1 Bipartite Representation and Network Architecture

We give full details of the Bipartite representation and GAT architecture described in Section 4. First, there is a feature extractor convert the MILP instance P to bipartite graph representation $P' = (G, V, C, E)$. The features we extracted is shown in Table 1. Then our policy takes the bipartite graph $P' = (G, V, C, E)$ as input and output a score vector with one score per decision variable. We use 2-layer MLPs with 64 hidden units per layer and ReLU as the activation function to map each node feature and edge feature to $\mathbb{R}^L$ where $L = 64$.

Let $\boldsymbol{v}_j, \boldsymbol{c}_i, \boldsymbol{e}_{i,j} \in \mathbb{R}^L$ be the embeddings of the i-th variable, j-th constraint and the edge connecting them output by the embedding layers V^1, C^1, E^1. We perform two rounds of message passing through the GAT. In the first round, each constraint node $\boldsymbol{c}_i$ attends to its neighbors $\mathcal{N}_i$ using an attention structure with $H = 8$ attention heads:

$$\boldsymbol{c}'_i = \frac{1}{H} \sum_{i=1}^{H} \left(\alpha_{ii,1}^{(h)} \boldsymbol{\theta}_{c,1}^{(h)} \boldsymbol{c}_i + \sum_{j \in \mathcal{N}_i} \alpha_{ij,1}^{(h)} \boldsymbol{\theta}_{v,1}^{(h)} \boldsymbol{v}_j \right)$$

where $\boldsymbol{\theta}_{c,1}^{(h)} \in \mathbb{R}^{L \times L}$ and $\boldsymbol{\theta}_{v,1}^{(h)} \in \mathbb{R}^{L \times L}$ are learnable weights. The updated constraints embeddings $\boldsymbol{c}'_i$ in updated embedding layer C^2 are averaged across H attention heads using attention weights

$$\alpha_{ij,1}^{(h)} = \frac{\exp(\boldsymbol{w}_1^T \rho([\boldsymbol{\theta}_{c,1}^{(h)} \boldsymbol{c}_i, \boldsymbol{\theta}_{v,1}^{(h)} \boldsymbol{v}_j, \boldsymbol{\theta}_{e,1}^{(h)} \boldsymbol{e}_{i,j}]))}{\sum_{k \in \mathcal{N}_i} \exp(\boldsymbol{w}_1^T \rho([\boldsymbol{\theta}_{c,1}^{(h)} \boldsymbol{c}_i, \boldsymbol{\theta}_{v,1}^{(h)} \boldsymbol{v}_k, \boldsymbol{\theta}_{e,1}^{(h)} \boldsymbol{e}_{i,k}]))}$$

where the attention coefficients $\boldsymbol{w}_1 \in \mathbb{R}^{3L}$ and $\boldsymbol{\theta}_{e,1}^{(h)} \in \mathbb{R}^{L \times L}$ are both learnable weights and $\rho(\cdot)$ refers to the LeakyReLU activation function with negative slope 0.2. In the second round, similarly, each variable node attends to its neighbors to get updated variable node embeddings

$$\boldsymbol{v}'_j = \frac{1}{H} \sum_{i=1}^{H} \left(\alpha_{jj,2}^{(h)} \boldsymbol{\theta}_{c,2}^{(h)} \boldsymbol{c}'_i + \sum_{j \in \mathcal{N}_i} \alpha_{ji,2}^{(h)} \boldsymbol{\theta}_{v,2}^{(h)} \boldsymbol{v}_j \right)$$

with attention weights

$$\alpha_{ji,2}^{(h)} = \frac{\exp(\boldsymbol{w}_2^T \rho([\boldsymbol{\theta}_{c,2}^{(h)} \boldsymbol{c}'_i, \boldsymbol{\theta}_{v,2}^{(h)} \boldsymbol{v}_j, \boldsymbol{\theta}_{e,2}^{(h)} \boldsymbol{e}_{i,j}]))}{\sum_{k \in \mathcal{N}_i} \exp(\boldsymbol{w}_2^T \rho([\boldsymbol{\theta}_{c,2}^{(h)} \boldsymbol{c}'_i, \boldsymbol{\theta}_{v,2}^{(h)} \boldsymbol{v}_j, \boldsymbol{\theta}_{e,2}^{(h)} \boldsymbol{e}_{i,k}]))}$$

where $\boldsymbol{w}_2 \in \mathbb{R}^{3L}$ and $\boldsymbol{\theta}_{c,2}^{(h)}, \boldsymbol{\theta}_{v,2}^{(h)}, \boldsymbol{\theta}_{e,2}^{(h)} \in \mathbb{R}^{L \times L}$ are learnable weights. After the two rounds of message passing, we have learned a MILP embedding consisting of (V_2, C_2) and it is pass to task-specific layers.

