Fast and Interpretable Mixed-Integer Linear Program Solving by Learning Model Reduction

作者：李熠轩
日期：2024.9.4

HUAWEI

问题背景

业务流程




抽象

通用建模

$$\min_{\bar{x}} f(\bar{\theta}, \bar{x})$$

$$g(\bar{\theta}, \bar{x}) = \begin{cases} g_1(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_1 \\ g_2(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_2 \\ g_3(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_3 \\ \ldots \\ g_q(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_q \end{cases}$$

$$x_I \in \mathbb{Z}^d$$
$$x \in \mathbb{R}^n$$

求解

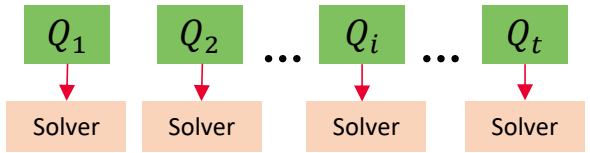
IBM CPLEX

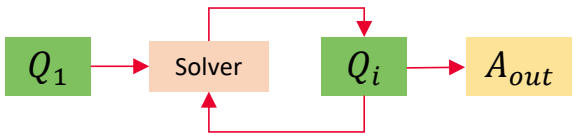
GUROBI OPTIMIZATION

Solver

应用场景

特点：问题参数改变，问题整体结构不变的场景





求解任务

- 针对大规模场景，传统求解方法已无法满足实时性需求
- 业务需求：加速大规模复杂MILP求解
- 应用场景：需要重复求解，具有相同的问题结构特征：

利用数据驱动的AI方法来加速问题求解

研究现状

主流AI加速方法

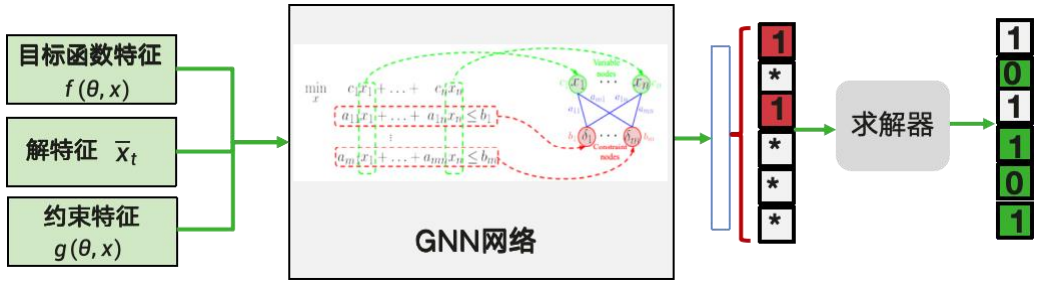
- **问题变量预测**
  - 预测变量
  - 预测部分变量+求解器补全
  - 预测初始变量+搜索优化
- **优化过程加速**
  - 调整算法参数
  - 选取求解规则（分支、切）
  - 启发式调优（邻域
- **问题模型约简**
  - 优化Presolve
  - 问题模型分解
  - 预测简化模型

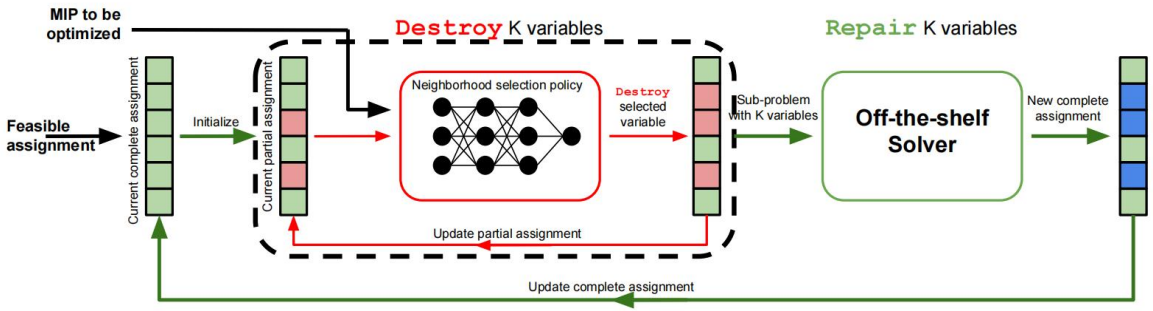
现存缺陷

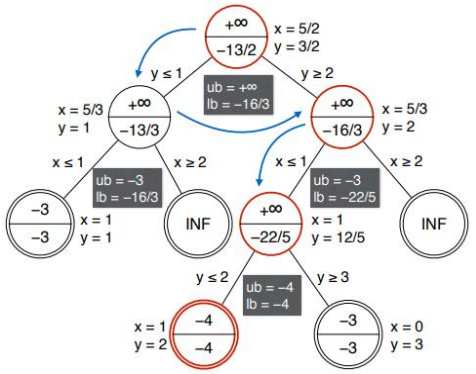
【可解释性差】通过ML方法预测出变量的intuition难理解
【搜索空间大】解空间维度高，从参数到解的映射精度、可扩展性差

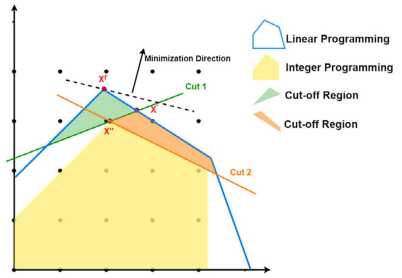
【加速效率受限】相比于端到端，结合求解过程或启发式框架迭代慢
【异构算法交互频繁】需要在AI算法和求解器内部算法间切换

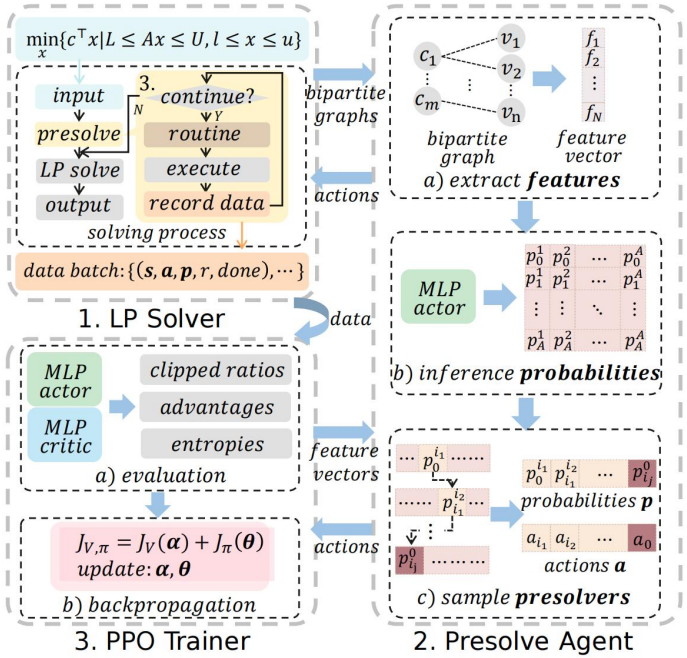
【约简程度受限】预求解后的数学模型依然拥有大量决策变量与约束
【性能与可靠性差】预测约简模型的性能与可靠性取决于ML拟合性能











