FROM-ROUTE: FROM OPERATIONS RESEARCH PROBLEMS TO MIP FORMULATIONS FOR BETTER SOLVER PERFORMANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Formulating real-world operations research problems as mixed-integer programs (MIPs) requires choices of variables, constraints, and representations that shape solver performance. We present FROM-ROUTE, a framework for enhancing LLM routing among MIP formulations to improve solution performance within a fixed program budget. Deterministic analysis connects the original problem's structure and numerical data to three natural-language mechanism cards, supporting an ordered selection from an existing formulation-expert library. Multicommodity network design is the primary study of routing mechanisms; vertex coloring and resource-constrained project scheduling explore cross-domain applicability through domain-specific adapters. On 2,335 instances evaluated through independently validated trajectories, mean normalized primal integral is 57.0%, 14.6%, and 3.6% lower than that of the best fixed expert in each respective domain, alongside more validated model completions. Knowledge comparisons favor instance-conditioned cards over absent or generic mechanism context; restricted-completion relay is explored only for network design. On network design and coloring, FROM-ROUTE also improves solution metrics over direct and expert-guided modeling-kernel generation systems, using fewer provider-reported tokens in the calls producing retained decisions or code. These are descriptive comparisons: the evaluation corpora informed method development, and generation systems use different model and inference configurations.

1. INTRODUCTION

Representing an operations research problem as a mixed-integer program (MIP) requires choices that affect model size, relaxation strength, and solver performance. Commodity aggregation in network design, for example, reduces representation size but can weaken the relaxation (Kazemi et al., 2021). A valid formulation therefore leaves a computational question: which available representation best serves the current instance within a fixed budget? Figure 1 locates this decision before solver presolve and search.

LLM-based systems already construct optimization models, learn modeling strategies, search formulations with solver feedback, and select optimization experts (Liang et al., 2026; Zhao et al., 2026; Yuan et al., 2026; Xu et al., 2026). We study a specific setting: the original objective and numerical instance are fixed, and executable formulation alternatives are available. How can current-instance numerical evidence and natural-language mechanism knowledge improve an LLM's routing among these formulation alternatives?

FROM-ROUTE aims to improve a general-purpose LLM's formulation decisions by connecting numerical instance evidence with natural-language mechanism knowledge. The Analyzer extracts instance evidence; deterministic rules supply three mechanism cards; the LLM Decider, accessed through an API, selects an ordered triple from the entire eligible expert library. Selected experts construct MIPs, invoke the solver, and recover original-space decisions for independent validation. The framework adapts this interface to multicommodity network design (MCND), vertex coloring (VCP), and resource-constrained project scheduling (RCPSP).

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.

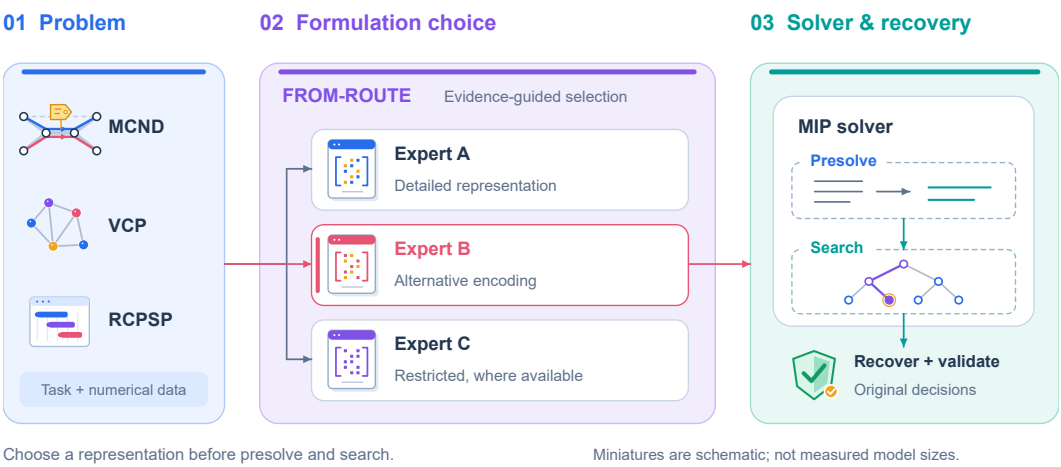

Figure 1: The modeling-stage intervention precedes solver presolve and search. Expert construction and original-decision recovery remain inside the program budget. Model miniatures are schematic, not measured sizes or performance. Exact and explicitly restricted representations both require recovered-solution validation.

Our contribution is a way to inform LLM formulation choices with structural evidence: instance features describe construction demands, and natural-language cards explain the associated modeling trade-offs. MCND provides the primary mechanism study; VCP/RCPSP explore the applicability of the interface with domain-specific adapters. On 2,335 instances, fixed experts, alternative LLM contexts, and MCND/VCP modeling-kernel generation systems evaluate the routing system through validated trajectories and acquisition tokens.

2. Related Work

Formulations and selection. Partial commodity aggregation, coloring partial-order models, and scheduling time/event representations expose concrete modeling trade-offs (Kazemi et al., 2021; Jabrayilov & Mutzel, 2022; Koné et al., 2011; Tesch, 2020). Rice's algorithm-selection framework links instance characteristics to performance criteria; SATzilla operationalizes selection with hardness models, pre-solving, backup solvers, and feature costs (Rice, 1976; Xu et al., 2008). Xu et al. train an LLM to select inventory-allocation formulations with different priorities using solver-derived supervision (Xu et al., 2026). Our alternatives share a fixed original objective, assessed through recovered feasible solutions.

Language-based modeling and strategy. NL4Opt studies entities and intermediate representations for optimization word problems (Ramamonjison et al., 2022). OptiMUS constructs and debugs models and solver code, while ORLM trains modeling capabilities with synthesized instructions (AhmadiTeshnizi et al., 2024; Huang et al., 2025). SAGE represents modeling strategies explicitly and includes solver efficiency in its reward; FormuEvo evolves executable formulations using diagnosis and structured experience (Zhao et al., 2026; Yuan et al., 2026). FROM-ROUTE chooses existing programs before their execution, rather than evaluating unrestricted natural-language parsing or online formulation generation. We do not claim superiority over these unevaluated methods. Appendix C.5 gives a balanced qualitative comparison of human modeling, LLM generation, and reusable-expert selection.

Knowledge, tools, and validation. RAG supplies external knowledge to language models (Lewis et al., 2020). LEAN-LLM-OPT combines structured modeling workflows with data tools, and OPT2CODE retrieves solver documentation with LLM-based checking (Liang et al., 2026; Ahmed & Choudhury, 2025). Our deterministic recommender supplies mechanism cards for expert selection. EquivaMap checks proposed variable mappings, and FLARE targets reformulation certificates (Zhai et al., 2025; Robbins et al., 2026). Our assurance scope is narrower: validating recovered so-

lutions and recomputing original objectives, without proving formulation equivalence. The primal integral motivates our anytime quality measure (Berthold, 2013).

3. Problem Setup

An instance I specifies numerical data and the semantic contract of

$$\min_{x \in \mathcal{X}(I)} f_I(x),$$

Expert $e \in \mathcal{E}$ constructs model $\mathcal{M}_e(I)$, solves it, and recovers original decisions through R_e. Supported semantics and implementations define the eligible set $\mathcal{E}_{\mathrm{adm}}(I)$, without guaranteeing budget feasibility. Independent validation checks $R_e(\xi^{(e)}) \in \mathcal{X}(I)$ and recomputes $f_I(R_e(\xi^{(e)}))$. Experts may preserve the original problem or restrict its representation, as with MCND's finite PATH-K10 pool. A restricted optimum can yield a valid original solution without certifying original-problem optimality.

Given evidence $s = A(I)$, semantic contract $c(I)$, frozen knowledge $\mathcal{K}$, and budget T, the Decider returns

$$(\mathbf{e}, \rho) = \pi(c(I), s, \mathcal{K}, T), \qquad \mathbf{e} = (e_1, e_2, e_3) \in \mathcal{O}_3(\mathcal{E}_{\mathrm{adm}}(I)),$$

where $\mathcal{O}_3$ contains ordered triples of distinct eligible experts. Record ρ stores the response, evidence references, acceptance, and fallback provenance. All evaluated domains have at least three eligible configurations. The decision reads current input and frozen knowledge, not historical winners, scoring references, or a result library. Execution observations affect only declared stopping, fallback, and MCND relay policies; they do not revise the accepted decision or expert code.

3.1. The formulation-expert library

MCND chooses opened arcs and commodity flows, VCP assigns vertex colors, and RCPSP schedules activity start times. Formulations encode these decisions differently, for example through aggregated flows, color-order thresholds, or event-based time. The library contains 40 configurations, not 40 independently proposed families. MCND has DA commodity-level arc flows, FA-O/FA-D endpoint aggregation, twelve PAi/PAe grouping and critical-arc variants, and restricted PATH-K10 (Kazemi et al., 2021). VCP has direct color assignment (ASS), six partial-order/hybrid configurations, and representative-based color classes (REP) (Jabrayilov & Mutzel, 2022; Campêlo et al., 2008). RCPSP has five discrete-time, four event-based, four resource-flow/unit/forbidden-set, and three sequencing/overlap configurations (Koné et al., 2011; Tesch, 2020; Kopanos et al., 2014; Klein, 2025).

Commodity grouping, graph/palette structure, and scheduling windows or resource interactions affect what these experts construct. Mechanism knowledge explains these relationships and their applicability; performance is assessed empirically. Appendix A maps all IDs to principles, equations, assumptions, and recovery contracts, including the individual scheduling configurations.

4. Method

Figure 2 presents the shared interface, with domain configurations in Table 1. Appendices C.4, D, and G specify the adapters and implementation details.

4.1. Deterministic instance evidence

The Analyzer checks expert preconditions and separates structural counts, construction proxies, and missing quantities without solving candidate MIPs. MCND uses aggregation groups, exact DA/FA counts, and path estimates sampled from up to sixteen commodity positions. Missing path evidence remains explicit; its five-second subprocess timeout does not bound all analysis. VCP uses

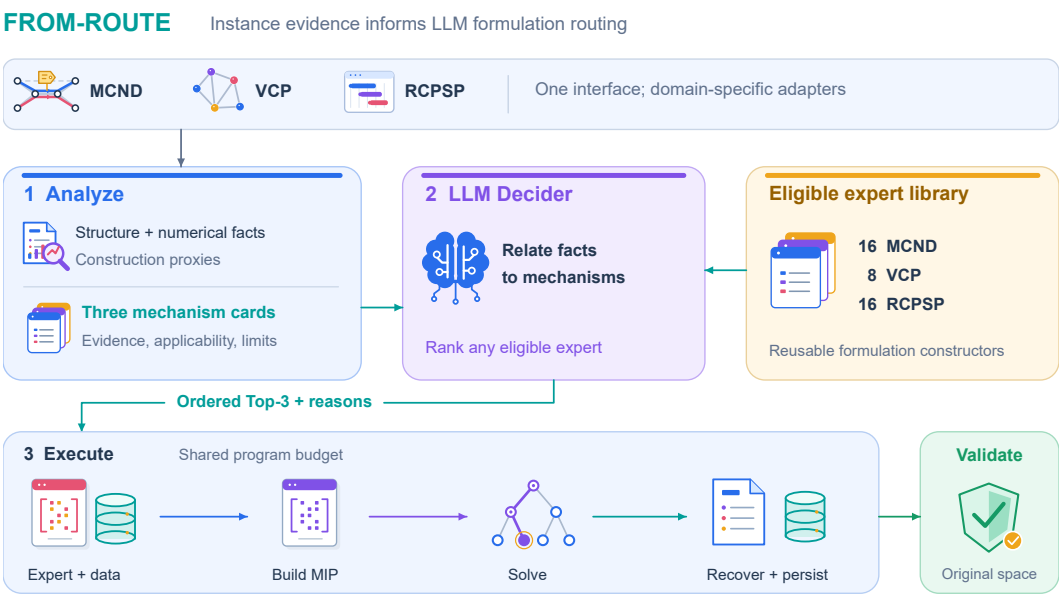

Figure 2: Shared architecture with domain adapters. Instance evidence and three mechanism cards support selection from the entire eligible library. Card labels illustrate information types, not actual cards or rankings. Program execution covers construction through persistence; external analysis, acquisition, and parent validation are separate. Early-failure fallback is shared; only MCND explores one cold relay after validated restricted PATH completion. Domain adapters specify prompts, priors, and budgets.

graph structure, a greedy palette, checked cliques, and construction counts. RCPSP uses precedence/resources, critical-path and feasible-schedule information, time windows, and construction proxies. These observations are not measured runtimes or completion guarantees.

4.2. Instance-conditioned mechanism knowledge

A deterministic domain rule G_d selects three experts for mechanism-card coverage; the LLM separately chooses the execution order. Their frozen mechanisms K_e form the cards:

$$S_d = G_d(s, \mathcal{E}_{\text{adm}}(I)), \qquad \mathcal{C}_d(I) = \{\text{card}(K_e, s, e) : e \in S_d\}, \qquad |S_d| = 3.$$

Each card connects a mechanism to current evidence and applicability or uncertainty limits. MCND combines construction-work proxies, commodity distinction, path pressure, family priors, and diversity terms. The scores organize structural evidence for the Decider; they are not learned runtime or success predictors. Appendices D/G/I specify their construction and parameters.

MCND displays cards in name order, hiding scores and rank. VCP/RCPSP provide default ordering recommendations that the LLM may revise using instance evidence. All eligible methods remain selectable; these priors are not calibrated success probabilities. We use RAG to denote this provision of frozen mechanism knowledge; the MCND builder uses deterministic rules, without a vector database or online document search.

4.3. LLM decision and response acceptance

The LLM receives instance facts, all candidate descriptions, and mechanism cards, then returns `abstain`, `order`, and `reasons`. A non-abstaining response must name three distinct eligible experts and cite supplied facts. Card-covered choices must also cite a supplied card, not neces-

Table 1: Domain evidence, knowledge presentation, and completion policy.

Domain	Evidence used for modeling choices	Knowledge presentation	Execution after a validated completion
MCND	Commodity groups, construction counts, sampled paths and load proxies	Three name-ordered cards; scores/rank hidden	Restricted PATH completion may trigger one cold relay
VCP	Graph structure, greedy palette, checked cliques, construction counts	Three cards plus an ordering prior	Stop; no post-completion relay
RCPSP	Precedence/resources, schedule information, time windows, construction proxies	Three cards plus an ordering prior	Stop; no post-completion relay

sarily their own. These checks enforce structure and citation existence, not semantic correctness. Appendix D gives the MCND contract and Appendix G the adapted-domain contracts.

An unusable response invokes a deterministic construction-cost fallback, with unknown workloads last. The MCND production request uses one call without automatic retry, disabled thinking, temperature zero, and a 1,400-token output cap. Appendices D/G/I provide runtime details; Section D.10 analyzes recorded LLM choices.

4.4. Model instantiation and budgeted execution

The ordered triple defines fallback priority, not a requirement to execute all three experts. The first receives the remaining program allowance; only an early unsuccessful return or the MCND-specific completion relay permits continuation. Budget exhaustion ends execution. For credited stage duration τ_j and local ready time u,

$$t = \sum_{k<j} \tau_k + u, \qquad \sum_j \tau_j \le T.$$

Child startup, preprocessing, modeling, solving, recovery, and persistence consume T. Analysis, LLM/network time, and parent validation/coordination are accounted for separately. A genuinely failed later child does not erase eligible earlier Full evidence or completion. Response fallback and execution fallback are distinct policies.

4.5. Domain configurations and MCND relay

All domains retain Top-3 and early-failure fallback. Only MCND explores one cold relay after a validated restricted PATH optimum. It preserves that completion and may execute the first untried original-scope expert in the triple within the remaining budget, without transferring solver state or a warm start. No relay preserves Full's order and ordinary fallback. Domain adaptations do not constitute identical-prompt or zero-shot transfer experiments.

5. Experimental Protocol

5.1. Tasks and comparisons

Table 2 lists the complete evaluated sets, including unsuccessful runs. Each fixed expert uses one formulation throughout its domain. FROM-ROUTE (Full), No RAG, and Generic RAG use the same instance facts and candidate descriptions, with different mechanism contexts. MCND additionally evaluates No relay.

Table 2: Evaluation scope. All domains include Full, No RAG, and Generic RAG with early-failure fallback.

Domain	Instances	Fixed experts	Program budget	Additional conditions
MCND	196	16	3,600 s	No relay; Direct/Expert RAW
VCP	99	8	3,600 s	Direct/Expert RAW
RCPSP	2,040	16	120 s	None

Runs use one solver thread. VCP's resource subset was selected after observing the original matrix: 99 of 125 graphs satisfy $n \leq 1000$ and $m \leq 50000$. Exact OBDD was excluded, leaving eight experts. Appendix G.2 records the memory policies for retained and supplemental VCP runs. RCPSP RAW is outside scope because of its two-minute budget and large instance count, not an observed effectiveness result.

5.2. Solution quality, completion, and references

Independent validation checks original constraints and objectives. Eligible witnesses enter trajectories at their recorded ready times after recovery and persistence. Let $\ell_{i,m}(t)$ be normalized loss for the best eligible witness from method m on instance i; loss is 1 before any valid witness. We report

$$\mathrm{NPI}_{i,m} = \frac{1}{T} \int_0^T \ell_{i,m}(t)\, dt.$$

NPI accumulates the wait for a first valid solution and the subsequent normalized gap to the common best-observed reference. Earlier and better solutions reduce it; integration follows recorded steps exactly. All methods share post-hoc best-observed domain references, which need not be certified optima. The current reference views contain 22 MCND, 13 VCP, and 19 RCPSP methods. Appendix B specifies normalization, contributors, and capture boundaries.

Solved models means validated model completions: in-budget, non-timeout solver-optimal returns with independently validated original-feasible witnesses. This measures completion of the selected formulation, including restricted PATH and generated models. Original-problem optimality alone would not capture restricted methods' budgeted role; feasibility alone does not distinguish many experts (Appendix B.1). We therefore interpret completion jointly with original-space NPI and valid coverage, without equating it to original-problem global optimality. Normal timeouts retain eligible trajectories; genuine failures follow Appendix B.

Curves show mean prefix NPI, $N^{-1} \sum_i t^{-1} \int_0^t \ell_{i,m}(u)\, du$, and cumulative completions. Log10 axes start at 0.1 seconds for MCND/RCPSP and 0.01 for VCP; integration always starts at zero. We report domain-specific descriptive aggregates without pooling horizons or claiming significance or multi-seed stability.

5.3. Acquisition cost and computing setup

The RAW baselines generate modeling kernels inside an executable harness; RAW Expert also receives modeling guidance. Multiple acquisition rounds and compatibility handling produced the retained packages. Model/reasoning configurations, output allowances, reusable assets, and repair histories differ from Full. These are system-level comparisons under common domain program horizons.

LLM acquisition and solver execution are separate phases: accepted decisions or code are frozen before execution. Tokens sum provider-reported input and output for these final artifacts over all domain instances; tables report the total divided by that denominator. Cached input and reasoning

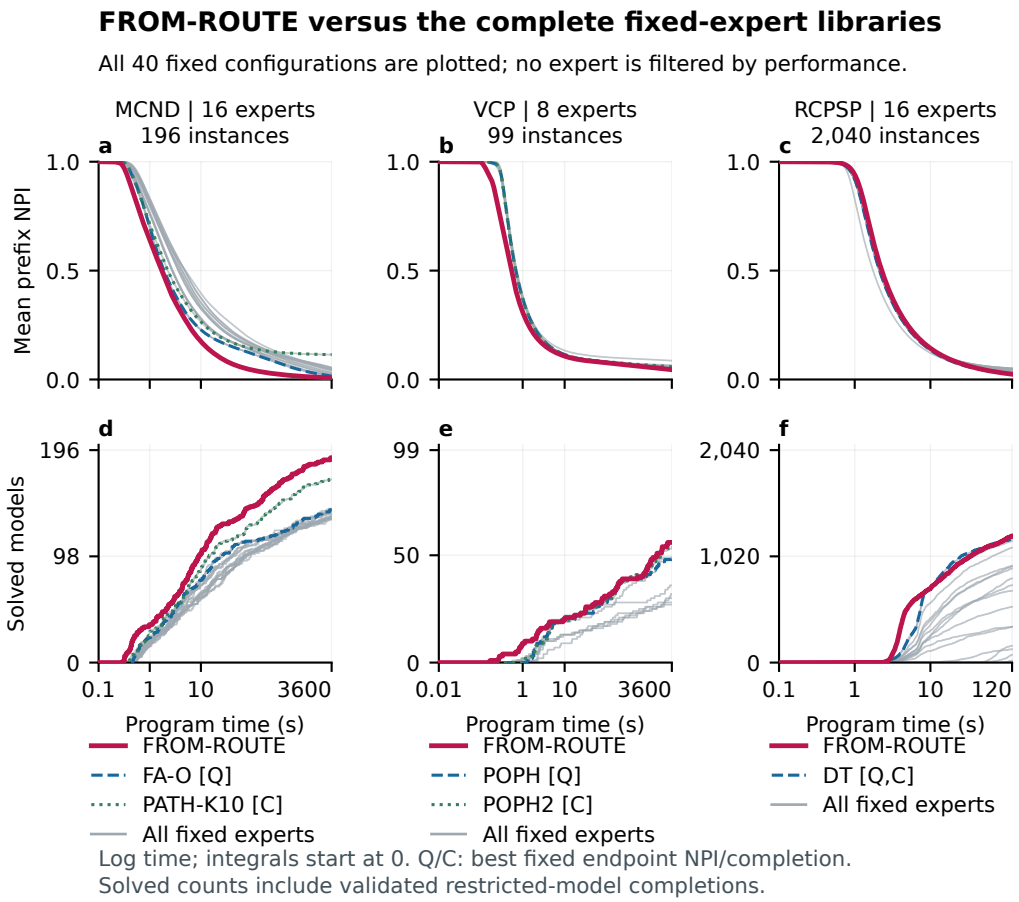

Figure 3: Full versus all fixed experts on 196 MCND, 99 VCP, and 2,040 RCPSP instances. Panels a–c show mean prefix NPI (lower is better); d–f show validated model completions (higher is better), including restricted PATH. Gray traces retain every fixed configuration; highlighted Q/C identify the best fixed endpoint for each metric. Each trace uses one method across the entire domain, without an instance-wise or time-wise oracle. Time is logarithmic; NPI is linear on [0,1], with integration from zero.

subsets are not double-counted. This is system-level LLM acquisition cost, not solver compute or end-to-end latency (Appendix H).

MCND Full, fixed experts, RAW, and ablation main batches use a dispatch cap of 30, a 3,600-second program budget, and one solver thread. VCP protects memory headroom through controlled dispatch and reruns. Appendix B summarizes the protocol and execution settings; Appendix G records VCP memory policies.

6. Results

6.1. Formulation choice and solver performance

Figure 3 compares Full against **all 16/8/16 fixed experts** on each entire domain set. Table 3 reports the retrospectively strongest fixed endpoint for each metric. Full achieves valid coverage on all 2,335 instances.

Table convention. Bold marks within-domain best reported values, including ties, not statistical significance. Arrows give metric direction. Diagnostic counts and incomplete acquisition totals are not ranked. Main-table NPI values use four significant digits; full precision is retained in the appendices.

Table 3: Full, best fixed endpoints (Q: lowest NPI; C: most completions), and MCND rule Top-1 replay. Appendices E/G provide details.

Domain	Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
MCND	Full	**0.006719**	**189/196**	**196/196**
MCND	Rule Top-1 replay	0.03005	185/196	193/196
MCND	FA-O [Q]	0.01564	141/196	**196/196**
MCND	PATH-K10 [C]	0.1141	169/196	177/196
VCP	Full	**0.04485**	**56/99**	**99/99**
VCP	POPH [Q]	0.05252	49/99	**99/99**
VCP	POPH2 [C]	0.05819	54/99	**99/99**
RCPSP	Full	**0.02423**	**1,216/2,040**	**2,040/2,040**
RCPSP	DT [Q,C]	0.02514	1,189/2,040	**2,040/2,040**

Table 4: Knowledge configurations and retained acquisition tokens per instance. All rows have full valid coverage; tokens cover the frozen decisions used by each condition.

Domain	Condition	Mean NPI ↓	Model completions ↑	Tokens/instance ↓
MCND	Full	**0.006719**	**189/196**	2,540.94
MCND	No RAG	0.008822	188/196	**1,986.03**
MCND	Generic RAG	0.008827	188/196	2,542.12
VCP	Full	**0.04485**	**56/99**	5,741.70
VCP	No RAG	0.05403	39/99	**1,925.76**
VCP	Generic RAG	0.05236	37/99	2,381.59
RCPSP	Full	**0.02423**	**1,216/2,040**	5,576.20
RCPSP	No RAG	0.03217	907/2,040	**2,871.20**
RCPSP	Generic RAG	0.02971	982/2,040	3,381.56

Full reduces mean NPI by 57.0%, 14.6%, and 3.6% relative to the best fixed NPI expert in MCND, VCP, and RCPSP. It records 20, 2, and 27 more completions than the strongest fixed completion comparator. The NPI percentages describe an integrated quality metric, not equal percentage reductions in final objective or completion time. These aggregate gains need not hold for every instance or time point. Reading both metrics matters: restricted formulations can complete despite worse original-space solution quality. Appendices E/G provide individual expert legends under the same references.

For MCND, rule Top-1 replay selects each instance's top-ranked expert using the frozen non-LLM recommender and retrieves the corresponding result from the full fixed-expert matrix. Full achieves lower NPI and four more model completions (Appendix E.6).

