组合优化问题的机器学习求解

范长俊

国防科技大学

fanchangjun@nudt.edu.cn

目录

目录

CO和ML4CO

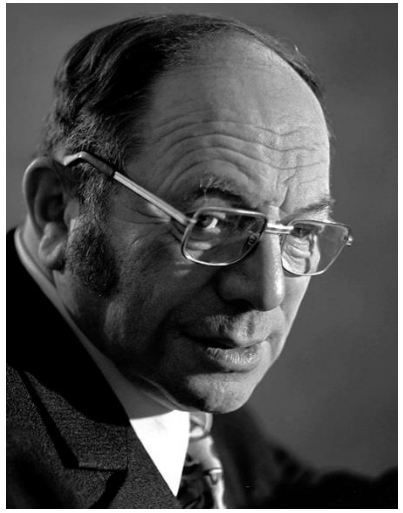
运筹学是运用科学方法（特别是数学方法）来解决工业、商业、政府部门中有关人力、机器、物资、金钱等大型系统的指挥和管理方面出现的问题，其目的是帮助管理者科学地制定策略和行动方案。

欧拉：“宇宙的构造是最完美的，它是最有智慧的造物主的杰作，宇宙的万物变化都是遵从某种最大或者最小的优化法则产生的”

广泛应用：交通运输、物流及供应链、生产计划、能源领域、医疗卫生、城市规划、经济管理、国防军事等

CO和ML4CO

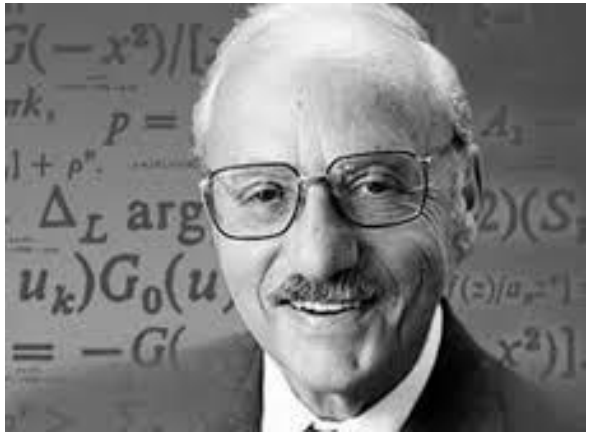
运筹学：数学规划、库存论、图与网络、决策论、博弈论、排队论、搜索论、可靠性理论等



康托洛维奇

希奇柯克

生产组织管理和制定交通运输方案方面首先研究和应用线性规划方法



丹齐格

线性规划：单纯形法

引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

运筹学-数学规划

- 线性规划
- 非线性规划：凸优化、非光滑优化、锥优化
- 整数规划：网络优化、组合优化
- 动态规划：多阶段优化

组合优化是运筹学的一个重要独立分支，是一类重要的优化问题

1、生物技术

例如：

（1）纳米生物学；
（2）合成生物学；
（3）基因组和基因工程；
（4）神经科学。

2、人工智能（AI）和机器学习技术

例如：

（1）神经网络和深度学习（例如：脑建模、时间序列预测、分类）；
（2）进化和遗传计算（例如：遗传算法、遗传算法）；
（3）强化学习；
（4）计算机视觉（例如：物体识别、图像理解）；
（5）专家系统（例如：决策支持系统，教学系统）；
（6）语音和音频处理（例如：语音识别和制作）；
（7）自然语言处理（例如：机器翻译）；
（8）规划（例如：调度、博弈）；
（9）音频和视频处理技术（例如：语音克隆、deepfakes）；
（10）AI云技术；

引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

组合优化（combinatorial optimization，CO)是通过对数学方法的研究，寻找离散事件的最优编排、分组、次序或筛选，所研究的问题涉及信息技术、经济管理、工业工程、交通运输、通信网络等领域。该问题可用数学模型描述为：

$$\begin{aligned} \min\ & f(x) \\ s.t.\quad & g(x) \geq 0 \\ & x \in D \end{aligned}$$

其中D表示有限个点组成的集合（定义域），

f 为目标函数，

$F=\{x \mid x \in D, g(x) \geq 0\}$为可行域

引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO


Auction Design
Data Center Management
Political Districting
Kidney Exchange
THE KIDNEY CHAIN
Energy Systems
Scientific Discovery
Ridesharing
Cancer Therapeutics
Airline Scheduling
Conservation Planning
Disaster Response
College Admissions

CO和ML4CO

例子 1. 0-1背包问题

设有一个容积为b的背包，n个体积分别为$a_i, i = 1,2,\dots,n$，价值分别为$c_i, i = 1,2,\dots,n$的物品，满足容量约束的基础上，如何以最大的价值装包？

$$\max \sum_{i=1}^{n} c_i x_i$$ 目标函数

$$s.t. \quad \sum_{i=1}^{n} a_i x_i \le b$$ 约束条件

$$x_i \in \{0,1\}, \quad i = 1,\cdots,n$$ 定义域$D = \{0,1\}^n$

CO和ML4CO

■ 例子 2. 旅行商问题(TSP，traveling salesman problem)

一个商人欲到n个城市推销商品，每两个城市i和j之间的距离为d_{ij}，如何选择一条道路使得商人每个城市正好走一遍后回到起点且所走路径最短。

$$\min \sum_{i\neq j}^{n} d_{ij}x_{ij}$$

目标函数

$$s.t. \quad \sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,\cdots,n$$

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,\cdots,n$$

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1, \qquad 2 \leq |S| \leq n-2, \qquad S \subset \{1,2,\cdots,n\}$$

约束条件

$$x_{ij} \in \{0,1\}, \quad i,j = 1,\cdots,n, \quad i \neq j.$$

定义域$D = \{0,1\}^{n*n}$

例子 3. 最小节点覆盖问题（MVC）

给定无向图 $G=(V,E)$，顶点集为V，边集为E，它的一个最小节点覆盖V'是顶点集V的一个子集，使得若 $(u,v)\in E$ ，则 $u\in V'$ 或 $v\in V'$

$$min \sum_{j\in V'} x_j$$ 目标函数

$$s.t.\ x_i + x_j \geq 1, \forall (i,j)\in E$$ 约束条件

$$x_j \in \{0,1\}, \forall j \in V$$ 定义域$D=\{0,1\}^n$

- 例子

4. 最大割问题（Max-Cut)

给定带权图 $G=(V,E,W)$，找到节点子集 $S\subset V$，使得割集 $C\subset E$ 中的边的权重之和最大，其中割集 C 中的边 (u,v) 满足 $u\in S, v\in \bar{S}$

$$\max \sum_{(i,j)\in E} w_{ij}(2-x_i-x_j)(x_i+x_j)$$ 目标函数

$$s.t.\quad x_i+x_j\geq 1, \forall i,j\in V$$ 约束条件

$$x_i\in\{0,1\}, \forall i\in V$$ 定义域$D=\{0,1\}^n$

CO和ML4CO

- 例子

5.装箱问题(bin packing)

如何把n个尺寸不超过1的物品装入尺寸为1的箱子，并使所用的箱子个数最少

6.车间作业调度问题

n个工件，J1,...,Jn在m台机器M1,M2,...,Mm上加工。每个工件Ji有ni个工序，Oi1,...,Oini,第Oij工序的加工时间为pij，必须按工序进行加工且每一工序必须一次加工完成。一台机器在任何时刻最多只能加工一个产品，一个工件不能同时在两台机器上加工，如何安排才能使所有工件完工时间最小

7.图的顶点着色问题

8.独立集问题

9.划分问题

10.选址问题...

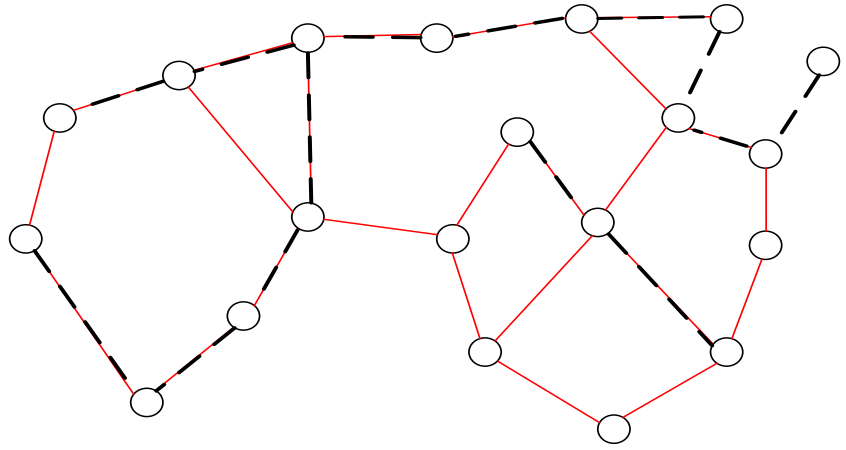
引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

军事应用

1.军事物资联合投送路径优化

联合投送是指运用两种以上运输方式，将人员、武器装备、后勤物资等分批次快速送达作战地域的大型军事行动。现计划通过公路、铁路投送多个编组(每个编组由若干梯队构成，梯队为运输的基本单位)。节点代表交叉路口、车站或城市，边代表连接节点的路段，其中红色实线代表公路，灰色虚线代表铁路



交通线路示意图

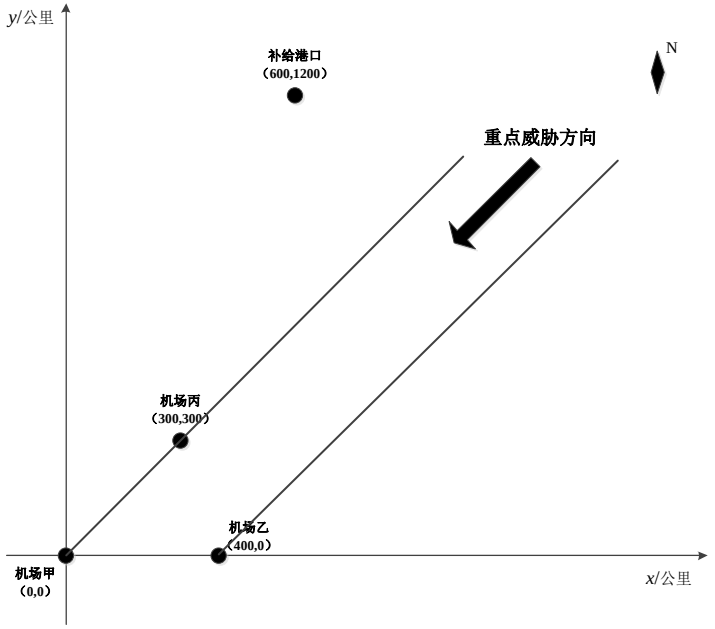
引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

军事应用

2.海域巡逻与岛礁保障任务规划

派出预警机对某海域岛礁的重点威胁方向进行全天候24小时预警巡逻。预警机在执行巡逻任务时，需要2架战斗机护航，战斗机由该海域甲、乙、丙3个岛礁出动，现3个岛礁配备了不同数量的某型战斗机



- 问题1：确定预警机的阵位（巡逻区域）
- 问题2：给出各岛礁战斗机一周的护航调度方案
- 问题3：给出运输机和运输船一周的补给保障方案

引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

军事应用

- 军事资源部署配置
- 火力-目标分配
- 军事物资装载优化
- 侦察卫星任务规划
- 无人机航迹规划……

问题特征

- 解是可数的，可行域（解） 是有限的
- 最优解一定存在
- 可行方案非常多，枚举法不可行，NP-hard

引自《王建江，大规模组合优化问题求解方法与应用，智慧调度网络公益系列讲座》

CO和ML4CO

- **数学优化算法-精确算法（指数时间）**
  - 分支定界，分支定价，分支-切割，割平面、拉格朗日松弛
- **启发式算法（多项式时间）**
  - 近似算法、贪婪算法，基于规则的构造算法，...
- **元启发式算法（多项式时间）**
  - 遗传算法，蚁群算法，禁忌搜索，模拟退火...
- **机器学习方法（多项式时间）**
  - 监督学习、强化学习、图神经网络...

CO和ML4CO

组合优化求解器

- 商用求解器
  - Gurobi, CPLEX, Matlab, Lingo，CMIP
- 开源求解器
  - SCIP, LP_Solve, CBC, GLPK, MOSEK

CO和ML4CO

人工时代

人类学习如何去设计算法

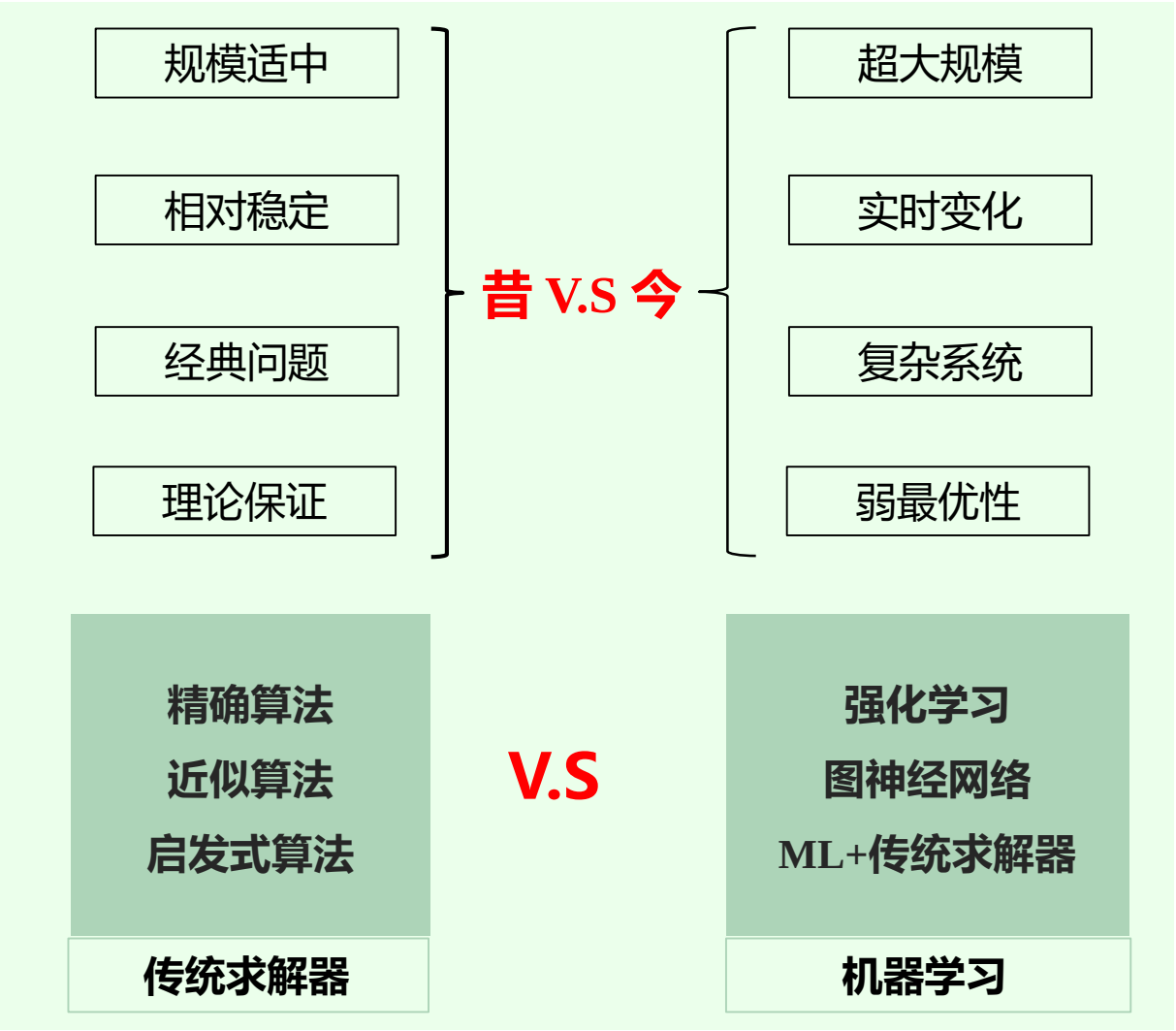
智能时代

“算法”可以学习如何去设计算法吗?

机器学习

组合优化

CO和ML4CO



- CVPR最佳论文：学习求解极小值问题
- EJOR21 Bengio综述：学习求解组合优化
- NSF投资2千万美元成立AI求解最优化研究所
- 商用求解器：Gurobi CPLEX SCIP FICO Xpress 杉数COPT MindOPT

CVPR nature machine intelligence AAAI ICML ICLR NeurIPS

CO和ML4CO

组合优化

令人兴奋和不断发展的交叉研究方向

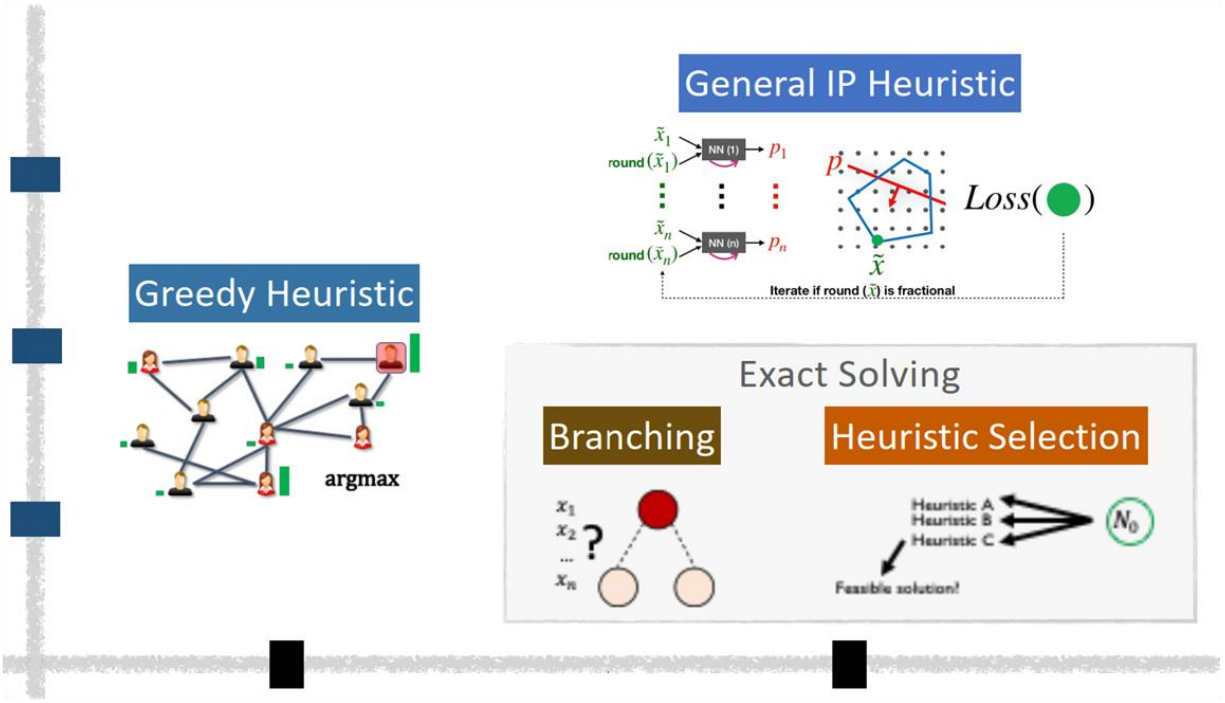
机器学习

机器学习范式

自监督学习

强化学习

监督学习



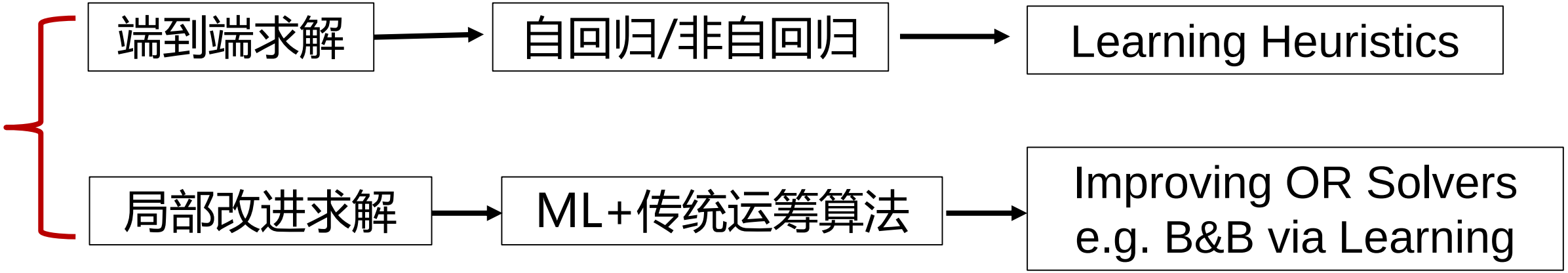
图优化

离散优化

组合优化问题

CO和ML4CO

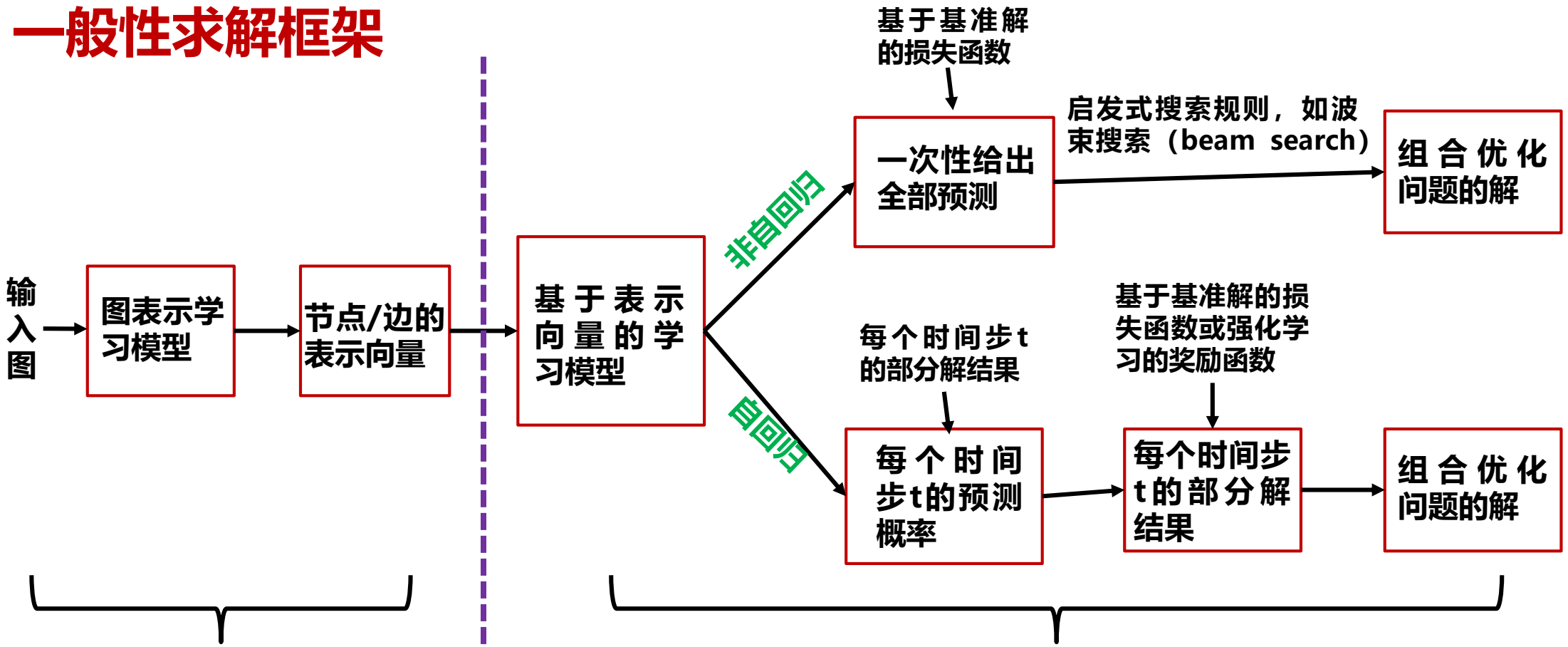
组合优化问题的机器学习求解范式



目录

图上的组合优化问题求解

一般性求解框架



图上的组合优化问题求解

非自回归方法

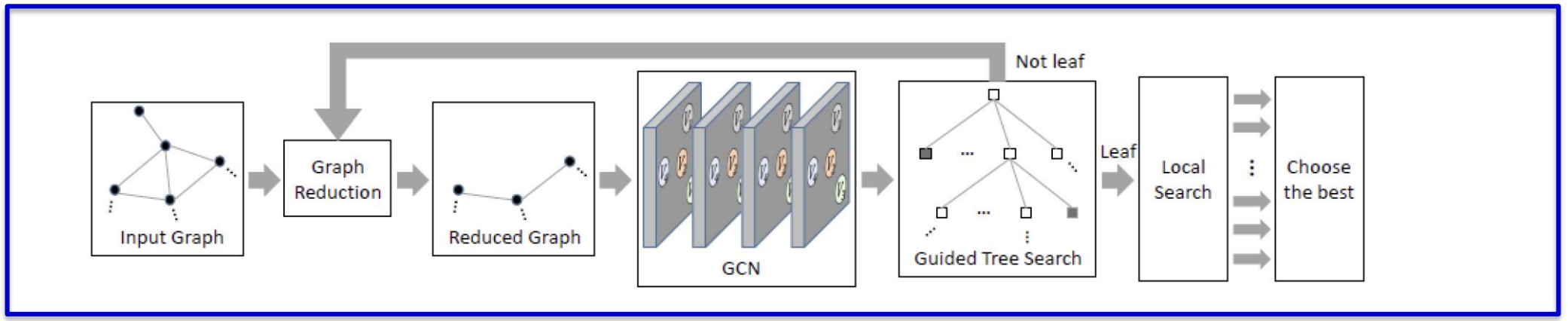
Combinatorial Optimization with Graph Convolutional Networks and Guided Tree Search

Zhuwen Li
Intel Labs

Qifeng Chen
HKUST

Vladlen Koltun
Intel Labs

监督学习+树搜索+图论技巧



Li, Zhuwen and Chen, Qifeng and Koltun, Vladlen. Combinatorial optimization with graph convolutional networks and guided tree search. Advances in neural information processing systems, 2018.

图上的组合优化问题求解

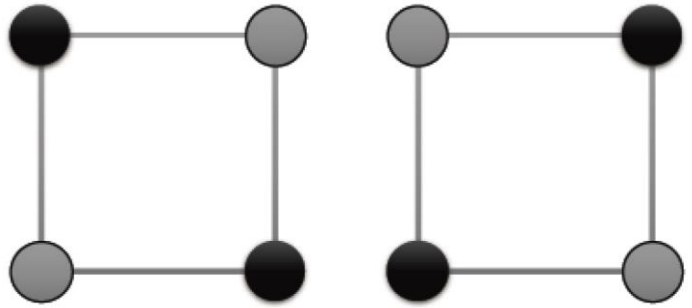
非自回归方法

损失函数

$$\ell(\mathbf{l}_i, f(\mathcal{G}_i;\boldsymbol{\theta})) = \sum_{j=1}^{N}\{\mathbf{l}_{ij}\log(f_j(\mathcal{G}_i;\boldsymbol{\theta})) + (1-\mathbf{l}_{ij})\log(1-f_j(\mathcal{G}_i;\boldsymbol{\theta}))\}$$

To enable the network to differentiate between different modes, we extend the structure of f to generate multiple probability maps. Given the input graph $\mathcal{G}$, the revised network f generates M probability maps: $\langle f^1(\mathcal{G}_i;\boldsymbol{\theta}), \ldots, f^M(\mathcal{G}_i;\boldsymbol{\theta})\rangle$. To train f to generate diverse high-quality probability maps, we adopt the hindsight loss [18, 8, 28]:

$$\mathcal{L}(\mathcal{D},\boldsymbol{\theta}) = \sum_i \min_m \ell(\mathbf{l}_i, f^m(\mathcal{G}_i;\boldsymbol{\theta})), \quad (3)$$



Solution 1 Solution 2

Figure 2: Two equivalent solutions for MIS on a four-vertex graph. The black vertices indicate the solution.

Li, Zhuwen and Chen, Qifeng and Koltun, Vladlen. Combinatorial optimization with graph convolutional networks and guided tree search. Advances in neural information processing systems, 2018.

图上的组合优化问题求解

非自回归方法

Method	Solved	MC	MIS	MVC
Classic	0.0%	30.03	21.53	991.72
Classic+GR+LS	0.0%	42.83	24.64	988.61
S2V-DQN	0.0%	40.40	23.76	989.49
S2V-DQN+GR+LS	0.0%	42.98	24.70	988.55
Gurobi	0.0%	39.75	24.12	989.13
ReduMIS	25.0%	44.95	24.87	988.38
Ours	**62.5%**	**45.55**	**25.06**	**988.19**

Table 3: Results on the BUAA-MC dataset. The table reports the fraction of solved MC problems and the average size of MC, MIS, and MVS solutions.

