Learning to Solve Combinatorial Optimization Problems on Real-World Graphs in Linear Time

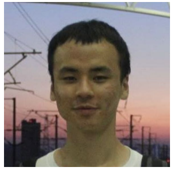

Iddo Drori [1,2,3]

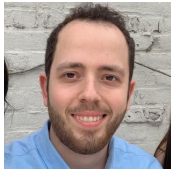

Anant Kharkar [3]


William Sickinger [3]


Brandon Kates [2]


Qiang Ma [3]


Suwen Ge [3]


Eden Dolev [3]


Brenda Dietrich [2]


David Williamson [2]


Madeleine Udell [2]

1 Massachusetts Institute of Technology

2 Cornell University

3 Columbia University in the City of New York

Example Problems Over Graphs

- Polynomial
  - Minimum Spanning Tree (MST)
  - Single-Source Shortest Paths (SSP)

- NP-hard
  - Traveling Salesman Problem (TSP)
  - Vehicle Routing Problem (VRP)

TSP Time Complexity

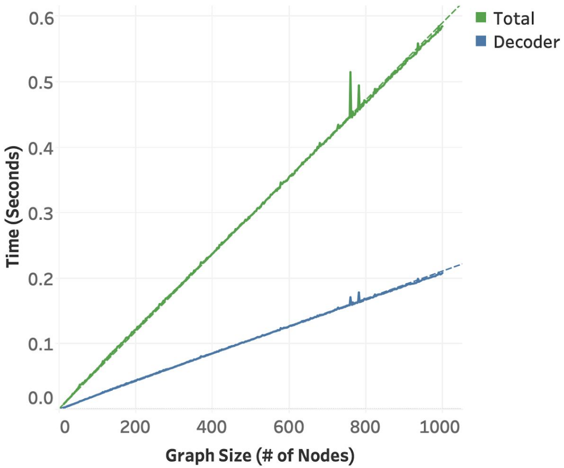
- NP-hard problem
- Approximation algorithms and optimality gaps
- Linear time approximation with optimality gap close to 1

Method	Runtime Complexity	Runtime (ms)	Speedup	Optimality Gap
Gurobi (Exact)	NA	$3,220$	$2,752.1$	1
Concorde (Exact)	NA	254.1	217.2	1
Christofides	$O(n^3)$	$5,002$	$4,275.2$	1.029
LKH	$O(n^{2.2})$	$2,879$	2460.7	1
2-opt	$O(n^2)$	30.08	25.7	1.097
Farthest	$O(n^2)$	8.35	7.1	1.075
Nearest	$O(n^2)$	9.35	8	1.245
S2V-DQN	$O(n^2)$	61.72	52.8	1.084
GPN	$O(n \log n)$	1.537	1.3	1.086
Ours	$O(n)$	1.17	1	1.074

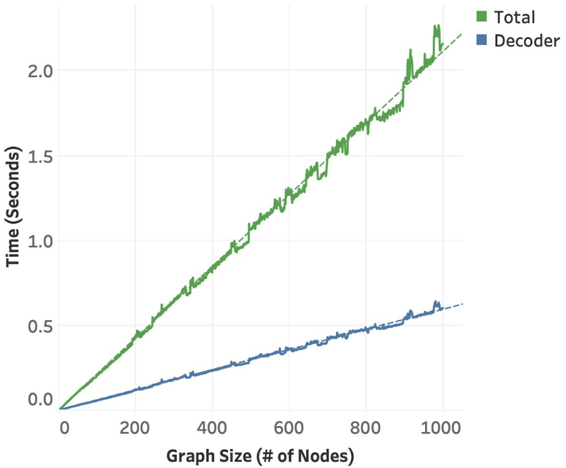
Massachusetts Institute of Technology

Running Time Complexity

- Graph attention is quadratic
- Attention approximation is linear and independent of data
- Practical GPU memory bottleneck



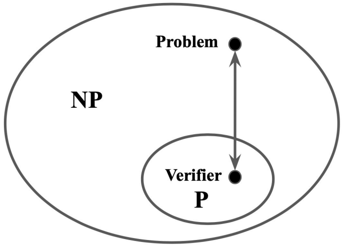
(a) MST



(b) TSP

Running Time Complexity

- Verification problems for NP-hard problems have polynomial time complexity.
- Polynomial vs. NP-hard problems:
  – Fast type-1 process: polynomial problems on graphs can be solved using GNN's without reinforcement learning or search
  – Slow type-2 process: NP-hard problems require RL or search
- GNN's can be used directly for verification



Minimum Spanning Tree (MST)

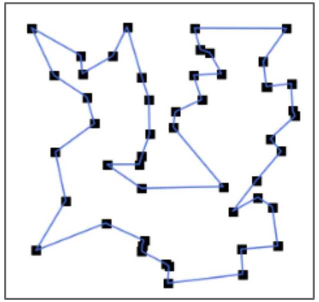
- Given connected and undirected graph G = (V, E, W)
- Find tree T = (V_T, E_T) with V_T = V, $E_T \subset$ E minimizing sum of edge weights $W_T \subset$ W.
- Greedy algorithms with time complexity O(|E|log|V|): Boruvka, Prim, Kruskal

Single-Source Shortest Paths (SSP)

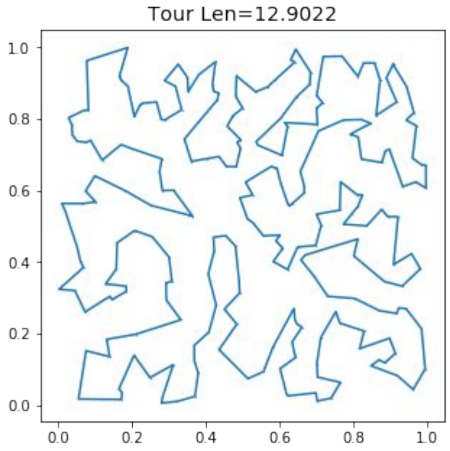
- Given connected and directed graph G = (V, E, W) and source vertex.
- Find shortest paths from source to all other vertices.
- For SSP with nonnegative weights: Dijkstra's algorithm complexity O(|V|log|V| + |E|) using a heap.
- For general SSP: Bellman-Ford runs in O(|V||E|).
- Floyd–Warshall algorithm solves SSP between all pairs of nodes with cubic time complexity O(|V|^3)

Traveling Salesman Problem (TSP)

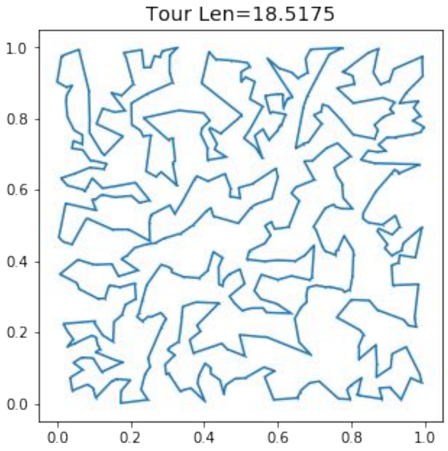
- Graph G = (V, E, W)
- V represents list of cities
- W represents distances between each pair of cities.
- Find shortest tour visiting each city once and returns to start.
- NP-hard problem.



