MOLT: A SELF-EVOLVING AGENTIC SYSTEM FOR CONSTRAINED MULTI-PROPERTY LEAD OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Lead optimization is inherently iterative, yet current agentic molecular optimizers do not systematically improve from the experience they accumulate. Optimization history is often retained as tool calls or trajectories, leaving the chemical interventions associated with their outcomes difficult to reuse. We introduce MOLT, a self-evolving agentic system for constrained multi-property lead optimization. MOLT consolidates evaluated molecular transitions into executable chemical transformations with outcome evidence; a context-conditioned action policy selects among these transformations, and an LLM controller plans across retained optimization branches. Exploration expands the available chemical knowledge, while optimization decisions provide supervision for updating the action policy. Across five constrained lead-optimization tasks, MOLT achieves the highest success rate on every evaluated task, improving over prior lead-optimization agents on the three shared tasks and over generative baselines on both dual-target tasks.

1 INTRODUCTION

Drug discovery is essential to developing new medicines, but remains a lengthy and costly process (Paul et al., 2010; Waring et al., 2015). Artificial intelligence can accelerate early discovery by guiding computational molecular design and prioritizing compounds for experimental testing (Vamathevan et al., 2019). Within this process, lead optimization refines biologically promising compounds to improve absorption, distribution, metabolism, and safety while preserving target activity and key structural features. This is an iterative, constrained design problem: selected properties must improve without compromising the broader molecular profile, often over successive rounds of structural modification and evaluation (Keserü & Makara, 2009; Jorgensen, 2009).

Computational approaches to molecular optimization include graph search, reinforcement learning, and generative modeling (Jensen, 2019; Zhou et al., 2019; Zhu et al., 2023). More recent large language models (LLMs) generate revised structures from input molecules and instructions specifying the desired property changes (Dey et al., 2025a;b; Liu et al., 2026; Li et al., 2026c). Agentic approaches extend optimization across multiple decisions, using evaluation feedback, retrieved experience, or coordinated search to guide subsequent steps (Krüger et al., 2026; Gao et al., 2026; Zhang et al., 2026a). TRACE applies this formulation to lead optimization under a similarity constraint: it uses an LLM to select molecular optimization tools and retrieves successful tool trajectories from structurally similar historical molecules (Li et al., 2026a).

A central limitation of current agentic optimizers is that accumulating more optimization experience does not by itself make subsequent optimization better. When optimization experience is retained as tool trajectories, it records which generators were called, but not the chemical changes responsible for the observed outcomes (Li et al., 2026a). A sequence that succeeded once may produce different edits when replayed, some satisfying the intended constraints and others failing them. The useful signal instead lies in the evaluated molecular transitions: what changed and how the molecular properties responded. These transitions provide a basis for extracting explicit chemical transformations and consolidating repeated observations into reusable chemical knowledge. Once such knowledge is available, however, two further problems remain. Lead optimization requires coordinated decisions over multiple steps, so useful local actions must be composed into an optimization path under

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.

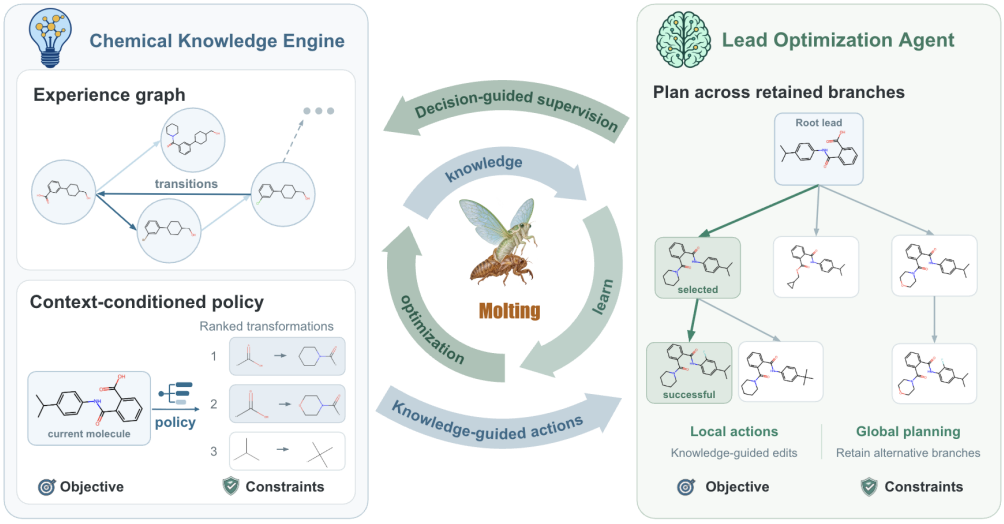


Figure 1: Overview of MOLT as a self-evolving agentic system. Exploratory transitions expand executable chemical knowledge, which a context-conditioned policy turns into knowledge-guided actions for retained-branch planning. Decisions made by agent are fed back as supervision for subsequent policy updates, forming a closed loop between knowledge acquisition and usage.

coupled property and similarity constraints. At the same time, the value of an action depends on molecular context (Papadatos et al., 2010); as the system observes the same or related actions in more contexts, it should become better at deciding which known transformations are useful for a new molecule. A system that learns continually from optimization therefore needs both a mechanism for discovering and consolidating chemical knowledge and a decision process whose action preferences can be updated from experience.

