MOLGOT: ORGANIZING LLM MOLECULAR REASONING WITH STRUCTURE-GUIDED GRAPHS OF THOUGHT

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly used to understand, modify, and generate molecules through natural language instructions. Recent approaches expose atom identities and bond relations in explicit molecular representations, reducing the need to reconstruct connectivity from compact string syntax before performing chemical operations. However, making structure accessible does not determine how intermediate decisions should be coordinated: observations from connected regions may depend on one another, while separately proposed edits can interact when combined. Graph of Thoughts offers a framework for this coordination by representing intermediate results as nodes and their dependencies as edges, enabling findings from different reasoning paths to be combined and revised. Molecular structure can ground these operations by identifying which atoms and regions each state concerns and where their consequences may interact. Building on this connection, we introduce MolGoT, a training-free method that retains SMILES and organizes molecular reasoning through structure-guided graphs of thought. MolGoT constructs a shared structural context and maintains local evidence and executable molecular candidates, with task-specific evaluation, integration, and revision for molecular understanding, analysis, editing, and generation. Across eleven molecular tasks, MolGoT achieves the highest average accuracy among the four evaluated pipelines. These results suggest that structure-guided thought graphs provide a controllable framework for improving both the accuracy and token efficiency of molecular reasoning in LLMs.

1 INTRODUCTION

Large language models (LLMs) increasingly connect natural-language instructions with molecular understanding, analysis, editing, and generation. Chemical language models and text–molecule systems make it possible to express a desired chemical operation or molecular property in language and obtain a structural answer (Edwards et al., 2022; Zhao et al., 2025; Yu et al., 2024; Pei et al., 2024). Such an interface depends on the model relating the request to the correct molecular objects: a formula must account for the complete structure, and a local edit must preserve the parts that the instruction leaves unchanged. Molecular benchmarks show that these tasks remain challenging even for models with substantial chemical knowledge (Cai et al., 2026; Bartmann et al., 2026; Runcie et al., 2026).

The input representation determines how readily those molecular objects can be identified. SMILES provides a compact description, but recovering its molecular graph requires interpreting branches, ring closures, and bond notation (Weininger, 1988). Recent representations make aspects of this structure more accessible: fragment-based languages expose larger chemical units (Wu et al., 2024), while MoleCode expresses atoms and bonds as typed entities with persistent identifiers and explicit relations (Yan et al., 2026). These references let a model locate structural elements and specify edits directly, reducing the need to reconstruct connectivity from sequential syntax. Figure 1 illustrates how different interfaces expose the same underlying molecule.

Improved structural access also creates an opportunity to guide the reasoning that follows. The information used to identify an atom or a chemical region can help relate intermediate observations

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.

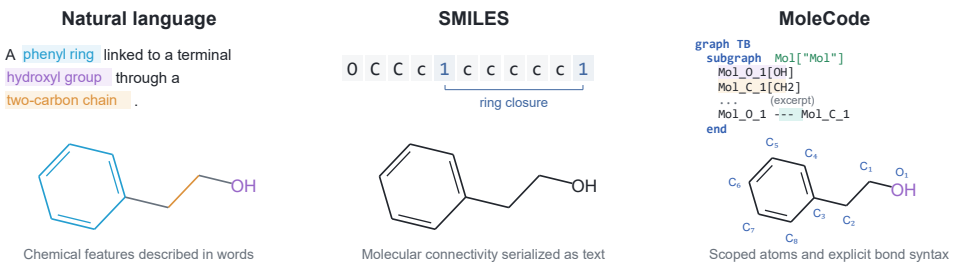

Figure 1: **Different interfaces to the same molecular structure.** A natural-language description, SMILES, and an abridged MoleCode representation describe 2-phenylethanol. The MoleCode excerpt exposes atom identities and bonds; omitted declarations are indicated by an ellipsis. Colored characters in the SMILES panel illustrate notation, not a model tokenizer. This is a representation example, not a recorded prediction.

and proposed changes to a common task. Local judgments may depend on evidence elsewhere in the molecule, while alternative edits need to be considered together before they are combined. For example, two proposals that each add one amide can jointly violate an instruction requiring exactly one addition. A solving procedure therefore needs to preserve what each intermediate result concerns and assess how its use affects the overall answer. This extends the role of molecular structure from making the input understandable to organizing the decisions made over it.

Graph of Thoughts (GoT) provides a general framework for organizing such intermediate results (Besta et al., 2024). It represents results as nodes and their dependencies as edges, allowing separate reasoning paths to contribute to a later state through aggregation and allowing feedback to support refinement. Applying this framework to molecules requires giving those operations a structural meaning. Integrating observations from several regions requires interpreting their evidence in the full molecule; composing edits requires executing the proposed changes and checking their joint consequences. Both operations draw on preceding states, but the information retained and the checks required differ.

Molecular structure supplies a shared reference for these states and operations. Persistent atom identities associate observations and edits with the same molecular objects across reasoning steps. Chemical regions locate the scope of each state, while adjacency and boundary bonds help identify where structural consequences may interact. These relationships provide a basis for connecting relevant evidence and checking whether proposed changes can be applied together. The resulting thought graph records dependencies between solving states, which may concern a local observation or a complete candidate molecule; its edges are not chemical bonds. Molecular structure thus informs the organization of reasoning without prescribing a single reasoning path for every task.

Building on this connection, we introduce MolGoT, a training-free method that organizes molecular reasoning through structure-guided graphs of thought. We retain SMILES as a compact, widely supported input format that allows existing molecular data to be used directly. MolGoT derives atom, bond, and region information from SMILES and uses this shared structural context to organize task-specific solving states. Understanding and analysis tasks organize evidence and cross-checks, while editing and generation tasks maintain candidate molecules with executable changes. The model proposes and integrates decisions; chemical tools execute edits and check validity and public constraints. Composition and revision are considered where the task and candidate checks permit them, and the selected answer can remain an individual candidate. The entire procedure operates without updating model weights.

We compare Molecode-CoT, Molecode-GoT, SMILES-CoT, and MolGoT across eleven molecular tasks. Molecode-CoT applies chain-of-thought reasoning to MoleCode inputs. Molecode-GoT first converts the original SMILES using MoleCode's conversion tools, then organizes reasoning as a graph that retains intermediate molecular states and edit operations, with changes performed in MoleCode. SMILES-CoT applies chain-of-thought reasoning to the original SMILES. Averaged across tasks and evaluated model configurations, MolGoT achieves the highest accuracy among these four pipelines. Budget controls further examine whether the observed advantages persist when

alternative reasoning procedures receive comparable or larger token budgets. Ablations and recorded trajectories complement these comparisons by relating final answer quality to the structural mechanisms and intermediate decisions involved in obtaining it.

Compared to previous work, MolGoT has three potential advantages:

1. **Cost efficiency:** compact SMILES inputs reduce the representational overhead associated with explicit graph descriptions. Coupling this economical input format with structure-guided graph reasoning enables MolGoT to achieve high accuracy within a modest token budget.
2. **Coherent reasoning:** linking intermediate states to atoms and chemical regions allows MolGoT to relate local evidence to whole-molecule requirements. Findings from different reasoning paths can then be integrated and revised with reference to the same molecular structure.
3. **Process interpretability:** by preserving the molecular states, operations, and dependencies behind an answer, MolGoT makes the solving process available for inspection. Successful and failed trajectories can then be compared to identify where candidate generation, evaluation, or selection leads to different outcomes.

2 Related work

Graph of Thoughts. Besta et al. (2024) organize LLM-generated thoughts as a graph whose edges express dependencies between intermediate states. Aggregation allows information from separate branches to contribute to a later thought, while refinement uses feedback to improve an existing state. These operations offer a natural basis for molecular problems in which several local judgments or candidate edits must inform one answer. Applying them to molecules requires defining what a state contains and what it means to combine states: individually useful edits may conflict or jointly violate the requested constraint. MolGoT grounds this graph in a shared molecular context and gives its transitions task-specific meaning. Evidence states support integration and cross-checking; molecular candidates carry executable edits that are checked before composition or revision. The resulting contribution is a molecular execution interface for these operations: persistent atom references, replayable edit programs, compatibility checks, and acceptance against public task constraints. Task-specific evidence routes extend the same state-dependency view to symbolic answers.

MoleCode. Yan et al. (2026) make molecular topology directly accessible through typed atoms, persistent identifiers, and explicit bonds in a Subgraph–Node–Edge grammar. This interface supports localized graph edits and iterative agentic workflows, allowing LLMs to inspect and modify molecular objects. Such structural access provides a foundation for reasoning over molecules. Our focus is how intermediate observations and alternative edits are organized once that structure is available: which states should be retained, which findings can be combined, and when revision is warranted. MolGoT addresses these decisions through task-adapted thought graphs and candidate checks. The two approaches are complementary: our SMILES-based pipeline constructs structured context that includes MoleCode-style subgraph text. We also evaluate a separate MoleCode-native GoT implementation with its own state and operation semantics; these comparisons concern complete pipelines rather than an isolated change of input representation.

3 Structure-guided molecular reasoning

Given a task instruction q, public constraints c, and an optional starting SMILES s_0, MolGoT returns a symbolic answer or a molecular SMILES. It separates the molecular graph $M = (A, B)$, whose edges are chemical bonds, from the thought graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, whose edges record dependencies between solving states. Figure 2 illustrates an executable route with a recorded trajectory; Figure 3 details the shared context, state types, and division of work between the model and tools. Model weights remain fixed throughout inference.

3.1 Building a shared structural context

RDKit parses a supplied SMILES and assigns persistent atom IDs (Landrum et al., 2024). The structural view identifies ring systems, functional-group matches, and residual connected compo-

nents, in that order. Previously assigned atoms retain their region; overlapping matches can remain annotations. Small regions are merged into strongly connected neighbors until at most three remain, preserving atom coverage. A coverage failure triggers a single-region view. Prompts expose the complete molecule, local atom identities, neighboring regions, boundary atoms, and cross-region bonds.

Generation first requests six seeds, repairs invalid responses where possible, removes invalid or duplicate structures, and ranks the valid pool by public constraints. If none is valid, a rule-based constructor supplies a starting structure, such as a carbon chain or a functional-group example. The selected seed becomes the root; seed calls count toward inference cost. These fallback structures need not satisfy every requested constraint.

3.2 Thought states and executable molecular changes

Thought nodes store solving states, not one fixed node per atom or region. Local evidence includes observations, provisional answers, atom identities, and neighboring dependencies. Counting states hold model-produced estimates or checks. Molecular candidate states contain a molecule and its executable edit program. Integration and revision states identify their predecessors and resulting answer or molecule. Edges therefore express evidence dependencies and alternative structural changes.

An executable candidate is $x = (M_x, P_x, F_x, R_x, c_x, p_x, v_x)$: molecule, cumulative program, touched-root-atom footprint, target regions, confidence, progress, and online score. The model proposes JSON atom, bond, or fragment operations; the executor applies them to persistent atom IDs and records the resulting structure and checks. New IDs are namespaced by patch so independent edits can be replayed together. A whole-SMILES fallback can be selected but cannot be expanded or composed as a program. Appendix C provides the schema and a recorded edit.

3.3 Adapting the graph to the task

Analysis. Carbon counting integrates local counts, checks them against an independent SMILES estimate, and adjudicates disagreements. Ring counting combines sequential and direct topology calculations with a SMILES ring-closure branch. Functional-group counting selects local evidence for a global readout. The model infers the counts; prompts omit precomputed global totals and formulae (Appendix B.1).

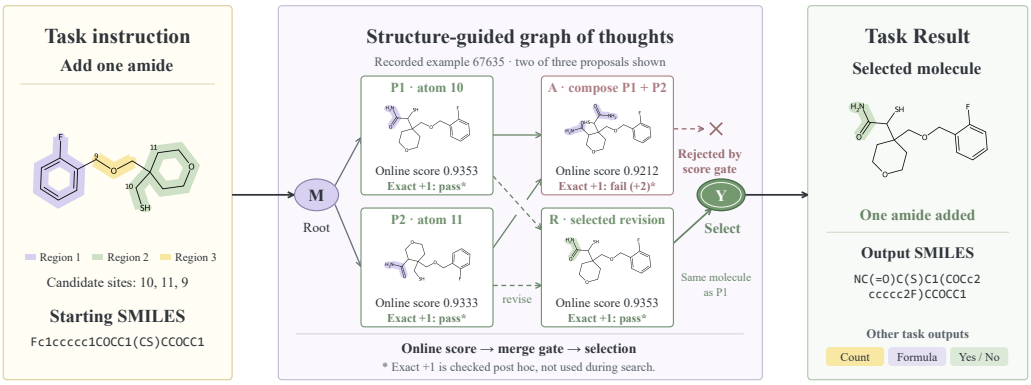

Figure 2: **MolGoT organizes molecular candidates and their dependencies.** In recorded GPT example 67635, proposals P1 and P2 attach an amide at different atoms. Their composition is executable but scores below the best parent and is rejected. Revision R reproduces P1 and is selected. Two of three proposals are shown. The displayed exact $+1$ checks were applied after inference; they were not feedback supplied to the search. Other routes produce counts, formulae, or equivalence judgments instead of edited molecules.

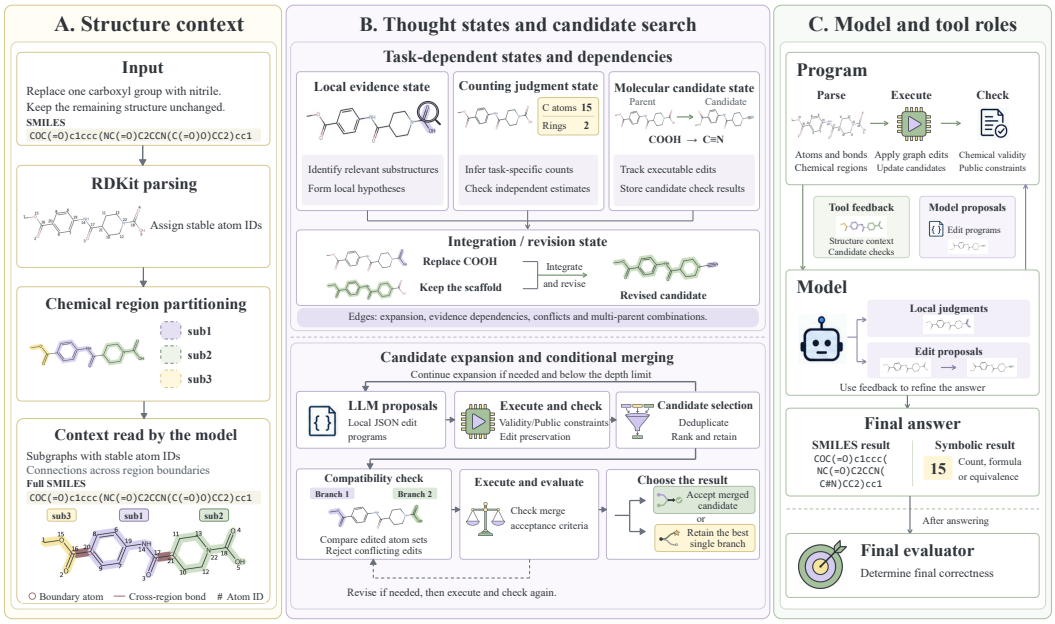

Figure 3: **Structural context, task-dependent states, and model–tool roles.** The program builds context and checks candidates; the model proposes and integrates decisions. Final evaluation follows selection. Molecules are mechanism illustrations, not recorded trajectories. Counting labels depict possible answers, not tool-computed counts supplied to the model.

Understanding. Formula prediction integrates subgraph evidence with the complete molecule. Equivalence uses selected local evidence and the full comparison task for its final judgment.