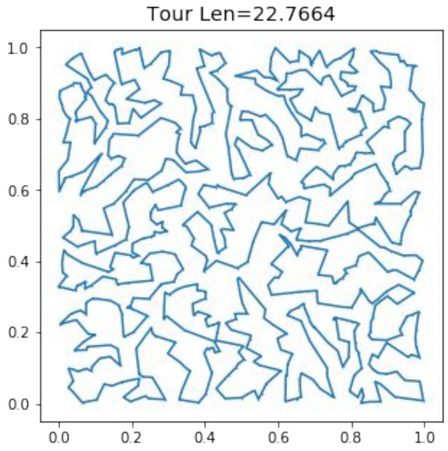
Examples



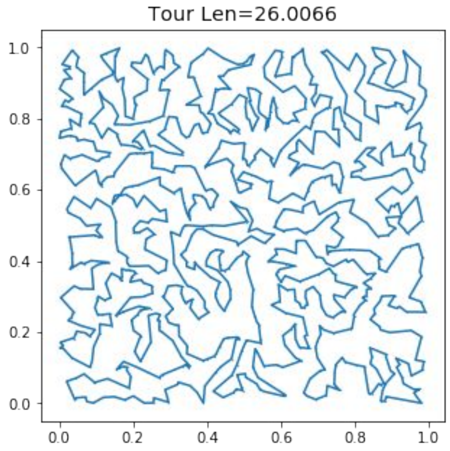
TSP250



TSP500



TSP750



TSP1000

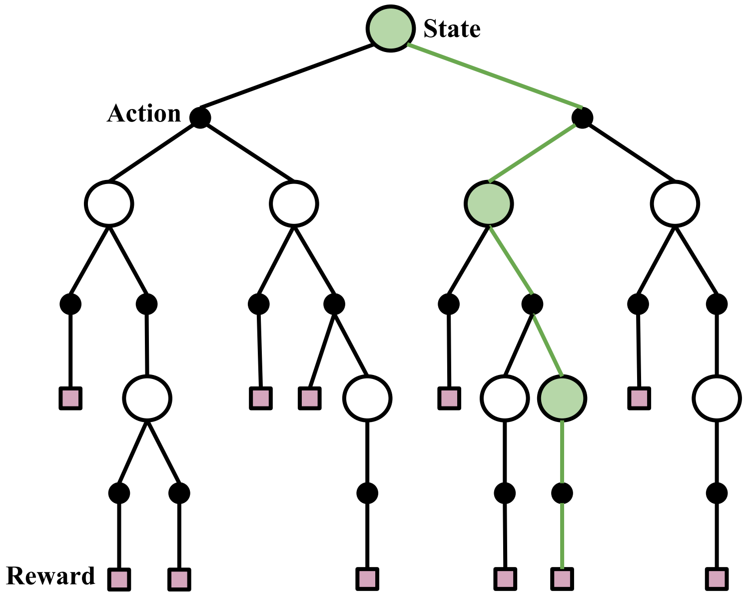
Vehicle Routing Problem (VRP)

- Given M vehicles and graph G = (V, E) with |V| cities
- Find optimal routes for vehicles.
- Each vehicle m ∈ {1, .., M} starts from same depot node, visits subset V(m) of cities, and returns to depot node.
- Routes of different vehicles do not intersect except at depot; together, the vehicles visit all cities.
- Optimal routes minimize longest tour length of any single route.
- TSP is special case of VRP for one vehicle.

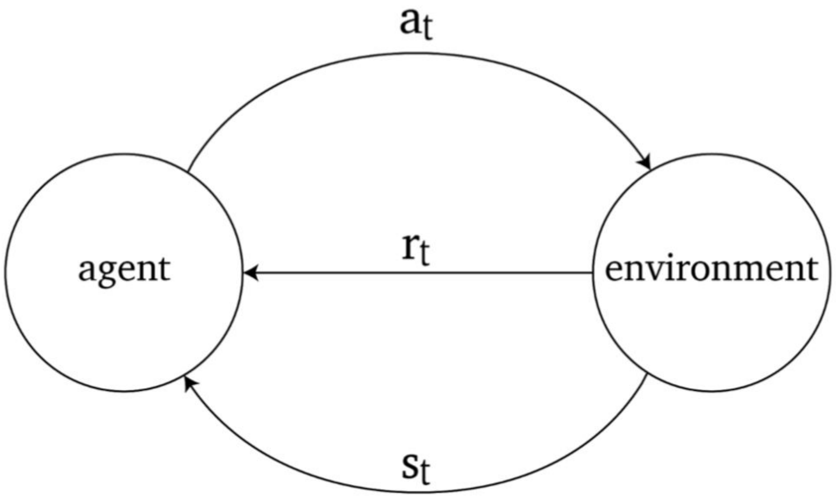
Learning Graph Algorithms as Single Player Games

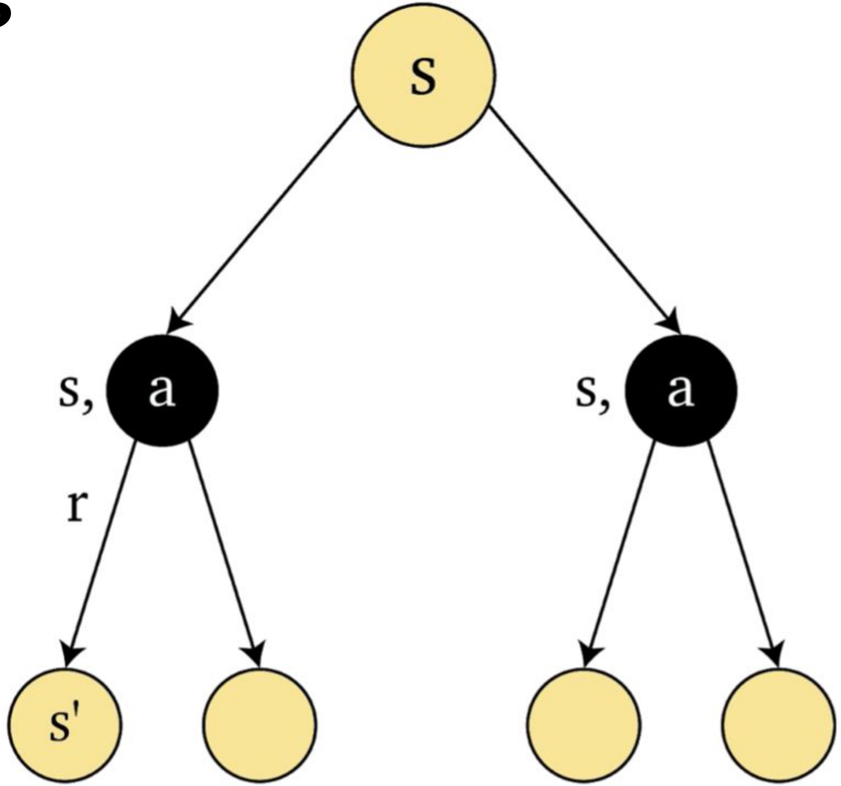
- Represent problem space as search tree.
- Leaves of search tree represent all (possibly exponentially many) possible solutions to problem.
- Search traverses tree, choosing path (guided by MCTS+NN)