解决方法

现存缺陷

【可解释性差】
【约简程度受限】
【异构算法交互频繁】

【加速效率受限】

【搜索空间大】

【性能与可靠性差】

解决思路

【1 建模策略】通过建模策略编码了问题在取得最优解时的所有有效信息，**可解释性强**；建模策略对应着问题的显式最小等价模型，**约简程度大**；建模策略包含了模型变量、约束选取等**可解释信息**，可辅助专家决策。

【2 通用加速框架】藉由建模策略，可将求解问题实例转换为寻找问题实例所对应的最优建模策略，为实例应用策略后可以通过KKT条件或调用求解器快速求解，过程仅需预测+求解简化模型，**效率高。**

加速框架可划分为【3 建模策略发现】与【4 建模策略预测】两个阶段:

【3 建模策略发现】以有限个建模策略作为标签可显著**减少学习任务的维度**， 在建模策略空间内预测精度高，且结合剪枝方法可大幅减少建模策略数量，搜索开销可控

【4 建模策略预测】提出通过reward来衡量每个建模策略的性能，**增强决策的可靠性**；构建更合理的ML预测模型，并在训练预测器时通过偏好学习衡量反事实策略的影响，利用更多信息进行决策，提升模型预测性能

1 建模策略（Strategy）

建模策略：一个等价的约简模型，包含了原MILP的有效信息，包括在最优情况下整数解的取值以及激活约束，可以通过约简模型快速恢复出问题的最优解[1]

模型约简

$$\min_{\bar{x}} f(\bar{\theta}, \bar{x})$$

$$g(\bar{\theta}, \bar{x}) = \begin{cases} g_1(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_1 \\ g_2(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_2 \\ g_3(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_3 \\ \ldots \\ g_q(x_I^1, \ldots, x_I^d, x_1, \ldots, x_n) \le b_q \end{cases}$$

$$x_I \in \mathbb{Z}^d$$
$$x \in \mathbb{R}^n$$

原问题模型

建模策略

等价转换

抽象模型　系数矩阵　解空间

$$\min_{\bar{x}} f'(\bar{\theta}_0, \bar{x})$$

$$g'(\bar{\theta}, \bar{x}) = \begin{cases} g_1'(x_{p1}, \ldots, x_{p1}) \le b_1' \\ g_2'(x_{p2}, \ldots, x_{q2}) \le b_2' \\ \ldots \\ g_k'(x_{p1}, \ldots, x_{pn}) \le b_k' \end{cases}$$

$$x \in \mathbb{R}^n$$

简化模型

建模策略

建模策略包含：

1 整数变量的取值$\bar{x}_I^*(\bar{\theta})$

2 激活约束$\mathcal{T}(\bar{\theta})$

整数变量
即在解取最优时的变量的取值

激活约束
$\mathcal{T}(\bar{\theta}) = \{i \in \{1, \ldots, m\} \mid g_i(\bar{\theta}, x^*) = 0\}$
即问题在取最优解时的生效约束
其他约束即为冗余约束

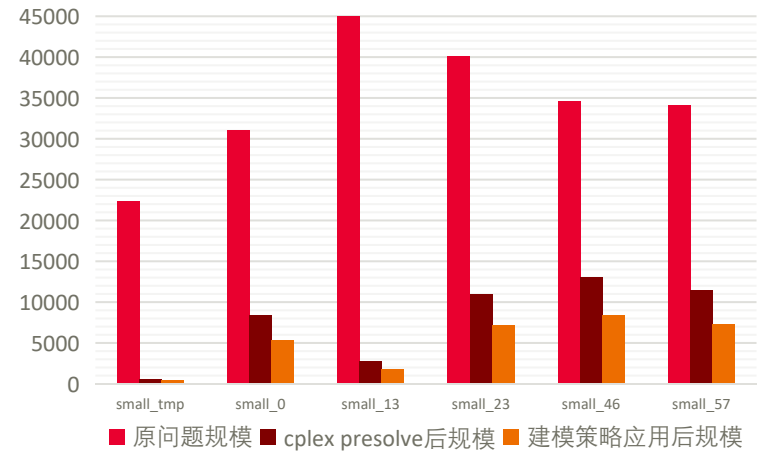
建模策略编码了最优时的有效信息

为实例应用最优建模策略，可将原问题MILP转化为为约束更少、凸的连续LP问题

建模策略可在现有技术预处理后**进一步压缩问题模型**

问题用例	原问题规模	cplex presolve后规模	建模策略后规模	优化比例
small_tmp	22331	608	386	36.51%
small_0	31092	8349	5,375	35.62%
small_13	195654	2725	1,758	35.49%
small_23	40101	10932	7,109	34.97%
small_46	34639	13095	8,395	35.89%
small_57	34117	11460	7,244	36.79%

建模策略与求解器presolve的对比



[1] Boyd, S. P.; and Vandenberghe, L. 2014. Convex Optimization. Cambridge University Press.

1 建模策略（Strategy）案例分析

模型约简

$$\min_{\bar{x}} f(\bar{\theta}, \bar{x})$$

$$g(\bar{\theta}, \bar{x}) = \begin{cases} g_1(x_I^1, \dots, x_I^d, x_1, \dots, x_n) \le b_1 \\ g_2(x_I^1, \dots, x_I^d, x_1, \dots, x_n) \le b_2 \\ g_3(x_I^1, \dots, x_I^d, x_1, \dots, x_n) \le b_3 \\ \dots \\ g_q(x_I^1, \dots, x_I^d, x_1, \dots, x_n) \le b_q \end{cases}$$

$x_I \in \mathbb{Z}^d$
$x \in \mathbb{R}^n$ 原问题模型

建模策略 等价转换

抽象模型　系数矩阵　解空间

$$\min_{\bar{x}} f'(\bar{\theta}_0, \bar{x})$$

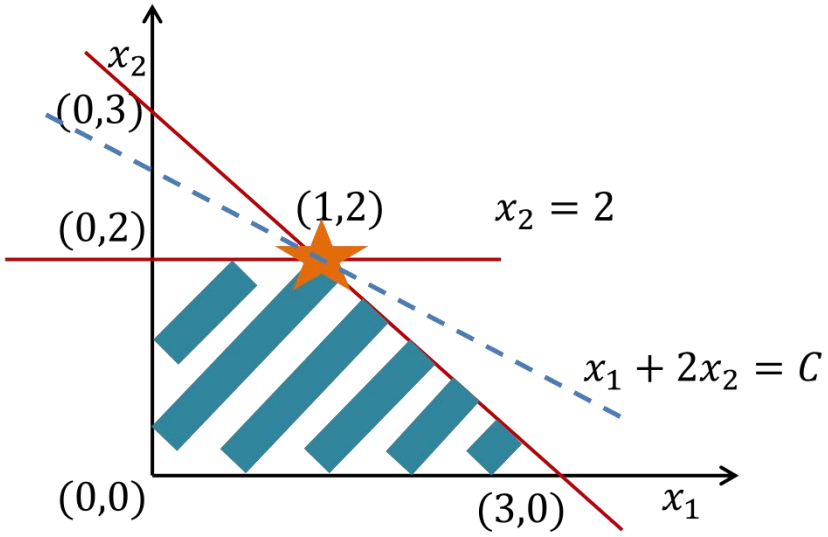
$$g'(\bar{\theta}, \bar{x}) = \begin{cases} g_1'(x_{p1}, \dots, x_{p1}) \le b_1' \\ g_2'(x_{p2}, \dots, x_{q2}) \le b_2' \\ \dots \\ g_k'(x_{p1}, \dots, x_{pn}) \le b_k' \end{cases}$$