Editing and generation. Both search executable candidates against public constraints, with model-based evaluation for open-ended text goals. Editing starts from the supplied molecule; generation starts from a checked seed. At each of at most two levels, up to three patches per parent are executed, checked, deduplicated, and narrowed to two retained states. Repair, reserved alternatives, and early stopping depend on validity and progress (Appendix B.1).

3.4 Checking, composing, and selecting states

An editing candidate receives an online composite that balances validity, the visible task predicate, structural preservation, and locality:

$$v(x) = 0.35V(x) + 0.35C_{\text{vis}}(x) + 0.15S(M_x, M_0) + 0.15L(x). \tag{1}$$

Here S is fingerprint Tanimoto similarity to the root. With public constraints, $p = C_{\text{vis}}$ is computed from their checks, overriding predicted progress. For executable patches, L is the fraction of touched root atoms within the target regions and their boundaries. Candidates are ordered lexicographically by $(V, p, v, c, -|P|)$, prioritizing validity and visible progress before the composite. Constrained generation uses $0.5V + 0.5C_{\text{vis}}$; open-ended generation uses validity and, where applicable, model-assessed semantic progress. Locality normalization and fallback behavior are specified in Appendix B.

These online checks guide search and are separate from final task correctness. They use the instruction and public constraints, not hidden labels or reference molecules. For editing, the online predicate rewards any strict increase or decrease in the requested group count, whereas final success requires exactly one and a conservative edit witness. Group-constrained generation likewise uses online lower bounds but exact final counts; the two checks also use different SMARTS dictionaries (Appendix B.2). These are properties of the frozen implementation, not hidden-reference requirements. Nonstructural online checks can establish that an answer has the expected form without establishing that its value is correct. The final evaluator is applied after answer selection and

Table 1: **Task categories and task success criteria.** The editing tasks retain the names used by the source records but operate on the requested chemical components.

Category	Task	Evaluation criterion
Understanding	Molecular formula	Exact element counts and molecular charge.
	Molecular equivalence	Correct structural-equivalence judgment.
Analysis	Carbon count	Exact number of carbon atoms.
	Ring count	Exact count under the frozen RDKit-based definition.
	Functional-group count	Exact count of the specified group.
Editing	Atom addition	Exact group increase and conservative addition witness.
	Atom deletion	Exact group decrease and conservative deletion witness.
	Atom substitution	Required group changes and conservative edit witness.
Generation	Atom-number constrained	All requested counts, no extra elements, one component.
	Functional-group guided	Exact requested group counts and one component.
	ChEBI text-based	Canonical isomeric SMILES identity with the reference.

does not feed its decision back into the recorded search. This separation also makes it possible to diagnose which apparently promising intermediate states fail the final endpoint.

Composition considers the top eight valid unique states and combines programs from at most two compatible parents by replaying them on the root. A true merge requires both parents to contribute and the result to differ from either parent. Compatibility alone does not justify selection. With best individual b and composition a, the gate requires

$$V(a) = 1, \qquad p(a) \geq p(b), \qquad v(a) \geq v(b). \tag{2}$$

Without explicit constraints, model-assessed semantic progress must improve strictly: $p(a) > p(b)$.

If no composition passes the gate, public constraints and two valid compatible parents can enable revision. The revised state faces the same gate; it may reproduce a parent. A true-merge tie-break follows validity, progress, and score and cannot rescue a worse candidate. Thus the graph records attempted as well as accepted reuse, and the final answer can remain a single-branch result (Appendix F.1).

4 Experimental Evaluation

4.1 Evaluation Protocol

We select eleven tasks from five benchmark sources: S2-TOMG-Bench (Li et al., 2026), ChEBI-20 (Edwards et al., 2022), ChemCoTBench and ChemCoTDataset (Li et al., 2025), and ChemIQ (Runcie et al., 2026). These tasks span molecular understanding, analysis, editing, and generation (Table 1). Formula, equivalence, and counting labels are derived offline with RDKit. We compare Molecode-CoT, Molecode-GoT, SMILES-CoT, and MolGoT (ours) across ten model configurations: five Qwen3 sizes, Llama-3.1-8B, and DeepSeek-R1-0528-Qwen3-8B (Yang et al., 2025; Grattafiori et al., 2024; DeepSeek-AI et al., 2025), together with three API models: **DeepSeek V4.1 Flash**, **Gemini 3 Flash**, and **GPT-5.6 Luna**. GPT-5.6 Luna uses reasoning effort `low`. We abbreviate these API models as DeepSeek, Gemini, and GPT; R1 denotes the open-weight DeepSeek-R1-Qwen3-8B model. Molecode-GoT uses its own native executor, so the main comparison concerns complete pipelines. Additional inference settings are documented in Appendix A.

Evaluation metrics. We report accuracy and mean logical tokens per answer. Accuracy measures final task correctness independently of online candidate scores; token use includes input, completion, and generation-seed calls. Token counts compare inference expenditure within a model, rather than monetary cost across providers. Validity, editing goal satisfaction, conditional structure preservation, counting error, and reference similarity are defined in Appendix A.4 and tabulated in Appendix D.

Table 2: **Main-task accuracy and inference cost.** Acc. is accuracy (%); kTok is mean logical tokens per completed answer, in thousands. Bold marks the highest accuracy within each model. GPT-5.6 Luna uses `low` reasoning effort.

Model	Molecode-CoT		Molecode-GoT		SMILES-CoT		MolGoT (ours)	
	Acc.	kTok	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	11.82	1.40	**14.18**	10.08	12.55	0.42	10.55	5.73
Qwen3-1.7B	10.91	1.82	**18.36**	11.18	15.73	0.46	14.09	5.50
Qwen3-4B	26.64	1.66	27.18	9.51	19.55	0.46	**34.73**	6.29
Qwen3-8B	27.73	1.61	27.73	10.33	20.73	0.43	**42.45**	5.89
Qwen3-14B	25.64	1.61	28.91	10.29	27.27	0.45	**41.09**	6.20
Llama-3.1-8B	15.36	1.46	18.09	10.34	16.91	0.41	**23.82**	6.30
R1-Qwen3-8B	24.64	6.39	**32.00**	26.35	15.27	5.04	27.64	28.92
DeepSeek V4.1 Flash	54.18	1.22	53.00	12.62	53.09	0.38	**55.73**	5.50
Gemini 3 Flash	77.09	3.90	**78.36**	35.13	62.64	0.49	74.18	8.56
GPT-5.6 Luna	70.09	1.49	74.55	17.83	74.45	0.75	**80.64**	6.48

4.2 Molecular task performance

MolGoT (ours) improves accuracy over SMILES-CoT in eight configurations and ranks first in six (Table 2). Qwen3-8B improves from 20.73% to 42.45%, GPT from 74.45% to 80.64%, and DeepSeek from 53.09% to 55.73%. Gemini improves from 62.64% to 74.18%, but favors Molecode-GoT at 78.36%. The two smallest Qwen configurations and R1 also favor Molecode-GoT. Thus, organizing reasoning around molecular structure improves several complete pipelines, but the preferred organization depends on the underlying model.

4.3 Ablation studies

4.3.1 Token-matched comparison

Table 2 shows substantial differences in token use: GoT requires more tokens than CoT, while Molecode expands the molecular input into explicit graph text. We therefore test whether allocating comparable or larger budgets to CoT can recover MolGoT's gains. We sample independent answers at $0.5\times$, $1\times$, and $2\times$ the corresponding graph budget and select by public candidate checks, with majority voting as a sensitivity analysis (Wang et al., 2023; Brown et al., 2024), using both original and context-enhanced SMILES inputs and a Molecode control. Figure 4 reports attained tokens, including overshoot from completing whole answers.

Increasing the sampling budget does not close the gap for Qwen3-8B: MolGoT achieves 42.36% accuracy with 5.98k tokens, whereas the two SMILES controls reach 22.18% and 25.64% at the $2\times$ target, using 12.20k and 13.58k tokens. GPT also retains higher accuracy than these controls (78.91% versus 76.50% and 76.32%). DeepSeek API and R1 instead favor sampling. This may reflect model-specific compatibility with the controller, consistent with their sensitivity to task guidance and output limits examined in Appendices E.3 and E.4.

MolGoT also improves the accuracy–cost tradeoff relative to Molecode-GoT. In Figure 4, it uses 42–63% fewer tokens for Qwen3-8B, DeepSeek API, and GPT, with higher accuracy for Qwen3-8B and GPT and comparable accuracy for DeepSeek (55.09% versus 55.45%). These results support a benefit from reasoning organization beyond budget alone and show that MolGoT can achieve comparable or higher accuracy than Molecode-GoT with substantially fewer tokens.

4.3.2 Task-adapted graph execution

The budget results motivate a second question: does MolGoT remain effective beyond specialized counting and under alternative execution choices? Figure 5a places MolGoT and its simplified variants against the same SMILES-CoT reference. The full controller has higher accuracy in eight of nine configurations. For Qwen3-8B, it reaches 42.36%, compared with 36.00% under uniform rout-

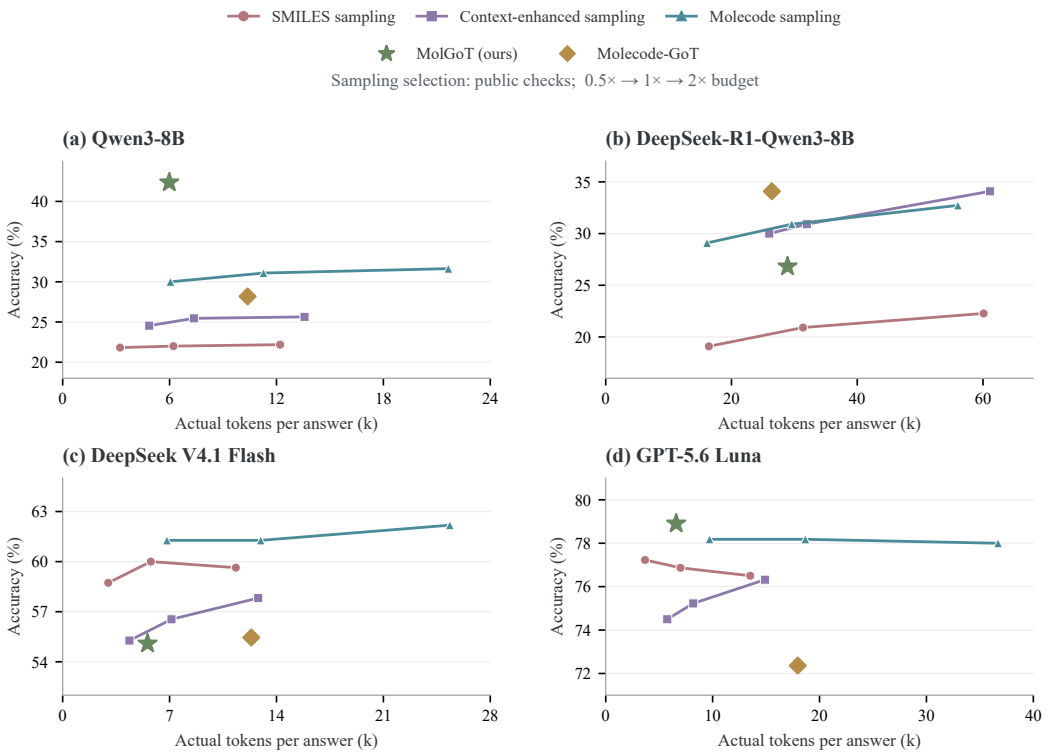

Figure 4: **Accuracy versus inference expenditure.** Each sampling curve's three points correspond to $0.5\times$, $1\times$, and $2\times$ its reference budget, from left to right. Stars mark MolGoT, the budget reference for both SMILES sampling controls; diamonds mark Molecode-GoT, the reference for Molecode sampling. The horizontal axis reports attained tokens. GPT-5.6 Luna uses `low` reasoning effort. Points summarize each condition's recorded answers; numerical values are in Appendices E.1 and E.2.

ing and 30.55% without specialized counting, while CoT reaches 21.64%. The simplified controllers retain an advantage here, and task-adapted execution strengthens it.

The effect extends beyond the dedicated counting routes. When carbon and ring counting are excluded from evaluation, MolGoT retains higher accuracy than CoT in six configurations (Figure 5c). The differences are 11.56 points for Qwen3-8B, 8.22 for Llama, and 5.33 for GPT. These results show that the complete workflow also supports the remaining molecular tasks, although the effect remains model-dependent.

Composition and repair are conditional operations within this workflow. The no-merge control excludes composed candidates from final selection while retaining search, evaluation, and repair; it does not remove graph reasoning. Its small or mixed changes therefore measure an optional operation's incremental value within an active controller. The repair intervention likewise does not reduce the system to CoT, and neither operation is uniformly beneficial (Figure 5a). The broader workflow also tolerates alternative partitions: both neutral-label chemical and random partitions exceed CoT in seven configurations (Figure 5b). Its observed usefulness is consequently not tied to one partitioning choice or to always selecting a merged answer.

Execution must also leave enough room for individual model responses. Increasing the requested node output caps improves R1 MolGoT by 13.64 points (Figure 5d), whereas the DeepSeek and GPT intervals include zero. Output allowance thus matters for R1, but increasing it does not benefit every model. The cap intervention remains subject to context limits; it is distinct from the total-budget sampling comparison in Section 4.3.1.

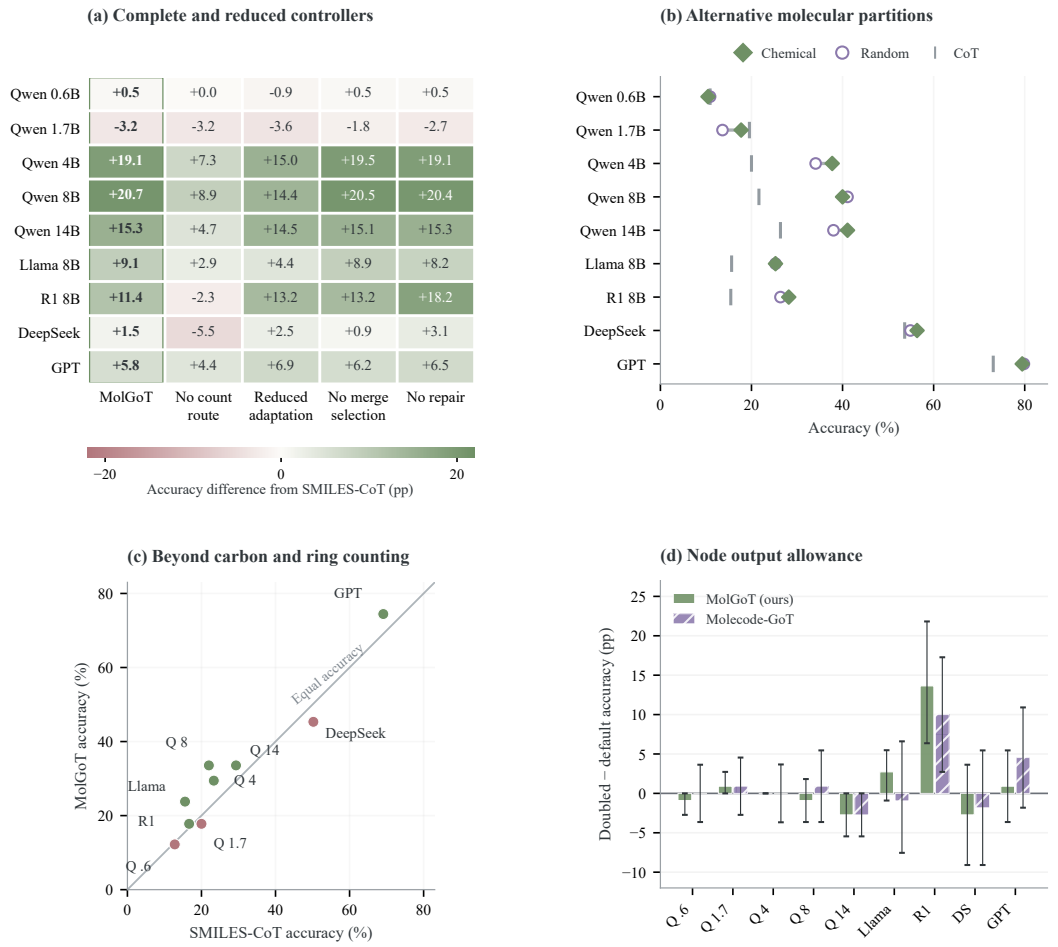

Figure 5: **Complete-workflow performance and execution choices.** (a) Accuracy differences from single-answer SMILES-CoT for MolGoT and four simplified controllers. (b) Chemical and random partitions with matched region sizes and neutral labels; gray ticks mark CoT. (c) Accuracy after excluding carbon and ring counting from evaluation; points above the diagonal favor MolGoT. All nine control configurations are retained. Panels (a)–(c) use the single-answer CoT reference, not matched token budgets. (d) Accuracy changes under doubled requested node output caps; error bars show exploratory paired 95% intervals. Exact values are in Table 7 and Appendix E.4.