Learning Graph Algorithms as Single Player Games



Reinforcement Learning

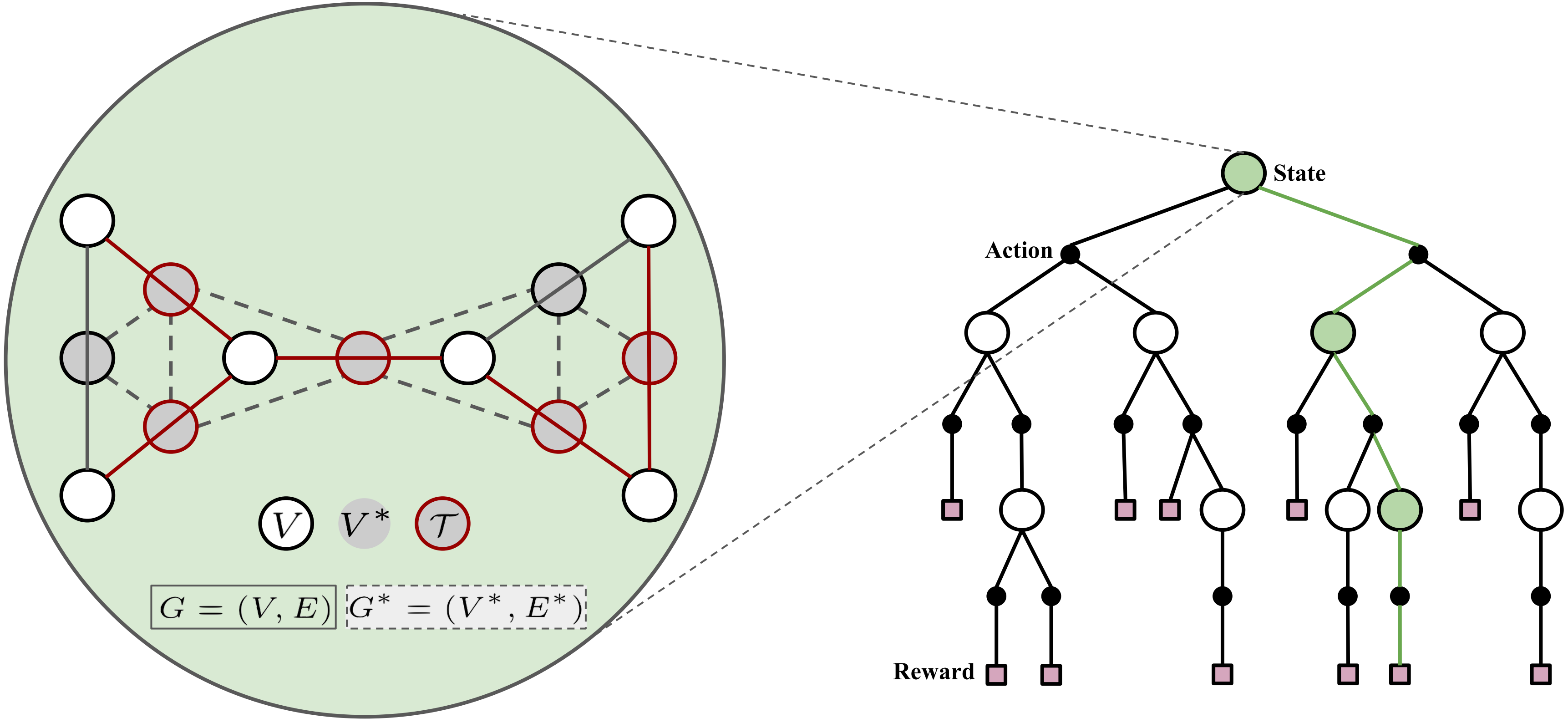




Learning Graph Algorithms as Single Player Games

- Initial state: represented by root node, may be empty set, a random state, or other initial state.
- Each path from root to a leaf consists of moving between nodes (states) along edges (taking actions) reaching a leaf node (reward).
- Actions: adding or removing a node or edge.
- Reward (or cost): value of solution, for example sum of weights or length of tour.

State



Actions

- Add or remove node or edge

Line Graph

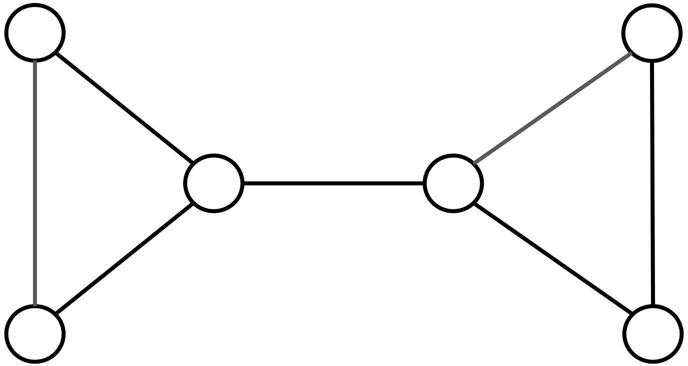
- Problem: problems involve both actions on nodes and edges.

- Solution: use edge-to vertex dual (line graph)

  Perform actions on nodes.

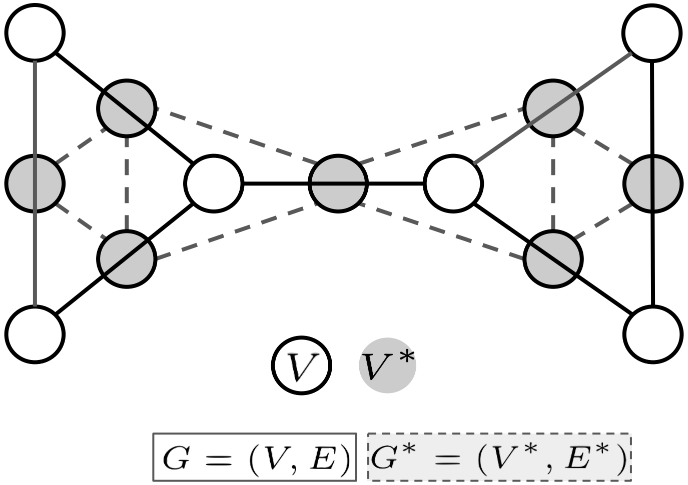
Graph

- G = (V, E)