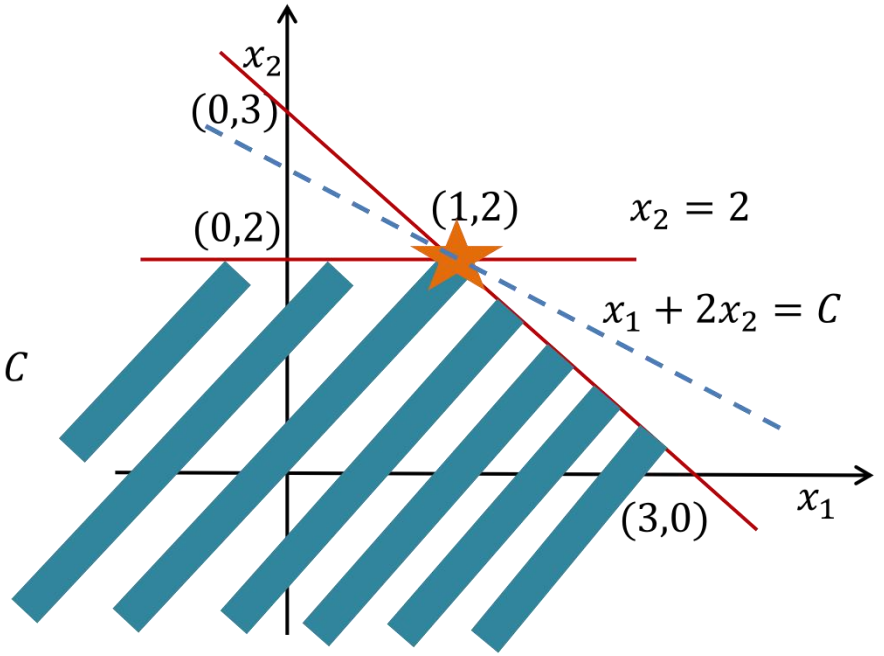
$x \in \mathbb{R}^n$ 简化模型

$$\begin{aligned} &\max_{x_1, x_2} x_1 + 2x_2 && (a) \\ \text{subject to} \quad &x_1 + x_2 \le 3 && (b) \\ &x_2 \le 2 && (c) \\ &x_1 \ge 0 && (e) \\ &x_2 \ge 0 && (f) \end{aligned}$$

等价

$$\begin{aligned} &\max_{x_1, x_2} x_1 + 2x_2 && (a) \\ &x_1 + x_2 \le 3 && (b) \\ &x_2 \le 2 && (c) \end{aligned}$$