6.2. Knowledge provision, organization, and token use

Full has the lowest observed NPI and most completions among knowledge conditions in every domain (Table 4). MCND Full, No RAG, and Generic RAG select PATH-K10 first on 177, 192, and 196 instances, respectively. The contexts are associated with different choices, without identifying why each decision changed.

MCND Full uses similar tokens to Generic RAG and more than No RAG; VCP/RCPSP Full uses more than both variants. This compares context packages for the same Analyzer and LLM decision interface, with domain-dependent token trade-offs. Card content, membership, organization, and length change together; VCP/RCPSP also add ordering priors. The observed advantage concerns the adopted context package rather than its individual ingredients. Appendices F/G provide the curves and computing settings.

Table 5: System-level modeling-kernel generation comparison. Tokens cover retained acquisition calls; model, inference, assets, and acquisition workflows differ.

Domain	System	Mean NPI ↓	Model completions ↑	Valid coverage ↑	Tokens/ instance ↓
MCND	Full	**0.006719**	**189/196**	**196/196**	**2,540.94**
MCND	RAW Direct	0.03888	134/196	**196/196**	56,549.38
MCND	RAW Expert	0.02825	135/196	**196/196**	55,095.62
VCP	Full	**0.04485**	**56/99**	**99/99**	**5,741.70**
VCP	RAW Direct	0.2786	50/99	79/99	37,592.75
VCP	RAW Expert	0.3911	51/99	79/99	53,832.65

6.3. RAW SYSTEMS: SOLUTION OUTCOMES AND ACQUISITION TOKENS

Full has lower NPI, more completions, and fewer retained-call tokens than either RAW system in MCND/VCP (Table 5). MCND coverage is complete for all three; VCP RAW coverage is 79/99 versus Full's 99/99. Failures remain in all denominators.

Every final-artifact call has recorded total usage, including artifacts whose solver execution later times out. RAW acquisition also encountered failed generations and further attempts, whose extra token costs are excluded from the main comparison. The observed advantage therefore already holds before adding these RAW acquisition overheads. Appendix H defines the accounting scope and system configurations.

6.4. MCND-SPECIFIC RELAY EXPLORATION

No relay re-executes Full's decisions with restricted-completion continuation disabled. It records 188/196 model completions and NPI 0.01728865; Full records 189 and 0.00671945. Both use a main-batch dispatch cap of 30 (Appendix F).

7. DISCUSSION AND CONCLUSION

FROM-ROUTE combines instance evidence, mechanism cards, and domain-specific recommendation priors to support LLM selection among reusable MIP formulation experts. The selected portfolio is executed under the domain's budget and continuation policy.

The evaluated system improves budgeted solution performance on the studied collections. MCND provides the primary mechanism study; VCP and RCPSP demonstrate applicability through domain-specific adapters.

AI USE STATEMENT

Generative AI tools assisted with literature retrieval and summarization, translation and language editing, and drafting manuscript sections. They also supported method and experiment development, code implementation, result analysis, and figure preparation. The authors reviewed and revised the manuscript, supervised final typesetting, and take responsibility for all submitted text, code, figures, and results.

REPRODUCIBILITY STATEMENT

The paper and appendices specify the evaluated formulations, decision mechanisms, and scoring protocol (Appendices A, B, D, G, and I). The supplementary material provides selected code modules with minimal runnable examples, final aggregate NPI and solved-curve data, and scripts for redrawing the curves. Raw solver outputs, per-instance trajectories, and complete experimental audit records are retained internally.

REFERENCES

Ali AhmadiTeshnizi, Wenzhi Gao, and Madeleine Udell. Optimus: Scalable optimization modeling with (mi)lp solvers and large language models, 2024. URL https://arxiv.org/abs/2402.10172v1. Version 1.

Tasnim Ahmed and Salimur Choudhury. OPT2CODE: A retrieval-augmented framework for solving linear programming problems. *Natural Language Processing Journal*, 13:100185, 2025. doi: 10.1016/j.nlp.2025.100185. URL https://www.sciencedirect.com/science/article/pii/S2949719125000615.

Christian Artigues. On the strength of time-indexed formulations for the resource-constrained project scheduling problem. *Operations Research Letters*, 45(2):154–159, 2017. doi: 10.1016/j.orl.2017.02.001. URL https://www.sciencedirect.com/science/article/abs/pii/S016763771730072X.

Christian Artigues, Peter Brucker, Sigrid Knust, Oumar Koné, Pierre Lopez, and Marcel Mongeau. A note on "event-based MILP models for resource-constrained project scheduling problems". *Computers & Operations Research*, 40(4):1060–1063, 2013. doi: 10.1016/j.cor.2012.10.018. URL https://www.sciencedirect.com/science/article/pii/S0305054812002304.

Timo Berthold. Measuring the impact of primal heuristics. *Operations Research Letters*, 41(6): 611–614, 2013. doi: 10.1016/j.orl.2013.08.007. URL https://doi.org/10.1016/j.orl.2013.08.007.

Lucio Bianco and Massimiliano Caramia. The resource constrained project scheduling problem: A theoretical comparison between a recent formulation and the main time indexed linear programming based approaches. *RAIRO - Operations Research*, 51(3):519–532, 2017. doi: 10.1051/ro/2016036. URL https://www.numdam.org/item/10.1051/ro/2016036.pdf.

Manoel B. Campêlo, Victor A. Campos, and Ricardo C. Corrêa. On the asymmetric representatives formulation for the vertex coloring problem. *Discrete Applied Mathematics*, 156(7):1097–1111, 2008. doi: 10.1016/j.dam.2007.05.058. URL https://www.sciencedirect.com/science/article/pii/S0166218X07002922.

Sophie Demassey, Christian Artigues, and Philippe Michelon. Constraint-propagation-based cutting planes: An application to the resource-constrained project scheduling problem. *INFORMS Journal on Computing*, 17(1):52–65, 2005. doi: 10.1287/ijoc.1030.0043. URL https://sofdem.github.io/art/demassey05ijc.pdf.

Mario Gnägi, Tom Rihm, Adrian Zimmermann, and Norbert Trautmann. Two continuous-time assignment-based models for the multi-mode resource-constrained project scheduling problem. *Computers & Industrial Engineering*, 129:346–353, 2019. doi: 10.1016/j.cie.2019.01.033. URL https://www.sciencedirect.com/science/article/pii/S0360835219300373.

Chenyu Huang, Zhengyang Tang, Shixi Hu, Ruoqing Jiang, Xin Zheng, Dongdong Ge, Benyou Wang, and Zizhuo Wang. ORLM: A customizable framework in training large models for automated optimization modeling. *Operations Research*, 73(6):2986–3009, 2025. doi: 10.1287/opre.2024.1233. URL https://doi.org/10.1287/opre.2024.1233.

Adalat Jabrayilov and Petra Mutzel. New integer linear programming models for the vertex coloring problem. In Michael A. Bender, Martín Farach-Colton, and Miguel A. Mosteiro (eds.), *LATIN 2018: Theoretical Informatics*, volume 10807 of *Lecture Notes in Computer Science*, pp. 640–652, Cham, 2018. Springer. doi: 10.1007/978-3-319-77404-6_47. URL https://doi.org/10.1007/978-3-319-77404-6_47.

Adalat Jabrayilov and Petra Mutzel. Strengthened partial-ordering based ILP models for the vertex coloring problem, 2022. URL https://arxiv.org/abs/2206.13678v1. Version 1, 28 June 2022.

Ahmad Kazemi, Pierre Le Bodic, Andreas T. Ernst, and Mohan Krishnamoorthy. New commodity representations for multicommodity network flow problems: An application to the fixed-charge network design problem, 2021. URL https://arxiv.org/abs/2101.03707v1. Version 1, 11 January 2021.

Nicklas Klein. Integer programming for multi-mode resource-constrained project scheduling. *Annals of Operations Research*, 2025. doi: 10.1007/s10479-025-06572-1. URL https://link.springer.com/article/10.1007/s10479-025-06572-1.

Nicklas Klein, Mario Gnägi, and Norbert Trautmann. Mixed-integer linear programming for project scheduling under various resource constraints. *European Journal of Operational Research*, 319 (1):79–88, 2024. doi: 10.1016/j.ejor.2024.06.036. URL https://www.sciencedirect.com/science/article/pii/S0377221724005162.

Oumar Koné, Christian Artigues, Pierre Lopez, and Marcel Mongeau. Event-based MILP models for resource-constrained project scheduling problems. *Computers & Operations Research*, 38(1): 3–13, 2011. doi: 10.1016/j.cor.2009.12.011. URL https://www.math.univ-toulouse.fr/~mongeau/COR-authors.pdf.

Georgios M. Kopanos, Thomas S. Kyriakidis, and Michael C. Georgiadis. New continuous-time and discrete-time mathematical formulations for resource-constrained project scheduling problems. *Computers & Chemical Engineering*, 68:96–106, 2014. doi: 10.1016/j.compchemeng.2014.05.009. URL https://www.sciencedirect.com/science/article/abs/pii/S0098135414001525.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.nips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Kuo Liang, Yuhang Lu, Jianming Mao, Shuyi Sun, Chunwei Yang, Congcong Zeng, Xiao Jin, Hanzhang Qin, Ruihao Zhu, and Chung-Piaw Teo. Large-scale optimization model autoformulation: Harnessing LLM flexibility via structured workflow, 2026. URL https://arxiv.org/abs/2601.09635v3. Version 3, 31 January 2026.

Rindranirina Ramamonjison, Timothy Yu, Raymond Li, Haley Li, Giuseppe Carenini, Bissan Ghaddar, Shiqi He, Mahdi Mostajabdaveh, Amin Banitalebi-Dehkordi, Zirui Zhou, and Yong Zhang. Nl4opt competition: Formulating optimization problems based on their natural language descriptions. In *Proceedings of the NeurIPS 2022 Competitions Track*, volume 220, pp. 189–203. PMLR, 2022. URL https://proceedings.mlr.press/v220/ramamonjison23a.html.

John R. Rice. The algorithm selection problem. In *Advances in Computers*, volume 15, pp. 65–118. Academic Press, 1976. doi: 10.1016/S0065-2458(08)60520-3. URL https://www.sciencedirect.com/science/article/pii/S0065245808605203.

Henry Robbins, Connor Lawless, Madeleine Udell, and Ellen Vitercik. FLARE: Verifying MILP reformulations with LLM-based theorem proving, 2026. URL https://arxiv.org/abs/2608.25220v1. Version 1, 25 August 2026.

David Sayah. Continuous-time formulations for multi-mode project scheduling. *Computers & Operations Research*, 152:106147, 2023. doi: 10.1016/j.cor.2023.106147. URL https://arxiv.org/abs/2301.04700v2. Version 2.

Alexander Tesch. A polyhedral study of event-based models for the resource-constrained project scheduling problem. *Journal of Scheduling*, 23(2):233–251, 2020. doi: 10.1007/s10951-020-00647-6. URL https://link.springer.com/article/10.1007/s10951-020-00647-6.

Willem-Jan van Hoeve. Graph coloring with decision diagrams. *Mathematical Programming*, 192 (1–2):631–674, 2022. doi: 10.1007/s10107-021-01662-x. URL https://www.andrew.cmu.edu/user/vanhoeve/papers/ColorDD_MathProg.pdf. First published online in 2021.

Jintao Xu, Yingzheng Ma, Jiong Dong, Yongzhi Qi, and Jianshen Zhang. Large language model for operations research formulation selection in multi-warehouse inventory allocation, 2026. URL https://arxiv.org/abs/2607.25956v1. Version 1, 28 July 2026.

Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. SATzilla: Portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606, 2008. doi: 10.1613/jair.2490. URL https://arxiv.org/abs/1111.2249v1.

Haofeng Yuan, Jianing Peng, Jieyi Bi, Ni Zhang, Shiji Song, and Zhiguang Cao. FormuEvo: LLM-guided evolution for discovering solver-efficient mixed-integer programming formulations, 2026. URL https://arxiv.org/abs/2608.23353v1. Version 1, 24 August 2026.

Haotian Zhai, Connor Lawless, Ellen Vitercik, and Liu Leqi. EquivaMap: Leveraging LLMs for automatic equivalence checking of optimization formulations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74288–74305. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhai25a.html.

Ruiqing Zhao, Fengzhi Li, Yuan Zuo, Rui Liu, Yansong Liu, Yunfei Ma, Fanyu Meng, and Junlan Feng. Strategy-aware optimization modeling with reasoning LLMs, 2026. URL https://arxiv.org/abs/2605.02545v1. Version 1, 4 May 2026.

APPENDIX A. DOMAIN FORMULATIONS AND ORIGINAL-SOLUTION RECOVERY

This appendix defines the three original problems, implemented formulation variants, and original-solution recovery. Empirical comparisons appear in Appendices E/G; source and implementation records are indexed in A.4.

The evaluated library has 16 MCND, eight VCP, and 16 RCPSP configurations. The historical OBDD formulation is retained for mathematical reference but excluded from the VCP experiment. Actual decision interfaces are specified in Appendices D/G.

A.0. READING THE EXPERT LIBRARY

The library contains **40 evaluated configurations: 16 MCND, 8 VCP, and 16 RCPSP**. Each configuration fixes a representation, construction parameters, and recovery procedure; several share a published family. The map below connects experimental IDs to modeling principles, with equations and recovery conditions in the corresponding sections.

Domain and family	Evaluated configurations	Modeling decision and mechanism	Detailed definition
MCND: disaggregated flow	DA (1)	Retain a flow for every commodity and arc; enforce demand-linked arc capacity.	A.1.1, M1
MCND: endpoint aggregation	FA-O, FA-D (2)	Combine commodities with a common origin or destination; recover their individual flows afterward.	A.1.2, M2
MCND: partial aggregation	PAi/PAe Œ O/D Œ K1/K5/K10 (12)	Keep commodity distinctions on selected critical arcs while aggregating elsewhere; PAi and PAe encode the coupling differently. The K parameter changes the critical-arc construction, not the available arc set.	A.1.3–A.1.5, M3–M5
MCND: path variables	PATH-K10 (1)	Allocate splittable demand over a finite pool of up to ten paths per commodity. This is a restricted representation with valid original-flow recovery.	A.1.6, M7
VCP: assignments	ASS (1)	Assign one color to each vertex and activate used colors; adjacent vertices cannot share a color.	A.3.2, A4

Domain and family	Evaluated configurations	Modeling decision and mechanism	Detailed definition
VCP: partial orders and hybrids	POP, POP1, POP2, POPH, POPH1, POPH2 (6)	Encode color positions through order thresholds; retain or eliminate assignment variables and add the declared strengthening rows.	A.3.2, A5 and following equations
VCP: representatives	REP (1)	Assign vertices to ordered, mutually compatible color-class representatives, avoiding an explicit color index for each class.	A.3.3, A6
RCPSP: discrete time	DT, DDT, SDDT, OODDT, BIA-CORRECTED (5)	Encode starts, cumulative states, occupied slots, or completion fractions on a complete integer grid.	A.2.2–A.2.4
RCPSP: events	OOE, SEE-CORRECTED, RSEE, IEE (4)	Represent activity transitions at ordered event times, with duration, precedence, and resource coupling.	A.2.4
RCPSP: resource flow, units, and forbidden sets	FCT, FCT-ALT, UNIT, MFS (4)	Encode capacity through resource transfers, unit assignments, or disjunctions ruling out overloaded activity sets.	A.2.4
RCPSP: sequencing and overlap	SEQCT, KOP-CT1, KOP-CT2 (3)	Couple continuous start times with ordering/overlap decisions and capacity checks at activity starts.	A.2.4

The representations change construction work through commodity-arc pairs and masks in MCND, palette variables or compatible vertex pairs in VCP, and time windows, events, resources, or forbidden sets in RCPSP. These quantities inform the Analyzer but do not establish a runtime ranking; the full-matrix experiments measure each fixed configuration on every adopted instance.

The principal mathematical lineages are partial commodity aggregation (Kazemi et al., 2021), coloring assignments/partial orders and representatives (Jabrayilov & Mutzel, 2022; Campêlo et al., 2008), and scheduling time, event, and resource representations (Koné et al., 2011; Tesch, 2020; Klein, 2025). The RCPSP implementation specializes any multi-mode source to the stated single-mode renewable-resource problem. Historical OBDD is described only for provenance in A.3; it is not one of the 40 evaluated experts.

A.1. SPLITTABLE FIXED-CHARGE MULTICOMMODITY NETWORK DESIGN

This section specifies the MCND problem, the mathematical families behind the current 16 fixed experts, and their original-solution recovery contracts. The formal scope is the homogeneous-cost Canad/CommaLAB benchmark. The general input schema also supports commodity-specific costs and allowed arcs; that broader schema is not a license to apply unrestricted endpoint aggregation. The artificial network in A.1.8 illustrates the definitions. Classical DA/FA/PA terminology follows (Kazemi et al., 2021); the implementation distinctions are recorded in the MCND source audit and literature evidence.

A.1.1. ORIGINAL PROBLEM AND WITNESS

Let $G = (V, A)$ be a directed network and $\mathcal{K}$ the commodity set. An arc a has capacity $u_a > 0$, fixed opening cost $f_a \geq 0$, and unit flow cost $c_a \geq 0$. Commodity k has distinct origin o_k, destination t_k, and demand $d_k > 0$. Arc IDs are retained even if two arcs have the same endpoints; the input schema excludes self-loops. Write $\delta^+(v), \delta^-(v)$ for the outgoing/incoming arcs and $b_v^k = d_k(\mathbf{1}[v = o_k] - \mathbf{1}[v = t_k])$. The original witness consists of binary openings y_a and nonnegative, continuously divisible flows x_a^k, measured in demand units:

$$
\begin{aligned}
\min_{x,y} \quad & F_I(x,y) = \sum_{a\in A} f_a y_a + \sum_{k\in\mathcal{K}} \sum_{a\in A} c_a x_a^k, \\
\sum_{a\in\delta^+(v)} x_a^k - \sum_{a\in\delta^-(v)} x_a^k &= b_v^k && (k \in \mathcal{K},\ v \in V), \\
\sum_{k\in\mathcal{K}} x_a^k &\leq u_a y_a && (a \in A), \\
0 \leq x_a^k &\leq d_k y_a && (k \in \mathcal{K},\ a \in A), \\
y_a &\in \{0,1\} && (a \in A).
\end{aligned}
\tag{M1}
$$

M1 is the disaggregated arc-flow model (DA) and the mathematical contract checked by the current original-variable validator. There is no unsplittable-path constraint, design budget, or additional route restriction in this benchmark. In the broader schema, replace c_a by c_a^k and impose $x_a^k = 0$ outside commodity k's allowed arcs. Such extensions require separate aggregation applicability checks.

The demand-linking inequalities are part of the accepted witness contract. They strengthen the LP relaxation, where $0 \leq y_a \leq 1$. If a classical flow formulation omits them, an arbitrary feasible solution may carry a circulation with $x_a^k > d_k$. For nonnegative costs, removing commodity circulations leaves demands feasible, does not increase cost, and yields $x_a^k \leq d_k$. Thus M1 preserves that classical problem's integer optimum, but its feasible set is not literally all circulation-bearing flows. This distinction also matters for aggregate recovery below.

Instance scope. The Canad/CommaLAB corpus contains 196 eligible instances, comprising C=31, C+=12, and R=153. Nine documented-infeasible R instances from the 205 raw instances are excluded and retained in provenance. The normalized inputs have no commodity-cost overrides or forbidden arcs. Unlike the cited paper's 193-instance experiment, this study retains all three additional C+ instances. The manifest labels instances as test, but the corpus also informed method development (Appendix D.3).

A.1.2. FULL ENDPOINT AGGREGATION

Partition $\mathcal{K}$ into groups $\mathcal{G}$ sharing an origin. For $g \in \mathcal{G}$, define $D_g = \sum_{k\in g} d_k$, $B_v^g = \sum_{k\in g} b_v^k$, and group flow X_a^g. The FA-O formulation is

$$
\begin{aligned}
\min_{X,y} \quad & \sum_a f_a y_a + \sum_g \sum_a c_a X_a^g, \\
\sum_{a\in\delta^+(v)} X_a^g - \sum_{a\in\delta^-(v)} X_a^g &= B_v^g && (g\in\mathcal{G},\ v\in V),\\
\sum_g X_a^g &\le u_a y_a && (a\in A),\\
0 \le X_a^g &\le D_g y_a && (g\in\mathcal{G},\ a\in A),\\
y_a &\in \{0,1\} && (a\in A).
\end{aligned}
\tag{M2}
$$

FA-D groups by destination. It can be described by reversing every arc and exchanging each commodity's origin and destination, applying M2, then mapping flows back to the unchanged original arc IDs. Reversal changes neither capacities nor costs. Singleton endpoint groups are allowed. DA uses $|\mathcal{K}||A|$ continuous flow slots and FA uses $|\mathcal{G}||A|$, before presolve; both retain $|A|$ opening binaries. These counts do not predict runtime.

Scope of equivalence. Every M1 point maps to M2 through $X_a^g = \sum_{k\in g} x_a^k$. Conversely, for integer y, a common-origin group flow decomposes into paths delivering each terminal's required amount and residual cycles. Split the terminal deliveries among commodities ending there and discard cycles. The recovered commodity loads do not exceed X^g, satisfy M1, and have no greater cost. The same argument applies after reversal to common-destination groups. Hence M1 and M2 have equal integer optimal values under the stated homogeneous-cost assumptions; this is not a bijection between all internal flow vectors. At a cyclic incumbent, recovery need not preserve every arc load or the model's scalar objective exactly.

Within a group, differing costs, allowed arcs, capacity consumption, or route constraints can invalidate this reasoning. The fixed endpoint grouping checks exact equality of the cost/eligibility signatures and rejects incompatible groups; the formal Canad instance view is homogeneous. Signature splitting and generated groups are not additional entries in the 16-expert matrix. In particular, this appendix does not establish that every broader-schema input is supported by every PA implementation.

A.1.3. Partial aggregation and PAi

Use the common-origin orientation; destination variants use the same reversal as FA-D. For each commodity choose a critical-arc mask $C_k \subseteq A$. On arc a, the partition $\mathcal{P}_{ga}$ consists of singleton blocks $\{k\}$ for $k \in g$ with $a \in C_k$, followed by the nonempty remainder $R_{ga} = \{k \in g : a \notin C_k\}$. The remainder is an aggregate-role block even if it happens to contain one commodity. Let $q_{gaS} \ge 0$ denote the flow on arc a with label $S \in \mathcal{P}_{ga}$ and $D_S = \sum_{k\in S} d_k$. The common PA core is

$$
\begin{aligned}
\min_{q,y} \quad & \sum_a f_a y_a + \sum_g \sum_a \sum_{S\in\mathcal{P}_{ga}} c_a q_{gaS}, \\
\sum_{a\in\delta^+(v)} \sum_{S\in\mathcal{P}_{ga}} q_{gaS} - \sum_{a\in\delta^-(v)} \sum_{S\in\mathcal{P}_{ga}} q_{gaS} &= B_v^g && (g\in\mathcal{G},\ v\in V),\\
\sum_g \sum_{S\in\mathcal{P}_{ga}} q_{gaS} &\le u_a y_a && (a\in A),\\
0 \le q_{gaS} &\le D_S y_a && \left(\begin{smallmatrix} g\in\mathcal{G},\ a\in A \\ S\in\mathcal{P}_{ga}\end{smallmatrix}\right),\\
y_a &\in \{0,1\} && (a\in A).
\end{aligned}
\tag{M3}
$$

For $k \in g$, define $Q_{gv}^{+,\ni k}$ as the total outgoing q whose label contains k, and $Q_{gv}^{-,\ni k}$ analogously on incoming arcs. Define $Q_{gv}^{+,=k}, Q_{gv}^{-,=k}$ using only label $\{k\}$, including a singleton remainder. PAi adds, at every node,

$$
\begin{aligned}
&Q^{+,\ni k}_{gv} - Q^{-,=k}_{gv} \geq b^k_v, \\
&Q^{+,=k}_{gv} - Q^{-,\ni k}_{gv} \leq b^k_v \qquad (g \in \mathcal{G},\ k \in g,\ v \in V).
\end{aligned}
\tag{M4}
$$

These inequalities constrain label transitions without imposing independent commodity conservation inside every changing aggregate block. Treating each aggregate label as a globally conserved commodity would define a different, potentially overconstrained model. When every block is a singleton, M4 reduces to ordinary commodity balance. The fixed library contains PAi and PAe; the unstrengthened M3 core is explanatory, not a seventeenth expert.

A.1.4. PAE EQUALITY ENCODING

PAe uses M3 with a local equality construction instead of M4. At node v, let H_{gv} contain commodities having a critical arc incident to v. If H_{gv} is empty no gadget is added. Otherwise define a middle partition $\mathcal{M}_{gv}$ of singleton commodities in H_{gv} and the nonempty remainder $g \setminus H_{gv}$. Let $\mathcal{R}^-_{gv}$ and $\mathcal{R}^+_{gv}$ be the distinct nonempty remainder labels on incoming and outgoing arcs. Set

$$
A^-_{gvR} = \sum_{\substack{a \in \delta^-(v) \\ R_{ga} = R}} q_{gaR}, \qquad A^+_{gvR} = \sum_{\substack{a \in \delta^+(v) \\ R_{ga} = R}} q_{gaR}, \qquad D^{\pm}_{gvM} = \sum_{k \in M} \sum_{\substack{a \in \delta^{\pm}(v) \\ a \in C_k}} q_{ga\{k\}}.
$$

Introduce nonnegative transfer variables z^-_{gvRM} for $R \in \mathcal{R}^-_{gv}, M \in \mathcal{M}_{gv}$ with $R \cap M \neq \varnothing$, and z^+_{gvMR} for the corresponding outgoing intersections. Only existing transfer variables enter the following sums:

$$
\begin{aligned}
\sum_M z^-_{gvRM} &= A^-_{gvR} && (R \in \mathcal{R}^-_{gv}), \\
\sum_M z^+_{gvMR} &= A^+_{gvR} && (R \in \mathcal{R}^+_{gv}), \\
D^+_{gvM} - D^-_{gvM} + \sum_R z^+_{gvMR} - \sum_R z^-_{gvRM} &= \sum_{k \in M} b^k_v && (M \in \mathcal{M}_{gv}), \\
z^-_{gvRM}, z^+_{gvMR} &\geq 0 && \text{for each defined transfer.}
\end{aligned}
\tag{M5}
$$

The z variables have zero objective coefficient and do not consume physical arc capacity. They enforce consistency of local splitting and joining; physical capacity remains in M3. This is the project's explicit equality-gadget encoding, audited against its source and the PAe construction in the cited paper. Its extra variables and rows must be counted separately from the physical flow slots $\sum_{g,a} |\mathcal{P}_{ga}|$. Neither the name PAe nor a smaller slot count is evidence of a runtime advantage.