2 Benchmark Problem Descriptions and MILP Formulations

We present the MILP formulations for Combinatorial Auction (CA), Maximal Independent Set (MIS), Minimum Vertex Cover (MVC).

Table 1: Description of the constraint, edge and variable features in our bipartite state representation $P' = (G, C, E, V)$.

Tensor	Feature	Description
C	obj_cos_sim	Cosine similarity with objective.
	bias	Bias value, normalized.
	is_tight	Tightness indicator in LP solution.
	dualsol_val	Dual solution value, normalized.
E	coef	Constraint coefficient, normalized.
V	type_CONTINUOUS	Indicator of continuous variable
	type_BINARY	Indicator of binary variable
	type_INTEGER	Indicator of integer variable
	coef	Objective coefficient, normalized.
	has_lb	Lower bound indicator.
	has_ub	Upper bound indicator.
	sol_is_at_lb	Solution value equals lower bound.
	sol_is_at_ub	Solution value equals upper bound.
	sol_frac	Solution value fractionality.
	basis_status_BASIC	Indicator of basic simplex basis status.
	basis_status_NONBASIC_LOWER	Indicator of lower simplex basis status.
	basis_status_NONBASIC_UPPER	Indicator of upper simplex basis status.
	basis_status_SUPERBASIC	Indicator of zero simplex basis status.
	reduced_cost	Reduced cost, normalized.
	sol_val	Solution value.

2.1 CA

In a CA instance, we are given n bids $\{(B_i, p_i) : i \in [n]\}$ for m items, where B_i is a subset of items and p_i is its associated bidding price. The objective is to allocate items to bids such that the total revenue is maximized:

$$\begin{aligned} \max \sum_{i \in [n]} p_i x_i \\ \text{s.t.} \sum_{i: j \in B_i} x_i \leq 1, \forall j \in [m], \\ x_i \in \{0,1\}, \forall i \in [n]. \end{aligned}$$

2.2 MIS

In a MIS instance, we are given an undirected graph $G = (V, E)$. The goal is to select the largest subset of nodes such that no two nodes in the subsets are connected by an edge in G:

$$\begin{aligned} \max \sum_{v \in V} x_v \\ \text{s.t.} x_u + x_v \leq 1, \forall (u, v) \in E, \\ x_v \in \{0,1\}, \forall v \in V. \end{aligned}$$

2.3 MVC

In the MVC problem, we are given an undirected graph $G = (V, E)$ with a weight w_v associated with each node $v \in V$. The objective is to select a subset of nodes $V' \subseteq V$ with the minimum sum of weights such that for every edge in E, at least one of its endpoints is selected in V':

$$\begin{aligned} & \min \sum_{v \in V} w_v x_v \\ \text{s.t.} \quad & x_u + x_v \geq 1, \forall (u, v) \in E, \\ & x_v \in \{0, 1\}, \forall v \in V. \end{aligned}$$

3 Hyperparameter Settings

We present the hyperparameter settings during data collection and evaluation for our experiments. During data collection, for Backdoor, we collect 50 candidates backdoors for every instance and select 5 as positive samples and 5 as negative samples. For PaS, we collect 50 best found solutions for each training instance with an hour runtime. Then we collect 10 negative samples for each solutions. For Configuration, we collect 50 configuration through *SMAC* for every instance and select 5 as positive samples and 5 as negative samples.

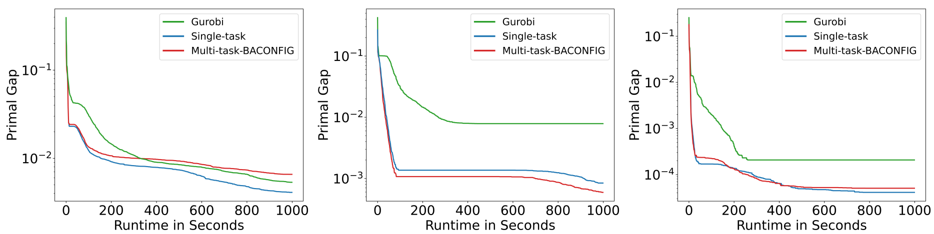
During the evaluation, for Backdoor, backdoor size K is set to 8 for all benchmarks. For PaS, (k_0, k_1, Δ) is set to $(1500, 0, 0)$ for CA, $(500, 500, 10)$ for MIS and $(500, 100, 15)$ for MVC.

