HINDSIGHT SUPERVISION FOR LEARNING TO BRANCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned branching rules for mixed-integer programming are usually trained by imitation: a network is fit to the choices of strong branching, whose score for a candidate is the immediate improvement in its children's LP bounds. That score is a one-step proxy. It ignores how the subtrees below the children are actually solved, so imitating it optimizes agreement with a heuristic rather than the size of the proof, and the label carries no information about how much each alternative would have cost. We instead supervise branching with the quantity of interest itself. On solved training instances, the known optimum fixes the objective cutoff, and each candidate branch is followed by a fixed continuation rule until its subtree is complete or a node budget is reached; completed rollouts give exact proof costs and interrupted rollouts give certified lower bounds. These observations define hard, soft, and set-valued training targets. A performance-difference identity for finite, subtree-local proof trees shows when such local improvements compose into a smaller proof, and a simple counterexample shows why a stateful solver calls for closed-loop evaluation instead. Controlled experiments on three MILP families show that hindsight targets reduce held-out tree size relative to matched behavior cloning on every family, by up to 45%, and information-source controls locate the origin of the gain: correct cost-to-action assignment on multidimensional knapsack, a cheap multi-action target on combinatorial auctions, and the composition of the proposal set on set cover. Hindsight targets also beat a compute-matched Branch Ranking collector given the same cutoff, transfer to a harder auction cohort, and cut solving time by 7% over behavior cloning in a practical SCIP configuration that receives no test optimum.

1 INTRODUCTION

Branching determines the shape of a branch-and-bound tree for a mixed-integer linear program (MILP). Strong branching tests candidate variables by solving their child LP relaxations and scoring immediate bound gains. Imitation learning can approximate this costly rule at deployment (Achterberg et al., 2005; Gasse et al., 2019; Zarpellon et al., 2021), but matching a one-step choice does not directly optimize the work required to prove optimality.

Solved training instances offer a more direct source of supervision. Given an optimal solution and objective value, we can fix the objective cutoff and compare alternative branches by completing their subtrees under a specified continuation rule. The resulting node count is a proof cost that, unlike a strong-branching score, accounts for later decisions and fathoming. Using this signal well raises two questions. First, how should rollouts that exhaust their budget enter the training target? Second, when a policy trained on such labels improves, how much of the gain comes from the measured costs rather than from cheaper by-products of the collection process, such as exposure to several plausible actions per state?

We answer both questions. We collect counterfactual rollouts from a common saved state, record each as an exact cost or a certified lower bound, and train policies on hard, soft, and set-valued targets built from these observations. A performance-difference identity for finite proof trees shows when local improvements compose into a smaller proof and makes explicit what a stateful solver adds. In the experiments, we keep the recorded states and encoder fixed while changing only the target, and we compare against inexpensive controls that keep the same support or target concentration but scramble or discard the observed costs.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Hindsight targets reduce held-out tree size relative to matched behavior cloning on multidimensional knapsack, combinatorial auctions, and set cover, but the controls show that the gain has a different origin on each family. Correct cost-to-action assignment drives the improvement on knapsack; on auctions, a uniform target over a handful of strong-branching candidates already delivers much of it, and the measured winner delivers the rest; on set cover, the composition of the proposal set matters most. We further compare against Branch Ranking with the same fixed cutoff and collection budget, evaluate transfer to a larger auction cohort, and measure wall-clock time in a SCIP configuration that receives no test optimum.

Contributions. Solved instances, counterfactual rollouts, and fixed cutoffs have each appeared before; our contribution is to combine them into a supervision signal whose observations are exact or certified, to say when that signal provably helps, and to measure where its benefit actually comes from.

1. **Proof-cost supervision from solved instances.** A labeling pipeline that restores a common solver state, evaluates every candidate branch under the same optimal cutoff and continuation rule, and records each outcome as an exact node count or a certified lower bound. Hard, soft, and set-and-ranking targets are defined on these interval observations without imputing the cost of interrupted rollouts (Section 4).

2. **A composition guarantee and its limits.** A performance-difference identity for finite proof trees, with a headroom-versus-error bound, that states exactly when local counterfactual improvements add up to a smaller proof, and a counterexample showing why the additive guarantee fails in a stateful solver, so that closed-loop evaluation is required (Section 5).

3. **Attribution, not just improvement.** A matched evaluation on three MILP families with information-source controls that hold the states, encoder, and target support fixed while removing proposal composition, multi-action support, or cost assignment one at a time, complemented by a fixed-cutoff Branch Ranking comparison, transfer and scale tests, and a practical SCIP evaluation without a test optimum (Section 6).

2 Related work

Imitation-based branchers use hand-crafted features (Khalil et al., 2016), bipartite graphs (Gasse et al., 2019), hybrid encoders (Gupta et al., 2020), tree context (Zarpellon et al., 2021), or sequence models (Zhang et al., 2026). Lookback modifies strong-branching labels using parent-child score relations (Gupta et al., 2022). These works learn a cheaper or better local decision rule; the label remains a one-step score. Retrospective trajectories learn from paths in observed search trees (Parsonson et al., 2023), whereas our counterfactuals evaluate alternative actions from a common saved state.

Long-horizon learning for branching includes tree-structured reinforcement learning (Etheve et al., 2020; Scavuzzo et al., 2022); the optimal-objective limit we use during training was introduced by Scavuzzo et al. (2022). Branch Ranking collects alternative-action rollouts and learns a ranking policy (Huang et al., 2022), and lifted branching evaluates alternatives from copied solver states and imitates the improved policy, estimating the final size of unfinished trees (Renard et al., 2026). We build on these ideas: our contribution is a fixed-cutoff proof-cost pipeline whose observations are exact counts or certified lower bounds rather than estimates, together with matched controls that separate proposal support from cost-based supervision.

Post-solve tree compression searches for smaller certificates (Muñoz et al., 2023); our evaluations produce training labels rather than rewriting a tree. Learner-state relabeling follows dataset aggregation (Ross et al., 2011). Full-tree MDP formulations (Strang et al., 2025) make explicit the coupling through shared incumbents and open nodes that a subtree-local model abstracts away; Appendix B connects the two views.

3 PROOF COSTS AND THE LEARNING PROBLEM

Consider a feasible MILP with an attained finite optimum,

$$z^\star = \min_x \left\{ c^\top x : Ax \le b,\ \ell \le x \le u,\ x_i \in \mathbb{Z}\ (i \in I) \right\}. \tag{1}$$

A branching state contains a subproblem and the information used to select an LP solution and a branching action. Given a fractional LP coordinate $\hat{x}_j$, variable branching imposes $x_j \le \lfloor \hat{x}_j \rfloor$ or $x_j \ge \lceil \hat{x}_j \rceil$. We write $\mathcal{A}(s)$ for the admissible actions and $s^-(a), s^+(a)$ for the children. With a fixed LP solution, an action is indexed by j; replay also preserves its split point.

Fixed-cutoff proof search. During label generation, the cutoff is $\bar{z} = z^\star$. A node is terminal when its relaxation is infeasible, its LP lower bound is at least $\bar{z}$, or its LP solution is integer feasible. A complete tree certifies that no feasible point has objective below $\bar{z}$; together with a feasible solution $x^\star$ satisfying $c^\top x^\star = \bar{z}$, it proves optimality. The analysis uses exact arithmetic; in computation, fathoming follows the solver's feasibility and bound tolerances.

Assumption 1 (Finite, subtree-local dynamics)**.** *Integer variables have finite bounds, rounded to integer endpoints before solving relaxations. Every reachable nonterminal LP has an attained finite optimum and a deterministically selected solution; infeasible nodes are terminal. Terminal status, admissible actions, and child states are determined by the current state and action. Processing one subtree does not change the transitions or decisions in another subtree through shared mutable statistics.*

Assumption 1 describes a controlled proof-tree model; Section 5 and Appendix B relate it to a full solver. It makes the reachable branching structure finite: along every nonterminal edge, at least one finite integer domain is strictly reduced. Under a deterministic policy ρ, let $C_\rho(s)$ count all nodes, including terminal nodes:

$$C_\rho(s) = \begin{cases} 1, & s \text{ terminal}, \\ 1 + C_\rho(s^-(\rho(s))) + C_\rho(s^+(\rho(s))), & \text{otherwise}. \end{cases} \tag{2}$$

The cost of choosing a once and then following ρ is

$$Q_\rho(s,a) = 1 + C_\rho(s^-(a)) + C_\rho(s^+(a)). \tag{3}$$

Thus $Q_\rho(s, \rho(s)) = C_\rho(s)$. Note that Q_ρ is a *continuation-policy cost*: it measures the proof completed by ρ, not the smallest possible proof.