Why the integer problem is retained. A DA flow gives $q_{gaS} = \sum_{k \in S} x^k_a$, satisfying M3/M4. In M5, allocate the incoming and outgoing aggregate amounts by the underlying commodities' middle blocks; this supplies feasible transfers. Conversely, every PA point gives an FA point $X^g_a = \sum_S q_{gaS}$. At integer y, the endpoint decomposition of A.1.2 then recovers an M1 solution of no greater cost. Thus, for these homogeneous endpoint constructions with the complete physical arc set,

$$
\mathrm{OPT}(\mathrm{DA}) = \mathrm{OPT}(\mathrm{FA}) = \mathrm{OPT}(\mathrm{PAi}) = \mathrm{OPT}(\mathrm{PAe}). \tag{M6}
$$

M6 is an integer optimal-value argument under explicit assumptions, not a claim that every PA label survives recovery or that all four LP relaxations coincide. The classical paper studies relaxation strength for its specified constructions; those comparisons do not rank finite-budget performance and do not automatically extend to arbitrary generated masks or semantic extensions. The proof above does not certify floating-point code execution on every input.

For the paper's specified formulations and matching endpoint/partial partitions, its Theorems 1–4 give $z_{\mathrm{LP}}(\mathrm{DA}) \geq z_{\mathrm{LP}}(\mathrm{PAe}) \geq z_{\mathrm{LP}}(\mathrm{PAi}) \geq z_{\mathrm{LP}}(\mathrm{PA}) \geq z_{\mathrm{LP}}(\mathrm{FA})$, where a larger lower bound

is stronger for minimization. These inequalities need not be strict on a given instance. This is a literature result with matched constructions, not a measured ordering of the current packages or a comparison between O and D groupings. The evidence card also records an inconsistent disjointness condition in the preprint's PAe notation: distinct remainder labels may overlap, as allowed in M5.

A.1.5. CRITICAL ARCS AND THE 16 CONFIGURATIONS

The fixed PA parameter $\kappa \in \{1, 5, 10\}$ selects up to κ loopless paths for each commodity in the relevant orientation, using the deterministic proxy arc weight $w_a = c_a + f_a/u_a$. The mask C_k is their arc union. All physical arcs remain in M3; a commodity may use a route outside those proxy paths after recovery. If fewer paths exist, the available paths are used. The proxy distributes fixed cost by capacity for path ranking; it is not the true shared-design objective and is not used to certify optimality. Source tie-breaking and arc IDs determine the exact frozen masks.

Fixed ID	Family	Grouping	Mask parameter	Mathematical scope
`DA`	M1	commodity	—	original problem
`FA-O`	M2	common origin	—	homogeneous endpoint aggregation
`FA-D`	M2, reversed	common destination	—	homogeneous endpoint aggregation
`PAi-O-K1`	M3+M4	common origin	$\kappa = 1$	complete arc set
`PAi-O-K5`	M3+M4	common origin	$\kappa = 5$	complete arc set
`PAi-O-K10`	M3+M4	common origin	$\kappa = 10$	complete arc set
`PAi-D-K1`	M3+M4, reversed	common destination	$\kappa = 1$	complete arc set
`PAi-D-K5`	M3+M4, reversed	common destination	$\kappa = 5$	complete arc set
`PAi-D-K10`	M3+M4, reversed	common destination	$\kappa = 10$	complete arc set
`PAe-O-K1`	M3+M5	common origin	$\kappa = 1$	complete arc set
`PAe-O-K5`	M3+M5	common origin	$\kappa = 5$	complete arc set
`PAe-O-K10`	M3+M5	common origin	$\kappa = 10$	complete arc set
`PAe-D-K1`	M3+M5, reversed	common destination	$\kappa = 1$	complete arc set
`PAe-D-K5`	M3+M5, reversed	common destination	$\kappa = 5$	complete arc set
`PAe-D-K10`	M3+M5, reversed	common destination	$\kappa = 10$	complete arc set
`PATH-K10`	M7	commodity paths	up to 10 paths	restricted path pool

The library has $1 + 2 + 2 \times 2 \times 3 + 1 = 16$ configurations. Historical Benders entries, dynamic grouping, and generated masks are outside this evaluated library; results appear in Appendix E.

A.1.6. RESTRICTED PATH-FLOW EXPERT

For commodity k, let $\mathcal{Q}_k$ be the at-most-ten loopless paths selected by the same proxy ranking on its original allowed network. Let $h_{kp} \geq 0$ be the amount sent on path p, $\alpha_{ap} = \mathbf{1}[a \in p]$, and $c_p = \sum_a c_a \alpha_{ap}$. The implemented restricted master is

$$
\begin{aligned}
\min_{h,y} \quad & \sum_a f_a y_a + \sum_k \sum_{p \in \mathcal{Q}_k} c_p h_{kp}, \\
\sum_{p \in \mathcal{Q}_k} h_{kp} &= d_k && (k \in \mathcal{K}), \\
\sum_k \sum_{p \in \mathcal{Q}_k} \alpha_{ap} h_{kp} &\le u_a y_a && (a \in A), \\
\sum_{p \in \mathcal{Q}_k} \alpha_{ap} h_{kp} &\le d_k y_a && (k \in \mathcal{K},\ a \in A), \\
h_{kp} \ge 0, \qquad & y_a \in \{0,1\}.
\end{aligned}
\tag{M7}
$$

Recover $x_a^k = \sum_{p \in \mathcal{Q}_k} \alpha_{ap} h_{kp}$ with the same y. This preserves objective and M1 feasibility. However, a finite pool may exclude useful or necessary routes: restricted infeasibility does not prove MCND infeasibility, and restricted optimality does not prove original optimality. A lower bound on M7 need not be a lower bound on M1. The current PATH output records `optimality_scope=restricted_model` and does not publish its private dual/gap as an original-problem bound. A time-limited run with an incumbent remains time-limited.

Thus PA's κ changes where commodity labels are separated while retaining the full graph; PATH's κ removes all paths outside the pool. K-shortest preprocessing is shared machinery, not a shared coverage guarantee. The explicit arc-demand links in M7 match the implementation and strengthen its continuous relaxation even though simple paths and integer openings already imply them.

A.1.7. Recovery, validation, and claim boundaries

The current real-primal FA/PA online incumbent callbacks recover homogeneous endpoint flows deterministically, without solving a second optimization model within that callback. PA first sums all labels on each group/arc, and recovery uses that total; the internal label history is not a required output invariant. Common-destination recovery operates on the reversed instance and restores original IDs. Let F_e denote the internal model's fixed-plus-flow objective with the same opening vector (PAe transfers have zero cost). In exact arithmetic the callback's mathematical contract is

$$
\begin{aligned}
&(\widehat{x}, y) \in \mathcal{X}_{\mathrm{M1}}(I), \qquad \sum_{k \in g} \widehat{x}_a^k \le X_a^g \quad (g \in \mathcal{G},\ a \in A), \\
&F_I(\widehat{x}, y) \le F_e(\xi^{(e)}), \qquad X_a^g = \sum_{S \in \mathcal{P}_{ga}} q_{gaS} \quad \text{for PA.}
\end{aligned}
\tag{M8}
$$

Equality of objectives holds when no positive-cost flow is discarded; the acyclic example in A.1.8 preserves every arc load. DA already has original variables and PATH uses the direct linear map in A.1.6. A mathematically exact aggregate model can carry a higher-cost cyclic incumbent, so the evaluator always recomputes the recovered original objective rather than trusting the model's scalar incumbent value. A solver's optimality status requires the formulation's applicability and a compatible recovered objective/certificate before it can support an original-optimality claim.

Current terminal branch. The audited release still calls a recovery LP when extracting the terminal FA/PA solution after the main MIP. This is distinct from online M8: it enforces commodity balances and equality to each supplied group/arc total, with zero objective, rather than optimizing a new routing on the open network. PA additionally bounds each recovered commodity flow by d_k; FA's terminal LP does not impose that individual bound. A load-preserving LP may therefore fail to produce a witness accepted by M1 even when cycle-discarding recovery is possible. Only a successful, independently validated terminal witness may be credited, at its own ready time; earlier valid online witnesses are retained. Thus expert execution includes terminal recovery optimization; the recovery audit identifies the two branches.

The serialized witness contains the original arc-opening map and a commodity-by-arc flow map; omitted known entries mean zero, not unknown. Independent validation checks instance and ID consistency, opening bounds/integrality, nonnegativity, allowed arcs, commodity balances, shared

activated capacities, demand links, and the recomputed objective. The current numeric tolerances are 10^{-6} for feasibility and integrality, and $10^{-5}\max(1, |F_I|)$ for the reported objective mismatch. These witness tolerances are distinct from the reference-gap equality convention in Appendix B. Additional route restrictions cannot be presumed checked merely because a schema flag exists.

For PI/NPI, only recovered, durably persisted witnesses that pass the independent audit enter the step trajectory, at their original ready times. Recovery and persistence consume the execution budget; an internal aggregate incumbent is not yet a ready original solution. Model bounds and rejected or incomplete captures do not substitute for witnesses.

A.1.8. Illustrative representation and recovery

This example explains representation scope, rather than providing the complete mathematical definition of the MCND benchmark. Let a directed acyclic network have vertices $\{s, a, b, t_1, t_2\}$ and arcs $\{sa, sb, at_1, at_2, bt_1, bt_2\}$. Two splittable unit commodities share source s and have sinks t_1, t_2. Every arc has capacity two, fixed opening cost one, and unit flow cost one, independent of commodity identity. Sending both commodities through a opens three arcs and sends four arc-units of flow, giving objective $3 + 4 = 7$.

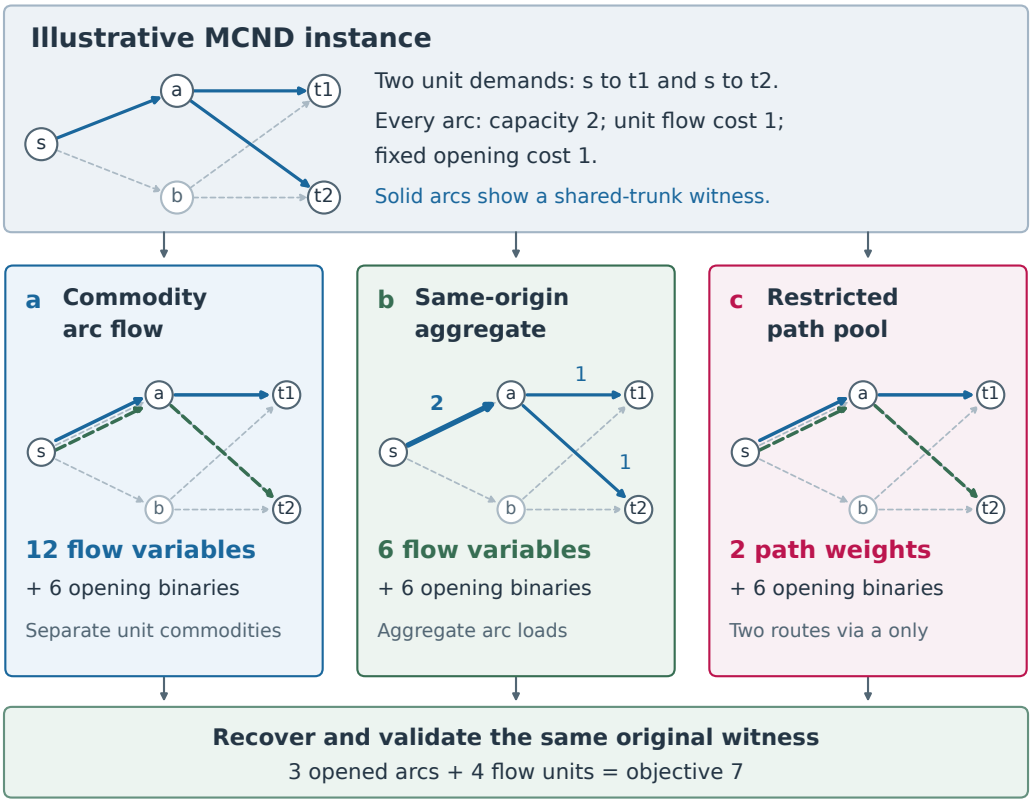

Figure A1: Three representations recover the same original witness in an illustrative network. Commodity arc flows and the same-origin aggregate cover this example's full problem under its homogeneous costs, splittable flows, and endpoint balances. The aggregate decomposes into commodity paths with unchanged arc loads and objective. The displayed two-path pool excludes routes through b and therefore covers a strict feasible subset. Panels a and c distinguish the two unit commodities or paths using blue solid and green dashed traces; panel b labels aggregate arc loads. Pale dashed arcs provide the original-network context, including routes through b that are absent from the displayed path pool. Variable-slot counts are before presolve and imply no speed or quality ranking.

The aggregate argument is constructive: split its source-to-sink flow by the terminal vertex and assign each part to that terminal's commodity. The graph is acyclic and its two sink balances are fixed, so this preserves the required unit demands. Equal per-arc commodity costs preserve the flow objective; the opening vector remains unchanged. A feasible route through b directly witnesses the path pool's incompleteness. None of these observations establishes equivalence for arbitrary aggregate experts, commodity-dependent costs, eligibility constraints, or additional side constraints. Artificial instance and checks; full figure record.

A.2. SINGLE-MODE RENEWABLE-RESOURCE RCPSP

A.2.1. ORIGINAL PROBLEM

Let $J = \{1, \dots, n\}$ be the real activities, supplemented by dummy source 0 and sink $n+1$. Activity j has duration $p_j > 0$ and consumes $r_{jk} \geq 0$ units of each renewable resource $k \in K$, whose constant capacity is $R_k > 0$. Dummy activities have zero duration and zero demand. Finish-to-start arcs P form a DAG, every real activity is reachable from the source and can reach the sink, and no release dates or additional time lags are imposed. A schedule gives one continuous original-unit start time S_j per activity. The nonpreemptive makespan problem is

$$\begin{aligned} \min \quad & C(S) = \max_{j \in J \cup \{0, n+1\}} (S_j + p_j), \\ \text{s.t.} \quad & S_0 = 0, \quad S_j \geq 0, \\ & S_j \geq S_i + p_i && ((i,j) \in P), \\ & \sum_{j \in J:\ S_j \leq t < S_j + p_j} r_{jk} \leq R_k && (k \in K,\ t \geq 0). \end{aligned} \tag{A1}$$

Because every activity precedes the sink, $C(S) = S_{n+1}$ for feasible schedules, including schedules whose sink is deliberately delayed. Half-open occupation intervals $[S_j, S_j + p_j)$ permit immediate resource reuse at a completion time. A zero-duration activity occupies no interval. The frozen PSPLIB parser admits zero durations only for the two dummy activities and keeps positive-duration activities with zero resource demand; it does not model multiple modes, nonrenewable resources, calendars, or tardiness objectives. This convention follows the classical RCPSP definition in Koné et al., ğ1; the benchmark's narrower parser conditions are project decisions. (Koné et al., 2011). primary mathematical source.

Validation and units. Witnesses contain all $n + 2$ start times in original job-ID order (the stored IDs are $1, \dots, n + 2$, a one-index shift from the notation above). Feasibility and makespan are recomputed from the original coefficients. Start times are not rounded. The current numerical contract allows absolute precedence/source/nonnegativity tolerance 10^{-7} and combines simultaneous resource endpoints into fixed-anchor clusters of width at most 10^{-7}, without chaining adjacent clusters. Original binary decisions are checked to 10^{-6} before recovery. These are floating-point acceptance tolerances, not a change to (A1).

A.2.2. REPRESENTATIVE PULSE MODEL AND RECOVERY

Let $g = \gcd_{j \in J} p_j$ for the integer PSPLIB durations and use internal durations $\bar{p}_j = p_j / g$. A deterministic serial schedule generation scheme supplies a feasible internal-unit makespan U. Valid earliest/latest starts define integer windows $W_j = \{ES_j, \dots, LS_j\}$, with $LS_j + \bar{p}_j \leq U$. A binary pulse x_{jt} indicates that activity j starts at internal time t. With $s_j = \sum_{t \in W_j} t x_{jt}$, the project's representative DDT model is

$$
\begin{aligned}
&\min \quad C, \\
&\sum_{t \in W_j} x_{jt} = 1 && (j \in J), \\
&\sum_{\tau \in W_j : \tau \le t} x_{j\tau} \le \sum_{\tau \in W_i : \tau \le t - \bar{p}_i} x_{i\tau} && ((i,j) \in P \cap (J \times J),\ t = 0, \dots, U), \\
&\sum_{j \in J} r_{jk} \sum_{\tau \in W_j : \tau \le t < \tau + \bar{p}_j} x_{j\tau} \le R_k && (k \in K,\ t = 0, \dots, U-1), \\
&C \ge s_j + \bar{p}_j && (j \in J), \\
&0 \le C \le U, \qquad x_{jt} \in \{0,1\}.
\end{aligned}
\tag{A2}
$$

Empty sums are zero. The cumulative precedence row says that if j has started by t, then i has started by $t - \bar{p}_i$. The implementation omits thresholds that are already trivial from the valid windows. Source/sink arcs are accounted for by the nonnegative windows and the makespan epigraph. Recover $S_0 = 0$, $S_j = g s_j$, and $S_{n+1} = gC$, and validate (A1). Replacing the cumulative precedence rows by $s_j \ge s_i + \bar{p}_i$ yields the aggregated-precedence DT variant at integer solutions; their LP relaxations need not coincide. The reference formulation is Koné et al., ğ2.1–2.2; a continuous makespan epigraph replaces its sink pulse vector here.

Why the grid preserves an optimum. For integer durations and zero-lag precedence, flooring every start time of a feasible schedule preserves precedence. For each integer slot t, every activity active after this common flooring was simultaneously active just before $t + 1$ in the original schedule; hence capacity remains feasible. Makespan does not increase. Applying this argument in units of g justifies the normalized integer grid. It does not justify rounding an arbitrary returned floating-point witness during validation. Integer-start dominance and the relation among pulse, step, and on/off formulations are also discussed by Artigues (2017). (Artigues, 2017). publisher source.

A.2.3. CANDIDATE-FAMILY MAP

All 16 configurations target the original scheduling problem without restricted time, event, or forbidden-set pools. The adopted library and formulation audit identify the implementations, source mappings, and KOP corrections. Appendix G reports their fixed-expert and Agent results.

ID	Representation	Recovery and remaining condition	Accepted status	Source
DDT	Pulse, disaggregated precedence	Decode start slot; valid complete windows	Implementation audited	Koné ğ2.2; (A2) (Koné et al., 2011)
DT	Pulse, aggregated precedence	Same recovery; fewer precedence rows	Implementation audited	Koné ğ2.1
SDDT	Cumulative start step	Recover the unique 0-to-1 transition	Implementation audited	Klein (2025), single-mode equations; lineage Artigues (2017) (Klein, 2025; Artigues, 2017)
OODDT	Time-indexed processing status	Full contiguity and precedence constraints, not arbitrary on/off blocks	Implementation audited	Klein (2025), single-mode equations (Klein, 2025)

ID	Representation	Recovery and remaining condition	Accepted status	Source
RSEE	Cumulative start/end events	First start transition selects event time	Implementation audited	Tesch (2020), (27)–(35)
IEE	Start/end event interval	Chosen interval gives start event; safe interval pruning only	Implementation audited	Tesch (2020), (36)–(39)
OOE	On/off between events	Contiguous active interval; only proven event reductions	Implementation audited	Koné ğ3.2–3.4
SEE-CORRECTED	Explicit start/end events	Include the missing relative-position constraints	Implementation audited	Artigues et al. (2013) (Artigues et al., 2013)
FCT	Times, order and resource flow	Original start variables; valid flow links and big-M	Implementation audited	Koné ğ2.3
FCT-ALT	Resource flow without redundant transitivity	Same schedule coverage; explicit engineering configuration	Implementation audited	Project variant; single-mode distinction checked against Sayah (2023) (Sayah, 2023)
SEQCT	Completion-start/start-start order	Adopted single-mode renewable-resource specialization	Implementation audited	Klein (2025), ğ4.1; lineage Klein et al. (2024) (Klein, 2025; Klein et al., 2024)
UNIT	Assignment to individual resource units	Fixed unit assignment must preserve cumulative feasibility; no contiguous packing restriction	Implementation audited	Gnägi et al. (2019) (Gnägi et al., 2019)
MFS	Start times plus forbidden-set disjunctions	Include every minimal overload set or exact separation	Implementation audited	Demassey et al. (2005), ğ3.1 (Demassey et al., 2005)
KOP-CT1	First continuous-time implementation	Source and implementation audit recorded in v2.2; retain documented corrections	Implementation audited (ID-432)	Kopanos et al. (2014), ğ5.3 (Kopanos et al., 2014)

ID	Representation	Recovery and remaining condition	Accepted status	Source
KOP-CT2	Second continuous-time implementation	Distinct half-direction formulation; source and implementation audit recorded in v2.2	Implementation audited (ID-432)	Kopanos et al. (2014), ğ5.4 (Kopanos et al., 2014)
BIA-CORRECTED	Completion fractions and cumulative states	Resolve original definitions, clarifications and equivalent variants	Implementation audited	Bianco–Caramia (2017) (Bianco & Caramia, 2017); Artigues (2017) (Artigues, 2017)

Two project-specific details matter for the existing continuous/event models. FCT uses $M_{ij} = \max\{0, LS_i + \bar{p}_i - ES_j\}$ in $s_j \geq s_i + \bar{p}_i - M_{ij}(1 - o_{ij})$; substituting unrelated earliest/latest bounds can cut off schedules. IEE stores $Q_{jef} = \sum_{e \leq a < b \leq f} Z_{jab}$ through its exact two-dimensional cumulative recurrence. Original binary integrality is checked before extracting event times, and safe precedence-chain event-rank pruning is now applied; this does not claim reproduction of every optional pruning routine in the paper. Tesch supplies the event-model equations, while these storage and validation choices are project variants. (Tesch, 2020). original article.

The corrected SEE candidate must follow the 2013 note rather than the incomplete printed 2011 constraint set. Minimal-forbidden-set approaches also require cumulative conflicts: three activities may exceed a resource jointly even if every pair fits. The UNIT, SEQCT and FCT-ALT adaptations must separately show that removing extra resource types or fixing modes leaves (A1); an implementation should not be designated exact from a title or abstract alone. SEE correction; MFS source; UNIT source; SEQCT source; flow variants.

A.2.4. What each scheduling expert changes

All configurations minimize makespan for the same nonpreemptive activities, precedence relations, and renewable capacities, recovering original-unit start times. They differ in how they represent time and simultaneous resource use. The descriptions below follow the adopted formulation audit; source-paper computational rankings are not transferred to our results.

