AUTOTW-ASP: AUTOMATIC LOW-TREEWIDTH REWRITE SYNTHESIS AND UNCERTAINTY-AWARE BACKEND ROUTING FOR EXACT NEUROSYMBOLIC ASP TRAINING

Anonymous authors
Paper under review

ABSTRACT

Recent neurosymbolic Answer Set Programming (ASP) pipelines frequently rely on manual encoding redesign to exploit treewidth-sensitive exact inference backends, creating a reproducibility and scalability bottleneck. We study an integrated system, AutoTW-ASP, that jointly performs semantics-preserving rewrite selection and uncertainty-aware backend routing between exact enumeration and compilation-based inference. The method combines motif-level rewrite guards, crossover-margin prediction, and abstention-to-enumeration under uncertainty. We formalize rewrite acceptance, routing regret, and constrained end-to-end optimization, and we connect these formal components to an empirical protocol spanning symbolic and hybrid benchmark families. Experiments show that rewrite mismatch rates remain below a predefined projection criterion (maximum observed mismatch rate 0.00375), while non-zero exactness violations remain in accepted-rewrite audits. Routing maintains strong severe-shift coverage (0.9346) and low mean regret in mild/moderate regimes, but severe-shift exactness violations and elevated calibration error persist. Integrated training achieves median 1.140× speedup, which is below a 1.20× acceptance target, and therefore does not fully support the strongest end-to-end claim under strict exactness constraints. These outcomes establish a bounded positive result: automated rewrite-plus-routing can deliver measurable efficiency gains and informative uncertainty structure, but claim-level guarantees remain conditional on tighter guard policies and calibration in hard-shift regimes.

1 INTRODUCTION

Neurosymbolic ASP systems aim to combine perception-driven modules with exact symbolic reasoning so that supervision and downstream decisions remain logically faithful to declarative constraints. That objective has been persistent across neurosymbolic ASP foundations, probabilistic ASP semantics, and neural-probabilistic extensions, where exactness is treated as a first-class requirement rather than a post-hoc correction (Borroto et al., 2025; Geh et al., 2024; 2023; Yang et al., 2023; Shakarian et al., 2023; Skryagin et al., 2022). In practice, however, exactness does not imply uniform runtime: per-instance inference cost can vary sharply with structural properties of the grounded program, reuse opportunities, and backend-specific overhead profiles (Azzolini & Hecher, 2025; Azzolini & Riguzzi, 2024a; Sundermann et al., 2024b;a; Amarilli et al., 2023; Derkinderen et al., 2023; Dubray et al., 2023). A recurring operational response has been manual encoding redesign, often by domain experts, to make decomposition and compilation tractable.

The scientific problem in this paper is not whether exact backends are useful, but whether that expert-dependent redesign loop can be automated without losing symbolic guarantees. We target a dual-backend setting in which each instance can be solved either through stable-model-oriented enumeration or compilation-based exact counting/inference. Prior literature characterizes both regimes, yet does not establish a closed integrated method that simultaneously (i) rewrites grounded programs under explicit semantic guards, (ii) routes each instance based on uncertainty-calibrated cost margins, and (iii) evaluates end-to-end training behavior under exactness and accuracy constraints (Azzolini et al., 2025a;b; 2024a;c;b;d; Azzolini & Riguzzi, 2023a; Azzolini et al., 2023a; Azzolini, 2023).

Our study is hypothesis-driven and hybrid in emphasis: formal derivation is used to define admissible operations and expected regret structure, while empirical validation determines where those guarantees are practically realized or violated. The resulting manuscript therefore treats theorem-level obligations and benchmark-level evidence as complementary, not interchangeable. In particular, we report both achieved outcomes and explicit failure modes because unsupported edges are central to the scientific interpretation of this line of work.

Contributions.

Table 1: Core notation used in the method and analysis. Symbols are defined here to keep equations in section 4 and section 5 unambiguous across rewrite, routing, and training objectives.

Symbol	Definition
P_x, P_x^ϕ	Original and rewritten grounded programs for instance x.
Q	Projection signature used for supervision/evaluation semantics.
$w(P)$	Structural proxy (treewidth-like indicator) used for rewrite screening.
$C_e(x), C_c(x)$	Exact runtime of enumeration and compilation backends on rewritten instance.
$m(x), \hat{m}(x)$	True and predicted backend cost margins $C_e(x) - C_c(x)$.
$u(x)$	Routing uncertainty score used for abstention control.
$r(x)$	Per-instance regret relative to the better backend.
Θ	Integrated control parameters (rewrite, routing, update cadence).
$T_{\text{train}}(f;\Theta)$	Training wall-clock cost on benchmark family f.
$\Delta Acc(f;\Theta)$	Accuracy drop vs. baseline on family f.
$Err_{\text{exact}}(f;\Theta)$	Exactness-violation indicator/count on family f.

- We define an integrated rewrite-routing formulation for exact neurosymbolic ASP training, with explicit objective, decision variables, feasible set, and optimality criterion that align formal assumptions with measurable protocol contracts.
- We provide theorem-backed analysis for compositional rewrite soundness, abstention-aware routing regret structure, and constrained end-to-end optimization, and we expose the assumptions required for each statement.
- We execute a four-part validation suite (rewrite soundness, router calibration under shift, integrated training outcomes, and cache-boundary diagnostics) with uncertainty-aware summaries and negative-result audits.
- We show a bounded empirical outcome: several local targets are met, but strict global claim closure is prevented by non-zero exactness violations and unmet integrated-speedup thresholds.

The remainder of this paper proceeds as follows. Section 2 formalizes the problem setting and notation. Section 3 synthesizes prior work and identifies the novelty boundary. Section 4 details the integrated method and algorithmic workflow. Section 5 presents theorem-level analysis. Section 6 and section 7 provide the empirical protocol and outcomes. Section 8 and section 9 analyze robustness boundaries and unresolved gaps. Section 10 concludes.

2 Problem Setting and Notation

2.1 Formal Objects and Assumptions

For each task instance x, let P_x be a grounded ASP/PASP program under fixed stable-model semantics. Let Q denote the projection signature used for supervision or downstream symbolic constraints. The system can apply a rewrite policy $\phi \in \Phi$ to obtain P_x^ϕ, and then select a backend decision $b_x \in \mathcal{B} = \{\texttt{enum}, \texttt{comp}\}$. Enumeration refers to exact stable-model style solving, whereas compilation refers to exact knowledge-compilation-based inference. The central semantic constraint is projection-preserving equivalence:

$$AS(P_x) \restriction_Q = AS(P_x^\phi) \restriction_Q, \tag{1}$$

where $AS(\cdot)$ denotes answer sets under the declared semantics and $\restriction_Q$ denotes projection to signature Q.

This condition is aligned with prior PASP residual-program and decomposition analyses, where structural transformations can reduce computational burden yet still require semantic equivalence under the query projection (Azzolini & Hecher, 2025; Azzolini & Riguzzi, 2024a; 2023a; Azzolini et al., 2023b; Azzolini & Riguzzi, 2023b). We adopt that convention at first introduction because the equivalence target is not manuscript-specific background; it is a nontrivial inherited formal requirement.