To address these challenges, we introduce MOLT (Molecular Optimization with Learning over Time; Figure 1), a self-evolving agentic system for constrained multi-property lead optimization. MOLT couples an evolving chemical knowledge engine with a knowledge-guided optimization agent. The knowledge engine bootstraps and continually expands the system's executable chemical knowledge from exploratory optimization experience, while a context-conditioned action policy learns which known transformations are useful for the current molecule and task. The optimization agent uses these actions to plan across retained branches under multiple constraints, while its decisions provide supervision for updating the action policy. This closed loop allows exploration to expand what the system knows and optimization to update how that knowledge is used. Across five constrained lead-optimization tasks, MOLT achieves the highest success rate on every evaluated task, improving over prior lead-optimization agents on the three single-target tasks and over generative baselines on both dual-target tasks.

2 Related Work

Molecular optimization has been studied through graph-based search, reinforcement learning, and generative modeling (Jensen, 2019; Zhou et al., 2019; Zhu et al., 2023; Jin et al., 2018). Language-model approaches generate modified molecules from property instructions, including GeLLM^{3}O, GeLLM^{4}O-Cs, DrugR, and RePO (Dey et al., 2025a;b; Liu et al., 2026; Li et al., 2026c). Matched molecular pair (MMP) analysis provides explicit local transformations and associated property changes (Hussain & Rea, 2010; Dalke et al., 2018), while medicinal-chemistry design moves and curated bioisosteric replacements have also been used as reusable molecular edits (Awale et al., 2021; Wirth et al., 2013). MolSearch combines such transformations with Monte Carlo tree search for multi-objective molecular optimization (Sun et al., 2022).

Agentic approaches use iterative feedback and prior experience to guide molecular optimization. SEISMO conditions proposals on optimization trajectories and oracle feedback, F2R retrieves past optimization episodes, ATOM coordinates objectives through tree search, and MolReAct learns sequential reaction selection (Krüger et al., 2026; Gao et al., 2026; Zhang et al., 2026a; Li et al.,

2026b). TRACE formulates lead optimization as tool planning under a structural-similarity constraint and retrieves tool trajectories from structurally similar molecules (Li et al., 2026a). More broadly, experiential agents such as ExpeL and Voyager accumulate reusable knowledge or skills across interactions (Zhao et al., 2024; Wang et al., 2023), and LLM agents have been applied across drug discovery and chemistry through tool-augmented reasoning and agent coordination (M. Bran et al., 2024; Inoue et al., 2025; Averly et al., 2025; Nguyen & Ji, 2026; Zhang et al., 2026b).

3 Background and Task Definition

Lead optimization starts from a promising compound and seeks structurally related analogs with improved molecular properties while preserving features important to the original lead (Bemis & Murcko, 1996; Jorgensen, 2009). Because a structural modification can affect several properties simultaneously, improving one property must be considered together with its effects on the rest of the molecular profile.

Let m_0 denote the initial lead and let $\mathbf{f}(m) = [f_1(m), \ldots, f_P(m)]$ denote the evaluated property profile of molecule m. An optimization task c specifies a set of target properties $\mathcal{T}$, a set of protected properties $\mathcal{G}$, and a structural-similarity threshold τ. For each target property $p \in \mathcal{T}$, the task specifies a desired direction $d_p \in \{+1, -1\}$ and a target threshold θ_p, where $d_p = +1$ denotes improvement by increasing the property and $d_p = -1$ denotes improvement by decreasing it. For each protected property $q \in \mathcal{G}$, the task specifies an acceptable range $\mathcal{I}_q$.

We define property satisfaction as

$$\mathrm{Sat}(m, c) = \mathbb{1}\left[\bigwedge_{p\in\mathcal{T}} d_p\big(f_p(m) - \theta_p\big) \geq 0 \ \wedge \ \bigwedge_{q\in\mathcal{G}} f_q(m) \in \mathcal{I}_q\right]. \tag{1}$$

A molecule is a feasible analog for (m_0, c) if it is chemically valid, satisfies the task constraints, and remains sufficiently similar to the original lead:

$$\mathcal{F}(m_0, c) = \{m \in \mathcal{M}_{\text{valid}} : \mathrm{Sat}(m, c) = 1, \ \mathrm{Sim}(m, m_0) \geq \tau\}.$$

Lead optimization is therefore a constrained multi-step search problem. Starting from m_0, the optimizer seeks a sequence of structural transformations $r_0, \ldots, r_{T-1}$ producing

$$m_0 \xrightarrow{r_0} m_1 \xrightarrow{r_1} \cdots \xrightarrow{r_{T-1}} m_T, \qquad m_T \in \mathcal{F}(m_0, c),$$

where each m_{t+1} is a valid product obtained by applying transformation r_t to m_t. The formulation is intentionally feasibility-based rather than scalarizing the different objectives: a successful analog must satisfy the target, protected-property, validity, and structural-continuity requirements jointly. The concrete property combinations and thresholds used in our experiments are given in Section 6.