5 Conclusion and Future Work

MolGoT brings molecular structure into the organization of LLM reasoning. It retains SMILES as the molecular interface and constructs a shared structural context in which local evidence, executable candidates, and their dependencies can be examined and revised without training the model. Task-adapted routes connect these intermediate states to a global answer, while conditional composition allows the controller to reuse compatible changes without requiring every result to be a merge. Evaluation across eleven molecular tasks shows that this organization can improve accuracy and the accuracy–cost tradeoff, with benefits that depend on the model and task. The budget controls further show that additional sampling alone does not consistently recover these gains. Future work will develop adaptive policies for choosing where to branch, cross-check, and revise, and strengthen the alignment between candidate checks and final chemical requirements. Extending the framework to richer structural evidence and property-based objectives, would test its utility beyond the present setting. More broadly, structure-guided thought graphs offer a concrete basis for building molecular assistants whose proposed changes and intermediate decisions can be inspected and tested.

AI USE STATEMENT

Generative AI assistance was used to draft and revise text, inspect supplied code and experimental records, organize literature, and develop figure layouts and editable presentation assets. Some schematic designs and decorative icons used image-generation assistance; statistical plots were computed from recorded data, and molecular depictions were rendered from SMILES with chemical drawing tools. No image-generation model was used to create statistical evidence. No new experimental model calls were made for this manuscript revision.

ETHICS STATEMENT

The study evaluates computational molecular reasoning on public source data. Success under the specified structural criteria does not establish biological activity, synthesizability, safety, or clinical utility. Generated molecules are computational outputs and have not been experimentally validated in this work.

REPRODUCIBILITY STATEMENT

The companion offline package contains the original endpoint scorer, source/condition manifests, deterministic selection code, portable per-example analyses, and two saved-response controller replays. It reproduces saved-outcome statistics and the two recorded editing executions without model access; the evaluator additionally requires RDKit and NetworkX. The supplement specifies endpoints, task routes, budget accounting, prompt excerpts, JSON operations, and recorded success/failure cases. Source hashes distinguish the frozen experimental implementation from the toolkit used for revision-time checks. No public hosting location is claimed.

REFERENCES

Christoph Bartmann, Johannes Schimunek, Mykyta Ielanskyi, Philipp Seidl, Günter Klambauer, and Sohvi Luukkonen. MolecularIQ: Characterizing Chemical Reasoning Capabilities Through Symbolic Verification on Molecular Graphs. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/5cf2e7b47773bcf98c9ad09cb524cf64-Paper-Conference.pdf.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024. doi: 10.1609/aaai.v38i16.29720. URL https://ojs.aaai.org/index.php/AAAI/article/view/29720.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6(5):525–535, 2024. doi: 10.1038/s42256-024-00832-8.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling. *arXiv preprint arXiv:2407.21787*, 2024. URL https://arxiv.org/abs/2407.21787.

Feiyang Cai, Jiahui Bai, Tao Tang, Guijuan He, Joshua Luo, Tianyu Zhu, Srikanth Pilla, Gang Li, Ling Liu, and Feng Luo. MolLangBench: A Comprehensive Benchmark for Language-Prompted Molecular Structure Recognition, Editing, and Generation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8668e1360acd941ed3bedd341428c7f9-Paper-Conference.pdf.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. Translation between Molecules and Natural Language. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 375–413, 2022. doi: 10.18653/v1/2022.emnlp-main.26. URL https://aclanthology.org/2022.emnlp-main.26/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Hongyu Guo, Hao Li, He Cao, Gongbo Zhang, and Li Yuan. From Answers to States: Verifiable Process-Level Evaluation of Chemical Reasoning in Large Language Models. *arXiv preprint arXiv:2606.03660*, 2026. URL https://arxiv.org/abs/2606.03660.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. doi: 10.1145/3600006.3613165. URL https://dl.acm.org/doi/10.1145/3600006.3613165.

Greg Landrum, Paolo Tosco, Brian Kelley, et al. rdkit/rdkit: 2023_09_6 (Q3 2023) Release. Zenodo, March 2024. URL https://zenodo.org/records/10793672.

Hao Li, He Cao, Bin Feng, Yanjun Shao, Xiangru Tang, Zhiyuan Yan, Yonghong Tian, Li Yuan, and Yu Li. Beyond Chemical QA: Evaluating LLM's Chemical Reasoning with Modular Chemical Operations. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5122. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/e0ed6d6c2ec6df05f929b8a67b78513a-Abstract-Datasets_and_Benchmarks_Track.html.

Jiatong Li, Junxian Li, Weida Wang, Yunqing Liu, Changmeng Zheng, Yatao Bian, Dongzhan Zhou, Xiao-yong Wei, and Qing Li. Speak-to-Structure: Evaluating LLMs in Open-domain Natural Language-Driven Molecule Generation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 9314–9325, 2026. doi: 10.1145/3770855.3817473. URL https://dl.acm.org/doi/10.1145/3770855.3817473.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2019. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1025. URL https://aclanthology.org/2025.emnlp-main.1025/.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Qizhi Pei, Lijun Wu, Kaiyuan Gao, Xiaozhuan Liang, Yin Fang, Jinhua Zhu, Shufang Xie, Tao Qin, and Rui Yan. BioT5+: Towards Generalized Biological Understanding with IUPAC Integration and Multi-task Tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1216–1240. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.71. URL https://aclanthology.org/2024.findings-acl.71/.

David Rogers and Mathew Hahn. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t. URL https://pubmed.ncbi.nlm.nih.gov/20426451/.

Nicholas T. Runcie, Charlotte M. Deane, and Fergus Imrie. Assessing the Chemical Intelligence of Large Language Models. *Journal of Chemical Information and Modeling*, 66(1):216–227, 2026. doi: 10.1021/acs.jcim.5c02145. URL https://pubs.acs.org/doi/10.1021/acs.jcim.5c02145.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally Can be More Effective than Scaling Parameters for Reasoning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Eric Wang, Samuel Schmidgall, Paul F. Jaeger, Fan Zhang, Rory Pilgrim, Yossi Matias, Joelle Barral, David Fleet, and Shekoofeh Azizi. TxGemma: Efficient and Agentic LLMs for Therapeutics. *arXiv preprint arXiv:2504.06196*, 2025. URL https://arxiv.org/abs/2504.06196.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract.html.

David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005. URL https://pubs.acs.org/doi/10.1021/ci00057a005.

Thomas Wolf, Lysandre Debut, Victor Sanh, et al. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Juan-Ni Wu, Tong Wang, Yue Chen, Li-Juan Tang, Hai-Long Wu, and Ru-Qin Yu. t-SMILES: a fragment-based molecular representation framework for de novo ligand design. *Nature Communications*, 15(1), 2024. doi: 10.1038/s41467-024-49388-6. URL https://www.nature.com/articles/s41467-024-49388-6.

Zhiyuan Yan, Chen Liu, Boxuan Zhao, Kaiqing Lin, Jixiang Zhao, Yimi Wang, Liuzhenghao Lv, Hao Li, Shanzhuo Zhang, Li Yuan, and Fanyang Mo. MoleCode unlocks structural intelligence in large language models. *arXiv preprint arXiv:2605.16480*, 2026. URL https://arxiv.org/abs/2605.16480.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. doi: 10.52202/075280-0517. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations*, 2023b. URL https://arxiv.org/abs/2210.03629.

Botao Yu, Frazier N. Baker, Ziqi Chen, Xia Ning, and Huan Sun. LlaSMol: Advancing Large Language Models for Chemistry with a Large-Scale, Comprehensive, High-Quality Instruction Tuning Dataset. *arXiv preprint arXiv:2402.09391*, 2024. URL https://arxiv.org/abs/2402.09391.

Jinwei Zhang, Xucheng Liang, Yu Zhang, Ruijie Yu, Xiaokang Yang, Yaohui Jin, and Yanyan Xu. ChemReason-bench: Benchmarking large language models for procedural reasoning in experimental chemistry. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33211–33248, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1535. URL https://aclanthology.org/2026.acl-long.1535/.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The Lessons of Developing Process Reward Models in Mathematical Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.547. URL https://aclanthology.org/2025.findings-acl.547/.

Zihan Zhao, Da Ma, Lu Chen, Liangtai Sun, Zihao Li, Yi Xia, Bo Chen, Hongshen Xu, Zichen Zhu, Su Zhu, Shuai Fan, Guodong Shen, Kai Yu, and Xin Chen. Developing ChemDFM as a large language foundation model for chemistry. *Cell Reports Physical Science*, 6 (4):102523, 2025. doi: 10.1016/j.xcrp.2025.102523. URL https://www.cell.com/cell-reports-physical-science/fulltext/S2666-3864(25)00122-5.

A EXPERIMENTAL SETTINGS AND EVALUATION

A.1 DATA AND TASK CONSTRUCTION

Table 3 lists the five benchmark resources in the project source collection. Tasks are organized by the required operation rather than by their source name. Molecular formulae, count labels, and equivalence targets are constructed offline; these reference values are removed before inference. Editing instructions and generation constraints remain visible to all pipelines.

Table 3: **Resources in the project source collection.**

Resource	Reference and scope
S2-TOMG-Bench	Natural-language molecular editing and constrained generation (Li et al., 2026).
ChEBI-20	Paired molecular structures and textual descriptions (Edwards et al., 2022).
ChemCoTBench	Evaluation of reasoning with modular chemical operations (Li et al., 2025).
ChemCoTDataset	Chemical-operation reasoning data from the same release (Li et al., 2025).
ChemIQ	Chemical intelligence and molecular reasoning evaluation (Runcie et al., 2026).

Final correctness. The endpoints are those in Table 1. Formula evaluation compares parsed element counts and charge with the frozen label; equivalence compares a parsed Boolean with its frozen label. Counting requires exact agreement with the recorded reference values. Editing requires the requested group-count change and a conservative graph-edit witness. A missing witness is a failure under this evaluator, not proof that no chemically acceptable interpretation exists. Atom-constrained generation requires exact counts, no additional elements, and one connected component. Functional-group generation requires exact requested counts and one component. ChEBI generation uses canonical isomeric identity with the reference, so an alternative molecule compatible with the description can still be marked incorrect. Online candidate scores are separate from all of these final endpoints.

A.2 DATA SELECTION AND CONDITION IDENTITY

The main set contains 1,100 examples, 100 per task, selected deterministically from the source records. Within each task, the builder sorts by example ID and samples without replacement using a task-specific seed derived from SHA-256. Replaying the archived selection reproduces the same records and order. The manifests preserve the selected records and selection rules.

A separate public-input audit covers 800 records and 900 molecular slots, including both molecules in equivalence questions. After removing atom-map labels, canonical isomeric SMILES give 857 unique structures; all 43 repetitions occur within individual equivalence questions. No structure is shared across different records or tasks, also when stereochemistry is ignored. Input-molecule dependency groups are therefore singletons. For Qwen3-8B's nine-task non-counting comparison, grouping those inputs and retaining generation records as individual units reproduces the original $+4.67$-point interval $[1.56, 7.78]$. The six tasks with molecular inputs alone give $+6.67$ points $[2.33, 11.00]$; the three generation tasks give $+0.67$ $[-3.33, 4.67]$. This revision-time audit uses RDKit 2026.03.6. It checks exact input identity; generation tasks have no supplied molecular input and remain record-level sampling units.

The control subsets are nested within this main set: 20 records per task for Qwen3-0.6B, 1.7B, 4B and R1, and 50 for Qwen3-8B, 14B, Llama and the two API control models. A deterministic per-task shuffle defines both subsets; the first ten records per task define output-cap probes. The four main-method controls import stored answers and usage. All 14,520 imported records were matched to the original parsed outputs, raw outputs and usage fields; they are not independent new repetitions. Additional guidance, sampling, component and repeated-run conditions are identified separately.

Archived upstream builders count carbon atoms by atomic number and use `GetRingInfo().NumRings()` for ring labels and `CalcMolFormula` for formula la-

bels. Equivalence positives reorder atoms into a different SMILES and verify canonical identity; negatives use different canonical molecules. The functional-group-count builder selects a nonzero, non-halo SMARTS group deterministically from the upstream record ID. Consequently that task evaluates positive group counts, not a balanced set of present and absent groups. Final scoring uses the frozen labels; the portable provenance package records the builder hashes and label fields rather than claiming that all labels were regenerated in this revision.

The condition manifest maps 40 main and 207 canonical control conditions to code, sample IDs, parameters and saved route markers. Thirty-eight older partial API conditions are preserved as history and excluded from these canonical totals. The scientific core is byte-identical across the ten complete code packages; the separate retry patch is a transport artifact, not a second scientific implementation. Carbon/ring and recovered-formula route markers agree between the main runs and their imported controls. The atom-batch formula ledger is not substituted into these default conditions. The no-count control changes carbon/ring routing; reduced adaptation changes general routing while leaving the specialized counting branch active. These are task-specific interventions, not identical deletions applied to every task.

The stored run identities and hashes establish consistency of the archived artifacts. API names are recorded request aliases: upstream weights and stable provider snapshots were not independently verified. The release manifest distinguishes code presence, recorded route execution, and unknown transport-patch attribution.

A.3 Models and inference settings

Open-weight configurations are Qwen3-0.6B, 1.7B, 4B, 8B, and 14B (Yang et al., 2025); Llama-3.1-8B-Instruct (Grattafiori et al., 2024); and DeepSeek-R1-0528-Qwen3-8B (DeepSeek-AI et al., 2025). Inference uses vLLM 0.8.5.post1 (Kwon et al., 2023), PyTorch 2.6.0+cu124 (Paszke et al., 2019), Transformers 4.57.6 (Wolf et al., 2020), and RDKit 2023.09.6 (Landrum et al., 2024). Four independent 80 GiB GPU replicas use bfloat16 and tensor parallelism one. No model weights are updated.

Table 4: **Inference configuration.** Temperatures shown are defaults; node-specific overrides remain active. API aliases identify the requested services in the recorded runs.

Configuration	Sampling	Reasoning and limits
Qwen3 / Llama	$T = 0.7, p = 0.95$	Context 16,384; Qwen uses the non-thinking template.
R1-Qwen3-8B	$T = 0.7, p = 0.95$	Native reasoning; context 32,768; additional 4,096 output tokens per node in each main arm.
DeepSeek V4.1 Flash	$T = 0.7$	Alias deepseek-flash; thinking disabled.
Gemini 3 Flash	$T = 0.2$	Alias gemini-3-flash; provider-default thinking.
GPT-5.6 Luna	$T = 0.2$	Alias gpt-5.6-luna; low reasoning effort; 4,096-token reasoning reservation.