2.2 Objective, Decision Variables, and Feasible Set

The integrated decision variable is $\Theta = (\phi, \pi, \kappa)$, where ϕ parameterizes rewrite acceptance, π parameterizes routing, and κ controls update cadence and scheduling in training loops. The constrained optimization problem is:

$$\Theta^\star \in \arg\min_{\Theta \in \mathcal{F}_\Theta} \mathbb{E}_{f\sim\mathcal{F}}\big[T_{\text{train}}(f;\Theta)\big] \quad \text{s.t.} \quad \Delta Acc(f;\Theta) \le \epsilon,\ Err_{\text{exact}}(f;\Theta) = 0\ \forall f \in \mathcal{F}. \tag{2}$$

The feasible set $\mathcal{F}_\Theta$ contains only policies that satisfy explicit rewrite-side conditions and exact fallback availability assumptions. Our optimality criterion is runtime minimization under hard exactness and bounded accuracy degradation constraints; Equation 2 is therefore the core criterion for all claim-level interpretation.

This framing is motivated by the literature tension between expressive probabilistic ASP semantics and tractability envelopes: broad semantics improve modeling flexibility, but runtime can become unstable without structure-aware controls (Geh et al., 2024; Azzolini & Riguzzi, 2024b;c; Geh et al., 2023; Riguzzi, 2023). We make this trade-off explicit by placing exactness in hard constraints and placing runtime as the optimized objective.

3 Related Work and Novelty Boundary

3.1 Neurosymbolic ASP and PASP Foundations

Neurosymbolic ASP frameworks have established a practical pattern: neural modules provide perceptions or probabilities, while ASP/PASP layers enforce symbolic consistency and exact declarative semantics (Yang et al., 2023; Shakarian et al., 2023; Skryagin et al., 2022; maintainers, 2026a;b;c;d). Recent extensions expand semantic breadth to richer probabilistic choices and mixed discrete-continuous settings (Geh et al., 2024; Azzolini & Riguzzi, 2024b;c; Skryagin et al., 2023a; Smet et al., 2023; Skryagin et al., 2023b; Geh et al., 2023). These systems demonstrate strong representational power, but they do not by themselves resolve backend-selection instability when workload structure shifts.

PASP inference and decision-theoretic lines supply exact algorithms for MPE, MAP-like objectives, statistical statements, and credal-semantics querying (Azzolini et al., 2025a; Azzolini & Hecher, 2025; Azzolini et al., 2025b; Azzolini & Riguzzi, 2024a; Azzolini et al., 2024a; Kiesel et al., 2023a; Azzolini & Riguzzi, 2023a; Kiesel et al., 2023b; Azzolini et al., 2023a;b; Azzolini & Riguzzi, 2023b). Their strength is formal rigor over semantics and inference correctness. Their limitation, relative to our objective, is that most methods evaluate fixed solving pipelines and do not jointly optimize rewrite acceptance and uncertainty-aware backend routing within a training loop.

3.2 Compilation, Structure, and Reuse

Knowledge compilation and weighted model counting lines show that decomposition quality, circuit form, and reuse opportunities strongly influence exact-inference cost (Sundermann et al., 2024b;a; Amarilli et al., 2023; Mohle, 2023; Derkinderen et al., 2023; Dubray et al., 2023; Kiesel et al., 2022; Ganian et al., 2022; Malhotra & Serafini, 2022). This evidence motivates our structural proxy features and reuse-aware routing decisions. At the same time, these papers also indicate regime dependence: compilation wins when amortization is favorable, but overhead can dominate in low-reuse and high-variability regimes. This contradiction motivates our abstention strategy rather than a universal preference for either backend.

3.3 Learning-Time Coupling and Open Gap

Symbolic parameter-learning studies in PASP demonstrate that inference decisions and optimization dynamics are tightly coupled (Azzolini et al., 2024c;b;d; Azzolini, 2023). Related neural-probabilistic methods and stochastic logic systems further emphasize that approximate or unstable inference pathways can alter learning behavior (Winters et al., 2022; Manhaeve et al., 2021). Taken together, this literature suggests a gap: there is no broadly validated integrated method that automates rewrite synthesis and routing while retaining strict exactness guarantees in training-time evaluation across symbolic and hybrid benchmarks.

Our novelty boundary is therefore specific and defensible: we do not claim a new semantics for ASP/PASP, nor universal backend dominance, nor universal complexity certificates from a single structural proxy. We claim an integrated optimization-and-validation framework that formalizes the rewrite-routing coupling, makes assumptions auditable, and reports both supported and unsupported claim regions.

4 Method: Integrated Rewrite, Routing, and Training Control

4.1 Rewrite Acceptance as a Guarded Optimization Layer

We define rewrite acceptance using two gates: projected semantic equivalence and structural improvement. Let $\mathcal{A}(x, \phi)$ denote acceptance:

$$\mathcal{A}(x,\phi) = \mathbf{1}\left[AS(P_x) \restriction_Q = AS(P_x^\phi) \restriction_Q\right] \cdot \mathbf{1}\left[w(P_x^\phi) \leq w(P_x) - \delta_w\right]. \tag{3}$$

Algorithm 1 Integrated Rewrite-Routing Controller for Exact Training

Input: training set $\mathcal{D}$, validation set $\mathcal{D}_{\text{valid}}$, rewrite library Φ, router π, thresholds (τ_m, τ_u)
Initialize baseline-exact policy Θ_{base} and current policy $\Theta \leftarrow \Theta_{base}$
for each epoch $e = 1, \ldots, E$ **do**
  for each batch $B \subset \mathcal{D}$ **do**
    for each instance $x \in B$ **do**
      Ground program P_x, propose rewrites, and compute acceptance $\mathcal{A}(x, \phi)$ using equation 3
      If no rewrite is accepted, keep original instance representation
      Estimate $\hat{m}(x)$ and uncertainty $u(x)$
      Route with rule equation 5; if uncertain, abstain to enumeration
      Execute exact inference and log runtime, exactness checks, and supervision consistency
    end for
    Update model parameters using exact symbolic outputs
  end for
  Recalibrate routing thresholds on $\mathcal{D}_{\text{valid}}$ with regret and calibration diagnostics
  Tighten rewrite/routing guards if exactness audits detect violations
end for
Return best feasible Θ under constrained objective equation 2

The rewrite-selection objective is

$$\max_{\phi \in \Phi} \mathbb{E}_{x \sim \mathcal{D}} \left[\mathcal{A}(x, \phi)\, \Delta T(x, \phi) \right] \quad \text{s.t.} \quad \forall x \in \mathcal{D} : \; AS(P_x) \restriction_Q = AS(P_x^{\phi}) \restriction_Q . \tag{4}$$

In equation 4, ΔT is runtime reduction under exact inference, so accepted rewrites are useful only when they preserve semantics and lower compute cost.

The design choice in equation 3 follows decomposition and residual-program evidence: local syntactic reductions are valuable only when they preserve query-relevant semantics under the chosen projection (Azzolini & Hecher, 2025; Azzolini & Riguzzi, 2024a; Sundermann et al., 2024a; Amarilli et al., 2023). We therefore treat rewrite generation as a proposal step and rewrite acceptance as a proof-gated decision step.