Operational state and observations. An actual solver snapshot σ also carries pseudocost history, local constraints, the open-node list, an LP basis, and random state. Let $Q_\rho^{\text{reset}}(\sigma, a)$ denote the node count obtained by evaluating a from that snapshot and following ρ within the reconstructed subtree. Restoring the same snapshot for every candidate makes these evaluations directly comparable. We write Q_ρ for the tree model and Q_ρ^{reset} for measured rollouts; the learning targets below apply to either.

4 HINDSIGHT COUNTERFACTUAL SUPERVISION

4.1 BUDGETED COUNTERFACTUAL EVALUATION

A behavior policy μ supplies recorded states on solved training instances. At each retained state, we form a proposal set $\mathcal{E}(s) \subseteq \mathcal{A}(s)$ that contains the behavior action; the guarantee of Section 5 additionally requires the continuation action $\rho(s)$. Every candidate starts from an independent restoration of the same solver snapshot and uses the same cutoff, continuation rule, and cost convention. The behavior subtree serves as the reference cost when it was produced under these same conditions; otherwise the behavior action is re-evaluated like any other candidate.

For each proposal, the observation is an interval

$$[L_{s,a}, U_{s,a}] = \begin{cases} [q_{s,a}, q_{s,a}], & \text{rollout completes with } q_{s,a} \text{ nodes}, \\ [n_{s,a}, +\infty), & \text{rollout stops after } n_{s,a} \text{ counted nodes}. \end{cases} \tag{4}$$

An interrupted rollout cannot finish with fewer nodes than it has already counted, so the lower bound is exact rather than estimated; $n_{s,a}$ is a node count, not an LP or wall-time budget. Appendix C gives the collection algorithm and restoration checks.

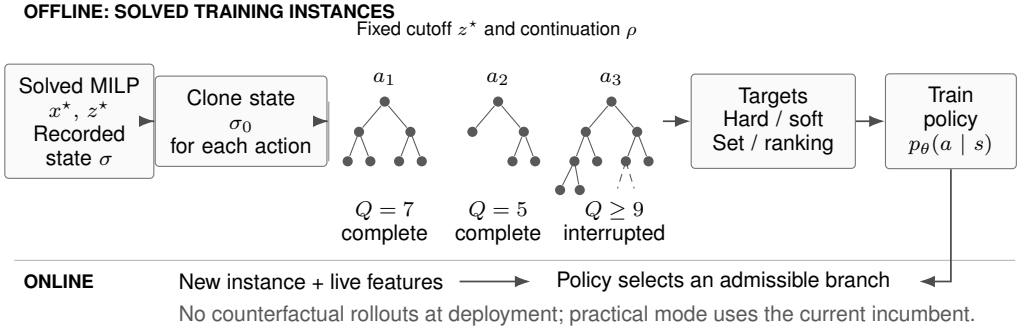


Figure 1: Overview. Labeling is offline: every candidate starts from the same saved state and uses the same cutoff and continuation rule (counts are illustrative). At deployment the policy uses live features and the current incumbent; no test optimum or counterfactual rollouts are needed.

4.2 Targets from exact and censored costs

Let $\mathcal{X}(s) = \{a \in \mathcal{E}(s) : L_{s,a} = U_{s,a} < \infty\}$ be the exactly evaluated proposals, and write q_a for an exact cost. The hard, soft, and set targets use states with $\mathcal{X}(s) \neq \varnothing$. Set $q_{\min} = \min_{a \in \mathcal{X}(s)} q_a$. Given policy features $\phi(s)$ and logits $f_\theta(s, a)$, define

$$p_\theta(a \mid s) = \frac{\exp f_\theta(s, a)}{\sum_{b \in \mathcal{A}(s)} \exp f_\theta(s, b)}. \tag{5}$$

The admissible-candidate mask is fixed across target variants at a state. At deployment the policy chooses the highest-scoring admissible action.

Hard and soft targets. The hard target selects $a^*(s) \in \arg\min_{a \in \mathcal{X}(s)} q_a$, with deterministic tie-breaking. The soft target retains relative cost information:

$$\mathcal{L}_{\mathrm{B}}(s) = -\log p_\theta(a^*(s) \mid s), \tag{6}$$

$$y_s(a) = \frac{(1 + q_a)^{-\tau}}{\sum_{b \in \mathcal{X}(s)} (1 + q_b)^{-\tau}}, \quad a \in \mathcal{X}(s), \qquad \mathcal{L}_{\mathrm{C}}(s) = -\sum_{a \in \mathcal{X}(s)} y_s(a) \log p_\theta(a \mid s). \tag{7}$$

Here $\tau > 0$ controls target concentration. Both targets live on $\mathcal{X}(s)$: a censored candidate is excluded rather than treated as worse, and unmeasured actions receive no target mass.

Set likelihood and certified ranking. For $\varepsilon \geq 0$, define the measured near-optimal set and the certified ordered pairs by

$$\mathcal{G}_\varepsilon(s) = \{a \in \mathcal{X}(s) : q_a \leq (1 + \varepsilon) q_{\min}\}, \qquad \mathcal{P}(s) = \{(a, b) \in \mathcal{E}(s)^2 : U_{s,a} < L_{s,b}\}. \tag{8}$$

A pair can compare two completed evaluations, or an exact cost against a censored lower bound that already exceeds it. The corresponding losses are

$$\mathcal{L}_{\mathrm{set}}(s) = -\log \sum_{a \in \mathcal{G}_\varepsilon(s)} p_\theta(a \mid s), \tag{9}$$

$$\mathcal{L}_{\mathrm{rank}}(s) = \frac{1}{\max\{1, |\mathcal{P}(s)|\}} \sum_{(a,b) \in \mathcal{P}(s)} \mathrm{softplus}(f_\theta(s, b) - f_\theta(s, a)), \tag{10}$$

$$\mathcal{L}_{\mathrm{D}}(s) = w(s) \left[\mathcal{L}_{\mathrm{set}}(s) + \lambda \mathcal{L}_{\mathrm{rank}}(s)\right]. \tag{11}$$

We use $\lambda \geq 0$; an empty pair set gives zero ranking loss. The nonnegative weight $w(s)$ emphasizes large subtrees with measured headroom (Appendix C). Because D differs from C in the set loss and weights as well as in its use of censored pairs, we isolate the contribution of censored pairs with a separate loss-matched ablation that removes only the exact-censored terms (Section 6).

4.3 SEPARATING SUPPORT FROM COST INFORMATION

Hindsight labels differ from behavior cloning in three ways at once: they spread mass over several actions, they use a proposal set assembled with the optimum in hand, and they distribute that mass according to measured costs. To attribute gains, we train controls that remove these ingredients one at a time. Uniform-over-exact supervision (U) assigns $1/|\mathcal{X}(s)|$ to each completed proposal in Eq. (7). USB assigns uniform mass to the top eight SB-ranked candidates, using neither $x^\star$ nor rollouts. RND selects a random completed proposal as a one-hot label, matching the hard target's concentration without its cost ordering. CSH permutes the exact costs across proposals before computing the soft target, preserving its support and per-state entropy while breaking the association between action and cost. USB against BC measures the value of multi-action support alone, U against USB measures proposal composition, and B against RND or C against CSH measures cost assignment.

Dataset aggregation (Ross et al., 2011) can optionally relabel states visited by the learner with the same collector; with ρ fixed, the labels remain costs under ρ.

5 WHEN DO LOCAL IMPROVEMENTS REDUCE A PROOF TREE?

This section works in the finite, subtree-local model of Assumption 1. Let $\mathcal{I}(\mathcal{T}_\pi(s_0))$ denote the internal node *occurrences* in the tree generated by π from s_0; we count occurrences because identical subproblems may appear more than once. Define the advantage relative to ρ as

$$A_\rho(s,a) = Q_\rho(s,a) - C_\rho(s). \tag{12}$$

A negative advantage means that choosing a and then continuing with ρ shrinks the local proof.

Theorem 1 (Proof-tree performance difference)**.** *For deterministic policies π and ρ satisfying Assumption 1,*

$$C_\pi(s_0) - C_\rho(s_0) = \sum_{v \in \mathcal{I}(\mathcal{T}_\pi(s_0))} A_\rho(s_v, \pi(s_v)). \tag{13}$$

Consequently, $A_\rho(s,\pi(s)) \le 0$ at every internal state reached by π implies $C_\pi(s_0) \le C_\rho(s_0)$.