The default decoding seed is 2027. API adapters do not send a remote seed or top-p; seed labels therefore do not imply deterministic remote decoding. GPT and Gemini use a proxy. Gemini usage includes provider-reported internal thoughts as well as visible completion tokens. Consequently, token counts quantify recorded inference expenditure within each model, not identical computation or monetary cost across providers. Resume runs reuse successful records rather than adding independent observations.

A.4 Metrics, denominators, and uncertainty

Accuracy is the proportion of final answers satisfying the task endpoint. Main accuracy uses the planned evaluation denominator; unrecorded answers contribute zero. Token means use completed outputs and include input, completion, and shared generation-seed calls. Paired comparisons use the intersection of completed example IDs. The main ledger contains 43,998 of 44,000 planned outputs; Gemini covers all four pipelines, while its additional control experiments were not run.

The diagnostic metrics separate different requirements:

- **Validity**: RDKit-valid outputs across editing and generation tasks.
- **Editing goal**: exact requested functional-group count changes. **Preservation** is conditional: the fraction of goal-satisfying outputs that also pass the edit witness. It is undefined if no output satisfies the goal.
- **Counting MAE**: absolute error averaged over parseable answers, computed separately for carbon, ring, and functional-group counts. The aggregate is the unweighted mean of these three MAEs; parseability is reported alongside it.
- **Reference similarity**: chiral radius-2, 2,048-bit Morgan fingerprint Tanimoto to the ChEBI reference (Rogers & Hahn, 2010). Invalid or missing structures contribute zero. Similarity does not replace exact reconstruction accuracy.
- **Generation diagnostics**: fraction of requested count predicates satisfied and fraction of single-component outputs. Invalid outputs contribute zero; satisfying some predicates is not task success.

Control intervals use 10,000 task-stratified paired bootstrap replicates and are exploratory, without multiplicity adjustment. Repeated-run comparisons average correctness within each example before resampling; they do not treat three answers to the same example as three independent molecules. The additional non-counting analysis uses bootstrap seed 20260925. These are task-stratified paired intervals for the evaluated examples.

Table 5: **Paired control coverage.** The final column is the common-ID denominator for shared-guidance graph–iteration. The middle column counts saved prediction records across the 23 canonical control conditions, not independent molecules or chemically valid answers.

Model	Saved / expected records	Paired IDs
Qwen3-0.6B	4400/4400	220
Qwen3-1.7B	4400/4400	220
Qwen3-4B	4399/4400	220
Qwen3-8B	10010/10010	550
Qwen3-14B	10010/10010	550
Llama-3.1-8B	10006/10010	550
R1-Qwen3-8B	4400/4400	220
DeepSeek V4.1 Flash	10010/10010	550
GPT-5.6 Luna	10006/10010	549

B IMPLEMENTATION DETAILS

B.1 STRUCTURAL CONTEXT AND CANDIDATE STATES

The context supplied to the model contains the full SMILES, persistent atom IDs, local regions, their adjacency, and boundary bonds. Region construction retains every atom and merges small regions until at most three remain; a coverage failure falls back to one complete region. Some prompts include MoleCode-style subgraph text. Thus SMILES is the external interface, while graph-structured text also appears inside the reasoning process.

An executable state stores its parent and root branch, current molecule, cumulative program, most recent patch, touched atoms, target regions, and candidate checks. New atoms use patch-specific identifiers. This representation permits exact replay on the common root and makes conflicting edits detectable. A terminal whole-SMILES fallback is not treated as a composable edit program. The model proposes and integrates states; chemical tools execute and check structures, a division also used in tool-augmented chemical systems (Bran et al., 2024; Wang et al., 2025).

Table 6: **Controller limits.** Task dispatch is specified in the pseudocode below.

Setting	Implementation
Search limits	Depth 2; beam width 2; up to 3 proposals per parent; up to 2 composition parents; top 8 valid unique states considered for composition.
Generation initialization	Request 6 seeds; check and deduplicate; select a valid seed. If all fail, use a public-rule fallback where supported.

Task routes and counting context. Carbon counting constructs local atom ledgers, integrates them into a global estimate, and compares that estimate with an independent reading of the original SMILES; disagreement invokes model adjudication. Ring counting combines topology-based sequential arithmetic, an independent direct topology calculation, and a SMILES ring-closure branch. The model synthesizes these judgments rather than receiving a precomputed ring answer. Functional-group counting generates local hypotheses, selects a state, and obtains a final judgment with access to the full molecule. Counting prompts omit global atom, carbon, bond, and ring totals, formulae, implicit-hydrogen totals, and explicit ring membership lists, while retaining structural information for reasoning.

Formula prediction integrates local subgraph evidence with the complete molecule. Main and default control runs use the recovered adaptive evidence route; the alternative atom-batch ledger remains in the code but is not the default evaluated route. Equivalence generates local judgments, selects one state, and reads the complete comparison task for the final answer. Neither route simply concatenates independent local answers.

Editing applies addition, deletion, or substitution objectives to the supplied molecule. Generation refines a selected seed against atom-count constraints, functional-group constraints, or a textual description. Public structural predicates are checked programmatically; open-ended text goals can additionally use model-based semantic comparisons. Each candidate-search step executes, checks, deduplicates, and ranks patches. Invalid or exhausted frontiers can invoke repair or a reserved alternative; visible satisfaction can stop search early. Table 6 specifies the search limits.

Candidate ordering. Equation 1 is the online editing score. If edit locality is inapplicable, the remaining weights are normalized by 0.85; a no-op with a target instead receives zero locality without renormalization. Candidates are ordered by validity, visible progress, score, confidence, and shorter program length. Composition adds a true-merge tie-break after the score, and acceptance follows Equation 2. A revision can reproduce a parent. These decisions use public constraints and model assessments, never the final reference answer.

The pseudocode summarizes task-specific graph routes (Besta et al., 2024; Yao et al., 2023a;b).

Algorithm: task-adapted MolGoT

```
Input:  instruction, public constraints, optional SMILES
1.  Initialize a supplied molecule or select a checked seed.
2.  Construct persistent IDs, regions, and boundary relations.
3.  Dispatch the reasoning route:
  carbon:  integrate local counts; audit SMILES; adjudicate;
  ring:  combine topology and SMILES-closure judgments;
  formula:  integrate local evidence with the full molecule;
  group count / equivalence:  select evidence; global readout;
  editing / generation:
    initialize the candidate frontier;
    repeat until the depth limit or visible stopping condition:
      propose localized JSON patches for each parent;
      execute, check, deduplicate, and retain candidates;
      repair failed states or use reserved alternatives if needed;
    retain the best individual state;
    replay compatible programs from at most two parents;
    accept only compositions passing the candidate gate;
    if eligible, request and check a revision under the same gate;
    select an accepted state or the best single branch.
4.  Return the answer and save the graph and call ledger.
5.  Apply final evaluation after the answer is fixed.
```

B.2 FROZEN ENDPOINTS AND INFERENCE-TIME CHECKS

Final answers and reference labels. The frozen evaluator consumes the parsed answer or parsed SMILES and the frozen task record. Carbon and ring counting require an exactly parsed integer equal to the recorded label. Functional-group counting requires an exact match for every requested group. Formula answers are parsed into element counts and net charge, so element order may differ but counts and charge must agree. Equivalence answers are normalized to a Boolean and compared with the frozen label. The molecular-property helper uses `GetRingInfo().NumRings()` and `CalcMolFormula`; final accuracy compares with the frozen labels rather than regenerating them during scoring. The archived label metadata identifies RDKit-derived labels, with `CalcMolFormula` explicitly named for formula prediction. This specification does not equate the ring endpoint with the graph cycle rank.

Functional-group definitions. For final editing and generation checks, groups are counted with `GetSubstructMatches` under the frozen SMARTS dictionary. The calls retain the RDKit defaults and do not impose an additional non-overlap or mutually exclusive assignment across groups. The released scoring dictionary overrides nitro with `[N+](=[OX1])[O-]` and adds anhydride, sulfide and borane definitions. Sulfide counts subtract the number of matches of the disulfide pattern `[#16X2H0][#16X2H0]`. The complete machine-readable dictionary and scoring source accompany this specification. The online dictionary differs from the final dictionary; in particular, it retains a different nitro pattern and lacks these added definitions.

Conservative editing witness. An edited structure must first be parseable and sanitized, with the requested old-group count decreasing by exactly one and/or the requested new-group count increasing by exactly one. A witness is then required. A removable mask is a nonempty connected atom subset of one matching group occurrence, smaller than the whole molecule and attached to its residual graph. For addition, removing a new-group mask from the product, with size equal to the positive atom-count increase, must recover a graph isomorphic to the input. For deletion, removing an old-group mask from the input, with size equal to the positive atom-count decrease, must recover the product. For substitution, masks from an old-group occurrence and a new-group occurrence must leave isomorphic residual graphs, including the same number of removed neighbors at corresponding residual atoms.

Residual-node matching preserves atomic number, formal charge, isotope, aromaticity and radical-electron count; residual edges preserve bond type and stereo. Total hydrogen count and CIP code must match away from cut boundaries. These two checks are relaxed at cut-boundary atoms; the remaining node attributes are not. The witness compares input and output structures and does not replay the model's edit program. Failure to find it is a failure of this conservative endpoint, not proof that every chemical interpretation is invalid. The editing endpoint does not add a global single-component or blanket radical-exclusion rule.

Generation endpoints. Atom-constrained generation requires exact requested element counts, no unrequested non-hydrogen elements, and one connected component. Hydrogen is counted when explicitly requested. Group-constrained generation requires exact requested group counts and one connected component, without globally prohibiting other groups. ChEBI generation requires identity of canonical isomeric SMILES after clearing atom-map numbers; its chiral Morgan-fingerprint similarity is diagnostic only.

Visible constraints and progress. Let C_{vis} denote the implemented mean of the applicable public-constraint scores. Invalid candidates receive zero. Atom-count constraints score the fraction of requested positive counts matched exactly. Functional-group count constraints score the fraction of requested positive counts reached or exceeded. For editing, a required group must strictly increase relative to the input, and a forbidden group must strictly decrease; these online checks do not require a change of exactly one. In non-editing tasks, a single required-group constraint checks presence. The online checks do not uniformly impose the final restrictions on extra elements or connected components. With explicit public constraints, candidate progress is set to $p = C_{\mathrm{vis}}$, overriding the proposal's predicted progress. Without them, progress can be updated by the comparative model assessment, clipped to $[0, 1]$. Analysis and understanding use answer-presence checks online, not a hidden-label correctness check.

Locality and ranking. For executable edits, let F be the numeric root-atom footprint of the program, excluding namespaced newly created atoms, and let U be the union of the target regions and their boundary atoms. The locality term is $L = |F \cap U|/|F|$ when the footprint is available. With no explicit footprint, the implementation can use a separately audited MCS-based proxy, averaging the fraction of changed anchors in U over the evaluated mappings and penalizing unanchored new atoms. Mapping enumeration is bounded and reports truncation or timeout. The editing score is

$$v = 0.35V + 0.35C_{\text{vis}} + 0.15S + 0.15L,$$

where S is radius-2, 2,048-bit Morgan Tanimoto similarity to the root, without the chirality option used by the ChEBI diagnostic. If locality is inapplicable, the other terms are divided by 0.85; a detected no-op instead receives zero locality contribution without renormalization. Single states are ordered lexicographically by $(V, p, v, c, -|P|)$, with c the model confidence and $|P|$ the program length. Aggregation inserts a true-merge indicator after v.

Composition and selection. Compatible parents have executable programs from different root branches, no shared operation signatures, and disjoint root-atom footprints. Their combined program is replayed on the common root. A composition must be valid and no worse than the best individual in both progress and score, with tolerance 10^{-12}. Without public constraints it must also improve progress strictly beyond that tolerance. Passing this gate does not by itself select the composition: its aggregation-ranking tuple must exceed that of the best individual. Repair, when enabled, requires public constraints, no eligible composition, and two valid compatible top-ranked parents, and remains subject to the same gate and ranking comparison. A repair may reproduce a parent; the true-merge audit additionally requires contributions from at least two parents, successful replay, and a result distinct from every parent.

Seed fallback. The six-seed preparation path removes reference-answer fields, requests and repairs model seeds, discards invalid or duplicate canonical structures, and ranks the valid pool using the visible score. If the entire pool is invalid, the deterministic fallback uses a fixed example for an available public functional-group request, otherwise a carbon chain guided by the requested carbon count, clipped to 1–40 atoms, or ethanol when no such constraint is available. This fallback supplies a valid starting structure, not a guarantee of final task success.

C PROMPTS AND STRUCTURED RESPONSES

These boxes reproduce recorded prompt excerpts and JSON from GPT example 67635. Bracketed omissions are editorial. Prompts are role-specific: expansion requests executable alternatives, while revision receives parent states and candidate-level feedback. No hidden target molecule or final correctness label is included.

Expansion prompt: recorded user message, excerpt

You are performing executable molecular graph search.
`/no_think`

Output contract:

- Return exactly one valid JSON object.
- Do not include markdown fences, prose outside JSON, chain-of-thought, or `<think>` tags.
- Return the JSON object immediately; do not explain your reasoning before or after it.
- Keep reasoning concise inside the requested JSON string fields only.

Task:
Modify the molecule `Fc1ccccc1COCC1(CS)CCOCC1` by adding a amide.

Visible constraint ledger:

```
{"required_functional_group": "amide",
 "task_family": "editing", "task_name": "tomg_atom_addition"}
```

[Goal dependencies, atom-ID map, persistent atom table, chemical subgraphs, and already executed program omitted.]

Generate up to 3 diverse NEXT patches. A patch is a short list of graph operations, not a complete rewritten molecule.

Rules:

1. Refer only to atom_id values shown above, except new_atom_id created inside the same patch.
2. Keep each patch minimal and localized; preserve the scaffold unless the task asks otherwise.
3. Use valid element symbols and single/double/triple/aromatic bond types.
4. Diversify candidates by target region or operation type.
5. Do not use a hidden target or reference answer.
6. candidate_smiles is only an emergency fallback when no graph operation can express the edit; normally leave it empty.
7. Return only the requested JSON object.

Executable response: first recorded candidate


```
{
  "candidate_id": "c1",
  "target_subgraph_ids": ["sub2"],
  "operations": [
    {"op": "add_atom", "anchor_atom_id": 10,
     "element": "C", "bond_type": "single", "new_atom_id": 19},
    {"op": "add_atom", "anchor_atom_id": 19,
     "element": "O", "bond_type": "double", "new_atom_id": 20},
    {"op": "add_atom", "anchor_atom_id": 19,
     "element": "N", "bond_type": "single", "new_atom_id": 21}
  ],
  "candidate_smiles": "",
  "expected_constraint_effect":
    "Adds a primary amide substituent, C(=O)N, at carbon 10.",
  "predicted_goal_progress": 1.0,
  "confidence": 0.86
}
```


Whitespace and key order are reformatted above; recorded values are unchanged. The executor subsequently namespaces new IDs by patch. Other allowed operations are atom replacement/deletion, bond addition/deletion/change, and fragment attachment. Confidence is model-provided metadata; it is not a correctness probability calibrated by the final evaluator.

Revision prompt: recorded user message, excerpt

[Task, root structural context, and the two complete parent states omitted.]

Observed composition errors:

```
["no composed candidate improved over the best parent"]
```

Return one minimal combined executable patch using only the operation schema from the parent programs.
Preserve a real contribution from at least two parents. Do not rewrite a complete SMILES.