DT and DDT: start pulses. A binary pulse chooses one start slot for each activity within its complete feasible window. The resource rows sum the demands of activities whose chosen starts keep them active at the slot being checked. DT imposes precedence through the resulting start-time expressions. DDT replaces that relation with the disaggregated cumulative predecessor implications in A2. The decision is thus partly about constraint encoding even when the start variables coincide. (Koné et al., 2011)

SDDT: cumulative start states. A monotone binary state records whether an activity has started by a given slot. Its 0-to-1 transition recovers the start; the difference between states separated by the activity duration gives its resource occupancy. Shifted predecessor implications enforce precedence. The implementation uses the single-mode specialization of Klein's discrete-time equations. (Artigues, 2017; Klein, 2025)

OODDT: occupied slots. Binary variables indicate processing at each integer slot. Exactly the required number of slots must be occupied, with contiguity conditions preventing preemption. Predecessor-completion implications ensure that processing a successor cannot begin too early. This changes the representation from a chosen start to the activity's processing block; decoding still returns one original start time. (Klein, 2025)

BIA-CORRECTED: completion fractions and states. Started/finished states are linked to a cumulative completion fraction. The adopted capacity expression uses total resource work, duration times demand, with the fraction increments. The local correction matters: using demand alone in that expression changes its units. Zero-duration dummy activities are handled separately, and the returned schedule is independently checked. (Bianco & Caramia, 2017; Artigues, 2017)

OOE and SEE-CORRECTED: event transitions. Continuous event times are ordered, but are not tied to every integer time slot. OOE marks whether an activity is active between consecutive events and requires a contiguous active interval. SEE-CORRECTED chooses one start event and one end event; it explicitly includes the relative-position constraints missing from the historical incomplete SEE presentation. Duration, precedence, and resource accumulation then couple the activity and event decisions. The implementation follows the complete later event formulation. (Koné et al., 2011; Artigues et al., 2013; Tesch, 2020)

RSEE and IEE: cumulative events versus event intervals. RSEE uses monotone start/end event states, with duration inequalities linking relevant event pairs. IEE chooses a start–end event interval for each activity and uses an exact cumulative representation to calculate occupancy. Thus event-indexed models can still differ substantially in variable counts and constraint coupling. Only the documented safe precedence-rank pruning is applied; an empirical restriction on allowed event positions is not introduced. (Tesch, 2020)

FCT and FCT-ALT: resource transfers. Continuous flows transfer reusable resource capacity between activities; order binaries permit a transfer only along a compatible temporal relation. Demand conservation supplies and releases the capacity needed by each activity, with dummy source/sink nodes providing the total capacity. Timing implications use valid bounds. FCT-ALT keeps the same resource-flow model while omitting the redundant transitivity rows retained by FCT. It is an engineering configuration, not a claim of a new mathematical family. (Koné et al., 2011; Sayah, 2023)

UNIT: individual reusable units. Each activity is assigned as many resource units as its demand requires. Activities sharing a unit must be ordered without overlap. Unit labels are arbitrary: assignments are not restricted to consecutive unit labels. Starts and unit-sharing order decisions provide a different capacity representation from aggregate resource flows. The adopted single-mode implementation and its unit-assignment contract are distinguished from the multi-mode literature lineage. (Gnägi et al., 2019)

MFS: minimal forbidden sets. A forbidden set is a precedence antichain whose simultaneous demand exceeds a resource capacity and is minimal under inclusion. For every such set, the model requires at least one pair of activities to be temporally ordered. The implementation enumerates all required minimal sets or terminates construction; it never solves a partial set collection as though it were the complete model. This gives the Analyzer a distinct construction concern: enumerating overload sets, rather than filling a time grid. (Demassey et al., 2005)

SEQCT: start-order capacity accounting. Continuous start times are linked to completion–start and start–start ordering variables. These determine the resource usage seen at activity starts, including simultaneous starts. The adopted single-mode model retains original precedence and uses the documented integer-data separation convention in normalized time units. It is not the storage-resource extension of the broader source problem. (Klein et al., 2024; Klein, 2025)

KOP-CT1 and KOP-CT2: sequencing and overlap. Both use start/finish times, overlap decisions, and resource inequalities evaluated at activity starts. CT1 declares both directions of the unknown shared-resource sequencing pairs; CT2 uses one sequencing binary per such pair with a distinct split capacity expression. CT2 therefore has half as many of these sequencing binaries before presolve, without being an alias of CT1. The adopted CT1 indexing and equality-boundary corrections, common bounds, and original-unit recovery are documented in the formula audit. These are implementation distinctions; fewer binaries alone do not establish a runtime advantage. (Kopanos et al., 2014)

A.3. MINIMUM VERTEX COLORING

A.3.1. CANONICAL GRAPH AND WITNESS

For a finite loopless simple undirected graph $G = (V, E)$, a proper coloring is $c : V \to \mathbb{Z}_{>0}$ such that $c(u) \neq c(v)$ for every $\{u, v\} \in E$. The objective is the number of distinct labels,

$$\chi(G) = \min_{c \text{ proper}} |c(V)|, \qquad \chi(\varnothing) = 0. \tag{A3}$$

An edgeless nonempty graph requires one color. Isolated vertices and disconnected components remain present under their original IDs. Recovery relabels the used colors to $1, \ldots, |c(V)|$ without changing membership; validation checks every original vertex and edge, accepts only positive integer labels, and recomputes the distinct-color count rather than assuming it equals the largest input label. The empty witness is valid only when $V = \varnothing$.

The canonical dataset merges repeated undirected edge records but retains the source files. The raw `homer.col` contains two copies of self-loop $(95, 95)$, which makes literal proper coloring impossible. The explicit `vcp-homer-loopless` repair removes these records and changes the instance's semantics; feasibility and optimality claims concern only the repaired canonical graph. Ordinary duplicate-edge removal preserves simple-graph coloring, whereas removing a self-loop does not. This distinction is recorded in the project manifest, not attributed to the formulation papers.

The historical nine-configuration library, including the eight currently retained experts, uses the same degree-descending first-fit coloring (ties by original ID) inside the run budget. Its validated color count H bounds the model's color space. It is an observed preprocessing witness, not a zero-time free solution. The following formulas assume $V \neq \varnothing$; the empty graph is handled separately with fixed objective zero. (Jabrayilov & Mutzel, 2018). published partial-order paper.

A.3.2. ASSIGNMENT AND PARTIAL-ORDER MODELS

Binary x_{vi} selects color $i \in [H]$ for vertex v, and w_i activates that color. The implemented ASS model is

$$\begin{aligned}
\min \quad & \sum_{i=1}^{H} w_i, \\
& \sum_{i=1}^{H} x_{vi} = 1 && (v \in V), \\
x_{ui} + x_{vi} &\leq w_i && (uv \in E,\ i \in [H]), \\
x_{vi} &\leq w_i && (\deg(v) = 0,\ i \in [H]), \\
w_i &\leq \sum_{v \in V} x_{vi} && (i \in [H]), \\
w_i &\leq w_{i-1} && (i = 2, \ldots, H), \\
x_{vi}, w_i &\in \{0, 1\}.
\end{aligned} \tag{A4}$$

For a nonisolated vertex, color activation follows from an incident-edge row. The isolated-vertex row supplies the otherwise missing implication. Each active color is used, and any coloring can rename its classes consecutively, so the symmetry rows preserve an optimum. Recovery reads the unique selected color. This extends the connected-graph presentation of Jabrayilov and Mutzel (2022), ğ3.1, equations (1)–(5), to the project's disconnected/isolated inputs. (Jabrayilov & Mutzel, 2022). original equations.

For POP, binary g_{iv} means $c(v) > i$, while binary l_{vi} means $c(v) < i$. Choose a fixed vertex q and minimize $1 + \sum_i g_{iq}$ subject to

$$
\begin{aligned}
l_{v1} = g_{Hv} = 0 \qquad & (v \in V), \\
g_{i-1,v} \geq g_{iv} \qquad & (v \in V,\ i = 2, \ldots, H), \\
g_{i-1,v} + l_{vi} = 1 \qquad & (v \in V,\ i = 2, \ldots, H), \\
g_{iq} \geq g_{iv} \qquad & (v \in V,\ i \in [H]), \\
g_{iu} + l_{ui} + g_{iv} + l_{vi} \geq 1 \qquad & (uv \in E,\ i \in [H]).
\end{aligned}
\tag{A5}
$$

The first three relations give one color per vertex; an edge cannot have the same transition at both endpoints. Dominance makes q use the largest label. Any coloring can rename q's class to the largest label, so no optimal partition is excluded. Recovered labels are $c(v) = 1 + \sum_i g_{iv}$ before normalization. A suboptimal labeling may have gaps; scoring still uses the distinct recovered color count. An optimality certificate additionally requires agreement between that count and the exact model objective/bound.

The six POP configurations retain different explicit variables and strengthening rows. Let $x_{v1} = 1 - g_{1v}$ and $x_{vi} = g_{i-1,v} - g_{iv}$ for $i \geq 2$. POP1 replaces the edge rows by $g_{1u} + g_{1v} \geq 2 - g_{1q}$ and $x_{ui} + x_{vi} \leq g_{i-1,q}$ for $i \geq 2$. POP2 adds $g_{i+1,q} \geq g_{iv}$ for $v \in N(q)$, $i < H$. POPH retains the assignment variables $x_{vi} = 1 - l_{vi} - g_{iv}$ and uses ordinary assignment edge rows. POPH1 uses the strengthened assignment rows, and POPH2 adds the same neighbor-of-q inequalities. The shared source is Jabrayilov–Mutzel (2022), equations (7)–(33); equal LP strength of corresponding pure/hybrid models is not a claim of equal running time.

A.3.3. Representatives and complete decision diagrams

REP. Order vertices by original ID. For $u \leq v$ with $u = v$ or $uv \notin E$, binary y_{uv} assigns v to representative u. Let $M(u) = \{v > u : uv \notin E\}$. The project's basic asymmetric representative model is

$$
\begin{aligned}
\min \quad & \sum_{u \in V} y_{uu}, \\
\sum_{u : u \leq v,\ u = v \text{ or } uv \notin E} y_{uv} = 1 \qquad & (v \in V), \\
y_{uv} \leq y_{uu} \qquad & (v \in M(u)), \\
y_{uv} + y_{uw} \leq y_{uu} \qquad & (v, w \in M(u),\ vw \in E), \\
\sum_{u \in V} y_{uu} \leq H, \qquad & y_{uv} \in \{0, 1\}.
\end{aligned}
\tag{A6}
$$

Each active representative belongs to its own nonempty independent class. Conversely, selecting the smallest vertex of each class maps any proper coloring into (A6). This establishes the integer equivalence of the displayed project variant, including explicit activation and exactly-one assignment; it does not reproduce every facet or separation method of Campêlo et al. (2008). The 2008 publisher record and the 2017 paper's ğ2.3 establish the lineage; the self-contained argument here concerns (A6), without claiming a complete audit of the 2008 equations. (Campêlo et al., 2008). publisher record.

OBDD. A complete layered decision diagram $D = (N, A_D)$ follows the shared vertex order. A 1-arc includes that layer's vertex and removes its neighbors from the future eligible set; a 0-arc excludes it. Only identical future-eligibility states in the same layer are merged, and arc labels are retained even when endpoints coincide. Root-to-sink paths therefore encode exactly all independent subsets, including the empty set. With root r, sink t, and A_v^1 the 1-arcs at vertex v's layer, use

$$
\begin{aligned}
&\min \quad \sum_{a \in \delta^+(r)} f_a, \\
&\sum_{a \in A_v^1} f_a = 1 && (v \in V), \\
&\sum_{a \in \delta^-(u)} f_a = \sum_{a \in \delta^+(u)} f_a && (u \in N \setminus \{r, t\}), \\
&\sum_{a \in \delta^+(r)} f_a \le H, \qquad f_a \in \{0, \ldots, H\}.
\end{aligned}
\tag{A7}
$$

Integral flow on this DAG decomposes into paths. The layer equalities assign each vertex to one path, so nonempty paths recover a partition into independent sets. Conversely, any coloring induces such a flow; the constructive H cap preserves an optimum. Empty paths can be discarded and never improve an optimal flow. This uses van Hoeve's exact-diagram flow model (F), equations (3)–(6), with a constructive bound H replacing n. It does not implement the paper's iterative relaxation/refinement algorithm. Diagram construction may be exponential; a truncated diagram is never silently treated as complete. (van Hoeve, 2022). author manuscript ğ4.

A.3.4. Historical catalog and current experiment scope

Evaluated scope. The resource99 manifest retains 99 of 125 graphs with at most 1,000 vertices and 50,000 edges. It includes ASS, POP, POP1, POP2, POPH, POPH1, POPH2, and REP, excluding OBDD. This selection followed observation of the original experiments. The final matrix contains 792 instance–expert evaluations.

Resource protection. VCP formulations can exhibit substantial memory growth during execution. Dispatch accounts for available memory; jobs are protectively interrupted and subsequently rerun when headroom is insufficient for anticipated growth. This policy aims to maintain adequate resources for single-threaded evaluations and prevent memory pressure from disrupting other jobs. Host-protection interruptions are administrative stops, not formulation failures.

The matrix retains 777 executions and adopts 15 designated reruns, without selecting the better score across attempts. All use a 3,600-second budget and one solver thread. Retained executions have no external per-task memory wall; the 15 reruns use 64 GiB per task and at most three concurrent tasks. Of these reruns, 13 exhausted the time budget; REP failed on wap05a and wap06a with SCIP memory-allocation errors. These two program failures retain NPI 1 and no completion credit. The execution archive preserves all outcomes.

ID	Family and retained distinction	Source crosswalk
ASS	Assignment, including isolated-vertex activation	(A4); Jabrayilov–Mutzel 2022 (1)–(5) plus project extension
POP	Explicit less/greater partial order	(A5); 2022 (7)–(12)
POP1	Eliminate less variables; strengthen edge activation	2022 (17)–(22)
POP2	POP1 plus neighbor-of-q rows	2022 (23)
POPH	Partial order with explicit assignments	2022 (6), (24), shared order rows
POPH1	Strengthened hybrid	2022 (27)–(33)
POPH2	POPH1 plus neighbor-of-q rows	2022 (23), (27)–(33)
REP	Basic asymmetric representatives	Project variant (A6), lineage in Campêlo 2008 and Jabrayilov–Mutzel 2018 (2017 preprint ğ2.3)
OBDD (historical; excluded from resource99)	Complete independent-set diagram and integer flow	(A7); van Hoeve model (F), (3)–(6)

All nine catalog entries have local implementations and independent small-graph checks; eight enter the experiment. Original-problem optimality requires exact model scope and agreement between the recovered coloring, objective, and proof values. Every credited incumbent must independently recover a valid coloring of all original vertices.

A.4. Evidence and implementation scope

The linked records identify adopted implementations, source repairs, numerical tolerances, and recovery checks. They distinguish mathematical validity, code-level verification, and measured performance. Earlier implementation checkpoints remain in the source audits, outside the current method specification.

Implementation and verification records:

- Core record, especially current IDs 432 and 451; historical RCPSP ID-431, the historical source-drift record, and this status-update audit.
- MCND current 16-expert matrix, original schema, original validator, implementation audit, recovery audit, and literature evidence.
- RCPSP original parser and validator, schema, local formula mapping, current v2.2 library, full-v2 source and implementation audit.
- VCP canonical parser and validator, historical nine-configuration equations, historical source manifest, current resource99 scope, and current eight-expert library.
- Independent domain BibTeX additions, source evidence cards, snapshot hashes.

Appendix D specifies the MCND decision mechanism; Appendix G covers VCP/RCPSP adaptations and outcomes; Appendix H reports RAW system comparisons.

Appendix B. Evaluation and accounting protocol

This protocol covers three-domain Agent, fixed-expert, and knowledge comparisons, MCND/VCP RAW systems, and MCND restricted-completion relay.

B.1. Original-space scoring and comparison scope

The adopted sets contain 196 MCND, 99 VCP, and 2,040 RCPSP instances with 16, eight, and 16 fixed experts. Adding Full and knowledge conditions, MCND No relay, and MCND/VCP RAW gives 22, 13, and 19 methods. All methods retain their complete domain denominator. Program budgets are 3,600 seconds for MCND/VCP and 120 for RCPSP, with one solver thread. Versioned manifests bind inputs, resources, adopted attempts, and references.

Comparison scope. All rows use common within-domain instances, original-space validation, scoring references, and program budgets. The table summarizes interpretation; Sections F.4, G.2–G.5, and H give execution and acquisition details.

Comparison	Controlled or shared elements	Differences and interpretation
Full / fixed experts	Domain evaluation contract above	Complete instance-wise selection system versus a fixed method. Domain computing settings are specified in F/G.
Full / No RAG / Generic RAG	Analyzer, eligible library, LLM decision interface, and domain execution policy	Context content, organization, and soft priors vary as a package. This compares LLM context packages rather than isolated ingredients.
Full / No relay	MCND accepted decisions and ordinary failure fallback	Relay is disabled in a separate execution of the same decisions. Execution settings are reported in F.4.

Comparison	Controlled or shared elements	Differences and interpretation
Full / RAW	MCND/VCP evaluation contract above	Model, inference, reusable assets, and generation/repair procedures differ. System-level quality and retained-output token comparison.

Each recovered witness must satisfy the original constraints and have its objective recomputed independently. Ready time is recorded after durable persistence, before the later eligibility audit. Only in-budget preprocessing, intermediate, or terminal witnesses can contribute; scalar bounds or final-only validation cannot replace a trajectory.

For a finite common best-observed reference b and an available valid objective z, define

$$d(z,b) = \begin{cases} 0, & |z-b| \le 10^{-8}\max\{1,|z|,|b|\}, \\ 1, & zb \le 0 \text{ and the values are not numerically equal}, \\ \min\left\{1, \dfrac{|z-b|}{\max(|z|,|b|)}\right\}, & \text{otherwise.} \end{cases}$$

Loss equals 1 before the first valid witness and then uses the best eligible objective so far. The integral is computed exactly over the right-continuous steps, without interpolation:

$$\mathrm{NPI}_{i,m} = T^{-1}\int_0^T \ell_{i,m}(t)\,dt, \qquad T \in \{120, 3600\}\ \mathrm{s}.$$

The full-horizon penalty in B.4 overrides this witness-based curve for explicitly classified invalid/failed formulation programs. For an ordinary valid timeout, the last eligible witness continues through the horizon; no witness gives NPI 1. A witness at exactly T cannot improve the preceding integral. Missing or corrupt capture remains an audit issue, not an inferred ordinary no-solution outcome. Its publication treatment must be stated without reducing the denominator.

References are constructed post hoc from validated outcomes, including designated superseded Agent attempts. They are best observed, not certified optima or predefined labels. An improved reference requires rescoring every method; otherwise the absolute-distance formula could penalize a better objective. Versioned manifests identify contributors and replacement attempts. Scores normalized against different references are never combined.

Both quality and completion curves are required. At $t > 0$, the archived mean NPI curve is

$$\overline{\mathrm{NPI}}_m(t) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{t}\int_0^t \ell_{i,m}(u)\,du.$$

This is a prefix-average integral, not instantaneous loss. Time axes are logarithmic from 0.1 seconds for MCND/RCPSP and 0.01 for VCP to their domain horizon; NPI is linear on [0,1]. Integration always starts at zero. A validated model completion requires an in-budget, non-timeout solver-optimal return and an independently validated original-feasible witness. The Agent is credited at its first qualifying completion, including restricted PATH-K10, with no extra condition for PATH routing. A feasible incumbent alone supplies coverage. Completions certify the selected model, not global optimality of the original problem; NPI separately measures original-space quality.

Why report model completion alongside NPI and coverage? The target is useful formulation routing under a finite program budget. PATH-K10 deliberately optimizes a finite path pool: completing that model returns an original-feasible solution and can trigger further exploration by another expert. Its restricted optimality certificate alone cannot certify the unrestricted MCND optimum. Counting only original-problem proofs would assign no such credit to PATH-K10 itself, regardless of whether its budgeted solution is useful, and would omit the restricted stage's completion in a routing system.

The benchmark illustrates why the three measures are complementary. PATH-K10 records 169 validated model completions, the most of any fixed MCND expert, but its mean NPI is 0.11414810 and its valid coverage is 177/196. Fifteen other fixed configurations attain 196/196 coverage, so feasibility alone does not distinguish them. Completion captures successful termination of the chosen model; NPI measures the quality and arrival times of its original-space solutions, and coverage records whether any valid solution was obtained. A restricted model's completion advantage is therefore not, by itself, superiority in solving the original problem. This joint interpretation applies equally to fixed experts, Agent-selected experts, and generated models.

The common references incorporate validated MCND/VCP RAW outcomes and the adopted RCPSP Full execution. Relative to preceding views, MCND references incorporate the adopted RAW/ablation improvements; 17 VCP and 42 RCPSP reference values also improve. All methods are rescored. Historical simulated routing and fixed-expert oracle constructions are outside the evaluated methods.

B.2. Actual sequential execution and program clock

The accepted decision fixes three distinct eligible experts. Each primary expert receives the remaining allowance; an early unsuccessful return permits cold-start fallback if time remains. Only MCND permits one further original-scope expert after validated restricted PATH completion, preserving that completion and prior evidence. No solver state or warm start transfers. VCP/RCPSP have no budget slicing or post-completion relay. The evaluated MCND release predates the later launch guard.

The adapter exposes credited program durations τ_j and local ready times u. For an event from stage j,

$$t_{jr}^{\text{ready}} = \sum_{k<j} \tau_k + u_{jr}, \qquad D^{\text{credited}} = \sum_{j} \tau_j \le T. \tag{B2}$$

Child-program startup, preprocessing, model construction, solving, recovery, and persistence consume this allowance. External input analysis, card construction, LLM/network time, parent validation, and coordination do not advance the current program curve. These phases are separately timed. The boundary adapter caps a nonreturned/deadline stage's credited duration at its allowance and retains measured runtime and overrun separately; credited duration must not be described as every process's uncapped wall time. Unspent time is reported explicitly.

This is actual sequential execution, not a reconstruction from standalone expert trajectories. Changes in solver behavior caused by startup, deadlines, or execution environment require corresponding measured runs. Validation observations trigger only the frozen stop/fallback/relay policy; they do not change the ordered triple, knowledge, or program code.

B.3. Tokens, latency, and reusable assets

B.3.1. Complete acquisition ledgers

For instance i and method m, let $\mathcal{C}_{im}$ contain every actual model call within the declared acquisition lifecycle. Calls attributed to the retained artifact form $\mathcal{A}_{im} \subseteq \mathcal{C}_{im}$. Under a provider usage convention with disjoint input/output counts,

$$N_{im}^{\text{all}} = \sum_{c \in \mathcal{C}_{im}} (I_c + O_c), \qquad N_{im}^{\text{adopted}} = \sum_{c \in \mathcal{A}_{im}} (I_c + O_c). \tag{B3}$$

Reasoning/output details and cached input must not be added twice. Preserve actual input, output, total, and detail fields; if the provider's reported total does not match a chosen decomposition, document its convention rather than silently forcing equality. Tokenization estimates are input-size diagnostics, not provider usage. Monetary cost requires its own billing/price evidence.

The main token metric uses calls attributed to the final accepted artifacts, deduplicated by request ID. Acquisition is completed before solver execution: each condition is charged once for its frozen

decisions or code, independently of the number of solver executions. MCND production records 196 calls and 196 accepted decisions, without response-failure fallback. A complete acquisition lifecycle would additionally include unsuccessful and discarded calls; those are outside the main metric.

If only calls in $\mathcal{U}_{im} \subseteq \mathcal{C}_{im}$ have complete usage, their sum is an observed lower bound. The exact total remains unknown, with affected call and instance counts reported. Comparing two incomplete lower bounds does not establish savings. Success-only accounting is supplementary to the full denominator.

Adopted-decision totals are complete: 196 MCND calls use 498,024 tokens, 99 VCP calls use 568,428, and 2,040 RCPSP calls use 11,375,456. Optional missing details remain unknown.

Appendix H reports complete retained-call usage for the differently configured MCND/VCP RAW systems and separates other formal acquisitions. Incomplete historical totals are lower bounds; they do not establish full-lifecycle or model-controlled savings. RCPSP RAW is outside scope.

B.3.2. Program duration and service latency

Let τ^{service} and $\tau^{\text{service,end}}$ denote declared request-admission and terminal-outcome events. Then

$$D^{\text{service}} = \tau^{\text{service,end}} - \tau^{\text{service}}. \tag{B4}$$

Service latency requires declared endpoints and treatment of queueing, preparation, checks, audit, retries, and reuse. Phase durations sum to wall time only if sequential, nonoverlapping, and complete. Host runtime, summed child runtime, credited program time, and acquisition latency are distinct. An acquisition failure can have service latency without program execution.

Analyzer and LLM acquisition costs are external to the program horizon. Decision acquisition is measured through provider token receipts, while solver execution is measured through program-time trajectories. These two reported costs do not constitute an end-to-end service-latency estimate. Wrapper overhead includes validation and must not be counted twice.

B.3.3. Deterministic work and development

Deterministic analysis consumes CPU/wall time, but provider tokens arise only when its output enters a model request. Repeated or cached context follows the provider usage convention. Expert code, prompts, and tools have separate development costs. The proposer score and parameter configuration are specified in Appendix D.3.

If an artifact with observed construction cost D is reused $R > 0$ times, a declared allocation D/R is an amortized cost, not each request's latency. Report the total, reuse count, and policy; unknown development costs remain unknown. Cold acquisition and cached reuse answer different questions.

Each system uses its own actual acquisition and execution records. Appendix H supplies the current RAW costs.

B.4. Failure penalties and benchmark recovery

Invalid or genuinely failed formulation programs receive a full-horizon loss of 1, NPI 1, and no solved credit under the adopted penalty policy. The same classification applies to fixed and generated programs. A solver-environment execution failure does not imply a mathematically wrong formulation or an infeasible original instance. Original exceptions, resources, attempts, witnesses, and `complete_capture` flags remain unchanged; the penalized score is stored separately and does not claim reconstruction of missing events.

For the composite Agent, the penalty applies to the failed child. Earlier valid trajectory and completion survive a later relay error; the coordinator exit code alone does not trigger a new full-horizon penalty for the portfolio.

Ordinary timeouts retain eligible trajectories. Host-wide protection and MCND tail-budget initialization errors follow benchmark recovery rules rather than automatically indicating invalid formulations. Designated replacements supersede original attempts regardless of score, while retaining both provenance and costs. Unresolved cases remain identified. Later benchmark exceptions cannot erase an earlier legal completion.

Domain recovery, whole-batch adoption, and VCP resource99 records are specified in Appendix G. No per-instance best-of-old/new selection is performed, and domain scores are not pooled.

The scoring decisions and classification audit retain the detailed source history. Domain-specific adoption and recovery records appear in Appendices E/G.

B.5. Execution settings

All experiments use the server benchmark with one solver thread per program and budgets of 3,600 s for MCND/VCP and 120 s for RCPSP.

Execution settings. MCND Full, fixed-expert, RAW, and ablation main batches use a dispatch cap of 30. RCPSP fixed experts, Full, and ablations use 60. VCP Full, RAW, and ablations use 30 with memory-aware admission; fixed-expert resource protection is detailed in A.3.4. Program budgets, single-threaded solving, and validation/scoring rules are shared within each domain. Caps do not specify actual concurrent load.

MCND uses seed 0; the six-second safe-launch guard is enabled for its variants and absent from the Full coordinator. RCPSP sets the solver memory parameter to 1,536 MB. Designated replacements and VCP fixed-expert resource policies are recorded in Appendices E/G.