Proof. Subtract the recursions for $C_\pi(s)$ and $C_\rho(s)$, adding and subtracting the two ρ-costs at the children selected by $\pi(s)$. This yields $C_\pi(s) - C_\rho(s) = A_\rho(s,\pi(s)) + \sum_{d\in\{-,+\}}(C_\pi - C_\rho)(s^d(\pi(s)))$. Expand over the finite tree. The terminal differences vanish, giving Eq. (13). □

Corollary 2 (Proposal-restricted improvement)**.** *Suppose $\rho(s) \in \mathcal{E}(s)$ at every state reached by π. If $\pi(s) \in \arg\min_{a\in\mathcal{E}(s)} Q_\rho(s,a)$, then $C_\pi(s_0) \le C_\rho(s_0)$. The same conclusion holds if, at every visited state, intervals valid for the theoretical Q_ρ satisfy $U_{s,\pi(s)} \le L_{s,\rho(s)}$.*

The premise concerns the states that π itself visits. A classifier fit to oracle choices on behavior states satisfies it only approximately, and including the behavior action in $\mathcal{E}(s)$ is not enough when $\mu \neq \rho$; the bound below quantifies the resulting slack.

Headroom and learning error. Let $m_\mathcal{E}(s) = \min_{a\in\mathcal{E}(s)} Q_\rho(s,a)$ and assume $\rho(s) \in \mathcal{E}(s)$. Define

$$h_\mathcal{E}(s) = C_\rho(s) - m_\mathcal{E}(s), \qquad e_\mathcal{E}(s) = [Q_\rho(s,\pi(s)) - m_\mathcal{E}(s)]_+ . \tag{14}$$

Both are nonnegative, even if the learned policy selects outside $\mathcal{E}(s)$. Equation (13) gives

$$C_\pi(s_0) - C_\rho(s_0) \le \sum_{v\in\mathcal{I}(\mathcal{T}_\pi(s_0))} \big(e_\mathcal{E}(s_v) - h_\mathcal{E}(s_v)\big). \tag{15}$$

The learner improves on ρ whenever the proposal-restricted headroom accumulated along its *own* tree exceeds its accumulated action-cost error; when π always selects from $\mathcal{E}$, the bound is an equality. Two practical consequences follow: what matters is headroom at learner-visited states rather than the size of training subtrees, and closed-loop evaluation is the natural way to measure the balance. Appendix A gives an approximate-value version and separates proposal coverage from selection error.

Family	Encoder	B / BC	C / BC	D / BC
MKP S1	TreeGate-lite	$0.919\ [0.910, 0.928]$	$0.916\ [0.902, 0.930]$	$0.990\ [0.959, 1.023]$
MKP S1	GNN	$0.929\ [0.909, 0.946]$	$0.891\ [0.859, 0.930]$	–
CA	TreeGate-lite	$0.549\ [0.503, 0.600]$	$0.619\ [0.556, 0.720]$	–
CA	GNN	$0.689\ [0.631, 0.747]$	$0.805\ [0.742, 0.881]$	–
Set cover	TreeGate-lite	–	–	$0.985\ [0.867, 1.109]$
Set cover	GNN	–	$0.855\ [0.706, 0.958]$	$0.879\ [0.722, 0.977]$

Table 1: Held-out closed-loop tree size relative to BC with the same encoder (paired ratios with 95% intervals; lower is better). A dash denotes a cell that was not run.

Censoring and solver scope. If $U_{s,a} < L_{s,b}$, interval validity certifies $Q_\rho(s,a) < Q_\rho(s,b)$ without imputation, and the same ordering holds for measured Q_ρ^{reset} when both evaluations use the common snapshot. In a stateful solver, however, processing one subtree can change the pseudocosts, cuts, or incumbent seen by its siblings. A two-subtree example makes the consequence concrete. A baseline processes open subtrees L and R in 5 and 3 nodes, for a total of $1 + 5 + 3 = 9$ with their parent. An alternative action at the parent solves L in 3 nodes but updates shared statistics so that the baseline then needs 101 nodes on R. An isolated evaluation correctly reports the local improvement $3 < 5$, while the complete search costs $1 + 3 + 101 = 105$. Restoring the initial state between candidate trials removes coupling *between trials*, not the downstream coupling *after an action is deployed*; the additive identity then holds only for the complete search state and its cost-to-go (Appendix B). Subtree-local labels are therefore a training signal whose closed-loop value we measure directly, rather than a guarantee.

6 Experiments

Design and metrics. We use a fixed-$z^\star$ branch-and-bound environment for controlled comparisons. It solves LP relaxations, branches on fractional variables, and disables in-tree cuts and primal heuristics. A reliability-style strong-branching rule (SB) supplies behavior states. We train behavior cloning (BC), hard hindsight (B), soft hindsight (C), and set-and-ranking hindsight (D) on matched state sets and encoders. TreeGate-lite uses candidate and tree features; a bipartite GNN tests a structural representation. Appendix C details data, masks, budgets, and checkpoint selection.

The primary outcome is a paired geometric mean of per-run node ratios, with smaller values better. All reported contrasts are direct paired ratios, and the 95% intervals come from a hierarchical bootstrap that resamples generator splits, model seeds, and instances while preserving instance pairing (Appendix C). Unless stated otherwise, target comparisons use three generator splits and five model seeds.

The label collector restores each candidate from a common state with its LP and pseudocost history, uses at most eight proposals per state, and spends at most 20 times the behavior proof's LP solves per instance. A byte-exact replay audit reproduced stored costs, completion flags, and winners on 64 set-cover and 36 CA states and was invariant to candidate order; altering the preceding LP-solve history changed some set-cover costs and winners (Appendix C). Labels are therefore costs under the recorded operational state, and we evaluate every policy in closed loop.

6.1 Matched learners and information sources

Hindsight targets improve on matched BC in every family (Table 1). The hard and soft targets reduce tree size by 7–11% on MKP S1 and by 20–45% on CA, and the GNN improves by 12–15% on set cover under C and D. On MKP S1, hard and soft targets are interchangeable for TreeGate-lite (C/B $= 0.997\ [0.986, 1.009]$). CA shows the largest gain, and there the GNN hard target comes within 6% of the SB rule that generated the training states (B/SB $= 1.056\ [0.977, 1.149]$; $1.085\ [1.037, 1.147]$ for TreeGate-lite). The hard and soft targets are the most robust across encoders; the set-and-ranking target D improves the GNN on set cover and depends more on the encoder, so encoder choice matters alongside target choice.

Comparison	CA / GNN	Set cover / GNN	MKP S1 / TreeGate-lite
USB / BC	$0.803\ [0.708, 0.923]$	$1.981\ [1.316, 2.830]$	$1.041\ [1.012, 1.071]$
U / USB	$1.019\ [0.953, 1.091]$	$0.438\ [0.295, 0.669]$	$1.038\ [1.004, 1.073]$
U / BC	$0.818\ [0.753, 0.893]$	$0.867\ [0.721, 0.979]$	$1.080\ [1.033, 1.112]$
RND / BC	$0.828\ [0.764, 0.906]$	$0.966\ [0.756, 1.238]$	$1.132\ [1.076, 1.201]$
B / RND	$0.832\ [0.796, 0.876]$	–	$0.811\ [0.764, 0.856]$
D / RND	–	$0.910\ [0.777, 1.005]$	–
CSH / BC	$0.818\ [0.751, 0.898]$	$0.904\ [0.734, 1.059]$	$1.092\ [1.057, 1.124]$
C / CSH	$0.983\ [0.968, 0.999]$	$0.947\ [0.878, 0.995]$	$0.839\ [0.811, 0.863]$

Table 2: Information-source controls on matched states, encoders, and model seeds (paired ratios with 95% intervals; lower is better). USB: uniform over the SB top eight; U: uniform over completed proposals; RND: random one-hot on that support; CSH: costs permuted on the same support, preserving soft-target entropy. D/RND also changes the objective and weighting, so it is not a pure cost-assignment contrast.

Family / states	Count	Achieved-cost recall@8	High-impact recall	Median / p90 regret
MKP S1	485	1.00	1.00	0.00 / 0.00
MKP S3	529	1.00	1.00	0.00 / 0.00
MKP learner states	225	1.00	1.00	0.00 / 0.00
CA	445	0.76	0.64	0.00 / 0.22
Set cover	486	0.84	0.69	0.00 / 0.15

Table 3: Coverage of the stored eight proposals relative to a broader audit pool with additional SB-prioritized and random candidates. Recall is the fraction of states whose audit winner is among the stored proposals; high-impact states are the top quartile by measured headroom; regret is the relative excess cost of the best stored proposal over the audit winner.