4 Motivation and Challenges

Current agentic molecular optimizers accumulate substantial experience during optimization, but turning this experience into reusable knowledge and improved decision making remains challenging. We identify three obstacles: optimization trajectories do not directly reveal reusable chemical transformations, useful transformations must be composed across multiple steps, and their utility depends strongly on molecular and task context.

4.1 Tool Trajectories Do Not Reveal Reusable Chemical Knowledge

Our first observation is that a successful tool trajectory does not reliably identify what should be reused. Figure 2a replays the same successful tool sequence on the same lead and task. The historical run satisfies the constraint with a QED of 0.90, whereas two reruns yield QED values of 0.77 and 0.52, with the latter falling below the same threshold. The trajectory records which models were called, but not the chemical edits associated with the successful outcome. Replaying the same tool sequence therefore does not reveal which chemical change should be reused.

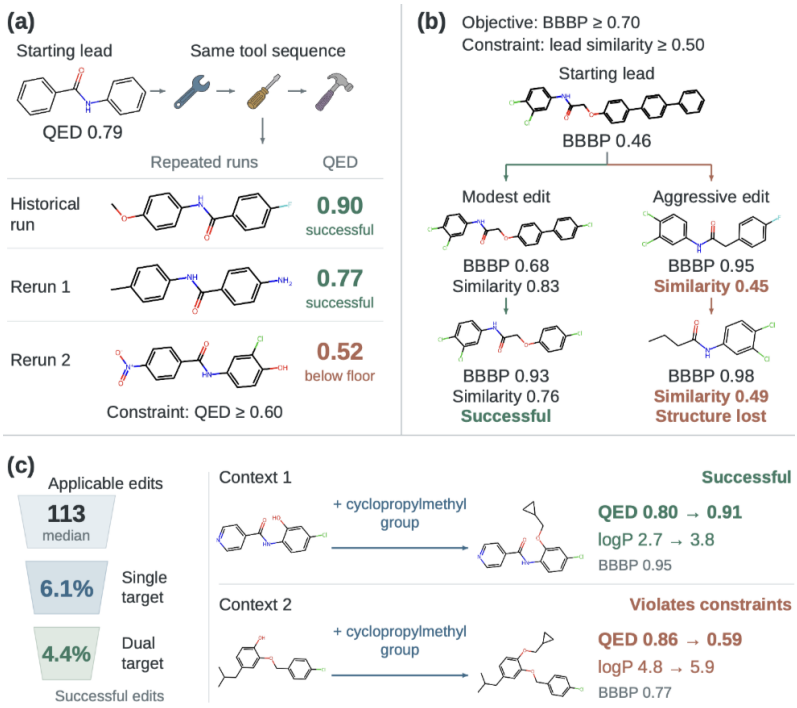


Figure 2: Three challenges in turning optimization experience into a self-evolving lead-optimization system. **(a)** Tool trajectories do not directly reveal reusable chemical knowledge: replaying the same tool sequence can produce different edits and constraint outcomes. **(b)** Chemical knowledge alone does not determine the best optimization path: locally attractive edits can stall or drift, while a more modest early choice may preserve a feasible continuation. **(c)** Useful transformations are sparse and context dependent: many rules may be applicable, but only a small fraction satisfy the full task, and the same edit can help in one context while hurt in another.

The execution history contains a more useful unit of experience: the molecules before and after each step together with their property evaluations. Each transition records a concrete structural change and the associated property response. Organizing these transitions across runs allows recurring edits to be consolidated into explicit transformations with accumulated outcome evidence. This motivates representing optimization experience at the level of evaluated chemical changes, rather than retaining only the tool trajectories that produced them.

4.2 Chemical Knowledge Alone Does Not Determine an Optimization Path

Even with a set of executable chemical transformations, lead optimization cannot be reduced to a sequence of independent local choices. Figure 2b illustrates this tension. An aggressive edit raises BBBP from 0.46 to 0.95 but drops seed-relative similarity below the required threshold. A more modest edit reaches only 0.68 BBBP initially, yet preserves similarity and enables a second step to a feasible analog with BBBP 0.93. An action therefore matters not only for its immediate property change, but also for the optimization paths it leaves available.

The challenge is therefore sequential: the optimizer must decide which partial trajectory to continue as it observes new outcomes. A single-path strategy loses earlier alternatives once it commits to one direction. Retaining multiple branches allows the agent to reconsider where to continue when the current branch stalls or drifts, while judging each choice against the property and structural constraints. Chemical knowledge provides the available actions, but an agentic planner is needed to compose them over a complete optimization episode without discarding useful alternatives too early. Importantly, the agent's decisions during planning provide a second stream of supervision, indicating which transformations are worth pursuing beyond one-step bootstrap labels.

4.3 Useful Transformations Are Sparse and Context Dependent

Multi-step planning still depends on choosing useful transformations at each molecule. Across source molecules, a median of 113 transformation rules are chemically applicable. Yet among queries with at least one successful one-step edit, only 6.1% of applicable edits satisfy the complete task for single-target queries and 4.4% for dual-target queries (Figure 2c). Chemical applicability therefore leaves a large action space in which successful transformations are sparse.