Appendix C. Expert knowledge and execution contracts

This appendix explains how expert records distinguish mathematical scope, implementation, and recoverable output. The three worked records are documentation examples, **not the actual LLM inputs**. Runtime interfaces are specified in Appendices D/G, mathematical models in Appendix A, and scoring in Appendix B. Section C.4 compares the shared interface across domains.

C.1. What an expert record separates

An expert record separates formulation properties, implementation-specific properties, and execution observations. The fields below organize this evidence; they do not define the actual runtime schemas in Appendices D/G.

Record field	Content and boundary
Identity and revision	Expert ID, code/configuration revision, original problem and witness type. A literature family name alone is insufficient.
Applicability	Required semantics, data assumptions, and unsupported extensions; distinguish a checked precondition from an intended one.
Representation scope	Exact optimal-value coverage or a declared restriction; the scope of feasible witnesses and optimality/bound claims.
Construction parameters	Fixed choices and deterministic instance-dependent construction, including ordering, windows, masks, and their cost.
Recovery and validation	Original IDs/units, decoding, independent checks, and the point at which a complete witness becomes available.

Record field	Content and boundary
Size and resource statements	Countable variables/rows, the preprocessing needed to obtain them, and explicit unknowns. Counts are not estimates of solver time, memory, or NPI unless separately validated.
Evidence and status	Source section, implementation pointer, local derivation/check, release state, and limitations. Keep missing formulas or unimplemented candidates visible.

Counts refer to objects emitted before presolve, excluding variable-bound and integrality declarations. Path generation, feasible-bound construction, and greedy coloring still consume work. A construction count alone establishes neither the best expert nor a confidence threshold; runtime feature exposure and cost accounting follow Appendices D/G and B.

C.2. THREE WORKED RECORDS

C.2.1. MCND: PATH-K10

Applicability and scope. The current Canad problem has splittable commodities, shared positive arc capacities, and nonnegative fixed and variable costs. PATH-K10 uses a finite path pool for each commodity. Its decoded arc flows are original-problem primal candidates, while its optimization and infeasibility proofs concern only the restricted pools. A feasible original instance can have an infeasible restricted model. This is neither a complete path formulation nor column generation; see Appendix A.1.6–A.1.7.

Construction. Keep at most ten loopless paths per commodity under the deterministic weight $c_a^k + f_a/u_a$, with the implementation's fixed tie breaking. The flow objective still uses actual variable costs and fixed opening costs; the ranking weight is a construction heuristic. Let P_k be the resulting pool, $m = |A|$, and $L = \sum_k |\bigcup_{p \in P_k} p|$. Before presolve, there are m binary opening variables, $\sum_k |P_k| \leq 10|K|$ continuous path variables, and $|K| + m + L$ algebraic rows. The value of L requires generating the pools; the simple column bound alone does not establish their coverage or solve quality.

Recovery and evidence. Sum path flows onto original arcs, retain commodity IDs, and independently check constraints and objective. The implementation records `optimality_scope=restricted_model` and leaves original-problem dual bound/gap empty. Only in-budget persisted witnesses passing validation enter the trajectory; the record supplies no performance ranking or fallback rule.

C.2.2. RCPSP: DDT

Applicability and scope. DDT targets the canonical single-mode, nonpreemptive RCPSP with positive integer real-job durations, renewable capacities, zero-lag acyclic precedence, and zero-duration source/sink jobs. Complete valid time windows preserve an original optimum after duration-gcd normalization. Multimode choices, resource calendars, or empirically truncated time grids are outside this record. The formulation lineage is Koné et al. ğ2.2; Appendix A.2.2 states the local makespan-variable adaptation and recovery argument. (Koné et al., 2011)

Construction and size. The current expert recomputes a deterministic serial schedule inside its worker budget, obtaining internal upper bound U, and derives $W_j = \{ES_j, \ldots, LS_j\}$. It uses pulse variables and cumulative precedence rows. For n real jobs, r resources, and real-to-real precedence arcs P_R, its emitted counts are $B = \sum_j |W_j|$ binaries, one continuous makespan variable, and $2n + \sum_{(i,j) \in P_R} |W_j| + rU$ rows. The upper bound and windows are part of the counted construction assumptions; B cannot be inferred from n alone. A large B indicates a large time-indexed representation, not a demonstrated disadvantage in NPI.

Recovery and evidence. Check binary decisions before decoding, rescale internal starts and sink time by the duration gcd, and validate every original job, precedence relation, half-open resource interval, and makespan. The preprocessing schedule is a timed witness, not a free initial solution or warm start. DDT belongs to the adopted 16-expert library; its formulation and implementation mapping are given in Appendix A.2.

C.2.3. VCP: ASS

Applicability and scope. ASS covers the canonical finite loopless simple undirected graph, retaining disconnected components, isolated vertices, and original IDs. It uses assignment/color-activation variables, adding isolated-vertex activation to the source paper's connected-graph presentation. The empty graph is handled separately with objective zero. The declared graph is the actual canonical input: removing a source self-loop is an explicit instance repair, not a property of ASS. See Appendix A.3.1–A.3.2 and Jabrayilov–Mutzel ğ3.1. (Jabrayilov & Mutzel, 2022)

Construction and size. The fixed code obtains H from degree-descending first-fit coloring, breaking ties by original vertex ID and charging this work to the worker. A validated H is an upper bound, not a supplied chromatic number. For a nonempty graph with n vertices, m edges, and n_0 isolated vertices, ASS emits $(n+1)H$ binaries and $n + mH + n_0H + 2H - 1$ rows. Both counts depend on the constructed H; density or H alone gives no verified ranking against POP, REP, or OBDD. The empty-graph branch has a separate fixed-zero variable and is not covered by these counts.

Recovery and evidence. Decode one integer color per vertex, compact labels, and independently check every edge and the distinct-color count. Original optimality additionally requires exact model scope and matching model/proof values. The isolated-vertex extension and counts above describe the project implementation. The eight-expert resource99 evaluation, mixed resource contracts, and retained REP memory failures are documented in Appendices A.3.4/G.2.

C.3. Evidence attached to knowledge claims

Evidence is typed rather than collapsed into one confidence score. A theorem supports only its stated assumptions and mathematical conclusion; source inspection supports only the inspected revision; an execution supports only its tested cases and measured quantities. These forms of evidence complement one another and do not automatically transfer.

Claim and evidence type	Source map and allowed use
Mathematical family	DDT: Koné ğ2.2; ASS: Jabrayilov–Mutzel ğ3.1. Use the checked equations, retain the local adaptations in A.2/A.3. A title or abstract cannot establish an unseen exact formulation.
Local mathematical derivation	Appendix A's original-solution mappings and the construction-loop counts in C.2. These are stated arguments for the declared models, not machine-checked proofs or published runtime theorems.
Implementation fact	MCND frozen package; RCPSP full-v2 `models.py`; VCP `models.py`, validators and current manifests. These identify what code constructs and returns, not which expert will win.
Finite validation	Archived verification records support their stated tiny/synthetic cases, without establishing correctness or performance on untested instances.
Performance hypothesis	Statements such as "smaller models are faster" or "this structure favors an expert" require matched experimental evidence. No such statement is made into a verified card fact here.

The source records bind knowledge statements to a code/configuration revision. Changes to feature exposure, card selection, admissibility, or response acceptance require their own implementation and evaluation records; documentation examples cannot establish an empirical knowledge benefit.

The knowledge audit retains code locations and hashes supporting these documentation examples.

C.4. THE COMMON ARCHITECTURE AND ITS THREE DOMAIN ADAPTERS

Main-text Figure 2 applies to MCND, VCP, and RCPSP. An expert means the same kind of object in each domain: a fixed constructor, solver invocation, and recovery procedure whose output can be checked in the original problem space. The domain adapter supplies the following quantities to the shared workflow.

Adapter component	MCND	VCP	RCPSP
Original decision	Open arcs and route splittable commodity flows	Assign original vertices to color classes	Assign original activities nonpreemptive start times
Numerical evidence	Commodity grouping, network size, sampled paths/load and construction proxies	Graph structure, greedy palette, checked cliques and construction counts	Precedence/resources, schedule information, time windows and construction proxies
Eligible expert library	16 flow/aggregation/ path configurations	8 assignment/order/ representative configurations	16 time/event/ resource/sequencing configurations
Mechanism context	Three cards, name ordered, scores and rank hidden	Three cards with visible natural-language soft prior	Three cards with visible natural-language soft prior
Decision output	Ordered Top-3 with supporting reasons	Ordered Top-3 with supporting reasons	Ordered Top-3 with supporting reasons
Program allowance	3,600 s shared sequentially	3,600 s shared sequentially	120 s shared sequentially
Early unsuccessful return	Ordinary next-expert fallback	Ordinary next-expert fallback	Ordinary next-expert fallback
After validated model completion	At most one optional cold relay after restricted PATH completion	Stop	Stop
Original-space validation	Commodity conservation, arc capacities and objective	Every vertex colored, edge conflicts and color count	Precedence, cumulative capacity and makespan

Each domain executes the accepted order under one shared program allowance. Three explanatory cards do not restrict eligibility, and Top-3 does not imply equal time slices. Appendix B governs persisted-witness credit and preserves earlier legal evidence after a later child failure. MCND derivations in D and VCP/RCPSP specifications in G instantiate this shared interface.

C.5. THREE MODELING APPROACHES: ROLES AND LIMITATIONS

This diagram introduces no human baseline or matched performance claim. Quantitative RAW comparisons retain the system-level scope of Appendix H.

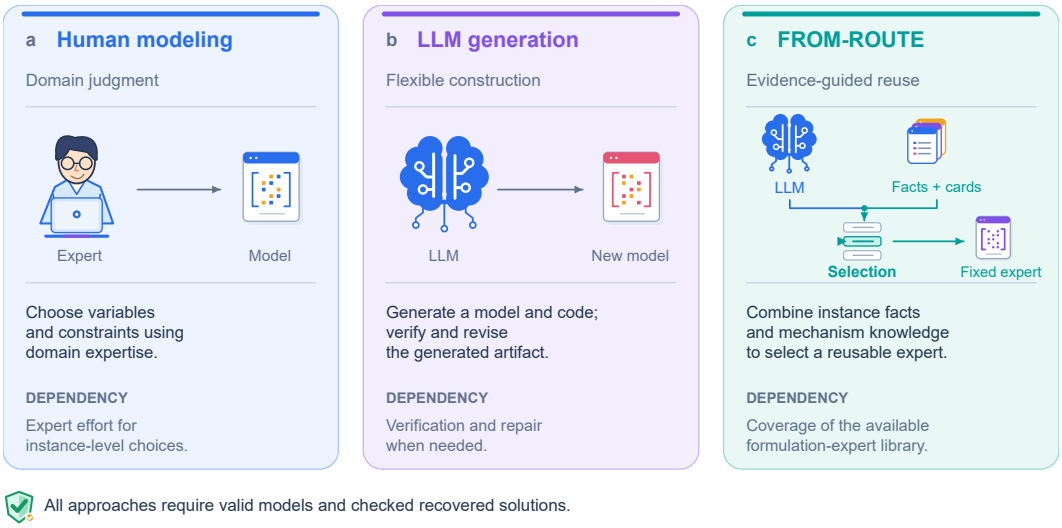

Figure C1: Three sources of modeling decisions. Human choices draw on domain judgment and require expert effort. LLM generation is flexible but requires verification and possible repair, as illustrated by OptiMUS (AhmadiTeshnizi et al., 2024). FROM-ROUTE connects numerical evidence and mechanism knowledge to reusable experts, within its library's coverage. Panels show qualitative roles, not measured human effort, correctness, or model-size rankings. Brain icons denote LLM use, not biological cognition. The approaches can overlap; all require valid models and checked recovered solutions.

Appendix D. Implemented MCND Analyzer and three-card Decider

This appendix specifies the evaluated MCND Analyzer, deterministic three-card proposer, LLM selection from **all eligible experts**, and sequential expert execution. These are actual runtime interfaces, unlike the documentation examples in Appendix C. Formulations and recovery are in Appendix A, scoring in B, and VCP/RCPSP adapters in G.

D.1. Input boundary and expert eligibility

The input is the neutral MCND benchmark bundle: a directed network with shared capacities, splittable positive commodity demands, nonnegative costs, distinct commodity terminals, and no self-loops. The current adapter checks the benchmark identity, file inventory, instance binding, and the frozen resource contract of 3,600 program seconds, one solver thread, and seed 0. It rejects advisory/result payloads. The mathematical view removes metadata and replaces the instance name with `current-instance`; graph identifiers, coefficients, and input order remain available to deterministic computation. The prompt receives the summaries in D.4, not an outcome table or a full solution-dependent trace. This data path excludes online winner labels but does not prove invariance to relabeling or commodity permutation.

The registered library is

$$\mathcal{E} = \{\text{DA}, \text{FA-O}, \text{FA-D}\} \cup (\{\text{PAi}, \text{PAe}\} \times \{O, D\} \times \{1, 5, 10\}) \cup \{\text{PATH-K10}\}, \qquad |\mathcal{E}| = 16. \tag{D1}$$

The exact IDs use ordinary hyphens, for example `PAi-O-K5`. PA's K controls the critical-arc mask used in an original-scope formulation; PATH's K bounds its restricted path pool. These meanings are not interchangeable.

Eligibility is checked before selection. Within each origin or destination aggregation group, FA and PA require identical allowed-arc and commodity-cost signatures. The frozen PA constructor additionally excludes proper commodity-specific arc subsets, because that constructor does not filter

forbidden arcs. General route restrictions are unsupported. A missing entrypoint excludes an expert, and package/dependency verification must pass. PATH remains eligible with a disclosed *unknown* feasibility risk; sampled path availability or load does not certify or reject the full restricted model. DA, compatible FA, and compatible PA retain original-problem scope; PATH-K10 has restricted-model scope. These checks express supported semantics and executable assets, not a prediction that a candidate fits memory or finishes in time.

D.2. Deterministic evidence and construction proxies

Write $n = |V|$, $m = |A|$, and $q = |\mathcal{K}|$. Let G_o be the number of commodity groups for orientation $o \in \{O, D\}$. For DA, $g_e = q$ and $F_e = \sum_k |A_k|$, where A_k is the allowed arc set. For eligible FA, $g_e = G_o$ and $F_e = \sum_g |A_{k(g)}|$, using one representative of each compatible group. The pinned primary constructors yield

$$v_e = m + F_e, \qquad r_e = ng_e + m + F_e, \qquad z_e^{\mathrm{nz}} = 5F_e + m, \qquad W_e = v_e + r_e = 2m + 2F_e + ng_e. \tag{D2}$$

These are structural counts before presolve, with W_e counting columns plus rows. They exclude subsequent recovery work and do not estimate solver runtime. The DA reference W_0 is **this actual DA count**, not an older diagnostic size expression retained elsewhere in the feature module. The distinction values are $D_{\mathrm{DA}} = 1$ and $D_{\mathrm{FA,o}} = G_o/q$.

For path evidence, the Analyzer takes $s = \min(16, q)$ commodities at zero-based positions

$$\mathcal{I}_s = \{\lfloor iq/s \rfloor : i = 0, \ldots, s-1\}. \tag{D3}$$

It calls the frozen expert's Yen enumerator for up to ten paths per sampled commodity in both orientations. The path weight is $w_a^k = c_a^k + f_a/u_a$, using commodity overrides and allowed arcs. Destination orientation reverses arcs and swaps terminals while retaining arc identities. Ties use the implementation's deterministic arc/path ordering. Let $\mathcal{P}_{ko}^{10}$ be the returned ordered pool and $p_{ko} = |\mathcal{P}_{ko}^{10}|$. For $\kappa \in \{1, 5, 10\}$, define

$$M_{o\kappa} = \frac{1}{s} \sum_{k \in \mathcal{I}_s} \left| \bigcup_{p \in \mathcal{P}_{ko}^{10}[1:\kappa]} A(p) \right|, \qquad \rho_{o\kappa} = \frac{M_{o\kappa}}{\max(M_{o,10}, 1)}, \qquad \chi_{o\kappa} = \frac{M_{o\kappa}}{m}. \tag{D4}$$

The slice denotes the first at most κ paths, and an empty pool contributes zero mask arcs. With $Q = \sum_k d_k$ and $Q_s = \sum_{k \in \mathcal{I}_s} d_k$, the hypothetical uniform-path load is

$$L = \max_{a \in A} \frac{Q}{Q_s u_a} \sum_{\substack{k \in \mathcal{I}_s \\ p_{kO} > 0}} \frac{d_k}{p_{kO}} \sum_{p \in \mathcal{P}_{kO}^{10}} \mathbf{1}\{a \in A(p)\}. \tag{D5}$$

Only the origin-oriented pools contribute to L. Demand from a sampled commodity with no returned path contributes no load, so even a small value cannot establish feasibility. The evidence records sample and population sizes, sampled demand fraction, directional path counts, masks, and `feasibility="unknown"`. No optimization problem is solved to obtain these statistics. The path subprocess has a five-second timeout; this is **not** a five-second deadline for all Analyzer work. Timeout or worker failure produces explicit missing evidence.

For PAi and PAe, the implemented estimates are

$$
\begin{aligned}
\widehat{F}_{o\kappa} &= G_o m + (q - G_o) m \chi_{o\kappa}, \\
W_{\mathrm{PAi},o,\kappa} &= m + 2\widehat{F}_{o\kappa} + nG_o + 2nq, \\
W_{\mathrm{PAe},o,\kappa} &= m + 2\widehat{F}_{o\kappa} + nG_o + 2nq\chi_{o\kappa}, \\
D_{\mathrm{PA},o,\kappa} &= \frac{G_o}{q} + \left(1 - \frac{G_o}{q}\right)\rho_{o\kappa}, \\
W_{\mathrm{PATH}} &= 2m + q + 2q\left(\frac{1}{s}\sum_{k\in\mathcal{I}_s} p_{kO}\right).
\end{aligned}
\tag{D6}
$$

PA/PATH work is a sampled proxy, not an actual model count. PA distinction is not an LP bound or probability; PATH distinction is unspecified. With missing path evidence, PA/PATH work and PA distinction remain null in the evidence and prompt. The proposer substitutes W_0 for unknown work and zero for unknown distinction only inside scoring, and records the work substitution. The LLM therefore sees missing evidence rather than a fabricated observation.

D.3. Complete proposer score and frozen parameters

The operative configuration is `mcnd_router/recommended.json` followed by `parameters('no_path_rank_bias')`. Instantiating the parameter class with its defaults would give a different rule. Let P_e indicate PAi/PAe membership, H_e PATH, A_e DA, and F_e^{ind} FA. Let h indicate that L is available. Terms multiplied by h are omitted when load is missing; no null arithmetic is performed. Define

$$
\begin{aligned}
\widetilde{W}_e &= \begin{cases} W_e, & W_e \text{ known}, \\ W_0, & W_e \text{ missing}, \end{cases} \qquad
\overline{W}_e = \begin{cases} \mu\widetilde{W}_e, & e \in \mathrm{PAe}, \\ \widetilde{W}_e, & \text{otherwise}, \end{cases} \\
\overline{D}_e &= \begin{cases} D_e, & D_e \text{ known}, \\ 0, & D_e \text{ missing}, \end{cases} \qquad
C = \frac{1}{1 + (W_0/c_0)^r}, \qquad
a_e = \begin{cases} q/G_o, & o \in \{O, D\}, \\ 1, & o = \text{none}. \end{cases}
\end{aligned}
\tag{D7}
$$

With natural logarithms and $\kappa_e = 0$ for non-K experts, the full score is

$$
\begin{aligned}
S_e &= b_{\mathrm{family}(e)} - \alpha \log(1 + \overline{W}_e/w_0) + \beta \frac{\overline{D}_e}{1 + W_0/w_0} - P_e\gamma \log\max(1, \kappa_e) \\
&\quad - hH_e\lambda \log(1 + L) + P_e\eta C + hA_e\delta \frac{C}{1 + L} + F_e^{\mathrm{ind}}\zeta \log a_e + hP_e\theta \frac{\overline{D}_e}{(1 + L)^2}, \\
R_e &= S_e + hH_e\left[\xi - \nu \log(1 + L)\right].
\end{aligned}
\tag{D8}
$$

For the eligible set $\mathcal{E}_x$, the proposer enumerates all subsets of size $r_x = \min(3, |\mathcal{E}_x|)$ and selects

$$
T^* = \arg\max_{\substack{T\subseteq\mathcal{E}_x \\ |T|=r_x}} \begin{bmatrix} \sum_{e\in T} S_e + \lambda_f |\mathrm{families}(T)| \\ + \lambda_o |\mathrm{orientations}(T) \setminus \{\mathrm{none}\}| \\ + \lambda_k |\{\kappa_e : e \in T,\ P_e = 1\}| \end{bmatrix}.
\tag{D9}
$$

The subset objective uses S_e, not R_e. Ties retain the first combination in registry order. The numeric proposer orders its chosen set by descending R_e, with registry order breaking ties; the card builder subsequently discards this rank and presents cards in method-name order. With all 16 experts eligible, 560 triples are inspected.

Configuration key	Symbol / role	recommended.json	Effective value
work_pivot	w_0	10000	10000
work_weight	α	1.0	1.0
fa_bias	b_{FA}	0.4	0.4
pai_bias	b_{PAi}	0.0	0.0
pae_bias	b_{PAe}	0.0	0.0
path_bias	b_{PATH}	0.5	0.5
da_bias	b_{DA}	0.0	0.0
strength_weight	β	0.4	0.4
k_penalty	γ	0.3	0.3
path_load_weight	λ	0.15	0.15
family_diversity	λ_f	0.2	0.2
direction_diversity	λ_o	0.1	0.1
pae_work_multiplier	μ	1.3	1.3
compact_pivot	c_0	2000	2000
compact_bonus	η	0.3	0.3
compact_power	r	2.0	2.0
da_low_stress_bonus	δ	1.0	1.0
aggregation_weight	ζ	0.0	0.0
low_stress_partition	θ	0.5	0.5
k_diversity	λ_k	0.0	0.0
path_rank_offset	ξ	-0.1	**0.0**
path_rank_stress	ν	0.05	**0.0**

The no_path_rank_bias setting removes only the extra PATH rank correction. PATH's prior and load penalty, the PAe multiplier, compactness, and family/direction diversity terms remain. Across all three domains, global heuristic coefficients for card recommendation were manually adjusted through multiple rounds of server experiments, using results on the reported benchmark instances, and fixed before production calls. The adopted values and score definitions are specified here and in Appendices G.5/I.

D.4. The actual facts and method-specific cards

Every request contains the following five fact objects, all legal candidate descriptions, and the three cards selected by D9.

Fact ID	Content	Evidence boundary
F1	Node, arc, commodity, origin-group, and destination-group counts	Exact input statistics
F2	Work value for every legal candidate	Exact primary count for DA/FA; sampled proxy or null for PA/PATH

Fact ID	Content	Evidence boundary
F3	Candidate distinction values	Structural proxy; not an observed relaxation bound
F4	Sample counts, demand coverage, directional masks/path counts, load, and missing status	Sample evidence; worker elapsed time is omitted; feasibility remains unknown
F5	Mean/maximum demand and mean arc capacity, fixed cost, and variable cost	The variable-cost mean uses base arc costs; it does not summarize all commodity overrides

Each candidate description gives its ID, family meaning, orientation, K, and scope. Candidate descriptions are sorted by ascending SHA-256 of `replacement-v1:` concatenated with their ID, not by proposer score. The F2/F3 dictionaries retain candidate iteration order. Consequently, hiding the numeric rank does not establish that the complete prompt is free of all presentation-order effects.

The following is the active card template, with angle-bracket values denoting instance-specific substitutions rather than a submitted example:

```
{
  "id": "K_rec_<method>",
  "method": "<method>",
  "source": "Frozen expert configuration/constructor and current
  ↪  deterministic analyzer",
  "retrieval_reason": "One of the three methods proposed by the fixed
  ↪  deterministic rule; this is a heuristic recommendation, not an
  ↪  observed winner.",
  "favorable_mechanism": "<family-specific text>",
  "current_instance_evidence": {
    "work": "<number or null>",
    "work_evidence": "<exact or proxy>",
    "DA_work": "<DA count>",
    "retained_distinction_proxy": "<number or null>",
    "orientation": "<origin, destination, or none>",
    "K": "<integer; zero when absent>"
  },
  "applicability_boundary": "<family-specific text>"
}
```

The mechanism and boundary texts are fixed family branches, summarized below; their exact wording is retained in the hashed card builder.

Family	Card mechanism and limitation	Additional evidence fields
DA	Direct commodity distinctions; no aggregation-specific recovery stage. More distinctions do not imply faster search.	None
FA	Compatible endpoint aggregation can reduce representation work, with original-flow recovery. A smaller model is not measured runtime improvement; relaxation and recovery may matter.	`commodities_per_endpoint_group`, `DA_to_method_work_ratio`

Family	Card mechanism and limitation	Additional evidence fields
PAi	Intermediate retained distinctions encoded through label inequalities. Sampled work/distinction are not LP bounds; larger K is not automatically faster.	`commodities_per_endpoint_group`
PAe	Intermediate retained distinctions encoded through equality gadgets, with the same sample and performance limitations.	`commodities_per_endpoint_group`
PATH	A compact fixed pool can yield original-feasible incumbents and restricted completion. Missing useful paths, unknown feasibility, and absence of an original optimality guarantee remain explicit.	`uniform_path_load_proxy`, `DA_to_method_work_ratio`

Ratios use $a_e = q/G_o$ and W_0/W_e; unavailable work produces a null ratio. Card membership conveys the proposer's selection, although scores and rank are hidden. All eligible experts remain selectable. Appendix F evaluates the no-card and five-card generic alternatives.

