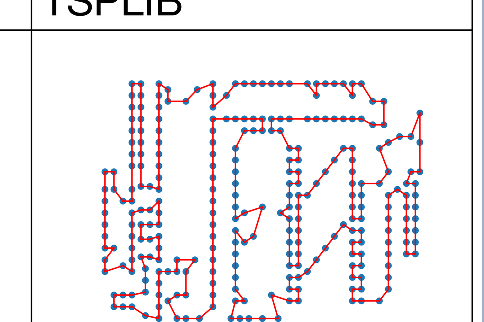
Greedy policy: $v^* = argmax_v \widehat{Q}(S, v)$

Update state S

Implement Q-function with structure2vec

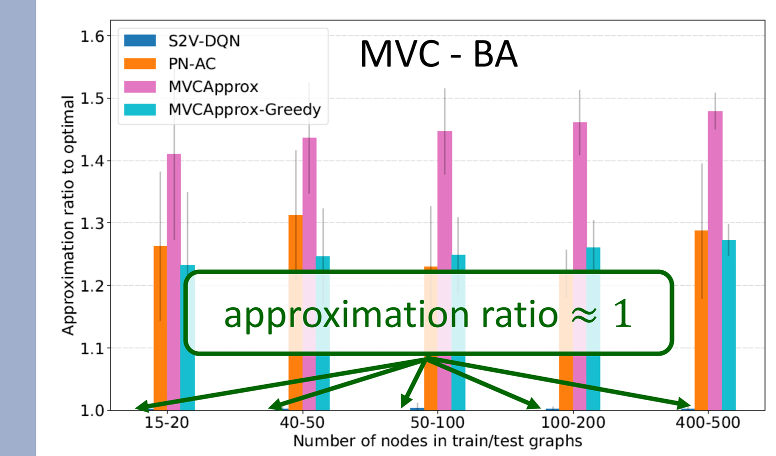


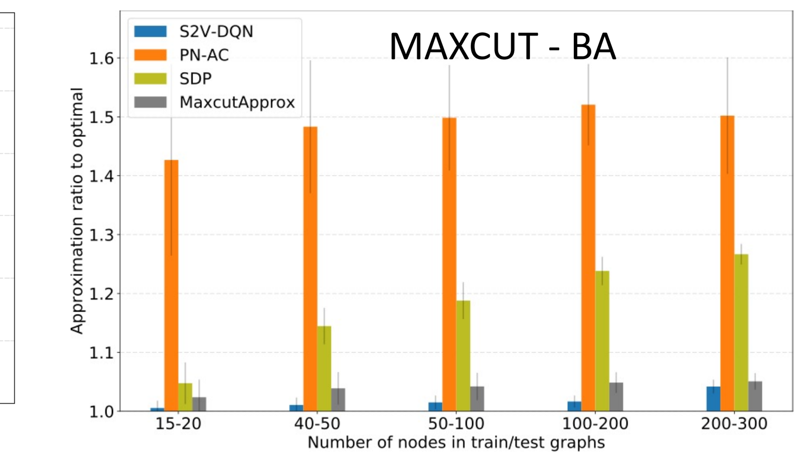
Experiment Settings

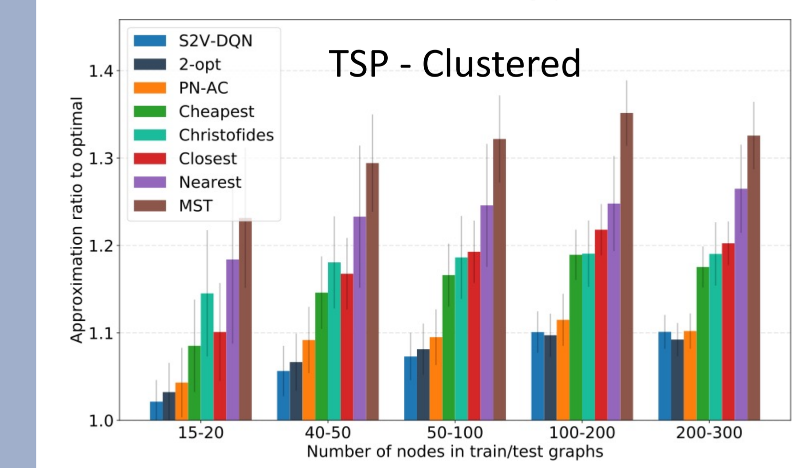
	Minimum Vertex Cover (MVC)	Maximum Cut (MAXCUT)	Traveling Salesman Problem (TSP)
Synthetic data			
Random gaphs	Erdos-Renyi (ER) or Barabasi-Albert (BA)	ER or BA	DIMACS generator; uniform grid or clustered
Statistics (# nodes)	training: 15 ~ 500 test: 15 ~ 1200	training: 15 ~ 300 test: 15 ~ 1200	training: 15 ~ 300 test: 15 ~ 1200
Real-world data			
Datasets	MemeTracker	Physics	TSPLIB
Visualization			
Statistics	1 graph, 960 nodes, 5000 edges	10 graphs, 125 nodes, 375 edges	38 graphs, 51 to 318 nodes
Opt obtained	ILP with CPLEX	IQP with CPLEX	Concorde