Applicability is not the only issue: the effect of a transformation also depends on molecular context. As illustrated in Figure 2c, the same structural edit can improve the desired profile in one molecule while violating the requirements in another. Useful action selection must therefore be conditioned on the current molecule and task, rather than assigning each transformation a fixed value from its historical frequency or average effect alone.

This motivates a context-conditioned action policy over accumulated chemical knowledge, together with a mechanism for learning from optimization decisions.

5 MOLT: Molecular Optimization with Learning over Time

MOLT implements the closed loop in Figure 1 through a chemical knowledge engine and a knowledge-guided optimization agent. Before optimization begins, exploratory experience bootstraps the executable chemical knowledge and the initial context-conditioned action policy. Exploration then continues in parallel with optimization: exploratory transitions can expand the known transformation space, while multi-step optimization provides decision-guided supervision for refining the action policy. The knowledge engine uses both sources to support subsequent optimization.

5.1 Experience-Driven Chemical Knowledge

Let $\mathcal{D}^{\text{exp}}$ denote the evaluated molecular transitions collected through exploratory optimization. The exploration process follows a fixed randomized policy over the available molecular optimization tools. Starting from training molecules, it repeatedly applies sampled tools (Dey et al., 2025a;b) and records the resulting source–product transitions together with their property evaluations. The same process has two roles: it bootstraps the initial chemical knowledge used by MOLT and continues afterward to explore additional molecules and generator outcomes.

We organize $\mathcal{D}^{\text{exp}}$ as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node represents a canonical molecule with its evaluated property profile, and each edge records an observed source–product transition, its task, and the corresponding property changes. We apply matched molecular pair analysis to the endpoints of each edge to identify the local structural change. These transformations are represented as Substitute, Append, or Remove operations, and chemically equivalent edits are aggregated into a common rule. Each rule therefore associates an explicit chemical transformation with the molecular contexts and property outcomes observed across its applications.

Repeated observations provide evidence for the effect of each transformation. For rule r and objective p, we align the observed property changes with the desired direction of improvement and compute their count $n_{r,p}$, mean $\mu_{r,p}$, and standard deviation $\sigma_{r,p}$. We use an uncertainty-adjusted effect estimate and a support-weighted score:

$$L_{r,p} = \mu_{r,p} - \kappa \frac{\sigma_{r,p}}{\sqrt{n_{r,p}}}, \qquad q_{r,p} = L_{r,p} \log(1 + n_{r,p})(1 + \rho_r), \tag{2}$$

where κ controls the adjustment for variability in the observed effect. A rule is eligible for an objective when it has sufficient support and $L_{r,p} > 0$. Among eligible rules, Equation 2 favors effects supported by more observations while accounting for their variability. The term ρ_r favors underrepresented combinations of operation type, attachment environment, and fragment size.

The resulting executable knowledge includes the eligible transformations and their accumulated multi-property outcome evidence. Continued exploration can introduce previously unseen transformations, while repeated observations strengthen the evidence for transformations already present.

5.2 KNOWLEDGE-GUIDED AGENTIC PLANNING

For a molecule m under task c, the chemical knowledge engine provides a shortlist $\mathcal{R}_K(m, c)$ of executable transformations. Section 5.3 describes how these transformations are selected. When a shortlisted transformation matches multiple sites, we retain all valid, distinct products as separate candidate continuations of the current optimization path.

The optimization agent maintains a search tree $\mathcal{S}$ rooted at the initial lead m_0. Each node v represents a molecule $m(v)$, and each edge records the transformation connecting a parent to its child. Expanding a selected node adds the candidate products generated from its shortlisted transformations as children, while leaves on the other branches remain available.

At iteration t, the frontier $\mathcal{U}_t = \mathrm{Leaf}(\mathcal{S}_t)$ contains the current leaf nodes of the search tree. The LLM controller selects one frontier node to expand based on the initial lead, the optimization task, and the information associated with the current alternatives. Candidate annotations include molecular structure, the incoming transformation, similarity to the original lead, and lightweight auxiliary property estimates. The decision and tree update are

$$v_t = \pi_{\mathrm{LLM}}(m_0, c, \mathcal{U}_t), \qquad v_t \in \mathcal{U}_t,$$
$$\mathcal{S}_{t+1} = \mathrm{Expand}(\mathcal{S}_t, v_t, c).$$

Expanding v_t adds its generated children to the search tree while leaving the other frontier branches available for later selection.

Because the frontier represents all currently available partial optimization paths, the controller can continue along the current direction or return to an earlier branch when subsequent outcomes indicate stalled progress or structural drift. Each selected and evaluated step also records the current molecule, task, applied transformation, resulting molecule, and property outcome. Next, these records form evaluated decision trajectories are used for decision-guided supervision.

5.3 SELF-EVOLVING CONTEXT-CONDITIONED ACTION POLICY

Many transformations may apply to a molecule, but only a small subset is worth evaluating. Let $\mathcal{A}(m)$ denote the deployed transformations applicable to molecule m. We score each transformation using the current molecule, optimization task, and accumulated rule evidence:

$$s_\phi(m, c, r) = g_\phi(z_{\mathrm{mol}}(m), z_{\mathrm{task}}(c), z_{\mathrm{rule}}(r)),$$

where $z_{\mathrm{mol}}(m)$ represents the current molecule using its Morgan fingerprint and property profile, $z_{\mathrm{task}}(c)$ encodes the optimization objectives, thresholds, and protected-property constraints, and $z_{\mathrm{rule}}(r)$ represents the transformation and its accumulated historical outcome evidence. Separate projections are fused by a feedforward network, so the same transformation can receive different scores across molecules and tasks.