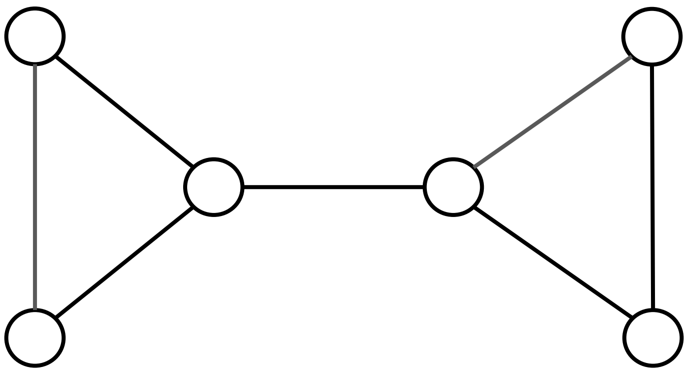
Line Graph: Edge-to-Vertex Dual

- Each edge in primal graph corresponds to node in line graph



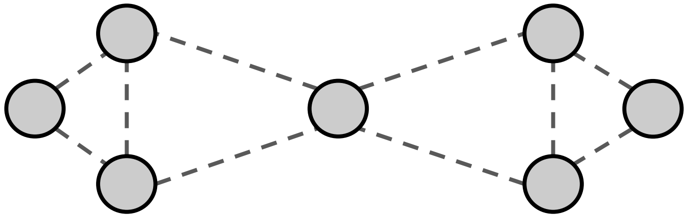
Graph

- Edges in primal graph



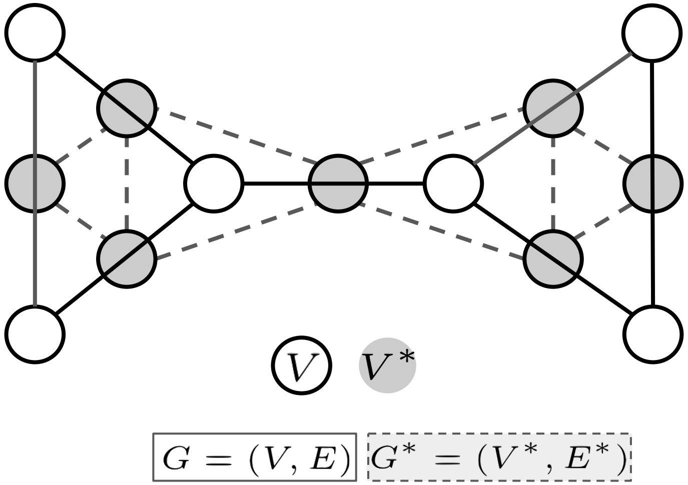
Line Graph: Edge-to-Vertex Dual

- Correspond to nodes in line graph

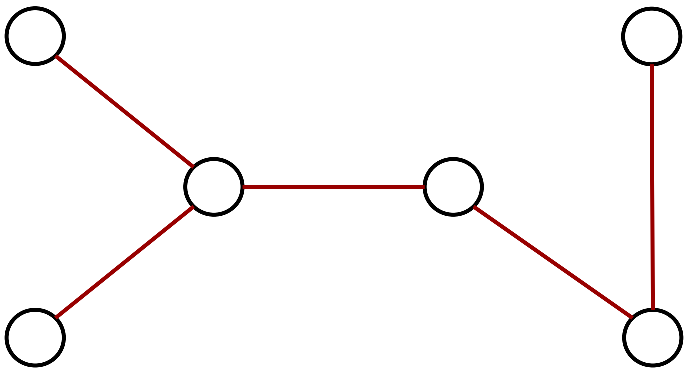


Line Graph: Edge-to-Vertex Dual

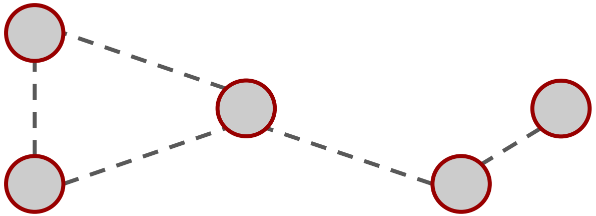
- Two nodes in line graph are connected if corresponding edges in primal graph share a node.
- Edge weights in primal graph become node weights in line graph.