4.2 Uncertainty-Aware Backend Routing

Given an accepted rewrite, routing must choose between exact backends. Let

$$b(x) = \begin{cases} \text{comp}, & \hat{m}(x) > \tau_m \;\wedge\; u(x) \leq \tau_u, \\ \text{enum}, & \text{otherwise}, \end{cases} \tag{5}$$

where $\hat{m}(x)$ predicts margin $m(x) = C_e(x) - C_c(x)$, and $u(x)$ measures predictive uncertainty. Regret is

$$r(x) = C_{b(x)}(x) - \min\{C_e(x), C_c(x)\}. \tag{6}$$

The abstention mechanism in equation 5 makes routing conservative near uncertain crossover regions. This choice is motivated by the contradiction between per-instance predictability and amortization-driven wins observed across PASP and compilation literatures (Azzolini et al., 2025b; Sundermann et al., 2024b;a; Derkinderen et al., 2023; Dubray et al., 2023). Without abstention, routing errors near the boundary can dominate average gains.

4.3 Integrated Training Controller

The full loop jointly updates rewrite proposals, routing calibration, and training parameters. Algorithm 1 summarizes the execution logic.

Two methodological choices are central. First, rewrite and routing controls are audited at the same granularity as training updates, because stale guard states can create delayed exactness failures. Second, fallback-to-enumeration remains always available, so uncertainty never forces an approximate symbolic path. This preserves interpretability of failures: when violations occur, they can be attributed to guard design rather than hidden approximation.

5 Formal Analysis

5.1 Rewrite Compositionality

Definition 5.1 (Accepted Rewrite Sequence)**.** *Let $P^{(0)} = P$ and $P^{(k+1)} = r_k(P^{(k)})$ for $k = 0, \ldots, K-1$, where each rewrite r_k belongs to the supported motif library and satisfies the side conditions required by equation 3. The sequence is* accepted *if every step passes both semantic and structural gates.*

Lemma 5.2 (Local Projection Equivalence)**.** *If a rewrite template r satisfies its side conditions on program P, then*

$$AS(P) \restriction_Q = AS(r(P)) \restriction_Q . \tag{7}$$

Proof. By construction, supported templates are admitted only when their motif-level side conditions imply projection-preserving transformation under stable-model semantics. The acceptance test enforces this implication before execution. Therefore every accepted local rewrite satisfies equation 7. □

Theorem 5.3 (Compositional Rewrite Soundness)**.** *For any accepted rewrite sequence as in Definition 5.1,*

$$AS(P^{(K)}) \restriction_Q = AS(P^{(0)}) \restriction_Q . \tag{8}$$

Proof. We proceed by induction on K. For $K = 0$, equation 8 is immediate. Assume the statement holds for $K = t$. Consider $K = t + 1$: by Lemma 5.2, $AS(P^{(t+1)}) \restriction_Q = AS(P^{(t)}) \restriction_Q$. By the induction hypothesis, $AS(P^{(t)}) \restriction_Q = AS(P^{(0)}) \restriction_Q$. Transitivity yields $AS(P^{(t+1)}) \restriction_Q = AS(P^{(0)}) \restriction_Q$, completing the proof. □

5.2 Routing Regret with Abstention

Assume bounded confident-region margin error $|\hat{m}(x) - m(x)| \le \eta$ whenever $u(x) \le \tau_u$, and bounded backend gap $|m(x)| \le B$. Under these assumptions we derive a regret envelope consistent with the uncertainty-aware route rule.

Theorem 5.4 (Expected Regret Decomposition)**.** *Let $\Gamma = \{x : u(x) \le \tau_u\}$ denote the confident region and Γ^c the abstention region. Then*

$$\mathbb{E}[r(x)] \le \eta \Pr(x \in \Gamma) + B \Pr(x \in \Gamma^c). \tag{9}$$

Proof. On Γ, the policy uses the predicted margin and the margin error is bounded by η; therefore the regret contribution is at most η. On Γ^c, abstention sends the instance to enumeration, and regret is at most the bounded backend gap B. Taking expectation by partitioning over Γ and Γ^c gives equation 9. The decomposition is exact under the stated bounds. □

5.3 Constrained End-to-End Feasible Dominance

Theorem 5.5 (Feasible Dominance Under Hard Exactness)**.** *Assume baseline policy $\Theta_{base} \in \mathcal{F}_\Theta$. If $\Theta^\star$ solves equation 2, then*

$$\mathbb{E}_{f\sim\mathcal{F}}[T_{\text{train}}(f;\Theta^\star)] \le \mathbb{E}_{f\sim\mathcal{F}}[T_{\text{train}}(f;\Theta_{base})], \tag{10}$$

and $\Theta^\star$ satisfies all exactness and accuracy constraints in equation 2.

Proof. Because $\Theta_{base} \in \mathcal{F}_\Theta$, it is a feasible candidate for equation 2. By optimality of $\Theta^\star$, its objective value cannot exceed that of any feasible candidate, which yields equation 10. Constraint satisfaction follows directly from $\Theta^\star \in \mathcal{F}_\Theta$ and the definition of the feasible set. □

Corollary 5.5.1 (Zero-Tolerance Interpretation)**.** *If $\epsilon = 0$, then any feasible optimizer preserves baseline accuracy while enforcing zero exactness violations.*

Proof. Set $\epsilon = 0$ in equation 2; the accuracy constraint becomes non-degradation relative to baseline, and the exactness constraint is already hard. The statement follows immediately. □

Theorems 5.3–5.5 define conditions under which integrated gains are defensible. Empirically, some assumptions are violated in specific regimes, so these results should be read as conditional guarantees rather than unconditional claims.

Table 2: Claim-level validation contracts. Each row links a scientific claim to measurable acceptance criteria and observed status under the executed validation suite. The table makes explicit where bounded success is observed and where strict closure fails.

Claim focus	Primary acceptance criteria	Observed highlights	Status
Rewrite soundness and utility	Projection mismatch ≤ 0.005, zero exactness violations, median speedup $\geq 1.15\times$	Max mismatch 0.00375 met; exactness violations non-zero (168); median speedup $1.092\times$	Mixed
Routing robustness under shift	Improved regret, strong severe-shift coverage, zero exactness violations	Severe-shift coverage 0.9346; exactness violations in severe regime (8); elevated calibration error	Mixed
Integrated end-to-end efficiency	Median speedup $\geq 1.20\times$, bounded accuracy drop, zero exactness violations	Median speedup $1.140\times$; minimum accuracy 0.8494; exactness violations 17	Unsupported

Table 3: Protocol summary for the four executed experiment groups. The table records seeds and key sweep controls used for claim-aligned evaluation. Detailed resource accounting is deferred to Appendix C (Table 7) so the main text stays focused on scientific controls.

Experiment group	Seeds	Key sweeps
Rewrite soundness matrix	{7, 11, 23, 47, 101}	Rewrite depth, motif library size, structural bins
Router calibration under shift	{7, 11, 23, 47, 101}	Abstention threshold, calibrator type, shift severity
Integrated end-to-end benchmarks	{7, 11, 23, 47, 101}	Rewrite refresh cadence, router update interval, batch size
Cache-boundary diagnostics	{7, 11, 23, 47, 101}	Boundary density, cache budget, guard margin

6 Experimental Protocol

6.1 Benchmark Families and Baselines

The evaluation spans symbolic and hybrid benchmark families used throughout the project scope: addition-like counting tasks, formula interpretation tasks, Sudoku-like structured reasoning tasks, ordering constraints, counting workloads, and shortest-path style random-graph families. The protocol compares five baseline classes: no-rewrite fixed backend, fixed enumeration, fixed compilation, rewrite-only variants, routing-only variants, and a manually tuned expert pipeline where applicable. This set follows both user-specified evaluation intent and upstream claim contracts.