Experiments

Generalization on new random graphs with same distribution







Approximation ratio:

$$\mathcal{R}(S, G) = \max\left(\frac{OPT(G)}{c(h(S))}, \frac{c(h(S))}{OPT(G)}\right)$$

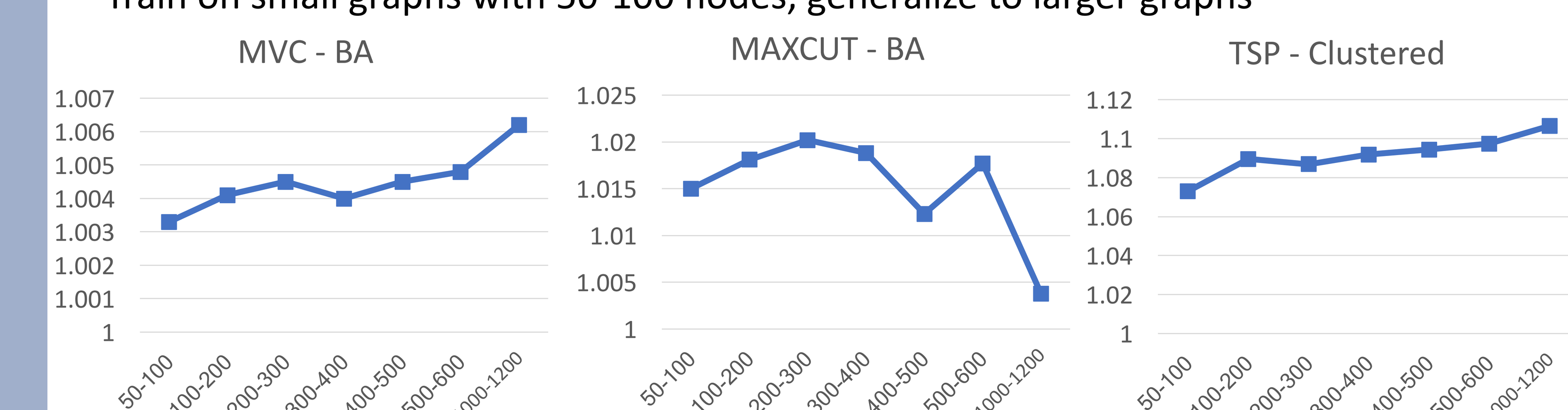
Lower the better

Methods:

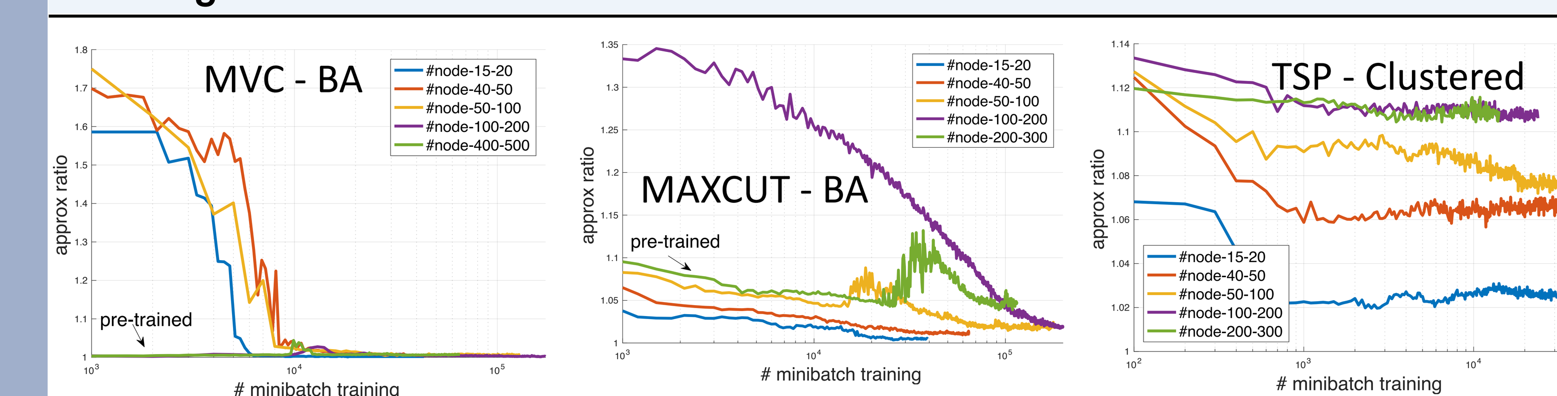
- S2V-DQN: our proposed algorithm
- PN-AC: Bello et.al. 2016
- Greedy heuristics;
- Approximation algorithms;