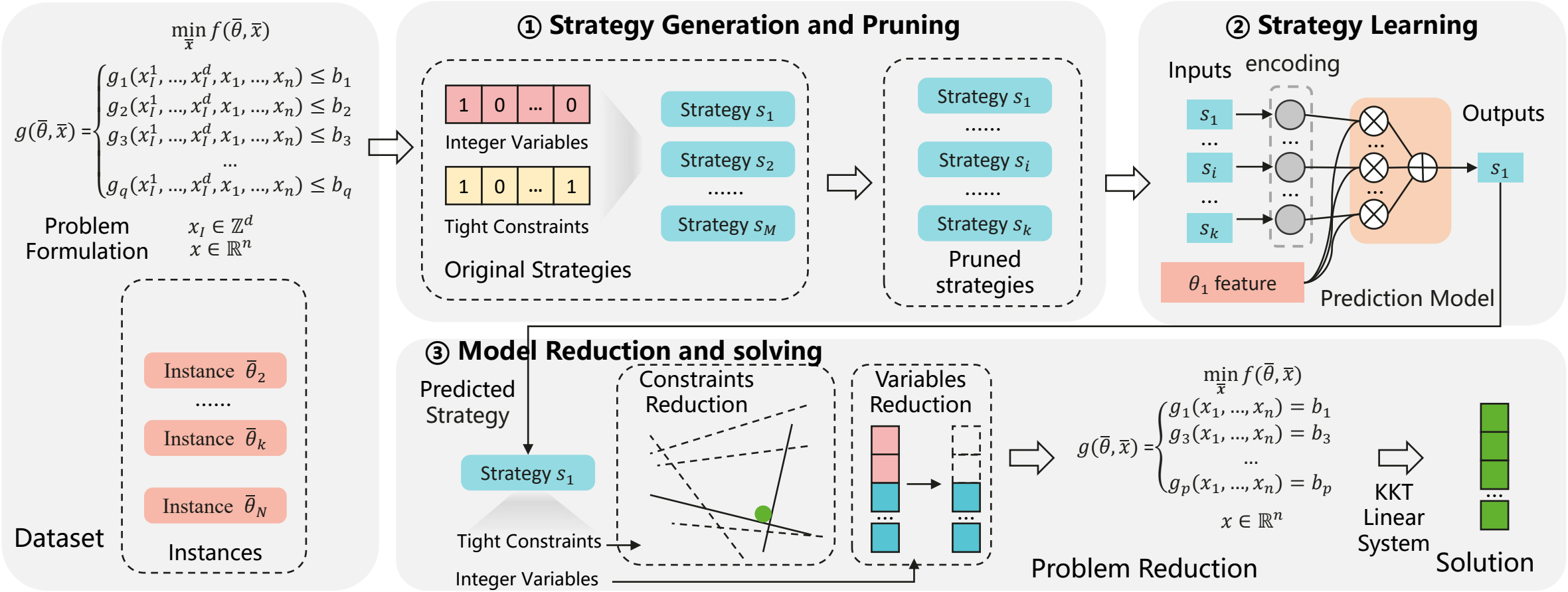




2 通用加速框架

由于通过建模策略可以快速求解问题，故利用建模策略，可将直接求解MILP问题实例转换为，为该MILP问题寻找最适合的建模策略

构建以下框架：首先发现/生成足够的建模策略，然后构建实例-策略映射，为每个实例寻找最优策略



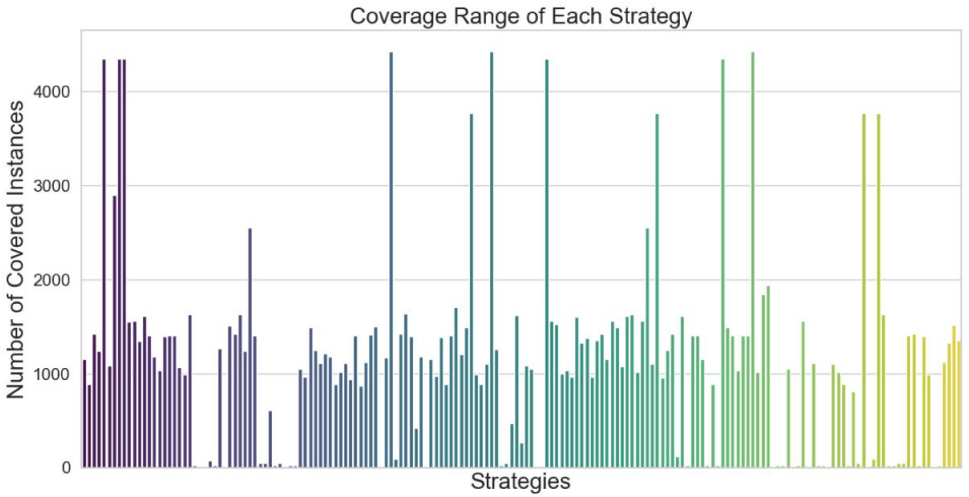
3 建模策略发现与剪枝

策略生成和剪枝：目的是生成一组有用的策略集，可以应用于所有MILP实例；

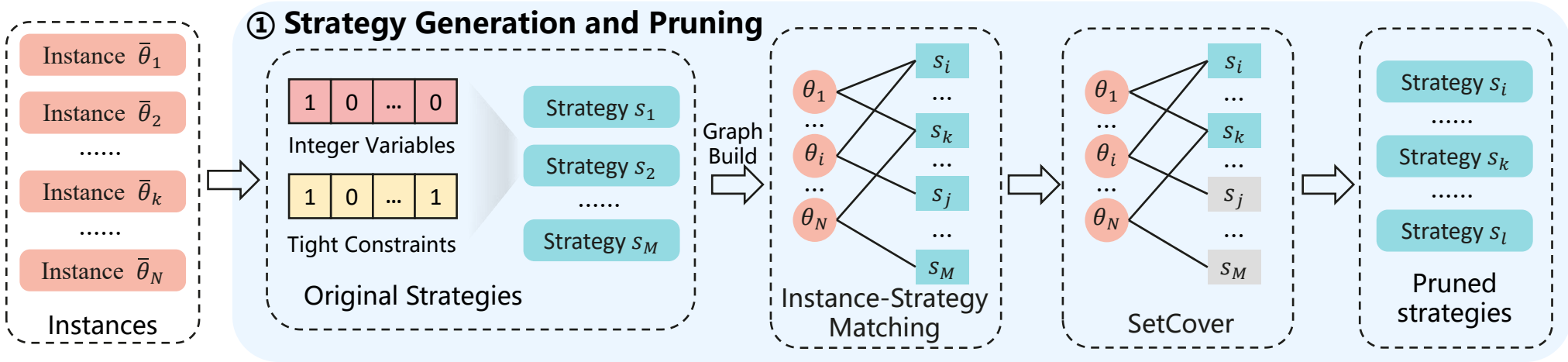
直观来看，可以探索尽可能多的MILP实例，并将它们的最优策略作为候选策略集；

然而，候选策略的数量会随实例的数量快速增长，从而影响策略预测任务的难度；

剪枝：保留最少策略并确保所有实例都能至少有一个可行的策略。



每个策略能平均覆盖接近10%的样本



4 建模策略学习-reward定义

传统预测方法直接拟合从MILP实例θ到策略s的映射，忽略了实例策略空间中可用的重要偏好信息

通过reward来衡量建模策略的性能，即求解应用该建模策略后的实例得到的解的目标函数值和约束可行性

reward

✓ 可行性（infeasiblity）：预测解违反约束的最大值：

$$p(\theta_i, s_j) = \|(g(A, \hat{x}^*_{i,j}) - b)_+\|_\infty / \|b\|$$

✓ 次优性（suboptimality）：预测解相对于最优解的差距：

$$d(\theta_i, s_j) = |f(c, \hat{x}^*_{i,j}) - f(c, x^*_{i,j})| / |f(c, x^*_{i,j})|$$

✓ reward：衡量该实例下该建模策略的优劣：

$$r(\theta_i, s_j) = -log(p(\theta_i, s_j) + d(\theta_i, s_j))$$

直观的方法可以通过直接拟合reward来获得所有策略应用于所有实例下的性能表现，但误差大，拟合困难

实际应用时，仅需要策略间的**相对偏好**，不需要绝对的reward的取值，故可将学习目标转换为策略间的**偏好关系**

4 建模策略学习–偏好模型构建

偏好定义与顺序采样

通过reward定义偏好关系：

$$(\theta_i, s_j) \succ (\theta_i, s_k) \Leftrightarrow r(\theta_i, s_j) > r(\theta_i, s_k)$$

通过样本间的偏好关系来训练模型

偏好训练采样：对于每个实例，若有M个策略，需要采样所有$\binom{M}{2}$个样本对，训练开销大

优化采样：考虑到策略间偏好的传递性，即：若$s_i > s_j, s_j > s_k$，则$s_i > s_k$

可按照reward对策略进行排序

$$s_{\sigma(j)} \in S_\sigma^P = \{s_{\sigma(1)}, \ldots, s_{\sigma(M^P)}\}$$

按相邻顺序采样样本对：

$$\{\langle(\theta_i, s_{\sigma(1)}) \succ (\theta_i, s_{\sigma(2)})\rangle, \cdots, \succ(\theta_i, s_{\sigma(M^P)})\rangle\}$$

则仅需$M-1$个策略对

特征提取与模型构建

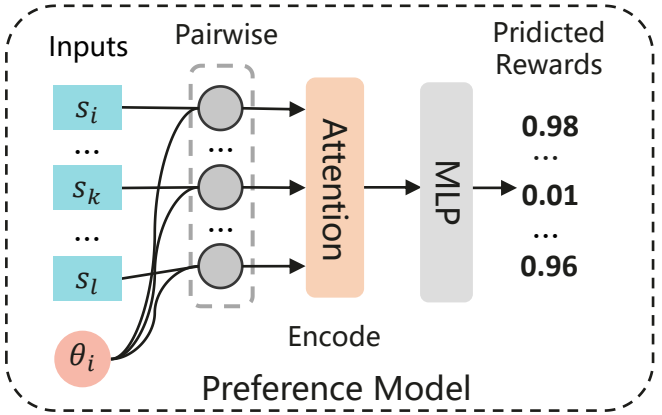
目的：为了提取策略间的内在相似性以及实例和策略之间的潜在联系

应用注意机制来编码实例-策略对

将每个[实例，策略对]视为一个token，在编码每个token时考虑所有的实例策略对

$$A([\theta_i, S^P]) = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V.\ A_L(A_{L-1}(\ldots A_1([\theta_i, S^P]))).$$

$$\hat{R}_i = R_\phi([\theta_i, S^P]) = y_L(A_L) = \psi_L(W_L A_L + b_L)$$



4 建模策略学习-偏好模型训练

$$L_{total}(\phi) = \lambda_1 L_p(\phi) + \lambda_2 L_d(\phi)$$

策略偏好loss

模型输出的reward $\hat{r}$决定了对该策略的偏好程度

$$p_{i,j} = \frac{\exp(\hat{r}_{i,\sigma(j)})}{\exp(\hat{r}_{i,\sigma(j)}) + \exp(\hat{r}_{i,\sigma(j+1)})}.$$

偏好loss最大化输出reward反应正确策略偏好的概率：

$$L_p(\phi) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M^P-1}[\mu_{i,j}\log(p_{i,j}) + (1-\mu_{i,j})\log(1-p_{i,j})]$$