Return:


```
{"candidates": [{"candidate_id": "repair",
  "target_subgraph_ids": [], "operations": [],
  "candidate_smiles": "", "confidence": 0.0}]}
```


The request for two-parent contributions does not guarantee that the model follows it. In the recorded success below, the revision reproduces P1 and is retained as a valid single-parent result. Keeping the requested operation separate from its realized outcome is necessary for interpreting the graph.

D COMPLETE RESULTS AND ADDITIONAL METRICS

D.1 TASK-LEVEL PERFORMANCE

Figure 6 compares the four pipelines within each task category, retaining all ten model configurations. Category accuracy averages its constituent tasks. Tables 8 and 9 provide the corresponding eleven-task results. The category summaries are descriptive: they do not imply equal difficulty or identical controller routes within a category.

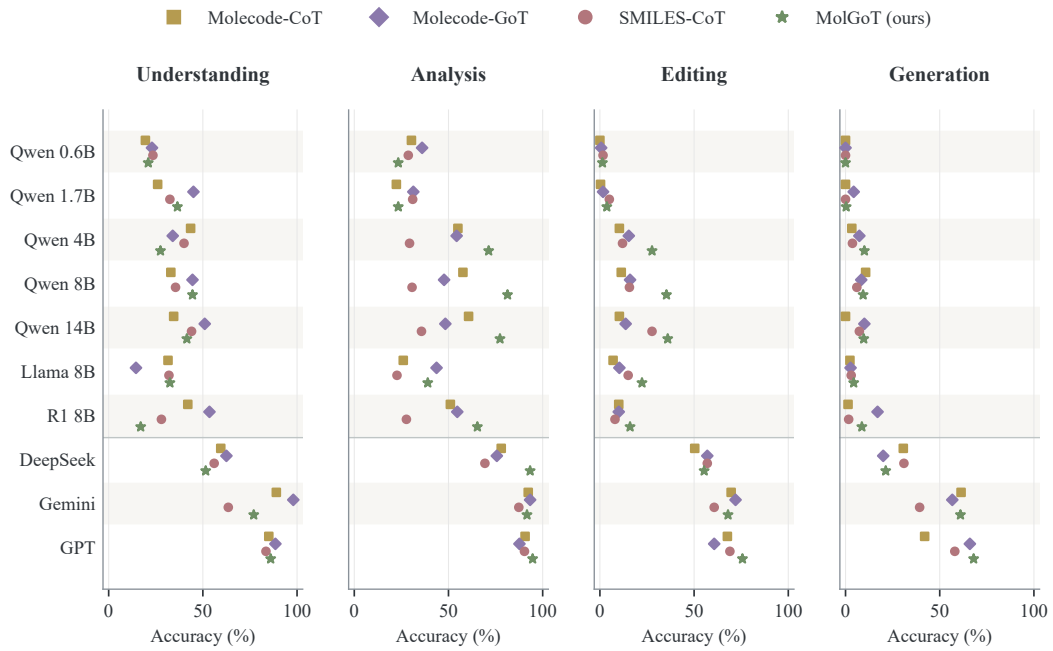

Figure 6: **Performance profiles by molecular task category.** Each row shows one model and each symbol one pipeline. The shared 0–100% scale permits comparison across categories. All main configurations, including Gemini, are included.

Excluding carbon and ring counting from evaluation leaves nine tasks. Table 7 compares the saved default MolGoT and single-answer CoT outcomes on common IDs; it supplies Figure 5c and is not token-matched. The shared-guidance iterative comparison is reported separately in Table 27.

Table 7: **Default MolGoT versus CoT beyond carbon/ring counting.** Accuracy is in percent; differences and exploratory intervals are percentage points. Paired n is the common completed denominator.

Model	Paired n	CoT (%)	MolGoT (%)	Δ (pp)	95% interval
Qwen 0.6B	180	12.78	12.22	-0.56	$[-2.78, 1.67]$
Qwen 1.7B	180	20.00	17.78	-2.22	$[-7.22, 2.22]$
Qwen 4B	180	23.33	29.44	$+6.11$	$[-0.56, 12.78]$
Qwen 8B	450	22.00	33.56	$+11.56$	$[7.78, 15.33]$
Qwen 14B	450	29.33	33.56	$+4.22$	$[0.44, 8.00]$
Llama-3.1-8B	450	15.56	23.78	$+8.22$	$[4.00, 12.22]$
R1-Qwen3-8B	180	16.67	17.78	$+1.11$	$[-4.44, 6.67]$
DeepSeek V4.1 Flash	450	50.22	45.33	-4.89	$[-9.11, -0.44]$
GPT-5.6 Luna	450	69.11	74.44	$+5.33$	$[2.00, 8.67]$

Table 8: **Task accuracy: open-weight configurations (%).** Form./Equiv.: formula/equivalence; C/Ring/FG: the three count tasks; Add/Del/Sub: editing; Atom/Group/Text: the three generation tasks.

Model	Pipeline	Form.	Equiv.	C	Ring	FG	Add	Del	Sub	Atom	Group	Text
Qwen3-0.6B	Molecode-CoT	1	38	59	4	28	0	0	0	0	0	0
	Molecode-GoT	0	46	73	1	34	0	2	0	0	0	0
	SMILES-CoT	0	47	1	20	65	2	1	2	0	0	0
	MolGoT (ours)	0	42	5	1	64	2	2	0	0	0	0
Qwen3-1.7B	Molecode-CoT	0	52	8	6	53	0	0	1	0	0	0
	Molecode-GoT	0	90	21	12	61	0	4	1	13	0	0
	SMILES-CoT	0	65	3	24	66	2	7	6	0	0	0
	MolGoT (ours)	0	73	6	0	64	5	6	0	0	1	0
Qwen3-4B	Molecode-CoT	1	86	74	25	66	0	9	22	10	0	0
	Molecode-GoT	1	67	94	14	55	0	31	15	21	0	1
	SMILES-CoT	0	80	2	16	70	12	9	15	3	8	0
	MolGoT (ours)	0	55	60	84	70	11	47	25	27	3	0
Qwen3-8B	Molecode-CoT	1	65	89	35	49	0	14	20	32	0	0
	Molecode-GoT	0	89	91	17	35	5	18	25	25	0	0
	SMILES-CoT	0	71	1	33	58	21	9	17	3	15	0
	MolGoT (ours)	0	89	75	92	77	21	53	32	20	8	0
Qwen3-14B	Molecode-CoT	2	67	100	14	68	0	11	20	0	0	0
	Molecode-GoT	2	100	98	18	29	5	7	29	30	0	0
	SMILES-CoT	2	86	9	25	73	27	13	43	5	17	0
	MolGoT (ours)	3	80	88	71	73	23	48	37	17	12	0
Llama-3.1-8B	Molecode-CoT	2	61	9	20	49	0	4	17	7	0	0
	Molecode-GoT	3	26	77	3	51	1	19	11	7	1	0
	SMILES-CoT	0	64	10	32	26	22	5	18	0	8	1
	MolGoT (ours)	0	65	45	5	67	16	31	20	5	8	0
R1-Qwen3-8B	Molecode-CoT	41	43	97	4	52	2	8	20	2	1	1
	Molecode-GoT	19	88	100	2	62	2	12	16	32	13	6
	SMILES-CoT	0	56	14	9	60	1	11	12	4	1	0
	MolGoT (ours)	0	34	93	48	55	15	21	12	20	5	1

Table 9: **Task accuracy: API configurations (%).** Columns follow Table 8.

Model	Pipeline	Form.	Equiv.	C	Ring	FG	Add	Del	Sub	Atom	Group	Text
DeepSeek V4.1 Flash	Molecode-CoT	27	92	100	54	80	35	32	84	72	15	5
	Molecode-GoT	25	100	100	60	67	63	36	72	40	16	4
	SMILES-CoT	17	95	43	84	81	61	23	87	19	51	23
	MolGoT (ours)	20	83	98	98	84	58	25	83	24	26	14
Gemini 3 Flash	Molecode-CoT	84	94	100	99	78	79	36	94	83	66	35
	Molecode-GoT	96	100	100	99	81	84	37	95	75	71	24
	SMILES-CoT	46	81	83	98	81	62	29	91	33	60	25
	MolGoT (ours)	74	80	100	95	80	97	26	81	85	69	29
GPT-5.6 Luna	Molecode-CoT	79	91	100	85	87	68	46	89	71	39	16
	Molecode-GoT	77	100	100	92	71	71	37	74	97	75	26
	SMILES-CoT	73	94	91	96	84	81	35	91	76	74	24
	MolGoT (ours)	78	94	100	98	86	93	49	85	92	81	31

D.2 COMPLEMENTARY EVALUATION ENDPOINTS

The following tables use the definitions in Appendix A.4. They distinguish valid structure production, task satisfaction, and reference proximity. For example, a parseable count can be wrong, and a valid edited molecule can satisfy the group-count target while failing the preservation check. These quantities are therefore reported separately rather than combined into another headline score.

Table 10: **Additional metrics: open-weight configurations.** Valid, Goal, Pres., and Parse are percentages; Pres. is conditional on goal satisfaction. MAE uses parseable count answers. Sim. is ChEBI reference Tanimoto, with invalid or missing outputs assigned zero.

Model	Pipeline	Valid↑	Goal↑	Pres.↑	MAE↓	Parse↑	Sim.↑
Qwen3-0.6B	Molecode-CoT	23.2	3.3	0.0	4.67	100.0	0.000
	Molecode-GoT	50.3	15.0	4.4	1.79	84.0	0.000
	SMILES-CoT	40.8	1.7	100.0	5.06	99.7	0.005
	MolGoT (ours)	100.0	5.3	25.0	11.43	98.3	0.053
Qwen3-1.7B	Molecode-CoT	37.0	1.3	25.0	2.55	99.0	0.000
	Molecode-GoT	65.7	11.7	14.3	2.22	97.0	0.000
	SMILES-CoT	55.3	8.3	60.0	3.06	99.7	0.036
	MolGoT (ours)	100.0	7.7	47.8	15.54	100.0	0.064
Qwen3-4B	Molecode-CoT	40.5	14.0	73.8	0.71	100.0	0.001
	Molecode-GoT	61.8	23.0	66.7	0.80	100.0	0.018
	SMILES-CoT	55.5	22.7	52.9	3.93	99.7	0.066
	MolGoT (ours)	100.0	37.7	73.5	1.08	99.3	0.091
Qwen3-8B	Molecode-CoT	28.5	21.3	53.1	0.60	97.7	0.000
	Molecode-GoT	60.8	19.7	81.4	0.93	100.0	0.015
	SMILES-CoT	59.7	26.7	58.8	1.99	100.0	0.094
	MolGoT (ours)	100.0	51.7	68.4	0.28	100.0	0.117
Qwen3-14B	Molecode-CoT	21.0	20.0	51.7	0.72	100.0	0.000
	Molecode-GoT	70.3	16.0	85.4	2.40	100.0	0.015
	SMILES-CoT	64.7	47.3	58.5	1.48	100.0	0.117
	MolGoT (ours)	100.0	51.7	69.7	0.71	100.0	0.144
Llama-3.1-8B	Molecode-CoT	17.2	13.0	53.8	2.46	98.0	0.000
	Molecode-GoT	59.7	14.3	72.1	1.20	65.0	0.011
	SMILES-CoT	63.2	32.0	46.9	2.09	99.7	0.108
	MolGoT (ours)	100.0	39.0	57.3	1.92	92.3	0.130
R1-Qwen3-8B	Molecode-CoT	15.7	17.7	56.6	0.15	55.3	0.011
	Molecode-GoT	61.3	10.7	93.8	1.06	79.7	0.075
	SMILES-CoT	6.8	9.7	82.8	0.33	34.7	0.000
	MolGoT (ours)	100.0	21.3	75.0	0.44	78.3	0.079

Table 11: **Additional metrics: API configurations.** Definitions match Table 10.

Model	Pipeline	Valid↑	Goal↑	Pres.↑	MAE↓	Parse↑	Sim.↑
DeepSeek V4.1 Flash	Molecode-CoT	64.5	70.7	71.2	0.31	99.7	0.125
	Molecode-GoT	78.7	73.7	77.4	0.30	99.7	0.163
	SMILES-CoT	94.5	85.0	67.1	0.67	100.0	0.599
	MolGoT (ours)	100.0	84.3	65.6	0.07	100.0	0.511
Gemini 3 Flash	Molecode-CoT	91.8	97.3	71.6	0.10	100.0	0.606
	Molecode-GoT	86.7	96.7	74.5	0.09	100.0	0.422
	SMILES-CoT	91.0	93.3	65.0	0.14	100.0	0.611
	MolGoT (ours)	100.0	92.7	73.4	0.11	100.0	0.653
GPT-5.6 Luna	Molecode-CoT	72.5	85.7	79.0	0.27	100.0	0.234
	Molecode-GoT	92.5	77.0	78.8	0.17	100.0	0.421
	SMILES-CoT	93.3	93.3	73.9	0.13	100.0	0.546
	MolGoT (ours)	100.0	92.7	81.7	0.06	100.0	0.614

Table 12: **Counting errors and generation checks: open-weight configurations.** MAE is conditional on parseability. Count constraints average the fraction of requested predicates satisfied; Connected is the fraction of single-component outputs. Invalid generation outputs contribute zero.

		Counting MAE ↓			Count constraints (%) ↑		Connected (%) ↑	
Model	Pipeline	Carbon	Ring	FG	Atoms	Groups	Atoms	Groups
Qwen3-0.6B	Molecode-CoT	1.67	6.47	5.86	0.0	0.0	0.0	1.0
	Molecode-GoT	1.28	3.70	0.40	1.5	0.0	1.0	0.0
	SMILES-CoT	12.13	2.60	0.44	0.5	2.9	5.0	21.0
	MolGoT (ours)	9.68	24.17	0.45	5.0	7.6	100.0	100.0
Qwen3-1.7B	Molecode-CoT	5.24	1.81	0.59	2.9	0.0	10.0	3.0
	Molecode-GoT	4.22	2.03	0.42	68.9	0.8	48.0	5.0
	SMILES-CoT	7.45	1.35	0.39	2.2	6.3	12.0	72.0
	MolGoT (ours)	8.69	37.50	0.44	9.9	9.8	97.0	97.0
Qwen3-4B	Molecode-CoT	0.49	1.25	0.38	66.8	0.2	13.0	13.0
	Molecode-GoT	0.16	1.69	0.54	32.5	1.8	31.0	11.0
	SMILES-CoT	9.90	1.55	0.34	13.0	21.7	26.0	76.0
	MolGoT (ours)	0.64	2.21	0.38	58.6	21.1	99.0	100.0
Qwen3-8B	Molecode-CoT	0.13	1.07	0.59	42.5	0.5	35.0	2.0
	Molecode-GoT	0.26	1.72	0.81	43.4	1.2	46.0	7.0
	SMILES-CoT	4.53	0.97	0.47	10.0	28.2	29.0	67.0
	MolGoT (ours)	0.50	0.08	0.27	58.1	29.3	97.0	100.0
Qwen3-14B	Molecode-CoT	0.00	1.76	0.39	0.0	0.0	0.0	0.0
	Molecode-GoT	0.02	6.30	0.88	75.7	1.8	80.0	13.0
	SMILES-CoT	3.09	1.07	0.29	14.0	36.6	26.0	79.0
	MolGoT (ours)	0.18	1.66	0.29	61.7	32.2	91.0	85.0
Llama-3.1-8B	Molecode-CoT	5.39	1.46	0.54	11.2	0.0	9.0	1.0
	Molecode-GoT	0.57	2.54	0.51	10.5	2.0	10.0	7.0
	SMILES-CoT	3.84	1.24	1.20	12.0	23.3	32.0	69.0
	MolGoT (ours)	1.91	3.45	0.40	42.4	31.2	85.0	90.0
R1-Qwen3-8B	Molecode-CoT	0.23	0.00	0.21	6.2	2.0	3.0	2.0
	Molecode-GoT	0.00	2.92	0.25	34.0	17.0	34.0	22.0
	SMILES-CoT	0.77	0.10	0.12	4.0	1.5	5.0	3.0
	MolGoT (ours)	0.06	1.14	0.11	45.8	16.0	100.0	100.0

Table 13: **Counting errors and generation checks: API configurations.** Definitions match Table 12.