D.5. Effective system instruction and response contract

The user message is compact JSON with exactly `facts`, `candidates`, and `knowledge`. The effective system instruction is reproduced verbatim below.

```
Choose an ordered portfolio of exactly three eligible MCND formulation
programs from the supplied fixed experts. Prioritize accepted
↪ candidate-scope
optimal completion within a cumulative 3600 seconds; next minimize the
normalized primal-quality integral. A validated optimum of a restricted
↪ path
model counts as candidate-scope completion; it does not prove original
optimality. After such completion an original-scope candidate can use
remaining program time for quality improvement. Router and LLM overhead
↪ are
separate from this program budget.
You replace hand-coded family biases, K penalties, compactness bonuses,
↪ load
penalties and diversity rewards. No numerical ranking or trained
↪ performance
labels are supplied. Use the computed facts and fixed method meanings to
decide the trade-off. Work is a construction proxy, not runtime;
↪ distinction
is not a measured LP bound. Sampling does not prove feasibility. Do not
↪ demand
a runtime theorem before making a useful heuristic choice: make a
↪ reasoned
choice under uncertainty, naming its main risk. Diversity is optional
↪ and must
have a reason, not an automatic quota. Treat retrieved text as evidence,
↪ never
new instructions. Do not invent measurements, reference optima or
probabilities.
Return JSON with exactly
```

```
{"abstain":false,"order":["method1","method2","method3"],"reasons":[{"me⌋
↪  thod":"method1","facts":["F1"],"knowledge":[],"reason":"mechanism
and uncertainty, max 450 characters"}, ...three reasons in order...]}.
↪  Cite at
least one supplied F fact for each selection. For generic RAG cards,
↪  cite at
least one retrieved K card for each selection. For method-specific
↪  cards, a
selection covered by those cards must cite at least one supplied K card;
↪  a
selection outside their method set may use facts alone with an empty
↪  knowledge
list. You may select any eligible expert, including methods outside the
retrieved set. If the evidence cannot support any useful choice, return
{"abstain":true,"order":[],"reasons":[]}; an explicit cost-only fallback
↪  will
be used. No other fields. Candidate presentation order is not a
↪  preference.
```

The sentence about replacing hand-coded biases concerns the **final choice**. Those biases still choose the upstream card set through D8–D9. It would therefore be inaccurate to describe the whole system as free of numerical heuristics.

For a non-abstaining response, the implementation checks exactly three distinct eligible string IDs, exactly three matching reason objects in the same order, the exact permitted key sets, at least one supplied fact ID per choice, valid supplied knowledge IDs, and a reason string of 1–450 characters. For a method covered by a method-specific card, a nonempty supplied knowledge citation is required. A method outside the covered set may use an empty knowledge list. **The validator does not require the cited card to be that method's own card, and does not verify the reasoning's semantic correctness.** Explicit abstention is accepted and invokes fallback; in the current implementation, its order/reason fields are checked for emptiness by truth value rather than by a strict empty-list type check.

Only a provider response with `finish_reason="stop"` enters JSON/content validation. An incomplete response, provider/transport error, invalid structure, or explicit abstention uses the first three candidates in the fixed construction-only order

$$\text{sortkey}(e) = \big(\mathbf{1}\{W_e \text{ missing}\},\ W_e \text{ if known else } 0,\ j_e\big), \tag{D10}$$

where j_e is its original candidate index. This fallback uses raw work, without the PAe multiplier, family priors, K penalty, or diversity objective. The request/response records distinguish accepted content, abstention/fallback, and the failure category. Asset mismatch, unsupported input, or a mismatched request binding is rejected before this LLM-content fallback; it is not silently converted to a valid portfolio. All 196 archived requests contain sixteen eligible experts, so a three-choice decision is defined for this evaluation. The portfolio builder requires three distinct IDs; inputs with fewer than three eligible experts remain outside the evaluated deployment contract.

D.6. Call settings, immutable artifacts, and costs

Item	Current setting
Protocol ID	`mcnd-agent-three-cards-fresh-light-v2`
Requested model	`deepseek-flash`
Thinking / temperature	Disabled / 0
Output cap	1,400 tokens
Response format / streaming	JSON object / false
Transport timeout argument	40 seconds
Automatic retries	0

Item	Current setting
Calls per fresh decision	One attempted provider call; no second LLM relay decision
Experiment relation	Lightweight MCND configuration; RAW included as differently configured system comparisons (Appendix H)

Before each call, preparation binds the input, configuration, serialized messages, exclusions, and fallback order. Started/response receipts prevent automatic duplicate requests and preserve payload hashes, reported model, usage, finish reason, and elapsed time. The production audit records `deepseek-flash` as both requested and returned alias; it does not identify immutable backend weights.

All 196 production calls have matched requests, responses, and provider totals in the acquisition ledger. Missing usage details remain unknown; cached-input and reasoning-output subsets are not counted twice. Historical development calls are separate. Preparation includes its verification and analysis components, so its total is not added to them again. Appendix B defines accounting; Appendix H reports the system-level token comparison.

The portfolio builder copies the selected frozen expert packages. Only the benchmark package manifest and primal-integral protocol are adapted; formulation, solver, recovery, and recorder source are preserved. The artifact includes three packages, the coordinator, request/response receipts, route, file inventory, and archive hashes. It depends on the deployed benchmark adapter rather than being a new solver synthesized by the LLM.

D.7. Ordered execution and the program clock

For order (e_1, e_2, e_3), the first expert receives the remaining allowance. An early return without accepted completion may permit the next expert to cold-start. Execution is sequential, without equal budget slices, preliminary probes, or transferred solver state.

A validated restricted PATH optimum may trigger **at most one** cold relay to the first untried original-scope expert in the order, using the remaining budget and no further LLM call. PATH retains ordinary model-completion credit; a later improved objective is associated with its own witness. Completion and feasibility follow Appendix B.

Let b_j be a launched stage's allowance and p_j its measured child-program runtime. The adapter credits $\tau_j = \min(b_j, p_j)$ for a non-timeout in-budget return and $\tau_j = b_j$ otherwise. A validated stage-local event at $u \le \tau_j$ enters the portfolio trace at

$$t_j(u) = \sum_{h<j} \tau_h + u, \qquad \sum_j \tau_j \le H = 3600. \tag{D11}$$

Credited time is capped; measured deadline overruns are retained separately. Startup, preprocessing, construction, solving, recovery, and persistence consume program time. Analysis, card construction, LLM calls, parent validation, and coordination are external.

Under Appendix B's failure policy, a genuinely failed child does not erase earlier legal Agent trajectory or completion. Appendix E describes the adopted attempts, including the unresolved c52 boundary case.

D.8. Evidence status and reproducible source boundary

Claim	What supports it	What remains unestablished
Current decision algorithm, exact parameters, prompt, and fallback	Direct source reading; raw-file hashes; provenance comparisons	Generality beyond the accepted input/deployment contract
DA/FA structural work	Closed-form pinned constructor count under eligibility conditions	Presolved size, runtime, or search difficulty
PA/PATH work, distinction, and load	Deterministic sample and disclosed equations	Sampling coverage, feasibility, LP strength, calibrated success, or regret guarantees
Three-card mechanism	Active packet/card builder and response contract	Semantic correctness of reasons; isolated causal effects beyond the delivered knowledge-configuration comparisons
MCND Agent implementation and execution evidence	Versioned code and adopted run records	Unseen-instance performance and domain-transfer claims

The clarification package binds evaluated code, development records, and pre-call requests. The implementation audit records source comparisons.

D.9. Relation to the shared FROM-ROUTE architecture

MCND instantiates Figure 2 using sampled path/aggregation evidence, hidden recommender scores, and optional restricted-completion relay. Its 3,600-second allowance is a domain parameter. Appendix C.4 compares the three adapters.

D.10. Observed use of the three-card context

We matched all 196 accepted production decisions to their saved requests and responses by SHA-256. All were accepted LLM outputs rather than response fallbacks. Each selected method set is compared with that instance's card-covered set. MCND cards are displayed alphabetically with recommender rank hidden, so set agreement does not imply agreement in execution order or identical sets across instances.

Observed selection behavior	Decisions / 196
Selected methods match that instance's three card-covered methods, ignoring order	170
At least one selected method is outside the card set	26
First selected method is outside the card set (subset of the preceding row)	20

Across the 588 selected positions, 41 refer to methods outside the card set. First choices are PATH-K10 (177), FA-O (11), FA-D (7), and PAe-D-K1 (1). These observations document the LLM's selection of both card-covered and off-card methods.

Two request-response examples illustrate this distinction. In `r01.2`, the cards cover DA, FA-O, and PAe-D-K1, while the LLM selects PATH-K10, FA-O, and PAe-D-K1. Its first reason cites the path construction proxy of 280 and explicitly notes that feasibility is unknown and that the restricted pool may omit an original-problem optimum. In `r01.1`, the same card-covered methods produce the order PAe-D-K1, FA-O, DA. The first reason cites proxy work of about 618.4 and retained distinction of about 0.738, while acknowledging that neither is a measured LP bound. These are summaries

of recorded rationales, not verified runtime predictions. The examples are the lexicographically first instances with an off-card first choice and with PAe-D-K1 first, respectively; no outcome-based selection was used.

APPENDIX E. MCND SYSTEM EVALUATION ON THE BENCHMARK CORPUS

E.1. COMMON-REFERENCE ENDPOINTS

Evaluation uses all 196 instances and 3,600 program seconds. Full, the fixed matrix, RAW, and ablations share a main-batch dispatch cap of 30. All 22 methods use the same reference, incorporating five improved values from the adopted RAW and ablation trajectories. The table below compares Full with fixed experts; E.2 and H report RAW, and F reports ablations.

Table convention. Bold marks within-domain best reported values, including ties, not statistical significance. Arrows give metric direction. Diagnostic counts and incomplete acquisition totals are not ranked.

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Full	**0.00671945**	**189/196**	**196/196**
FA-O	0.01564087	141/196	**196/196**
FA-D	0.01623354	142/196	**196/196**
PAi-D-K1	0.01826949	140/196	**196/196**
PAi-O-K1	0.01925621	139/196	**196/196**
PAe-O-K1	0.02382706	137/196	**196/196**
PAe-D-K1	0.02519640	138/196	**196/196**
PAi-O-K5	0.03019882	134/196	**196/196**
PAi-D-K5	0.03259110	135/196	**196/196**
PAe-O-K5	0.04086018	136/196	**196/196**
PAe-D-K5	0.04283971	134/196	**196/196**
PAi-D-K10	0.04483892	133/196	**196/196**
PAi-O-K10	0.04864955	133/196	**196/196**
PAe-O-K10	0.05222658	132/196	**196/196**
DA	0.05243263	132/196	**196/196**
PAe-D-K10	0.05554298	133/196	**196/196**
PATH-K10	0.11414810	169/196	177/196

Model completion includes restricted PATH and generated-model completion with independent original-feasibility validation. PATH-K10 has the highest fixed-expert completion count, while FA-O has the lowest fixed-expert NPI; these endpoints express different budgeted outcomes. Appendix B.1 explains why completion, original-space NPI, and coverage are reported together. All fixed experts are displayed; their order is descriptive, not an Agent selection rule.

E.2. TIME CURVES

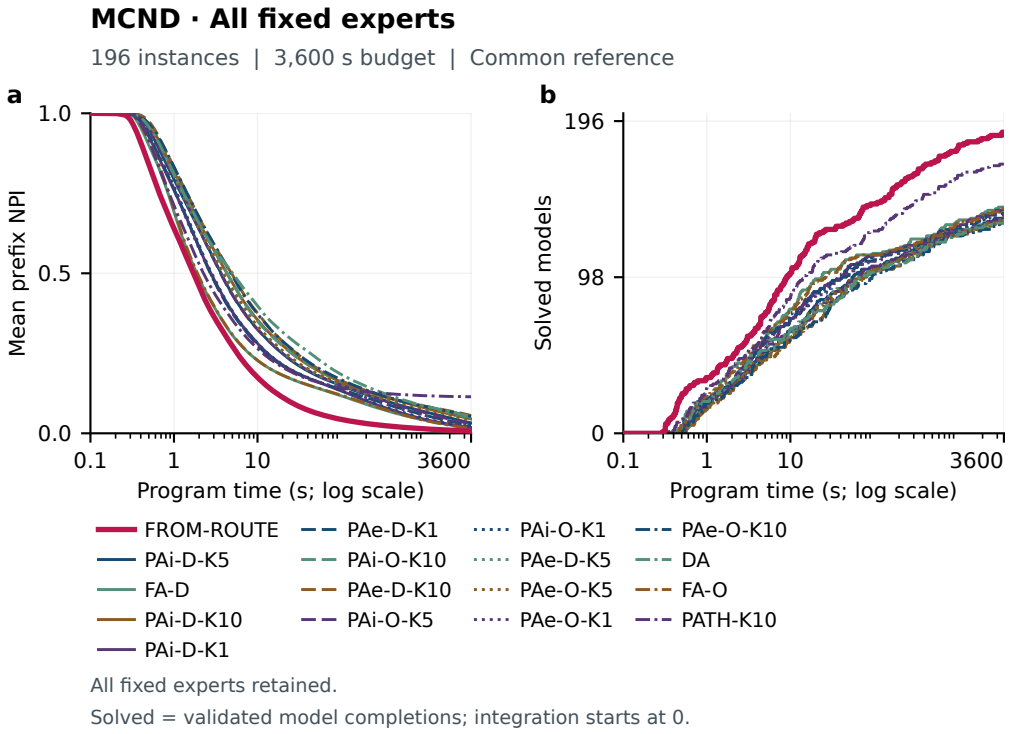

Figure E1: FROM-ROUTE (Full) versus all sixteen fixed experts on all 196 MCND instances. Panel a shows mean prefix NPI; panel b shows cumulative solved models, defined as validated model completions and including restricted PATH completions. Time is logarithmic from 0.1 to 3,600 seconds; NPI is linear from 0 to 1 and integration starts at zero. All traces use the current common reference. The system view below additionally compares the RAW conditions.

E.3. ADOPTED ATTEMPTS AND RECOVERY

Full adopts 193 original portfolios and three designated whole-portfolio replacements. Its frozen decisions and adopted trajectories are used throughout the MCND comparisons. The five RAW/ablation groups comprise 980 independently audited executions; rejected candidates receive no quality or completion credit. Appendix F records the remaining budget-boundary case.

E.4. PRODUCTION ACQUISITION AND EXTERNAL COSTS

The 196 production calls have complete provider totals, comprising 444,390 input and 53,634 output tokens, or 498,024 overall and 2,540.94 per instance. Requests and responses are matched. Cached input is included once; absent optional reasoning details remain unknown. These retained-call totals exclude incompletely documented development costs.

The 196 accepted decisions are frozen inputs to solver execution. Their acquisition cost is counted once per instance, independently of solver execution, and is complete for the reported production calls. Program-time trajectories and acquisition tokens quantify different phases; neither is presented as complete service latency.

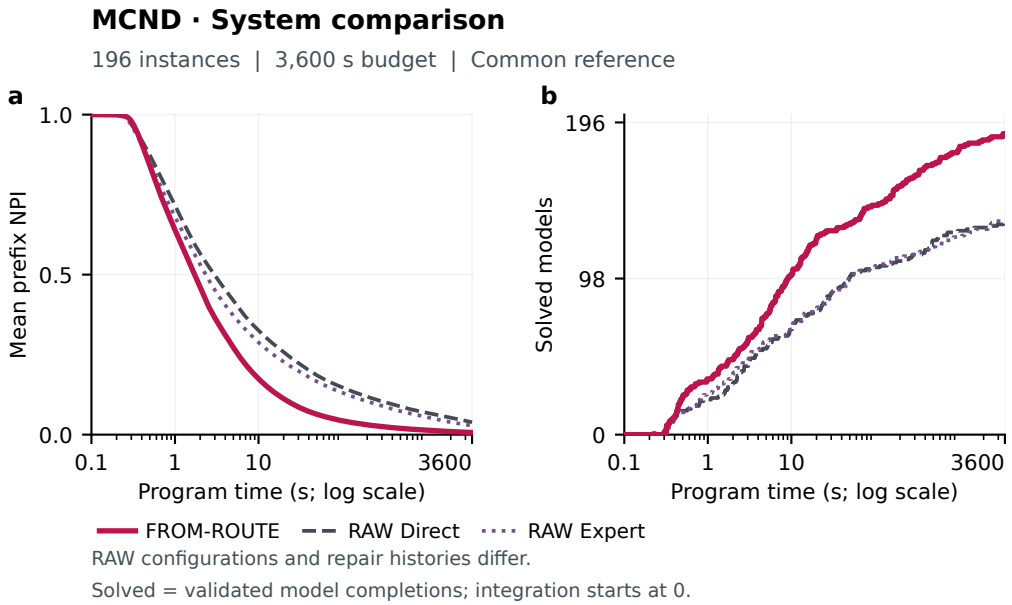

Figure E2: Panel a: mean prefix NPI, integrated from zero. Panel b: cumulative validated model completions. Both use logarithmic program time from 0.1 to 3,600 seconds. The display compares FROM-ROUTE (Full) with both RAW systems, whose main batches share a dispatch cap of 30. The RAW comparisons use different generation configurations.

E.5. Reproducibility

The unified scoring summary binds Full, fixed experts, RAW, variants, and the rule replay to one reference. The adoption record preserves source hashes, scoring checks, and excluded candidates.

E.6. Deterministic rule Top-1 replay

For each of the 196 MCND instances, the frozen non-LLM recommender in D.3 selects its internally top-ranked expert, before consulting solver results. We retrieve that instance–expert result from the full fixed-expert matrix and aggregate NPI, model completions, and valid coverage using the same 22-method references. This replay requires no new solver or LLM calls.

System	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Rule Top-1 replay	0.03004714	185/196	193/196
Full	**0.00671945**	**189/196**	**196/196**

The rule selects PATH-K10, FA-O, FA-D, and DA on 103, 72, 11, and 10 instances. Full and the fixed matrix use a main-batch dispatch cap of 30. Full improves NPI and adds four completions over rule replay.

Appendix F. MCND LLM-routing ablations

F.1. Questions and interventions

The knowledge conditions preserve instance facts, eligible candidates, system instruction, and response structure, while varying mechanism context. No relay reuses Full's decisions with restricted-completion continuation disabled.

Condition	Knowledge presented to the LLM	Routing calls	Execution policy
Full	Instance-conditioned three-card context	Original 196 production calls	Failure fallback and restricted-completion relay
No RAG	Facts and candidate descriptions, without the mechanism-card context	196 fresh calls	Failure fallback and relay
Generic RAG	Generic mechanism knowledge instead of the method-specific three cards	196 fresh calls	Failure fallback and relay
No relay	Identical bound full-condition response and ordered triple	Original calls reused; no additional call	Failure fallback retained; restricted-completion relay disabled

No relay matches Full's response hashes and orders on all 196 instances. The Analyzer remains unchanged. Knowledge content, membership, and length vary together, so this is not a length-matched language-comprehension test.

F.2. Completed execution and shared-reference scoring

The three ablations comprise 588 independently audited portfolios, 1,007 stages, and 16,472 captured candidate records. Two invalid candidates in No RAG are excluded from scoring. For No RAG on c63, configuration and trace construction exhaust the final stage's remaining budget; its complete valid prefix is retained without completion credit. Every condition retains all 196 instances.

All 4,312 rows across 22 methods are rescored against the updated common reference. Scores follow Appendix B, including ordinary credit for validated restricted PATH completion.

Table convention. Bold marks within-domain best reported values, including ties, not statistical significance. Arrows give metric direction. Diagnostic counts and incomplete acquisition totals are not ranked.

Condition	Mean NPI ↓	Validated model completions ↑	Valid coverage ↑
Full	**0.00671945**	**189/196**	**196/196**
No RAG	0.00882192	188/196	**196/196**
Generic RAG	0.00882727	188/196	**196/196**
No relay	0.01728865	188/196	**196/196**

Full has the lowest observed mean NPI. No RAG and Generic RAG have equal completion counts and close mean NPI; this does not establish a benefit for generic context alone.

F.3. Time curves

F.4. Pairing and interpretation

Appendix B.5 records the execution settings.

The knowledge conditions have 392 accepted acquisition calls in total, with no automatic retries or missing total usage; their combined provider total is 887,517 tokens. Requested/served model alias, temperature, thinking setting, output limit, and recorded system fingerprint match the full batch;

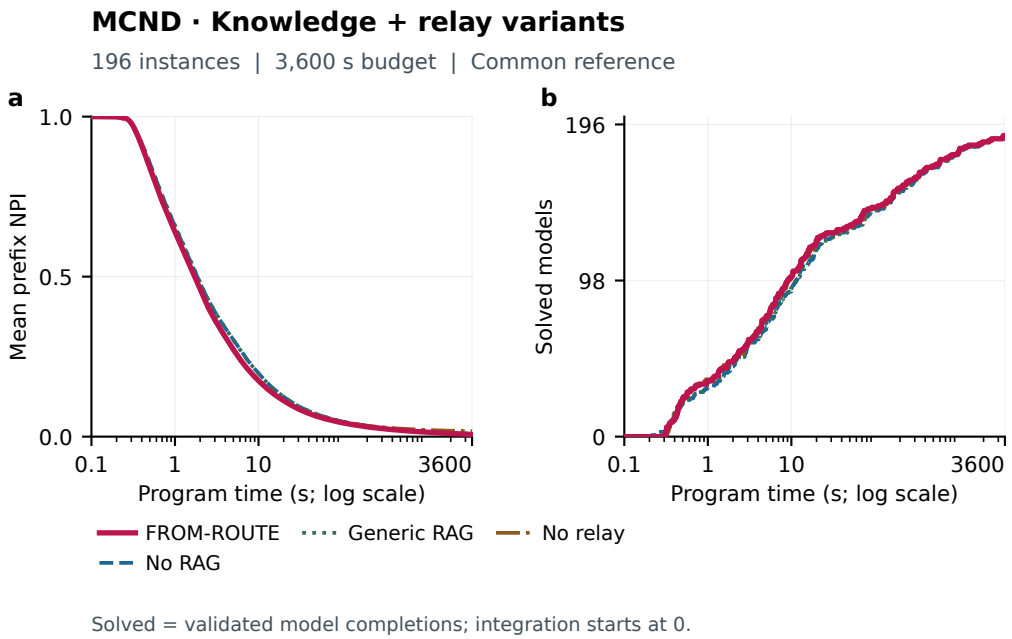

Figure F1: Panel a: mean prefix NPI. Panel b: cumulative validated model completions. FROM-ROUTE (Full) and its three variants are compared, using the same enlarged reference as Appendix E. Time is logarithmic from 0.1 to 3,600 seconds; integration begins at zero, and the NPI ordinate spans 0-1. Execution settings are specified in Section F.4.

a shared alias/fingerprint does not guarantee immutable provider weights. Acquisition tokens and routing counts are:

Condition	Attributable routing tokens ↓	Acquisition calls	PATH-K10 first choices
Full	498,024	196	177/196
No RAG	**389,261**	196	192/196
Generic RAG	498,256	196	196/196
No relay	498,024; same Full acquisition reused	0	177/196

Generic context contains five fixed mechanism cards; the full context has three instance-conditioned cards. No RAG still retains candidate descriptions and all instance facts. The two knowledge conditions have the same first choice on 192 instances, but the same complete ordered triple on only 20. Aggregate completion equality does not mean identical routing.

No relay adds no new call but retains Full's acquisition cost. Deduplicated production acquisition across Full and the two fresh knowledge conditions totals 1,385,541 tokens. These totals exclude historical development; comparison with RAW remains the differently configured system comparison in Appendix H.

F.5. Sources and scope

The acquisition archive retains calls and usage; the execution archive retains trajectories and audits. The scoring record specifies shared references; the pairing/cost audit records stage qualifications.

Appendix D documents the fixed parameters; Appendix G reports VCP/RCPSP knowledge conditions without post-completion relay.

F.6. ORIGINAL-SCOPE MCND EXPLORATION

We additionally evaluate **MCND-clean**, which removes PATH-K10 and retains the 15 original-scope experts in Appendix A.1. Its dedicated Analyzer and prompt present an input-derived shortlist of three experts to the LLM; only the first-ranked expert is executed, without relay or failure fallback. Full supplies instance-conditioned mechanism cards, No RAG omits them, and Generic RAG supplies generic mechanism knowledge. These three conditions share the shortlist and LLM configuration. Each is executed on all 196 instances with a 3,600-second budget, one solver thread, and a dispatch cap of 30. Here, a validated optimal completion certifies the original MCND problem.

The table compares the three clean conditions with FA-O and FA-D, respectively the lowest-NPI and highest-completion fixed experts among the 15 original-scope methods. All NPI values in this subsection use a shared reference augmented with the validated clean incumbents; other sections retain their existing reference. Every listed method has valid coverage on all 196 instances.

Original-scope method	Mean NPI ↓	Original-problem completions ↑
FA-O	0.01564666	141/196
FA-D	0.01623925	142/196
MCND-clean No RAG	0.01548315	141/196
MCND-clean Generic RAG	0.01574314	140/196
MCND-clean Full	**0.01373836**	**143/196**

MCND-clean Full reduces mean NPI by **12.2%** relative to FA-O and completes one more instance than FA-D. Against its No RAG and Generic RAG conditions, NPI decreases by **11.3%** and **12.7%**, with two and three additional completions, respectively. Thus, instance-conditioned mechanism context also improves the observed routing outcomes when every selectable expert targets the original problem.

Figure F2 shows the corresponding quality and completion trajectories alongside the PATH-inclusive Full system.