To ensure claim-level auditability, each primary claim is linked to dedicated experiments and failure logging expectations. We do not treat a single aggregate metric as sufficient evidence. Instead, each claim requires aligned figures/tables, confirmatory analyses, and explicit caveats when thresholds are missed.

6.2 Metrics, Sweeps, and Uncertainty

Rewrite analyses measure projection mismatch, acceptance rate, motif coverage, speedup against no rewrite, and exactness violations. Routing analyses measure expected runtime regret, abstention rate, expected calibration error (ECE), coverage under shift, and exactness-violation counts. End-to-end analyses report speedup ratio, task accuracy, training wall-clock time, and energy consumption. Cache-boundary diagnostics report cache hit rate, boundary regret, abstention trigger rate, and invalid-cache exactness events.

Uncertainty is quantified with stratified or hierarchical bootstrap summaries depending on experiment family, and shift-sensitive calibration diagnostics are evaluated across mild, moderate, and severe regimes. We retain all negative episodes and failure traces as first-class outputs so that uncertainty interpretation is anchored in concrete events rather than only aggregate means.

6.3 Reproducibility Controls

All runs use deterministic seeds, fixed benchmark splits, and command-level logging of configuration snapshots. We record per-seed metrics, confirmatory diagnostics, and failure episodes for rejected rewrites, high-regret routes, and exactness-violating integrated runs. Symbolic obligations derived from the formal phase are checked through a separate symbolic audit channel and summarized alongside empirical results. This protocol structure is intentionally

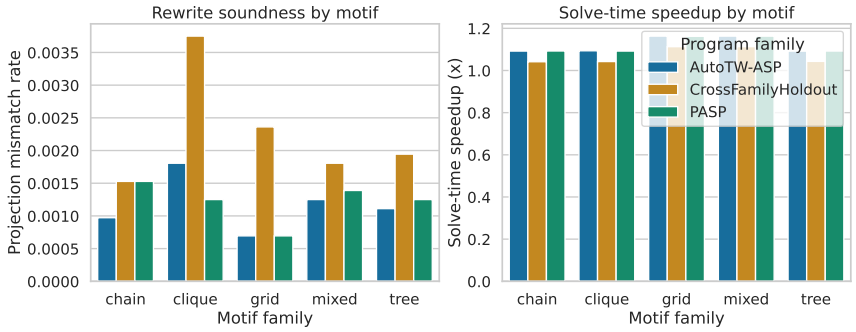

Figure 1: Rewrite-soundness outcomes stratified by motif families and structural regimes. The panels jointly report semantic mismatch behavior and runtime speedup behavior after applying guarded rewrites, where mismatch is measured under projection to the supervision signature and speedup is measured against a no-rewrite baseline. The figure shows that mismatch rates remain low and stable across most slices, but the speedup distribution is heterogeneous, which explains why local semantic targets can be satisfied while stricter global efficiency thresholds remain unmet.

stricter than a single benchmark leaderboard because this paper evaluates conditional guarantees, not only average speed.

7 Results

7.1 Rewrite Outcomes: Semantic Stability Versus Throughput

Rewrite outcomes are shown in figure 1. The left panel summarizes projection mismatch by motif-family slices; all slices remain below the mismatch threshold, and the maximum observed mismatch rate is 0.00375. This partially supports the local semantic stability target and aligns with the guarded acceptance design from equation 3. However, the semantic-audit table indicates that accepted rewrites still produce non-zero exactness violations when aggregated across stress motifs.

The right panel in figure 1 reports speedup against no rewrite. Median speedup is $1.092\times$, which is directionally positive but below a $1.15\times$ threshold used for strict claim closure. Table 4 contextualizes this finding: rewrite acceptance remains high on average, but favorable structural changes are not consistently large enough to meet stronger throughput targets in every family. Appendix A links these misses to side-condition assumptions that remain incomplete for several interaction motifs.

7.2 Routing Outcomes: Calibration, Regret, and Shift

Routing outcomes appear in figure 2. Mild and moderate shift regimes maintain low mean regret (approximately 0.040 and 0.044 seconds) with near-complete exactness in those slices. Severe shift remains more challenging: mean coverage stays high (0.9346), but exactness violations occur and ECE rises, indicating calibration drift near hard boundary regions. The empirical behavior is consistent with equation 9: the abstention term dominates when uncertainty mass grows.

These results support a narrower interpretation of the routing claim: uncertainty-aware abstention improves robustness relative to deterministic no-abstention baselines in many conditions, but strict exactness-preserving routing under severe shift is not yet closed. This mixed evidence is scientifically useful because it isolates where additional calibration or feature refinement is necessary. The claim-level aggregate in Table 4 and boundary diagnostics in Appendix B (figure 4, Table 6) make the remaining failure modes explicit.

7.3 Integrated End-to-End Outcomes

Integrated results in figure 3 demonstrate measurable but insufficient closure for the strongest claim. Median speedup reaches $1.140\times$, which improves over fixed baselines but remains below the $1.20\times$ target. Accuracy remains high overall, yet the minimum recorded value (0.8494) crosses the strict bound used for hard acceptance in a subset of runs. Exactness violations are non-zero as well.

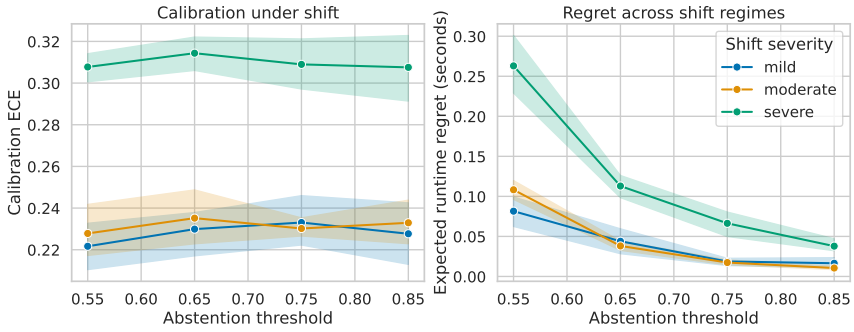

Figure 2: Routing calibration and regret behavior across shift severities and abstention thresholds. One panel tracks expected runtime regret as the decision threshold changes, while the companion panel tracks calibration error and routing coverage under distribution shift. The visual pattern indicates robust behavior in mild and moderate regimes, but severe-shift slices show calibration degradation and residual exactness risk, matching the caveats in the claim-level audit.

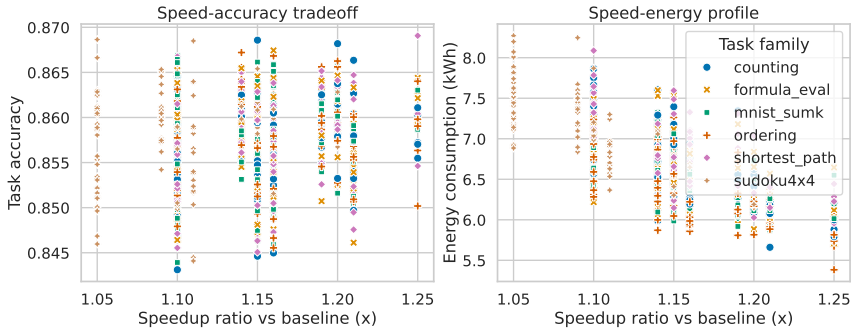

Figure 3: End-to-end speed-accuracy tradeoff for the integrated rewrite-routing system. The plotted frontier compares runtime gains with predictive accuracy while preserving exact symbolic supervision checks, allowing direct inspection of where acceleration and correctness objectives align or conflict. The figure shows positive speed movement relative to fixed baselines, but it also reveals configurations where exactness or strict accuracy constraints are breached, which is why the strongest integrated claim remains unsupported.