		Counting MAE ↓			Count constraints (%) ↑		Connected (%) ↑	
Model	Pipeline	Carbon	Ring	FG	Atoms	Groups	Atoms	Groups
DeepSeek V4.1 Flash	Molecode-CoT	0.00	0.68	0.25	75.5	28.3	78.0	39.0
	Molecode-GoT	0.00	0.53	0.37	47.5	34.6	43.0	45.0
	SMILES-CoT	1.65	0.17	0.19	54.6	74.8	81.0	85.0
	MolGoT (ours)	0.02	0.02	0.17	69.1	64.0	91.0	80.0
Gemini 3 Flash	Molecode-CoT	0.00	0.01	0.28	87.0	77.8	87.0	82.0
	Molecode-GoT	0.00	0.01	0.26	77.0	80.2	76.0	85.0
	SMILES-CoT	0.20	0.02	0.21	53.7	78.8	71.0	95.0
	MolGoT (ours)	0.00	0.06	0.26	98.6	88.1	97.0	99.0
GPT-5.6 Luna	Molecode-CoT	0.00	0.66	0.15	72.0	50.8	72.0	53.0
	Molecode-GoT	0.00	0.14	0.36	100.0	89.6	99.0	91.0
	SMILES-CoT	0.18	0.04	0.18	84.7	86.8	86.0	94.0
	MolGoT (ours)	0.00	0.02	0.17	100.0	92.1	100.0	100.0

E ADDITIONAL INFERENCE CONTROLS

E.1 SAMPLING BUDGETS AND REALIZED TOKEN USE

The sampling controls use independent chain-of-thought responses (Wei et al., 2022). This sensitivity analysis retains majority vote, in contrast to the public-check selector in Section 4.3.1; neither uses reference-based best-of-N selection (Wang et al., 2023; Brown et al., 2024; Snell et al., 2025; Muennighoff et al., 2025). Original SMILES and context-enhanced SMILES use the corresponding MolGoT token budget; Molecode sampling uses Molecode-GoT's budget. For each example, the selected prefix is the first completed-answer prefix reaching the nominal target when attainable. Ties in the majority vote use the earliest candidate. Completing a whole answer can overshoot the target, so tables report attained tokens.

Figure 7 retains majority selection for all nine control configurations; Figure 4 instead uses the public-check selector. In Figure 7, the normalized horizontal axis uses each model's MolGoT mean token cost solely for display; it does not redefine the experiment's per-example target. Its vertical axis is the accuracy difference from MolGoT. The three points on each curve retain the $0.5\times$, $1\times$, and $2\times$ target order. Additional context changes the cost of a sampled answer and thus the number of answers affordable within a nominal budget.

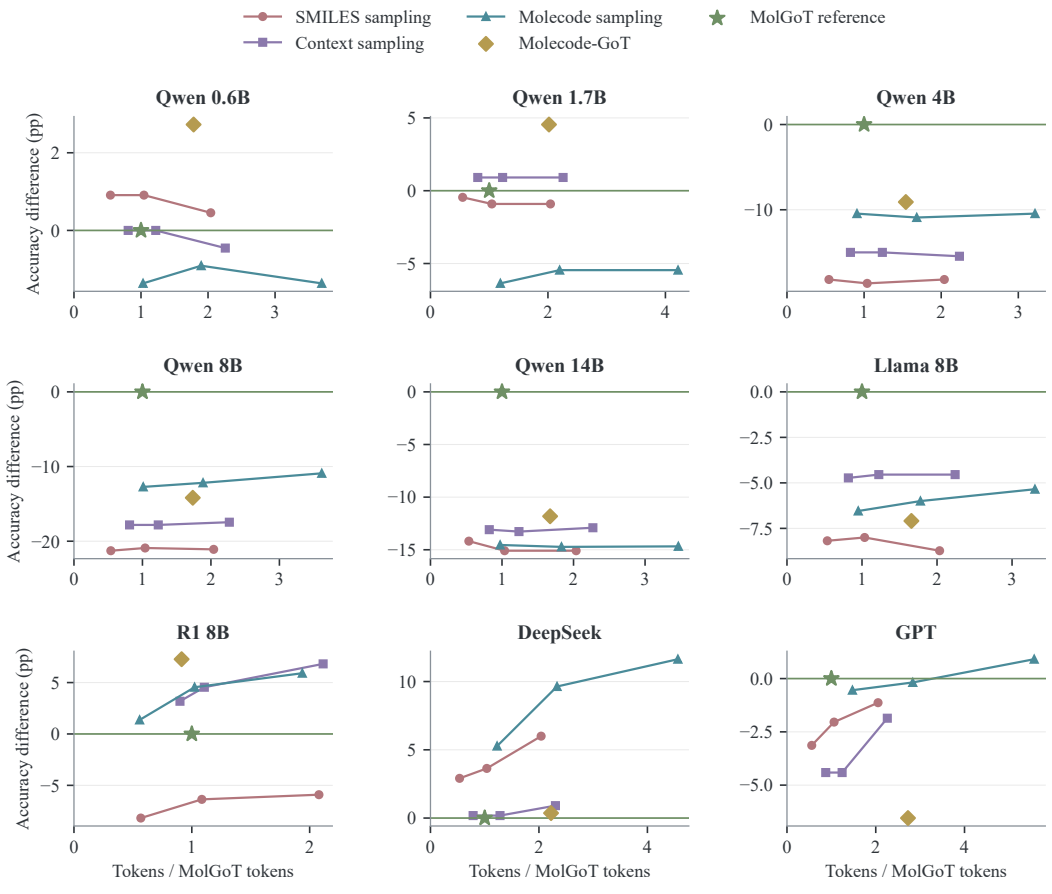

Figure 7: **Majority-selection sensitivity across all control models.** The star at $(1, 0)$ is MolGoT; curves show sampled controls and diamonds show Molecode-GoT. Positive vertical values favor the comparator. Ratios are ratios of mean attained tokens, not exact per-example budget matches. The curves use completed outcomes; paired inference uses common IDs.

Table 14: **Original-input SMILES sampling, majority selection.** Accuracy (%) and attained tokens in thousands at each nominal multiplier of the MolGoT reference budget.

	0.5×		1×		2×	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	12.27	3.11	12.27	5.97	11.82	11.66
Qwen3-1.7B	15.91	3.09	15.45	5.88	15.45	11.51
Qwen3-4B	20.91	3.40	20.45	6.49	20.91	12.74
Qwen3-8B	21.09	3.22	21.45	6.22	21.27	12.20
Qwen3-14B	27.45	3.32	26.55	6.42	26.55	12.61
Llama-3.1-8B	16.55	3.41	16.73	6.57	16.00	12.92
R1-Qwen3-8B	18.64	16.40	20.45	31.37	20.91	60.10
DeepSeek V4.1 Flash	58.00	2.98	58.73	5.77	61.09	11.34
GPT-5.6 Luna	75.77	3.67	76.87	6.99	77.78	13.52

Table 15: **Context-enhanced SMILES sampling, majority selection.** Accuracy (%) and attained kTok; the reference is MolGoT. Structural context and supplementary task guidance are provided to this control.

	0.5×		1×		2×	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	11.36	4.62	11.36	6.97	10.91	12.91
Qwen3-1.7B	17.27	4.53	17.27	6.92	17.27	12.72
Qwen3-4B	24.09	5.14	24.09	7.72	23.64	13.96
Qwen3-8B	24.55	4.86	24.55	7.37	24.91	13.58
Qwen3-14B	28.55	5.09	28.36	7.66	28.73	14.06
Llama-3.1-8B	20.00	5.20	20.18	7.77	20.18	14.23
R1-Qwen3-8B	30.00	26.01	31.36	31.98	33.64	61.12
DeepSeek V4.1 Flash	55.27	4.37	55.27	7.12	56.00	12.80
GPT-5.6 Luna	74.50	5.75	74.50	8.19	77.05	14.89

Table 16: **Original-input Molecode sampling, majority selection.** Accuracy (%) and attained kTok; multipliers refer to Molecode-GoT, not MolGoT.

	0.5×		1×		2×	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	10.00	5.89	10.45	10.85	10.00	21.13
Qwen3-1.7B	10.00	6.70	10.91	12.38	10.91	23.74
Qwen3-4B	28.64	5.67	28.18	10.50	28.64	20.08
Qwen3-8B	29.64	6.05	30.18	11.28	31.45	21.65
Qwen3-14B	27.09	6.03	26.91	11.34	26.96	21.45
Llama-3.1-8B	18.18	6.03	18.73	11.30	19.38	20.99
R1-Qwen3-8B	28.18	16.10	31.36	29.59	32.73	56.01
DeepSeek V4.1 Flash	60.36	6.82	64.73	12.97	66.73	25.35
GPT-5.6 Luna	78.36	9.72	78.73	18.67	79.82	36.70

Table 17: **Graph references for the main budget figure.** Accuracy (%) and kTok summarize the recorded graph conditions, separately from the main-cohort totals.

Model	Reference / control	Acc.	kTok
Qwen3-8B	MolGoT (ours)	42.36	5.98
	Molecode-GoT	28.18	10.38
R1-Qwen3-8B	MolGoT (ours)	26.82	28.91
	Molecode-GoT	34.09	26.41
DeepSeek V4.1 Flash	MolGoT (ours)	55.09	5.56
	Molecode-GoT	55.45	12.35
GPT-5.6 Luna	MolGoT (ours)	78.91	6.59
	Molecode-GoT	72.36	17.96

Table 18: **Paired graph–sampling differences at the nominal $1\times$ target.** Values are percentage points with exploratory 95% intervals. Each graph is compared with sampling using its own reference budget.

	MolGoT (ours) minus sampling		Molecode-GoT minus
Model	Original SMILES	With graph context	Molecode sampling
Qwen3-0.6B	-0.91 $[-3.64, 1.82]$	$+0.00$ $[-3.18, 3.18]$	$+3.64$ $[-0.45, 7.73]$
Qwen3-1.7B	$+0.91$ $[-3.64, 5.45]$	-0.91 $[-5.45, 3.64]$	$+10.00$ $[5.45, 15.00]$
Qwen3-4B	$+18.64$ $[12.27, 25.00]$	$+15.00$ $[9.55, 20.45]$	$+1.82$ $[-3.64, 7.27]$
Qwen3-8B	$+20.91$ $[17.09, 24.73]$	$+17.82$ $[14.18, 21.64]$	-2.00 $[-5.45, 1.45]$
Qwen3-14B	$+15.09$ $[11.45, 18.73]$	$+13.27$ $[9.64, 16.91]$	$+2.91$ $[-0.18, 6.00]$
Llama-3.1-8B	$+8.00$ $[4.18, 12.00]$	$+4.55$ $[1.09, 8.00]$	-1.09 $[-4.73, 2.55]$
R1-Qwen3-8B	$+6.36$ $[1.36, 11.36]$	-4.55 $[-10.00, 0.91]$	$+2.73$ $[-1.82, 7.27]$
DeepSeek V4.1 Flash	-3.64 $[-7.27, 0.00]$	-0.18 $[-3.82, 3.45]$	-9.27 $[-13.09, -5.27]$
GPT-5.6 Luna	$+2.19$ $[-0.55, 4.92]$	$+4.37$ $[1.82, 6.92]$	-6.36 $[-9.27, -3.45]$

Table 19: **Attained budget ratios.** Sampling columns give mean per-example actual/target ratios at $1\times$. The final column is the ratio of graph/iteration mean tokens under shared guidance. Overshoot and different per-answer costs preclude exact budget equality.

Model	SMILES sampling	Context sampling	Molecode sampling	Graph / iterative
Qwen3-0.6B	1.049	1.274	1.087	0.750
Qwen3-1.7B	1.048	1.267	1.116	0.737
Qwen3-4B	1.047	1.323	1.118	0.707
Qwen3-8B	1.046	1.314	1.107	0.670
Qwen3-14B	1.046	1.321	1.127	0.686
Llama-3.1-8B	1.042	1.276	1.094	0.699
R1-Qwen3-8B	1.174	1.151	1.154	0.851
DeepSeek V4.1 Flash	1.049	1.342	1.066	0.658
GPT-5.6 Luna	1.075	1.331	1.066	0.735

E.2 SELECTION WITH PUBLIC CHECKS

The budget traces save both majority and public-check choices for the same sampled prefixes. Public-check selection takes the candidate with the highest inference-time score, breaking ties in sampling order. It uses the task's visible constraints and checks, not the frozen correctness label. Majority groups nonstructural answers after trimming and lowercasing and molecular answers by the saved canonical molecular key; ties favor the earliest candidate. If every key is empty, it selects the last candidate. Different valid molecules can form different voting groups. The two selectors therefore test distinct selection rules without changing the sampled proposals or their accumulated cost.

The tables below replay the saved selection indices and final outcomes rather than rescore molecules with a newer toolkit. All 60,952 saved public-check prefix choices match this argmax rule. Local re-canonicalization with RDKit 2026.03.6 differs on 1,735 of 116,232 structural candidates; one saved molecular key also differs between the candidate and voting ledger. We therefore retain the frozen keys and selections. This sensitivity concerns canonical grouping, and does not alter the public-check argmax audit. The portable analysis includes per-prefix outcomes, common IDs, attained tokens, and both stopping brackets.

Each table uses completed graph–sampling pairs. The per-condition denominators are 220 for Qwen3-0.6B, 1.7B, 4B and R1, and 550 for Qwen3-8B, 14B, Llama and DeepSeek. At the $2\times$ Molecode upper bracket, Qwen3-14B has 549 pairs and Llama has 547 because some saved prefixes do not attain that target. GPT has 549 pairs for original and context-enhanced SMILES and 550 for Molecode at all three displayed budgets. The graph reference uses 550 GPT answers. These small coverage differences can change paired and marginal point estimates.

Table 20: **Original SMILES sampling with public-check selection.** Accuracy (%) and attained kTok. SMILES targets use MolGoT budgets; Molecode targets use Molecode-GoT budgets.

	$0.5\times$		$1\times$		$2\times$	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	12.27	3.11	12.73	5.97	12.73	11.66
Qwen3-1.7B	16.36	3.09	16.82	5.88	17.27	11.51
Qwen3-4B	22.73	3.40	23.18	6.49	23.64	12.74
Qwen3-8B	21.82	3.22	22.00	6.22	22.18	12.20
Qwen3-14B	28.00	3.32	28.36	6.42	28.55	12.61
Llama-3.1-8B	19.27	3.41	20.00	6.57	20.36	12.92
R1-Qwen3-8B	19.09	16.40	20.91	31.37	22.27	60.10
DeepSeek	58.73	2.98	60.00	5.77	59.64	11.34
GPT	77.23	3.67	76.87	6.99	76.50	13.52

Table 21: **Context-enhanced SMILES sampling with public-check selection.** Accuracy (%) and attained kTok. SMILES targets use MolGoT budgets; Molecode targets use Molecode-GoT budgets.