Tree

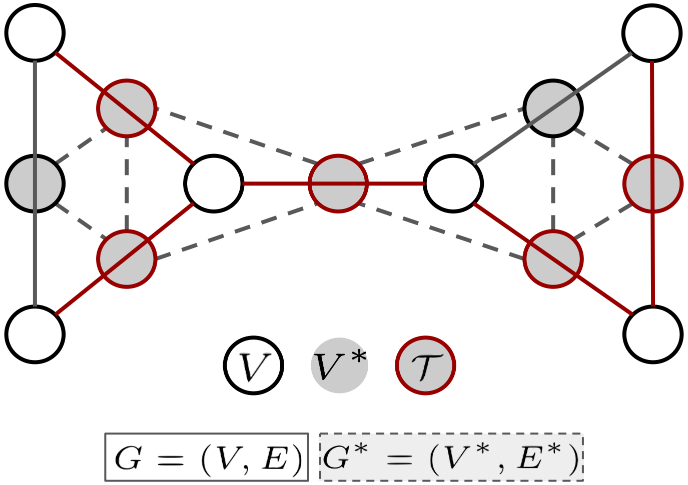


Line Graph

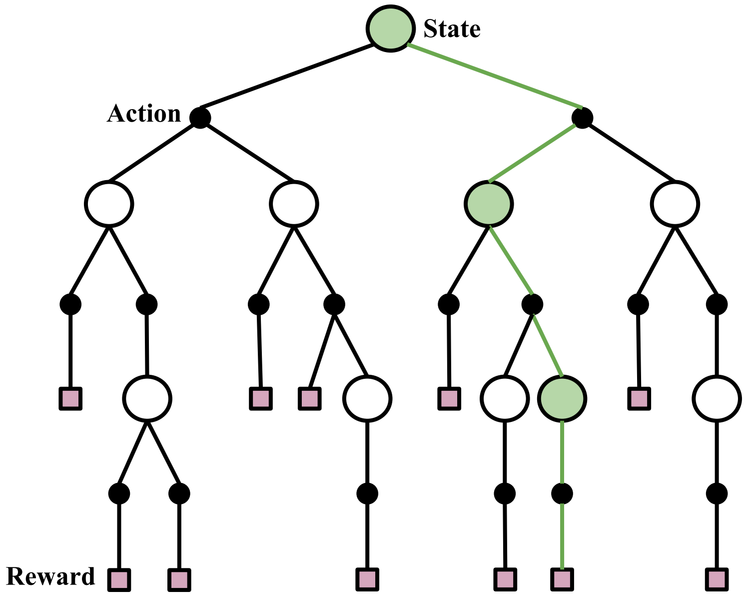


Line Graph: Edge-to-Vertex Dual

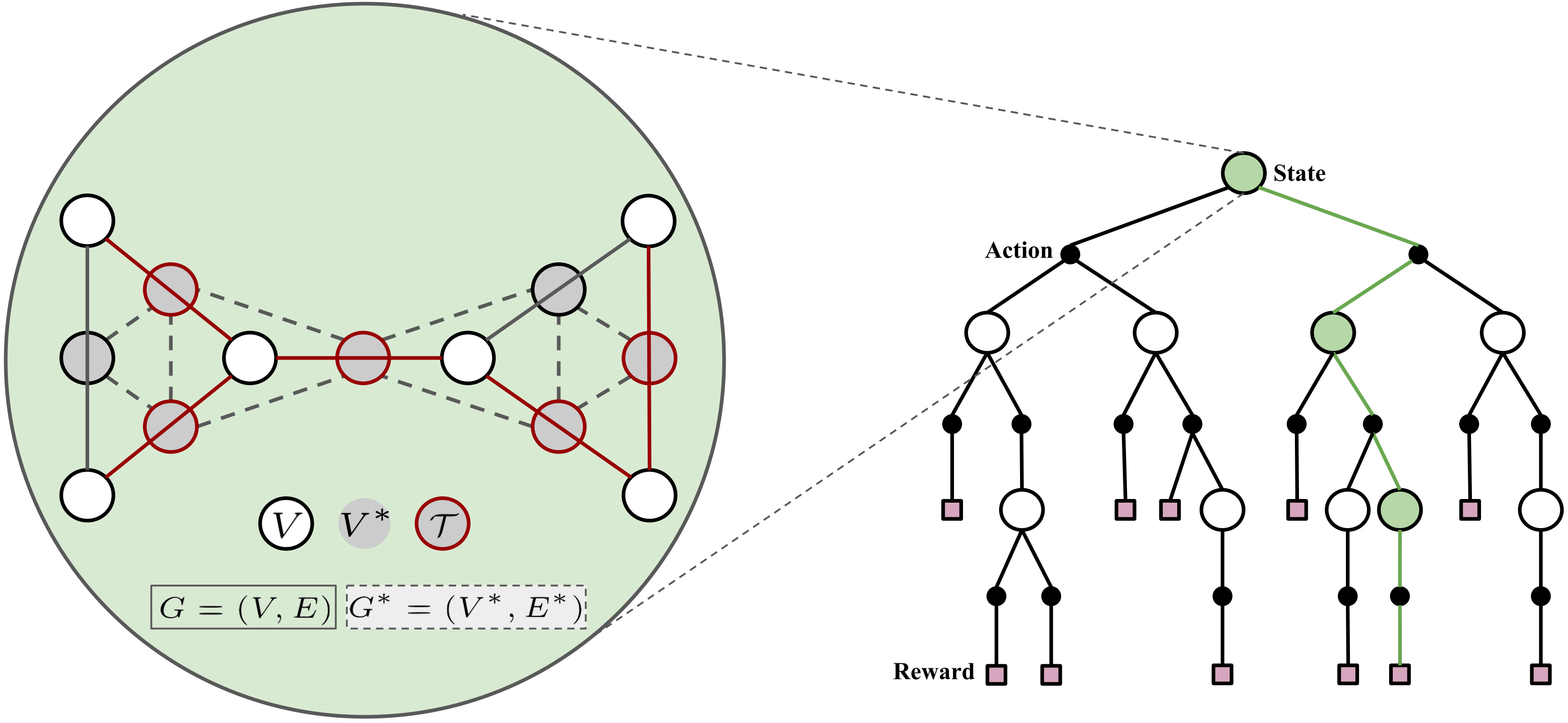
- Two nodes in line graph are connected if corresponding edges in primal graph share a node.
- Edge weights in primal graph become node weights in line graph.



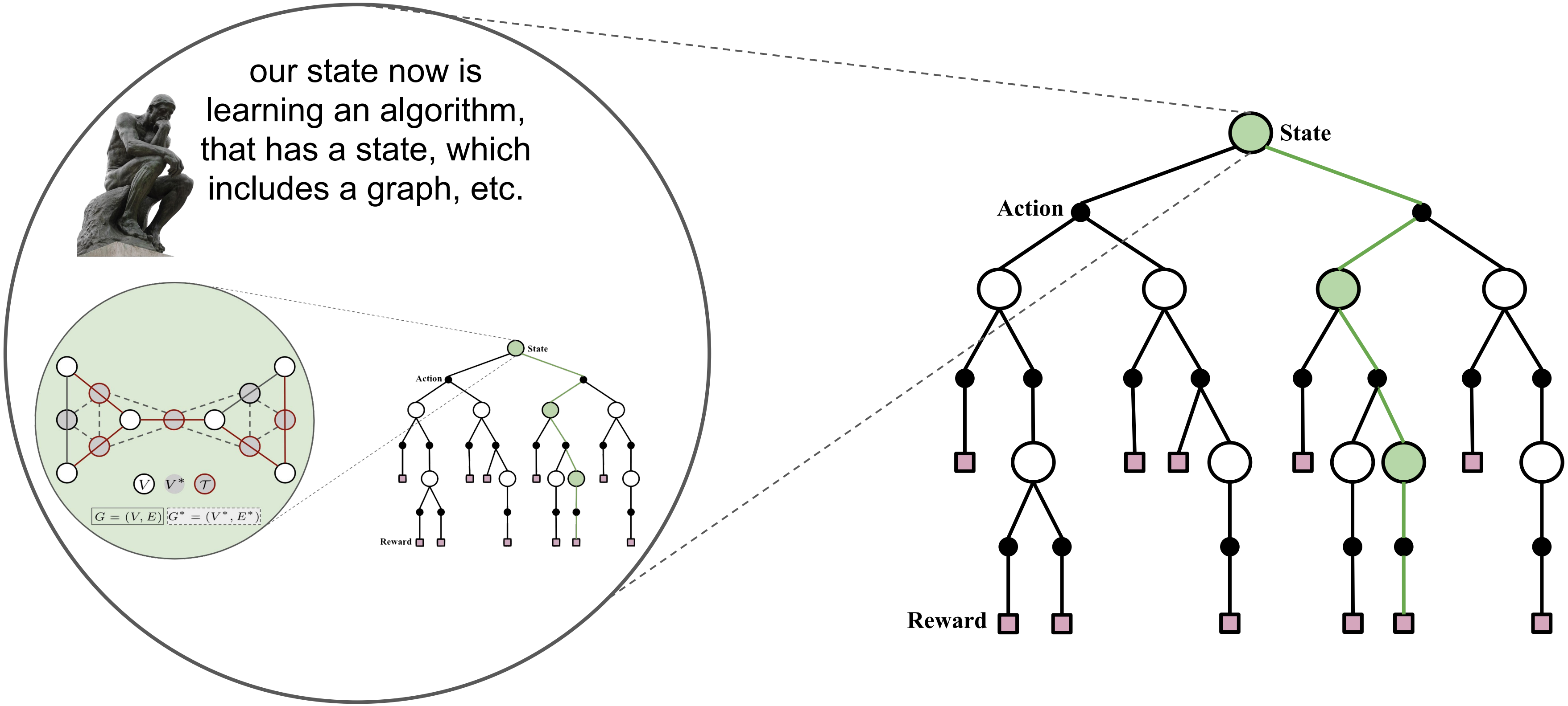
Learning Graph Algorithms as Single Player Games



State



Food for Thought: Learning to Learn to Learn..



Actions

- Add/remove nodes/edges.

Problem	State	Action	Reward
MST	$G = (V, E), G^* = (V^*, E^*), W, \mathcal{T}$	$\mathcal{T} = \mathcal{T} \cup \{e\}$	$-\left(I(\mathcal{T}) + \sum_{e \in E_\pi} W(e)\right)$ (Eq. 4)
SSP	$G = (V, E), G^* = (V^*, E^*), W, \mathcal{Q}_i$	$\mathcal{Q}_i = \mathcal{Q}_i \cup \{e\}$	$-\sum_{i=1}^{\lvert V \rvert} \left(I(\mathcal{Q}_i) + \sum_{e \in \mathcal{Q}_i} W(e)\right)$
TSP	$G = (V, E), \bar{V} = \{\tau(1), .., \tau(i)\}$	$\bar{V} = \bar{V} \cup \{\tau(i+1)\}$	$-\sum_{i=1}^{\lvert V \rvert} \lVert \mathbf{v}_{\tau(i)} - \mathbf{v}_{\tau(i+1)} \rVert_2$ (Eq. 5)
VRP	$G = (V, E), \bar{V}_m = \{\tau(d), \tau_m(2), .., \tau_m(i)\}, M$	$\bar{V}_m = \bar{V}_m \cup \{\tau_m(i+1)\}$	$-\max_{m \in \{1,..,M\}} \left\{ \sum_{i \in V_{\mathcal{T}}(m)} \lVert \mathbf{v}_{\tau_m(i)} - \mathbf{v}_{\tau_m(i+1)} \rVert_2 \right\}$