Table 2 attributes the gain to its sources, and the answer differs by family. On CA, USB already improves over BC, and U/USB $= 1.019\ [0.953, 1.091]$: the full proposal mixture adds nothing over the SB top eight for a uniform target. The measured winner still matters for a sharp label, B/RND $= 0.832\ [0.796, 0.876]$, whereas for the soft target at $\tau = 3$ the correct cost assignment contributes less, C/CSH $= 0.983\ [0.968, 0.999]$. On CA, then, much of the B/BC gain is available from a cost-free multi-action target, and the rest requires identifying the measured winner.

MKP is the opposite case. USB/BC $= 1.041\ [1.012, 1.071]$ and U/USB $= 1.038\ [1.004, 1.073]$, so both uniform targets are worse than BC, as are RND and CSH, while the cost-informed targets beat their matched controls clearly, B/RND $= 0.811\ [0.764, 0.856]$ and C/CSH $= 0.839\ [0.811, 0.863]$: here the improvement comes from assigning costs to the right actions. Set cover is different again. USB/BC $= 1.981\ [1.316, 2.830]$ but U/USB $= 0.438\ [0.295, 0.669]$, so the proposal mixture, which combines SB-prioritized, $x^\star$-aware, random, and diverse candidates, is what makes multi-action supervision work on this family; cost assignment adds a further C/CSH $= 0.947\ [0.878, 0.995]$. Removing the $x^\star$-aware component worsens U by $1.212\ [1.061, 1.384]$ (Appendix D.1): optimum-guided proposals help, but random and diverse proposals carry the larger part of the effect.

A broader-reference audit re-evaluated roughly 500 behavior states per family and 225 learner-visited states on MKP with additional SB-prioritized and random candidates (Table 3). The stored eight proposals contain the audit winner on every MKP state, including learner-visited ones, and on 76% of CA and 84% of set-cover states. On CA and set cover a larger pool would therefore expose additional headroom beyond what the present labels already use.

Incomplete evaluations. Only 0.3–0.8% of stored evaluations are interrupted on CA and set cover, and 2.6% on MKP, so the exact costs carry almost all of the information in these datasets. The rollout budget that produces them matters: reducing the per-rollout cap from $1.5S$ to S, where S is the reference subtree size, worsens the set-cover exact-only target by $1.239\ [1.105, 1.381]$. The interval convention keeps the truncated rollouts from being misread as completed costs, and a loss-matched ablation that adds the certified exact-censored pairs to the set-and-ranking loss, holding

Comparison, MKP S1	Paired ratio	95% interval
Branch Ranking (BR) / BC	0.995	[0.970, 1.022]
BR with fixed $z^\star$ / BC	0.932	[0.917, 0.948]
C / BR with fixed $z^\star$	0.976	[0.967, 0.985]
C / original BR	0.914	[0.893, 0.936]

Table 4: Compute-matched collectors on one MKP generator split, 100 held-out instances, and five model seeds. Each collector has a budget of 20 times the reference proof's LP solves.

	Controlled B&B			SCIP proof search		
MKP scale	SB nodes†	B/BC	C/BC	BC nodes	C nodes	C/BC
S1: 32×6	≈ 200	0.919	0.916	75.9	65.4	0.862
S2: 44×8	≈ 520	0.799	0.782	233.8	185.2	0.792
S3: 56×8	≈ 870	0.818	0.820	473.8	337.0	0.711

Table 5: Separate MKP training at each scale. †Approximate SB node geometric means. The SCIP proof-search columns use separate 40-instance cohorts and selected seed-0 policies, so the two halves are different metrics on different cohorts.

its exact-exact terms, states, and normalization fixed, leaves closed-loop performance unchanged (Appendix D.1).

6.2 Comparison with rollout supervision

BR samples alternative actions and combines long-term and behavior supervision (Huang et al., 2022). Giving its collector the same fixed optimal cutoff improves BR under this budget, from 0.995 to 0.932 relative to BC (Table 4), and C improves further over this stronger variant, C/BR $= 0.976\ [0.967, 0.985]$. Because the cutoff changes which rollouts complete within a fixed budget, this compares full pipelines rather than losses on identical observations.

6.3 Transfer, scale, and practical evaluation

A frozen GNN checkpoint trained on CA instances with 200 items and 800 bids was evaluated on a harder held-out cohort with 300 items and 1200 bids. The improvement over BC transfers: B/BC $= 0.671\ [0.591, 0.785]$ and C/BC $= 0.744\ [0.650, 0.867]$ on 40 instances and five model seeds (further cohort statistics in Appendix D).

Larger MKP instances benefit more from hindsight targets relative to scale-matched BC (Table 5); each scale is trained separately. The proof-only SCIP evaluation injects an optimal incumbent and disables cuts, primal heuristics, conflicts, restarts, and symmetry handling; its C/BC node ratio improves from 0.862 at S1 to 0.711 at S3. Additional SCIP baselines and work counters are in Appendix E.

The practical configuration uses the current incumbent, enables cuts, primal heuristics, and conflict analysis, and disables presolve and restarts so that the learned policy's variable space is preserved. Neither the training cutoff $z^\star$ nor the optimal solution used for offline proposals is available at test time. C reduces solving time by 7% over matched BC on the 80-instance cohort and by 19% over `relpscost` (Table 6); the latter gap also includes the saving from replacing an expensive brancher with a learned one. A matched USB policy gains only $0.975\ [0.951, 0.999]$ over BC, and C/USB $= 0.956\ [0.936, 0.977]$: in wall-clock terms most of C's saving requires the measured costs, a larger share than in the controlled node comparison. In this batch C expands 16% fewer nodes than BC and performs 10% fewer LP iterations, with inference overhead near 4% of solving time (Appendix E). On the harder CA cohort, all 40 practical runs solve within 600 seconds, with C/BC time $0.946\ [0.884, 1.011]$.

Time and node counts answer different questions. SB-style probing spends LP work outside the main search-node count, so a policy may expand more nodes yet use less wall time; the matched CA timing cohort is the practical comparison, and the proof-only node counts in Table 5 isolate the

CA branching rule	Shifted GM time (s)	Paired time / BC	Paired time / relpscost
`relpscost`	1.51	–	1.000
BC	1.19	1.000	0.874 [0.848, 0.901]
USB (matched)	1.14	0.975 [0.951, 0.999]	–
B	1.16	0.987 [0.962, 1.013]	–
C	1.04	0.932 [0.911, 0.952]	0.814 [0.793, 0.836]

Table 6: Practical SCIP on the same 80 CA instances in one idle-machine batch, without a test optimum. All methods solved every instance. USB is a TreeGate-lite policy trained and selected exactly like BC, B, and C. Times are shifted geometric means; the ratio columns are paired.

search-work effect. Offline labeling consumed 0.7, 3.1, and 4.3 CPU-hours on MKP S1–S3, 24.6 on CA, and 7.4 on set cover, excluding training optima, model training, and selection (Appendix E).

Scope. The controlled comparisons concern homogeneous instance families with matched encoders, and the practical timing results use one SCIP configuration without presolve and restarts. Appendix F reports an exploratory experiment that applies the same labels to heterogeneous MILP-S instances with a sequence-model brancher.

7 Discussion and conclusion

Proof costs measured on solved training instances are a useful supervision signal for learned branching. With a fixed optimal cutoff and a common saved state, counterfactual rollouts yield exact costs or certified lower bounds, and the resulting hard, soft, and set-valued targets reduce held-out tree size relative to matched behavior cloning on all three MILP families we tested, beat a compute-matched Branch Ranking collector given the same cutoff, transfer to harder auction instances, and reduce solving time in a practical SCIP configuration that never sees the test optimum.

The controls make the result more informative than a single improvement number. The gain comes from cost-to-action assignment on MKP, from the measured winner on top of a cheap multi-action target on CA, and from the composition of the proposal set on set cover, to which random and diverse proposals contribute more than the optimum-guided ones. Any evaluation of hindsight-style supervision should therefore report the corresponding support and cost controls; without them, label smoothing or proposal selection can be mistaken for the effect of proof costs. On the theoretical side, the proof-tree identity gives a sufficient condition for improvement in a finite, subtree-local model and explains why closed-loop evaluation is the right test in a stateful solver.