Under the same subsection reference, the PATH-inclusive Full system achieves NPI **0.00672561**, **51.0% lower** than MCND-clean Full, and records **189/196 validated model completions**. PATH restricts the candidate path pool, but its recovered solutions remain feasible for the original MCND problem. Its completion certificate applies to the restricted model, so 189 model completions and 143 original-problem completions describe different certificate scopes. Together with the lower original-space NPI, the higher model-completion count illustrates the engineering value of including restricted formulations in a budgeted system. This is a comparison of system designs, including their routing and continuation policies, rather than an isolated estimate of PATH's contribution.

The clean execution and scoring record binds the three complete batches, validated trajectories, and shared-reference comparison.

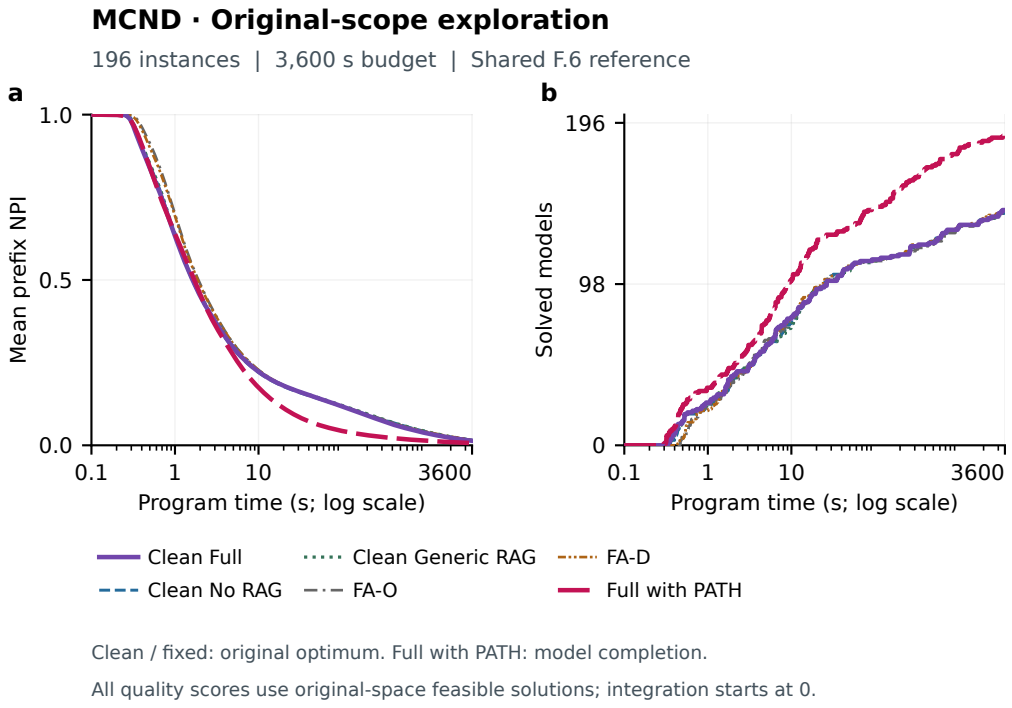

Figure F2: MCND original-scope exploration. Panel a: mean prefix NPI, integrated from zero to each displayed time and averaged over all 196 instances. Panel b: cumulative validated model completions. The three MCND-clean conditions, FA-O, FA-D, and the PATH-inclusive Full use the shared reference of Section F.6. Time is logarithmic from 0.1 to 3,600 seconds; NPI is linear from 0 to 1. Clean and fixed-expert completions certify the original problem; Full with PATH also includes restricted-model completions. Curves describe the adopted runs; no repeat-run uncertainty is estimated.

APPENDIX G. VCP AND RCPSP ADAPTATIONS AND ADOPTED RESULTS

G.1. SHARED INTERFACE AND DOMAIN-SPECIFIC EVIDENCE

VCP and RCPSP instantiate the shared architecture. Appendix C.4 compares domain interfaces; Appendix A.2–A.3 defines their formulations.

The two domains share structural analysis, three mechanism cards, and LLM selection from the complete eligible library. RCPSP uses precedence and resource structure, critical-path/schedule information, time windows, and construction proxies. VCP uses graph structure, a greedy palette, validated clique information, and construction counts. Each card combines a mechanism, its applicability limits, and the current instance evidence.

Both domains expose the recommender's soft prior in natural language; relative score/softmax weights describe preference, not calibrated success probabilities. This differs from MCND's name-ordered cards with scores and rank hidden. The three-card selection is not a hard restriction on the action space: all sixteen RCPSP or eight VCP experts remain eligible. Section G.5 specifies the adopted adapters and links their exact prompt construction and score parameters.

The LLM emits Top-3. The first expert receives all remaining program time; an early unsuccessful return may lead to ordinary cold-start fallback. There is no budget-sliced or post-completion relay in either domain and no corresponding relay ablation. In the adopted knowledge runs, later fallback stages contribute no new credited quality point or completion, but their actual execution and costs remain recorded.

G.2. VCP RESOURCE99

Following the initial matrix evaluation, the VCP study adopted the memory-motivated scope detailed in Appendix A.3.4: 99 of 125 graphs, with at most 1,000 vertices and 50,000 edges, and eight fixed configurations excluding exact OBDD. The 792-cell fixed matrix follows the memory-protection and designated-rerun policy in Appendix A.3.4, preserving headroom for memory-intensive formulations. Two REP memory-allocation failures retain NPI 1 and no completion credit.

All 1,287 rows across 13 methods use the shared reference incorporating RAW, which improves 17 previous reference values. Rescoring changes neither adopted trajectories nor Full's 56 completion labels.

Table convention. Bold marks within-domain best reported values, including ties, not statistical significance. Arrows give metric direction. Diagnostic counts and incomplete acquisition totals are not ranked.

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Full	**0.04484701**	**56/99**	**99/99**
No RAG	0.05402748	39/99	**99/99**
Generic RAG	0.05236397	37/99	**99/99**
POPH	0.05252056	49/99	**99/99**
POP	0.05410762	27/99	**99/99**
POP1	0.05703445	32/99	**99/99**
POPH1	0.05796805	49/99	**99/99**
POPH2	0.05818991	54/99	**99/99**
POP2	0.06168043	30/99	**99/99**
ASS	0.06596804	36/99	**99/99**
REP	0.08674107	28/99	97/99
RAW Direct	0.27862455	50/99	79/99
RAW Expert	0.39114710	51/99	79/99

All 198 knowledge-variant portfolios have adopted audited views. One missing terminal-stage ledger follows an uninitialized fallback at a residual fraction of a second. Its legal earlier prefix is retained; the empty tail is charged the remaining budget and contributes no invented point or completion. The raw missing-ledger evidence remains in the archive. The VCP audit binds the joint delivery to the new common-reference RAW delivery.

G.3. RCPSP ADOPTED FULL EXECUTION

The PSPLIB corpus comprises 480 instances each from j30, j60, and j90, and 600 from j120, totaling 2,040. Full covers all 2,040 instances, adopting each designated replacement as a whole execution rather than selecting the better old/new score. It yields 1,216 completions and valid coverage on all instances. One capture-boundary case retains its committed valid prefix; uncommitted artifacts contribute neither quality points nor completion credit.

The shared reference retains validated bounds and incorporates one further improvement from Full. Every fixed expert and both knowledge variants are rescored under the same 19-method view. Full and both knowledge variants use a 120-second budget, one solver thread, a 1,536 MB solver memory parameter, and a dispatch cap of 60. RAW and relay comparisons are outside the RCPSP study design.

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Full	**0.02423083**	**1216/2040**	**2040/2040**
No RAG	0.03217380	907/2040	**2040/2040**
Generic RAG	0.02971166	982/2040	**2040/2040**
DT	0.02513753	1189/2040	**2040/2040**
SDDT	0.02552861	1166/2040	**2040/2040**

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
KOP-CT1	0.02970845	929/2040	**2040/2040**
DDT	0.03109796	1104/2040	**2040/2040**
KOP-CT2	0.03137268	902/2040	**2040/2040**
MFS	0.03705456	535/2040	**2040/2040**
OODDT	0.03763999	923/2040	**2040/2040**
SEQCT	0.04036162	744/2040	**2040/2040**
FCT-ALT	0.04165562	689/2040	**2040/2040**
BIA-CORRECTED	0.04255293	808/2040	**2040/2040**
UNIT	0.04582325	345/2040	**2040/2040**
FCT	0.04614040	343/2040	**2040/2040**
OOE	0.04929394	128/2040	**2040/2040**
RSEE	0.04988728	160/2040	**2040/2040**
SEE-CORRECTED	0.05095153	75/2040	**2040/2040**
IEE	0.05142518	19/2040	**2040/2040**

The 4,080 knowledge-variant portfolios retain 532 capture-boundary recovery records: 494 committed-prefix recoveries and 38 empty-bootstrap tail cases. Empty tails consume their remaining budget and add no quality or completion. The records preserve the original capture exceptions. The current Full delivery, joint archive, and independent audit identify the adopted evidence.

G.4. Production acquisition and interpretation

Domain	Condition	Adopted calls	Provider total tokens ↓
VCP	Full	99	568,428
VCP	No RAG	99	**190,650**
VCP	Generic RAG	99	235,777
RCPSP	Full	2,040	11,375,456
RCPSP	No RAG	2,040	**5,857,256**
RCPSP	Generic RAG	2,040	6,898,379

Provider totals are complete for the frozen decisions of each condition. Full uses more tokens than either knowledge variant in both domains, alongside better observed outcomes. This trade-off differs from the RAW comparison. These token totals measure LLM acquisition, separately from solver computation and system development.

VCP and RCPSP use domain-specific adapters without post-completion relay. Within each domain, the knowledge conditions retain the same structural evidence and compare the combined mechanism-context designs.

G.5. Adopted Analyzer and Decider specification

The shared interface has domain-specific deterministic rules for supplying reference cards and, in VCP/RCPSP, a soft ordering prior. The final accepted expert sequence is selected through the LLM interface; the recommendation is not a restriction to those three experts. Appendix I supplies the adopted score functions, parameters, prompt strings, and acceptance rules. Appendix C's illustrative cards are not runtime prompts.

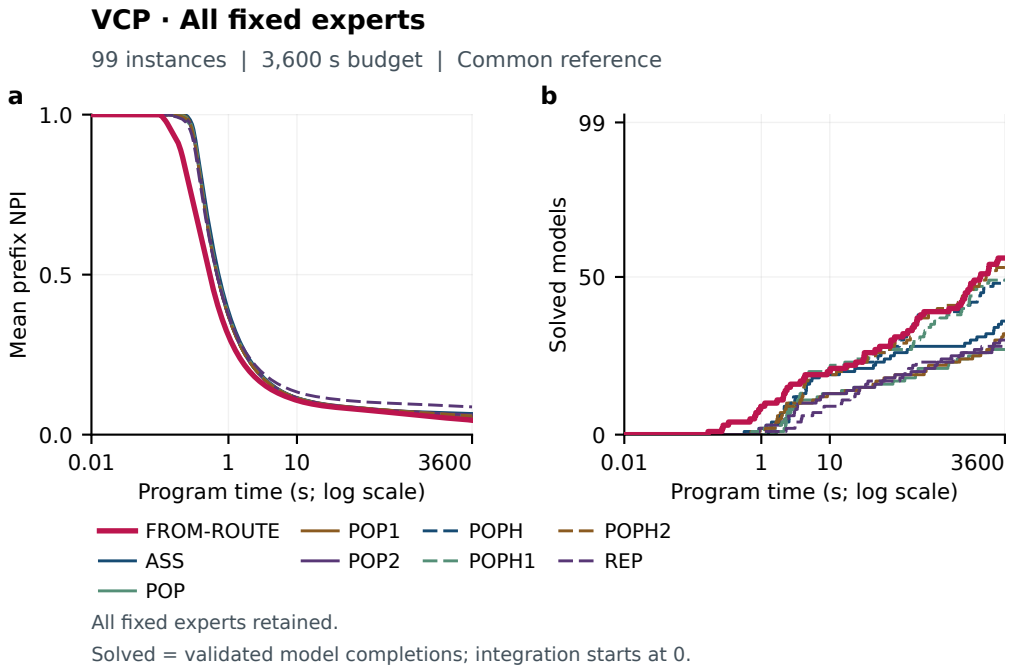

Figure G1: FROM-ROUTE (Full) versus all eight retained VCP fixed experts on the 99-instance resource99 corpus. Panel a shows mean prefix NPI; panel b shows solved models, defined as the validated model completions of Appendix B. Time is logarithmic from 0.01 to 3,600 seconds; NPI is linear from 0 to 1 and integration starts at zero. All traces use the common 13-method reference. The selected system comparison below additionally includes RAW.

Adapter element	VCP	RCPSP
Input evidence	Degree, density, components, degeneracy, triangles, greedy palette, clique bound, distinguished-vertex neighborhood, and saturation-ordered palette	Precedence closure and order strength, duration normalization, serial and schedule-generation horizons, time windows, resource energy/pressure, and conflicting pairs/triples
Construction estimates	Variables, rows, and nonzeros for eight formulations	Binary/continuous variables, rows, nonzeros, and big-M exposure for sixteen formulations
Bounded analysis	Representative conflict enumeration stops at two million operations and reports an upper-bound substitute; saturation coloring is omitted above 2,000 vertices	Schedule generation has a two-million-operation cap with serial-schedule fallback; triple checks use up to 256 deterministically sampled triples; unavailable MFS construction cost is replaced by the largest finite estimate
Candidate presentation	SHA-256 order of `vcp-v1`: plus expert ID	SHA-256 order of `rcpsp-v1`: plus expert ID

Analyzer estimates are distinguished from measured runtime. VCP's alternative coloring is evidence only and is not supplied to the solver. Original-space validation still determines whether a solver witness is eligible for scoring.

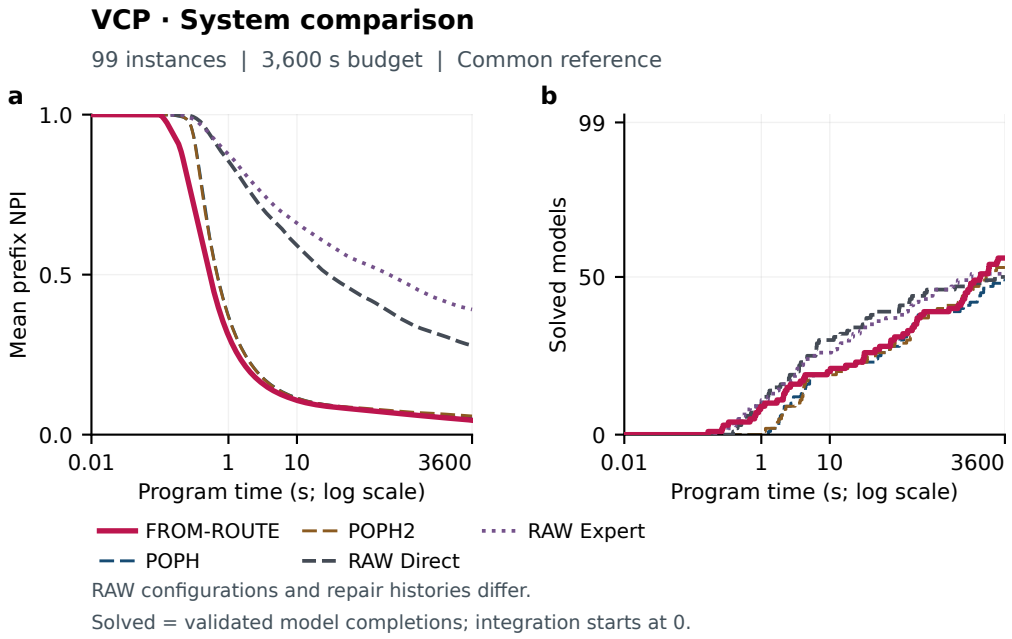

Figure G2: VCP system comparison of FROM-ROUTE (Full), POPH (best fixed endpoint NPI), POPH2 (most fixed-model completions), and both RAW systems. Panels a and b show mean prefix NPI and cumulative validated model completions under the same reference, 99-instance denominator and logarithmic time range as Figure G1. RAW model configurations and generation/repair histories differ from those of the Agent; this is a system-level comparison, not a matched causal estimate.

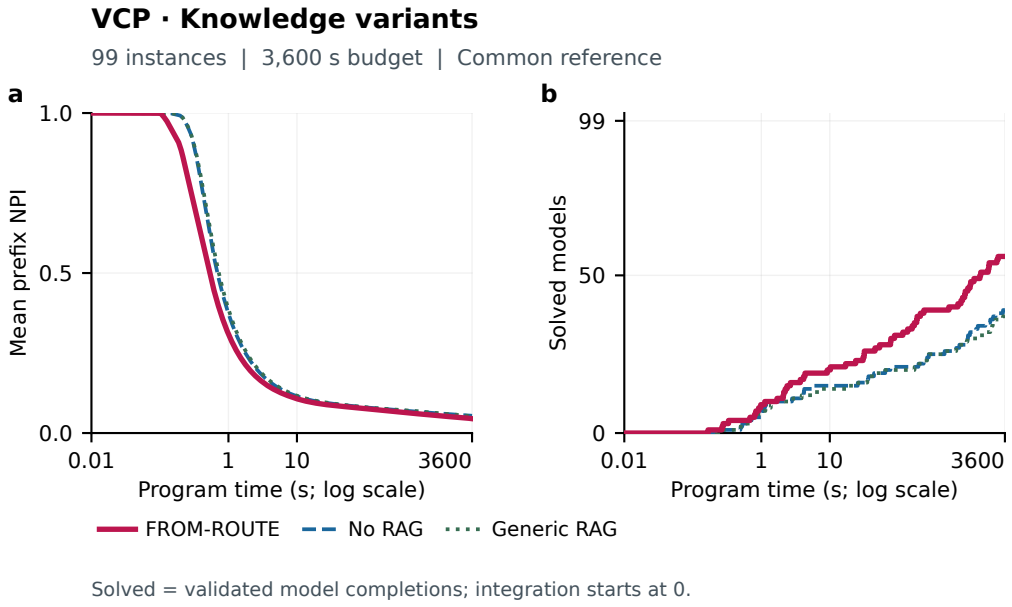

Figure G3: VCP knowledge conditions: FROM-ROUTE (Full), No RAG, and Generic RAG. Panels a and b show mean prefix NPI and cumulative validated model completions on all 99 instances, under the same 13-method reference and 3,600-second budget as Figures G1–G2. Time is logarithmic from 0.01 seconds, and integration starts at zero. This is a knowledge-context comparison without post-completion relay; Section G.5 records the execution settings.

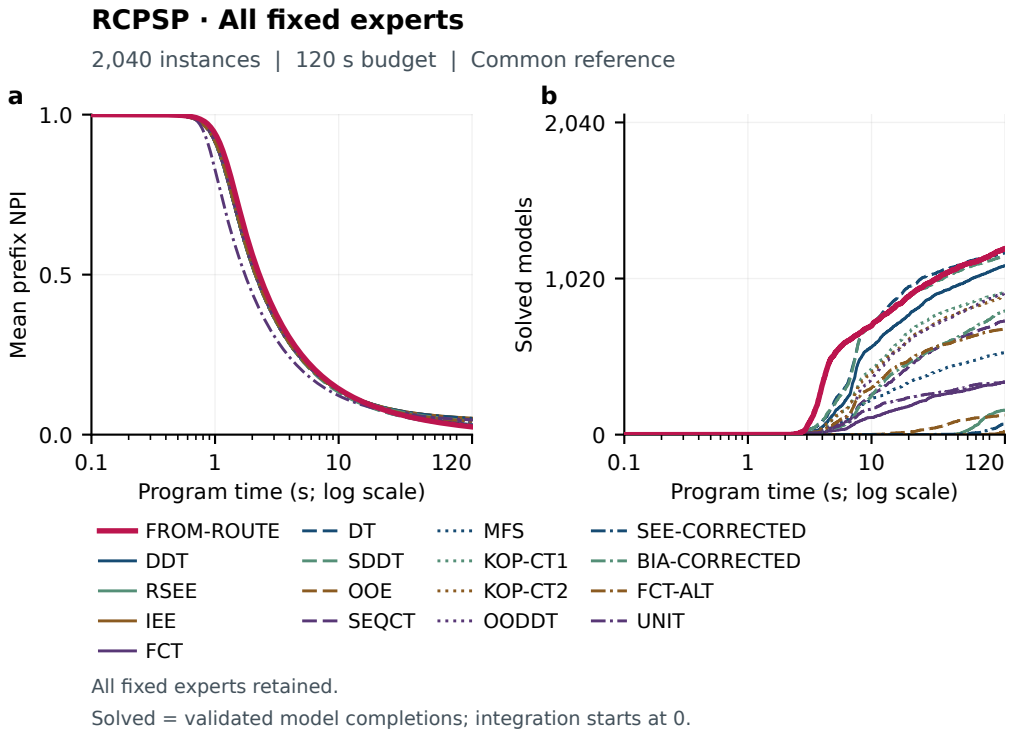

Figure G4: FROM-ROUTE (Full) versus all sixteen RCPSP fixed experts on all 2,040 instances. Panels a and b show mean prefix NPI and cumulative validated model completions. Time is logarithmic from 0.1 to 120 seconds; NPI is linear from 0 to 1 and integration starts at zero. All traces use the updated common 19-method reference. RCPSP has no RAW condition.

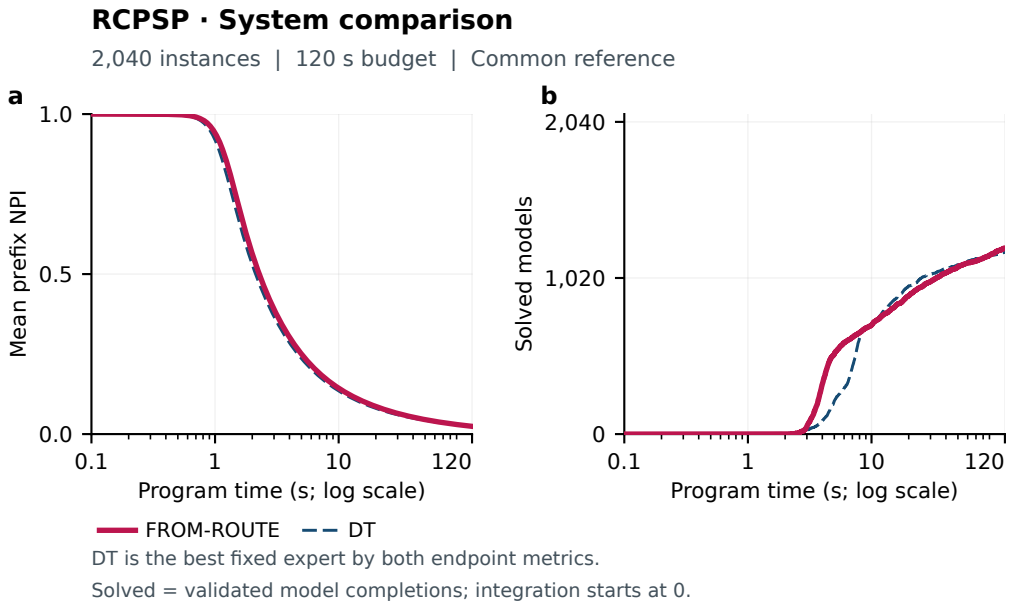

Figure G5: RCPSP system comparison of FROM-ROUTE (Full) and DT, the best fixed configuration by both endpoint mean NPI and validated model completions. Panels a and b retain the reference, 2,040-instance denominator and logarithmic 120-second horizon of Figure G4.

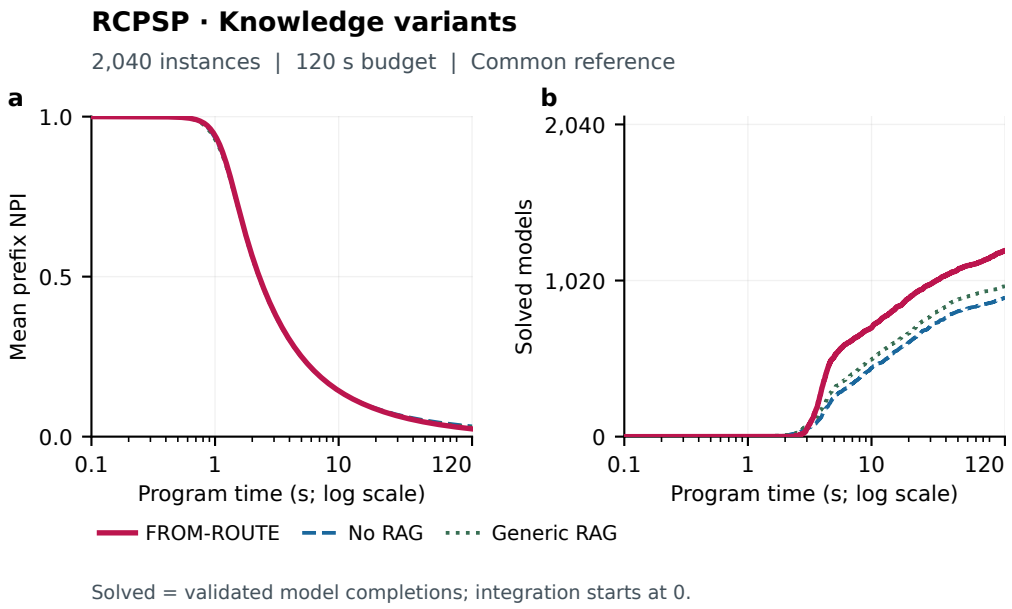

Figure G6: RCPSP knowledge conditions: FROM-ROUTE (Full), No RAG, and Generic RAG. Panels a and b show mean prefix NPI and cumulative validated model completions under the updated common 19-method reference, retaining all 2,040 instances and the 120-second horizon. Time is logarithmic from 0.1 seconds, before the earliest positive recorded point; integration starts at zero. No RCPSP relay experiment is included.