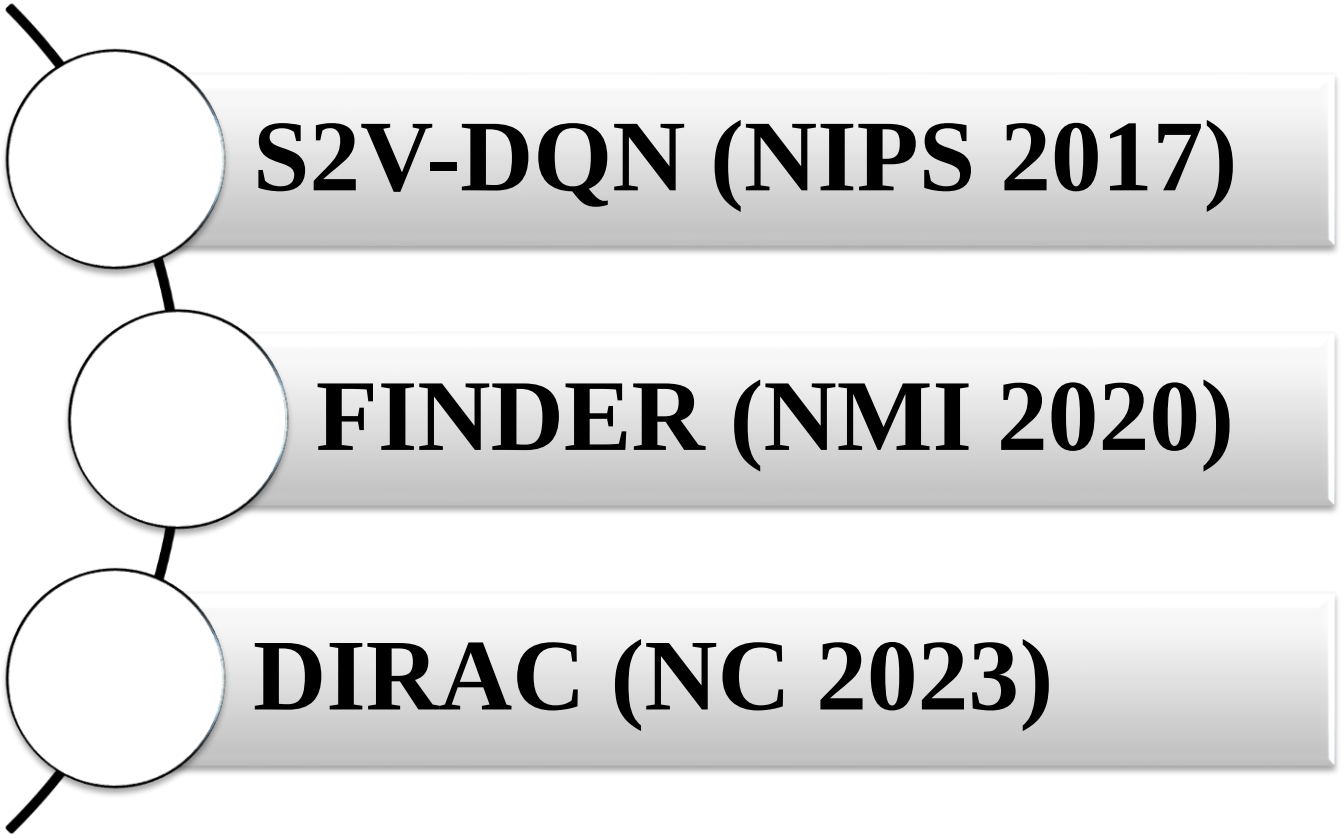
Local search作用很大

Method	Solved	MIS
Basic	18.8%	425.55
Basic+Tree	59.2%	426.52
No local search	42.4%	426.41
No reduction	91.0%	426.81
Full w/o parallel	98.8%	426.86
Full with parallel	**100.0%**	**426.88**

图上的组合优化问题求解

自回归方法

图表示学习+强化学习



图上的组合优化问题求解

自回归方法

图表示学习+强化学习

- **S2V-DQN (NIPS 2017)**
- **FINDER (NMI 2020)**
- **DIRAC (NC 2023)**

S2V-DQN

Learning Combinatorial Optimization Algorithms over Graphs

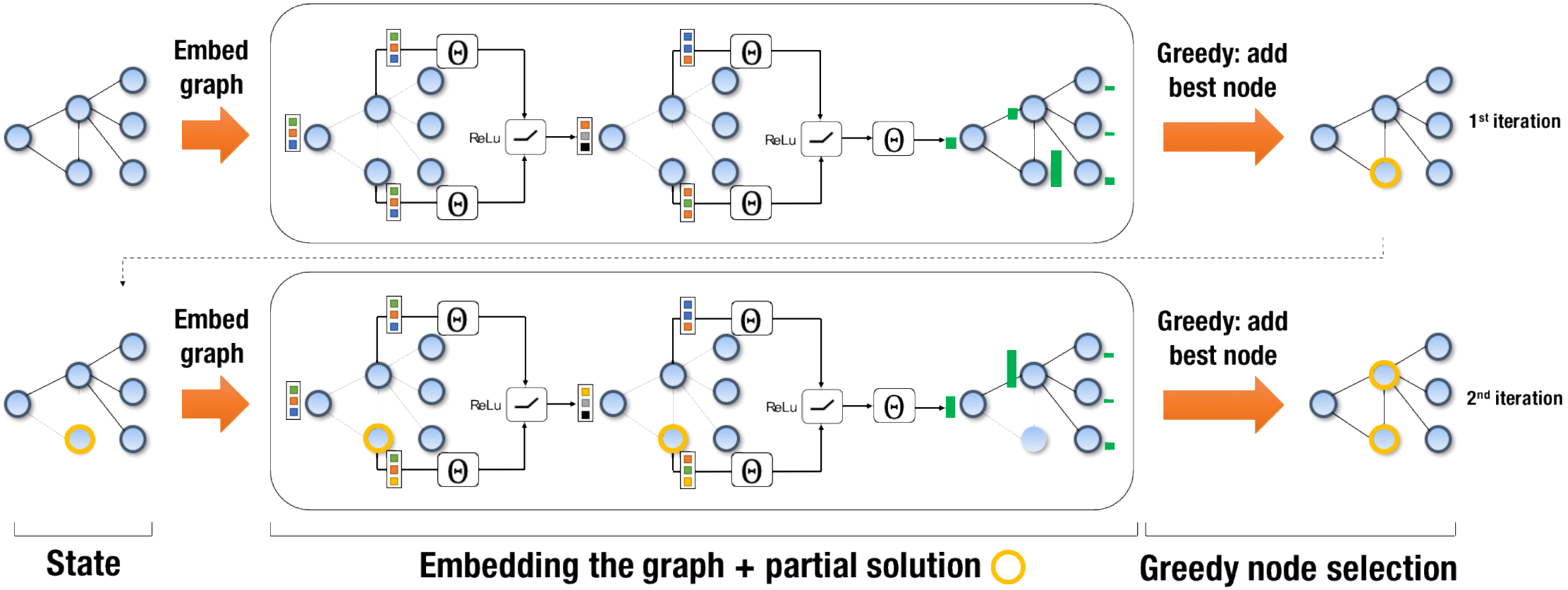
Hanjun Dai*, Elias B. Khalil*, Yuyu Zhang, Bistra Dilkina, Le Song
College of Computing, Georgia Institute of Technology
{hanjun.dai, elias.khalil, yuyu.zhang, bdilkina, lsong}@cc.gatech.edu

Learning combinatorial optimization algorithms over graphs
E Khalil, H Dai, Y Zhang, B Dilkina… - Advances in neural …, 2017 - proceedings.neurips.cc
… In order to represent such complex phenomena over combinatorial structures, we will leverage a deep learning architecture over graphs, in particular the structure2vec of [9], to …
☆ 保存 99 引用 被引用次数：1237 相关文章 所有 11 个版本 导入BibTeX »

- **NeurIPS-2017**
- **机器学习组合优化的里程碑工作，引领并启发了该领域后续许多的研究**

S2V-DQN

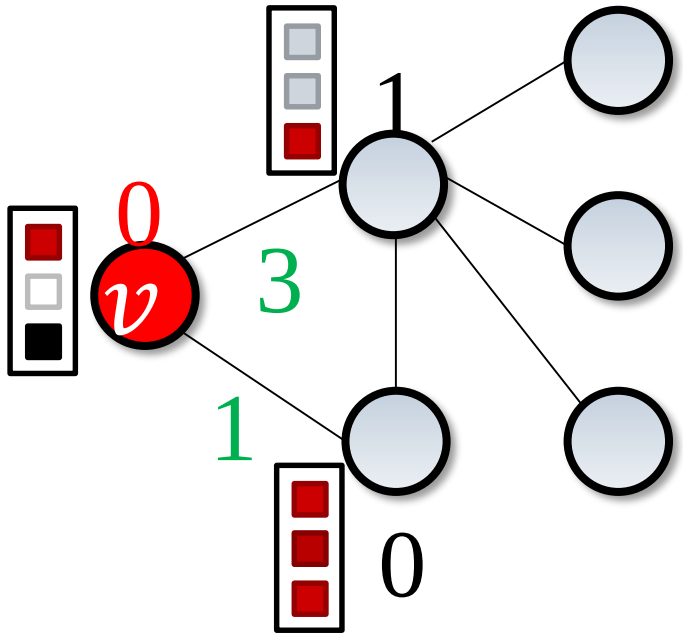
整体框架


Embed graph
ReLu
Greedy: add best node
1st iteration
Embed graph
ReLu
Greedy: add best node
2nd iteration
State
Embedding the graph + partial solution
Greedy node selection

S2V-DQN

整体框架 —— 图表示学习

[Dai, et al., ICML 2016]



非线性变换: $\text{relu}(x) = \max(0, x)$

重复嵌入 T 次:

更新特征向量

$$\mu_v^{(t+1)} \leftarrow \text{relu}\Big(\theta_1 x_v + \theta_2 \sum_{u \in \mathcal{N}(v)} \mu_u^{(t)} + \theta_3 \sum_{u \in \mathcal{N}(v)} \text{relu}(\theta_4\, w(v,u))\Big)$$

邻域边权

Θ: 模型参数

S2V-DQN

整体框架 —— 强化学习

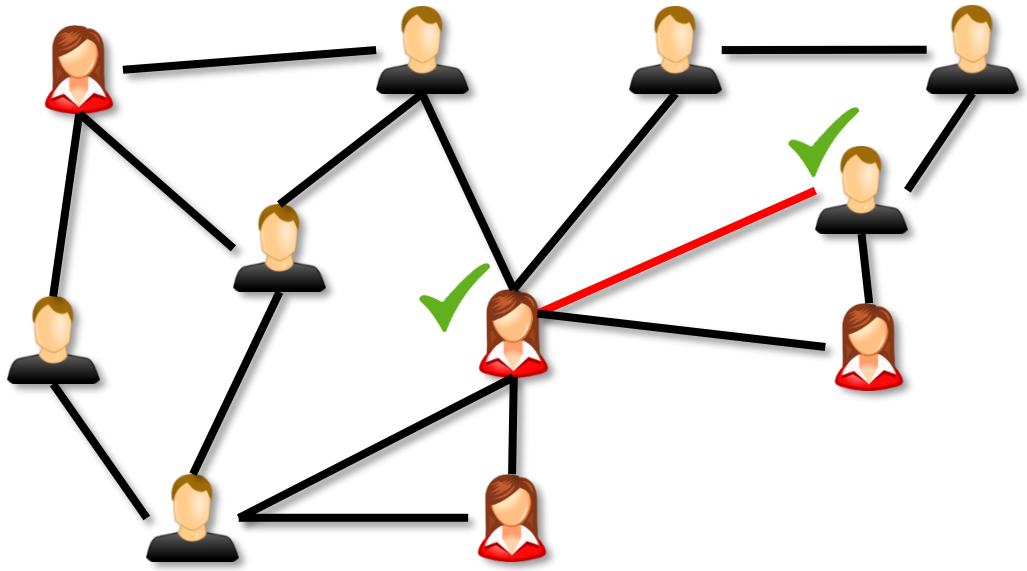
- **奖励函数 $R(t)$**：当前步的分数
- **状态 S**：当前游戏屏幕
- **动作 i**：左移/右移板子
- **动作值函数$\hat{Q}(S,i)$**：预测的未来累计奖励函数值
- **策略 $\pi(s)$**：如何选择下一步

贪心策略:
$i^* = argmax_i \hat{Q}(S,i)$

[Minh, et al. Nature 2015]

S2V-DQN

整体框架 —— 强化学习



- **奖励函数** $R(t)$: -1
- **状态** S: 当前的图（选择的节点做好标记）
- **动作** i: 选择下一个节点v
- **动作值函数**$\hat{Q}(S,i)$: $\hat{Q}(S,v)$
- **策略** $\pi(s)$: 如何选择下一个节点

贪心策略:
$i^* = argmax_i \hat{Q}(S,i)$

[Dai, et al. NeurIPS 2017]

S2V-DQN

Algorithm 1 Q-learning for the Greedy Algorithm

Initialize experience replay memory $\mathcal{M}$ to capacity N
for episode $e = 1$ **to** L **do**
  Draw graph G from distribution $\mathbb{D}$
  Initialize the state to empty $S_1 = ()$
  for step $t = 1$ **to** T **do**
    $v_t = \begin{cases} \text{random node } v \in \overline{S}_t, & \text{w.p. } \epsilon \\ \text{argmax}_{v \in \overline{S}_t} \widehat{Q}(h(S_t), v; \Theta), & \text{otherwise} \end{cases}$
    Add v_t to partial solution: $S_{t+1} := (S_t, v_t)$
    if $t \geq n$ **then**
      Add tuple $(S_{t-n}, v_{t-n}, R_{t-n,t}, S_t)$ to $\mathcal{M}$
      Sample random batch from $B \overset{iid.}{\sim} \mathcal{M}$
      Update Θ by SGD over (6) for B
    end if
  end for
end for
return Θ

Θ: 模型参数
依赖节点和边特征

生成训练图数据

探索和利用

更新状态

优化模型参数

$$(y - \widehat{Q}(h(S_t), v_t; \Theta))^2$$
$$y = \gamma \max_{v'} \widehat{Q}(h(S_{t+1}), v'; \Theta) + r(S_t, v_t)$$

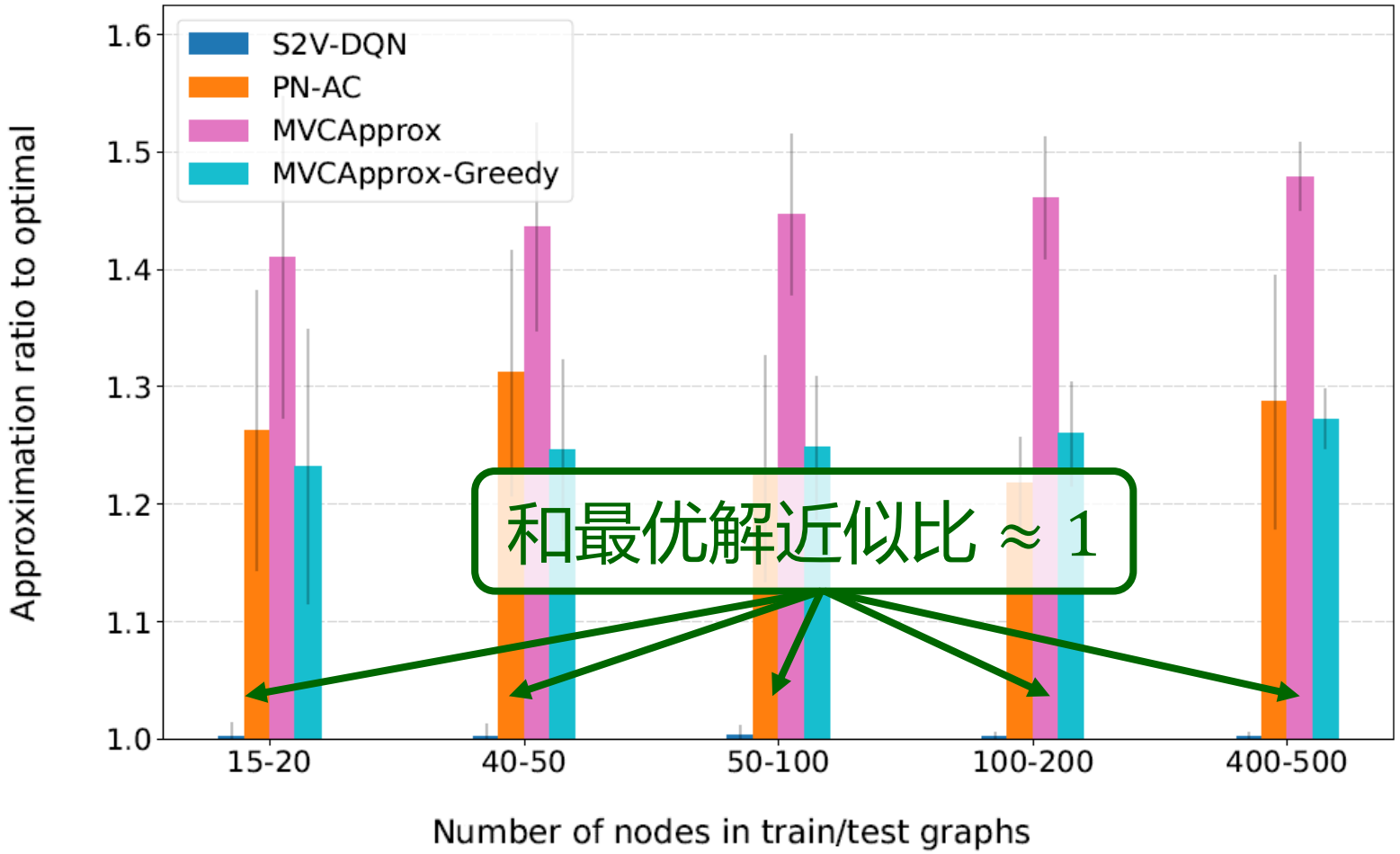
S2V-DQN

实验设置

	Minimum Vertex Cover (最小节点覆盖)	Maximum Cut (最大割)	Traveling Salesman Problem (TSP)
训练图数据	Erdos-Renyi (ER) or Barabasi-Albert (BA)	ER or BA	DIMACS generator; uniform grid or clustered
基准解求解器	ILP with CPLEX	IQP with CPLEX	Concorde