Using the task-satisfaction function defined in Eq. 1, the initial policy is trained from one-step labels on exploratory training molecules:

$$y^{\mathrm{step}}(m_0, m, c, r) = \mathbb{1}[\exists\, m' \in \mathrm{Apply}(m, r) : \mathrm{Sat}(m', c) \wedge \mathrm{Sim}(m', m_0) \geq \tau].$$

Here $\mathrm{Apply}(m, r)$ denotes the set of valid, distinct products obtained by applying r to m. A per-query softmax-contrastive objective ranks positive transformations above sampled negatives. At inference, the policy returns the shortlist $\mathcal{R}_K(m, c) = \mathrm{TopK}_{r \in \mathcal{A}(m)}\, s_\phi(m, c, r)$ with $K = 5$ (Appendix A.7). Candidate products and their predicted properties are not inputs to this ranking.

Once optimization begins, exploration and agent decisions provide complementary experience. Continued exploration can discover new transformations and strengthen evidence for known ones, while the agent's decisions during multi-step optimization provide additional supervision for updating the action policy. For an optimization state with current molecule m and task c, let $r^\star$ denote the transformation associated with the continuation selected by the agent. We define

$$y^{\mathrm{dec}}(m, c, r) = \mathbb{1}[r = r^\star],$$

Because these decisions are made during multi-step optimization, they provide supervision beyond the one-step labels used to initialize the policy. This decision-guided supervision is then used to update the context-conditioned policy for subsequent rounds.

6 EXPERIMENTS

6.1 EXPERIMENTAL SETUP

We use MuMOInstruct, a molecular-pair dataset for instruction-guided molecular optimization (Dey et al., 2025a). Exploratory transitions from the training partition bootstrap the experience graph, transformation knowledge, and context-conditioned action policy; the initial deployed set contains 468 transformations. End-to-end evaluation uses queries from the test partition; Appendix A.4 details the separation between the exploratory and evaluation partitions.

Our benchmark contains five constrained lead-optimization tasks. *CNS*, *Oral*, and *Safety* respectively improve blood–brain barrier permeability (BBBP), improve human intestinal absorption (HIA), and reduce predicted Mutagenicity. *CNS+Safety* and *Oral+Safety* optimize two targets jointly. All tasks constrain the remaining molecular profile and require seed-relative Tanimoto similarity of at least 0.50, a permissive structural-continuity constraint that empirically tracks Bemis–Murcko scaffold retention (Appendix A.6). BBBP, HIA, and Mutagenicity are evaluated with ADMET-AI (Swanson et al., 2024); exact task definitions, source-selection criteria, and threshold rationale are provided in Appendices A.1 and A.2.

We compare against general-purpose LLMs, chemistry-specialized LLMs, molecular-optimization models including GeLLM^{3}O and its controllable variants (Dey et al., 2025a;b), and the TRACE lead-optimization agents (Li et al., 2026a). Each query allows up to ten selected outputs. We report **Val** for chemical validity, **Sim** for validity and structural-similarity satisfaction, **Prop** for validity and satisfaction of all property requirements, and **SR** for satisfying all requirements jointly.

6.2 MAIN PERFORMANCE

Table 1: End-to-end performance on five constrained lead-optimization tasks. Val, Sim, Prop, and SR are chemical validity, structural-similarity satisfaction, property satisfaction, and joint success. Best SR per task is shown in bold; dashes denote unreported dual-target results for TRACE.