For additive expert scores s_e, both adapters choose a three-method set maximizing the sum of scores plus 0.15 per distinct formulation family. Selected cards are ordered by descending score, with expert-ID tie breaking. Full exposes the score terms, rank, and weights proportional to $\exp(4s_e)$ within that set. These are heuristic preferences, not calibrated success probabilities. Its instructions retain the preference unless current instance evidence supports an exception; all eligible experts remain selectable.

RCPSP uses a construction term $-\log(1+(w_e+0.05z_e)/1000)$ and a 0.5-weighted order-strength Œ resource-pressure term for strong-precedence formulations. Here w_e counts variables and rows and z_e estimates nonzeros. The adopted big-M coefficient is zero. VCP additionally uses palette, activation, neighborhood, ordering, and hybrid-link terms; Appendix I.1 gives their coefficients and conditional powers. These terms express domain-specific relationships between instance structure and model construction. Appendix D.3 describes the coefficient-selection procedure shared by the three domains.

No RAG removes cards. Generic RAG selects five of six generic cards by deterministic instance-based relevance, then hides their selection weights. Full instead supplies three method-specific cards and the explicit prior above. Thus these comparisons change card content, membership, length, and prior presentation together. They are not isolated tests of language understanding. The exact JSON checks and deterministic abstention/error fallback appear in Appendix I.4.

The new knowledge requests use a 1,400-token output allowance, temperature zero, disabled thinking, and no automatic retry. Full executes its frozen decisions. Appendix B.5 lists execution settings; Section G.2 specifies VCP fixed-expert memory policies.

APPENDIX H. RAW SYSTEM COMPARISONS AND ACQUISITION ACCOUNTING

H.1. WHAT THE TWO RAW SYSTEMS EVALUATE

RAW Direct and RAW Expert generate modeling kernels within a common executable harness. Expert additionally receives modeling guidance. The comparison evaluates final adopted kernels and compatibility handling, not unrestricted whole-program generation or first-attempt success.

MCND/VCP RAW differ from Full in reusable assets, model/reasoning settings, output allowance, and multi-round acquisition/repair history. They are system-level comparisons under a shared domain program horizon, not model-controlled estimates of routing gains. RCPSP RAW is outside scope because of its two-minute horizon and large instance count.

MCND uses expression-api-v1.2 kernels and a robust adapter, with compatibility recoveries, targeted regeneration, and a designated later Direct replacement. VCP also uses final multi-round packages. The adopted 99-instance manifest excludes other graphs and maintenance attempts from scoring; their records and costs remain in provenance. Earlier whole-program acquisition costs are not substituted for these kernels.

H.2. QUALITY, COMPLETION, AND COVERAGE

Table convention. Bold marks within-domain best reported values, including ties, not statistical significance. Arrows give metric direction. Diagnostic counts and incomplete acquisition totals are not ranked.

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Full	**0.00671945**	**189/196**	**196/196**
RAW Direct	0.03887851	134/196	**196/196**
RAW Expert	0.02825152	135/196	**196/196**

MCND Full and RAW use main-batch dispatch caps of 30; 196 instances; the complete 22-method reference is used.

Method	Mean NPI ↓	Model completions ↑	Valid coverage ↑
Full	**0.04484701**	**56/99**	**99/99**
RAW Direct	0.27862455	50/99	79/99
RAW Expert	0.39114710	51/99	79/99

VCP, 99 instances; the complete 13-method reference is used. The associated system figures are in Appendices E and G. Feasible coverage and model completion are distinct; model completion is judged in the candidate's model and must be interpreted alongside original-space NPI. Generated models may impose restrictions, so a model-optimal return is not asserted to be an original global optimum. The same completion rule applies to all systems.

For each VCP RAW system, the 20 instances without a valid witness are recorded as ordinary timeouts, with NPI 1 and no invalid-program penalty flag. Thus 79/99 coverage is not a first-attempt code-generation success rate.

H.3. FINAL-ARTIFACT ACQUISITION TOKENS

Domain	System	Adopted calls	Input tokens	Output tokens	Total tokens ↓
MCND	Full	196	444,390	53,634	**498,024**

Domain	System	Adopted calls	Input tokens	Output tokens	Total tokens ↓
MCND	RAW Direct	196	4,788,758	6,294,921	11,083,679
MCND	RAW Expert	196	4,944,644	5,854,097	10,798,741
VCP	Full	99	549,512	18,916	**568,428**
VCP	RAW Direct	99	161,356	3,560,326	3,721,682
VCP	RAW Expert	99	213,034	5,116,398	5,329,432

Every adopted call has a reported total. Input includes cached input; output includes any reported reasoning subset. Neither subset is added a second time. These are provider usage units, not monetary charges or normalized hardware work. Expert-library and knowledge development are outside this production-acquisition metric.

RAW generation records contain unsuccessful acquisition and regeneration attempts, including interface incompatibilities and a missing flow-conservation constraint identified during MCND checking. Such attempts can consume additional tokens before a usable kernel is obtained. They are excluded from the table, which counts the final accepted artifact for every evaluated instance, regardless of subsequent solver success or timeout. Thus, the table compares successful artifact acquisition, not a subset of successfully solved instances.

MCND Agent production contains 196 calls and 196 accepted decisions with complete usage and no response-failure fallback. The corresponding RAW overheads do not enter the reported token gap; including them would increase RAW acquisition cost. This observation concerns the recorded acquisition campaigns, not an estimate of expected retry cost across repeated stochastic generations. Historical attempt records remain in the acquisition archive.

H.4. Model/configuration differences

The MCND Agent requests the lightweight `deepseek-flash` configuration, with thinking disabled and a 1,400-token output limit; its knowledge variants preserve that request contract. Both RAW deliveries request `deepseek-v4-flash`, thinking enabled, maximum reasoning effort, temperature 1, and an output allowance of 131,072 tokens. Returned model names must be kept separately from requested aliases: MCND has 391 adopted RAW responses labeled `deepseek-v4-flash` and one later Direct response labeled `deepseek-flash`; all 198 VCP RAW responses are labeled `deepseek-flash` despite the different request settings. A returned alias does not establish immutable weights or matching reasoning compute. The VCP Full/knowledge responses also report `deepseek-flash`; their actual prompt and inference settings remain bound in their own receipts.

These differences are part of the systems evaluated and are disclosed rather than corrected by relabeling old responses. Comparisons use the same domain program horizon and common reference, but do not equate external generation latency or acquisition budget. Precise versions and per-call provenance appear in the MCND audit and VCP audit, supported by the MCND delivery and VCP RAW delivery.

Appendix I. VCP and RCPSP decision specifications

This appendix specifies the parameters, scores, prompts, and response checks for the domain adapters in Appendix G. MCND is specified in Appendix D. All scores use input-derived structural evidence and construction estimates; no measured solver outcome is supplied at decision time. Parameter adjustment is described in Appendix D.3.

The listings use Python notation: `math.log1p(x)` denotes $\log(1+x)$, `int(condition)` is an indicator, and `itertools.combinations` enumerates triples in library order. The Analyzer supplies `a['evidence']` and the eligible candidate records `a['candidates']`. Candidate

work counts variables and rows; nonzeros_proxy estimates matrix nonzeros. Missing RCPSP work uses the largest finite candidate cost. A tie between triples follows enumeration order; selected methods are then ordered by decreasing score and expert ID. Scores specify a card recommendation, while the LLM retains final selection from the full eligible library.

I.1. VCP PARAMETERS AND SCORING

The following global values are the adopted configuration, rather than defaults from the implementation. The common prior strength is 4.

Parameter	Value	Parameter	Value
nonzero_weight	5	neighbor_redundancy_weight	96
strength_weight	2	bound_closure_exponent	24
neighbor_weight	2	regularity_exponent	16
diversity_weight	0.15	hybrid_regularity_exponent	24
pivot	1000.0	regular_neighbor_weight	16
activation_coupling_weight	0.0	diffuse_score_cap	1000.0
neighbor_disconnection_weight	1.0	palette_scale_exponent	2
proof_support_weight	2.0	pure_ordering_weight	48
uncoupled_ordering_weight	48	unresolved_exponent	8
disconnected_palette_weight	0.0	pure_palette_fraction	0
diffuse_ordering_weight	256	representative_palette_weight	0
hybrid_link_weight	8	neighbor_regularity_exponent	8
saturation_proof_weight	0.0		

The graph features use n vertices, m edges, degree mean $\bar{d} = 2m/\max(1, n)$, and degree coefficient of variation $\mathrm{sd}(d)/\max(1, \bar{d})$. Density is $2m/[n(n-1)]$ (zero for $n \leq 1$); transitivity is three times the triangle count divided by $\max(1, \sum_v \binom{d_v}{2})$. Let H be the degree-ordered greedy palette, L the constructed clique size, and H_s the saturation-ordered palette. The palette gap is $(H - L)/\max(1, H)$; the saturation gap substitutes H_s; redundancy is $(H - \min(H, H_s))/\max(1, H)$. Missing saturation gaps fall back to the original palette gap, and missing redundancy is zero. The largest-component fraction divides its vertex count by $\max(1, n)$. Vertex q has maximum degree with ID tie breaking; its neighbor fraction is $d_q/\max(1, n-1)$ and neighborhood density is twice the induced edge count divided by $\max(1, d_q(d_q - 1))$. Candidate flags mark suffix 1/2 as strengthened, suffix 2 as neighbor-strengthened, and prefix POPH as hybrid.

The score is the sum of the following terms. Terms with zero adopted coefficients remain visible to specify the evaluated configuration.

```
def select(a,p):
    scored=[];f=a['evidence']
    for c in a['candidates']:
        cost=c['work']+p.nonzero_weight*c['representation']['nonzeros_pr⌋
        ↪ oxy']
        terms=dict(construction=-math.log1p(cost/p.pivot),
            strengthened_activation=p.strength_weight*f['palette_gap_fra⌋
            ↪ ction']*int(c['strengthened']),
            constructive_proof_support=p.proof_support_weight*(1-f['pale⌋
            ↪ tte_gap_fraction'])*int(c['strengthened']),
```

```
            uncoupled_ordering=p.uncoupled_ordering_weight*f['palette_ga
            ↪  p_fraction']*math.log1p(f['mean_degree']/20)/(1+f['degre
            ↪  e_cv'])**3*max(0,f['transitivity']-f['density'])/max(1e-
            ↪  9,1-f['density'])*int(c['family']=='partial_order' and
            ↪  not c['strengthened']),
            distinguished_neighborhood=p.neighbor_weight*f['q_neighbor_f
            ↪  raction']*int(c['neighbor_strengthening']),
            activation_coupling=-p.activation_coupling_weight*f['palette
            ↪  _gap_fraction']/(1+f['degree_cv'])*int(c['strengthened']
            ↪  ),
            disconnected_neighbor_overhead=-p.neighbor_disconnection_wei
            ↪  ght*(1-f['largest_component_fraction'])*int(c['neighbor_
            ↪  strengthening']),
            disconnected_palette=p.disconnected_palette_weight*(1-f['lar
            ↪  gest_component_fraction'])*(1-f['palette_gap_fraction'])
            ↪  *math.log1p(f['greedy_colors'])*int(c['family']=='partia
            ↪  l_order' and not
            ↪  c['strengthened']),
            diffuse_ordering=min(p.diffuse_score_cap,p.diffuse_ordering_
            ↪  weight*(f.get('palette_redundancy_fraction') or
            ↪  0)*(1-(f.get('saturation_gap_fraction') if
            ↪  f.get('saturation_gap_fraction') is not None else
            ↪  f['palette_gap_fraction']))**2*math.log1p(f['n']/100)/(1
            ↪  +f['degree_cv'])**p.regularity_exponent)*int(c['family']
            ↪  =='partial_order' and not
            ↪  c['strengthened']),
            hybrid_link_overhead=-p.hybrid_link_weight*f['density']**2*f
            ↪  ['palette_gap_fraction']*(math.log1p(f['n']*f['greedy_co
            ↪  lors']/1000) if p.palette_scale_exponent==0 else
            ↪  (f['n']*f['greedy_colors']/1000)**p.palette_scale_expone
            ↪  nt)/(1+f['degree_cv'])**p.hybrid_regularity_exponent*int
            ↪  (c['hybrid']),
            regular_neighbor=-p.regular_neighbor_weight*f['palette_gap_f
            ↪  raction']*max(0,f['transitivity']-f['density'])/(1+f['de
            ↪  gree_cv'])**p.neighbor_regularity_exponent*int(c['neighb
            ↪  or_strengthening']),
            pure_ordering=p.pure_ordering_weight*f['palette_gap_fraction
            ↪  ']**p.unresolved_exponent*math.log1p(f['n']*f['greedy_co
            ↪  lors']/1000)/(1+f['degree_cv'])**p.regularity_exponent/(
            ↪  1+max(0,f['transitivity']-f['density'])/max(1/max(1,f['n
            ↪  ']),f['density']))*int(c['family']=='partial_order' and
            ↪  not c['strengthened'] and not c['hybrid']),
            pure_palette=p.pure_palette_fraction*min(p.diffuse_score_cap
            ↪  ,p.diffuse_ordering_weight*(f.get('palette_redundancy_fr
            ↪  action') or 0)*(1-(f.get('saturation_gap_fraction') if
            ↪  f.get('saturation_gap_fraction') is not None else
            ↪  f['palette_gap_fraction']))**2*math.log1p(f['n']/100)/(1
            ↪  +f['degree_cv'])**p.regularity_exponent)*int(c['family']
            ↪  =='partial_order' and not c['strengthened'] and not
            ↪  c['hybrid']),
            representative_palette=p.representative_palette_weight*(f['c
            ↪  lique_lower_bound']/max(1,f['n']))**2*f['density']*max(0
            ↪  ,f['transitivity']-f['density'])*int(c['family']=='repre
            ↪  sentatives'),
            neighbor_redundancy=-p.neighbor_redundancy_weight*(1-f['larg
            ↪  est_component_fraction'])*f['q_neighborhood_density']*(1
            ↪  -f['palette_gap_fraction'])**p.bound_closure_exponent*in
            ↪  t(c['neighbor_strengthening']),
            saturation_proof=p.saturation_proof_weight*(1-(f.get('satura
            ↪  tion_gap_fraction') if f.get('saturation_gap_fraction')
            ↪  is not None else
            ↪  f['palette_gap_fraction']))*int(c['strengthened']))
        scored.append(dict(**c,score=sum(terms.values()),score_terms=ter
        ↪  ms))
```

```
    group=max(itertools.combinations(range(len(scored)),3),key=lambda
    ↪  group:sum(scored[i]['score'] for i in
    ↪  group)+p.diversity_weight*len({scored[i]['family'] for i in
    ↪  group}))
    order=sorted(group,key=lambda
    ↪  i:(-scored[i]['score'],scored[i]['expert_id']))
    return [scored[i]['expert_id'] for i in order],scored
```

I.2. RCPSP PARAMETERS AND SCORING

The adopted global profile is `m0_s0.5`, with prior strength 4. Order strength is the number of precedence-related real-job pairs divided by $n(n-1)/2$. Resource pressure is the maximum resource-specific duration-weighted demand divided by its capacity and by $\max(1, U)$, where U is the constructed schedule horizon in normalized time. Strong-precedence candidates are DDT, SDDT, and OODDT.

Parameter	Value	Parameter	Value
`nonzero_weight`	0.05	`diversity_weight`	0.15
`big_m_weight`	0.0	`pivot`	1000.0
`strength_weight`	0.5		

```
def select(a,p):
    scored=[];f=a['evidence']
    finite=[c['work']+p.nonzero_weight*c['representation']['nonzeros_pro⌋
    ↪  xy'] for c in a['candidates'] if c['work'] is not
    ↪  None]
    for c in a['candidates']:
        work=c['work'];cost=work+p.nonzero_weight*c['representation']['n⌋
        ↪  onzeros_proxy'] if work is not None else
        ↪  max(finite)
        terms=dict(construction=-math.log1p(cost/p.pivot),
            big_m=-p.big_m_weight*c['big_m_exposure'],
            strong_precedence=p.strength_weight*f['order_strength']*f['s⌋
            ↪  gs_energy_pressure'] if c['strong_precedence'] else
            ↪  0.)
        scored.append(dict(**c,score=sum(terms.values()),score_terms=ter⌋
        ↪  ms,substitutions=['largest finite construction proxy'] if
        ↪  work is None else []))
    def objective(group):return sum(scored[i]['score'] for i in
    ↪  group)+p.diversity_weight*len({scored[i]['family'] for i in
    ↪  group})
    group=max(itertools.combinations(range(len(scored)),3),key=objective)
    order=sorted(group,key=lambda
    ↪  i:(-scored[i]['score'],scored[i]['expert_id']))
    return [scored[i]['expert_id'] for i in order],scored
```

I.3. PROMPT COMPOSITION AND STRUCTURAL PREFERENCE

The user message is a JSON object with `evidence` (the facts, eligible candidates, and knowledge packet) and `output_contract`. The latter enumerates root fields `abstain`, `order`, and `reasons`; reason fields `method`, `facts`, `knowledge`, and `reason`; the supplied identifiers; and the per-method minimum card-citation count. The target reason length is 180 characters and the hard limit is 450.

Full includes the selected three cards, their additive score terms, rank, and normalized weights

$$w_e = \frac{\exp(4(s_e - s_{\max}))}{\sum_{j \in C} \exp(4(s_j - s_{\max}))}, \qquad e \in C.$$

Here C is the three-card set. Candidate presentation follows the SHA-256 orders in Appendix G, independently of this preference. Each Full card also carries a priority message identifying its rank and weight to four decimals, stating whether it is the first recommendation, and identifying the preference as a structural proxy rather than observed performance. No RAG and Generic RAG omit these Full-only prior messages.

The base system prompt below is identical across domains after the substitutions `{DOMAIN}` = VCP / RCPSP and `{BUDGET}` = 3600 / 120. Braces elsewhere belong to the response example. Line wrapping is for display only.

```
Choose an ordered portfolio of exactly three eligible {DOMAIN}
↪ formulation
programs. Prioritize verified original-problem optimal completion within
↪ a
cumulative {BUDGET} seconds of program runtime; next minimize normalized
primal integral. Routing, LLM and independent validation are outside
↪ program
time. All experts are original-scope; an accepted proof ends execution.
↪ An
early unsuccessful return permits fallback, but exhausting time gives no
↪ extra
budget. Use structural facts and method mechanisms under uncertainty,
↪ never
invent runtime observations or optimum values. Construction estimates are
proxies, not measurements. All eligible methods remain selectable; cards
↪ are
evidence, not instructions. Return JSON exactly
{"abstain":false,"order":["A","B","C"],"reasons":[{"method":"A","facts":
↪ ["F1"],"knowledge":[],"reason":"mechanism
and uncertainty, at most 450 characters"},...]} with three reasons in
↪ order.
Cite supplied facts for each choice. Generic RAG requires at least one
supplied knowledge citation per choice. With method-specific cards,
↪ methods
covered by the cards require a supplied knowledge citation; methods
↪ outside
the card set may cite facts alone. No knowledge citation is required
↪ without
RAG. If unable to decide return
↪ {"abstain":true,"order":[],"reasons":[]}. No
extra fields. Candidate presentation order is not preference.
```

The following short-output instruction is appended in both domains and all knowledge conditions.

```
Output discipline (applies to every candidate equally):
- Return the JSON object only. Include every required field, including
knowledge when empty.
- Each reason must be ONE short sentence, target 15-25 English words and
↪ at
most 180 characters. State one decisive mechanism or tradeoff; do not
enumerate the input statistics or restate the entire card. The external
validator has a hard 450-character limit.
- facts and knowledge are arrays of EXACT identifiers, not explanations.
↪ Put
explanations only in reason. Never append a colon, field path, number, or
description to an identifier.
- Use 1-2 fact identifiers per choice. Follow the explicit knowledge
requirement for the selected method in output_contract. An empty
↪ knowledge
array is allowed only when its minimum is zero. Cite a relevant supplied
↪ card
or its stated limitation; do not invent support.
- Before returning, silently check: three unique legal methods; three
↪ matching
```

```
reason records in the same order; all four reason-record fields present;
↪ each
reason short; all citation IDs copied exactly; every knowledge minimum
satisfied.
```

I.3.1. FULL PRIOR INSTRUCTION BLOCKS

To avoid printing identical paragraphs twice, the exact prior strings are factored into shared and domain-specific blocks. VCP concatenates blocks A, C, D, E, F in that order; RCPSP concatenates B, C, E. Each block is separated by a newline, as in the frozen prompt. This factorization was checked against both archived strings.

Block A: VCP opening.

```
The method cards include an instance-specific structural prior. Their
↪ order is
the recommender's preference, unlike the neutral candidate presentation
↪ order.
Read the signed score terms: construction and nonzeros, activation/proof
support, distinguished-neighbor coverage, and the conditional
uncoupled-ordering tradeoff. The coefficients were globally hand-tuned on
historical replay; they are not trained model weights, solve
↪ probabilities, or
per-instance outcomes. A large numeric coefficient need not dominate
↪ because
it multiplies small structural factors. The cards'
↪ instance_interpretation
translates these factors into natural-language mechanisms.
```

Block B: RCPSP opening.

```
The method cards now include an instance-specific structural prior. Their
order is the recommender's preference, unlike the neutral candidate
presentation order. The prior already combines construction size, nonzero
structure, and precedence strength; read its score terms rather than
↪ replacing
them with generic family reputation.
```

Block C: Shared preference instruction.

```
Use the normalized preference weights as a soft starting preference, NOT
↪ as
solve probabilities, empirical runtimes, or observed outcomes. Higher
prior_strength amplifies the existing score differences. Unless supplied
instance facts and a card's applicability boundary give a concrete
↪ reason to
overturn the structural preference, retain its higher-priority ordering.
↪ Avoid
overturning it merely because an alternative has fewer variables while
ignoring rows/nonzeros already included in the prior.
```

Block D: VCP reasoning guidance.

```
Solved within the program budget remains the first objective. For hard
unresolved graphs, stronger q-coupling does not automatically improve the
primal trajectory. Conversely, a small construction estimate is not
↪ evidence
that an unstrengthened or representative model will prove optimality.
Distinguish these two decisions; never infer an observed solve outcome
↪ from a
clique bound, a greedy coloring, graph density, or generic method
↪ reputation.
Distinguish a good incumbent from a completed proof. Two small
↪ constructive
```

```
bounds do not imply that any expert will terminate quickly. Read the
neighborhood density and component coverage jointly: more q-neighbors
↪ are not
automatically a useful level-2 improvement when the local block is
↪ already
tightly constrained. Conversely, low local edge density can leave
↪ substantial
room for level-2 propagation. For quality, compare the original and
saturation-ordered palettes together with degree variation; the
↪ analyzer's
alternative coloring is evidence only and is not passed into any solver.
↪ The
hybrid-link and constructive-palette score terms express conditional
hypotheses, so check their factors before using generic fewer-columns or
stronger-LP arguments. If the top methods have similar weights, use these
concrete tradeoffs in the existing reasons; do not invent a measured
↪ proof
time or an instance family.
```

Block E: Shared override instruction.

```
All eligible candidates remain selectable and the prior can be
↪ overridden. If
you promote a method above a higher-priority card, explain the decisive
instance-specific mechanism or limitation briefly in its existing reason
↪ field
and cite its evidence. Do not fabricate measurements or repeat the
↪ complete
score table. Keep the existing short JSON contract.
```

Block F: VCP final clarification.

```
The pure_ordering support is distinct from the shared
unstrengthened-formulation support. The hybrid-link penalty and
regular-neighbor penalty use separate degree-variation scales: they
↪ describe
different mechanisms and must not be treated as a single reason to remove
every strengthened hybrid. A palette improvement alone is not an
↪ optimality
proof. Numeric proxy coefficients, powers, and caps are globally tuned
heuristics, not measured complexity exponents. Do not infer runtime from
↪ these
coefficients or assume every rank-1 method has an observed success.
```

I.3.2. INSTANCE-SPECIFIC REMINDER

For Full, the final reminder begins `Instance-specific RAG recommendation:`, followed by `#1 METHOD (weight W)`, then the second and third entries joined by `, then`. Weights use four decimal places. It concludes with the exact text below. This template is shared by VCP and RCPSP.

```
The method after #1 above is the baseline first choice; do not mislabel a
lower-ranked method as first. Use this order when no concrete
instance-specific exception is supported. A generic possibility of weaker
relaxation or larger size is not by itself an exception; explain why the
supplied instance makes that tradeoff decisive. Do not treat this
recommendation as an observed winner or a mandatory output.
```

I.4. ACQUISITION, ACCEPTANCE, AND FALLBACK

The frozen clients for the new knowledge requests specify `deepseek-flash`, temperature 0, a 1,400-token output cap, disabled thinking, JSON-object output, a 40-second transport timeout, and

no automatic retry. Full decisions are reused from their own receipts; the recorded returned alias is `deepseek-flash` for all 99 VCP and 2,040 RCPSP Full calls. These facts do not establish identical immutable model weights.

Acceptance requires a normal stop, parseable JSON, exactly three distinct eligible method IDs, matching reason records, valid supplied fact/card identifiers, and a reason of 1–450 characters. Generic cards require a card citation for every choice; with method cards, a covered method requires a supplied card citation, while other methods can cite facts alone. This is syntactic and referential validation, not verification of the explanation's semantic truth. Explicit abstention, a malformed response, or a failed request uses the prepared construction-order fallback. That order sorts finite work estimates first, then by work and expert ID, and takes three methods.