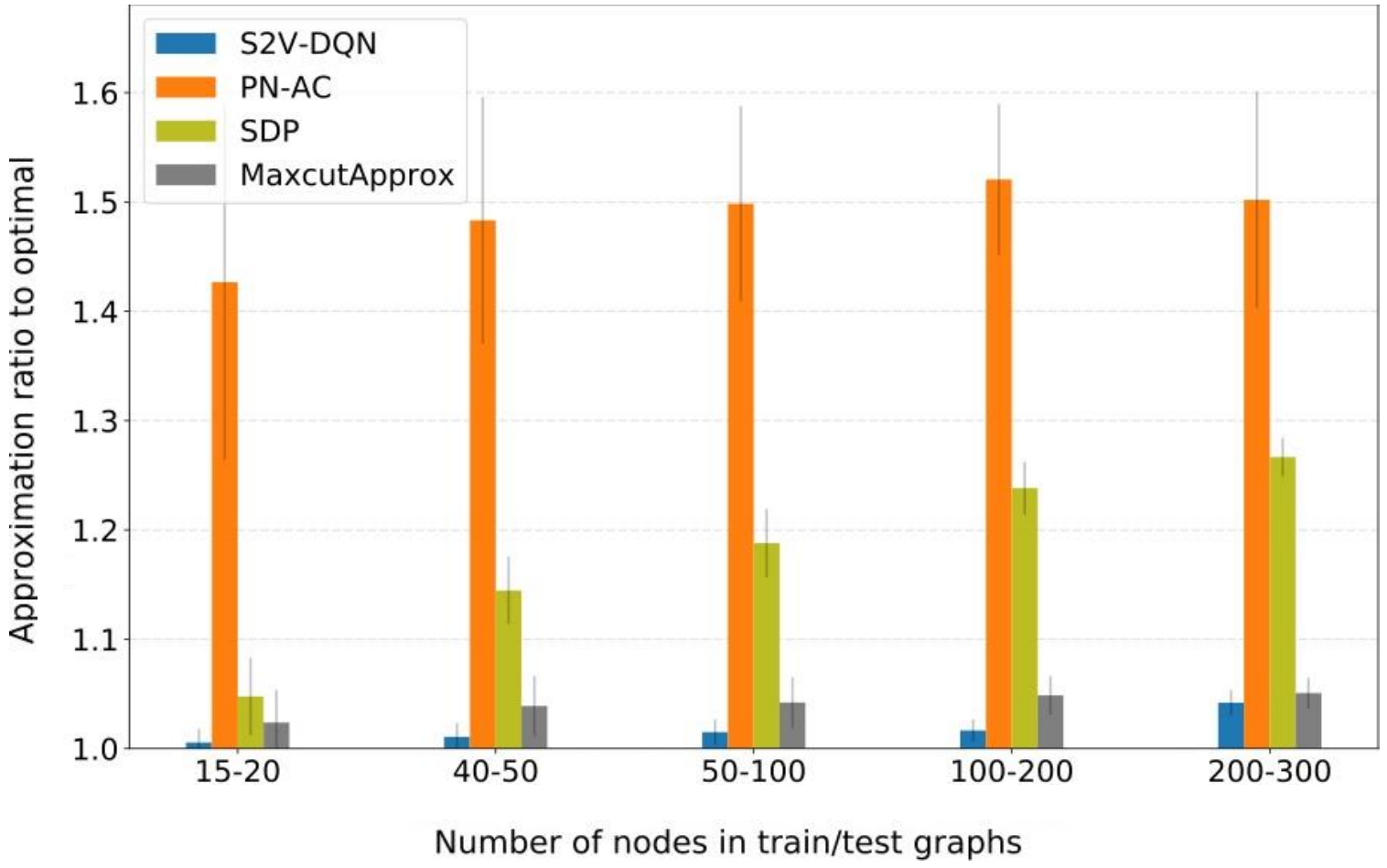
S2V-DQN

最小节点覆盖问题



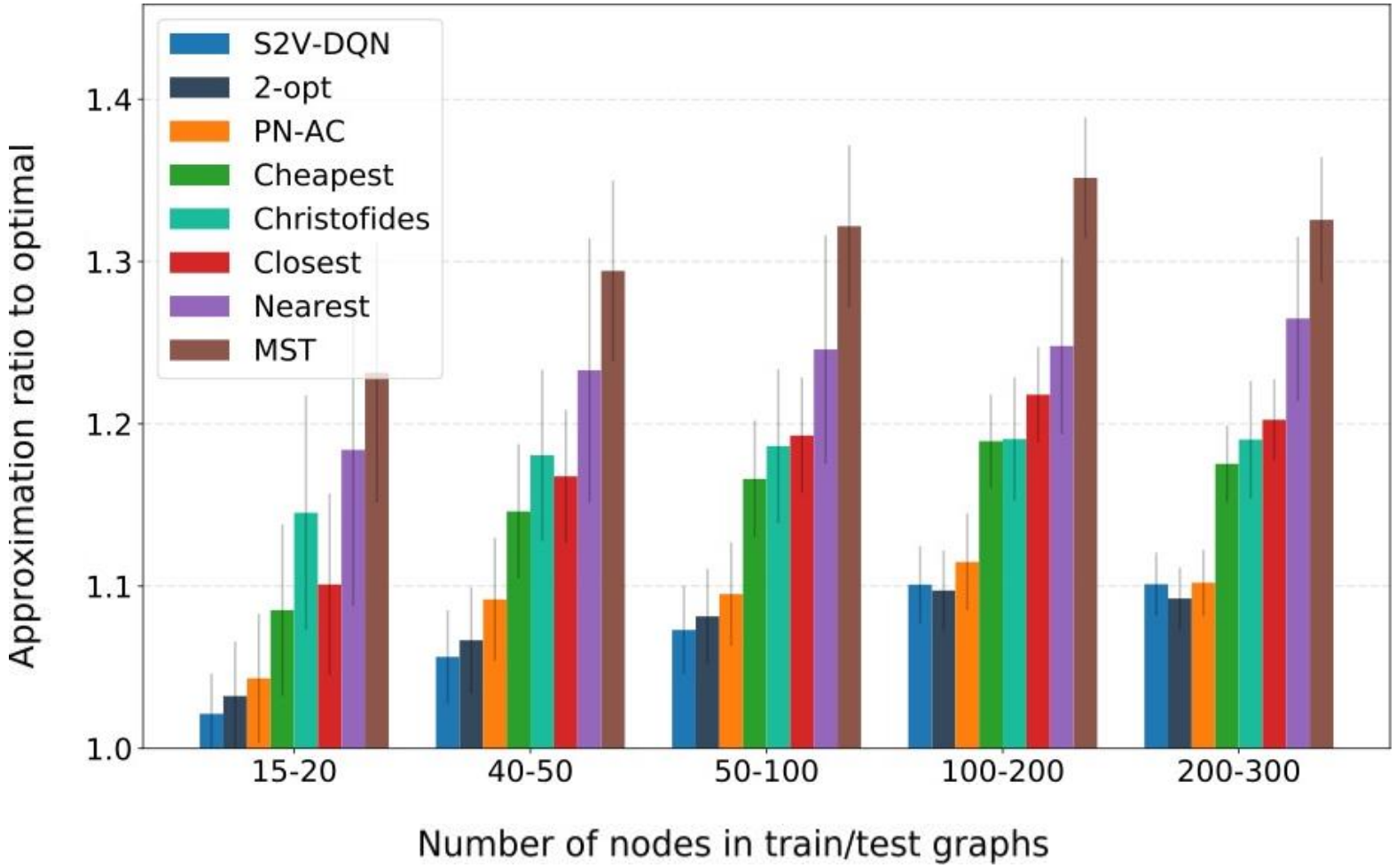
S2V-DQN

最大割问题

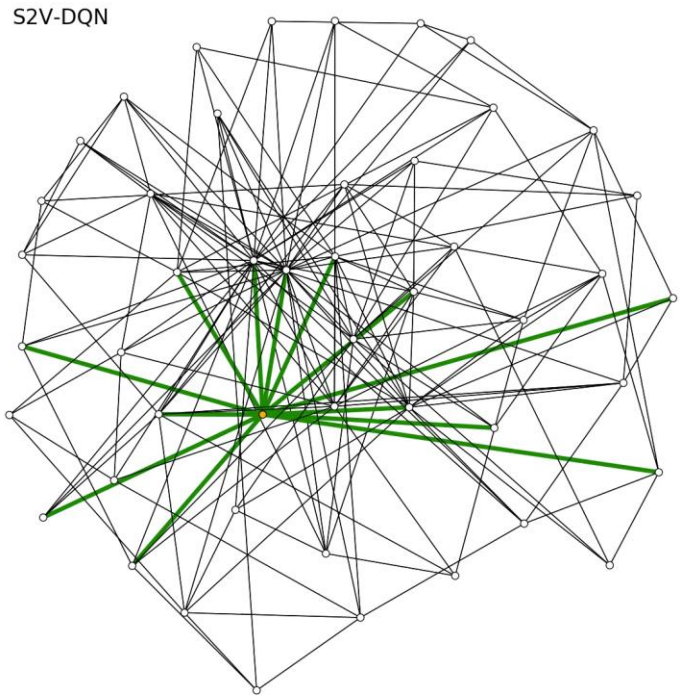


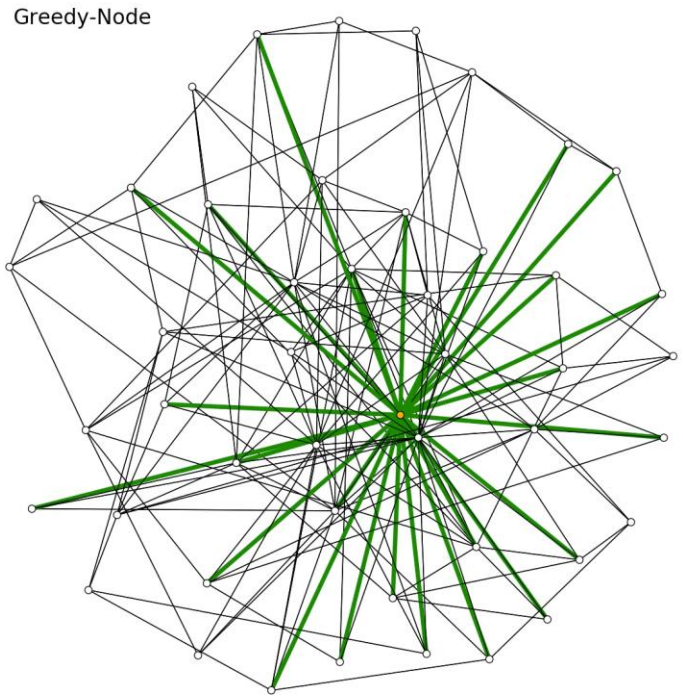
S2V-DQN

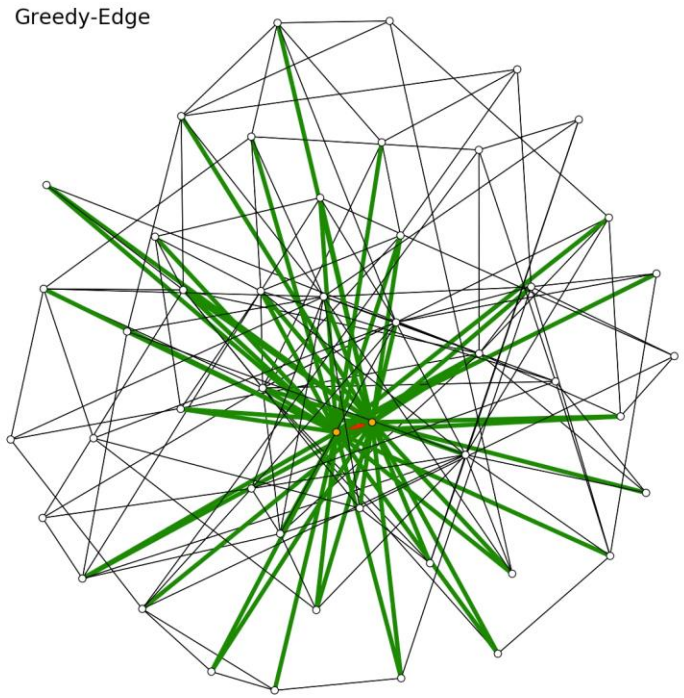
TSP问题



S2V-DQN



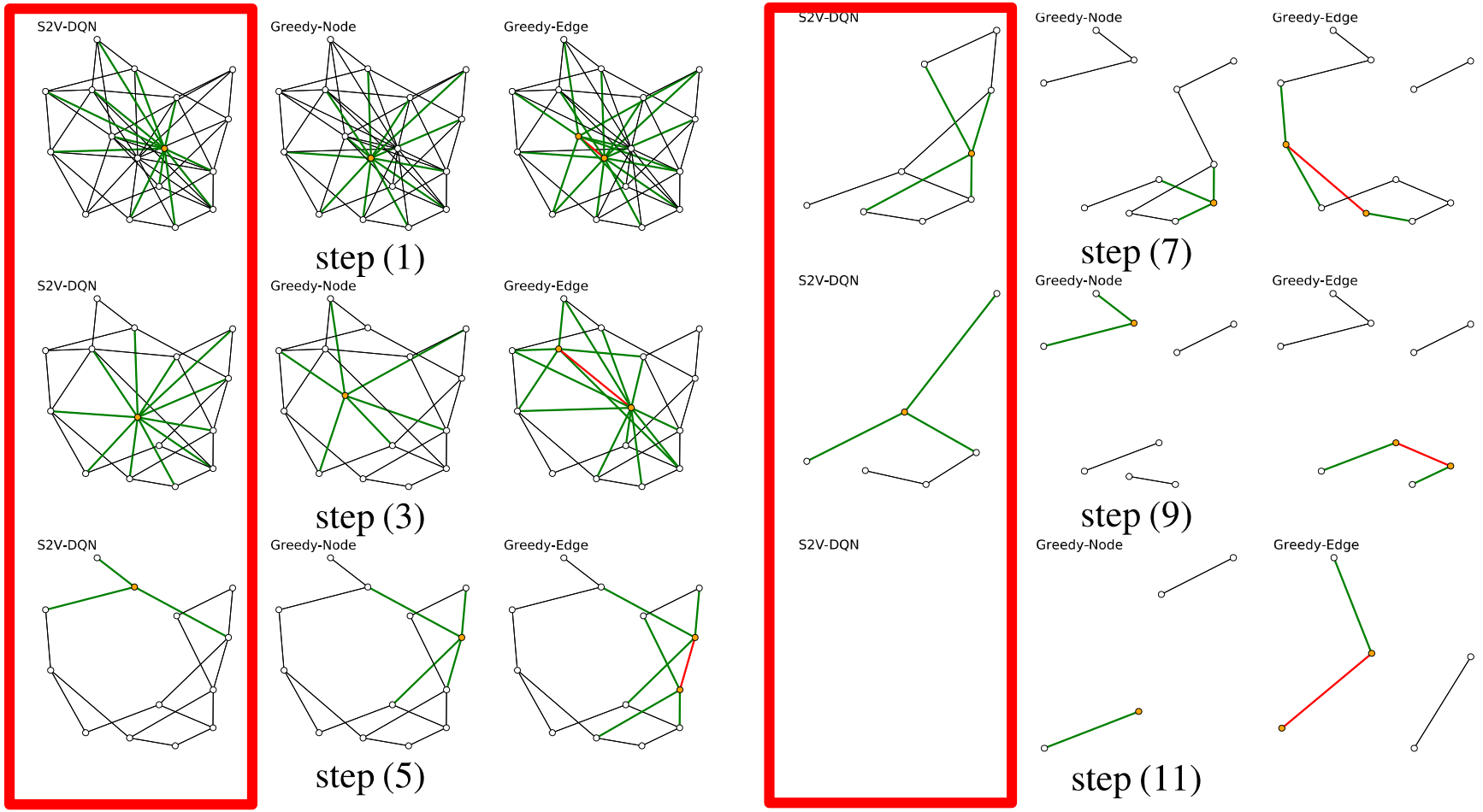




S2V-DQN

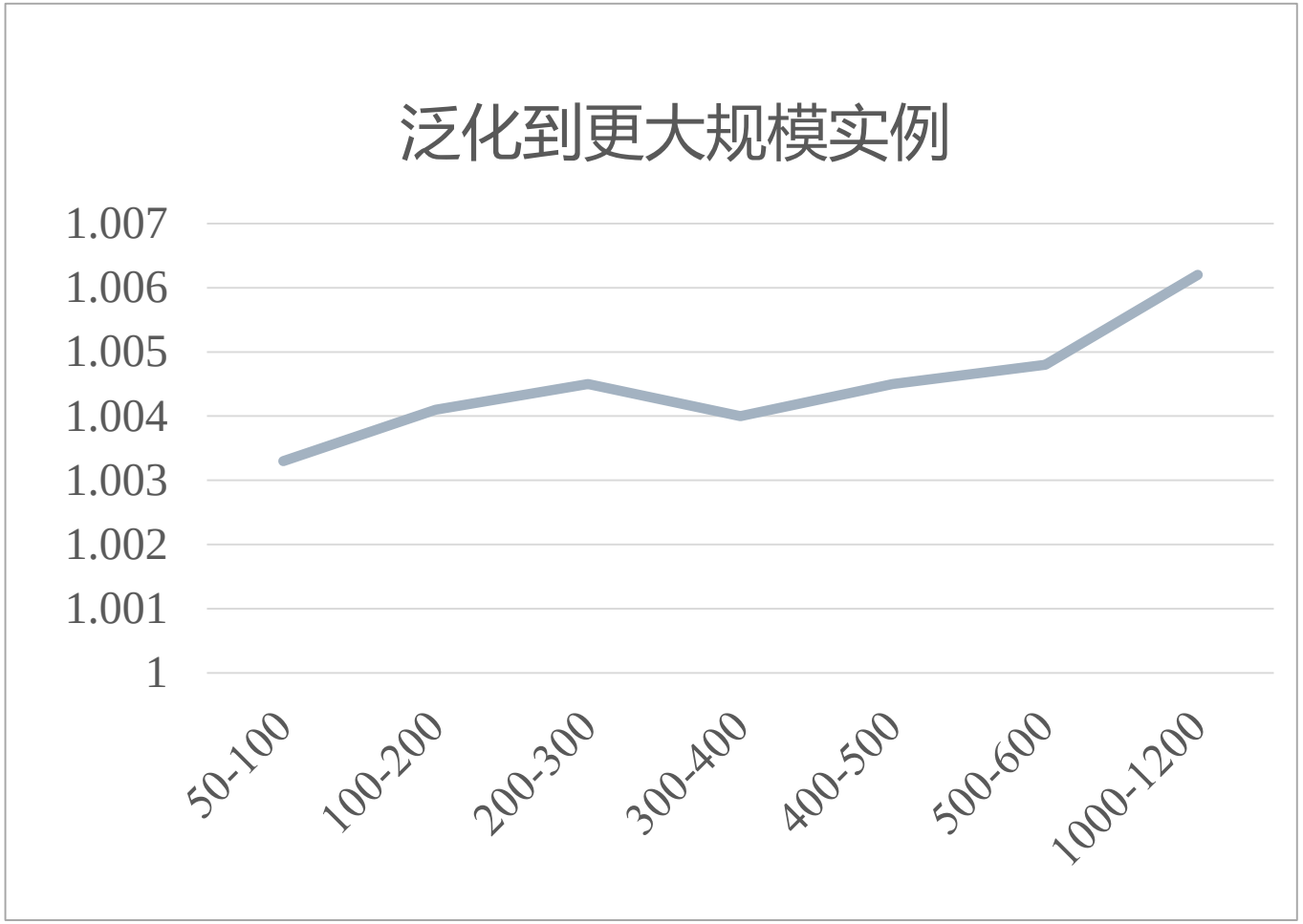
算法可解释分析

每次都倾向于选择那些覆盖更多节点，同时避免破坏连通性的节点，从长远看，需要的节点更少。



S2V-DQN

- 训练图包含50-100个节点
- 可以较好地泛化到更大规模的实例（同一分布）
- 和最优解近似比< 1.007



S2V-DQN

评价

- 统一的框架，适用多种图上的组合优化问题
- 论文代码开源，结果可复现，是一个非常扎实可信的工作
- 模型效果非常好，在最小节点覆盖问题接近最优解
- **不足之处**在于基准方法都是比较弱的贪心算法，难以证明在单个问题上比现有最优算法强，且模型泛化性能有待提高。

图上的组合优化问题求解

自回归方法

图表示学习+强化学习

- S2V-DQN (NIPS 2017)
- FINDER (NMI 2020)
- DIRAC (NC 2023)

FINDER

nature machine intelligence

ARTICLES

https://doi.org/10.1038/s42256-020-0177-2

Check for updates

Finding key players in complex networks through deep reinforcement learning

Finding key players in complex networks through deep reinforcement learning

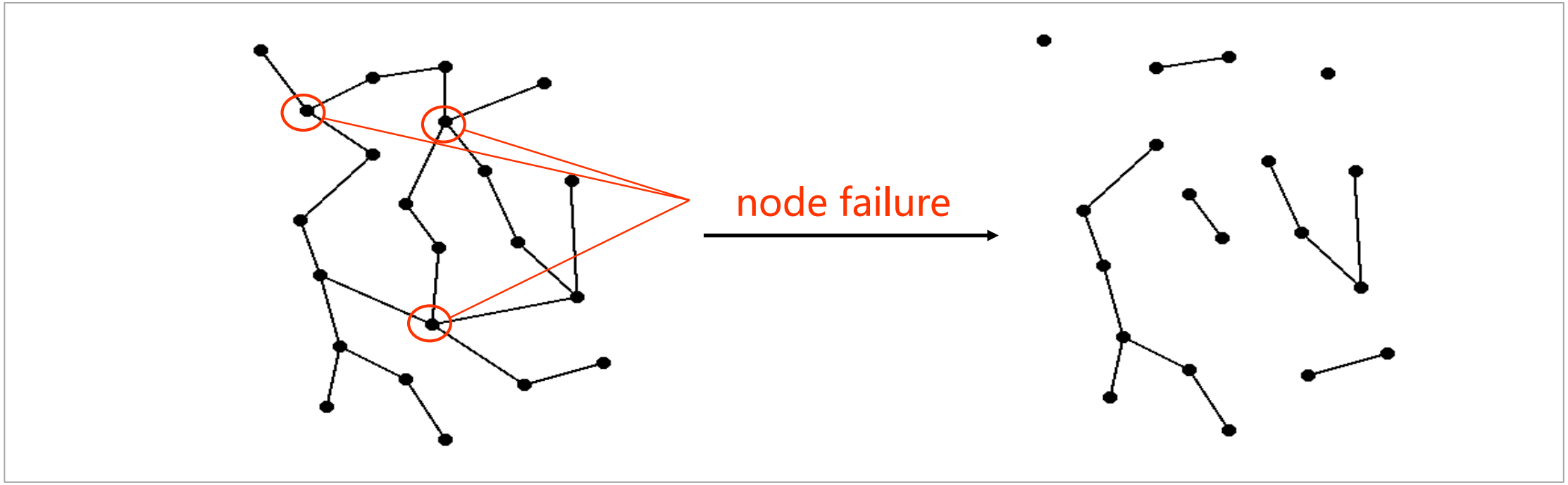
C Fan, L Zeng, Y Sun, YY Liu - Nature machine intelligence, 2020 - nature.com

… search … (FInding key players in Networks through DEep Reinforcement learning), a generic and scalable deep reinforcement learning framework to find key players in complex networks (…

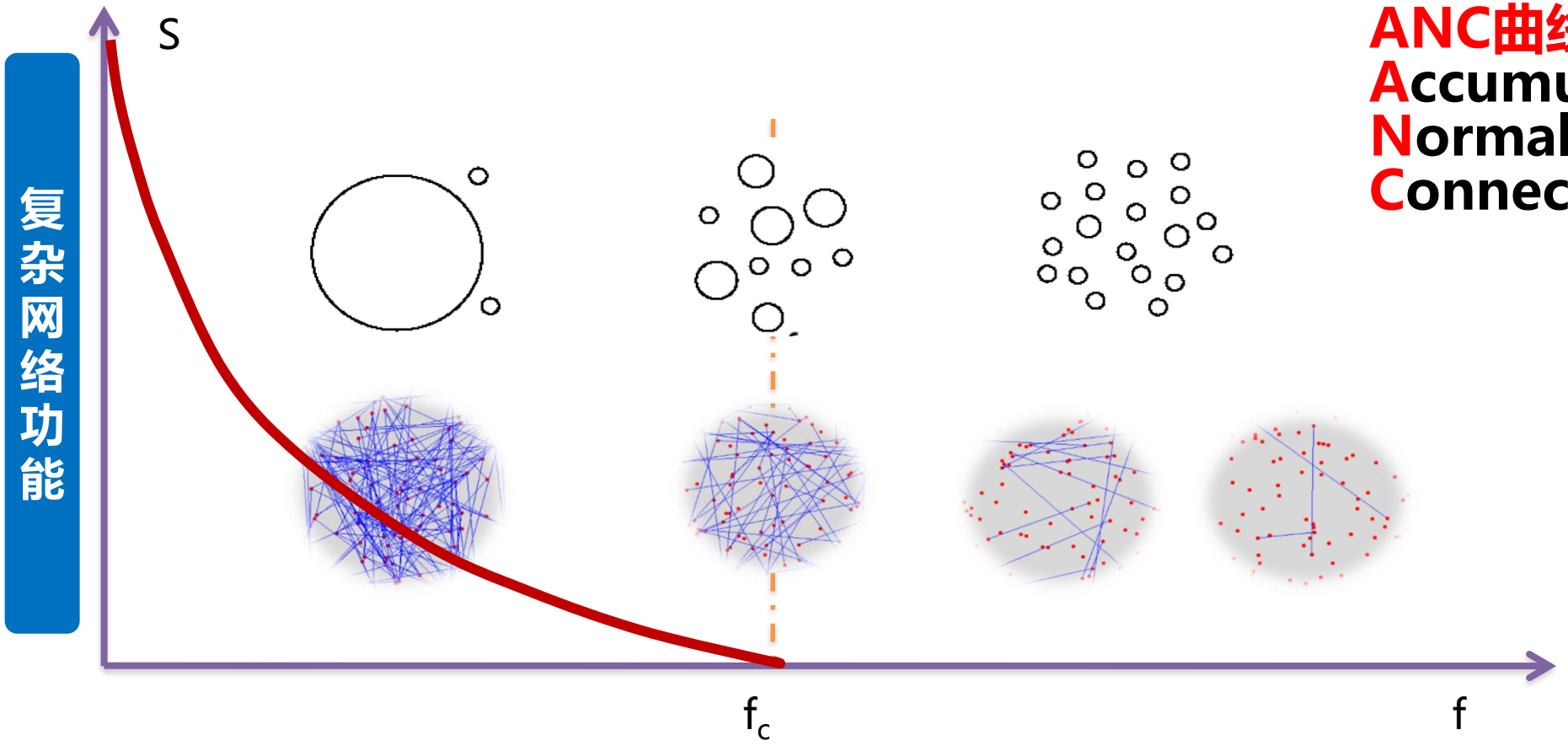
☆ 保存 99 引用 被引用次数：139 相关文章 所有 8 个版本 导入BibTeX

FINDER

牵一发而动全身，网络中的某些节点（或节点集）一旦被移除（或攻击），就会对网络的连通性（或鲁棒性）产生雪崩式的影响。如何寻找到这样的节点（或节点集）是网络科学中非常重要的课题。—— **网络瓦解问题**



FINDER



ANC曲线：
Accumulated
Normalized
Connectivity

FINDER

复杂网络的结构性功能指标

- 1、连通片的数量
- 2、**最大连通片规模**
- 3、信息熵
- 4、连通节点对的数量
- 5、源节点和目标节点之间的最短路径
- 6、节点对之间的平均最短路径长度
- 7、自然连通度
- 8、最大介数

FINDER

复杂网络的**非结构性功能指标**

- 1、体系预警范围
- 2、体系火力打击范围
- 3、体系平均通信时间

体系破击制胜机理：分析对手作战目标级联过程和体系威胁，通过**关键目标**的精确打击，打破敌作战体系演化平衡关系，使失衡，“涌现”出大量负面效应。

FINDER

剩余最大连通片（GCC）

- **定义：当前网络的连通片分支中节点个数最多的分支**
- 最大连通片规模是评价当前网络连通性的重要指标，因为**基本上网络上所有的行为都需要在连通的环境下才能运行，最大的连通片代表了当前网络所能承载的网络行为的上限。**
- 如果网络行为是**谣言传播**，那么最大连通片代表了**最多的谣言知情者**；
- 如果网络行为是**疾病传播**，最大连通片则代表**最多的疾病感染者**；
- 如果网络行为是**战场信息传达与指挥决策**，最大连通片则代表了**最多可以进行商讨决策的团体规模**。如果一个决策的正确制定需要至少M个人，那么我们对敌方目标体系瓦解的任务就是移除节点直至剩余最大连通片规模不超过M个节点。

FINDER

- **挑战一：**发现最优的一组关键节点是典型的NP难的**图上组合优化问题**，因此**不存在多项式复杂度的精确算法；**
- **挑战二：**现有许多近似算法**难以有效平衡求解质量和求解速度**；
- **挑战三**：**缺乏统一求解框架**，许多现有方法都是定制的（ad-hoc）。

FINDER

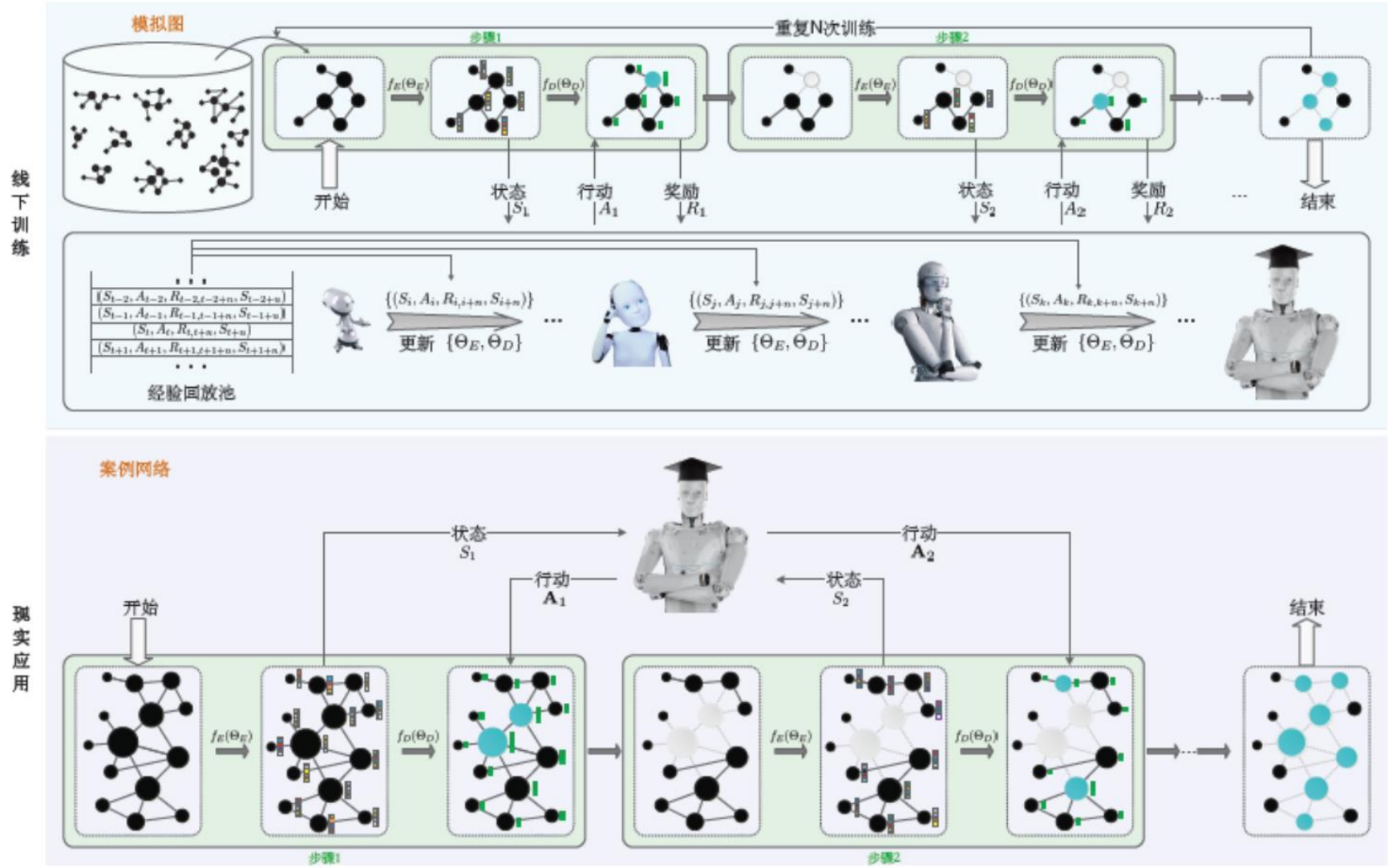
- **FINDER (FInding key players in Networks through DEep Reinforcement learning)，一种通用且可扩展的深度强化学习框架。**

优点：

- **灵活性**：一套框架多种应用场景，一次训练多次调用。
- **有效性**：在六种不同的网络瓦解场景中效果平均提升28.7%。
- **高效率**：计算效率提升超过两个数量级。

FINDER

模型框架


线下训练
模拟图
重复N次训练
步骤1
步骤2
开始
状态
行动
奖励
结束
经验回放池
更新 $\{\Theta_E, \Theta_D\}$
现实应用
案例网络
状态 S_1
行动 A_1
状态 S_2
行动 A_2
开始
结束
步骤1
步骤2

FINDER vs S2V-DQN

主要区别

- 节点表示
- 状态表示
- 动作值函数Q计算
- 训练损失函数
- 测试策略

FINDER vs S2V-DQN

主要区别——节点表示

S2V-DQN

Input: parameter $\boldsymbol{W}$ in $\widetilde{\mathcal{T}}$
Initialize $\widetilde{\mu}_i^{(0)} = \mathbf{0}$, for all $i \in \mathcal{V}$
for $t = 1$ **to** T **do**
  for $i \in \mathcal{V}$ **do**
    $l_i = \sum_{j \in \mathcal{N}(i)} \widetilde{\mu}_i^{(t-1)}$
    $\widetilde{\mu}_i^{(t)} = \sigma\left(W_1 x_i + W_2 l_i + W_3 \sum_{j \in \mathcal{N}(i)} x_j\right)$
  end for
end for{fixed point equation update}
return $\{\widetilde{\mu}_i^T\}_{i \in \mathcal{V}}$

structure2vec

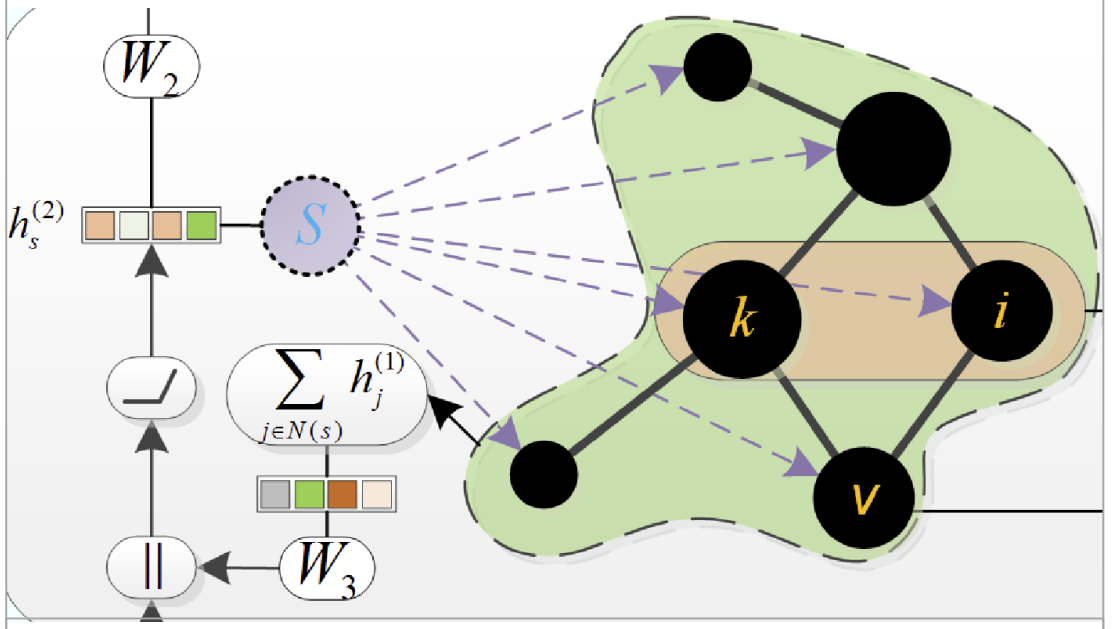
FINDER

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1 \ldots K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

GraphSAGE

FINDER vs S2V-DQN

主要区别——状态表示

S2V-DQN	FINDER
$\theta_6 \sum_{u \in V} \mu_u^{(T)}$	
简单的**节点嵌入之和**表示状态	引入虚拟节点表示状态

FINDER vs S2V-DQN

主要区别——动作值函数Q计算

S2V-DQN	FINDER
$\widehat{Q}(h(S), v; \Theta) = \theta_5^\top \operatorname{relu}([\theta_6 \sum_{u \in V} \mu_u^{(T)}, \theta_7 \mu_v^{(T)}])$	$Q(s, a) = W_5^\top \operatorname{ReLU}(z_a^\top \cdot z_s \cdot W_4)$
状态和动作向量的**拼接**	状态和动作向量的**交叉积**

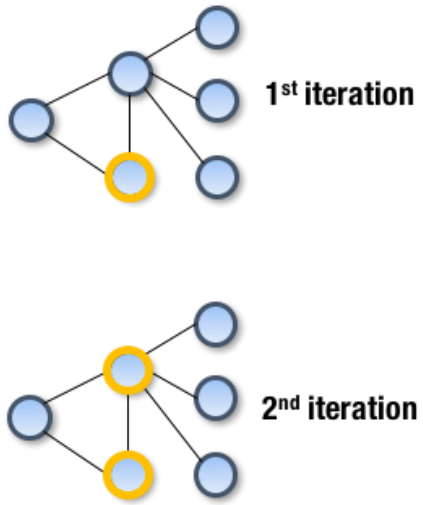
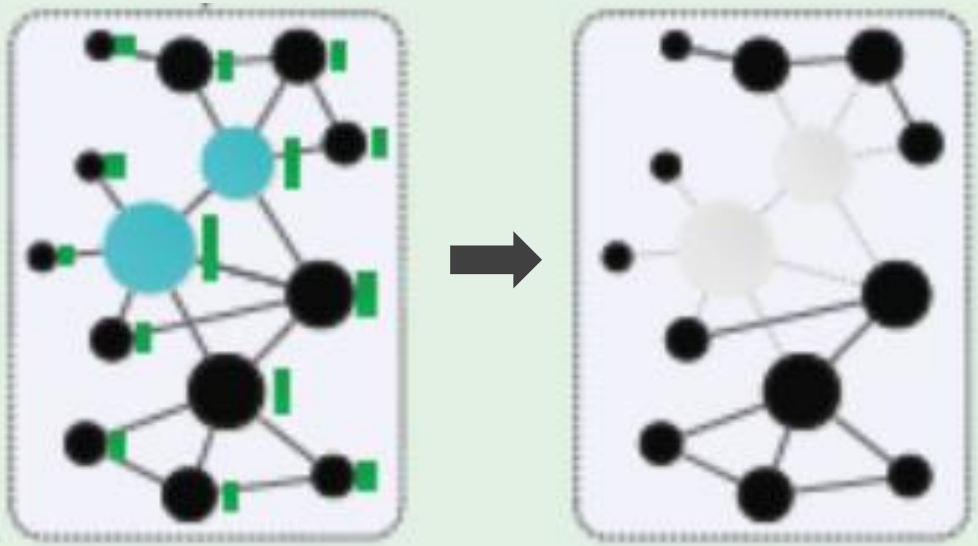
FINDER vs S2V-DQN

主要区别——训练损失函数

S2V-DQN	FINDER
$$(y - \widehat{Q}(h(S_t), v_t; \Theta))^2$$ $$y = \gamma \max_{v'} \widehat{Q}(h(S_{t+1}), v'; \Theta) + r(S_t, v_t)$$	$$\mathrm{Loss}(\Theta_Q, \Theta_{\mathrm{Embed}}) = \underbrace{\mathrm{E}_{(s_t, a_t, r_{t,t+n}, s_{t+n}) \sim U(D)}[(r_{t,t+n} + \gamma \max_{a'} \hat{Q}(s_{t+n}, a'; \hat{\Theta}_Q) - Q(s_t, a_t; \Theta_Q))^2]}_{\text{Q-learning loss}} + \underbrace{\alpha \sum_{i,j=1}^{N} s_{i,j} \|y_i - y_j; \Theta_{\mathrm{Embed}}\|_2^2}_{\text{Graph reconstruction loss}}$$
贝尔曼损失	增加了图重构损失