The framework suggests several extensions. Relabeling at learner-visited states targets the headroom exactly where Theorem 1 locates it; broader proposal pools on CA and set cover would expose the additional headroom identified by the coverage audit; and larger instances, on which interrupted rollouts are more common, bring the certified comparisons into play.

REFERENCES

Tobias Achterberg, Thorsten Koch, and Alexander Martin. Branching rules revisited. *Operations Research Letters*, 33(1):42–54, 2005.

Marc Etheve, Zacharie Alès, Côme Bissuel, Olivier Juan, and Safia Kedad-Sidhoum. Reinforcement learning for variable selection in a branch and bound algorithm. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)*, 2020.

Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2019.

Prateek Gupta, Maxime Gasse, Elias B. Khalil, M. Pawan Kumar, Andrea Lodi, and Yoshua Bengio. Hybrid models for learning to branch. In *Advances in Neural Information Processing Systems*, 2020.

Prateek Gupta, Elias B. Khalil, Didier Chételat, Maxime Gasse, Yoshua Bengio, Andrea Lodi, and M. Pawan Kumar. Lookback for learning to branch. *Transactions on Machine Learning Research*, 2022.

Zeren Huang, Wenhao Chen, Weinan Zhang, Chuhan Shi, Furui Liu, Hui-Ling Zhen, Mingxuan Yuan, Jianye Hao, Yong Yu, and Jun Wang. Branch ranking for efficient mixed-integer programming via offline ranking-based policy learning. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, volume 13717 of *Lecture Notes in Computer Science*, pp. 377–392, 2022. doi: 10.1007/978-3-031-26419-1_23.

Elias B. Khalil, Pierre Le Bodic, Le Song, George Nemhauser, and Bistra Dilkina. Learning to branch in mixed integer programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

Gonzalo Muñoz, Joseph Paat, and Álinson S. Xavier. Compressing branch-and-bound trees. In *Integer Programming and Combinatorial Optimization (IPCO)*, 2023.

Christopher W. F. Parsonson, Alexandre Laterre, and Thomas D. Barrett. Reinforcement learning for branch-and-bound optimisation using retrospective trajectories. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4061–4069, 2023. doi: 10.1609/aaai.v37i4.25521.

Simon Renard, Quentin Louveaux, and Bernard Fortz. Lifted branching: learning to improve branching strategies. *European Journal of Operational Research*, 2026. doi: 10.1016/j.ejor.2026.07.046.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, 2011.

Lara Scavuzzo, Feng Yang Chen, Didier Chételat, Maxime Gasse, Andrea Lodi, Neil Yorke-Smith, and Karen Aardal. Learning to branch with tree MDPs. In *Advances in Neural Information Processing Systems*, 2022.

Paul Strang, Zacharie Alès, Côme Bissuel, Olivier Juan, Safia Kedad-Sidhoum, and Emmanuel Rachelson. A markov decision process for variable selection in branch & bound. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Giulia Zarpellon, Jason Jo, Andrea Lodi, and Yoshua Bengio. Parameterizing branch-and-bound search trees to learn branching policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Ce Zhang, Bin Zhang, and Guoliang Fan. Towards better branching policies: Leveraging the sequential nature of branch-and-bound tree. In *International Conference on Learning Representations*, 2026.

A MATHEMATICAL DETAILS

A.1 NOTATION AND TERMINATION

Symbol	Meaning
μ, ρ, π	Behavior, continuation, and evaluated learned policy
s, σ	Subtree-local theoretical state; operational solver snapshot
$\mathcal{A}(s), \mathcal{E}(s), \mathcal{X}(s)$	Admissible, proposed, and exactly evaluated actions
$C_\rho(s)$	Complete proof-tree node count under ρ
$Q_\rho(s, a)$	Choose a once, then use ρ, in the local model
$Q_\rho^{\text{reset}}(\sigma, a)$	Isolated-subtree rollout cost from an operational snapshot
$[L_{s,a}, U_{s,a}]$	Valid interval for one specified rollout cost
$\mathcal{G}_\varepsilon(s), \mathcal{P}(s)$	Measured near-optimal actions; certified ordered pairs
$h_\mathcal{E}(s), e_\mathcal{E}(s)$	Proposal-restricted headroom; positive action-cost error

Table 7: Notation. The measured rollout cost, the best proposal cost, and the smallest possible proof are distinct objects.

Replace integer-variable bounds by their ceilings and floors, which preserves the feasible integer points. For each variable with a nonempty finite integer domain, let $d_i(s) = \lfloor u_i(s) \rfloor - \lceil \ell_i(s) \rceil + 1$. At a fractional LP coordinate, either child's nonempty integer domain is a strict subset of its parent's, so $\sum_{i \in I}(d_i(s) - 1)$ decreases by at least one along every nonterminal path, and empty domains are terminal. This gives finite depth and a finite binary tree. Continuous variables need not have finite domains. Assumption 1 additionally requires a finite attained optimum and a selected solution for every reachable nonterminal LP, with infeasible relaxations terminal.

A.2 PROOF OF THE PERFORMANCE-DIFFERENCE IDENTITY

Write $\Delta(s) = C_\pi(s) - C_\rho(s)$. At a terminal state both costs equal one, so $\Delta(s) = 0$. At an internal state, add and subtract $Q_\rho(s, \pi(s))$:

$$\Delta(s) = C_\pi(s) - Q_\rho(s, \pi(s)) + Q_\rho(s, \pi(s)) - C_\rho(s) \tag{16}$$

$$= A_\rho(s, \pi(s)) + \Delta(s^-(\pi(s))) + \Delta(s^+(\pi(s))). \tag{17}$$

Induction on the finite remaining depth proves Eq. (13); equivalently, Eq. (17) telescopes over all node occurrences in the learned tree. The sum runs over the learner's tree, not over the behavior tree or the training set.

For the argmin case of Corollary 2, its premise and inclusion of $\rho(s)$ in $\mathcal{E}(s)$ imply $Q_\rho(s, \pi(s)) = \min_{a \in \mathcal{E}(s)} Q_\rho(s, a) \le Q_\rho(s, \rho(s)) = C_\rho(s)$, so each summand in Eq. (13) is nonpositive. The interval condition gives the same result because $Q_\rho(s, \pi(s)) \le U_{s,\pi(s)} \le L_{s,\rho(s)} \le C_\rho(s)$. If at least one visited internal node has a strictly negative advantage, the inequality is strict.

A.3 APPROXIMATE ACTION VALUES AND PROPOSAL ERROR

Proposition 3 (Approximate proposal improvement)**.** *At every internal state visited by π, suppose $\rho(s) \in \mathcal{E}(s)$, $\pi(s) \in \mathcal{E}(s)$,*

$$\max_{a \in \mathcal{E}(s)} |\widehat{Q}(s, a) - Q_\rho(s, a)| \le \eta(s), \quad \widehat{Q}(s, \pi(s)) \le \min_{a \in \mathcal{E}(s)} \widehat{Q}(s, a) + \xi(s), \tag{18}$$

where $\eta(s), \xi(s) \ge 0$. Then

$$C_\pi(s_0) - C_\rho(s_0) \le \sum_{v \in \mathcal{I}(\mathcal{T}_\pi(s_0))} \left(2\eta(s_v) + \xi(s_v) - h_\mathcal{E}(s_v)\right). \tag{19}$$

Proof. For an exact minimizer $a_\mathcal{E}$ over the proposals,

$$\begin{aligned} Q_\rho(s, \pi(s)) &\le \widehat{Q}(s, \pi(s)) + \eta(s) \\ &\le \widehat{Q}(s, a_\mathcal{E}) + \xi(s) + \eta(s) \\ &\le m_\mathcal{E}(s) + \xi(s) + 2\eta(s). \end{aligned}$$

Subtract $C_\rho(s) = m_\mathcal{E}(s) + h_\mathcal{E}(s)$ and apply Theorem 1. □

The condition is a uniform value bound over the proposals at learner-visited states, which is stronger than a low average training loss or a small validation regret; a censored interval with infinite upper endpoint does not supply a finite η.