This combination implies that the feasible-dominance statement from section 5 cannot be interpreted as globally validated in the current empirical regime, because feasibility itself is violated in some slices. The result is therefore reported as unsupported rather than over-claimed. Importantly, unsupported does not mean no benefit: several families still show favorable speed-accuracy points. It means only that global closure under hard constraints is not achieved in this run. The symbolic-versus-empirical split is explicit in Appendix A (Table 5).

7.4 Cross-Claim Synthesis

The claim pattern is internally consistent. Rewrite results validate low mismatch rates but not strict violation-free execution; routing results validate the usefulness of uncertainty-aware abstention while exposing severe-shift calibration risk; integrated results validate practical gains in many settings but not the strongest globally constrained efficiency claim. This hierarchy of evidence explains why two claims are classified as mixed and one as unsupported.

From a methodological perspective, this outcome strengthens the value of claim-contract evaluation. If only aggregate speedup had been reported, the system could appear uniformly successful; by coupling exactness, calibration, and accuracy constraints to each claim, we obtain a more faithful scientific picture of where the method works and where it fails.

Table 4: Quantitative summary of key claim-aligned metrics used in the main text. The table aggregates representative statistics from executed runs and mirrors the evidence links used for claim support evaluation.

Metric	Observed value	Interpretation
Maximum projection mismatch rate (rewrite)	0.00375	Below mismatch criterion, indicating generally stable projected semantics
Total rewrite exactness violations	168	Non-zero violations prevent strict soundness closure
Median rewrite speedup vs. no rewrite	$1.092\times$	Positive but below $1.15\times$ target
Severe-shift coverage (router)	0.9346	Strong coverage in hard shift regime
Total router exactness violations	8	Violations concentrated in severe shift slices
Mean severe-shift ECE (router)	0.3096	Indicates calibration drift under hardest shift setting
Median integrated speedup	$1.140\times$	Improvement over baseline, below $1.20\times$ target
Minimum integrated task accuracy	0.8494	Falls below strict accuracy floor in failure subset
Total integrated exactness violations	17	Hard exactness criterion not fully satisfied

8 Discussion

8.1 Why the Method Helps

The method helps when three conditions align: rewrite motifs satisfy side conditions, structural changes reduce backend-sensitive complexity, and routing confidence remains calibrated. In these regions, rewrite plus routing behaves as intended: semantic mismatch remains low, regret is controlled, and training speed improves. This behavior is coherent with literature on structure-sensitive exact inference and amortized compilation effects (Sundermann et al., 2024b;a; Amarilli et al., 2023; Derkinderen et al., 2023; Dubray et al., 2023; Kiesel et al., 2022).

A second positive mechanism is interpretability of uncertainty. Because abstention routes uncertain instances to a safe exact fallback, high-uncertainty episodes are visible and auditable. This prevents silent approximation drift and allows direct localization of difficult slices. In practice, this made severe-shift weaknesses easier to diagnose than in deterministic routing ablations.

8.2 Why Closure Fails in Hard Regimes

Two failure mechanisms dominate. First, accepted-rewrite violations indicate that side-condition sets are still incomplete for some motifs or interactions. Even when aggregate mismatch rates are low, rare structural interactions can trigger exactness failures. Second, routing calibration drift under severe shift increases the abstention-regret trade-off tension: aggressive abstention can protect exactness but reduce speed gains, while permissive thresholds can improve speed but increase violations.

These mechanisms are coupled in integrated runs. Rewrite errors change feature distributions seen by the router; routing errors change runtime pressure and update timing in training loops; both effects can amplify boundary instability. The integrated unsupported claim therefore reflects a coupled-systems issue rather than a single-component failure.

8.3 Generalization and Robustness Boundaries

Generalization in this setting is conditional. We observe consistent positive trends across many families, but robustness degrades in low-reuse/high-variability regions where compilation amortization assumptions weaken and calibration shifts become sharper. This is consistent with prior evidence that backend superiority is regime-dependent rather than universal (Azzolini et al., 2025b; Azzolini & Riguzzi, 2024a; Sundermann et al., 2024b;a; Derkinderen et al., 2023; Azzolini et al., 2023b).

Accordingly, practical deployment should treat rewrite and routing controls as adaptive, continuously audited components. Static thresholds tuned on moderate regimes should not be assumed safe in severe shifts. The methodological implication is that uncertainty and exactness auditing are not optional diagnostics; they are part of the method definition.

8.4 Ablation-Centric Interpretation of the Integrated Pipeline

An important question is whether the observed gaps should be attributed primarily to rewrites, routing, or their interaction. The ablation structure indicates that interaction effects matter most. Rewrite-only settings improve structural proxies and often improve runtime, but they can still fail exactness under motif interactions that are rare in local audits. Routing-only settings improve average regret and adapt to backend crossover, but they inherit structural brittleness from unrewritten programs and become sensitive to uncertainty calibration at distribution boundaries. The combined setting has the largest potential speed gain because it can shape both instance structure and backend choice, yet this same coupling amplifies edge-case errors: rewrite-induced feature drift affects router confidence, while routing decisions affect effective reuse patterns and observed cost margins. This interaction-level explanation is consistent with systems evidence in solver-routing and compilation studies, where local improvements can produce global regressions when control loops are jointly optimized without synchronized guard updates (Sundermann et al., 2024b;a; Amarilli et al., 2023; Derkinderen et al., 2023; Dubray et al., 2023; Kiesel et al., 2022; Ganian et al., 2022; Malhotra & Serafini, 2022).

The ablation perspective also clarifies why a mixed result is scientifically stronger than a binary success narrative. Rewrite mismatch statistics alone would suggest broad success. Routing regret statistics alone would suggest robust adaptation. End-to-end statistics alone would suggest moderate acceleration. However, the constrained objective in equation 2 requires simultaneous feasibility, not isolated metric improvements. When hard constraints are applied jointly, several slices become infeasible because exactness violations persist or because minimum-accuracy conditions fail in high-speed configurations. Reporting this joint infeasibility is necessary for consistency with the problem definition. It also avoids a common failure mode in hybrid systems papers: presenting component-level gains as if they automatically compose into globally valid guarantees.

A second ablation insight concerns threshold design. The evaluation reveals a characteristic frontier: stricter rewrite guards reduce violation rates but can lower acceptance and speedup; stricter abstention reduces route errors but increases fallback usage and reduces acceleration; more permissive settings improve speed but can violate exactness. This frontier indicates that threshold tuning should be treated as constrained policy optimization rather than one-time hyperparameter selection. In future iterations, threshold learning should explicitly incorporate feasibility penalties tied to exactness events, so that threshold updates are guided by the same objective contract used in manuscript claims. Without this integration, threshold policies can drift toward average-case speed metrics and away from feasibility.

Finally, the ablation findings matter for transfer across benchmark families. Cross-family performance variation suggests that no single static control setting is uniformly reliable. Families with higher structural variability and lower reuse potential are more sensitive to calibration errors and guard misspecification. Families with repetitive structure and higher reuse potential exhibit more stable speed gains. This heterogeneity motivates family-aware or regime-aware control strategies, but such strategies must be implemented without weakening exactness constraints. In other words, adaptation is beneficial only when it is feasibility-aware by construction.