	$0.5\times$		$1\times$		$2\times$	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	11.36	4.62	11.36	6.97	11.36	12.91
Qwen3-1.7B	17.27	4.53	17.27	6.92	17.27	12.72
Qwen3-4B	24.09	5.14	24.09	7.72	24.55	13.96
Qwen3-8B	24.55	4.86	25.45	7.37	25.64	13.58
Qwen3-14B	28.55	5.09	29.45	7.66	30.36	14.06
Llama-3.1-8B	20.18	5.20	20.55	7.77	20.73	14.23
R1-Qwen3-8B	30.00	26.01	30.91	31.98	34.09	61.12
DeepSeek	55.27	4.37	56.55	7.12	57.82	12.80
GPT	74.50	5.75	75.23	8.19	76.32	14.89

Table 22: **Molecode sampling with public-check selection.** Accuracy (%) and attained kTok. SMILES targets use MolGoT budgets; Molecode targets use Molecode-GoT budgets.

	0.5×		1×		2×	
Model	Acc.	kTok	Acc.	kTok	Acc.	kTok
Qwen3-0.6B	10.45	5.89	10.45	10.85	10.45	21.13
Qwen3-1.7B	10.00	6.70	10.00	12.38	10.00	23.74
Qwen3-4B	30.00	5.67	30.00	10.50	30.00	20.08
Qwen3-8B	30.00	6.05	31.09	11.28	31.64	21.65
Qwen3-14B	27.64	6.03	27.64	11.34	27.69	21.45
Llama-3.1-8B	18.18	6.03	18.73	11.30	19.01	20.99
R1-Qwen3-8B	29.09	16.10	30.91	29.59	32.73	56.01
DeepSeek	61.27	6.82	61.27	12.97	62.18	25.35
GPT	78.18	9.72	78.18	18.67	78.00	36.70

Table 23: **MolGoT minus public-check sampling at 2×.** Differences are percentage points with exploratory task-stratified paired 95% intervals. Sampling uses its representation-specific graph reference budget; these comparisons do not claim equal realized cost.

Model	SMILES	Context-enhanced	Molecode
Qwen3-0.6B	−1.36 [-4.55, 1.82]	+0.00 [-3.18, 3.18]	+0.91 [-2.73, 4.55]
Qwen3-1.7B	−0.91 [-5.91, 4.09]	−0.91 [-5.45, 3.64]	+6.36 [1.82, 10.91]
Qwen3-4B	+15.45 [9.09, 21.82]	+14.55 [8.64, 20.45]	+9.09 [3.18, 15.00]
Qwen3-8B	+20.18 [16.36, 24.00]	+16.73 [13.09, 20.36]	+10.73 [6.91, 14.73]
Qwen3-14B	+13.09 [9.27, 16.91]	+11.27 [7.45, 14.91]	+14.03 [10.56, 17.31]
Llama-3.1-8B	+4.36 [0.18, 8.55]	+4.00 [0.36, 7.64]	+5.85 [1.83, 9.69]
R1-Qwen3-8B	+4.55 [-0.45, 9.55]	−7.27 [-12.73, -1.82]	−5.91 [-11.82, 0.00]
DeepSeek	−4.55 [-8.00, -0.91]	−2.73 [-6.36, 0.91]	−7.09 [-10.91, -3.27]
GPT	+2.55 [-0.18, 5.28]	+2.55 [0.18, 5.10]	+0.91 [-2.18, 3.82]

Budget ledger and stopping rules. For example i, let B_i be the default graph's logical token use: input plus completion plus any shared-seed cost. The two SMILES samplers use the MolGoT reference; the Molecode sampler uses Molecode-GoT. For each target rB_i, $r \in \{0.5, 1, 2\}$, $A_i(k)$ is the cumulative input-plus-completion cost through the kth complete sampled answer. Context-enhanced sampling adds its shared draft cost once; original-input sampling does not add that draft. The displayed upper bracket is the first $A_i(k) \geq rB_i$; the saved lower bracket is the last $A_i(k) \leq rB_i$. Sampling ends after reaching the largest target or 128 candidates. Unreachable prefixes remain unavailable, rather than being padded with dummy tokens.

Reasoning-token details are a subset of reported completion usage, not an extra term to add again. Shared seed cost is charged once in full to each logical pipeline answer; it is neither repeated per candidate nor divided among methods. Cached prompts are not subtracted from logical input cost. These conventions describe recorded logical inference expenditure, not billed or hardware-normalized computation.

E.3 Iteration, guidance, and repeated runs

The iterative alternative retains one current answer and repeatedly evaluates and revises it, following the rationale of self-feedback methods (Madaan et al., 2023; Shinn et al., 2023). It targets the corresponding graph condition's recorded input-plus-completion budget before common seed costs; the same seed cost is then charged once to each logical answer. The iterative budget check occurs before a new round after at least two completed rounds, allowing overshoot. Context capacity or a 64-round limit can also stop the process; all 9,238 saved iterative budget records examined here report reaching the budget. The shared-guidance comparison aligns supplementary strategy cards while preserving the native graph and iterative schemas. Turning cards off removes these additional reminders, not the task, chemical tools, or core controller.

Qwen3-8B's graph condition gains 14.55 paired accuracy points with 6.04k rather than 9.02k tokens. R1 and DeepSeek favor iteration with guidance; GPT's interval includes zero. Removing guidance changes DeepSeek's point estimate from -6.73 to $+0.36$ points, with the latter interval crossing zero. These differences show that controller settings affect the comparison; they do not identify a single model-internal cause. Figure 8 separates the process comparison, repeated runs, and guidance effect.

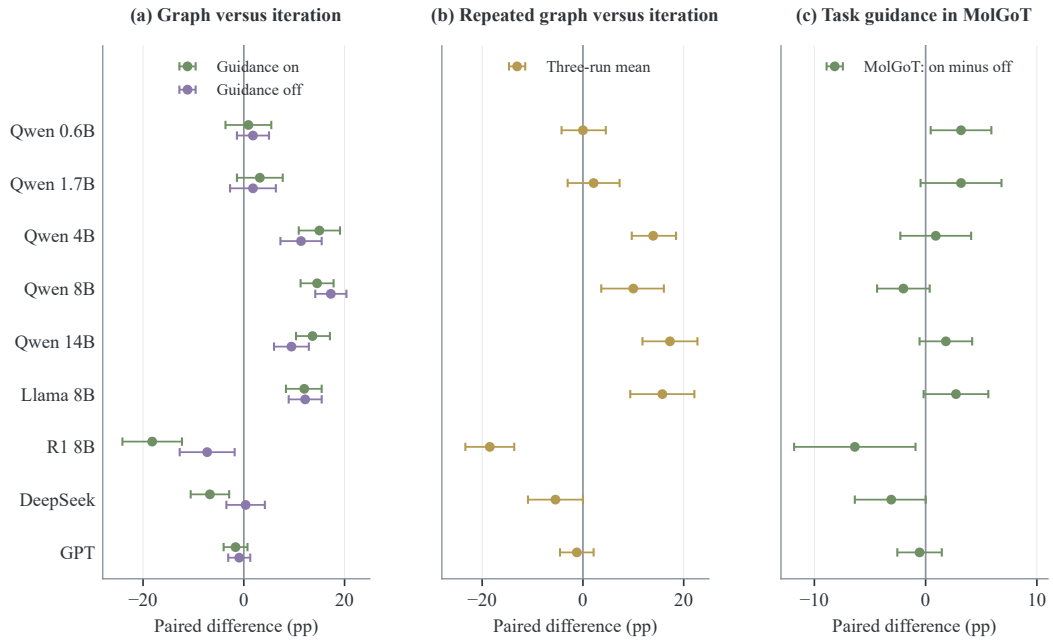

Figure 8: **Process and guidance controls.** (a) Graph minus iteration, with and without supplementary guidance. (b) Graph minus iteration averaged over three run seeds with shared guidance. (c) Guidance on minus off within graph reasoning. Points are paired differences; error bars are exploratory 95% intervals. All nine configurations are shown.

Table 24: **Graph versus iteration with shared guidance.** Marginal accuracies are percentages; paired differences and intervals are percentage points. Differences use common IDs and can differ from subtraction of marginal means.

Model	Graph	Iterative	Δ (pp)	95% interval
Qwen3-0.6B	14.09	13.18	+0.91	[−3.64, 5.45]
Qwen3-1.7B	20.00	16.82	+3.18	[−1.36, 7.73]
Qwen3-4B	40.45	25.45	+15.00	[10.91, 19.09]
Qwen3-8B	40.36	25.82	+14.55	[11.27, 17.82]
Qwen3-14B	44.00	30.36	+13.64	[10.36, 17.09]
Llama-3.1-8B	26.18	14.18	+12.00	[8.36, 15.45]
R1-Qwen3-8B	24.55	42.73	−18.18	[−24.09, −12.27]
DeepSeek V4.1 Flash	54.00	60.73	−6.73	[−10.55, −2.91]
GPT-5.6 Luna	78.36	79.96	−1.64	[−4.01, 0.73]

Table 26 breaks down shared-guidance graph–iteration differences by task scope. This is distinct from the default single-answer CoT baseline in Table 7.

Table 25: **Guidance on/off in graph and iterative reasoning.** Accuracy is in percent. On/off concerns only supplementary strategy cards.

Model	Graph on	Graph off	Iterative on	Iterative off
Qwen3-0.6B	14.09	10.91	13.18	9.09
Qwen3-1.7B	20.00	16.82	16.82	15.00
Qwen3-4B	40.45	39.55	25.45	28.18
Qwen3-8B	40.36	42.36	25.82	25.09
Qwen3-14B	44.00	42.18	30.36	32.73
Llama-3.1-8B	26.18	23.45	14.18	11.27
R1-Qwen3-8B	24.55	30.91	42.73	38.18
DeepSeek V4.1 Flash	54.00	57.09	60.73	56.73
GPT-5.6 Luna	78.36	78.91	79.96	79.78

Table 26: **Shared-guidance graph minus iteration by task scope.** Values are percentage-point differences. No C/ring excludes carbon and ring counting; the remaining columns cover the four task categories.

Model	No C/ring	Understanding	Analysis	Editing	Generation
Qwen3-0.6B	+3.33	−10.00	+5.00	+5.00	+0.00
Qwen3-1.7B	+3.89	−5.00	+8.33	+8.33	−1.67
Qwen3-4B	+1.11	−15.00	+53.33	+6.67	+5.00
Qwen3-8B	+4.67	+3.00	+41.33	+9.33	+0.67
Qwen3-14B	+5.33	−1.00	+35.33	+11.33	+4.00
Llama-3.1-8B	+7.56	+9.00	+36.67	+0.67	+0.67
R1-Qwen3-8B	−20.00	−7.50	−13.33	−35.00	−13.33
DeepSeek V4.1 Flash	−8.00	−22.00	+0.00	+1.33	−11.33
GPT-5.6 Luna	−2.00	−10.00	+0.00	+3.33	−2.67

Table 27: **Shared-guidance graph versus iteration without carbon and ring counting.** Accuracy (%), paired differences (pp), and exploratory 95% intervals use the nine remaining tasks. This iterative comparison is separate from Figure 5c, which uses the default single-answer CoT baseline in Table 7.

Model	Paired n	Graph	Iteration	Δ (pp)	95% interval
Qwen3-0.6B	180	16.67	13.33	+3.33	[-1.11, 8.33]
Qwen3-1.7B	180	22.22	18.33	+3.89	[-0.56, 8.35]
Qwen3-4B	180	31.11	30.00	+1.11	[-2.78, 5.56]
Qwen3-8B	450	32.89	28.22	+4.67	[1.56, 7.78]
Qwen3-14B	450	35.11	29.78	+5.33	[2.00, 8.67]
Llama-3.1-8B	450	23.56	16.00	+7.56	[3.78, 11.12]
R1-Qwen3-8B	180	17.22	37.22	-20.00	[-26.67, -13.89]
DeepSeek	450	44.67	52.67	-8.00	[-12.44, -3.56]
GPT	450	73.56	75.56	-2.00	[-4.89, 0.89]

E.4 Controller variants and task scope

The variants in Figure 5 change one implemented choice while keeping the remaining process active. No-count disables the specialized counting routes; the reduced-adaptation control disables adaptive routing but still enters the specialized carbon/ring branch before general dispatch. Its formula route changes to uniform cross-subgraph integration. No-merge excludes composed candidates from final selection while retaining search, evaluation, and repair. No-repair disables the repair option. These interventions estimate conditional changes within the controller and do not establish that all operations are individually necessary or synergistic.

Chemical-neutral and random-neutral partitions preserve region counts, sizes, atom/bond coverage, and neutral names; chemical annotations are removed from both. The Full condition retains its original labels. Some routes do not use partitions and some molecules have only one region. Output-

cap probes request twice the original node allowance, capped at 8,192 and limited by remaining context. They do not exactly double total inference computation or rerun the iterative comparator at doubled caps.

Table 28: **Component interventions relative to Full.** Full is MolGoT accuracy (%); all changes are intervention minus Full in percentage points. These are the incremental effects underlying Figure 5a.

Model	Full (%)	No count	No merge	No repair	Uniform
Qwen3-0.6B	11.36	−0.45	+0.00	+0.00	−1.36
Qwen3-1.7B	16.36	+0.00	+1.36	+0.45	−0.45
Qwen3-4B	39.09	−11.82	+0.45	+0.00	−4.09
Qwen3-8B	42.36	−11.82	−0.18	−0.36	−6.36
Qwen3-14B	41.64	−10.55	−0.18	+0.00	−0.73
Llama-3.1-8B	24.73	−6.18	−0.18	−0.91	−4.73
R1-Qwen3-8B	26.82	−13.64	+1.82	+6.82	+1.82
DeepSeek V4.1 Flash	55.09	−6.91	−0.55	+1.64	+1.09
GPT-5.6 Luna	78.91	−1.45	+0.36	+0.73	+1.09

Table 29: **Complete and reduced controllers relative to CoT.** CoT is accuracy (%); other entries are paired percentage-point differences from single-answer SMILES-CoT, as in Figure 5a. Token use is not matched.

Model	CoT (%)	Full Δ	No count	Uniform	No merge	No repair
Qwen 0.6B	10.91	+0.45	+0.00	−0.91	+0.45	+0.45
Qwen 1.7B	19.55	−3.18	−3.18	−3.64	−1.82	−2.73
Qwen 4B	20.00	+19.09	+7.27	+15.00	+19.55	+19.09
Qwen 8B	21.64	+20.73	+8.91	+14.36	+20.55	+20.36
Qwen 14B	26.36	+15.27	+4.73	+14.55	+15.09	+15.27
Llama-3.1-8B	15.64	+9.09	+2.91	+4.36	+8.91	+8.18
R1-Qwen3-8B	15.45	+11.36	−2.27	+13.18	+13.18	+18.18
DeepSeek V4.1 Flash	53.64	+1.45	−5.45	+2.55	+0.91	+3.09
GPT-5.6 Luna	73.09	+5.82	+4.36	+6.91	+6.18	+6.55

Table 30: **Matched neutral-label partitions.** Accuracy is in percent; paired chemical-minus-random differences have exploratory 95% intervals. Intervals including zero do not establish equivalence.

Model	Chemical	Random	Δ and 95% interval
Qwen3-0.6B	10.45	10.91	−0.45 [−2.27, 1.36]
Qwen3-1.7B	17.73	13.64	+4.09 [0.45, 7.73]
Qwen3-4B	37.73	34.09	+3.64 [−0.45, 7.73]
Qwen3-8B	40.00	41.09	−1.09 [−3.82, 1.64]
Qwen3-14B	41.09	38.00	+3.09 [0.73, 5.64]
Llama-3.1-8B	25.27	25.27	+0.00 [−3.09, 2.91]
R1-Qwen3-8B	28.18	26.36	+1.82 [−3.18, 6.82]
DeepSeek V4.1 Flash	56.36	54.91	+1.45 [−1.64, 4.55]
GPT-5.6 Luna	79.45	79.82	−0.36 [−2.55, 1.82]

Table 31: **Output-cap and repeated-process probes.** Output columns compare doubled/default requested caps; the final column compares three-run graph/iteration averages. Values and exploratory intervals are percentage points.