To separate proposal coverage from selection error, define

$$m_\mathcal{A}(s) = \min_{a \in \mathcal{A}(s)} Q_\rho(s, a), \quad d_\mathcal{E}(s) = m_\mathcal{E}(s) - m_\mathcal{A}(s), \quad h_\mathcal{A}(s) = C_\rho(s) - m_\mathcal{A}(s). \tag{20}$$

Then $d_\mathcal{E} \geq 0$ and $h_\mathcal{E} = h_\mathcal{A} - d_\mathcal{E}$. Substitution into Eq. (15) yields

$$C_\pi(s_0) - C_\rho(s_0) \leq \sum_{v \in \mathcal{I}(\mathcal{T}_\pi(s_0))} \left(e_\mathcal{E}(s_v) + d_\mathcal{E}(s_v) - h_\mathcal{A}(s_v)\right), \tag{21}$$

which separates the available improvement $h_\mathcal{A}$, the coverage loss $d_\mathcal{E}$ of a restricted proposal set, and the learner's selection error $e_\mathcal{E}$.

A.4 Censored observations and partial identification

Proposition 4 (Valid comparisons from interrupted rollouts)**.** *Fix the initial state, cutoff, continuation rule, random state if applicable, and counting convention. For each candidate action, if its rollout is interrupted after visiting n counted nodes and its continuation does not delete already counted work, its completed node count satisfies $Q \geq n$. Thus valid intervals with $U_a < L_b$ certify $Q_a < Q_b$. If $U_b \leq L_a$ for every $a \in \mathcal{E} \setminus \{b\}$, action b is a minimizer over the proposal set.*

Proof. The final visited-node sequence extends the observed prefix, so its length is at least n. Interval inclusion then gives $Q_a \leq U_a < L_b \leq Q_b$. For the last statement, $Q_b \leq U_b \leq L_a \leq Q_a$ for every alternative proposal. □

These bounds concern the specified continuation: a different policy may prove the same subtree with fewer nodes than the observed prefix, so they are not lower bounds on $\min_\pi C_\pi(s)$. The bound counts nodes, not wall time.

The distinction matters for targets. An exact cost of 50 and a censored interval $[100, \infty)$ certify that the exact action is better; an exact cost of 50 and a censored interval $[20, \infty)$ do not. The hard target in Eq. (6) is therefore the *best exactly evaluated* action, and it is proposal-optimal when the remaining intervals rule out all competitors.

A.5 Stability of targets on a fixed candidate support

Proposition 5 (Hard-label stability)**.** *Let a^* uniquely minimize the cost vector q on a fixed finite set X with $|X| \geq 2$, with gap $\Delta = \min_{a \neq a^*}(q_a - q_{a^*}) > 0$. If $\|\widetilde{q} - q\|_\infty \leq \eta$ and $\Delta > 2\eta$, then a^* also uniquely minimizes $\widetilde{q}$.*

Proof. For any $a \neq a^*$, $\widetilde{q}_a - \widetilde{q}_{a^*} \geq q_a - q_{a^*} - 2\eta \geq \Delta - 2\eta > 0$. □

With an exact tie, deterministic tie-breaking still defines a label, but small perturbations can change it.

For the soft target, set $d_a = \log(1 + q_a)$, $\widetilde{d}_a = \log(1 + \widetilde{q}_a)$, and assume nonnegative costs. If $\|\widetilde{d} - d\|_\infty \leq \delta$ on the same support, then

$$e^{-2\tau\delta} \leq \frac{\widetilde{y}(a)}{y(a)} \leq e^{2\tau\delta}, \tag{22}$$

since each unnormalized weight changes by a factor in $[e^{-\tau\delta}, e^{\tau\delta}]$ and the normalizing sum obeys the same bounds.

Set membership follows the same pattern. For $g_a(q) = q_a - (1+\varepsilon)\min_b q_b$, a perturbation bounded by η changes g_a by at most $(2 + \varepsilon)\eta$, so the measured set is unchanged whenever every action has

$|g_a(q)| > (2+\varepsilon)\eta$; actions near the threshold can enter or leave. All statements in this subsection assume a common candidate support and cost definition.

Finally, the set of optimal feasible MILP solutions and the set of minimum-cost branching actions are different objects. Multiple optima neither force a branching-cost tie nor make the argmin undefined, and distinct branch actions can have equal proof cost even when the MILP optimum is unique. The optimal-face probes in Appendix D concern the first set.

B Stateful solver dynamics

B.1 A performance difference for the complete search state

Let σ include all information affecting future computation: the open subproblems, local and global constraints, incumbent, statistics, LP state, random state, and policy memory when relevant. Consider decision epochs with a deterministic transition $F(\sigma, a)$ and incremental counted work $k(\sigma, a) \geq 0$. Assume termination and $V_\pi(\sigma) = V_\rho(\sigma) = 0$ at every terminal state. Let k account for all counted work. Then

$$Q_\rho^{\text{full}}(\sigma, a) = k(\sigma, a) + V_\rho(F(\sigma, a)), \qquad V_\rho(\sigma) = Q_\rho^{\text{full}}(\sigma, \rho(\sigma)). \tag{23}$$

For the complete trajectory $\sigma_0, \ldots, \sigma_T$ generated by π,

$$V_\pi(\sigma_0) - V_\rho(\sigma_0) = \sum_{t=0}^{T-1} \left(Q_\rho^{\text{full}}(\sigma_t, \pi(\sigma_t)) - V_\rho(\sigma_t) \right). \tag{24}$$

This follows by substituting $F(\sigma_t, \pi(\sigma_t)) = \sigma_{t+1}$ and telescoping the V_ρ terms. It accommodates shared pseudocosts, propagation, and an evolving incumbent if they are represented in σ. It requires a cost-to-go for the *entire remaining search*, however, which is far more expensive to measure than the subtree-local labels used here. The two-subtree example of Section 5 is an instance of this identity in which the second summand is large and positive. To apply Corollary 2 literally, one must therefore either enforce subtree-local dynamics or evaluate the full-state costs in Eq. (24); in practice we instead validate the learned policy in closed loop.

B.2 Replay validation

Counterfactual replay needs more than matching integer bound boxes: it should preserve the action's variable identity and split threshold, the transformed formulation and local constraints, and the history used by the continuation rule. A native-copy diagnostic compared live nodes with independent SCIP copies whose variable bounds were set to local bounds. With cuts disabled, 418 states had no false infeasibility and an LP-objective relative gap with median/p95 of $0/2 \cdot 10^{-16}$, but only 52% had candidate-set Jaccard at least 0.99. With cuts enabled, 302 states included five false infeasibilities, and the median/p95 relative gap was $3.3 \cdot 10^{-3}/0.21$.

Objective-level agreement therefore does not imply identical LP solutions, fractional candidates, or continuation trees. This is why the controlled pipeline restores the full LP and pseudocost history rather than only bounds (Appendix C).

C Implementation and experimental protocols

C.1 Controlled data collection

Offline collection procedure. For each retained state, restore its subproblem, bounds, LP history, split identities, and cutoff; save a common snapshot and form the proposal set. Restore that snapshot independently for each action, apply the recorded split, and follow the continuation policy to completion or budget exhaustion. Record the interval in Eq. (4), the work, the status, and the action identity, and store all observations together with the policy features. The custom environment solves LP relaxations with HiGHS, traverses depth-first, uses a fixed optimal cutoff, and applies no in-tree cutting planes, conflict analysis, or primal heuristics. The reliability-style behavior rule uses pseudocosts to prioritize candidates and probes up to 15 of them; it differs from SCIP's `fullstrong` and `relpscost`.

For the shared-state dataset, the proposal cap is eight actions, including the behavior action. The alternative mixture uses approximately 50% SB-based priorities, 25% priorities based on the stored optimal solution, 15% random proposals, and 10% diversity proposals; the optimal solution serves only to propose candidates. Per instance, the counterfactual budget is 20 times the behavior proof's LP solves. Up to 200 states are retained, combining large-subtree priorities with random sampling. Individual counterfactual rollouts are capped at 1.5 times the reference subtree size, and visited-node lower bounds are recorded separately from these budget settings. Corollary 2 requires the continuation action $\rho(s)$ among the proposals at learner-visited states. The collection guarantees inclusion of the behavior action $\mu(s)$, which satisfies this when $\rho = \mu$; coverage of learner-visited states is addressed through closed-loop evaluation and, optionally, relabeling (Section 4).

A pseudocost snapshot/restore step prevents one candidate trial from changing another's initial history. A faithful replay audit recovered the exact stored bounds, including signed zero, and the per-state LP sequence; costs, completion flags, and winners then matched on all 64 audited set-cover and 36 audited CA states, including large-subtree states, and reversing candidate order had no effect. Different preceding LP histories changed some costs and set-cover winners, as expected for a stateful solver: the labels are reproducible under the recorded protocol and are costs relative to the recorded history.