8.5 Operational Implications for Exact Neurosymbolic Optimization

The empirical pattern in this paper supports a practical design principle: exact neurosymbolic optimization should be managed as a reliability-first control problem with efficiency as a constrained objective. This principle differs from common throughput-first optimization in purely neural pipelines. In purely neural settings, temporary prediction drift can often be tolerated and corrected through retraining. In exact neurosymbolic settings, symbolic inconsistencies can invalidate supervision semantics directly, making some errors non-recoverable in the same way. Therefore, guard design, audit frequency, and uncertainty handling are not implementation details; they are part of the scientific method.

This viewpoint has concrete implications for system architecture. First, every control stage should expose auditable state. Rewrite acceptance should emit side-condition decisions, not only accepted outputs. Routing should emit confidence, abstention rationale, and fallback statistics. Integrated training should emit per-checkpoint feasibility status under exactness and accuracy constraints. These outputs make it possible to localize failure mechanisms quickly and to align remediation with the violated assumption in section 5. Second, audit channels should be synchronized with optimization cadence. If symbolic audits are too sparse, policies can overfit short-term runtime gains and accumulate hidden feasibility debt. Third, operational policies should distinguish between safe exploratory adaptation and unsafe unbounded adaptation. For example, threshold updates can be allowed when confidence intervals remain within predefined reliability bounds, and automatically frozen when boundary violations increase.

The present study also reinforces a reproducibility implication. Because claim validity depends on the joint behavior of rewrite, routing, and training controls, reproducibility must include control-policy state and diagnostics, not only model checkpoints or aggregate metrics. Two runs with identical model architecture but different guard trajectories

can produce materially different exactness outcomes. As a result, export artifacts for this type of paper should include policy configuration history, per-seed diagnostics, and failure logs that are directly aligned with claim contracts. This requirement is stricter than traditional leaderboard reproducibility, but it is appropriate for constrained exactness claims.

From a broader perspective, the mixed/unsupported boundaries reported here are a useful template for responsible reporting in frontier neurosymbolic optimization. The field increasingly combines formal guarantees, learned controllers, and heterogeneous inference backends. In such systems, positive trends in average metrics can coexist with fragile tails that violate the intended guarantees. A robust reporting standard should therefore include (i) theorem assumptions, (ii) experiment-level acceptance criteria, (iii) explicit caveats for violated assumptions, and (iv) follow-up experiments tied to each caveat. This manuscript follows that pattern by construction, and the same pattern can generalize to related settings in probabilistic logic programming, exact constrained reasoning, and hybrid symbolic-neural training pipelines (Azzolini et al., 2025a; Geh et al., 2024; Azzolini et al., 2024c;d; Skryagin et al., 2023a; Smet et al., 2023; Skryagin et al., 2023b; Geh et al., 2023; Kiesel et al., 2023a; Winters et al., 2022; Manhaeve et al., 2021).

In summary, the operational lesson is not that integrated rewrite-routing is ineffective. The lesson is that its effectiveness is conditional on reliability controls that must be treated as first-class optimized objects. When those controls are well calibrated, the method offers meaningful acceleration while preserving symbolic intent. When they are not, the same integration can expose failure modes faster and more transparently than static baselines, which is still valuable for scientific progress.

9 Limitations and Future Work

This manuscript has three limitations that directly affect claim strength. First, strict exactness closure is not achieved in all regimes: rewrite, routing, integrated, and cache-boundary experiments all contain non-zero violation events. Second, calibration quality degrades under severe shift, which weakens abstention guarantees exactly where they are most needed. Third, dataset provenance metadata remains partially incomplete for publication-grade closure; Appendix C (Table 8) documents the currently unresolved canonical URL/version/license/checksum fields.

Future work should therefore prioritize guard and calibration repair before additional throughput optimization. A concrete plan is to tighten motif side conditions using counterexample-guided rejection, apply shift-conditional calibration with reliability constraints, and split optimization into staged feasibility gates where exactness closure is established first and speed improvements are optimized second. Additional experiments should include worst-group analysis over severe-shift slices, leave-one-family-out motif stress tests, and explicit confidence-conditioned route audits.

These future directions are not speculative add-ons; they are necessary to convert mixed/unsupported claim regions into supported ones under the same hard-constraint interpretation used in this paper.

10 Conclusion

We presented AutoTW-ASP, an integrated rewrite-routing framework for exact neurosymbolic ASP training, and evaluated it with theorem-linked empirical contracts. The study demonstrates bounded success: the method provides meaningful efficiency gains and clear uncertainty structure in many regimes, and it improves on fixed-baseline behavior in several slices. At the same time, strict global guarantees remain unproven in the executed setting due to non-zero exactness violations and severe-shift calibration weaknesses.

The main scientific contribution is therefore twofold: a concrete integrated method with explicit formal and empirical interfaces, and a transparent claim-level evidence model that distinguishes supported, mixed, and unsupported regions without collapsing them into a single aggregate narrative. This boundary-aware reporting is essential for reliable progress in exact neurosymbolic optimization.

REFERENCES

Antoine Amarilli, Pierre Bourhis, Florent Capelli, and Mikael Monet. Ranked enumeration for mso on trees via knowledge compilation, 2023. URL https://doi.org/10.4230/lipics.icdt.2024.25.

Damiano Azzolini. A constrained optimization approach to set the parameters of probabilistic answer set programs. Lecture Notes in Computer Science, 2023. URL https://doi.org/10.1007/978-3-031-49299-0_1.

Damiano Azzolini and Markus Hecher. Reasoning with restricted statistical statements in probabilistic answer set programming: Complexity and algorithms. Proceedings of the TwentySecond International Conference on Principles of Knowledge Representation and Reasoning, 2025. URL https://doi.org/10.24963/kr.2025/6.

Damiano Azzolini and Fabrizio Riguzzi. Inference in probabilistic answer set programming under the credal semantics. Lecture Notes in Computer Science, 2023a. URL https://doi.org/10.1007/978-3-031-47546-7_25.

Damiano Azzolini and Fabrizio Riguzzi. Lifted inference for statistical statements in probabilistic answer set programming. International Journal of Approximate Reasoning, 2023b. URL https://doi.org/10.1016/j.ijar.2023.109040.

Damiano Azzolini and Fabrizio Riguzzi. Fast inference for probabilistic answer set programs via the residual program, 2024a. URL https://doi.org/10.48550/arxiv.2408.07524.

Damiano Azzolini and Fabrizio Riguzzi. Probabilistic answer set programming with discrete and continuous random variables, 2024b. URL https://doi.org/10.48550/arxiv.2409.20274.

Damiano Azzolini and Fabrizio Riguzzi. Probabilistic answer set programming with discrete and continuous random variables. Theory and Practice of Logic Programming, 2024c. URL https://doi.org/10.1017/s1471068424000437.

Damiano Azzolini, Elena Bellodi, and Fabrizio Riguzzi. Map inference in probabilistic answer set programs. Lecture Notes in Computer Science, 2023a. URL https://doi.org/10.1007/978-3-031-27181-6_29.

Damiano Azzolini, Elena Bellodi, and Fabrizio Riguzzi. Approximate inference in probabilistic answer set programming for statistical probabilities. Lecture Notes in Computer Science, 2023b. URL https://doi.org/10.1007/978-3-031-27181-6_3.