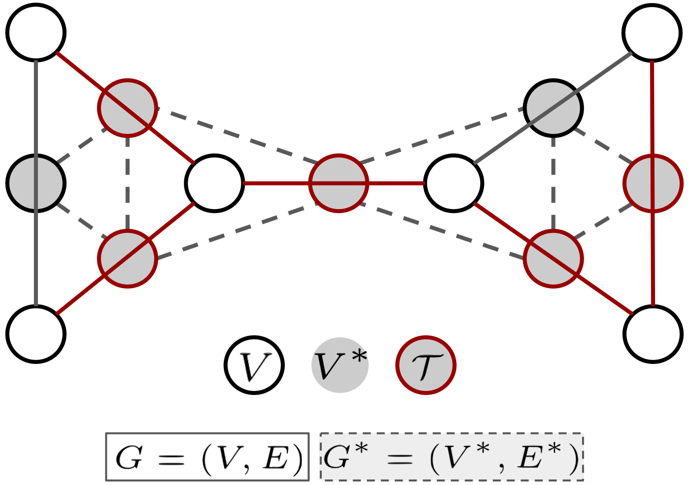
Reward

- Objective function

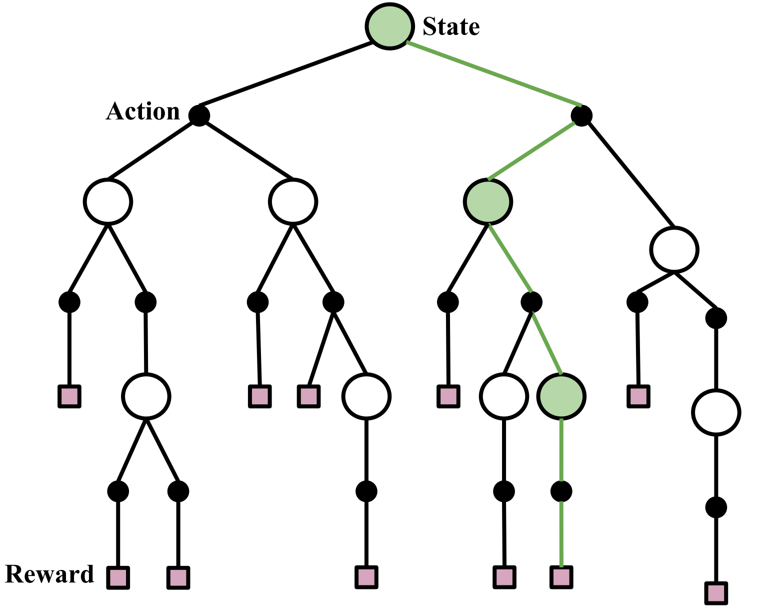
Problem	State	Action	Reward
MST	$G=(V,E), G^*=(V^*,E^*), W, \mathcal{T}$	$\mathcal{T}=\mathcal{T}\cup\{e\}$	$-\left(I(\mathcal{T})+\sum_{e\in E_\pi} W(e)\right)$ (Eq. 4)
SSP	$G=(V,E), G^*=(V^*,E^*), W, \mathcal{Q}_i$	$\mathcal{Q}_i=\mathcal{Q}_i\cup\{e\}$	$-\sum_{i=1}^{\|V\|}\left(I(\mathcal{Q}_i)+\sum_{e\in\mathcal{Q}_i} W(e)\right)$
TSP	$G=(V,E), \bar{V}=\{\tau(1),..,\tau(i)\}$	$\bar{V}=\bar{V}\cup\{\tau(i+1)\}$	$-\sum_{i=1}^{\|V\|}\\|\mathbf{v}_{\tau(i)}-\mathbf{v}_{\tau(i+1)}\\|_2$ (Eq. 5)
VRP	$G=(V,E), \bar{V}_m=\{\tau(d),\tau_m(2),..,\tau_m(i)\}, M$	$\bar{V}_m=\bar{V}_m\cup\{\tau_m(i+1)\}$	$-\max_{m\in\{1,..,M\}}\left\{\sum_{i\in V_\mathcal{T}(m)}\\|\mathbf{v}_{\tau_m(i)}-\mathbf{v}_{\tau_m(i+1)}\\|_2\right\}$

Unified Framework

A



B



C

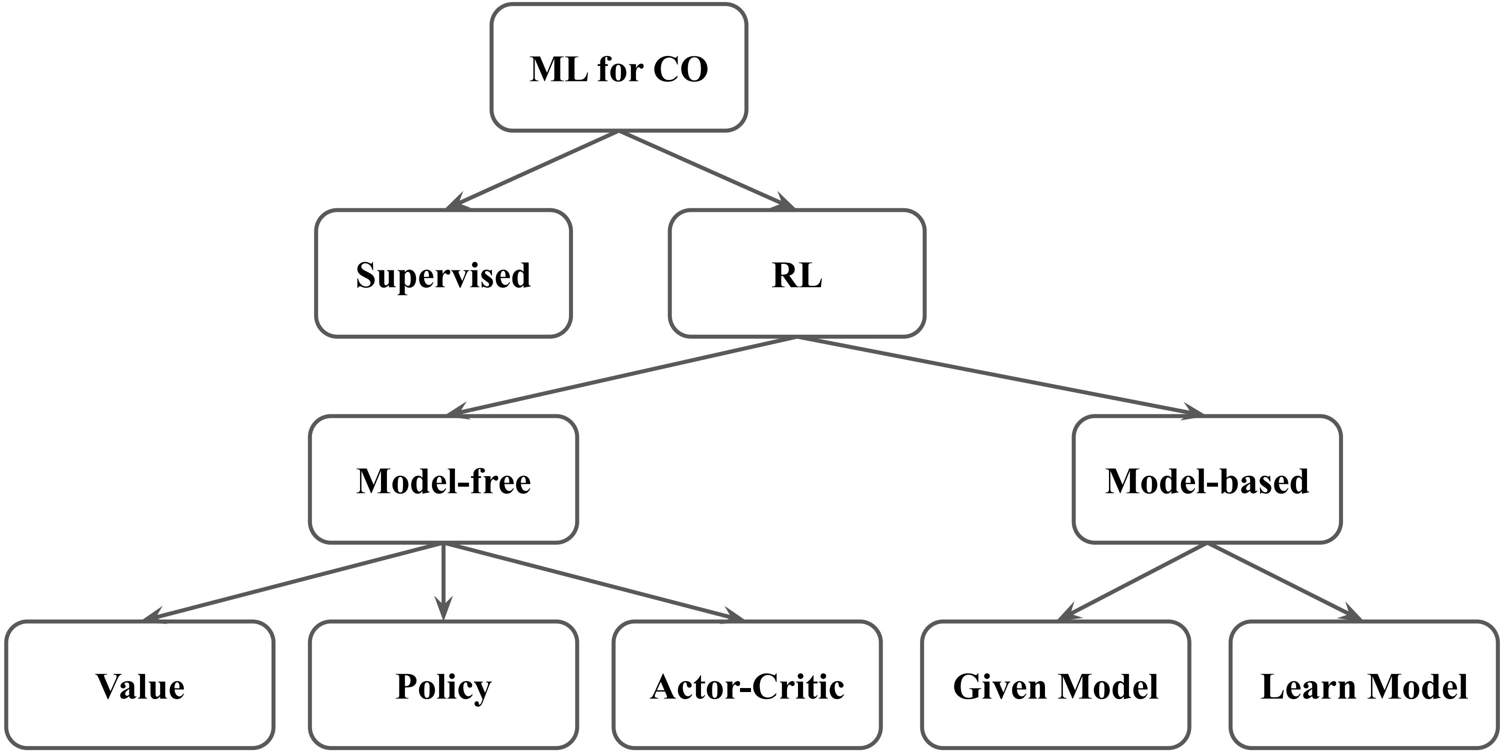
Problem	State	Action	Reward
MST	$G = (V, E), G^* = (V^*, E^*), W, \mathcal{T}$	$\mathcal{T} = \mathcal{T} \cup \{e\}$	$-\left(I(\mathcal{T}) + \sum_{e \in E_\pi} W(e)\right)$ (Eq. 4)
SSP	$G = (V, E), G^* = (V^*, E^*), W, \mathcal{Q}_i$	$\mathcal{Q}_i = \mathcal{Q}_i \cup \{e\}$	$-\sum_{i=1}^{\lvert V \rvert} \left(I(\mathcal{Q}_i) + \sum_{e \in \mathcal{Q}_i} W(e)\right)$
TSP	$G = (V, E), \bar{V} = \{\tau(1), .., \tau(i)\}$	$\bar{V} = \bar{V} \cup \{\tau(i+1)\}$	$-\sum_{i=1}^{\lvert V \rvert} \lVert \mathbf{v}_{\tau(i)} - \mathbf{v}_{\tau(i+1)} \rVert_2$ (Eq. 5)
VRP	$G = (V, E), \bar{V}_m = \{\tau(d), \tau_m(2), .., \tau_m(i)\}, M$	$\bar{V}_m = \bar{V}_m \cup \{\tau_m(i+1)\}$	$-\max_{m \in \{1, .., M\}} \left\{ \sum_{i \in V_\tau(m)} \lVert \mathbf{v}_{\tau_m(i)} - \mathbf{v}_{\tau_m(i+1)} \rVert_2 \right\}$