FINDER vs S2V-DQN

主要区别——测试策略

S2V-DQN	FINDER
	
一步选一个点	一步选一批点

FINDER vs S2V-DQN

主要区别——消融实验

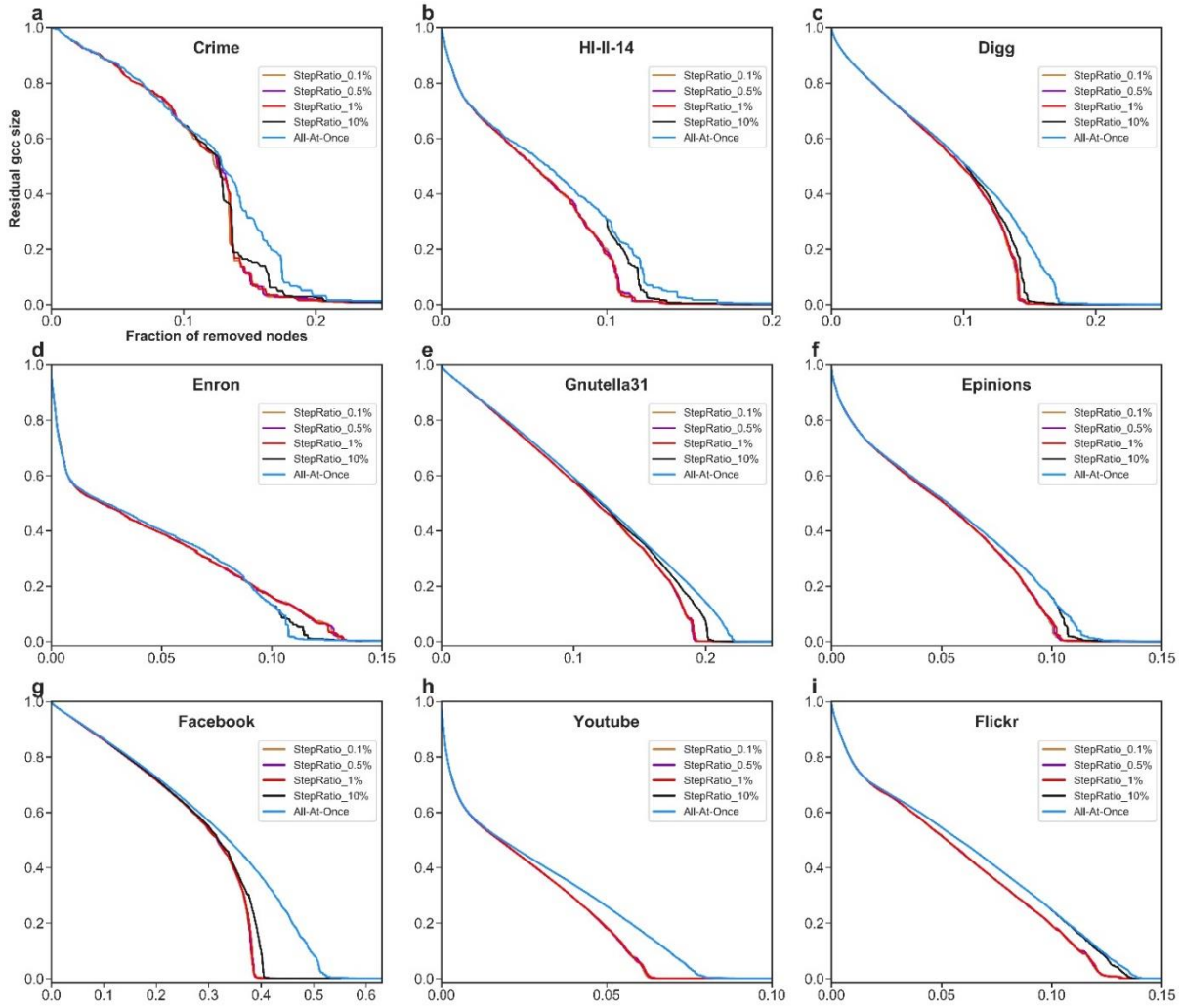
ANC (×0.01)	Crime	HI-II-14	Digg	Enron	Gnutella31	Epinions	Facebook	Youtube	Flickr
FINDER	10.99	**5.54**	**8.66**	**4.30**	11.10	**4.99**	26.84	**2.37**	**5.46**
No-GraphSAGE	11.39	5.93	9.23	34.15	11.36	6.31	43.33	3.04	5.88
No-VirtualNode	12.18	9.91	13.31	12.33	11.64	9.90	**26.71**	7.45	9.05
No-CrossProduct	11.00	6.49	9.56	8.85	**10.50**	8.17	31.55	3.95	7.77
No-ReconsLoss	**10.97**	5.80	8.70	48.41	10.55	5.29	47.89	2.53	6.08
S2V-DQN	12.70	6.43	11.27	13.64	11.69	24.74	29.95	18.92	38.34

FINDER vs S2V-DQN

主要区别——消融实验

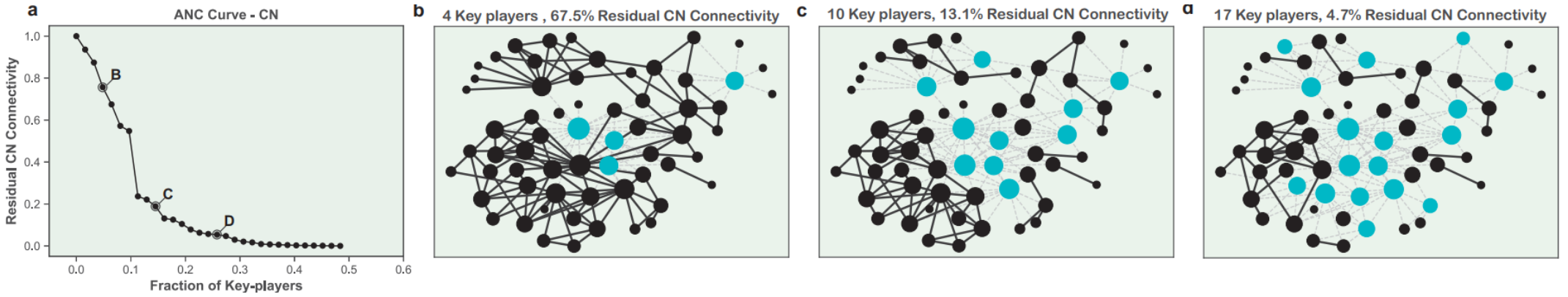
测试阶段每次批量移除1%

几乎不会对结果有影响。



FINDER

- **评价指标：ANC曲线下的面积** $R(v_1, v_2, \cdots, v_N) = \frac{1}{N}\sum_{k=1}^{N}\frac{\sigma(\mathcal{G}\backslash\{v_1, v_2, \cdots, v_k\})}{\sigma(\mathcal{G})}$



- **实验场景：两种连通性（CN,ND）,三种节点赋权方式（无权、度权、随机权）,共六种实验场景。**

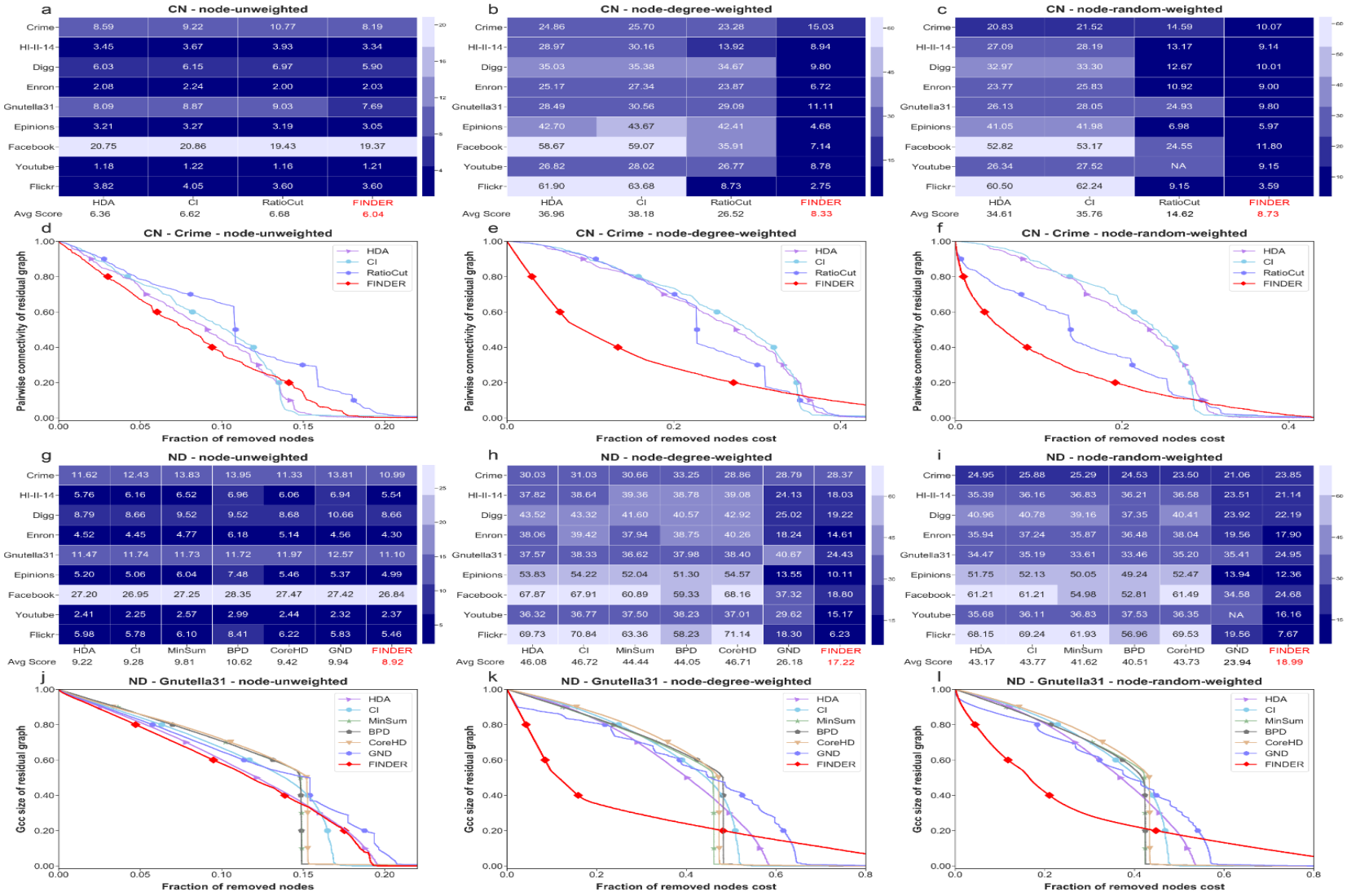
$$\sigma_{\text{pair}}(\mathcal{G}) = \sum_{C_i \in \mathcal{G}} \frac{\delta_i(\delta_i - 1)}{2} \qquad \sigma_{\text{gcc}}(\mathcal{G}) = \max\{\delta_i; C_i \in \mathcal{G}\}$$

FINDER

六种不同的网络

瓦解场景中效果

平均提升28.7%



FINDER

最大网络达百万节点，千万边

Statistics	Crime	HI-II-14	Digg	Enron	Gnutella31	Epinions	Facebook	Youtube	Flickr
Nodes num (N)	829	4,165	29,652	33,696	62,561	75,877	63,392	1,134,890	1,624,991
Edges num (E)	1,473	13,087	84,781	180,811	147,878	405,739	816,831	2,987,624	15,473,043
maximum degree	25	286	310	1,383	95	3,044	1,098	28,754	27,224
average degree	3.55	6.28	5.72	10.73	4.73	10.69	25.77	5.27	19.04
diameter	10	11	5.72	11	11	15	15	24	24
mean shortest path length	5.04	4.16	4.68	4.03	5.96	4.40	4.31	5.55	5.19
clustering coefficient	0.0058	0.0444	0.0054	0.5092	0.0055	0.1378	0.2218	0.0808	0.1892
assortativity	-0.1645	-0.2016	0.0027	-0.1165	-0.0927	-0.0406	0.1768	-0.0369	-0.0167

FINDER

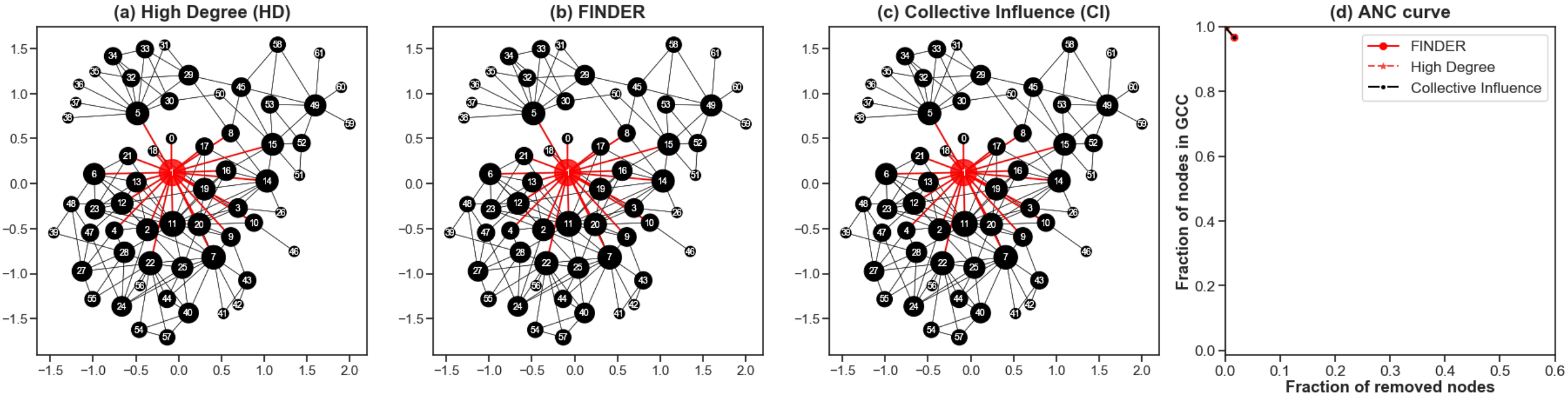
计算效率提升超过两个数量级

Nature 2016

PNAS 2019

Time (s)	Crime	HI-II-14	Digg	Enron	Gnutella31	Epinions	Facebook	Youtube	Flickr
HDA	**0.0**	**0.0**	**0.3**	**2.5**	**4.5**	**7.9**	**12.8**	1,963.8	5,098.2
CI	0.0	0.5	4.9	70.7	9.4	668.4	2,723.2	7,056.4	>5d
MinSum	2.0	20.0	108.2	323.6	225.4	681.1	1,408.0	5,478.2	35,215.2
BPD	0.3	9.0	39.0	86.0	41.0	254.0	839.0	462.0	17,942.0
CoreHD	0.1	2.0	8.5	36.2	11.9	142.9	214.8	2,018.9	36,837.3
GND	0.1	1.3	12.7	35.7	386.3	233.1	4,614.5	52,246.2	>5d
FINDER	0.3	1.2	7.1	12.7	13.3	22.6	73.4	**281.22**	**989.66**

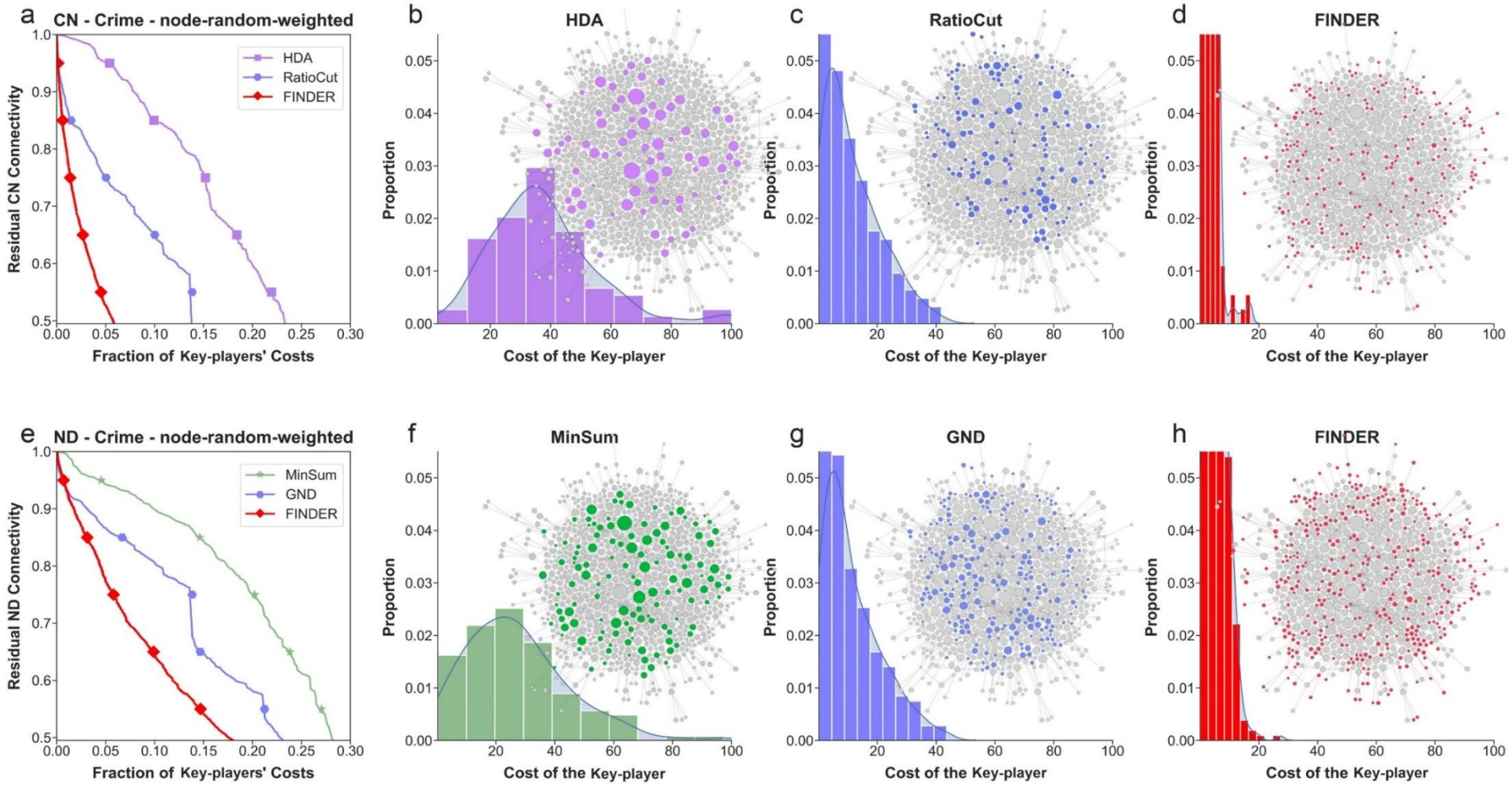
FINDER



FINDER自己发现了更快地瓦解复杂网络的策略

FINDER

FINDER优先选择 cost-effective 节点，即代价不那么大，却能达到相同的瓦解效果。



FINDER

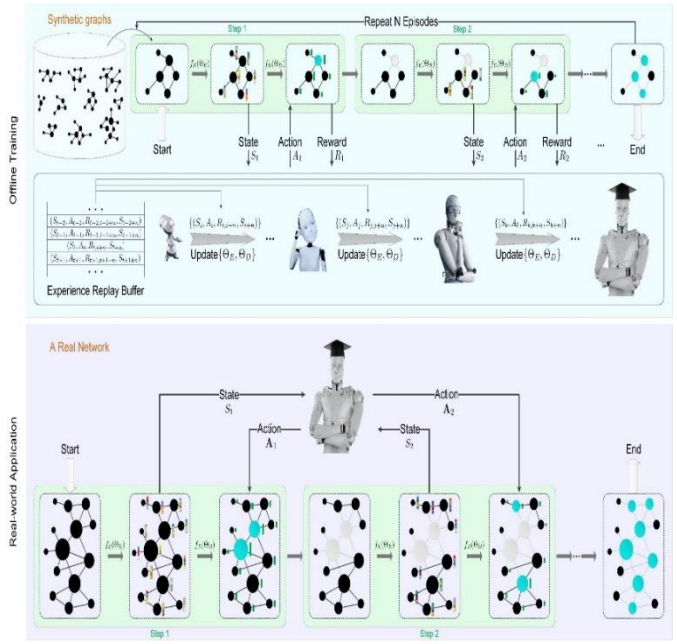
结论

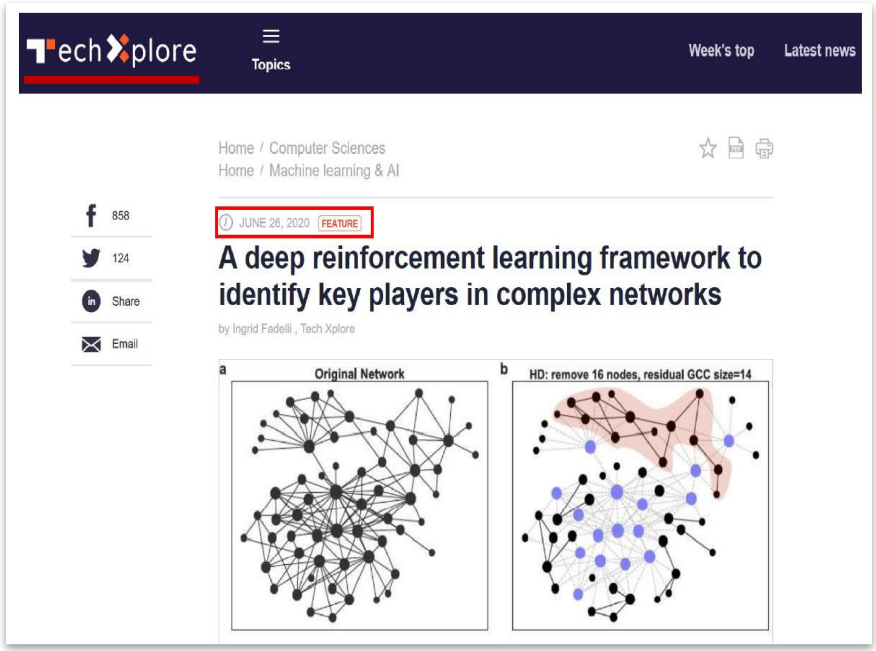
（1）FINDER在网络瓦解问题取得优异表现，速度快且质量高；

（2）FINDER是一个通用的求解框架，适用于不同的瓦解场景；

（3）FINDER在小规模模拟图（50个节点的BA图）上训练，可以在大规模真实网络（百万节点的不同类型真实图）表现达到SOTA，证明其强大的泛化能力。

FINDER



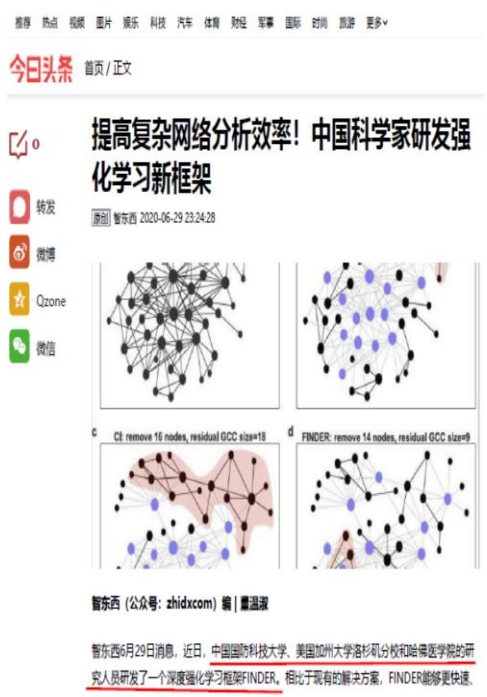


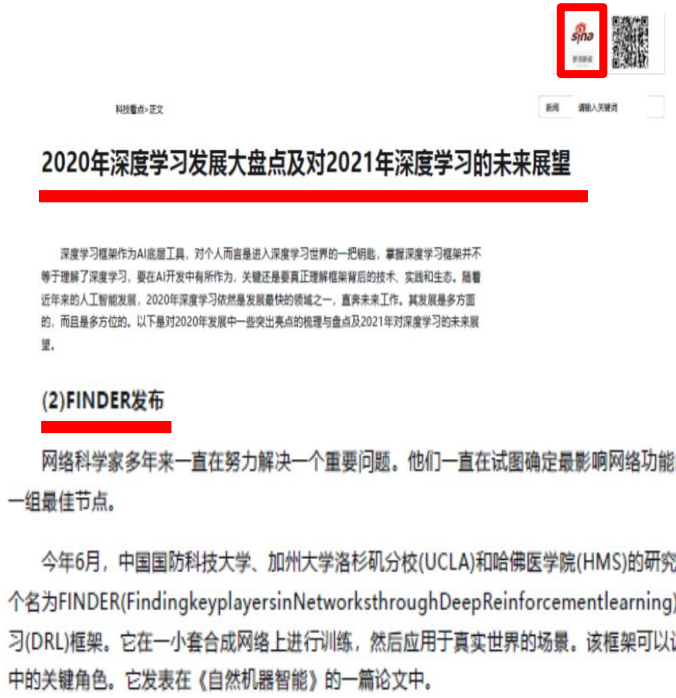
工作被国外知名科技媒体TechXplore以亮点新闻报道

TechXplore是ScienceX旗下的科技网站，刊载内容主要涵盖工程、电子和技术， 发展等前沿领域最新研究，提供每日科学、研究、技术的高质 量新闻媒体，覆盖面广，每个月全球读者多达500万，包括科学家、研究人员和 工程师等。

FINDER

研究成果被科技日报、今日头条、新浪新闻、集智俱乐部等国内外知名媒体、公众号报道，新浪新闻将该工作列为2020年全球深度学习发展的突出亮点。





FINDER

国家重点研发计划指南将该工作列为主流基准算法

“网络空间安全治理”重点专项
2022 年度项目申报指南

为落实“十四五”期间国家科技创新有关部署安排，国家重点研发计划启动实施“网络空间安全治理”重点专项。根据本重点专项实施方案的部署，现发布 2022 年度项目申报指南。

本重点专项总体目标是：围绕全球网络公害、涉及民生的数据资产和“新基建”基础设施等领域的安全挑战，开展互联网基

1.5 受限感知能力下社交网络虚假信息检测与溯源方法（青年科学家项目）

研究内容：针对虚假信息防控需求，分析虚假信息在不同类型社交网络内部与跨平台传播特性，建立典型社交网络平台虚假信息传播的动力学模型；研究社交网络结构特征对虚假信息传播的影响机理，设计面向信息传播的超大规模网络结构拆解算法；

考核指标：建立不少于 3 种典型社交网络的虚假信息传播模型；虚假信息检测准确率不小于 80%；设计超大规模网络拆解算法不小于 5 个，实现千万级网络拆解内存消耗不超过 64GB，计算时间不超过 10 分钟，综合性能优于 CI、FINDER 等主流算法；侦测典型网络高传播风险虚假信息所需监控的网络节点占比不高于 10%，发现时平均传播范围低于 1%；给定信息资源场景下的溯源定位源集不大于网络规模的 1%；关键考核指标达到同期国际先进水平。

FINDER

2023世界人工智能大会青年优秀论文奖拟获奖名单

(按论文英文题目首字母排序，共10篇)

1. ADMM-CSNet: A Deep Learning Approach for Image Compressive Sensing，杨燕，西安交通大学，IEEE Transactions on Pattern Analysis and Machine Intelligence 2020

2.Finding key players in complex networks through deep reinforcement learning，范长俊，国防科技大学，Nature Machine Intelligence 2020

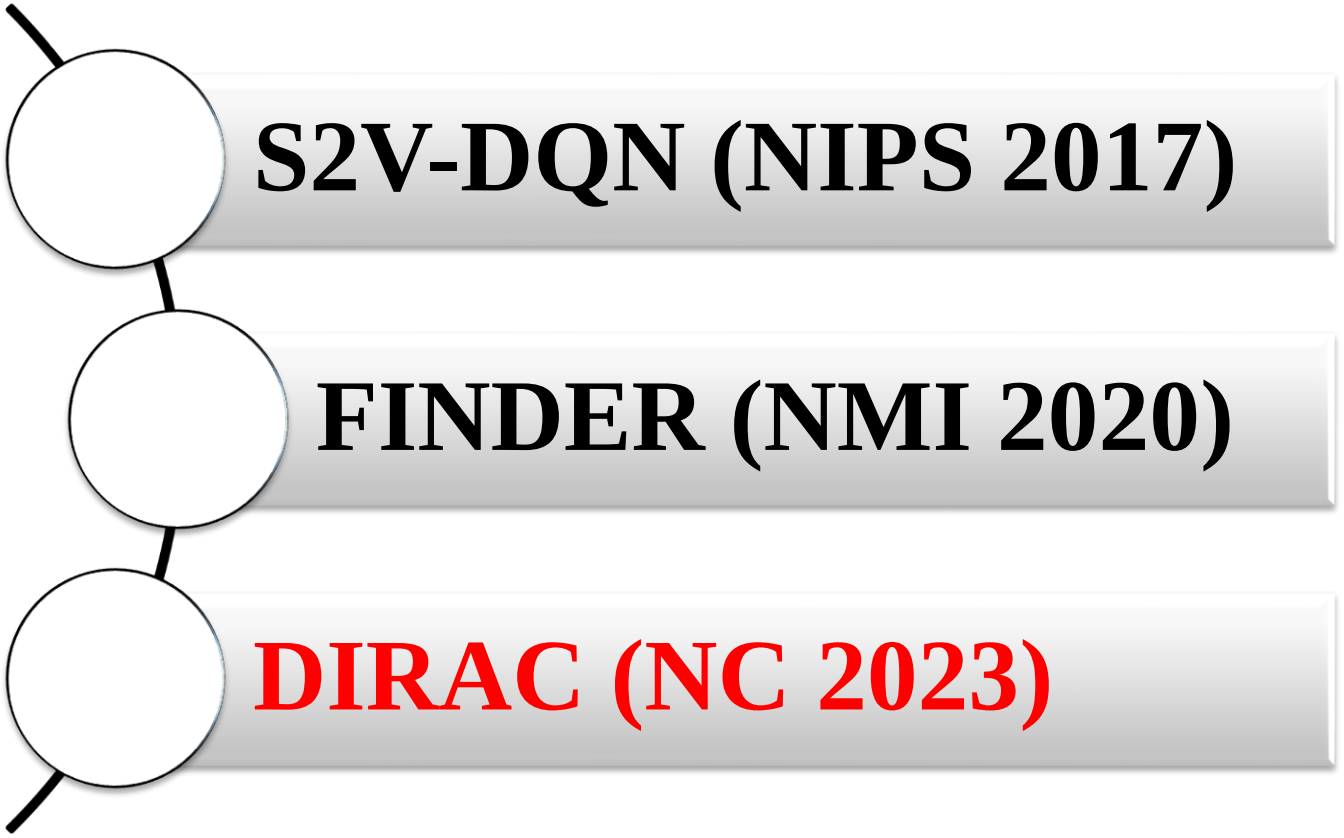
3. Identifying degradation patterns of lithium ion batteries from impedance spectroscopy using machine learning，张云蔚，剑桥大学，Nature Communications 2020