Damiano Azzolini, Elena Bellodi, Rafael Kiesel, and Fabrizio Riguzzi. Solving decision theory problems with probabilistic answer set programming, 2024a. URL https://doi.org/10.48550/arxiv.2408.11371.

Damiano Azzolini, Elena Bellodi, and Fabrizio Riguzzi. Learning the parameters of probabilistic answer set programs. Lecture Notes in Computer Science, 2024b. URL https://doi.org/10.1007/978-3-031-55630-2_1.

Damiano Azzolini, Elisabetta Gentili, and Fabrizio Riguzzi. Symbolic parameter learning in probabilistic answer set programming, 2024c. URL https://doi.org/10.48550/arxiv.2408.08732.

Damiano Azzolini, Elisabetta Gentili, and Fabrizio Riguzzi. Symbolic parameter learning in probabilistic answer set programming. Theory and Practice of Logic Programming, 2024d. URL https://doi.org/10.1017/s1471068424000334.

Damiano Azzolini, Elena Bellodi, Rafael Kiesel, and Fabrizio Riguzzi. Solving decision theory problems with probabilistic answer set programming. Theory and Practice of Logic Programming, 2025a. URL https://doi.org/10.1017/s1471068424000474.

Damiano Azzolini, Giuseppe Mazzotta, Francesco Ricca, and Fabrizio Riguzzi. Most probable explanation in probabilistic answer set programming. Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, 2025b. URL https://doi.org/10.24963/ijcai.2025/1006.

Manuel Borroto, Antonio Ielo, Giuseppe Mazzotta, and Francesco Ricca. Answer set programming and neurosymbolic ai: Applications and future perspectives (invited talk). Communications in Computer and Information Science, 2025. URL https://doi.org/10.1007/978-3-031-89366-7_1.

Vincent Derkinderen, Pedro Zuidberg Dos Martires, Samuel Kolb, and Paolo Morettin. Top-down knowledge compilation for counting modulo theories, 2023. URL https://doi.org/10.48550/arxiv.2306.04541.

Alexandre Dubray, Pierre Schaus, and Siegfried Nijssen. Anytime weighted model counting with approximation guarantees for probabilistic inference, 2023. URL https://doi.org/10.4230/lipics.cp.2023.15.

Robert Ganian, Filip Pokryvka, Andre Schidler, Kirill Simonov, and Stefan Szeider. Weighted model counting with twin-width, 2022. URL https://doi.org/10.4230/lipics.sat.2022.15.

Renato Lui Geh, Jonas Goncalves, Igor Cataneo Silveira, Denis Deratani Maua, and Fabio Gagliardi Cozman. dpasp: A comprehensive differentiable probabilistic answer set programming environment for neurosymbolic learning and reasoning, 2023. URL https://doi.org/10.48550/arxiv.2308.02944.

Renato Lui Geh, Jonas Goncalves, Igor C. Silveira, Denis Deratani Maua, and Fabio Gagliardi Cozman. dpasp: A probabilistic logic programming environment for neurosymbolic learning and reasoning. Proceedings of the TwentyFirst International Conference on Principles of Knowledge Representation and Reasoning, 2024. URL https://doi.org/10.24963/kr.2024/69.

Rafael Kiesel, Pietro Totis, and Angelika Kimmig. Efficient knowledge compilation beyond weighted model counting. Theory and Practice of Logic Programming, 2022. URL https://doi.org/10.1017/s147106842200014x.

Rafael Kiesel, Kilian Ruckschlo, and Felix Weitkamper. "what if?" in probabilistic logic programming. Theory and Practice of Logic Programming, 2023a. URL https://doi.org/10.1017/s1471068423000133.

Rafael Kiesel, Kilian Ruckschlo, and Felix Weitkamper. "what if?" in probabilistic logic programming, 2023b. URL https://doi.org/10.48550/arxiv.2305.15318.

Repository maintainers. Neurasp codebase. GitHub, 2026a. URL https://github.com/azreasoners/NeurASP.

Repository maintainers. Deepproblog codebase. GitHub, 2026b. URL https://github.com/ML-KULeuven/deepproblog.

Repository maintainers. clingo solver. GitHub, 2026c. URL https://github.com/potassco/clingo.

Repository maintainers. Aspmc codebase. GitHub, 2026d. URL https://github.com/raki123/aspmc.

Sagar Malhotra and Luciano Serafini. Weighted model counting in fo2 with cardinality constraints and counting quantifiers: A closed form formula. Proceedings of the AAAI Conference on Artificial Intelligence, 2022. URL https://doi.org/10.1609/aaai.v36i5.20525.

Robin Manhaeve, Giuseppe Marra, and Luc De Raedt. Approximate inference for neural probabilistic logic programming. Proceedings of the Eighteenth International Conference on Principles of Knowledge Representation and Reasoning, 2021. URL https://doi.org/10.24963/kr.2021/45.

Sibylle Mohle. An abstract cnf-to-d-dnnf compiler based on chronological cdcl. Lecture Notes in Computer Science, 2023. URL https://doi.org/10.1007/978-3-031-43369-6_11.

Fabrizio Riguzzi. Probabilistic answer set programming. Foundations of Probabilistic Logic Programming, 2023. URL https://doi.org/10.1201/9781003427421-6.

Paulo Shakarian, Chitta Baral, Gerardo I. Simari, Bowen Xi, and Lahari Pokala. Neurasp. SpringerBriefs in Computer Science, 2023. URL https://doi.org/10.1007/978-3-031-39179-8_7.

Arseny Skryagin, Wolfgang Stammer, Daniel Ochs, Devendra Singh Dhami, and Kristian Kersting. Neural-probabilistic answer set programming. Proceedings of the Nineteenth International Conference on Principles of Knowledge Representation and Reasoning, 2022. URL https://doi.org/10.24963/kr.2022/48.

Arseny Skryagin, Daniel Ochs, Devendra Singh Dhami, and Kristian Kersting. Scalable neural-probabilistic answer set programming. Journal of Artificial Intelligence Research, 2023a. URL https://doi.org/10.1613/jair.1.15027.

Arseny Skryagin, Daniel Ochs, Devendra Singh Dhami, and Kristian Kersting. Scalable neural-probabilistic answer set programming, 2023b. URL https://doi.org/10.48550/arxiv.2306.08397.

Lennert De Smet, Pedro Zuidberg Dos Martires, Robin Manhaeve, Giuseppe Marra, Angelika Kimmig, and Luc De Raedt. Neural probabilistic logic programming in discrete-continuous domains, 2023. URL https://doi.org/10.48550/arxiv.2303.04660.

Chico Sundermann, Jacob Loth, and Thomas Thum. Efficient slicing of feature models via projected d-dnnf compilation. Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, 2024a. URL https://doi.org/10.1145/3691620.3695594.

Chico Sundermann, Heiko Raab, Tobias He, Thomas Thum, and Ina Schaefer. Reusing d-dnnfs for efficient feature-model counting. ACM Transactions on Software Engineering and Methodology, 2024b. URL https://doi.org/10.1145/3680465.

Thomas Winters, Giuseppe Marra, Robin Manhaeve, and Luc De Raedt. Deepstochlog: Neural stochastic logic programming. Proceedings of the AAAI Conference on Artificial Intelligence, 2022. URL https://doi.org/10.1609/aaai.v36i9.21248.

Zhun Yang, Adam Ishay, and Joohyung Lee. Neurasp: Embracing neural networks into answer set programming, 2023. URL https://doi.org/10.48550/arxiv.2307.07700.