4 Additional Tables and Figures

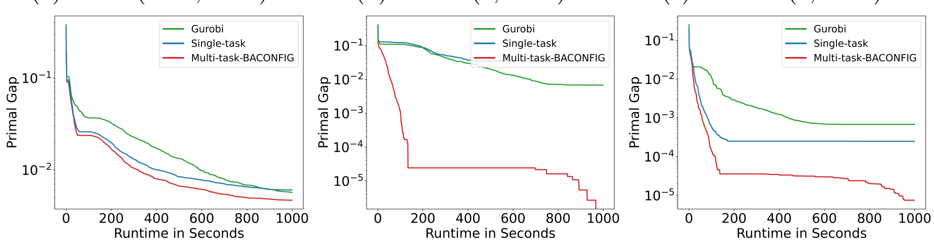
We provide supplementary results in the form of tables and figures to illustrate the performance and comparison of multitask models. Table 2 shows the full primal integral results of new task generalization of our multi-task model. Figure 1 shows the primal gap results of our cross-validation results on PaS. Table 3 and 4 and Figure 2 shows comparisons among different multitask models on the same task, Backdoor and PaS.

Table 2: Primal Integral for CONFIGURATION averaged over 100 test instances for each benchmark. We compare the performance of *SCIP*, *SMAC*, *Single-task* (trained on CONFIGURATION), and *Multi-task-BAPAS*. Results include the mean, standard deviation, and the number of instances each approach wins. The best-performing entries are highlighted in bold for clarity.

		CONFIGURATION Primal Integral		
Benchmarks	Approaches	Mean	Std Dev	Wins
CA-S (2000, 4000)	*SCIP*	597.36	206.55	13
	SMAC	598.30	207.27	9
	Single-task	489.10	**155.86**	34
	Multi-task-BAPAS	**485.84**	194.10	**44**
MIS-S (4, 3000)	*SCIP*	439.27	310.01	**37**
	SMAC	411.70	227.07	24
	Single-task	422.58	190.61	16
	Multi-task-BAPAS	**351.60**	**167.71**	23
MVC-S (4, 3000)	*SCIP*	576.36	213.6	0
	SMAC	264.36	185.60	4
	Single-task	87.35	60.55	37
	Multi-task-BAPAS	**72.94**	**58.54**	**59**
CA-L (3000, 6000)	*SCIP*	247.97	72.19	6
	SMAC	269.15	116.17	13
	Single-task	306.77	74.60	4
	Multi-task-BAPAS	**162.21**	**55.02**	**77**
MIS-L (5, 6000)	*SCIP*	782.65	**166.61**	1
	SMAC	732.51	176.52	5
	Single-task	768.61	172.79	4
	Multi-task-BAPAS	**234.45**	222.67	**90**
MVC-L (5, 6000)	*SCIP*	714.79	**176.49**	0
	SMAC	576.32	204.4	8
	Single-task	398.38	322.22	21
	Multi-task-BAPAS	**231.92**	282.19	**71**



(a) CA-S (2000, 4000) (b) MIS-S (5, 6000) (c) MVC-S (5, 6000)



(d) CA-L (3000, 6000) (e) MIS-L (5, 9000) (f) MVC-L (5, 9000)

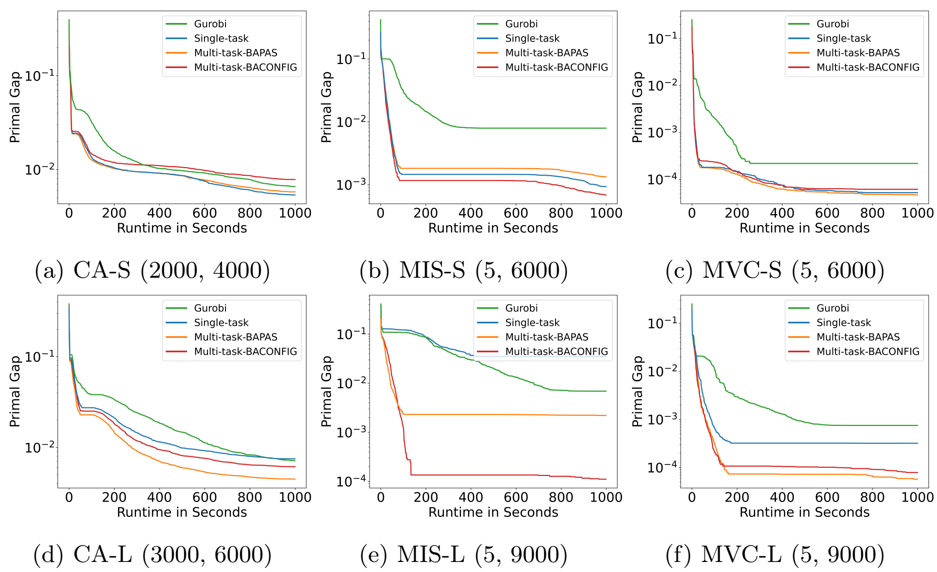
Fig. 1: The Primal Gap (the lower, the better) as a function of runtime averaged over 100 test instances on PaS and each benchmark. We compare the performance of *Gurobi* (green line), *Single-task* on PaS (blue line), and *Multi-task-BACONFIG* (red line).