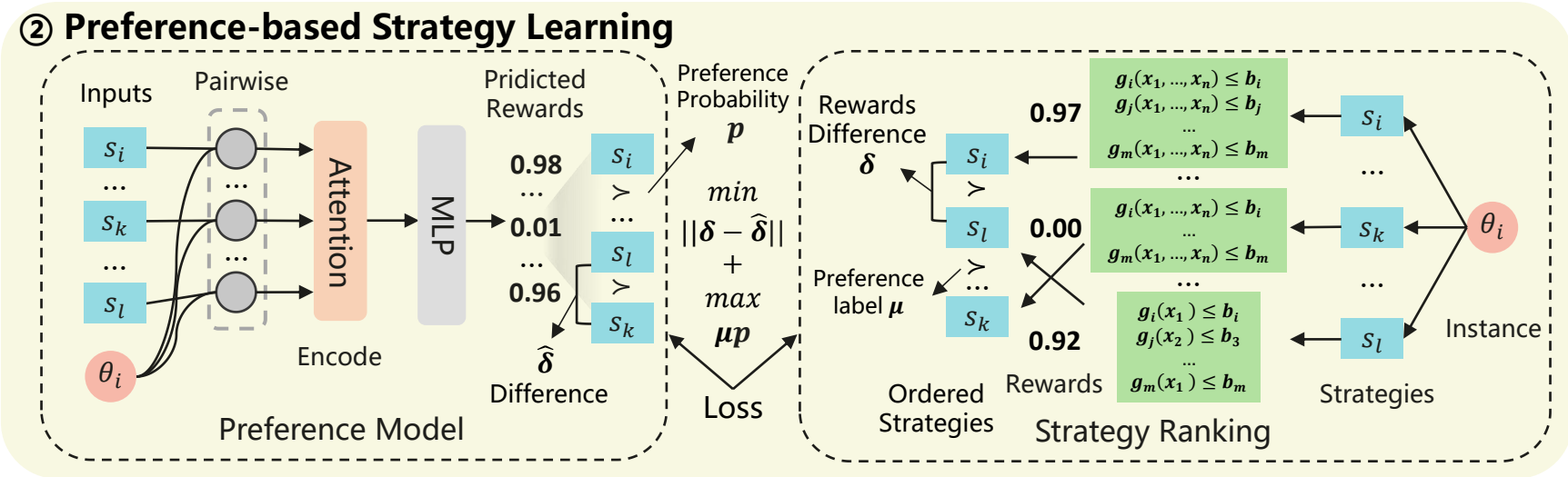
序列差分loss

偏好loss的缺陷：若模型输出了错误的$\hat{r}_{\sigma(2)}$，导致$\hat{r}_{\sigma(1)} < \hat{r}_{\sigma(2)}, \hat{r}_{\sigma(2)} > \hat{r}_{\sigma(3)}$，则虽然$\hat{r}_{\sigma(1)} < \hat{r}_{\sigma(2)}$被偏好loss惩罚，但$\hat{r}_{\sigma(2)} > \hat{r}_{\sigma(3)}$依然会降低偏好loss

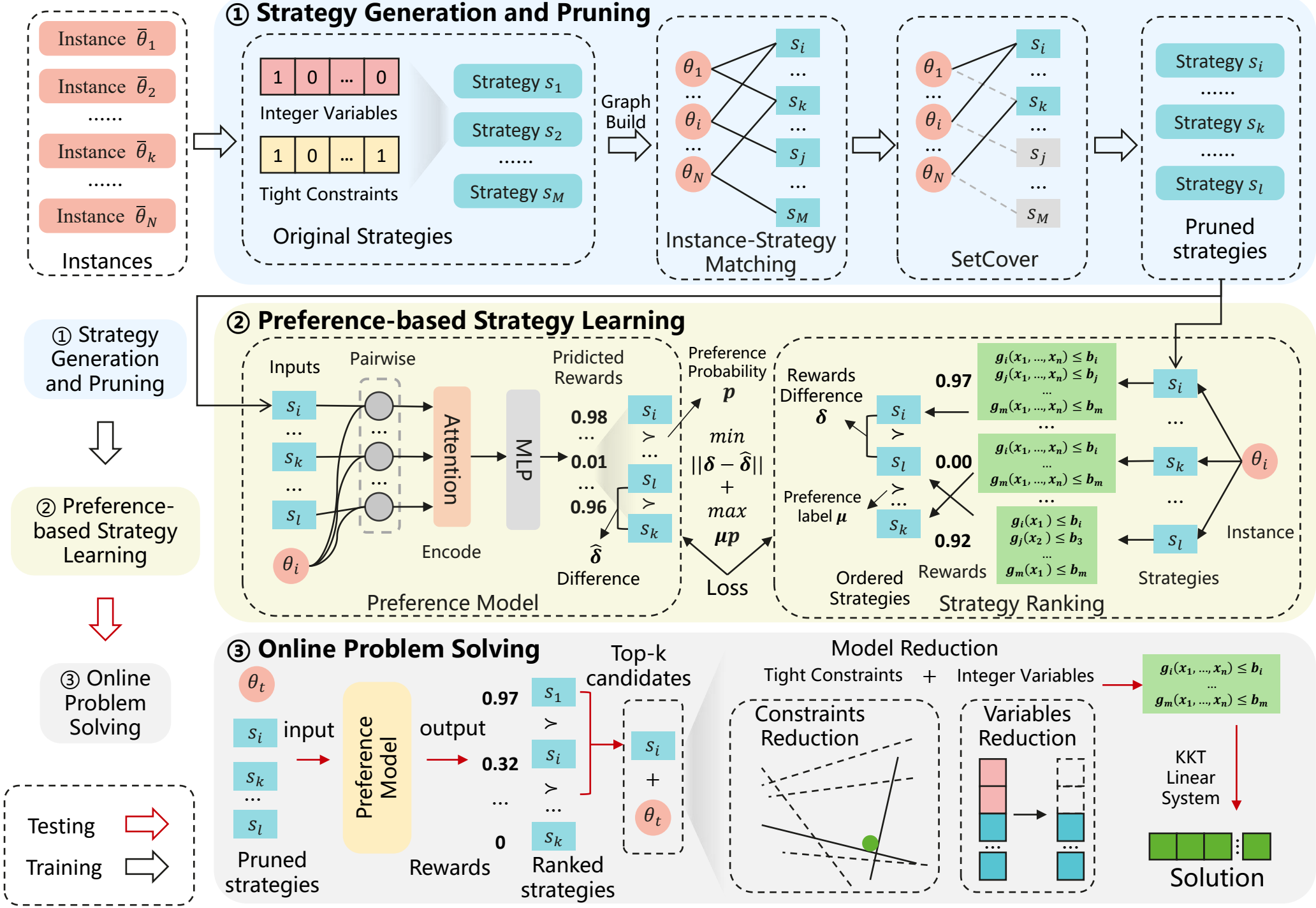
构建差分loss：

$$L_d(\phi) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M^P-1}(\hat{r}_{i,\sigma(j)} - \hat{r}_{i,\sigma(j+1)} - \delta_{i,j})^2$$

增加策略间的相对偏好程度信息，强化序列关系



overview


① Strategy Generation and Pruning
Instance $\bar{\theta}_1$
Instance $\bar{\theta}_2$
.....
Instance $\bar{\theta}_k$
.....
Instance $\bar{\theta}_N$
Instances
1 0 ... 0
Integer Variables
1 0 ... 1
Tight Constraints
Original Strategies
Strategy s_1
Strategy s_2
.....
Strategy s_M
Graph Build
Instance-Strategy Matching
SetCover
Strategy s_i
.....
Strategy s_k
.....
Strategy s_l
Pruned strategies
① Strategy Generation and Pruning
② Preference-based Strategy Learning
③ Online Problem Solving
Testing
Training
② Preference-based Strategy Learning
Inputs
Pairwise
Attention
MLP
Encode
Pridicted Rewards
0.98
0.01
0.96
Difference
Preference Model
Preference Probability
p
min
$||\delta - \hat{\delta}||$
+
max
μp
Loss
Rewards Difference
δ
Preference label μ
0.97
0.00
0.92
Ordered Strategies
Rewards
Strategy Ranking
Instance
Strategies
③ Online Problem Solving
input
Pruned strategies
Preference Model
output
Rewards
0.97
0.32
0
Ranked strategies
Top-k candidates
Model Reduction
Tight Constraints + Integer Variables
Constraints Reduction
Variables Reduction
KKT Linear System
Solution