Machine Learning for Combinatorial Optimization

- Rapidly growing field
- Leading architecture
  – Outer loop: RL / search
  – Inner loop: GNN's

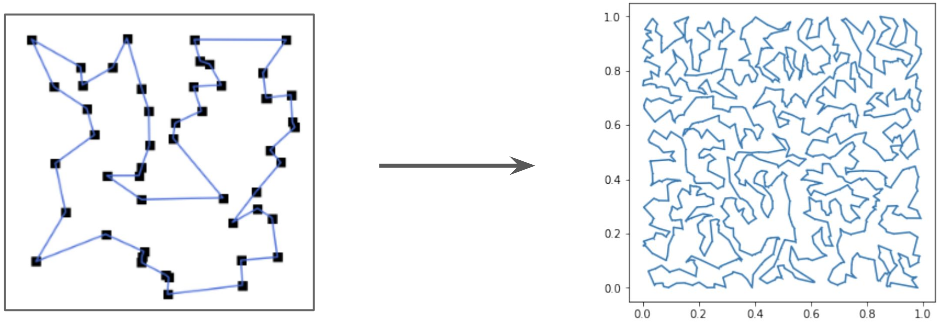
NP-hard Problem	Method	Type
Towers of Hanoi	AlphaZero: Recursive MCTS + LSTM [52]	Model-based, Given model
Integer Programming	RL + LSTM [62]	Model-free, Policy-based
Minimum Dominating Set (MDS)	RL + Decision Diagram [13]	Model-free, Value-based
	RL + GNN [72]	Model-free, Policy-based
Maximum Common Subgraph (MCS)	DQN + GNN [6]	Model-free, Value-based
Maximum Weight Matching (MWM)	DDPG [23]	Model-free, Policy-based
Boolean Satisfiability (SAT)	MPNN [57]	Supervised, Approximation
	RL + GNN [72]	Model-free, Policy-based
	Tree search + GCN [39]	Model-based, Given model
Graph Coloring	RL + GNN [72]	Model-free, Policy-based
	AlphaZero: MCTS + LSTM [30]	Model-based, Given model
Maximum Clique (MC)	RL + GNN [72]	Model-free, Policy-based
	Tree search + GCN [39]	Model-based, Given model
Maximum Independent Set (MIS)	Tree search + GCN [39]	Model-based, Given model
	AlphaZero: MCTS + GCN [2]	Model-based, Given model
Minimum Vertex Cover (MVC)	Q-Learning + GNN [19]	Model-free, Value-based
	DQN, Imitation learning [59]	Model-free, Value-based
	RL + GNN [72]	Model-free, Policy-based
	Tree search + GCN [39]	Model-based, Given model
Maximum Cut (MaxCut)	Q-Learning + GNN [19]	Model-free, Value-based
	DQN + MPNN [8]	Model-free, Value-based
	PPO + CNN, GRU [11]	Model-free, Actor-Critic
Traveling Salesman Problem (TSP)	Pointer network [66]	Supervised, Approximation
	GCN + Search [31]	Supervised, Approximation
	Q-Learning + GNN [19]	Model-free, Value-based
	Hierarchical RL + GAT [44]	Model-free, Policy-based
	REINFORCE + LSTM with attention [47]	Model-free, Policy-based
	REINFORCE + attention [20]	Model-free, Policy-based
	RL + GAT [36]	Model-free, Policy-based
	DDPG [23]	Model-free, Policy-based
	REINFORCE + Pointer network [10]	Model-free, Policy-based
	RL + NN [45]	Model-free, Actor-Critic
	RL + GAT [14]	Model-free, Actor-Critic
	AlphaZero: MCTS + GCN [51]	Model-based, Given model
Knapsack Problem	REINFORCE + Pointer network [10]	Model-free, Policy-based
Bin Packing Problem (BPP)	REINFORCE + LSTM [29]	Model-free, Policy-based
	AlphaZero: MCTS + NN [38]	Model-based, Given model
Job Scheduling Problem (JSP)	RL + LSTM [16]	Model-free, Actor-Critic
Vehicle Routing Problem (VRP)	REINFORCE + LSTM with attention [47]	Model-free, Policy-based
	RL + LSTM [16]	Model-free, Policy-based
	RL + GAT [36]	Model-free, Policy-based
	RL + NN [43]	Model-free, Policy-based
	RL + GAT [25]	Model-free, Actor-Critic
Global Routing	DQN + MLP [40]	Model-free, Value-based
Highest Safe Rung (HSR)	AlphaZero: MCTS + CNN [71]	Model-based, Given model