C.2 Targets, weights, and optimization

The principal hyperparameters are $\tau = 3$, $\varepsilon = 0.05$, $\lambda = 0.5$, and impact exponent $\alpha = 0.5$. For reference cost $S(s)$ and best exact cost $q_{\min}(s)$, the unnormalized D weight is

$$\widetilde{w}(s) = \min\left\{ w_{\max},\, \log(1 + S(s)) \left(\frac{[S(s) - q_{\min}(s)]_+}{S(s) + 1} \right)^{\alpha} \right\}. \tag{25}$$

Weights are normalized to mean one in a batch, and a zero-total-weight batch contributes zero D loss. The positive part also covers states where reference and proposal coverage differ. We use $w_{\max} = 4$. Non-admissible candidates are masked with $-\infty$ logits before the softmax in Eqs. (9)–(10), states without an exact evaluation are excluded from the D loss, and a batch whose weights sum to zero is skipped.

TreeGate-lite uses hidden width 64 and 15 candidate plus 10 tree features. The shared-state comparison trains for 60 epochs with Adam, learning rate 10^{-3}, and batch size 256, saving a checkpoint every five epochs. The historical CA runs selected checkpoints by validation action regret (Appendix D); every table in the main text uses closed-loop selection, which retains the checkpoint with the smallest shifted geometric-mean node count on the validation instances of Table 8 (manifest `select-cl-v3`). The GNN follows Gasse et al. (2019): two bipartite half-convolutions with 64 hidden units, prenorm feature scaling from training statistics, Adam at 10^{-3} with batches of 32 states, 100 epochs, and the same checkpoint rule. BR uses its published ranking loss with the TreeGate-lite encoder on the output of our collector, with the budget and fixed-cutoff intervention stated in Table 4, trained for 60 epochs and selected by the same rule.

The soft target excludes censored values from its target distribution, whereas D uses their certified pairwise comparisons. The policy normalizer in Eq. (5) runs over admissible candidates, while the soft-target normalizer runs over exact evaluations.

C.3 Datasets and repeated measurements

MKP S1 contains 300 distinct test instances and the target comparison uses 1,500 model-instance pairs; CA and set cover each contain 240 distinct test instances and 1,200 pairs; MILP-S uses 120 pairs over eight test instances. Seeds add repetition, not new instances, and the bootstrap below respects this hierarchy.

For paired outcomes N_i and N_i^{ref}, a geometric ratio with shift $\kappa \geq 0$ is

$$R_\kappa = \exp\left[\frac{1}{n} \sum_{i=1}^{n} \log \frac{N_i + \kappa}{N_i^{\mathrm{ref}} + \kappa} \right]. \tag{26}$$

A shifted geometric mean for a single method is $\mathrm{SGM}_\kappa(N) = \exp(n^{-1} \sum_i \log(N_i + \kappa)) - \kappa$. When both quantities are computed over the same runs with equal weights, $R_\kappa = (\mathrm{SGM}_\kappa(N) +$

Dataset	Dimensions	Train / validation / test	Repetition
MKP S1	32 items, 6 constraints	200 / 50 / 100	3 splits, 5 model seeds
MKP S2	44 items, 8 constraints	150 / 40 / 80	5 model seeds
MKP S3	56 items, 8 constraints	150 / 40 / 80	5 model seeds
CA	200 items, 800 bids	150 / 40 / 80	3 splits, 5 model seeds
Set cover	70 rows, 140 columns	150 / 40 / 80	3 replication splits, 5 seeds
MILP-S	heterogeneous MILPs	19 train / 8 test instances	3 model, 5 solver seeds

Table 8: Dataset configurations. Set-cover counts refer to the replication, not the earlier 30-test-instance pilot.

$\kappa)/(\mathrm{SGM}_\kappa(N^{\mathrm{ref}}) + \kappa)$; the ratio of the shift-subtracted means themselves differs from R_κ, and a ratio computed over a different set or weighting of runs differs as well, so we report paired ratios directly and list aggregate summaries separately. We use $\kappa = 0$ for node counts, so node geometric means are unshifted and Eq. (26) is the plain geometric mean of paired ratios, and $\kappa = 1$ s for every time ratio and shifted geometric-mean time in Tables 6 and 14. Intervals use a hierarchical paired bootstrap over generator split, model seed, and instance, reusing resampled indices across methods. Timed-out runs enter aggregate means with their observed node counts, which are not completed proof sizes; paired ratios use only run pairs that both policies finish within the limit (Appendix F).

D Additional empirical results

D.1 Controls for support, concentration, and cost assignment

All controls in Table 2 retain the reported state sets, candidate mask, model seeds, and checkpoint protocol of their matched target comparison. USB uses SB-prioritized actions without the stored optimum or counterfactual evaluation; U is uniform over the completed proposal set; RND samples a completed action as a one-hot target; CSH randomly permutes the exact costs among actions within a state before applying Eq. (7). RND preserves hard-target sharpness and CSH preserves soft-target entropy. On set cover, U trained on a matched proposal mixture that omits the $x^\star$-aware component (its 25% share redistributed proportionally to the remaining components) separates optimum-guided proposals from random and diverse ones. Removing the $x^\star$-aware proposals worsens U by 1.212 $[1.061, 1.384]$, while the $x^\star$-free mixture still improves over USB by 0.531 $[0.372, 0.758]$. Optimum-guided proposals thus account for part of the mixture's advantage and random and diverse proposals for the larger part, consistent with the frequent alternative optima on this family (Table 10). $x^\star$ is used only offline, on solved training instances.

For the exact-censored comparison, let $\mathcal{P}_{\mathrm{EE}}$ contain completed-completed pairs and $\mathcal{P}_{\mathrm{EC}}$ contain completed-censored pairs certified by Eq. (8). Both arms use $K_s = \max\{1, |\mathcal{P}_{\mathrm{EE}}(s)| + |\mathcal{P}_{\mathrm{EC}}(s)|\}$, the same set loss and impact weights. The control uses $\lambda K_s^{-1} \sum_{\mathcal{P}_{\mathrm{EE}}} \ell_{ab}$; the treatment adds $\lambda K_s^{-1} \sum_{\mathcal{P}_{\mathrm{EC}}} \ell_{ab}$, with $\ell_{ab} = \mathrm{softplus}(f_\theta(s,b) - f_\theta(s,a))$. Thus exact-exact gradients keep their normalization. At a cap of S nodes, an interrupted rollout recorded with lower bound S does not

Direct contrast, GNN	CA	Set cover
With EC / without EC, cap $1.0S$	1.010 $[0.999, 1.031]$	0.954 $[0.842, 1.110]$
With EC / without EC, cap $1.5S$	0.968 $[0.909, 1.013]$	0.987 $[0.942, 1.011]$
Exact-only cap $1.0S$ / cap $1.5S$	0.955 $[0.900, 1.002]$	1.239 $[1.105, 1.381]$

Table 9: Loss-matched censored-pair ablation and budget sensitivity. Closed-loop performance is unchanged by the EC pairs; the rollout cap matters on set cover.

certify that an exact reference of cost S is better under Eq. (4). We do not add a margin, since an arbitrary margin would not be a certified comparison.

Metric	MKP	Set cover
Winner agreement, reversed vs. unreversed candidate order (same LP history)	–	1.00
Winner agreement with altered preceding LP history	–	0.906
Behavior action in measured $\mathcal{G}_{5\%}$	72.1%	24.2%
Normalized measured headroom	0.055	0.154
Mean exact evaluations / state	3.31	7.90
Soft entropy / log support size	0.953	0.972
Alternative optimum in $x^\star$-excluding child	0 / 493	406 / 568

Table 10: Label diagnostics. The first two rows come from the faithful replay audit, which isolates candidate order from LP-history changes on set cover; the earlier 0.58–0.67 winner-agreement figure mixed both effects.

D.2 Label and optimal-solution diagnostics

The optimal-solution probes use eight instances per family, random objectives constrained to the original optimal objective level, and per-child feasibility checks. The probe is a sample: finding an alternative optimum establishes multiplicity, while not finding one does not establish uniqueness.