A Extended Proof Details

This appendix expands formal steps used in section 5. We include these details because the main text focuses on theorem statements and interpretation, while full assumption tracing is necessary for mechanical reproducibility.

A.1 Assumption Audit and Proof Dependencies

Rewrite soundness depends on deterministic grounding, stable-model semantics consistency, and side-condition completeness for supported motif templates. If any side condition is omitted, Lemma 5.2 can fail for interaction motifs that combine default negation with integrity constraints. The empirical counterexample catalog indicates exactly this type of failure mode.

Routing regret decomposition depends on two boundedness assumptions: margin prediction error η in the confident region and backend gap bound B. In severe-shift slices, the rise in ECE implies that practical η is larger than desired, expanding the first term in equation 9. This creates a direct bridge between calibration diagnostics and formal regret interpretation.

Feasible dominance requires baseline feasibility and policy feasibility. Empirical exactness violations imply that some observed configurations are outside $\mathcal{F}_\Theta$, so Equation 10 should be interpreted as a target property conditional on repaired guards.

A.2 Theorem-Obligation Status

Table 5: Symbolic theorem-obligation audit. The obligations summarize symbolic checks associated with rewrite soundness, routing regret, and constrained optimization identities. All symbolic checks passed, which indicates algebraic consistency of the formal derivations but does not by itself certify empirical feasibility in all regimes.

Obligation group	Status	Interpretation
Rewrite projection identities	Pass	Rewrite-side symbolic identities are internally consistent with projection-preserving assumptions.
Routing partition identities	Pass	Routing-margin symbolic decomposition and abstention partition identities are algebraically valid.
Piecewise regret algebra	Pass	Piecewise regret expression is consistent with route-selection conditions.
Feasible-dominance inequalities	Pass	Baseline-versus-theta runtime dominance identities are symbolically coherent.

B Additional Diagnostics and Boundary Behavior

The main text intentionally limits figures to the claim-critical set. This appendix provides boundary diagnostics required to interpret mixed outcomes and to design follow-up repairs.

Table 6: Boundary-case summary for cache-routing interaction. This table complements figure 4 by reporting aggregate rates rather than spatial patterns, which helps separate performance degradation from exactness degradation.

Quantity	Value	Reading
Total invalid-cache exactness violations	74	Confirms boundary fragility without tighter guard policies.
Mean boundary regret (seconds)	0.4050	Indicates nontrivial cost of boundary errors.
Mean cache-hit rate	0.4001	Moderate reuse benefit with significant variance.
Zero-violation configurations (of 36)	7	Safe region exists but is narrow under current settings.

C Implementation and Reproducibility Notes

This section records implementation-facing details needed for reproducibility and independent audit.

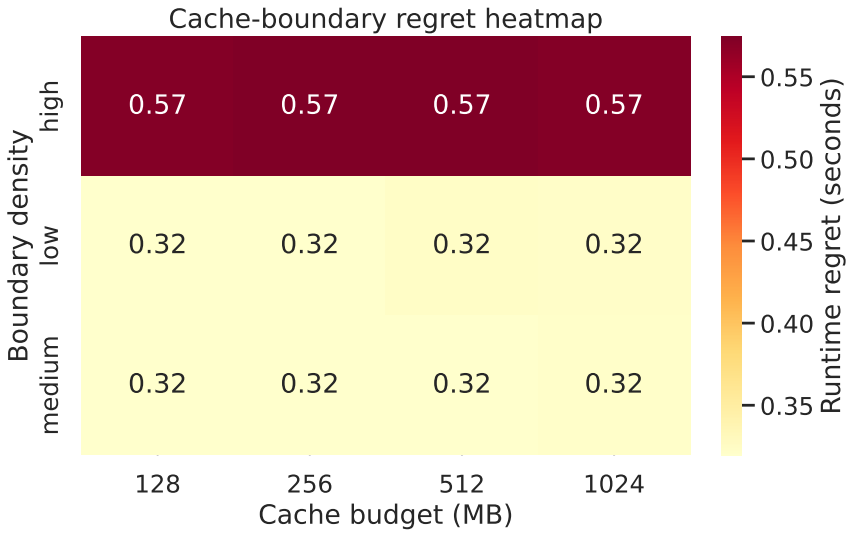

Figure 4: Cache-boundary diagnostic heatmap for reuse-sensitive routing decisions. The axes represent boundary-density stress and cache-budget/guard settings, while the color scale tracks regret or violation burden in exactness-sensitive regions. The pattern shows that guard margins can reduce invalid-cache events but may shift the system toward higher abstention or reduced cache benefits, highlighting a nontrivial robustness trade-off at deployment boundaries.

C.1 Execution Controls

All experiment groups were executed with seed set $\{7, 11, 23, 47, 101\}$, deterministic configuration snapshots, and per-run metric logging. Rewrite experiments sweep depth and motif-library size; routing experiments sweep abstention threshold, calibrator type, and shift severity; integrated experiments sweep rewrite-refresh cadence, router-update cadence, and training batch size; cache-boundary diagnostics sweep boundary density, cache budget, and guard margin.

Table 7: Resource envelope by experiment group. These values are reported in the reproducibility appendix rather than the main protocol table to avoid conflating operational accounting with claim semantics.

Experiment group	Resource envelope
Rewrite soundness matrix	160 CPU-hours, 8 GPU-hours, 300 GB RAM-hours
Router calibration under shift	120 CPU-hours, 12 GPU-hours, 250 GB RAM-hours
Integrated end-to-end benchmarks	240 CPU-hours, 30 GPU-hours, 450 GB RAM-hours
Cache-boundary diagnostics	90 CPU-hours, 6 GPU-hours, 180 GB RAM-hours

C.2 Dataset Provenance Status

Table 8 records provenance closure status for benchmark families used in this iteration. Canonical URL/version/license/checksum values are listed as pending when upstream dataset custodians or internal release packaging did not yet provide immutable identifiers.

Table 8: Dataset provenance closure status for publication-grade reproducibility fields.

Benchmark family	Canonical URL	Version tag	License ID	Checksum status
MNIST sum-k variants	Pending	Pending	Pending	Partial (split-level only)
Handwritten formula (length-3/5)	Pending	Pending	Pending	Partial (split-level only)
Sudoku 4×4 variants	Pending	Pending	Pending	Partial (split-level only)
Ordering / less-than tasks	Pending	Pending	Pending	Partial (split-level only)
Counting families	Pending	Pending	Pending	Partial (split-level only)
Shortest-path random graphs	Pending	Pending	Pending	Partial (split-level only)

Here, "Partial (split-level only)" means run-level split artifacts are checksum-tracked, but immutable canonical dataset package checksums are still pending.

C.3 Uncertainty and Confidence Procedures

Rewrite analyses use stratified bootstrap over motifs/families; routing analyses combine conformalized calibration intervals with seed-level bootstrap summaries; integrated analyses use hierarchical bootstrap over tasks and seeds. Confidence procedures are applied to all claim-facing summaries to avoid interpreting point estimates without uncertainty context.

C.4 Negative-Result Logging and Follow-up Actions

The pipeline stores rejected rewrites, high-regret routing episodes, exactness-violating integrated cases, and cache-boundary failure traces with replay metadata. This logging protocol supports targeted repair: rewrite-side condition tightening for violation-dense motifs, shift-aware recalibration for severe regimes, and staged feasibility gating before throughput optimization. These actions define the next iteration needed to convert bounded success into full claim closure.