Generalization with different distribution

- Train on small graphs with 50-100 nodes, generalize to larger graphs



Convergence

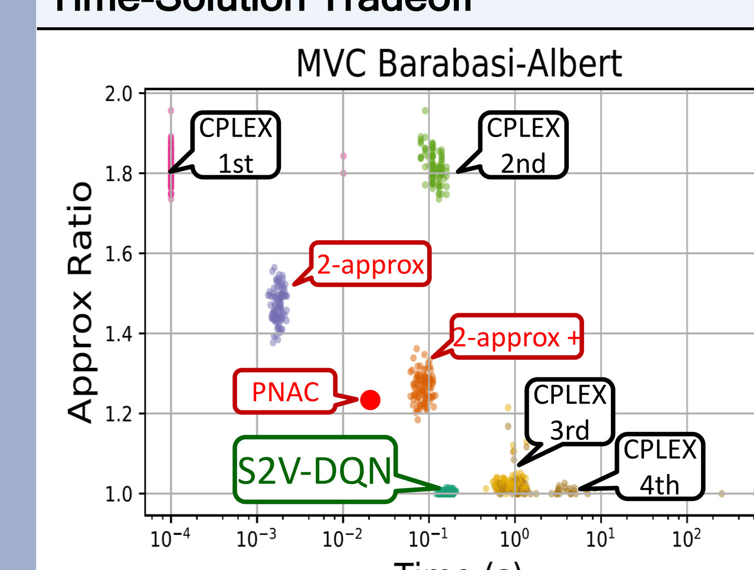


Experiments on real-world graphs

Table 3: Realistic data experiments, results summary. Values are average approximation ratios.

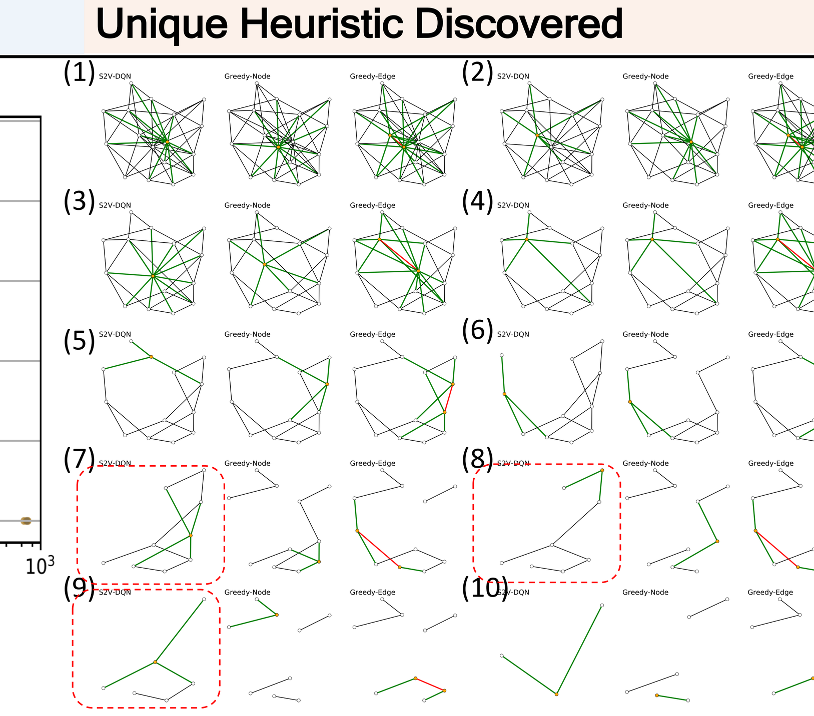
Problem	Dataset	S2V-DQN	Best Competitor	2nd Best Competitor
MVC	MemeTracker	**1.0021**	1.2220 (MVCApprox-Greedy)	1.4080 (MVCApprox)
MAXCUT	Physics	**1.0223**	1.2825 (MaxcutApprox)	1.8996 (SDP)
TSP	TSPLIB	**1.0475**	1.0947 (2-opt)	1.1771 (Cheapest)

Time-Solution Tradeoff



- Generate 200 Barabasi-Albert networks with 300 nodes
- Let CPLEX produces 1st, 2nd, 3rd, 4th feasible solutions

Unique Heuristic Discovered



Our method keeps the connectivity!

Mailto: {hanjun.dai, elias.khalil, yuyu.zhang, bdilkina, lsong}@cc.gatech.edu