ML Approaches for NP-Hard Combinatorial Optimization Problems



Generalization on Graphs

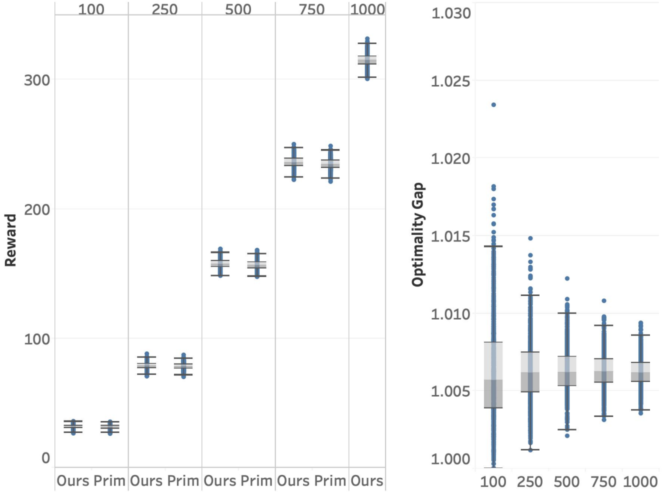
1. From small to large graphs
2. Between different types of random graphs
3. From random to real-world graphs



Massachusetts Institute of Technology

Generalization on Graphs

- From small to large random regular graphs
- Training on 100 node graphs
- Testing on 100/250/500/750/1000 node graphs

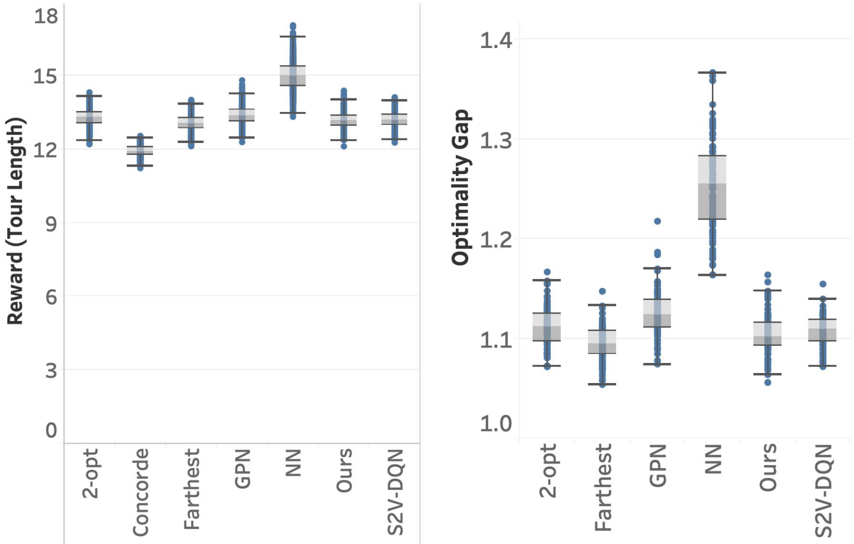


(a) MST Weights (b) MST Opt. Gaps

Massachusetts Institute of Technology

Generalization on Graphs

- From small to large random regular graphs
- Trained on graphs with 100 nodes, tested on 250 nodes



(a) TSP Tour Length (b) TSP Opt. Gaps

Generalization on Graphs

- From random graphs to real world graphs
- Trained on random Euclidean graphs with 100 nodes

TSPLIB Instance	Exact Concorde	Ours	RL GPN	S2V-DQN	Farthest	Approx. 2-opt	Nearest
eil51	426	**439**	485	**439**	448	452	514
berlin52	7,542	**7,681**	8,795	7,734	8,121	7,778	8,981
st70	675	**684**	701	685	729	701	806
eil76	538	**555**	591	558	583	597	712
pr76	108,159	112,699	118,032	**111,141**	119,649	125,276	153,462
rat99	1,211	1,268	1,472	**1,250**	1,319	1,351	1,565
kroA100	21,282	**21,452**	24,806	22,335	23,374	23,306	26,856
kroB100	22,141	**22,488**	24,369	22,548	24,035	23,129	29,155
kroC100	20,749	**21,427**	24,780	21,468	21,818	22,313	26,327
kroD100	21,294	**21,555**	23,494	21,886	22,361	22,754	26,950
kroE100	22,068	**22,267**	23,467	22,820	23,604	25,325	27,587
rd100	7,910	**8,243**	8,844	8,305	8,652	8,832	9,941
eil101	629	**650**	704	667	687	694	825
lin105	14,379	**14,571**	15,795	14,895	15,196	16,184	20,363
pr107	44,303	44,854	55,087	**44,780**	45,573	46,505	48,522
pr124	59,030	**59,729**	67,901	61,101	61,645	61,595	69,299
bier127	118,282	**120,672**	134,089	123,371	127,795	136,058	129,346
ch130	6,110	**6,208**	6,457	6,361	6,655	6,667	7,575
pr136	96,772	**98,957**	110,790	100,185	104,687	103,731	120,778
pr144	58,537	60,492	67,211	**59,836**	62,059	62,385	61,651
ch150	6,528	**6,729**	7,074	6,913	6,866	7,439	8,195
kroA150	26,524	**27,419**	30,260	28,076	28,789	28,313	33,610
kroB150	26,130	27,165	29,141	**26,963**	28,156	28,603	32,825
pr152	73,682	79,326	85,331	75,125	**75,209**	77,387	85,703
u159	42,080	43,687	52,642	45,620	46,842	**42,976**	53,637
rat195	2,323	**2,384**	2,686	2,567	2,620	2,569	2,762
d198	15,780	17,754	19,249	16,855	**16,161**	16,705	18,830
kroA200	29,368	**30,553**	34,315	30,732	31,450	32,378	35,798
kroB200	29,437	**30,381**	33,854	31,910	31,656	32,853	36,982
ts225	126,643	**130,493**	147,092	140,088	140,625	143,197	152,494
tsp225	3,916	4,091	4,988	4,219	4,233	**4,046**	4,748
Mean Opt. Gap	1	**1.032**	1.144	1.045	1.074	1.087	1.238

Generalization on Graphs

- From small random graphs to large real world graphs
- Trained on random Euclidean graphs with 100 nodes

TSPLIB Instance	Exact Concorde	Ours	RL GPN	S2V-DQN	Farthest	Approx. 2-opt	Nearest
pr226	80,369	86,438	85,186	**82,869**	84,133	85,306	94,390
gil262	2,378	**2,523**	5,554	2,539	2,638	2,630	3,218
pr264	49,135	**52,838**	67,588	53,790	54,954	58,115	58,634
a280	2,579	**2,742**	3,019	3,007	3,011	2,775	3,311
pr299	48,191	53,371	68,011	55,413	52,110	**52,058**	61,252
lin318	42,029	**45,115**	47,854	45,420	45,930	45,945	54,034
rd400	15,281	16,730	17,564	16,850	16,864	**16,685**	19,168
fl417	11,861	13,300	14,684	**12,535**	12,589	12,879	15,288
pr439	107,217	126,849	137,341	122,468	122,899	**111,819**	131,258
pcb442	50,778	**55,750**	58,352	59,241	57,149	57,684	60,242
Mean Opt. Gap	1	1.095	1.331	1.106	1.105	1.096	1.252

Conclusions

- Approximation with linear running time complexity and optimality gaps close to 1

- Generalizations on graphs

- Unified framework for approximating combinatorial optimization problems over graphs

Learning to Solve Combinatorial Optimization Problems on Real-World Graphs in Linear Time

Iddo Drori[1,2,3] Anant Kharkar[3] William Sickinger[3] Brandon Kates[2] Qiang Ma[3] Suwen Ge[3] Eden Dolev[3] Brenda Dietrich[2] David Williamson[2] Madeleine Udell[2]

1 Massachusetts Institute of Technology

2 Cornell University

3 Columbia University in the City of New York