实验–数据集

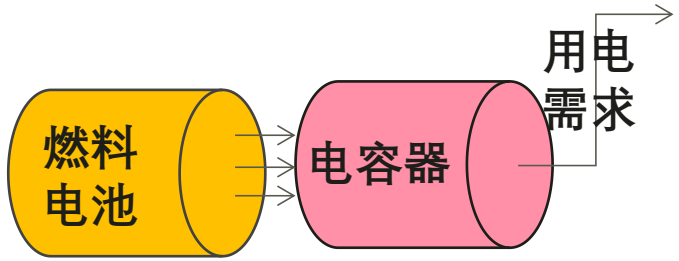
场景：MIPLIB中的六个实例、燃料电池[1]

Problem	N_{con}	$\lvert\mathcal{I}\rvert$	$\lvert\mathcal{C}\rvert$
ns1830653	3,274	1,458	171
mas76	14	150	1
binkar10_1	3,154	170	2,128
markshare_4_0	4	30	4
beasleyC3	3,000	1,250	1,250
neos-827175	25,341	21,350	11,154

T	n_{var}	n_{constr}
10	88	207
20	178	417
30	268	627
40	358	837
50	448	1047
60	538	1257

MIPLIB 约束变量数量 (MILP)

Fuel Cell 约束变量数量 (MIQP)



- 控制电池每个时刻T的供电量P_t，最小化能量损失、保证电池寿命
- 问题参数：$(E_{\text{init}}, z_{\text{init}}, s_{\text{init}}, d_{past} = (d_{-\text{T}},\dots,d_{-1}), P_t^{load})$
- 决策变量，**连续**：P_t；**整数**：$z_t \in \{0,1\}$，$d_t \in \{0,1\}$

$$\min_{P,z} \sum_{t=0}^{T-1} f(P_t, z_t) = \sum_{t=0}^{T-1} \alpha P_t^2 + \beta P_t + \gamma z_t$$

$$\text{s.t.,} \quad E_{t+1} = E_t + \tau\left(P_t - P_t^{load}\right), \quad t = 0,\dots,T$$

$$E^{\min} \le E_t \le E^{\max}, \quad t = 0,\dots,T-1,$$

$$0 \le P_t \le z_t P^{\max}, \quad t = 0,\dots,T$$

$$z_{t+1} = z_t + \omega_t, \quad t = 0,\dots,T$$

$$s_{t+1} = s_t + d_t - d_{t-T}, \quad t = 0,\dots,T-1$$

$$s_t \le n^{\text{sw}}, \quad t = 0,\dots,T$$

$$\text{G}(\omega_t, z_t, d_t) \le h, \quad t = 0,\dots,T$$

$$E_0 = E_{\text{init}}, \; z_0 = z_{\text{init}}, \; s_0 = s_{\text{init}}$$

$$z_t \in \{0,1\}, \quad d_t \in \{0,1\}, \omega_t \in \{-1,0,1\}$$

[1] Frick D, Domahidi A, Morari M, “Embedded Optimization for Mixed Logical Dynamical Systems,” Comput. Chemical Engrg. 72:21–33, 2015.

实验设置

评价指标：可行性（infeasiblity）、次优性（suboptimality）、准确率（accuracy）、耗时（time）

✓ 可行性（infeasiblity）：预测解违反约束的最大值：

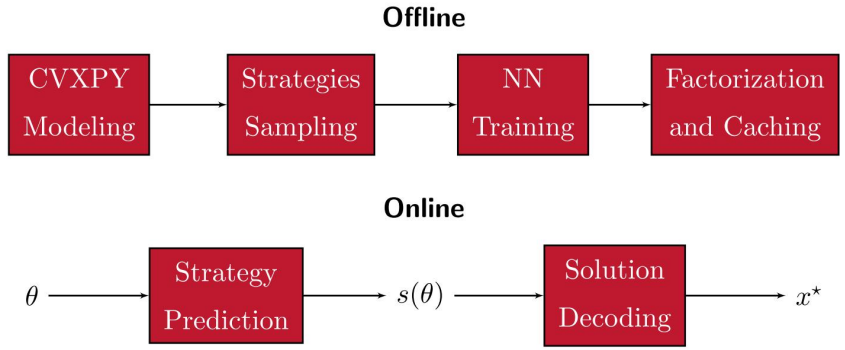
$$p(\theta_i, s_j) = \|(g(A, \hat{x}^*_{i,j}) - b)_+\|_\infty / \|b\|$$

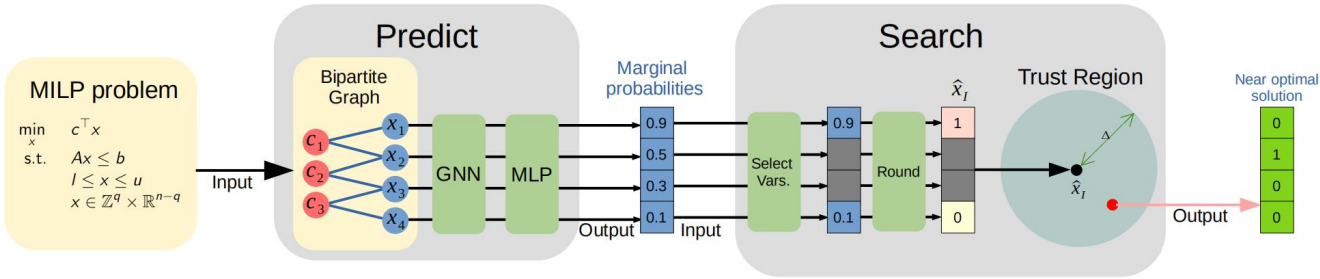
✓ 次优性（suboptimality）：预测解相对于最优解的差距：

$$d(\theta_i, s_j) = |f(c, \hat{x}^*_{i,j}) - f(c, x^*_{i,j})| / |f(c, x^*_{i,j})|$$

✓ 准确率（accuracy）：两种特性都满足（在容忍度内）：

$$accuracy = \frac{1}{N} |\{\theta_i \mid p(\theta_i, \hat{s_j}) \leq \epsilon_1 \wedge d(\theta_i, \hat{s_j}) \leq \epsilon_2\}|$$





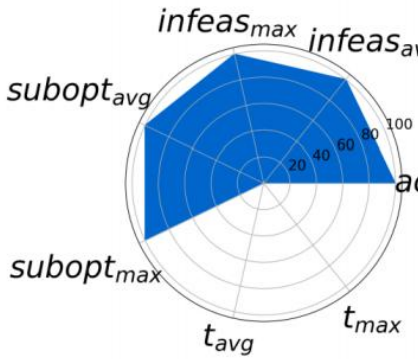
MIPLIB：训练集：1000，测试集：100
Fuel Cell：训练集：10000，测试集：1000