	CNS				Oral				Safety				CNS+Safety				Oral+Safety			
Method	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**
General-purpose LLMs																				
Llama-3.1-8B	92.8	83.2	25.8	22.0	92.8	72.0	42.2	33.0	94.2	82.8	22.4	17.0	87.9	80.5	5.8	5.3	87.9	80.5	42.4	39.1
Mistral-7B	85.8	81.8	16.4	15.8	75.8	73.0	32.6	31.4	80.6	73.6	18.0	15.8	85.3	77.9	3.7	2.6	59.6	53.5	24.4	22.1
Foundational chemistry LLMs																				
LlaSMol-CodeLlama	98.8	11.0	15.0	3.2	98.6	18.2	19.4	5.0	98.2	17.0	21.6	4.8	98.9	6.8	6.3	0.0	97.7	16.7	19.8	3.1
LlaSMol-Galactica	98.2	1.8	10.0	0.8	98.0	7.0	24.6	1.4	98.8	7.4	15.8	2.4	97.9	1.6	3.7	0.0	98.2	8.2	17.0	1.3
LlaSMol-Llama2	95.0	10.0	17.6	3.0	97.6	25.2	21.8	7.0	91.0	11.8	20.2	2.6	91.1	6.8	4.7	1.1	92.0	14.7	17.5	3.1
LlaSMol-Mistral	99.6	2.4	11.0	0.6	98.8	9.0	17.8	1.4	99.0	6.2	13.0	1.6	100.0	3.2	1.1	0.0	98.5	8.2	21.3	2.1
Lead-optimization LLMs																				
GeLLM^{3}O-P(4)-Llama	96.0	85.6	29.4	26.4	97.2	76.8	38.6	30.2	91.4	79.4	20.0	16.6	86.3	77.4	11.6	7.9	91.5	80.2	40.1	36.2
GeLLM^{3}O-P(4)-Mistral	95.6	91.4	33.4	32.2	88.6	81.6	35.2	33.8	91.8	88.2	29.0	28.2	88.4	85.3	9.5	8.9	77.1	71.5	37.3	35.7
GeLLM^{3}O-P(6)-Llama	97.2	89.0	27.2	24.6	94.8	83.0	30.4	27.6	94.2	79.4	24.6	21.8	92.6	84.7	11.6	9.5	89.7	78.4	37.8	33.7
GeLLM^{3}O-P(6)-Mistral	65.6	57.6	20.0	16.4	69.6	62.0	22.4	20.2	63.2	55.8	18.4	15.6	87.4	72.1	11.6	7.9	77.1	68.1	25.7	22.6
GeLLM^{3}O-C-P(10)-Llama	86.4	80.2	28.0	25.6	90.8	71.4	36.6	30.2	89.2	79.4	26.4	22.8	86.3	74.7	12.1	10.0	86.9	77.9	44.7	41.9
GeLLM^{3}O-C-P(10)-Mistral	94.8	87.2	31.4	29.6	94.4	85.0	38.0	34.8	93.8	86.4	32.0	29.6	94.7	88.4	24.7	24.2	88.9	79.2	38.0	35.2
Lead-optimization agents																				
TRACE-Online	100.0	100.0	42.6	42.6	100.0	100.0	29.0	29.0	100.0	100.0	41.8	41.8				—				
TRACE-Parallel	100.0	100.0	51.2	51.2	100.0	100.0	27.4	27.4	100.0	100.0	44.8	44.8				—				
MOLT (ours)	100.0	95.2	86.6	**81.8**	100.0	97.2	66.4	**63.6**	100.0	97.2	84.2	**81.8**	100.0	95.8	57.9	**53.7**	100.0	95.4	71.0	**67.1**

MOLT achieves the highest SR on all five tasks (Table 1). It improves over both TRACE variants on the three shared single-target tasks and substantially outperforms the generative baselines on both dual-target tasks. The component metrics show why. LlaSMol models are generally valid but often fail the similarity requirement, while TRACE preserves similarity but satisfies the full property requirements less often. MOLT maintains high validity and similarity while substantially increasing property satisfaction, yielding higher joint success.

Beyond the main benchmark, a supplementary single-target evaluation, on which the strongest lead-optimization methods approach ceiling, is reported in Appendix A.3. MOLT also transfers to out-of-distribution leads sampled from ChEMBL (Appendix A.8), and, because it applies transformations with RDKit rather than a neural molecular generator, attains this performance at substantially lower per-query inference cost than the agentic baselines (Appendix A.9).

6.3 System Analysis and Case Studies

Beyond aggregate performance, we examine how MOLT changes as experience accumulates. We first isolate the contribution of agentic planning, then use three complementary case studies to illustrate how the system acquires new chemical knowledge, accumulates objective-specific evidence, and learns from its own optimization decisions.

Agentic planning. Chemical knowledge alone is not sufficient when useful transformations must be composed over multiple steps. Random and greedy local selection achieve SRs of 32.4% and 36.8%, respectively, while the same knowledge-guided agent reaches 58.9% when restricted to a single trajectory (Table 2). Retaining alternative branches further increases SR to 69.6%. It also raises later success, defined as the fraction of solved queries whose first feasible output is found only after the initial selection, from 32.4% to 48.9%. This indicates that branch retention is particularly important when the initial choice is insufficient and a successful solution must be reached through subsequent planning.

Decision strategy	SR (%)	Later success (%)
Random transformation selection	32.4	18.3
Greedy top-1 action	36.8	16.0
MOLT (single trajectory)	58.9	32.4
MOLT (retained branches)	**69.6**	**48.9**

Table 2: Planning ablation across decision strategies. Later success is the fraction of solved queries whose first feasible output is found after the initial selection.

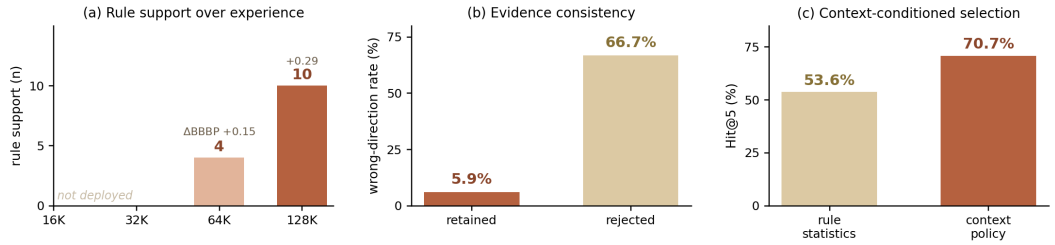


Figure 3: Quantitative support for the case studies. **(a)** Support for the Case 1 transformation across experience snapshots. **(b)** Wrong-direction rates for retained and rejected transformations. **(c)** Hit@5 for rule-statistics ranking and the context-conditioned policy.