Table 3: Solve Time (seconds) for Backdoor averaged over 100 test instances for each benchmark. We compare the performance of *SCIP*, *Single-task* (trained on Backdoor), same task multitask model *Multi-task-BAPAS*, and new task multitask model *Multi-task-PASCONFIG*. Results include the mean, standard deviation, and the number of instances each approach wins. The best-performing entries are highlighted in bold for clarity.

		Backdoor Solve Time		
Benchmarks	Approaches	Mean	Std Dev	Wins
CA-S (175, 850)	*Gurobi*	253.08	3	132.57
	Single-task	231.23	24	124.91
	Multi-task-BAPAS	218.75	27	121.10
	Multi-task-PASCONFIG	**209.06**	**46**	**117.74**
MIS-S (4, 1250)	*Gurobi*	183.51	24	**221.31**
	Single-task	172.50	**31**	**216.93**
	Multi-task-BAPAS	**165.02**	30	213.47
	Multi-task-PASCONFIG	**164.73**	15	**202.90**
MVC-S (5, 1500)	*Gurobi*	77.47	4	93.26
	Single-task	72.19	30	96.69
	Multi-task-BAPAS	71.43	31	91.52
	Multi-task-PASCONFIG	**67.70**	**35**	**78.47**
CA-L (200, 1000)	*Gurobi*	793.10	13	450.03
	Single-task	749.08	18	479.99
	Multi-task-BAPAS	708.07	31	**413.69**
	Multi-task-PASCONFIG	**696.39**	**38**	**419.18**
MIS-L (4, 1500)	*Gurobi*	530.41	17	935.79
	Single-task	470.73	27	837.01
	Multi-task-BAPAS	448.30	**33**	792.42
	Multi-task-PASCONFIG	**446.61**	23	**791.89**
MVC-L (5, 2000)	*Gurobi*	427.86	17	697.71
	Single-task	390.04	17	623.76
	Multi-task-BAPAS	379.41	20	653.14
	Multi-task-PASCONFIG	**354.35**	**46**	**607.36**

Table 4: Primal Integral for PaS averaged over 100 test instances for each benchmark. We compare the performance of *SCIP*, *Single-task* (trained on PaS), same task multitask model *Multi-task-BAPAS*, and new task multitask model *Multi-task-BACONFIG*. Results include the mean, standard deviation, and the number of instances each approach wins. The best-performing entries are highlighted in bold for clarity.

		PaS Primal Integral		
Benchmarks	Approaches	Mean	Std Dev	Wins
CA-S (2000, 4000)	*Gurobi*	15.09	**5.19**	3
	Single-task	10.26	5.34	36
	Multi-task-BAPAS	**10.23**	5.57	**39**
	Multi-task-BACONFIG	12.25	5.99	22
MIS-S (5, 6000)	*Gurobi*	17.03	3.78	0
	Single-task	3.58	1.44	31
	Multi-task-BAPAS	3.81	1.28	16
	Multi-task-PASCONFIG	**3.03**	**1.04**	**53**
MVC-S (5, 6000)	*Gurobi*	1.45	0.65	0
	Single-task	0.55	**0.10**	27
	Multi-task-BAPAS	**0.53**	0.13	31
	Multi-task-PASCONFIG	0.54	0.14	**42**
CA-L (3000, 6000)	*Gurobi*	21.46	5.98	3
	Single-task	15.58	5.90	11
	Multi-task-BAPAS	**11.10**	**5.14**	**55**
	Multi-task-PASCONFIG	13.60	5.80	31
MIS-L (5, 9000)	*Gurobi*	38.06	10.75	0
	Single-task	56.23	8.76	0
	Multi-task-BAPAS	4.51	1.82	29
	Multi-task-PASCONFIG	**3.24**	**0.95**	**71**
MVC-L (5, 9000)	*Gurobi*	3.91	2.28	0
	Single-task	1.56	0.57	5
	Multi-task-BAPAS	**1.11**	**0.44**	47
	Multi-task-PASCONFIG	**1.11**	0.46	**48**



(a) CA-S (2000, 4000)
(b) MIS-S (5, 6000)
(c) MVC-S (5, 6000)
(d) CA-L (3000, 6000)
(e) MIS-L (5, 9000)
(f) MVC-L (5, 9000)

Fig. 2: The Primal Gap (the lower, the better) as a function of runtime averaged over 100 test instances on PaS and each benchmark. We compare the performance of *Gurobi* (green line), *Single-task* on PaS (blue line), *Multi-task-BAPAS*(orange line), and *Multi-task-BACONFIG* (red line).