对比方法：GUROBI、GUROBI启发式、MLOPT[1]、Predict and Search[2]

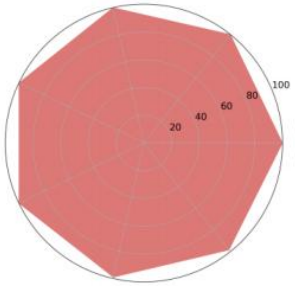
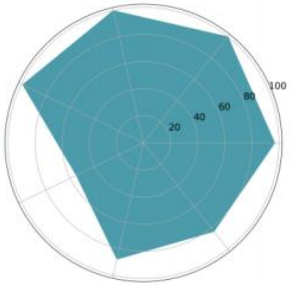
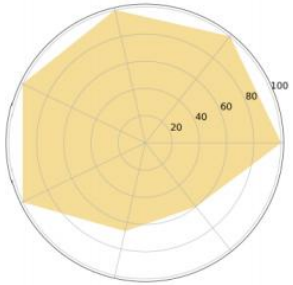
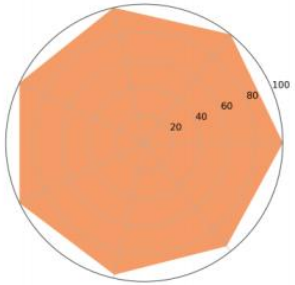
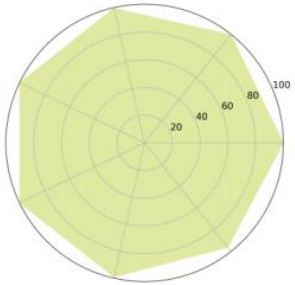
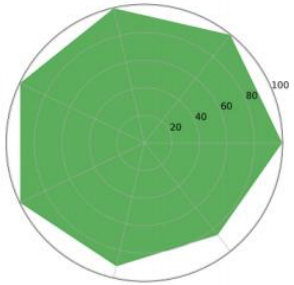
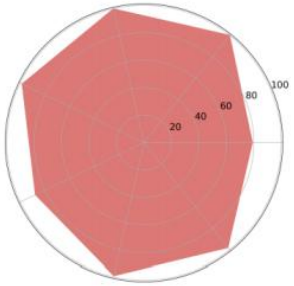
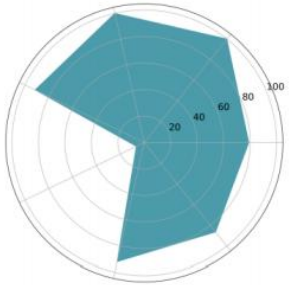
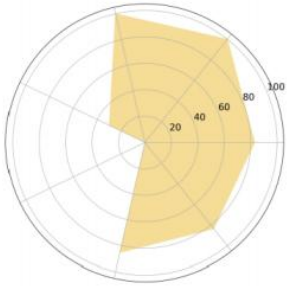
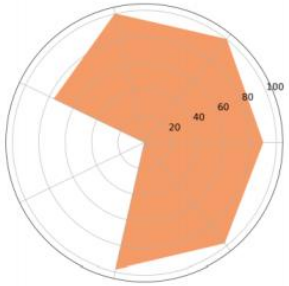
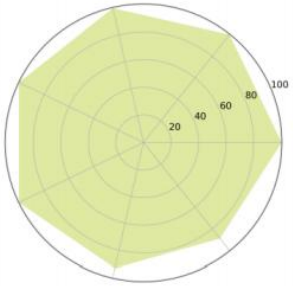
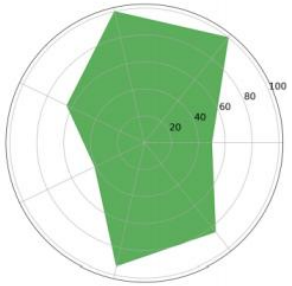
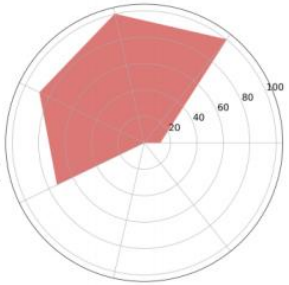
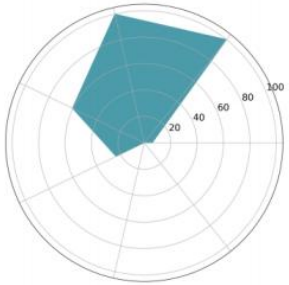
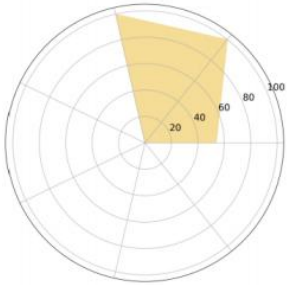
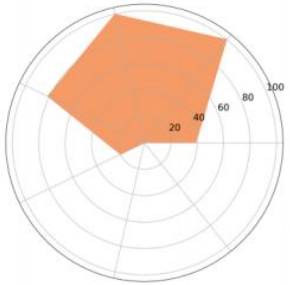
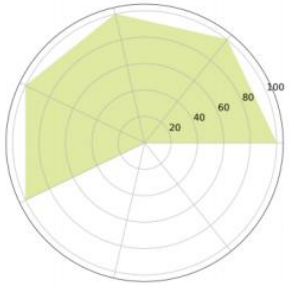
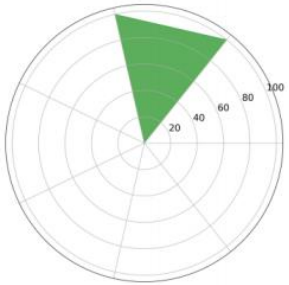
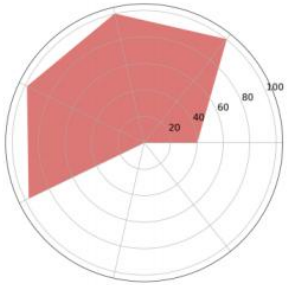
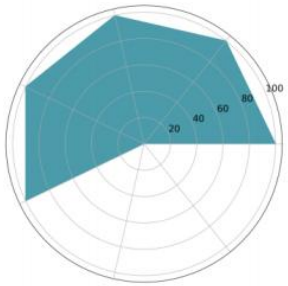
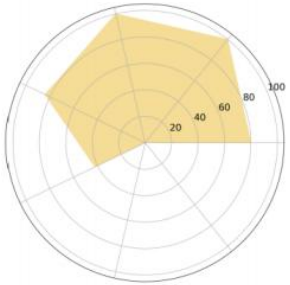
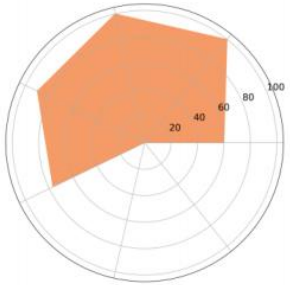
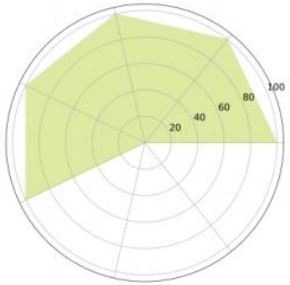
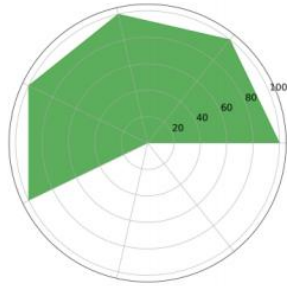
[1] Bertsimas, D.; and Stellato, B. 2022. Online mixed-integer optimization in milliseconds. INFORMS Journal on Com_x0002_puting, 34(4): 2229–2248

[2] Han, Q.; Yang, L.; Chen, Q.; Zhou, X.; Zhang, D.; Wang, A.; Sun, R.; and Luo, X. 2023. A GNN-Guided Predict-and_x0002_Search Framework for Mixed-Integer Linear Programming. In ICLR’ 23.