D.3 Sparse oracle switching

An exploratory test in the custom fixed-cutoff environment annotates the eight largest recorded SB subtrees. A subsequent rollout switches to the measured action at matching bound boxes and otherwise follows SB. Matching bound boxes does not reproduce pseudocost history, and an early switch

Family	Proposals	Instances	Oracle-switch / SB	States with local improvement
CA	16	25	$0.982\ [0.711, 1.276]$	145 / 200
CA	32	12	0.998	80 / 96
MKP S3	16	16	$0.953\ [0.904, 0.995]$	95 / 128

Table 11: Sparse oracle switching at eight annotated states per instance. MKP S3 shows a whole-tree gain; the CA intervals are wide.

can make later annotated states unreachable, so this experiment approximates rather than implements the greedy policy of Corollary 2, and its whole-tree effect reflects only eight decisions per instance rather than a full policy.

D.4 Selection, low-data, and transfer details

For the historical CA runs, closed-loop selection chooses checkpoints with ratios around 0.68–0.69 relative to BC, whereas open-loop regret selected early checkpoints at 3.37/5.34. On the three fresh CA splits, a burn-in variant gives Bsel/BC $= 0.554\ [0.504, 0.608]$, close to B/BC $= 0.549\ [0.503, 0.600]$.

In the nested MKP data ablation, B/BC and C/BC are 0.987/0.970 at 5% of the training instances and 0.984/0.973 at 10%. These point estimates favor soft targets at low data volume, while at full data the two are indistinguishable (C/B $= 0.997\ [0.986, 1.009]$, Section 6).

CFL screening at 50 customers and 25 facilities found 5.7% measured headroom, with 9.5% of states meeting the action-gap criterion; we therefore concentrated the training experiments on the three families with larger measured headroom.

On the harder CA transfer cohort (300 items, 1200 bids; 40 instances, five model seeds), the frozen GNN checkpoints give B/BC $= 0.671\ [0.591, 0.785]$, C/BC $= 0.744\ [0.650, 0.867]$, and B/SB $= 1.482\ [1.406, 1.560]$; at training scale B/SB is $1.056\ [0.977, 1.149]$.

E SCIP EXPERIMENTS AND COMPUTATIONAL COST

E.1 MKP SCALE COMPARISON

The scale-specific results appear in Table 5. Both larger scales show larger gains against BC than S1; each scale is trained separately.

E.2 PROOF-ONLY EVALUATION

This evaluation uses the SCIP-presolved transformed problem, an injected optimal incumbent, and disabled cuts, primal heuristics, conflicts, restarts, and symmetry handling; propagation remains active. Forty test instances per listed family/scale are evaluated with the selected seed-0 policies. Strong-branching probes perform useful work outside the main search-node count. At MKP S3,

Family	`relpscost`	`fullstrong`	BC	Hard	Soft / set
MKP S1	30.6	20.9	75.9	—	C: 65.4
MKP S2	101.0	49.9	233.8	204.7	C: 185.2
MKP S3	189.1	81.3	473.8	344.2	C: 337.0
CA	15.2	6.2	115.2	105.9	C: 91.9
Set cover (GNN)	2.9	2.7	9.1	—	D: 8.6

Table 12: Geometric-mean SCIP node counts in the proof-only configuration. These are a different metric and cohort from the custom-environment comparisons.

mean LP iterations are 1,025 for `relpscost` and 1,247 for C, a much smaller relative difference than their node counts, so node counts overstate the work gap between SB-style rules and learned policies.

E.3 PRACTICAL EVALUATION WITHOUT THE OPTIMAL CUTOFF

The solver uses the current incumbent, enables cuts, primal heuristics, and conflicts, and disables presolve and restarts. No test optimal solution is supplied. The matched idle-machine CA cohort of 80 instances reports the times and paired intervals in Table 6; all methods solve within 300 seconds. All runs use SCIP 9.1.0 with SoPlex 7.1.0, single-threaded, one run per core on an otherwise idle AMD EPYC 7452 (2.35 GHz) node; policy inference uses PyTorch 2.3 on CPU, and the time shift is $\kappa = 1$ s. Table 13 explains how a learned policy can be faster despite a larger search tree.

CA branching rule	GM nodes	Mean LP iterations	Inference (s / instance)
`relpscost`	17.9	7,140	–
BC	117.6	4,380	0.04
B	111.2	4,300	0.04
USB (matched)	109.6	4,230	0.04
C	98.8	3,940	0.04

Table 13: Work counters for the idle-machine CA batch of Table 6. Node counts are geometric means; LP iterations are arithmetic means per instance and include strong-branching probes.

`relpscost` expands less than a fifth as many nodes as C but performs 81% more LP iterations, because its strong-branching probes dominate its LP work. Among the learned policies, node counts, LP iterations, and times move together: C expands 16% fewer nodes than BC and performs 10% fewer LP iterations for a 7% time saving, and the TreeGate-lite inference overhead of about 0.04 s per instance, roughly 4% of C's solving time, is identical across the learned policies.

An earlier batch, run under a different machine condition, is shown below for context and is not pooled with Table 6. All reported time ratios come from the idle-machine batch of Table 6, whose search work matches the earlier batch.

Family	Method	Shifted-GM time (s)	GM nodes	Mean PD integral
CA	`relpscost`	2.27	17.4	9.3
CA	BC	1.77	119.3	8.7
CA	C	1.53	99.6	8.4
MKP S2	`relpscost`	0.28	70.1	1.1
MKP S2	BC	0.54	171.5	1.6
MKP S2	C	0.44	135.8	1.5
MKP S3	`relpscost`	0.48	126.0	1.5
MKP S3	BC	0.89	278.4	2.2
MKP S3	C	0.70	207.9	1.8

Table 14: Initial practical batch, run under a different machine condition from Table 6; shown for context only.

E.4 Offline cost

Label generation cost 0.7, 3.1, and 4.3 CPU-hours for MKP S1–S3, 24.6 for CA, and 7.4 for set cover, excluding preliminary optimal solves, training, and model selection. For CA, adding the optimal solves (1.9 CPU-hours), training of the three targets over five seeds (4.4), and closed-loop selection (5.8) gives a complete offline bill of 36.7 CPU-hours. In the idle-machine batch, C saves a paired arithmetic mean of 0.21 ± 0.07 s per instance over BC and 0.67 ± 0.12 s over `relpscost` (95% intervals).

F MILP-S sequence-model experiment

The collection contains 1,330 episodes from 19 training instances, 14 policy-pair configurations, and five SCIP seeds. The merged labeling record covers 1,215 episodes, with 1,136 successful episodes and 2,334 labeled states. The four target modes share the same labeled HDF5 dataset. Training uses the grouped Mamba policy with a TreeGate-derived representation, sequence length 50, and three model seeds. Evaluation uses eight test instances and five solver seeds per model seed, with a time limit of 3,600 s per run.

On the training side, completed counterfactual measurements show a geometric-mean local subtree decrease from 23.5 to 20.0 nodes and a first-branch continuation decrease from 408 to 330. These are oracle comparisons on labeled training states; the deployed-policy results follow.

MILP-S target	Fair-node GM	Ratio to BC	95% interval	Solved / 120
BC	2966	–	–	117
Hard	3450	1.155	[0.955, 1.501]	115
Soft	2818	0.949	[0.834, 1.075]	118
Routed	3085	1.038	[0.903, 1.217]	116

Table 15: Eight distinct MILP-S test instances with five solver seeds and three model seeds. Fair nodes count branch-and-bound nodes plus strong-branching probes, each probe converted to node equivalents at the run's mean LP iterations per node, so that rules with different probing effort are compared on equal LP work. The fair-node column pools all 120 runs, with timed-out runs at their observed counts; the ratio column pairs the runs that both the target and BC finish within the limit, which is why it differs slightly from the ratio of the pooled means.

The hard policy's five timeouts are concentrated on `neos18`; we report the full instance set. With eight heterogeneous instances the intervals are wide; the soft target's point estimate is 5% below BC, and the experiment serves as an exploratory test of the labels with a sequence-model encoder.

Reproducibility statement

Appendix A contains full proofs, Appendix C gives collection and evaluation protocols, and Appendix D.1 gives the control protocol and the censored-pair ablation. An anonymized archive with

the collection and training code, the labeled datasets, the run manifests, and the evaluation ledger that regenerates every table is included as supplementary material; it will be released publicly on acceptance.

STATEMENT ON THE USE OF GENERATIVE AI

Generative AI assisted with literature retrieval, experimental planning and interpretation, mathematical formulation and proof drafting, manuscript writing and editing, and figure preparation. Numerical results originate from research experiments rather than generated observations. The authors re-derived every proof independently of the drafted versions, regenerated every table from the run ledger with the released scripts, and checked each citation against its source. The authors take responsibility for the final paper.