4. Parameter-efficient fine-tuning of large-scale pre-trained language models，丁宁，清华大学，Nature Machine Intelligence 2023

图上的组合优化问题求解

自回归方法

图表示学习+强化学习



DIRAC

nature communications

Article

https://doi.org/10.1038/s41467-023-36363-w

Searching for spin glass ground states through deep reinforcement learning

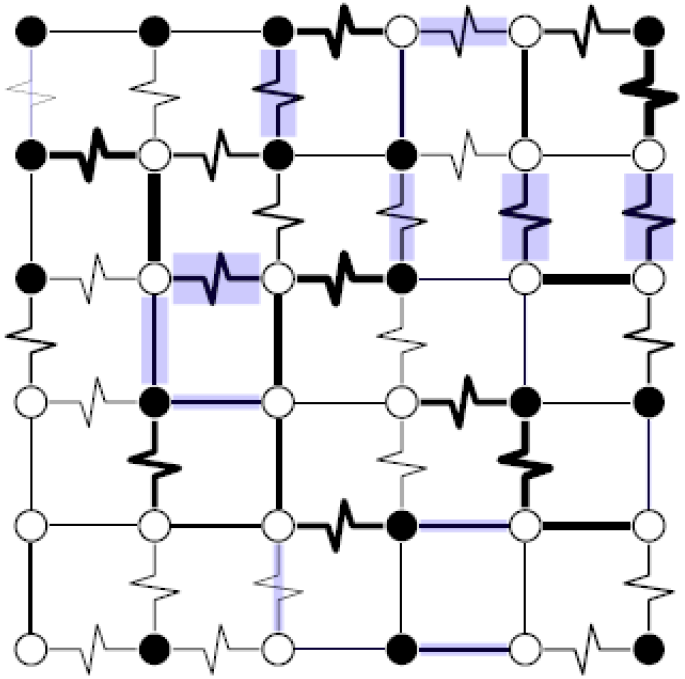
Received: 18 December 2022

Accepted: 25 January 2023

Changjun Fan[1,8], Mutian Shen[2,8], Zohar Nussinov[2,3,4], Zhong Liu[1], Yizhou Sun[5] & Yang-Yu Liu[6,7]

自旋玻璃的基态求解（Spin glass ground states ）

- 每个节点一个状态$\sigma_i \in \{+1,-1\}$（黑色实心表示+1，白色空心表示-1）；
- 每条边一个权$J_{ij} \in N(0,1)$（直线表示正权边，折线表示负权边，边的宽度和$|J_{ij}|$呈正比，蓝色阴影部分表示边的能量$-J_{ij}\sigma_i\sigma_j$为正）。



问题定义：确定最优的节点状态，使得以下目标函数值最低：

$$\mathcal{H} = -\sum_{\langle i,j \rangle} J_{ij}\sigma_i\sigma_j.$$

典型的组合优化问题，可以把lattice看成一种特殊的图结构。

DIRAC

研究意义

1. 该问题是**揭开自旋玻璃奇怪而复杂行为的关键**，特别是在不同边界条件下的基态能量可以用来计算自旋玻璃的刚度指数，从而确保有限温度下自旋玻璃相的存在；
2. 该问题**有助于解决许多组合优化难题**，因为这些难题中都存在自旋玻璃的形式；
3. 该问题**有助于神经网络优化工具的开发**，有关自旋玻璃及其基态的研究推动了空腔法（cavity method）和置信传播（Belief Propagation）等优化工具的发展，并将进一步推动计算复杂性理论。

DIRAC

[HTML] Ising formulations of many NP problems

A Lucas - Frontiers in physics, 2014 - frontiersin.org

… maps from partitioning and satisfiability to an Ising spin glass. In particular, we will describe how “all of the famous NP problems” 5 Karp [18], Garey and Johnson [19] can be written …

☆ 保存 99 引用 被引用次数：1639 相关文章 所有 9 个版本 导入BibTeX »

We provide Ising formulations for many NP-complete and NP-hard problems, including all of Karp's 21 NP-complete problems. This collects and extends mappings to the Ising model from partitioning, covering, and satisfiability. In each case, the required number of spins is at most cubic in the size of the problem. This work may be useful in designing adiabatic quantum optimization algorithms.

DIRAC

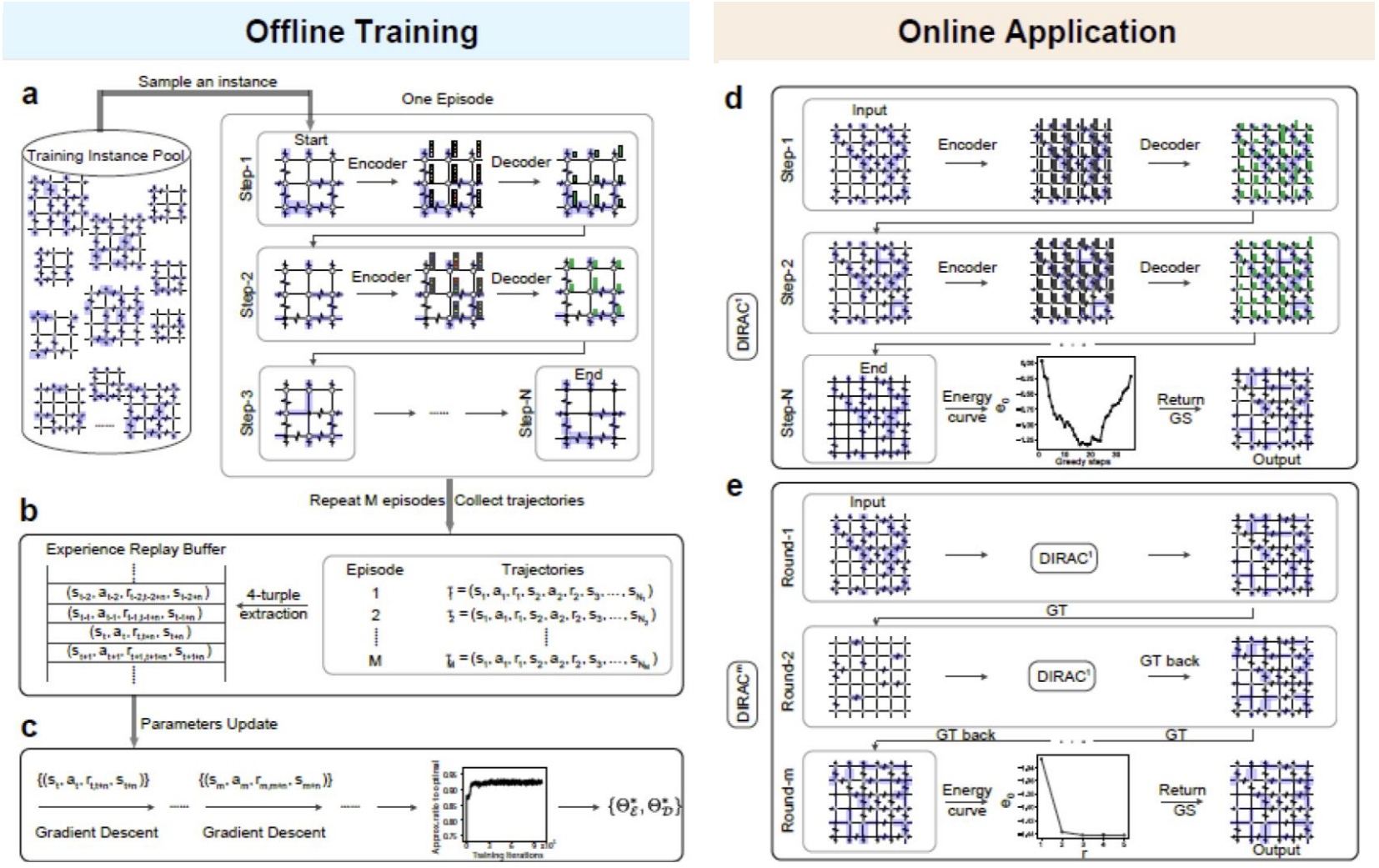
- **DIRAC** (**D**eep reinforcement learning for sp**I**n-glass g**R**ound-st**A**te **C**alculation)

优点：

- **精度高**，中等尺寸的系统可以达到全局最优解，且适用于三维及以上的高维系统；
- **可扩展性强**，最大系统尺寸达到三维的L=20，这是目前研究最大的系统；
- **可以和任意退火算法结合**，并进一步提高这些算法的表现。

DIRAC

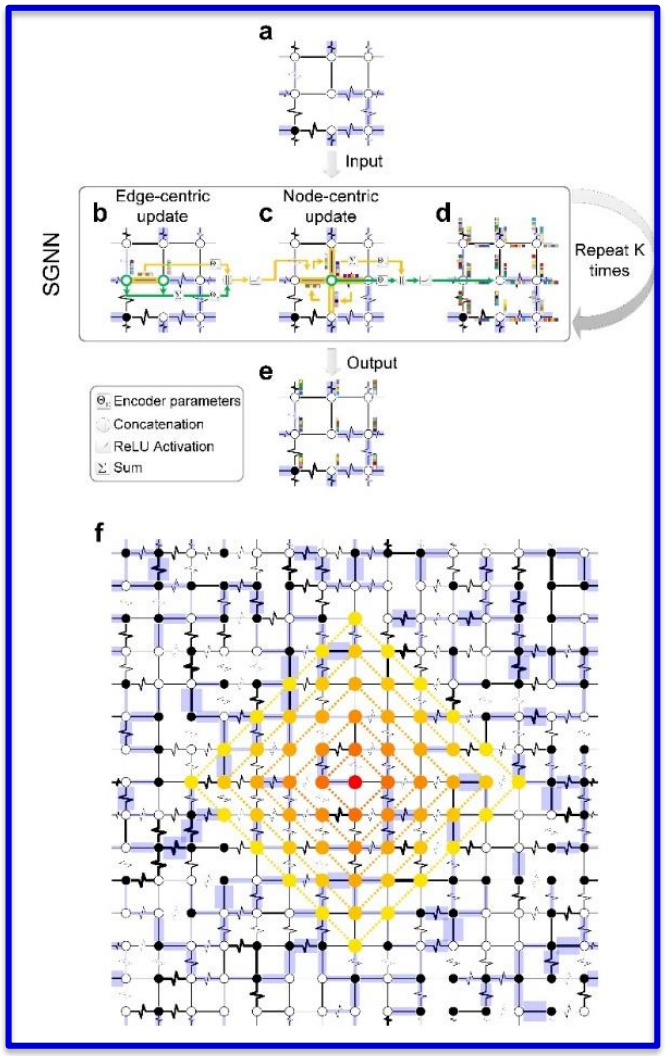
模型框架

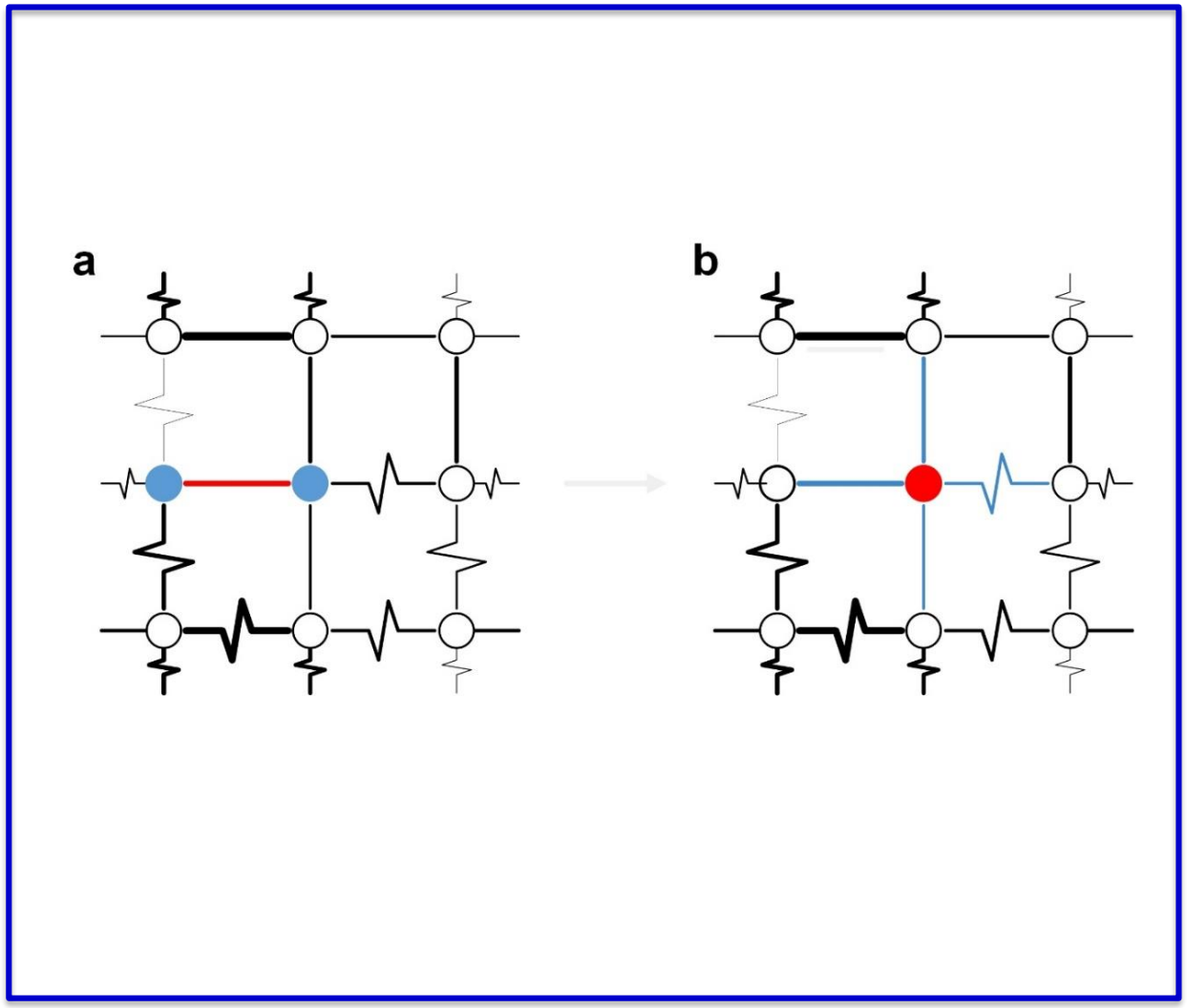

Offline Training
Online Application
a
Sample an instance
One Episode
Training Instance Pool
Start
Step-1
Step-2
Step-3
Step-N
End
Encoder
Decoder
b
Repeat M episodes
Collect trajectories
Experience Replay Buffer
4-turple extraction
Episode
Trajectories
c
Parameters Update
Gradient Descent
Gradient Descent
d
Input
Energy curve
Return GS
Output
e
Round-1
Round-2
Round-m
GT
GT back

DIRAC

主要模块

图表示学习



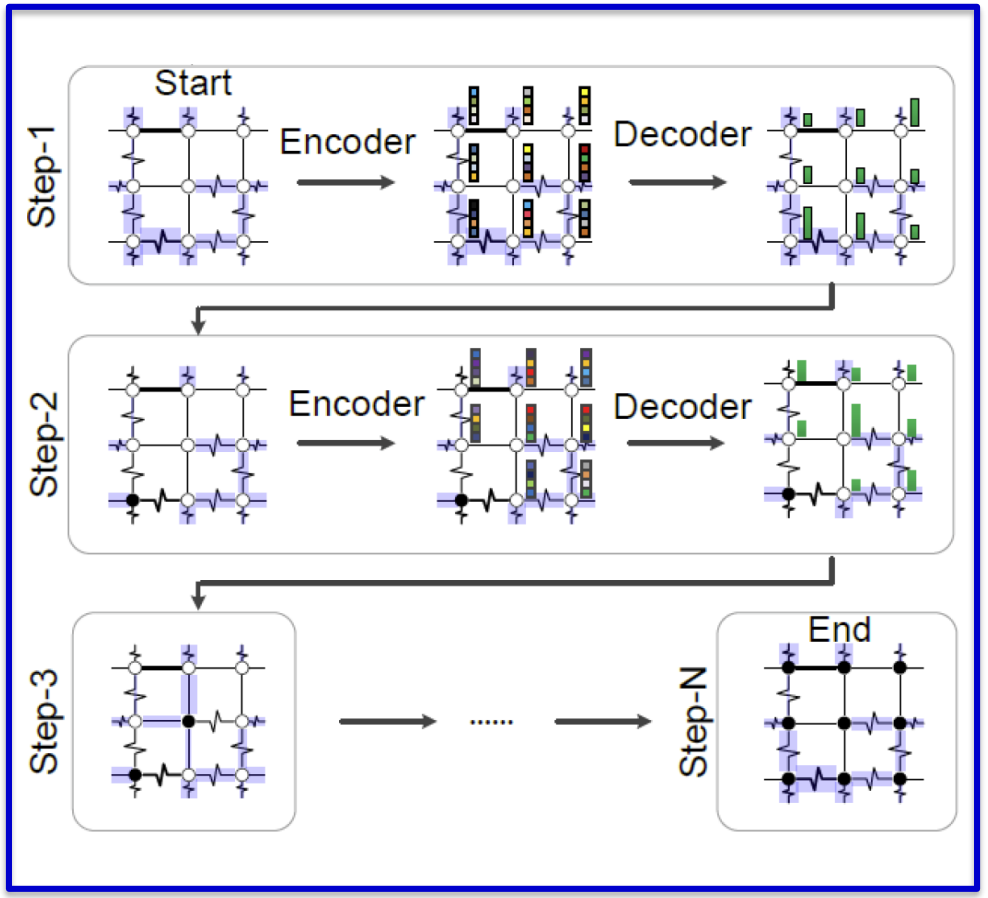


DIRAC

主要模块

强化学习

- **状态**：状态s代表观察到的自旋玻璃实例，包括自旋构型σ_i和耦合强度J_{ij}，在此基础上将选择最佳行动。
- **行动**：行动a(i)意味着翻转自旋i 。
- **奖励**：将自旋i从状态s翻转到一个新的状态s′之后的能量变化。
- **终止条件**：当主体将每个自旋翻转一次后，达成终态S_T 。



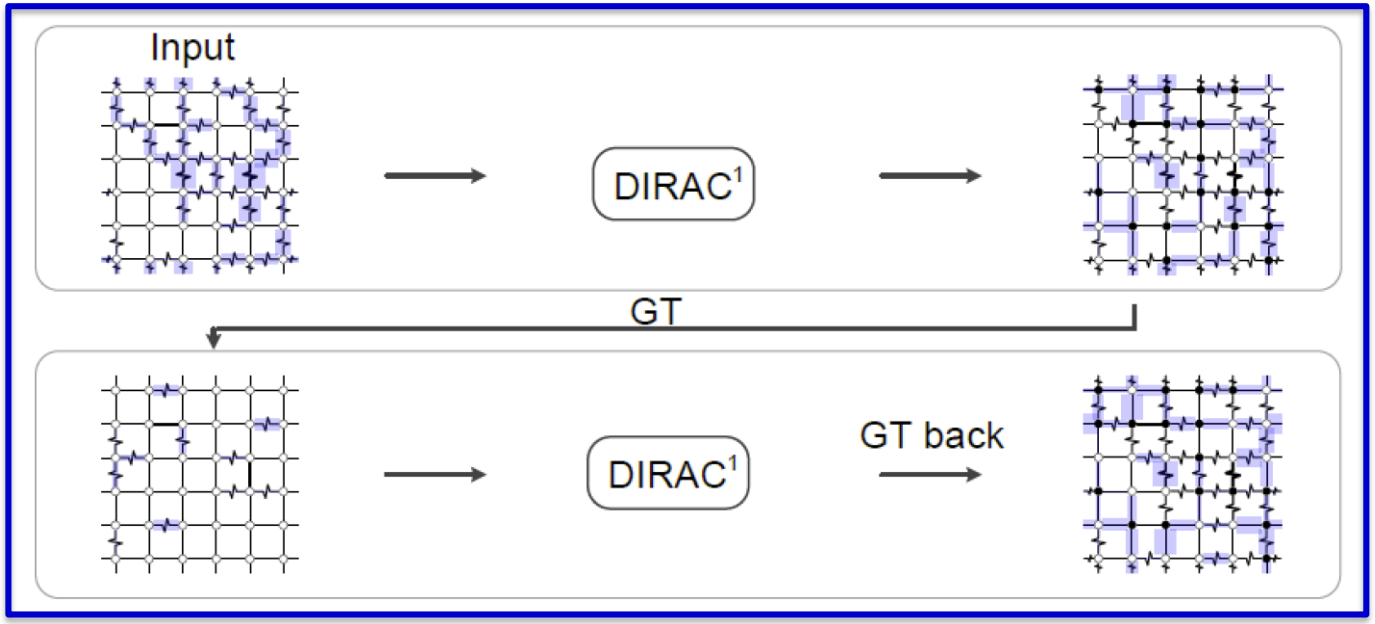
DIRAC

主要模块

规范变换（Gauge transformation）

$$J'_{ij} = J_{ij}t_i t_j, \sigma'_i = \sigma_i t_i$$

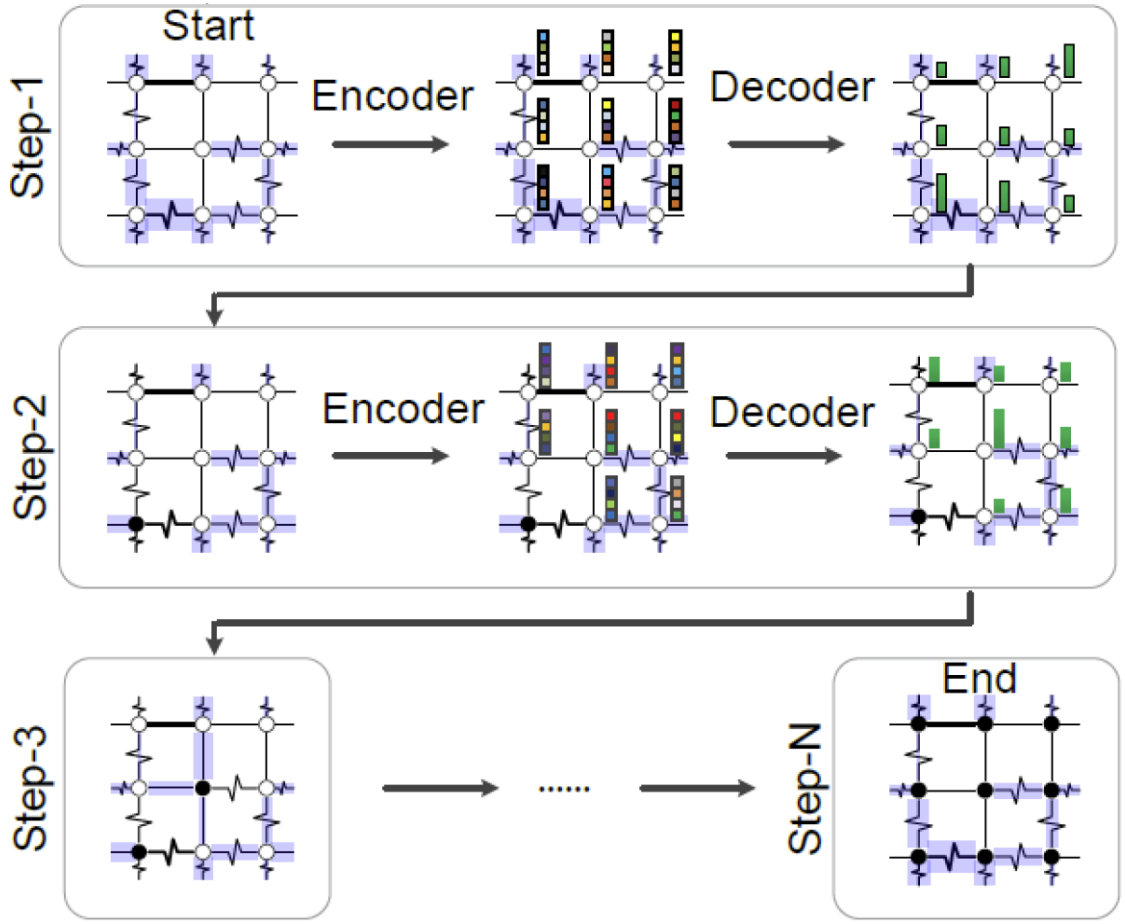
其中$t_i = \pm 1$是独立变量以便σ'_i可以取到任意想要的值。



DIRAC

为什么需要规范变换?

- **传统的强化学习工作流程（S2V-DQN）**：从某个状态出发，逐步增加元素（动作选择）构建解集，直到满足终止条件。
- **每个节点选择且仅选择一次。**



DIRAC

为什么需要规范变换?

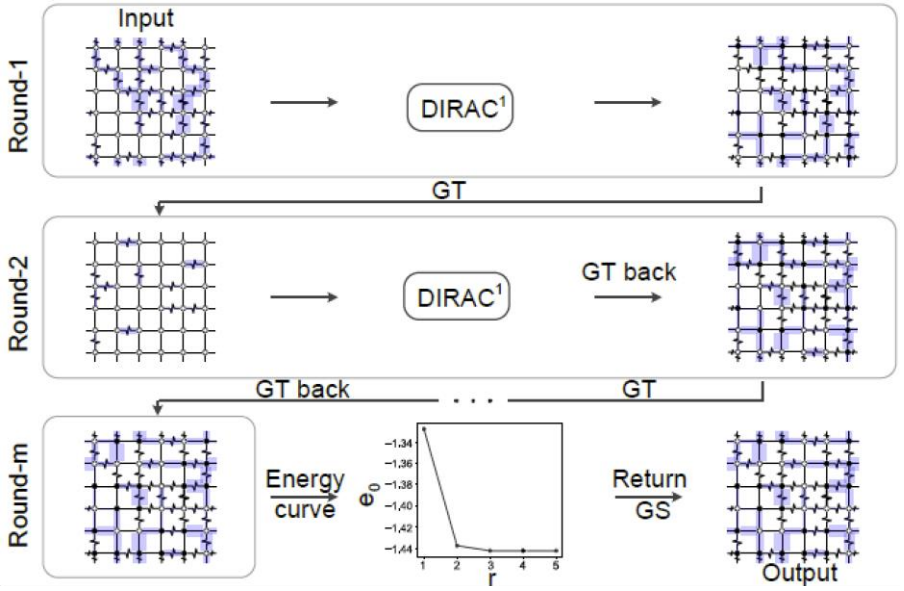
（1）这种方式最大的问题在于动作不可重复选择，即只能产生单个“最佳猜测”；

（2）组合优化问题巨大的探索空间要求智能体能够重复多次“试错探索”以避免陷入局部最优；

（3）规范变换能够将任意状态转化为指定的状态，从而适合智能体的重复探索。

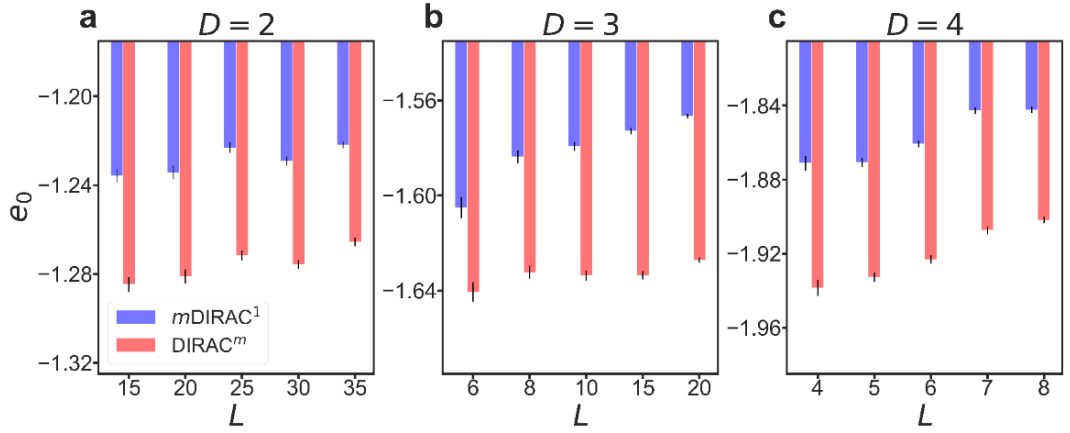
DIRAC

为什么需要规范变换?