Model	MolGoT (ours) cap	Molecode-GoT cap	Repeated graph–iteration
Qwen3-0.6B	-0.91 $[-2.73, 0.00]$	$+0.00$ $[-3.64, 3.64]$	$+0.00$ $[-4.24, 4.55]$
Qwen3-1.7B	$+0.91$ $[0.00, 2.73]$	$+0.91$ $[-2.73, 4.55]$	$+2.12$ $[-3.03, 7.27]$
Qwen3-4B	$+0.00$ $[0.00, 0.00]$	$+0.00$ $[-3.67, 3.67]$	$+13.94$ $[9.70, 18.48]$
Qwen3-8B	-0.91 $[-3.64, 1.82]$	$+0.91$ $[-3.64, 5.45]$	$+10.00$ $[3.64, 16.06]$
Qwen3-14B	-2.73 $[-5.45, 0.00]$	-2.73 $[-5.45, 0.00]$	$+17.27$ $[11.82, 22.73]$
Llama-3.1-8B	$+2.73$ $[-0.91, 5.48]$	-0.94 $[-7.55, 6.60]$	$+15.76$ $[9.39, 22.12]$
R1-Qwen3-8B	$+13.64$ $[6.36, 21.82]$	$+10.00$ $[2.73, 17.27]$	-18.48 $[-23.33, -13.64]$
DeepSeek V4.1 Flash	-2.73 $[-9.09, 3.64]$	-1.82 $[-9.09, 5.45]$	-5.45 $[-10.91, 0.00]$
GPT-5.6 Luna	$+0.91$ $[-3.64, 5.45]$	$+4.55$ $[-1.82, 10.91]$	-1.21 $[-4.55, 2.12]$

E.5 Observed composition and final selection

We inspect all 3,960 saved editing/generation traces from the default and shared-guidance graph conditions at seed 2027. These are model–example–condition records, not independent molecules. "Attempt" means at least one compatible pair was replayed; "gate" means at least one composed candidate passed the online gate; "true" means the selected result retained replayable contributions from multiple parents and differed from each parent. Merely entering aggregation does not count as a successful merge. A selected revision may copy a parent.

For the final-choice comparison, we reconstruct the highest-ranked individual candidate within the same trace and read its already frozen correctness label. Of 3,960 records, 3,151 permit a selected-node score match and 807 are identity comparisons after returning the unchanged root. Two DeepSeek shared-guidance generation records lack an unambiguous reconstruction and are excluded from correction/damage counts, but retained in operation counts. "Fix" is incorrect individual to correct final; "harm" is correct individual to incorrect final. This is a descriptive within-trace comparison, not the counterfactual outcome of a separately rerun no-merge policy.

Table 32: **Default: editing traces.** Counts of records with the indicated event. Attempt and gate need not lead to selection; true and revision describe the selected result. Fix/harm use the observable subset described above.

Model	Traces	Attempt	Gate	True	Revision	Fix	Harm
Qwen3-0.6B	60	0	0	0	0	0	0
Qwen3-1.7B	60	6	0	0	0	0	0
Qwen3-4B	60	26	1	0	0	0	0
Qwen3-8B	150	44	1	1	0	0	1
Qwen3-14B	150	43	1	1	0	0	0
Llama-3.1-8B	150	29	0	0	0	0	0
R1-Qwen3-8B	60	6	0	0	0	0	0
DeepSeek	150	48	1	1	0	0	1
GPT	150	81	3	3	3	0	2

Across default editing traces, 0 of 594 incorrect individual candidates become correct, whereas 4 of 396 correct individuals become incorrect after final selection. For default generation, the corresponding figures are 3/829 and 4/161. Shared-guidance editing gives 0/584 and 7/406; shared-guidance generation gives 7/828 and 6/160. These observations do not support uniform accuracy gains from composition. They distinguish the incremental behavior of the final aggregation stage from the full controller's proposal generation, task routing, checking, and candidate selection.

Table 33: **Default: generation traces.** Counts of records with the indicated event. Attempt and gate need not lead to selection; true and revision describe the selected result. Fix/harm use the observable subset described above.

Model	Traces	Attempt	Gate	True	Revision	Fix	Harm
Qwen3-0.6B	60	0	0	0	0	0	0
Qwen3-1.7B	60	11	10	10	1	0	0
Qwen3-4B	60	25	12	12	0	0	0
Qwen3-8B	150	94	50	50	1	1	1
Qwen3-14B	150	101	66	66	1	0	0
Llama-3.1-8B	150	70	47	47	0	1	0
R1-Qwen3-8B	60	2	2	2	0	0	0
DeepSeek	150	85	53	49	1	1	3
GPT	150	66	59	59	0	0	0

Table 34: **Shared guidance: editing traces.** Counts of records with the indicated event. Attempt and gate need not lead to selection; true and revision describe the selected result. Fix/harm use the observable subset described above.

Model	Traces	Attempt	Gate	True	Revision	Fix	Harm
Qwen3-0.6B	60	0	0	0	0	0	0
Qwen3-1.7B	60	20	0	0	0	0	0
Qwen3-4B	60	24	1	1	0	0	1
Qwen3-8B	150	45	4	4	0	0	2
Qwen3-14B	150	43	2	1	0	0	1
Llama-3.1-8B	150	36	3	2	1	0	0
R1-Qwen3-8B	60	1	0	0	0	0	0
DeepSeek	150	44	1	1	0	0	1
GPT	150	80	3	3	8	0	2

Table 35: **Shared guidance: generation traces.** Counts of records with the indicated event. Attempt and gate need not lead to selection; true and revision describe the selected result. Fix/harm use the observable subset described above.

Model	Traces	Attempt	Gate	True	Revision	Fix	Harm
Qwen3-0.6B	60	0	0	0	0	0	0
Qwen3-1.7B	60	25	15	15	2	0	0
Qwen3-4B	60	26	13	13	0	1	0
Qwen3-8B	150	70	34	33	0	2	1
Qwen3-14B	150	77	38	39	2	1	0
Llama-3.1-8B	150	57	41	41	0	2	1
R1-Qwen3-8B	60	3	0	0	0	0	0
DeepSeek	150	70	35	34	1	0	4
GPT	150	77	72	70	0	1	0

F RECORDED MOLECULAR TRAJECTORIES

The two cases below show the complete molecular-candidate paths of a successful and an unsuccessful editing run. Every displayed molecule and online score comes from the saved GPT trace; drawing coordinates and shading are added for readability. The records expose model proposals and executed states, not hidden reasoning. Final checks are applied after selection and were not supplied as search feedback.

F.1 SUCCESS: SELECTING A CORRECT ADDITION AFTER REJECTED COMPOSITIONS

Example `tomg_atom_addition::67635` requests one additional amide. The model proposes attachments at atoms 10, 11, and 9. All three are valid single-amide candidates. The controller evaluates every pairwise composition, but each has a lower online score than P1; all three are rejected. Revision uses P1 and P2, returns P1's molecule, and is selected with score 0.9353. Final evaluation confirms an increase of exactly one amide and a valid edit witness. This is successful selection and revision, not a successful multi-parent merge.

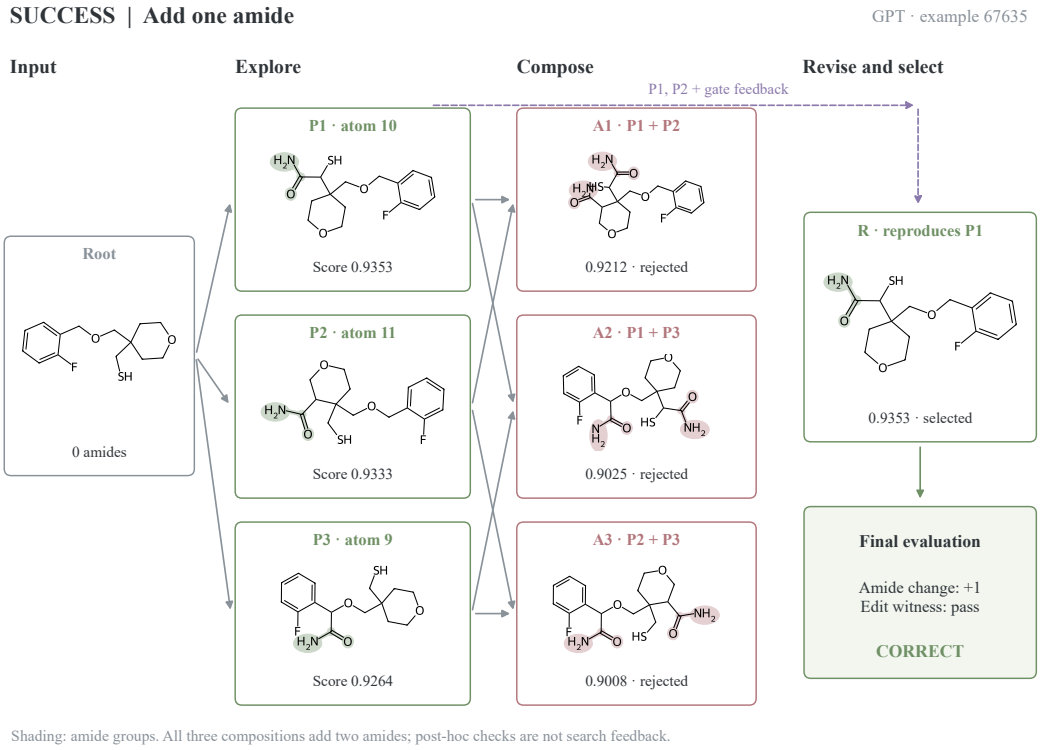

Figure 9: **Complete recorded success trajectory.** All three proposals and three pairwise compositions are shown. Gray arrows record expansion/composition dependencies; the dashed route supplies the selected parents and gate feedback to revision. Amide groups are shaded. The final correctness check is post hoc.

F.2 FAILURE: CORRECT COUNT CHANGE WITHOUT AN ACCEPTED EDIT WITNESS

Example `tomg_atom_deletion::228566` begins with two amides and requests deletion of one. P1 deletes atoms 5 and 19, producing two disconnected components. P2 deletes atoms 3, 4, and 18, giving the higher online score, 0.9500. Composition and revision both yield the same lower-scoring structure, 0.9092, so the controller selects P2. The selected molecule contains one amide, but also retains a bracketed `[CH]` radical at the edited site. Final evaluation records the correct count decrease and no conservative graph-edit witness; the answer is incorrect under the task criterion.

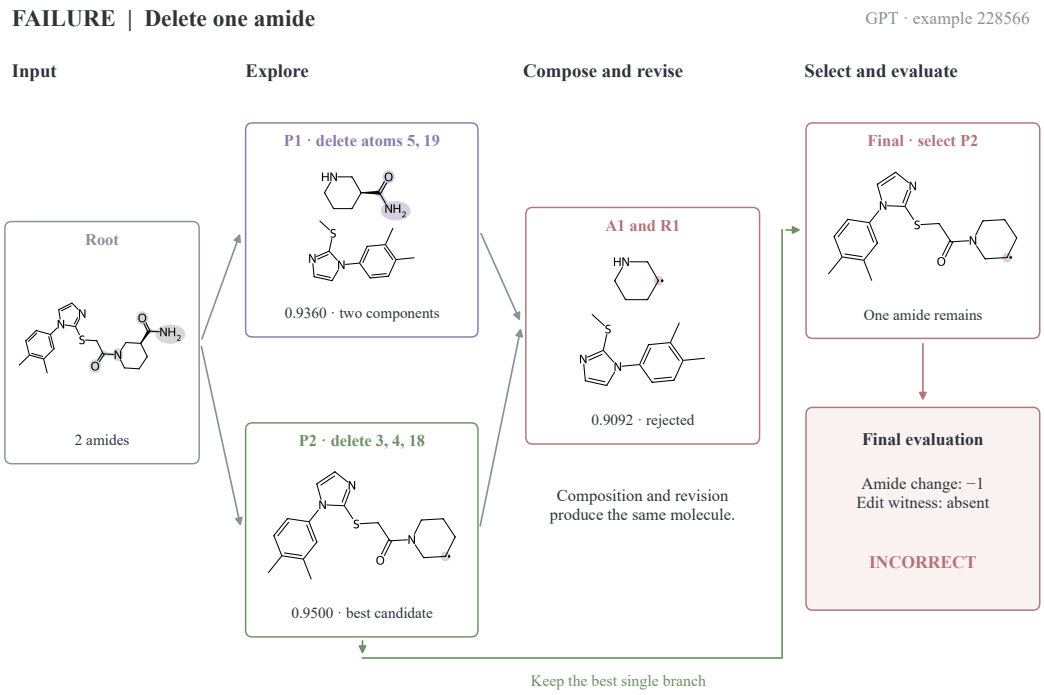

Figure 10: **Complete recorded failure trajectory.** Both proposals and the attempted composition/revision are shown; A1 and R1 have the same molecule. The controller returns the highest-scoring single branch. The highlighted radical and final witness result distinguish a chemically parseable output from a correct edit.

What the pair of cases establishes. The graph preserves enough information to distinguish proposal quality, candidate selection, and endpoint failure. Rejecting an inferior composition can retain a correct answer, but a favorable candidate score cannot guarantee final correctness. This motivates closer alignment of local checks with the requested edit semantics. The examples illustrate mechanisms rather than their frequency; aggregate performance remains defined by the full evaluation. This separation between intermediate assessments and final answers also motivates process-level evaluation in chemical and general reasoning (Guo et al., 2026; Zhang et al., 2026; Lightman et al., 2024; Zhang et al., 2025).

Offline controller replay. The companion package executes the original structural controller for both cases using their saved JSON responses, with network access disabled and reference-only fields removed before inference. All 168 recorded comparisons match, including four regenerated prompts, executed patches, online scores, selection decisions, and final endpoints. The replay preserves 251 scientific functions/classes verbatim; only transport and external-path discovery are removed. It was tested with RDKit 2026.03.6 rather than the recorded 2023.09.6. Agreement is established for these two cases only: the fixture validates these controller executions; it does not regenerate model responses or test other task routes.

G CONTROLLER AND COMPARATOR SEMANTICS

The method specializes the proposal, evaluation, aggregation, and revision operations of GoT with persistent molecular states, executable programs, root-atom footprints, and replay-based acceptance. These operations inherit the general graph-of-thought organization; the molecular execution semantics define how candidate changes can be checked and combined. Table 36 makes the strong iterative comparator explicit. It changes the solving procedure as a whole, so its difference from MolGoT does not isolate the causal effect of a single merge edge.

Table 36: **Shared components and changed control semantics.** These are properties of the recorded implementation, not independent tests of each design choice.

Component	Graph–iteration comparison
Task guidance	Both arms receive the same optional task card through a common wrapper. Native node duties and response schemas remain. Removing the card does not remove all task adaptation.
Structural tools	Editing and generation share seed preparation, chemical views, executable states, patch expansion, public scoring, and the candidate rank key.
Search policy	Iteration keeps one incumbent and proposes one continuation per round. It retains public history but does not re-expand discarded branches, maintain an alternative-candidate archive, or aggregate multiple parents. A whole-SMILES fallback can rebase the current state.
Other task routes	Iteration reads SMILES and answer-blind structural context, then checks and revises one textual answer. It replaces the graph arm's specialized counting and formula routes. Feedback checks answer presence, not reference correctness.
Budget and stopping	Both arms charge input, completion, and one shared-seed cost per logical answer. Iteration tests the target before another round after at least two rounds; context or 64 rounds can also stop it. Equal targets do not imply equal attained cost.