Case 1: A newly acquired transformation enables CNS rescue.

Lead: `O=C(O)c1[nH]c2ccccc2c1Sc1ccc(Cl)cc1`
Optimized: `CN(C)C(=O)c1[nH]c2ccccc2c1Sc1ccc(Cl)cc1`

The lead is a 3-(arylthio)indole-2-carboxylic acid with poor predicted blood–brain barrier permeability. At the 16K and 32K accumulated transitions, the deployed knowledge does not yet contain the carboxylic-acid-to-N,N-dimethylamide transformation used in the successful solution. The rule first enters the deployed set at 64K transitions, where applying it yields an analog with predicted BBBP of 0.95. The edit replaces a strongly ionizable carboxylic acid with a neutral tertiary amide while preserving the surrounding indole scaffold, a chemically plausible direction for increasing passive BBB permeability.

The rule is subsequently reinforced as more experience is collected. Its support increases from four observations at 64K to ten at 128K, with all observed effects remaining in the desired BBBP direction (Figure 3a; Appendix A.5). This case shows a concrete change in MOLT's executable knowledge: later experience makes available a useful chemical transformation that was absent from the earlier system and then strengthens the evidence supporting it.

Case 2: Accumulated evidence changes which transformations are trusted for each objective.

Retained for QED: `*[N+](=O)[O-] >> *Cl`
Rejected for QED: `*C(=O)O >> *Cl`

MOLT uses repeated observations to distinguish transformations with consistent effects from those whose outcomes vary across molecular contexts. Across sufficiently observed rule–objective pairs, transformations retained by the evidence criterion have a median wrong-direction rate (the fraction of observations moving the property opposite to the intended direction) of only 5.9%, compared with 66.7% for rejected transformations (Figure 3b). The two QED rules above illustrate this difference. The nitro-to-chloro substitution is retained after all 11 observed applications increase QED, whereas the carboxyl-to-chloro replacement produces mixed QED outcomes and is therefore rejected for this objective.

The filtering is objective-specific rather than a global judgment on the transformation itself. The same transformation can therefore be reliable for one objective but unreliable for another. MOLT uses accumulated outcome evidence to retain transformations selectively for the objectives on which their effects are consistent.

Case 3: Decision supervision captures utility missed by bootstrap labels.

Lead: `O=C(Cc1ccc(O)cc1)c1ccc(O)cc1O`
Step 1: `O=C(Cc1ccc(Br)cc1)c1ccc(O)cc1O` BBBP $= 0.21$ (below the 0.70 target)
Step 2: `O=C(Cc1ccc(Br)cc1)c1ccc(F)cc1O` BBBP $= 0.69$ (still below 0.70)
Step 3: `O=C(Cc1ccc(C(F)(F)F)cc1)c1ccc(F)cc1O` BBBP $= 0.81$ (successful)

Exploratory experience first bootstraps a useful context-conditioned policy: a task-satisfying one-step transformation appears in its top five for 70.7% of eligible queries, compared with 53.6% under rule-statistics ranking (Figure 3c). Multi-step optimization then provides a different signal that one-step labels cannot capture.

In this CNS trajectory, the first two selected edits do not yet satisfy the BBBP threshold. The first replaces one phenolic hydroxyl with bromine, and the second replaces another with fluorine, progressively reducing hydrogen-bond donation and polarity. BBBP rises from 0.21 to 0.69 but remains below the required 0.70. A third edit replaces the bromine with a trifluoromethyl group, yielding a feasible analog with BBBP 0.81.

Under the one-step bootstrap criterion, the first transformation receives a negative label because its immediate outcome does not satisfy the task. The bootstrap policy therefore learns to rank such a continuation poorly. During multi-step optimization, however, the planner still selects this action and eventually reaches a feasible analog through subsequent edits. This selected action is recorded as positive decision supervision and used to update the policy. After this self-evolving update, the policy can recognize the same type of transformation as a useful stepping stone in similar contexts, even though its immediate outcome remains unfavorable. In other words, decision supervision teaches MOLT a behavior that is absent from the bootstrap policy: an action can be worth taking because it leads toward a better downstream state, rather than because it succeeds immediately.

Takeaway. The three cases expose complementary parts of MOLT's self-evolving loop. Continued exploration expands the executable transformation knowledge and strengthens its outcome evidence, while the agent's own optimization decisions provide new supervision for updating how that knowledge is used. MOLT therefore evolves both its executable chemical knowledge and its context-conditioned action policy as experience accumulates.

7 Conclusion

We introduced MOLT, a self-evolving agentic system for constrained multi-property lead optimization. MOLT converts exploratory molecular transitions into executable chemical knowledge, uses a context-conditioned policy and retained-branch planning to optimize leads, and feeds its own optimization decisions back as supervision for subsequent policy updates. Across five constrained tasks, MOLT substantially improves success while maintaining structural continuity to the original lead. Together with the case studies, these results show how accumulated experience can change both the chemical actions available to an optimizer and how those actions are selected in later rounds.