■ **规范变换带来的好处:**

（1）允许模型在一次探索（DIRAC^1）结束后接着继续探索（DIRAC^m）

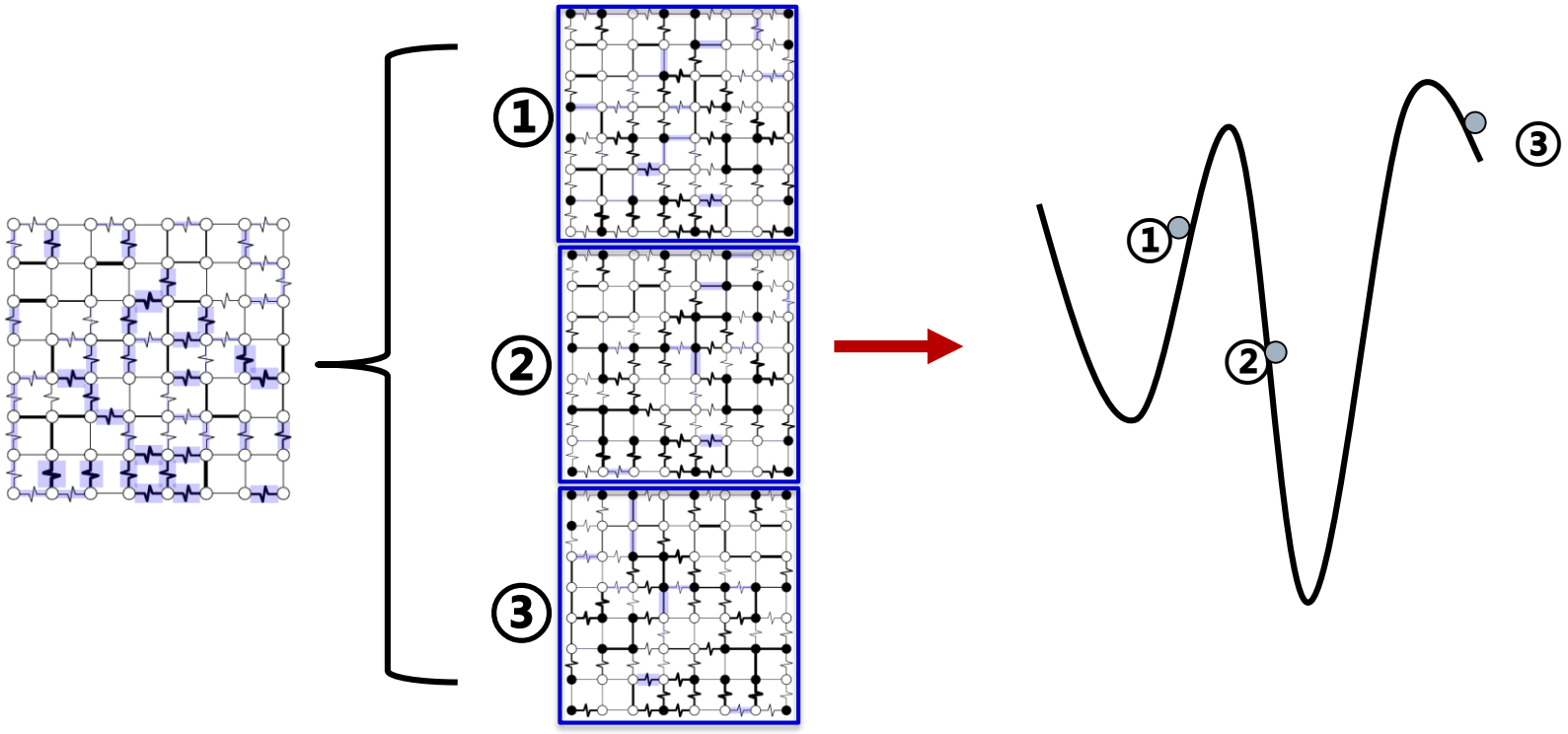


DIRAC

为什么需要规范变换?

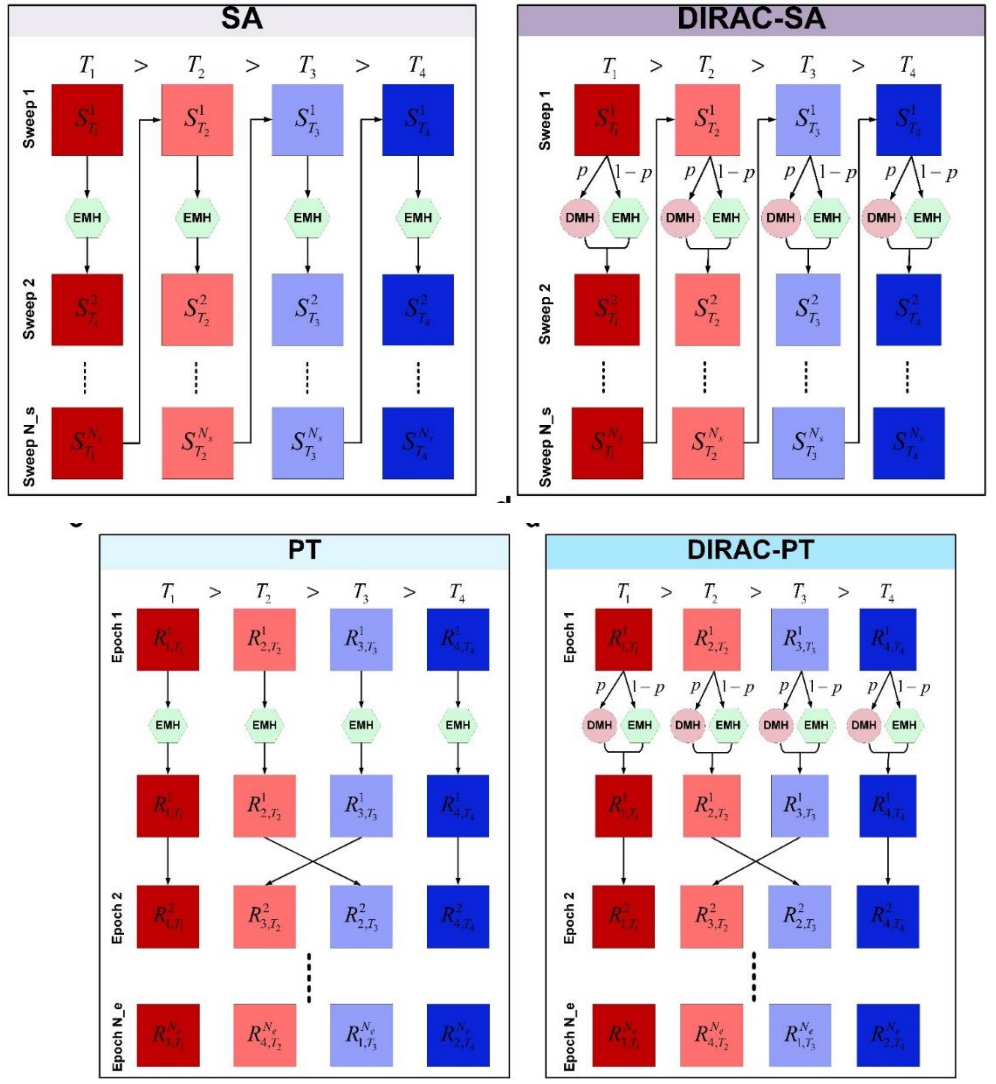
■ **规范变换带来的好处:**

（2）允许模型从多个不同初态出发，从而取得更优的结果;



DIRAC

为什么需要规范变换?



■ **规范变换带来的好处:**

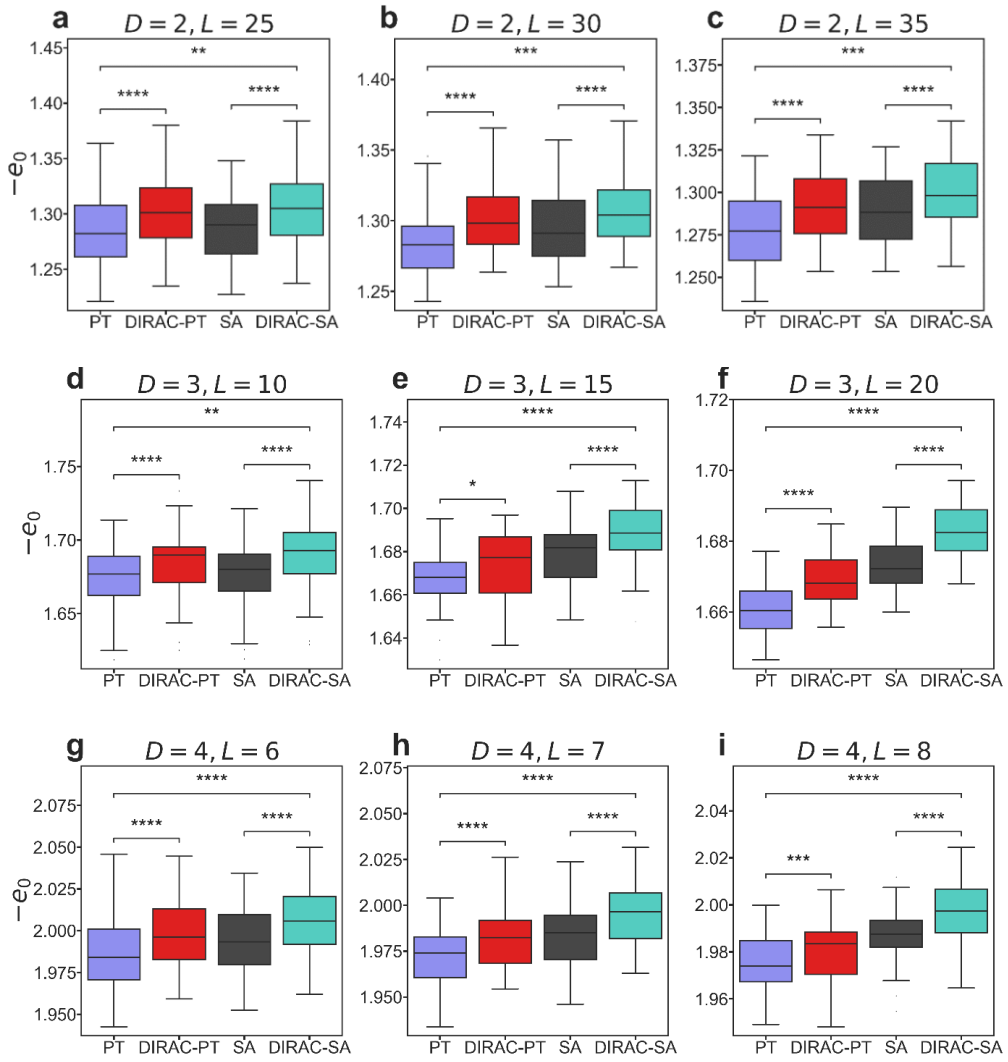
（3）允许模型和任意启发式算法结合，从而进一步提高启发式算法的结果;

DIRAC

为什么需要规范变换?

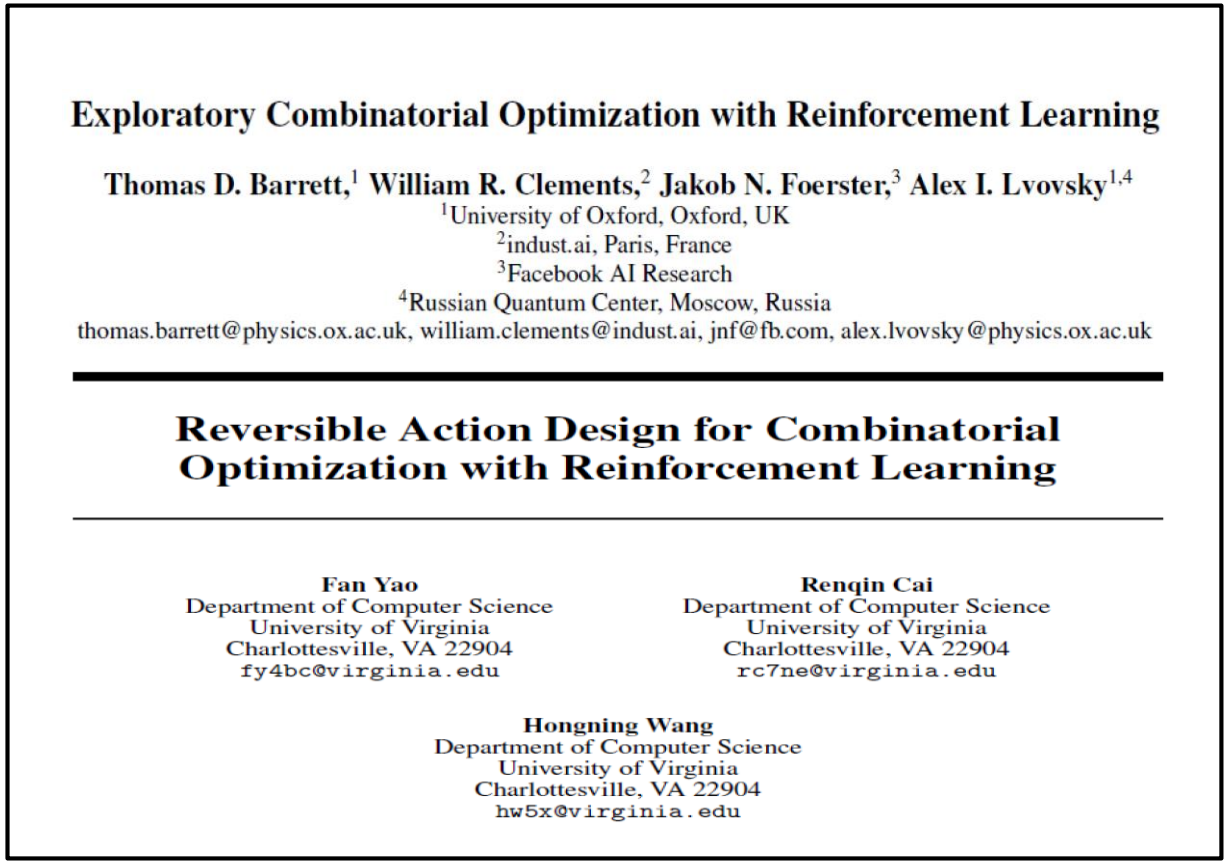
■ **规范变换带来的好处:**

（3）允许模型和任意启发式算法结合，从而进一步提高启发式算法的结果;



DIRAC

让强化学习重复探索是一个挑战，DIRAC用了1个公式，10行代码就解决了！



Exploratory Combinatorial Optimization with Reinforcement Learning

Thomas D. Barrett,[1] William R. Clements,[2] Jakob N. Foerster,[3] Alex I. Lvovsky[1,4]
[1]University of Oxford, Oxford, UK
[2]indust.ai, Paris, France
[3]Facebook AI Research
[4]Russian Quantum Center, Moscow, Russia
thomas.barrett@physics.ox.ac.uk, william.clements@indust.ai, jnf@fb.com, alex.lvovsky@physics.ox.ac.uk

Reversible Action Design for Combinatorial Optimization with Reinforcement Learning

Fan Yao
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
fy4bc@virginia.edu

Renqin Cai
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
rc7ne@virginia.edu

Hongning Wang
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
hw5x@virginia.edu

$$J'_{ij} = J_{ij} t_i t_j, \sigma'_i = \sigma_i t_i$$

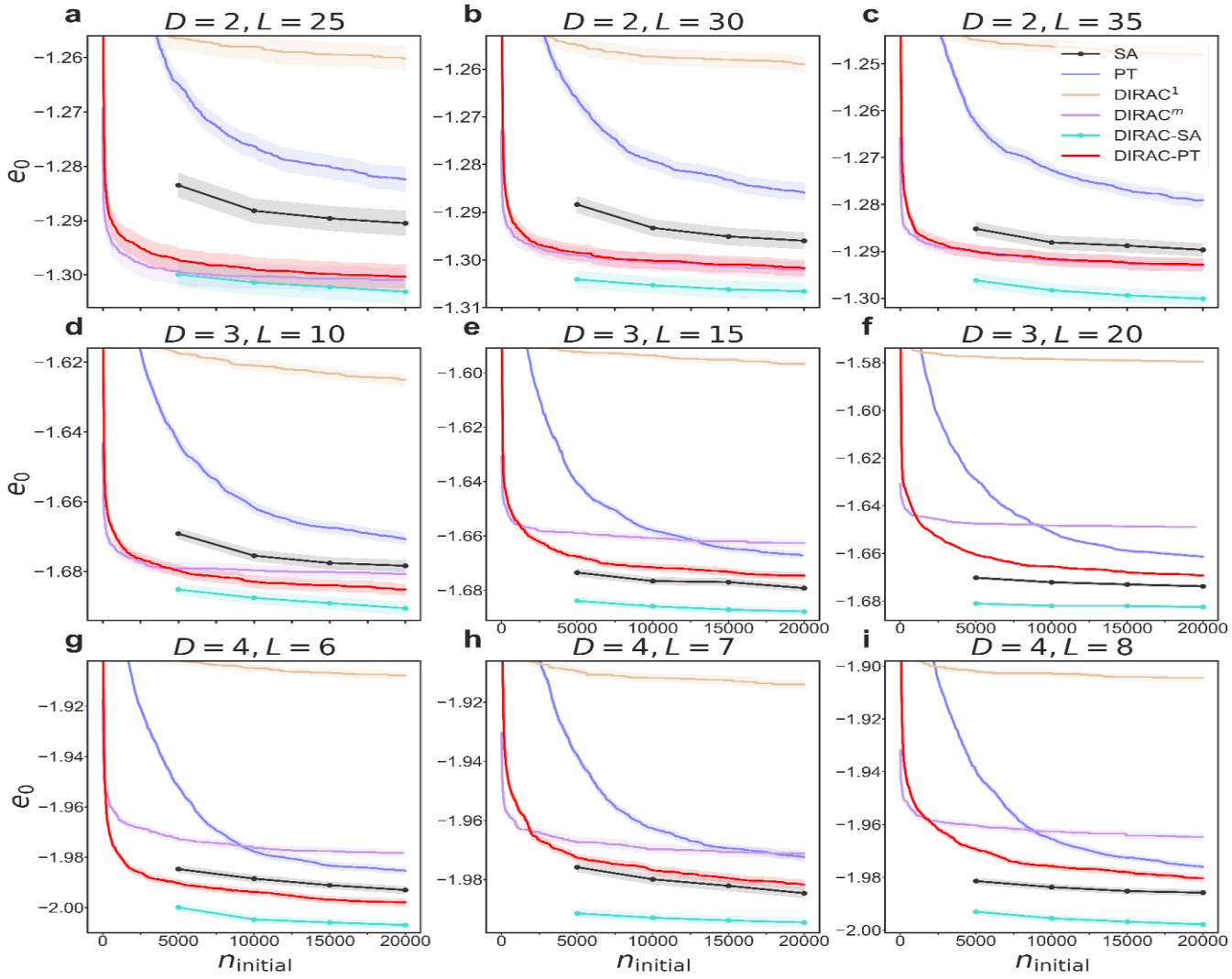
```
for node in g.nodes():
    g_temp.nodes[node]['state'] = 1
for edge in g_temp.edges():
    src, tgt = edge[0], edge[1]
    g_temp[src][tgt]['weight'] *= init_states[src] * init_states[tgt]

energy, states, _ = sg.Evaluate(g_temp, isNetworkx=1, step=step)  # Q
temp_states = []
for node in range(len(g_temp)):
    temp_states.append(states[node]/init_states[node])
init_states = temp_states
```

1个公式10行代码！

DIRAC

模型表现

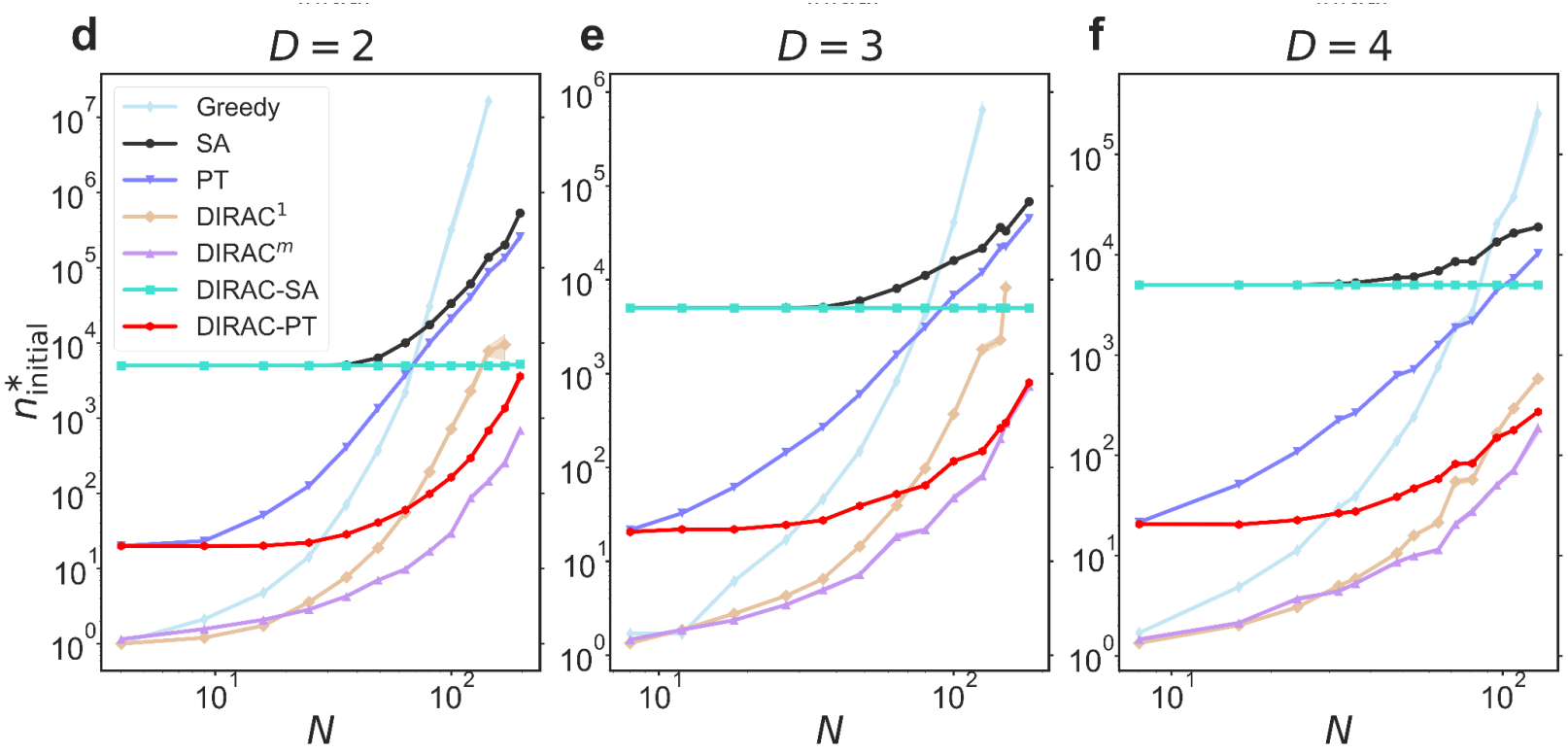
在大尺寸系统（基态未知），DIRAC相比目前最先进的退火算法，可以取得更低的能量。



DIRAC

模型表现

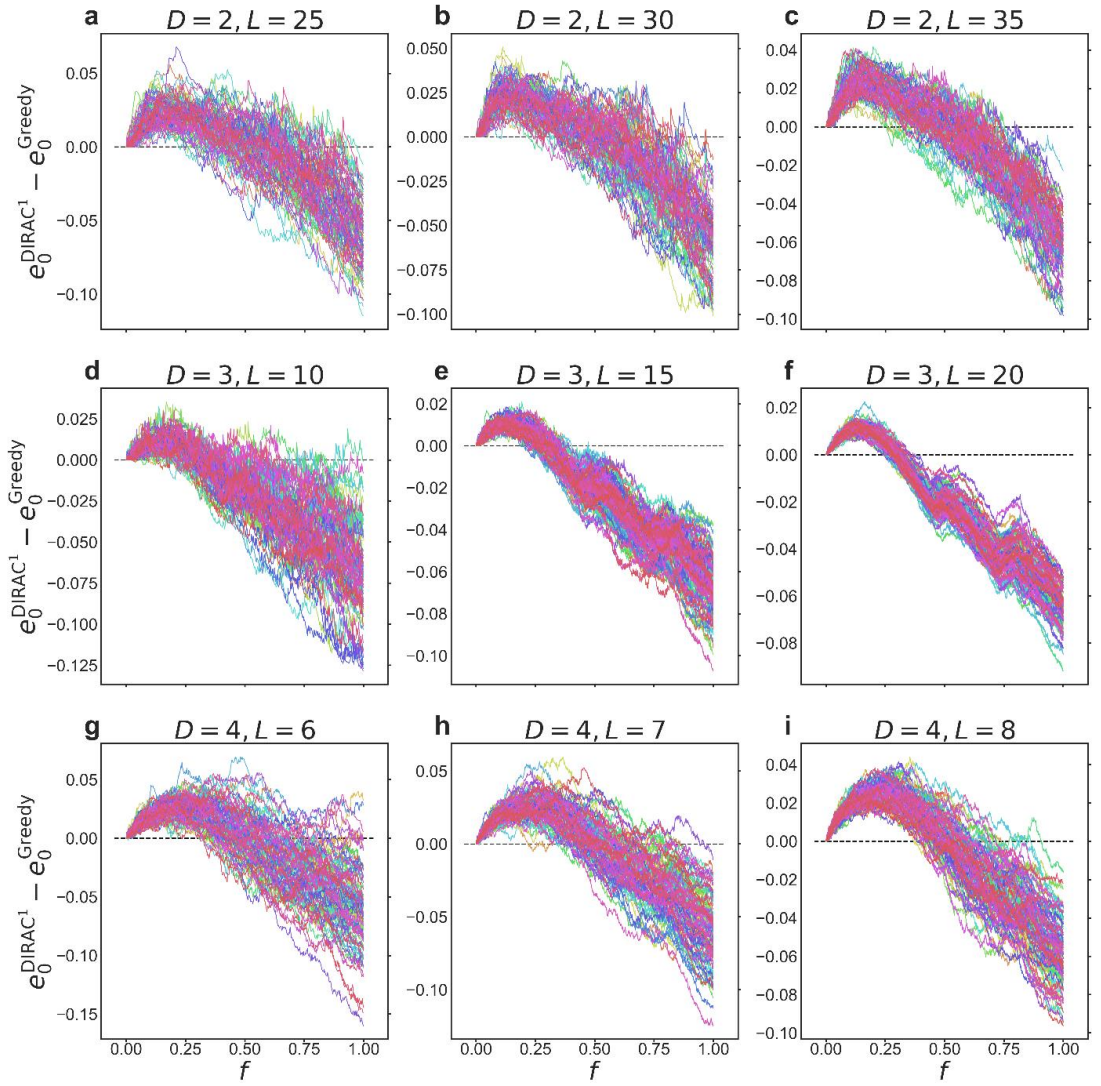
在小尺寸系统（基态已知），DIRAC均能求出全局最优解！



DIRAC

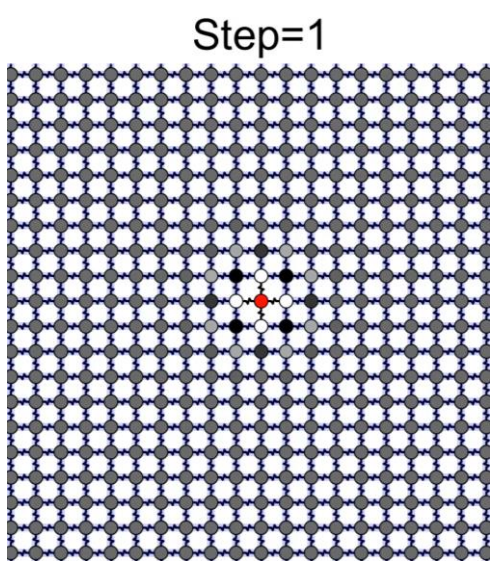
模型可解释

- 欲扬先抑
- 牺牲短期收益，获取长远收益
- 强化学习的优势

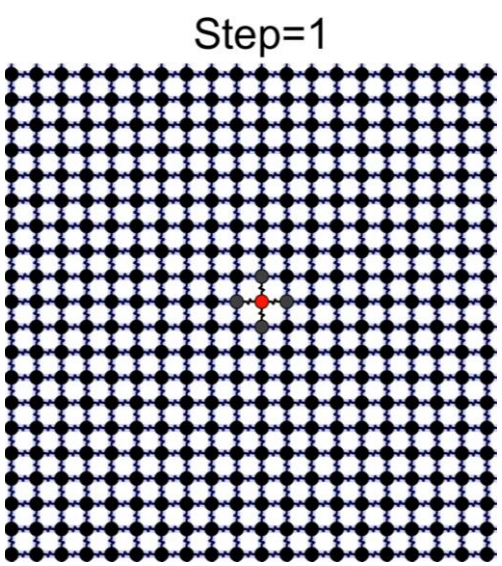


DIRAC

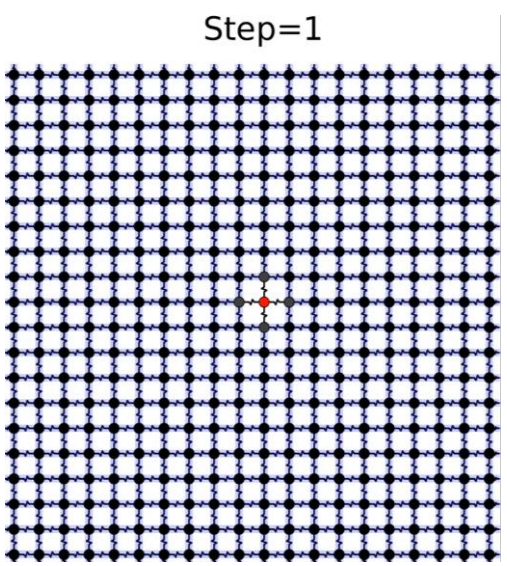
模型可解释



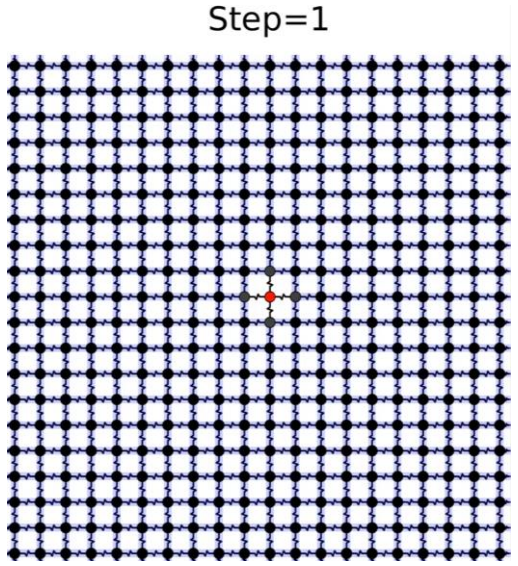
DIRAC



Greedy



PT



SA

图上的组合优化问题求解

自回归方法

“图表示学习+强化学习”范式

- **Pros:**
  - 模型泛化性能好
  - 模型效果好
  - 具有一定的可解释性
- **Cons:**
  - 模型设计、学习难
  - 实现困难