实验结果-MIPLIB



Gurobi Benchmark

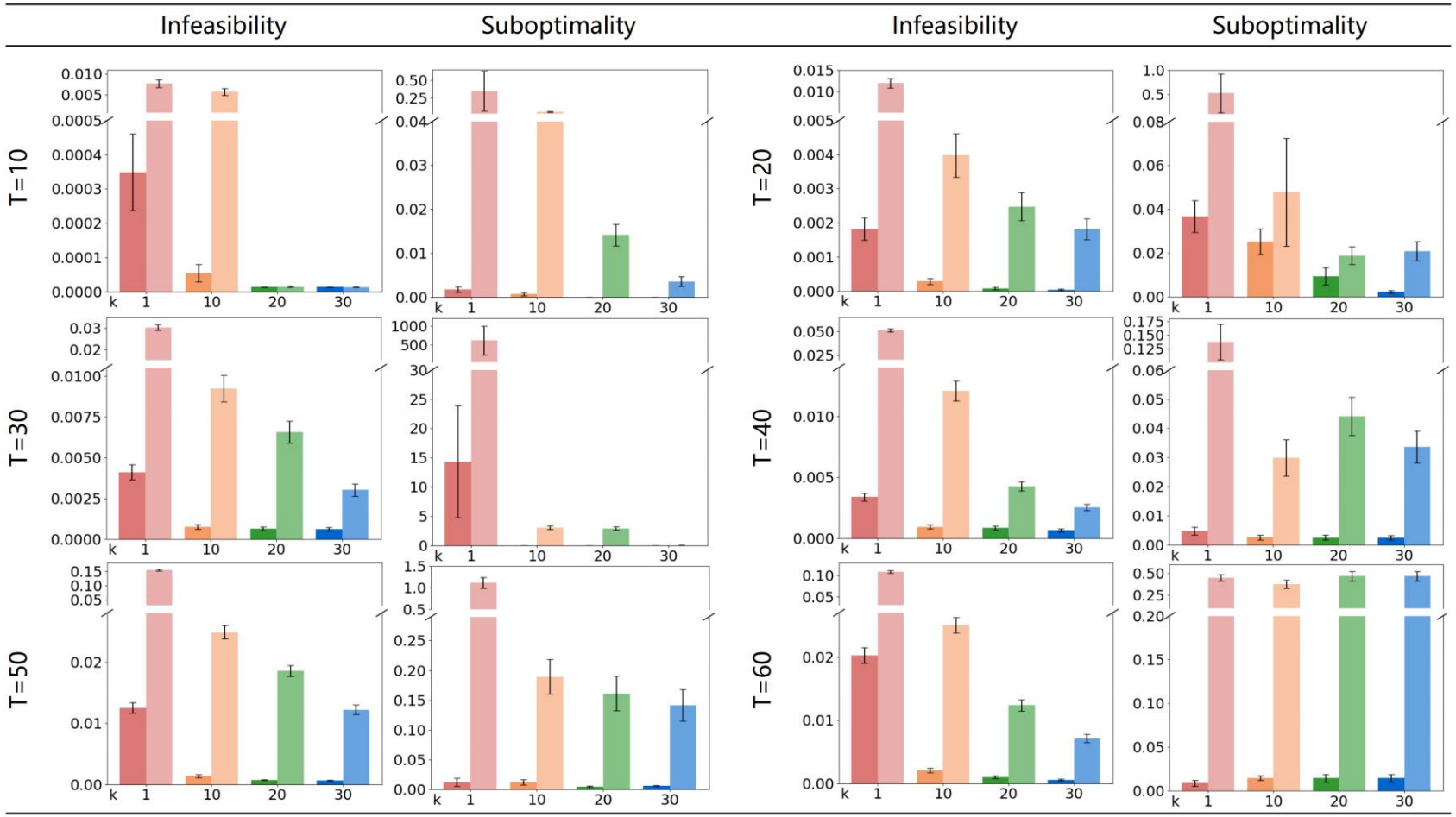
Methods	Binkar	Beasley	Markshare	Mas	Ns	Neos
Ours						
MLOPT						
Gurobi Heuristic						
Predict and Search						

实验结果-Fuel Cell

T	$t^H_{\max}(s)$	$t^G_{\max}(s)$	M	method	M^P	$t_{\max}(s)$	accuracy	gain
20	0.366	0.379	578	Ours	41	0.061	97.66%	**10.98%**
				MLOPT	110	0.047	86.68%	
40	1.120	4.840	2041	Ours	75	0.069	94.20%	**16.35%**
				MLOPT	2026	0.054	77.85%	
60	1.184	33.733	2924	Ours	129	0.086	84.90%	**38.93%**
				MLOPT	2863	0.069	45.97%	

时间策略准确率

可行性与次优性



实验结果-附加

直接拟合

$T=10$					$T=20$				$T=30$			
method	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain
Ours	95.65%	0.00035	0.00176	**4.58%**	83.04%	0.00182	0.03676	**10.60%**	73.88%	0.00409	14.33399	**9.38%**
RF	91.07%	0.00680	0.00836		72.43%	0.00685	0.03883		64.51%	0.00705	0.04398	
$T=40$					$T=50$				$T=60$			
method	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain
Ours	73.21%	0.00338	0.00477	**9.60%**	56.14%	0.01250	0.01210	**14.29%**	39.71%	0.02029	0.00865	**6.82%**
RF	63.62%	0.00740	0.01307		41.85%	0.02385	0.01668		32.90%	0.01387	0.04394	

采样方式

$T=10$					$T=20$				$T=30$			
method	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain
Ours	95.65%	0.00035	0.00176	**5.08%**	83.04%	0.00182	0.03676	**8.48%**	73.88%	0.00409	14.33399	**6.70%**
NR	89.84%	0.00119	0.00069		74.55%	0.00315	0.01899		67.19%	0.00639	0.02675	
$T=40$					$T=50$				$T=60$			
method	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain	accuracy	$infeas_{avg}$	$subopt_{avg}$	gain
Ours	73.21%	0.00338	0.00477	**9.49%**	56.14%	0.01250	0.01210	**5.25%**	39.71%	0.02029	0.00865	**13.80%**
NR	63.73%	0.00623	0.01223		50.89%	0.01184	0.00366		25.91%	0.01411	0.00581	

总结

Conclusion

建模策略：编码问题模型取得最优时的有效信息

求解框架：建模策略发现+建模策略学习+快速求解

建模策略发现：利用Setcover剪枝以控制策略空间

建模策略学习：

定义reward衡量策略适应程度

通过策略偏好克服拟合reward难的问题

定义基于attention的偏好模型

通过顺序采样偏好对与序列差分训练偏好模型

Future Work

可利用的信息：矩阵、背景、语义

建模策略泛化

建模策略 for Solver

大模型 for 建模策略

Thank you !

Questions or Comments?