图上的组合优化问题求解

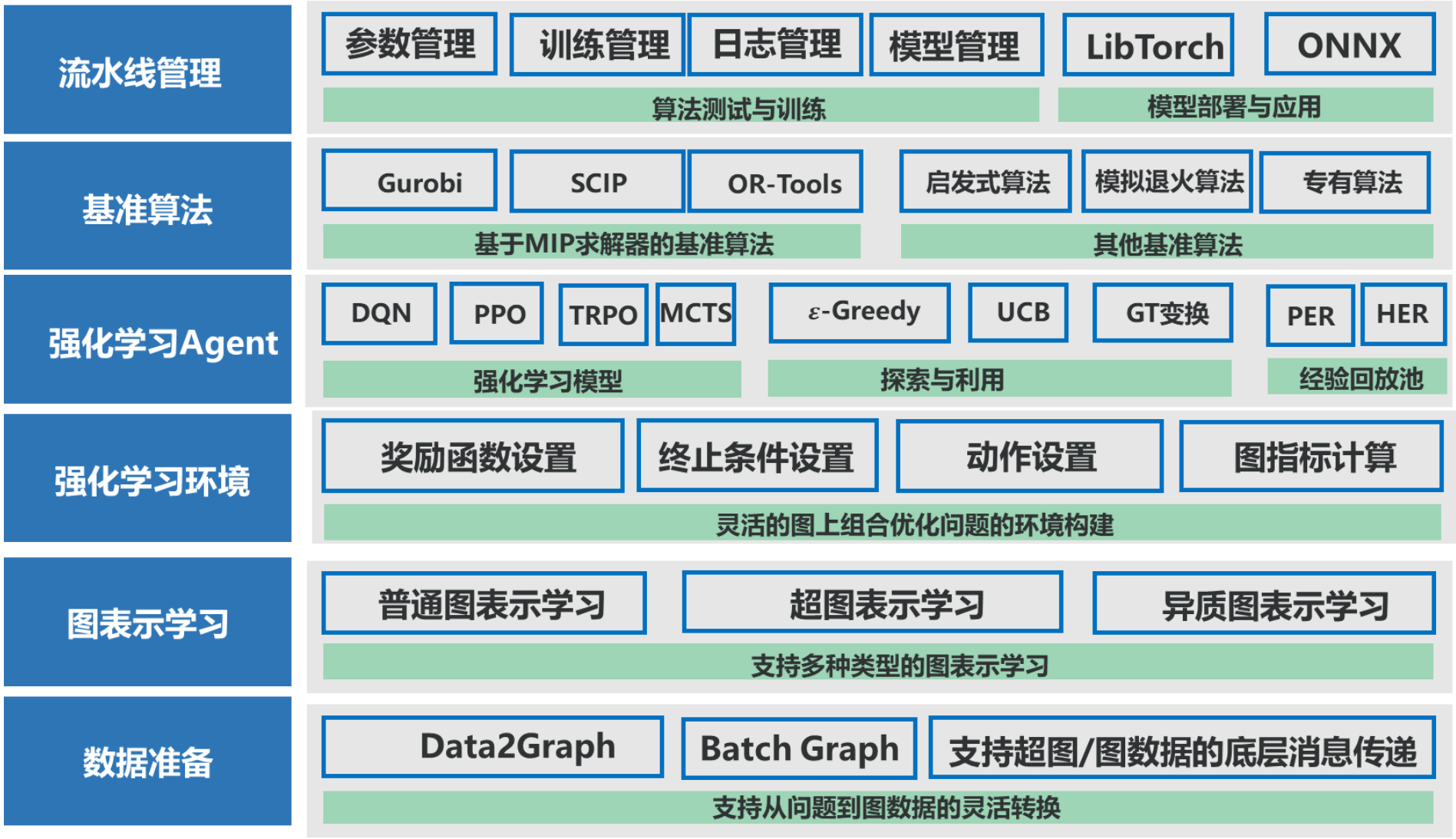
OptiGNN：基于图表示学习和深度强化学习的图上组合优化问题求解工具

A Python package for solving combinatorial optimization problems on graphs using Graph Neural Networks and Deep Reinforcement Learning

- ✓ 完全开源，基于Apache License 2.0协议
- ✓ 支持主流深度学习框架：Pytorch等
- ✓ 支持主流图表示学习框架：Torch-Geometric、DeepHypergraph、OpenHGNN
- ✓ 支持从问题数据到图数据的灵活转换，涵盖普通图、超图、异构图等类型。
- ✓ 支持多种组件灵活配置，可根据问题灵活定义强化学习的各个组件
- ✓ 支持基于Pipeline方式的训练测试灵活配置
- ✓ 支持算法模型到实际应用场景的模型部署与开发

图上的组合优化问题求解

OptiGNN 设计框架



目录

一些有趣的研究点

- 如何与精确算法结合?
- 如何与启发式算法结合?
- 如何提高在大规模实例的泛化表现?
- 如何处理复杂的约束?
- 如何与成熟求解器结合?
- 如何处理更自然的原始输入?

（1）如何与精确算法结合？

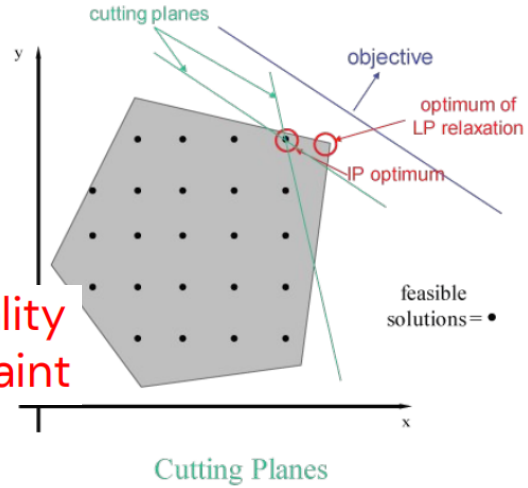
用神经网络求解混合整数规划

Mixed Integer Programming (MIP)

$$\min_x c^T x \quad \text{Objective function}$$

$$s.t. \quad Ax \leq b \quad \text{Linear constraints}$$

$$x_i \in \mathbb{Z}, \quad i \in \mathcal{I} \quad \text{Integrality constraint}$$

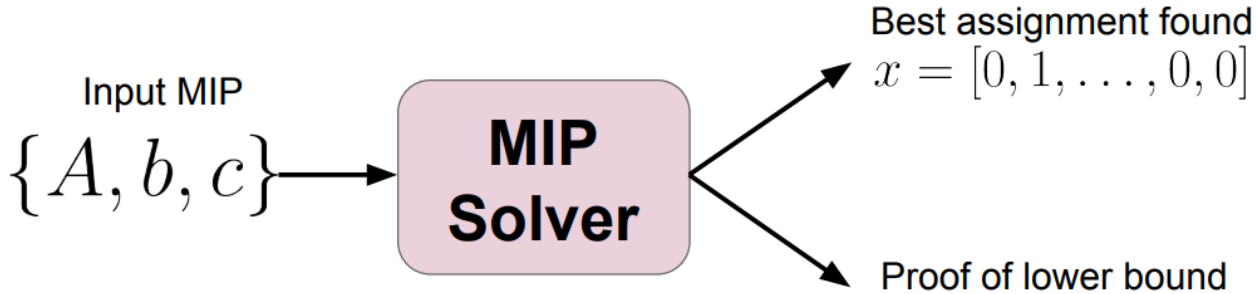


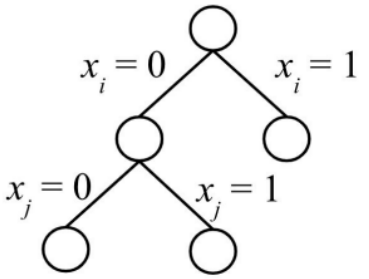
- "Mixed" → x can also contain continuous variables
- Many real-world applications!

（1）如何与精确算法结合？

用神经网络求解混合整数规划

$$\begin{aligned} \text{minimize} \quad & c^\top x \\ \text{subject to} \quad & Ax \le b \\ & l \le x \le u \\ & x_i \in \mathbb{Z}, \quad i \in \mathcal{I} \end{aligned}$$





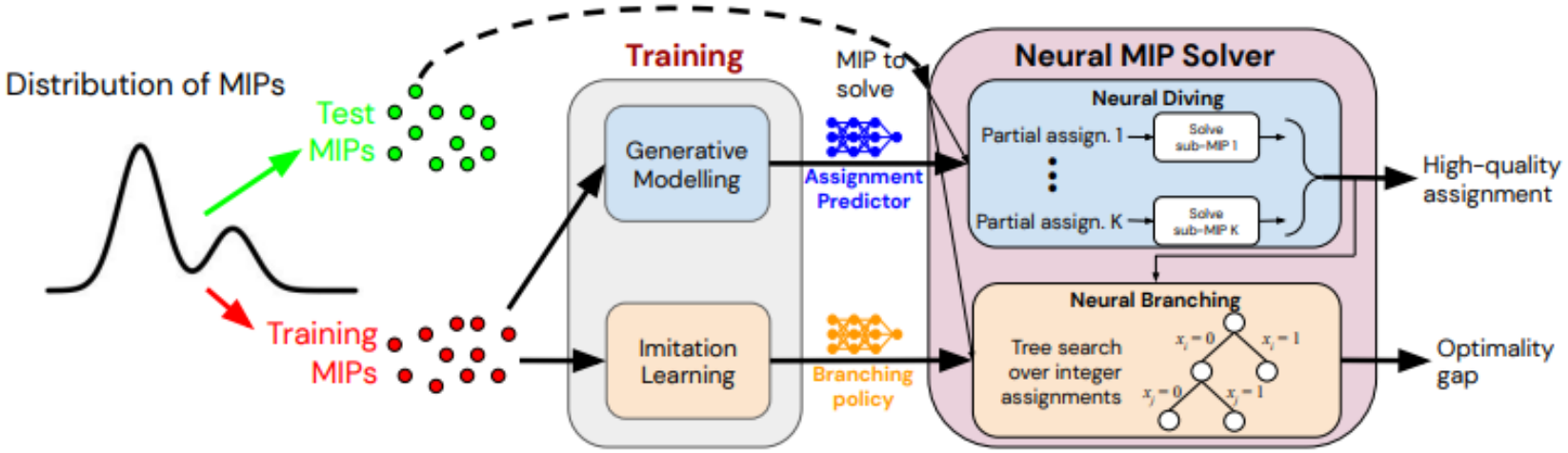
经典求解器的求解MILP类似于分支定界的思想，包含两个过程：①首先是使用原始启发式生成一个高质量的可行解作为初始原始界；②然后采用分支定界的方法不断缩小原始界和对偶界的间隙。

- Nerual Diving
- Nerual Branching

Nair V, Bartunov S, Gimeno F, et al. Solving mixed integer programs using neural networks[J]. arXiv preprint arXiv:2012.13349, 2020.

(1) 如何与精确算法结合？

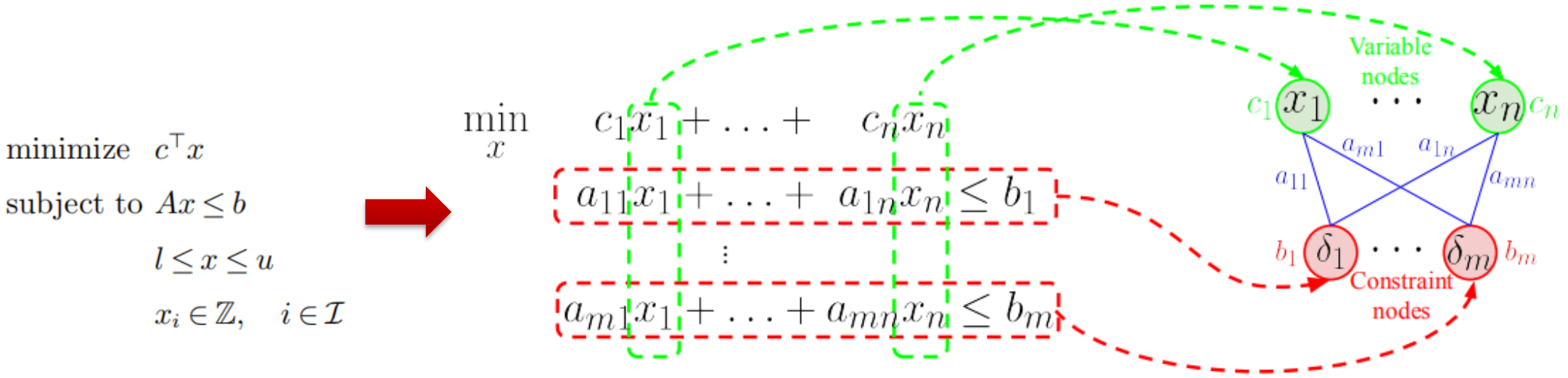
- **Neural Diving**: 训练一个深度神经网络来产生MIP问题部分整数变量的赋值，剩余未赋值的变量定义了一个更小的“子MIP”，这个子问题使用现成的MIP求解器(如SCIP)来求解，进而产生一个高质量的联合变量赋值。
- **Neural Branching**: 训练一个深度神经网络，在经典求解器进行分支定界的过程中，针对给定节点模仿专家的分支变量选择策略进行分支变量的选择。



Nair V, Bartunov S, Gimeno F, et al. Solving mixed integer programs using neural networks[J]. arXiv preprint arXiv:2012.13349, 2020.

（1）如何与精确算法结合？

Neural Diving和Neural Branching的架构都是图神经网络（图卷积神经网络）

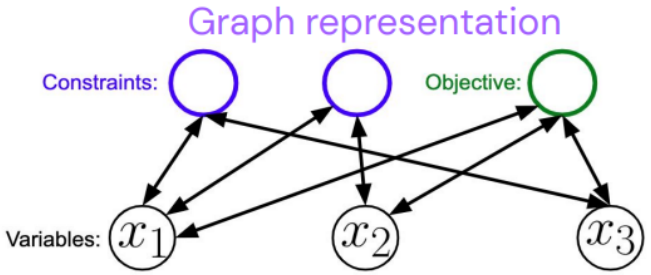


（1）如何与精确算法结合？

- 用二部图表示MILP

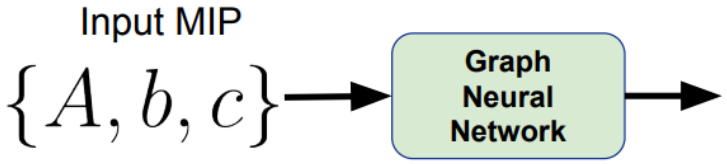
Mixed Integer Program

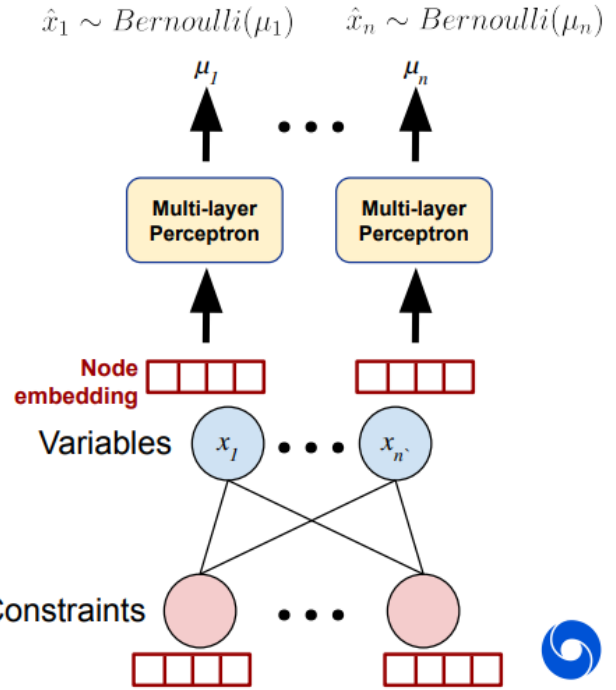
$$\begin{aligned} \min_x \; & d_1x_1 + d_2x_2 + d_3x_3 \\ s.t. \; & A_{11}x_1 + A_{13}x_{13} \le b_1 \\ & A_{21}x_1 + A_{22}x_2 \le b_2 \\ & x \in \mathbb{Z} \end{aligned}$$



- 用神经网络生成部分解

将二部图输入图卷积，经过MLP输出各个变量为某值的概率

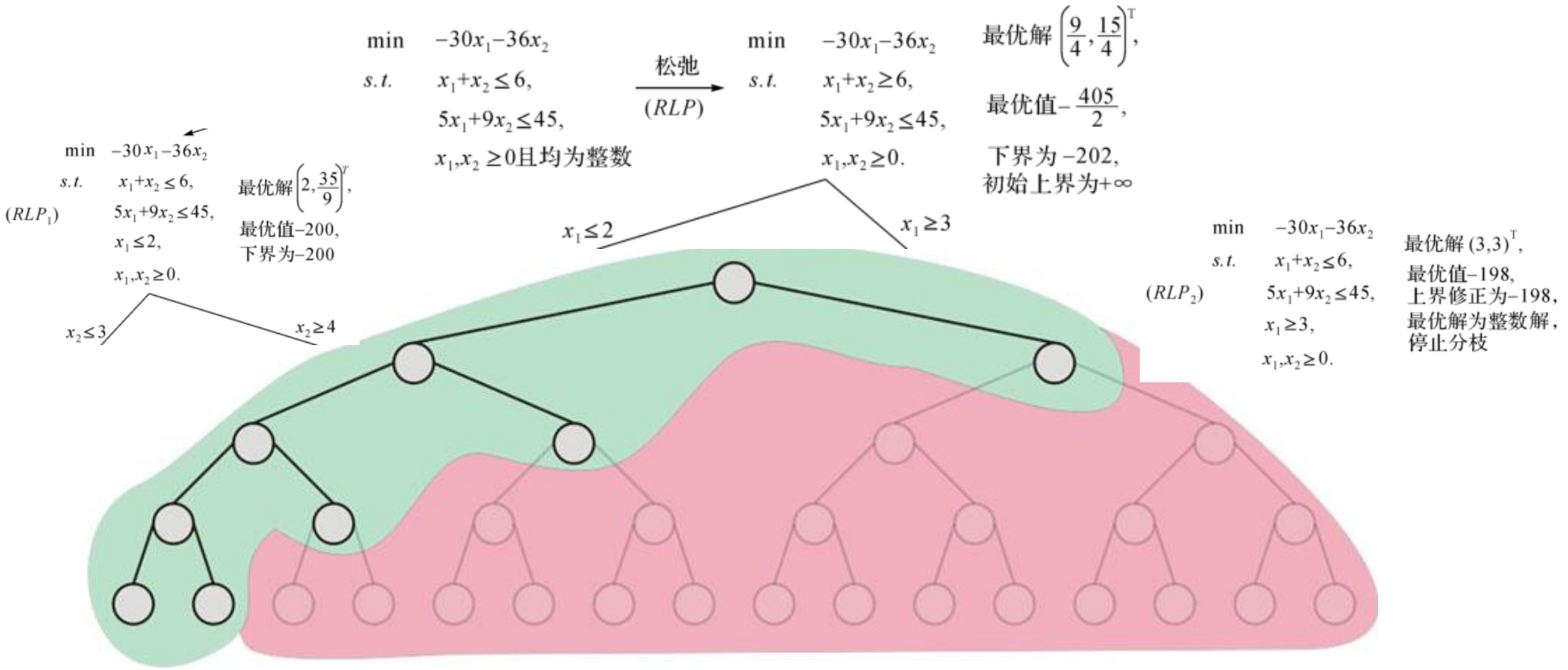




（1）如何与精确算法结合？

■ Neural Branching

Branch and Bound Algorithm, B&B

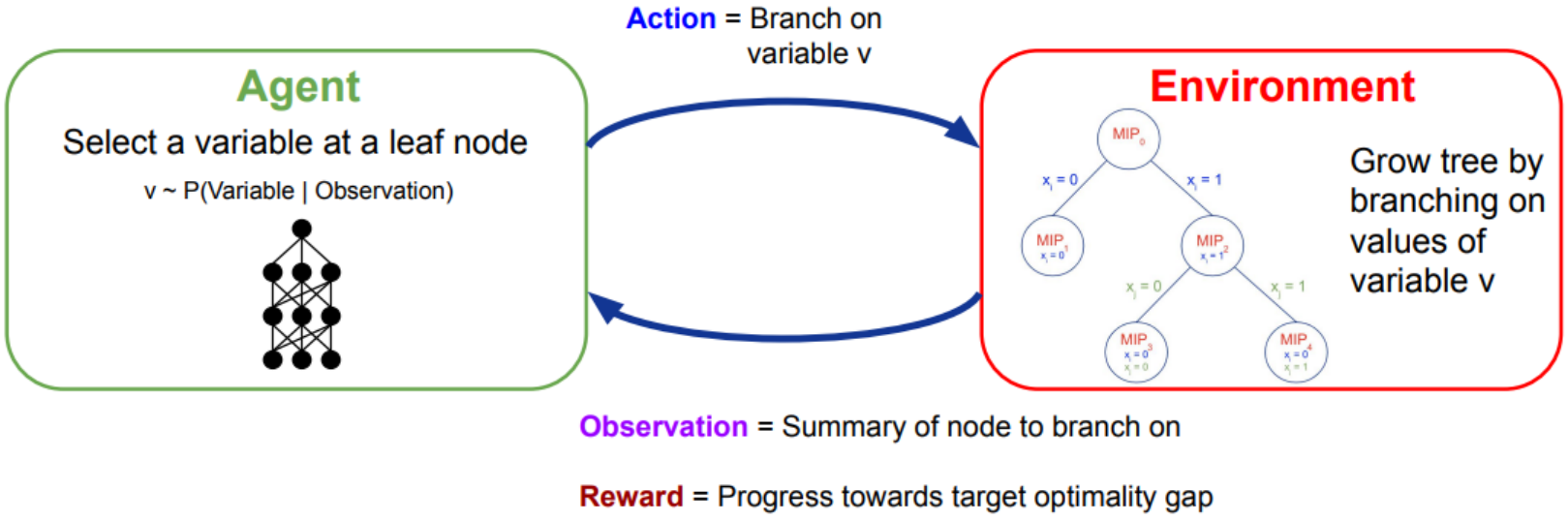


gap = global primal bound − global dual bound.

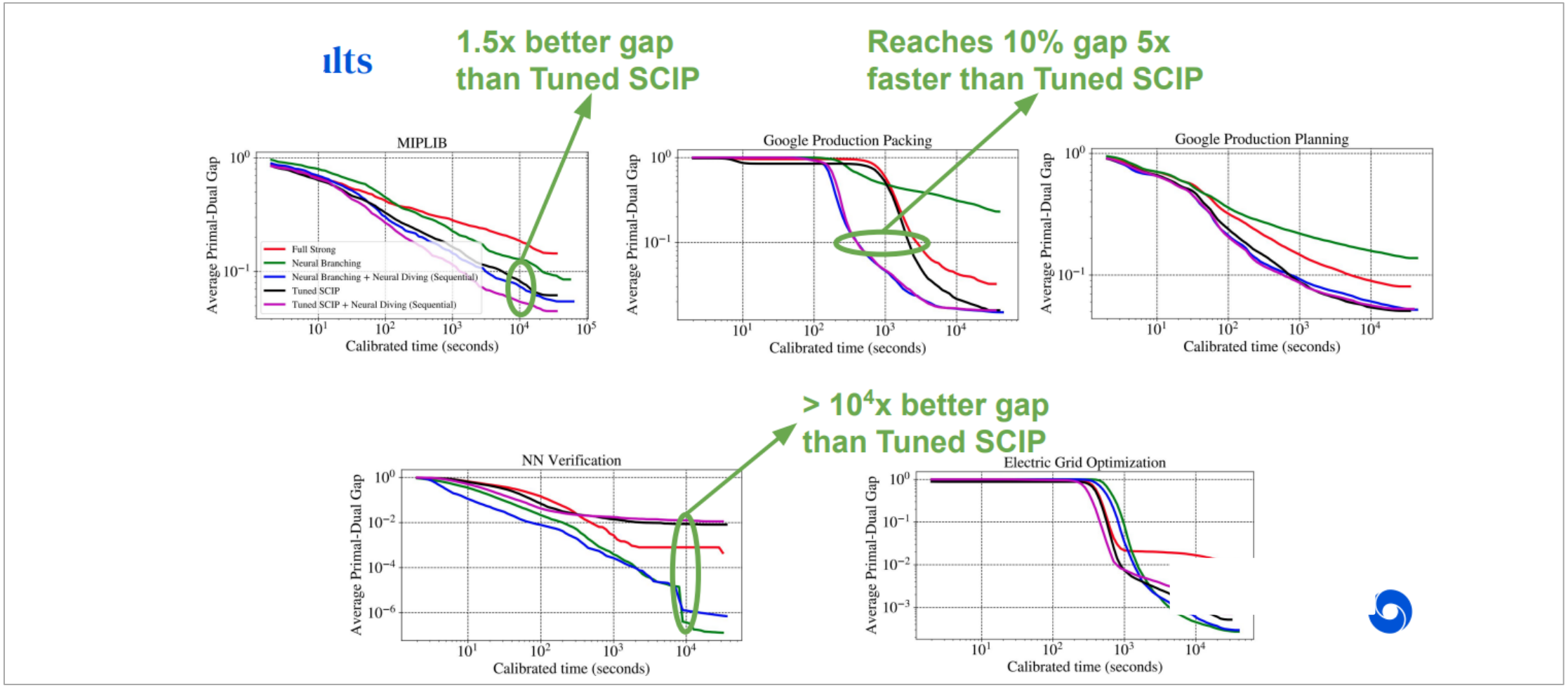
（1）如何与精确算法结合？

■ Neural Branching

在分支定界过程中，每次迭代需要决定展开哪个叶节点以及在哪个变量上分支。变量选择的决策质量直接影响分支定界的效果。为了提高决策质量，**使用模仿学习，来学习strong branching的专家经验**。



(1) 如何与精确算法结合？



模型在速度和精度均取得出色的表现！

（1）如何与精确算法结合？

B&B via Learning：Recent Progress

Node selection

- [He et al., 2014]
- [Song, Lanka, Zhao, et al., 2018]

Variable selection

- [Khalil, Le Bodic, et al., 2016]
- [Hansknecht et al., 2018]
- [Balcan et al., 2018]
- [Gasse et al., 2019]
- [Gupta et al., 2020]
- [Nair et al., 2020]

Cutting planes selection

- [Baltean-Lugojan et al., 2018]
- [Tang et al., 2019]

Primal heuristic selection

- [Khalil, Dilkina, et al., 2017]
- [Hendel et al., 2018]

Formulation selection

- [Bonami et al., 2018]

Neighborhood search heuristics

- [Ding et al., 2019]
- [Song, Lanka, Yue, et al., 2020]
- [Addanki et al., 2020]

Diving heuristics

- [Song, Lanka, Zhao, et al., 2018]
- [Yilmaz et al., 2020]
- [Nair et al., 2020]

（2）如何与启发式算法结合？

nature machine intelligence

ARTICLES

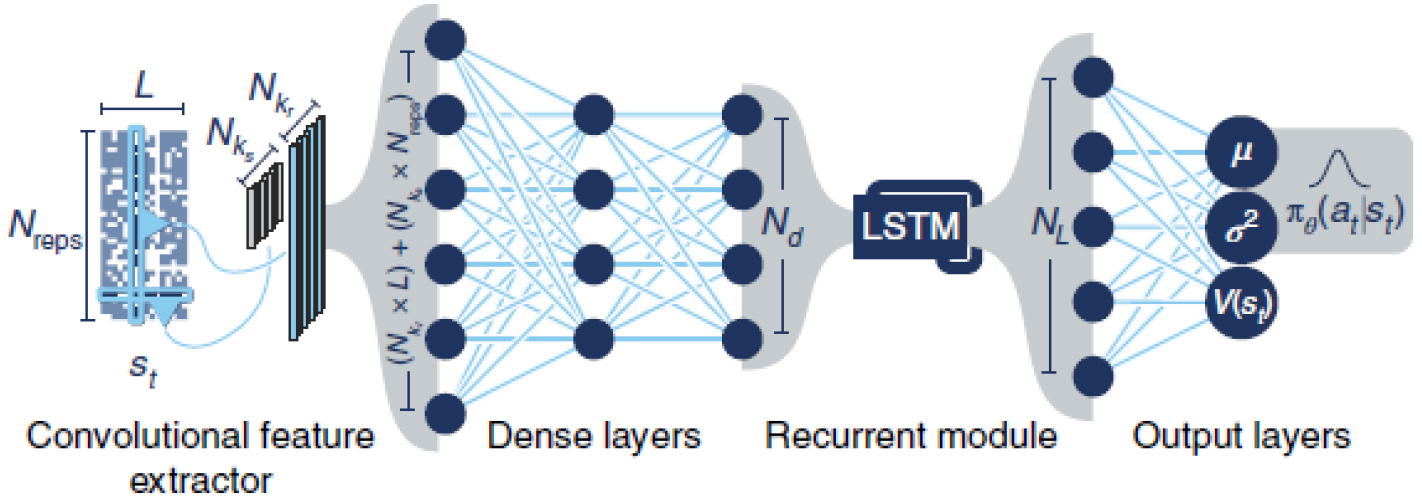
https://doi.org/10.1038/s42256-020-0226-x

Check for updates

Finding the ground state of spin Hamiltonians with reinforcement learning

Kyle Mills[1,2,3] ✉, Pooya Ronagh[1,4,5] ✉ and Isaac Tamblyn[2,3,6] ✉

利用强化学习学习模拟退火算法的最佳退火策略



Mills, Kyle and Ronagh, Pooya and Tamblyn, Isaac. Finding the ground state of spin Hamiltonians with reinforcement learning. Nature Machine Intelligence, 2020.

(3) 如何提高在大规模实例的泛化表现？

AAAI-21

Generalize a Small Pre-trained Model to Arbitrarily Large TSP Instances

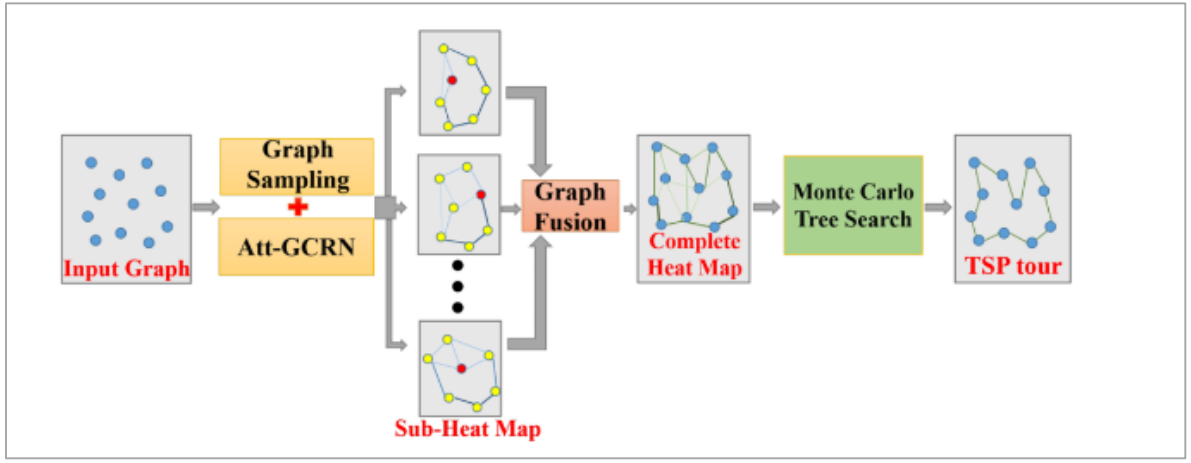
Zhang-Hua Fu [1,2], **Kai-Bin Qiu** [2], **Hongyuan Zha** [1,3 *]

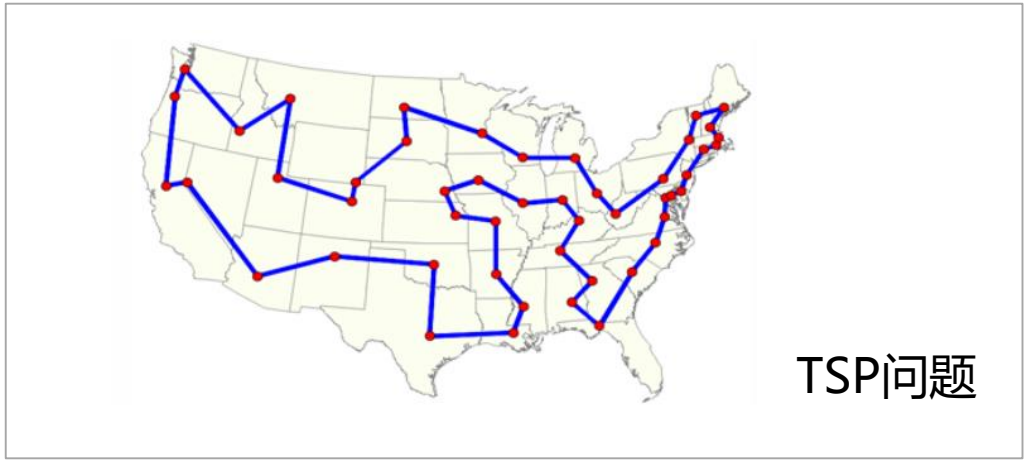
[1] Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China
[2] Institute of Robotics and Intelligent Manufacturing, The Chinese University of Hong Kong, Shenzhen, China
[3] School of Data Science, The Chinese University of Hong Kong, Shenzhen, China
fuzhanghua@cuhk.edu.cn, 220019002@link.cuhk.edu.cn, zhahy@cuhk.edu.cn

（3）如何提高在大规模实例的泛化表现？

分而治之

文章提出的是一种混合算法，级联了SL和RL模块。SL模块运用预训练的模型与三种图技巧，为任意规模的TSP构建热力图。热力图刻画了节点之间每条边属于该TSP最优环游的概率。随后，热力图将作为RL模块的重要输入。RL算法将进一步搜索寻优，输出TSP的最优环游。

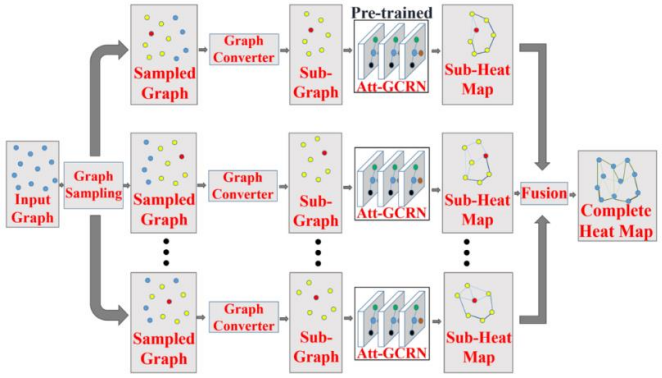




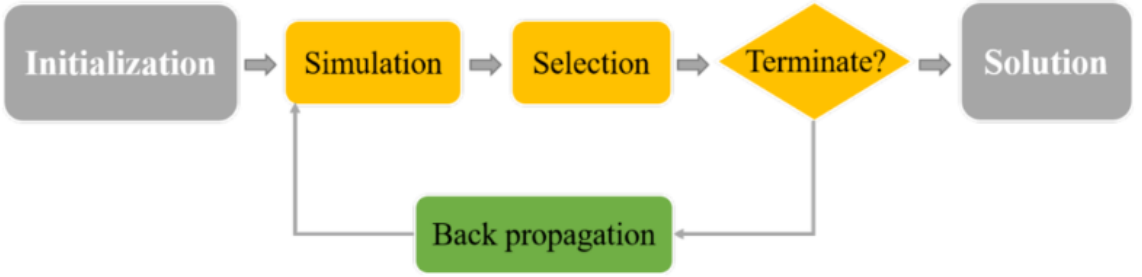
Fu, Z. H., Qiu, K. B., & Zha, H. (2021, May). Generalize a small pre-trained model to arbitrarily large tsp instances. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 8, pp. 7474-7482).

(3) 如何提高在大规模实例的泛化表现？

- SL部分：图采样、图转换、图合并



- RL部分：MCTS



AttGCRN+MCTS表现良好，
在**平均差距（Gap）**和**总运行时间（Time）**上具有竞争力

Method	Type	TSP20			TSP50			TSP100		
		Length	Gap	Time	Length	Gap	Time	Length	Gap	Time
Concorde	Exact Solver	3.8303	0.0000%	2.31m	5.6906	0.0000%	13.68m	7.7609	0.0000%	1.04h
Gurobi	Exact Solver	3.8302	-0.0001%	2.33m	5.6905	0.0000%	26.20m	7.7609	0.0000%	3.57h
LKH3	Heuristic	3.8303	0.0000%	20.96m	5.6906	0.0013%	26.65m	7.7611	0.0026%	49.96m
GAT [9]	RL, S	3.8741	1.1443%	10.30m	6.1085	7.3438%	19.52m	8.8372	13.8679%	47.78m
GAT [9]	RL, S, 2OPT	3.8501	0.5178%	15.62m	5.8941	3.5759%	27.81m	8.2449	6.2365%	4.95h
GAT (Kool et al. 2018)	RL, S	3.8322	0.0501%	16.47m	5.7185	0.4912%	22.85m	7.9735	2.7391%	1.23h
GAT (Kool et al. 2018)	RL, G	3.8413	0.2867%	6.03s	5.7849	1.6568%	34.92s	8.1008	4.3791%	1.83m
GAT (Kool et al. 2018)	RL, BS	3.8304	0.0022%	15.01m	5.7070	0.2892%	25.58m	7.9536	2.4829%	1.68h
GCN (Joshi et al. 2019)	SL, G	3.8552	0.6509%	19.41s	5.8932	3.5608%	2.00m	8.4128	8.3995%	11.08m
GCN (Joshi et al. 2019)	SL, BS	3.8347	0.1158%	21.35m	5.7071	0.2905%	35.13m	7.8763	1.4828%	31.80m
GCN (Joshi et al. 2019)	SL, BS*	3.8305	0.0075%	22.18m	5.6920	0.02509%	37.56m	7.8719	1.4299%	1.20h
Att-GCRN+MCTS(**Ours**)	SL+RL	3.8303	**0.0000%**	23.33s + 1.25m	5.6914	**0.0145%**	2.59m + 5.33m	7.7638	**0.0370%**	3.94m + 10.62m

Fu, Z. H., Qiu, K. B., & Zha, H. (2021, May). Generalize a small pre-trained model to arbitrarily large tsp instances. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 8, pp. 7474-7482).

(3) 如何提高在大规模实例的泛化表现？

配置模型（Configuration Model，CM）

—— 定制匹配目标网络的训练数据

1. 首先计算目标网络的度分布P(k)
2. 从P(k)中采用一个度序列，每个度值的选取概率和它在P(k)中的取值正相关；
3. 根据采样的度序列，使用配置模型生成一个小规模图；
4. 在这些生成的图上训练模型

ANC (×0.01)	Crime	HI-II-14	Digg	Enron	Gnutella31	Epinions	Facebook	Youtube	Flickr
FINDER	10.99	5.54	8.66	**4.30**	11.10	4.99	26.84	**2.37**	5.46
FINDER_CM	**10.81**	**5.43**	**8.52**	4.33	**10.64**	**4.82**	**26.56**	2.40	**5.34**

（4）如何处理复杂的约束？

DC3

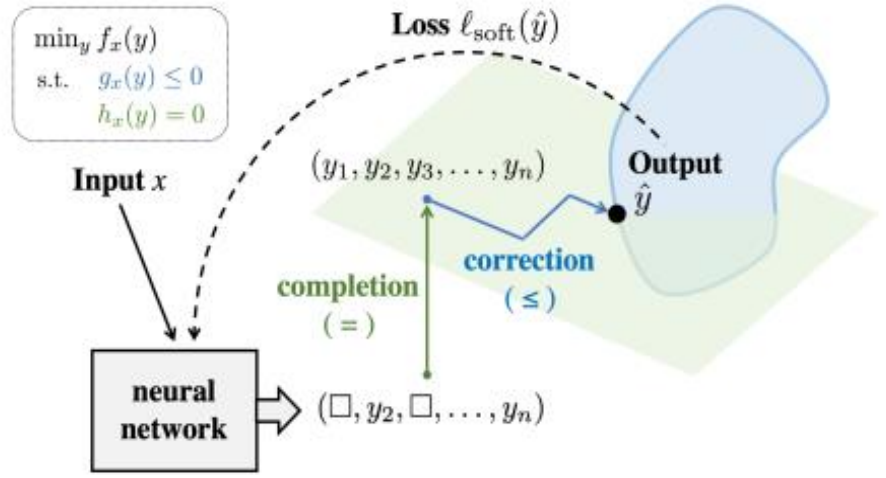
Algorithm 1 Deep Constraint Completion and Correction (DC3)

assume equality completion procedure $\varphi_x : \mathbb{R}^m \to \mathbb{R}^{n-m}$ *// to solve equality constraints*

procedure TRAIN(X)
  init neural network $N_\theta : \mathbb{R}^d \to \mathbb{R}^m$
  while not converged **do** for $x \in X$
    compute partial set of variables $z = N_\theta(x)$
    complete to full set of variables $\tilde{y} = \begin{bmatrix} z^T & \varphi_x(z)^T \end{bmatrix}^T \in \mathbb{R}^n$
    correct to feasible (or approx. feasible) solution $\hat{y} = \rho_x^{(t_{\text{train}})}(\tilde{y})$
    compute constraint-regularized loss $\ell_{\text{soft}}(\hat{y})$
    update θ using $\nabla_\theta \ell_{\text{soft}}(\hat{y})$
  end while
end procedure

procedure TEST(x, N_θ)
  compute partial set of variables $z = N_\theta(x)$
  complete to full set of variables $\tilde{y} = \begin{bmatrix} z^T & \varphi_x(z)^T \end{bmatrix}^T$
  correct to feasible solution $\hat{y} = \rho^{(t_{\text{test}})}(\tilde{y})$
  return $\hat{y}$
end procedure

$$\begin{aligned} \min_{y} \quad & f_x(y) \\ \text{s.t.} \quad & g_x(y) \le 0 \\ & h_x(y) = 0 \end{aligned} \qquad x \in \mathbb{R}^d, y \in \mathbb{R}^n$$



$$\ell_{\text{soft}}(\hat{y}) = f_x(\hat{y}) + \lambda_g \| \text{ReLU}(g_x(\hat{y})) \|_2^2 + \lambda_h \| h_x(\hat{y}) \|_2^2$$

Donti P L, Rolnick D, Kolter J Z. DC3: A learning method for optimization with hard constraints. ICLR2021.

(4) 如何处理复杂的约束？

- Simple QP:

$$\underset{y\in\mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}y^TQy + p^Ty, \quad \text{s.t. } Ay = x,\ Gy \leq h$$

	Obj. value	Max eq.	Mean eq.	Max ineq.	Mean ineq.	Time (s)
Optimizer (OSQP)	-15.05 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.002 (0.000)
Optimizer (qpth)	-15.05 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	1.335 (0.012)
DC3	-13.46 (0.01)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.017 (0.001)
DC3, $\neq$	-12.58 (0.04)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.008 (0.000)
DC3, $\not\leq$ train	-1.39 (0.97)	0.00 (0.00)	0.00 (0.00)	0.02 (0.02)	0.00 (0.00)	0.017 (0.000)
DC3, $\not\leq$ train/test	-1.23 (1.21)	0.00 (0.00)	0.00 (0.00)	0.09 (0.13)	0.01 (0.01)	0.001 (0.000)
DC3, no soft loss	-21.84 (0.00)	0.00 (0.00)	0.00 (0.00)	23.83 (0.11)	4.04 (0.01)	0.017 (0.000)
NN	-12.57 (0.01)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.001 (0.000)
NN, $\leq$ test	-12.57 (0.01)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.008 (0.000)
Eq. NN	-9.16 (0.75)	0.00 (0.00)	0.00 (0.00)	8.83 (0.72)	0.91 (0.09)	0.001 (0.000)
Eq. NN, $\leq$ test	-14.68 (0.05)	0.00 (0.00)	0.00 (0.00)	0.89 (0.05)	0.07 (0.01)	0.018 (0.001)

(4) 如何处理复杂的约束？

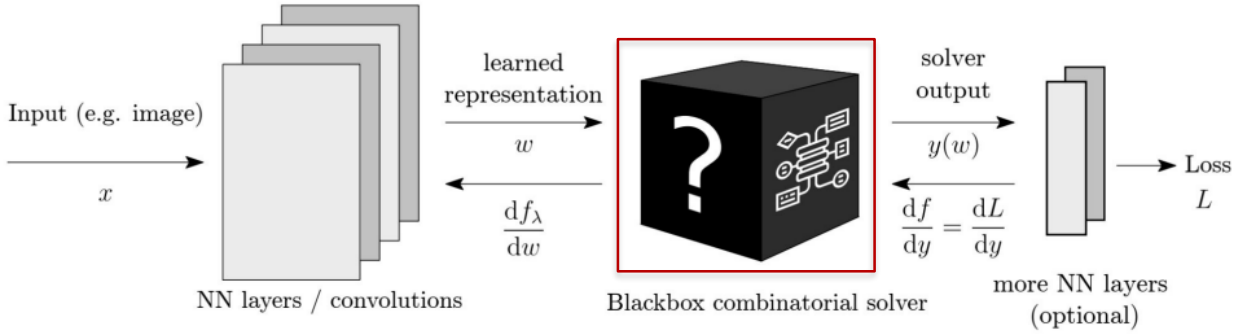
- Simple non-convex:

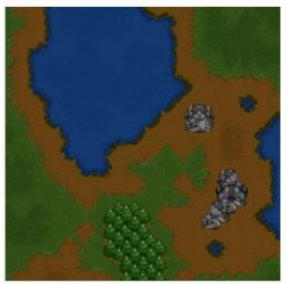
$$\underset{y \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2} y^T Q y + p^T \sin(y), \quad \text{s.t. } Ay = x, \; Gy \leq h,$$

	Obj. value	Max eq.	Mean eq.	Max ineq.	Mean ineq.	Time (s)
Optimizer	-11.59 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.121 (0.000)
DC3	-10.66 (0.03)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.013 (0.000)
DC3, $\neq$	-10.04 (0.02)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.009 (0.000)
DC3, $\not\leq$ train	-0.29 (0.67)	0.00 (0.00)	0.00 (0.00)	0.01 (0.01)	0.00 (0.00)	0.010 (0.004)
DC3, $\not\leq$ train/test	-0.27 (0.67)	0.00 (0.00)	0.00 (0.00)	0.03 (0.03)	0.00 (0.00)	0.001 (0.000)
DC3, no soft loss	-13.81 (0.00)	0.00 (0.00)	0.00 (0.00)	15.21 (0.04)	2.33 (0.01)	0.013 (0.000)
NN	-10.02 (0.01)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.001 (0.000)
NN, $\leq$ test	-10.02 (0.01)	0.35 (0.00)	0.13 (0.00)	0.00 (0.00)	0.00 (0.00)	0.009 (0.000)
Eq. NN	-3.88 (0.56)	0.00 (0.00)	0.00 (0.00)	6.87 (0.43)	0.72 (0.05)	0.001 (0.000)
Eq. NN, $\leq$ test	-10.99 (0.03)	0.00 (0.00)	0.00 (0.00)	0.87 (0.04)	0.06 (0.00)	0.013 (0.000)

(5) 如何与成熟求解器结合？

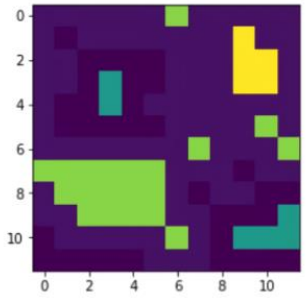
BlackBox: 将CO求解器变成深度学习模型中的可微构建块





魔兽争霸地图

卷积



权重矩阵

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

k × k 指示矩阵
(同Label做Loss)

左上到右下的最短路

Pogančić M V, Paulus A, Musil V, et al. Differentiation of blackbox combinatorial solvers[C]//International Conference on Learning Representations. 2020.

(5) 如何与成熟求解器结合？

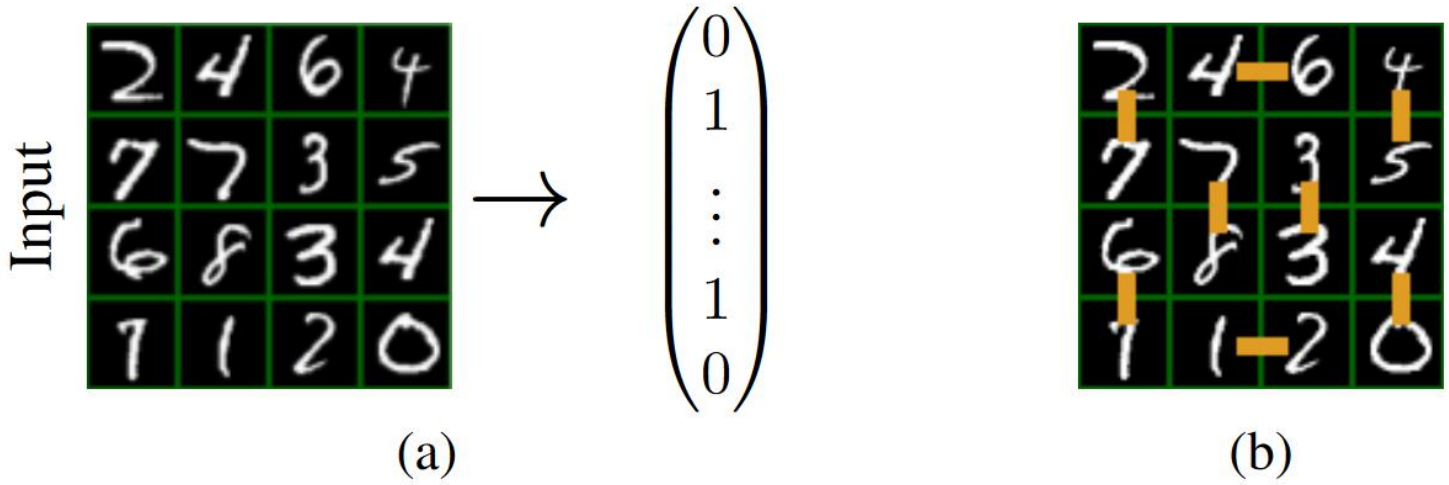


Figure 6: Visualization of the PM dataset. (a) shows the case of PM(4). Each input is a 4 × 4 grid of MNIST digits and the corresponding label is the indicator vector for the edges in the min-cost perfect matching. (b) shows the correct min-cost perfect matching output from the network. The cost of the matching is 348 (46 + 12 horizontally and 27 + 45 + 40 + 67 + 78 + 33 vertically).

Pogančić M V, Paulus A, Musil V, et al. Differentiation of blackbox combinatorial solvers[C]//International Conference on Learning Representations. 2020.

(5) 如何与成熟求解器结合？

Network Planning with Deep Reinforcement Learning

Hang Zhu
Johns Hopkins University

Varun Gupta
Facebook Inc.

Satyajeet Singh Ahuja
Facebook Inc.

Yuandong Tian
Facebook Inc.

Ying Zhang
Facebook Inc.

Xin Jin
Peking University

$$\min \sum_{l \in L} (C_l \times cost_{IP} + \sum_{f \in \Psi_l} cost_f)$$

$$\text{s.t.} \sum_{l:l_{src}=n} Y(l,\omega,\lambda) - \sum_{l:l_{dst}=n} Y(l,\omega,\lambda) = Traffic(\omega, n$$

$$\forall \omega \in \Omega, \lambda \in \Lambda$$

$$C_l \geq \sum_{\omega} Y(l,\omega,\lambda), \forall \lambda \in \Lambda$$

$$\sum_{l \in \Delta_f} C_l \times \phi_{lf} \leq S_f$$

$$C_l \geq C_l^{min}$$

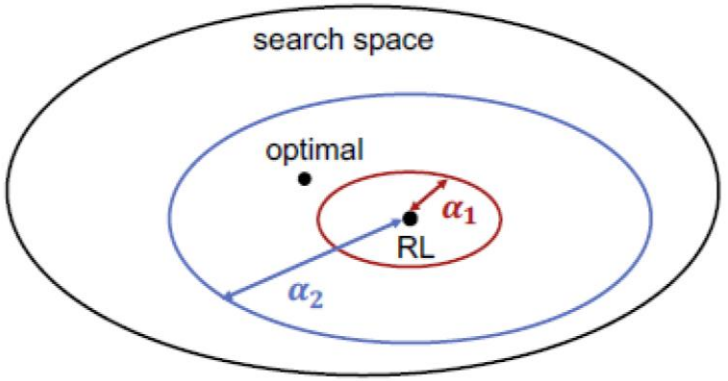


Figure 2: Two-stage hybrid approach in NeuroPlan. The first stage uses RL to find an initial solution, and the second stage uses ILP to find the final solution in a sub search space near the initial solution bounded by the relax factor a.The factor a provides a knob for the trade-off between optimality and tractability.

Zhu, Hang and Gupta, Varun, et al. Network planning with deep reinforcement learning[C]//Proceedings of the 2021 ACM SIGCOMM 2021 Conference. 2021.

（5）如何与成熟求解器结合？

小规模

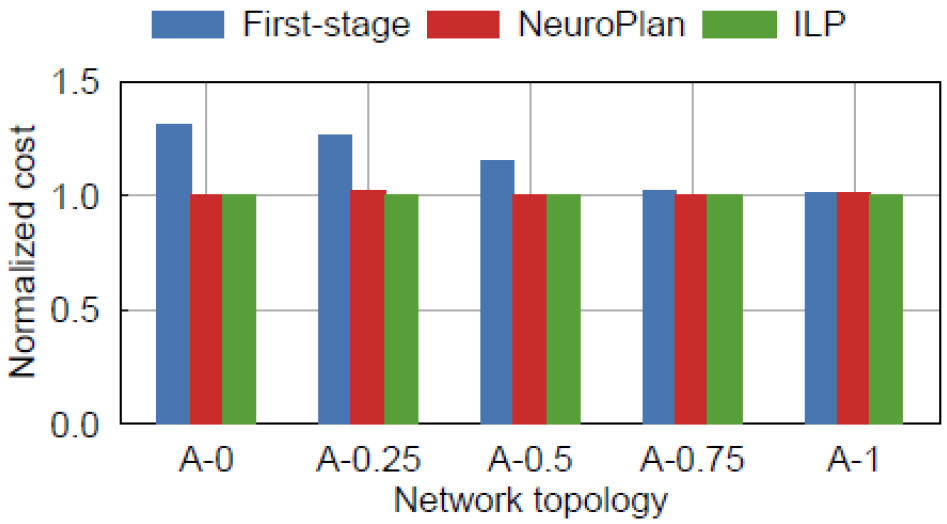


Figure 8: NeuroPlan can produce optimal solutions for small-scale problems. The cost results are normalized with that of ILP on each topology.

大规模

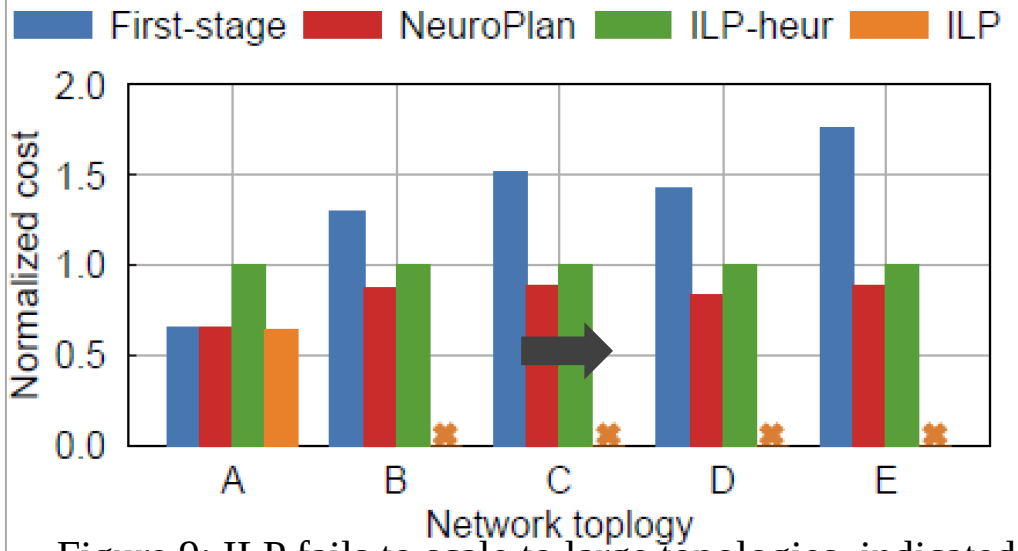


Figure 9: ILP fails to scale to large topologies, indicated by the crosses. NeuroPlan outperforms ILP-heur on large topologies and avoids human efforts to tune the heuristics. The cost results are normalized to that of ILP-heur on each topology.

（6）如何处理更自然的原始输入？

ChatGPT能够理解自然语言的问题需求，
自动建模并调用求解器求解

……

某机床厂生产甲、乙两种机床，每台销售后的利润分别为4千元与3千元。

生产甲机床需用A、B机器加工，加工时间分别为每台2小时和1小时；

生产乙机床需用A、B、C三种机器加工，加工时间为每台各一小时。

若每天可用于加工的机器时数分别为A机器10小时、B机器8小时和C机器7小时，

问该厂应生产甲、乙机床各几台，才能使总利润最大?

……

```
from gurobipy import *

# 创建模型
m = Model("TSP")

# 创建变量
x1 = m.addVar(vtype=GRB.INTEGER, name="x1") # 生产甲机床数量
x2 = m.addVar(vtype=GRB.INTEGER, name="x2") # 生产乙机床数量

# 创建目标函数
m.setObjective(4000*x1 + 3000*x2, GRB.MAXIMIZE)

# 创建约束条件
m.addConstr(2*x1 + x2 <= 10) # A机器的时数约束
m.addConstr(x1 + x2 <= 8) # B机器的时数约束
m.addConstr(x2 <= 7) # C机器的时数约束

# 求解
m.optimize()

# 输出结果
for v in m.getVars():
    print('%s %g' % (v.varName, v.x))

print('Obj: %g' % m.objVal)

x1 4
x2 4
Obj: 28000
```

致谢

感谢**蒲天乐、张柄旭、曾利**等人

为本次报告提供的素材！

敬请批评指正！