GENERATIVE AI DISCLOSURE

We used general-purpose language models and coding assistants to support manuscript writing and code development. They were used to improve clarity and presentation of the text, and to assist with implementation and experiment automation. All code, experiments, reported results, and manuscript content were reviewed and verified by the authors, who take full responsibility for the work.

REFERENCES

Reza Averly, Frazier N Baker, Ian A Watson, and Xia Ning. Liddia: Language-based intelligent drug discovery agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 12015–12039, 2025.

Mahendra Awale, Jérôme Hert, Laura Guasch, Sereina Riniker, and Christian Kramer. The playbooks of medicinal chemistry design moves. *Journal of Chemical Information and Modeling*, 61 (2):729–742, 2021.

Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996.

G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

Andrew Dalke, Jerome Hert, and Christian Kramer. mmpdb: An open-source matched molecular pair platform for large multiproperty data sets. *Journal of chemical information and modeling*, 58 (5):902–910, 2018.

Vishal Dey, Xiao Hu, and Xia Ning. Gellm^{3}o: Generalizing large language models for multi-property molecule optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25192–25221, 2025a.

Vishal Dey, Xiao Hu, and Xia Ning. Large language models for controllable multi-property multi-objective molecule optimization. Association for Computational Linguistics, 2025b.

Wenhan Gao, Xiran Fan, Chin-Chia Michael Yeh, Jiarui Sun, Yuzhong Chen, Menghai Pan, Mahashweta Das, and Yi Liu. Feedback to reasoning: Llm-assisted molecular optimization with domain feedback and historical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 12735–12754, 2026.

Jameed Hussain and Ceara Rea. Computationally efficient algorithm to identify matched molecular pairs (mmps) in large data sets. *Journal of chemical information and modeling*, 50(3):339–348, 2010.

Yoshitaka Inoue, Tianci Song, Xinling Wang, Augustin Luna, and Tianfan Fu. Drugagent: Multi-agent large language model-based reasoning for drug-target interaction prediction. *ArXiv*, pp. arXiv–2408, 2025.

Jan H Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

William L Jorgensen. Efficient drug lead discovery and optimization. *Accounts of chemical research*, 42(6):724, 2009.

György M Keserü and Gergely M Makara. The influence of lead discovery strategies on the properties of drug candidates. *nature reviews Drug Discovery*, 8(3):203–212, 2009.

Fabian P Krüger, Andrea Hunklinger, Adrian Wolny, Tim J Adler, Igor Tetko, and Santiago David Villalba. Seismo: Increasing sample efficiency in molecular optimization with a trajectory-aware llm agent. *arXiv preprint arXiv:2602.00663*, 2026.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *International conference on machine learning*, pp. 1945–1954. PMLR, 2017.

Greg Landrum. Rdkit: Open-source cheminformatics software. 2016. URL https://github.com/rdkit/rdkit/releases/tag/Release_2016_09_4.

Lingxiao Li, Haobo Zhang, Ruohao Fan, Bin Chen, and Jiayu Zhou. Molecular lead optimization via agentic tool planning. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 11328–11338, 2026a.

Tao Li, Kaiyuan Hou, Tuan Vinh, Monika Raj, Zhichun Guo, and Carl Yang. Reinforcement learning with llm-guided action spaces for synthesizable lead optimization. *arXiv preprint arXiv:2604.07669*, 2026b.

Xuan Li, Zongze Li, Jiangchao Yao, Yu Rong, Lu Zhang, Bo Han, et al. Reference-guided policy optimization for molecular optimization via llm reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 65266–65297, 2026c.

Christopher A Lipinski, Franco Lombardo, Beryl W Dominy, and Paul J Feeney. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced drug delivery reviews*, 46(1-3):3–26, 2001.

Haoran Liu, Zheni Zeng, Yukun Yan, Yuxuan Chen, and Yunduo Xiao. Drugr: Optimizing molecular drugs through llm-based explicit reasoning. *arXiv preprint arXiv:2602.08213*, 2026.

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.

Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of chemical information and modeling*, 52(6):1686–1697, 2012.

Kristien Mortelmans and Errol Zeiger. The ames salmonella/microsome mutagenicity assay. *Mutation research/fundamental and molecular mechanisms of mutagenesis*, 455(1-2):29–60, 2000.

Thao Nguyen and Heng Ji. Mollingo: Molecule-native representations for llm-powered scientific agents. *arXiv preprint arXiv:2605.27853*, 2026.

George Papadatos, Muhammad Alkarouri, Valerie J Gillet, Peter Willett, Visakan Kadirkamanathan, Christopher N Luscombe, Gianpaolo Bravi, Nicola J Richmond, Stephen D Pickett, Jameed Hussain, et al. Lead optimization using matched molecular pairs: inclusion of contextual information for enhanced prediction of herg inhibition, solubility, and lipophilicity. *Journal of chemical information and modeling*, 50(10):1872–1886, 2010.

Steven M Paul, Daniel S Mytelka, Christopher T Dunwiddie, Charles C Persinger, Bernard H Munos, Stacy R Lindborg, and Aaron L Schacht. How to improve r&d productivity: the pharmaceutical industry's grand challenge. *Nature reviews Drug discovery*, 9(3):203–214, 2010.

Mengying Sun, Jing Xing, Han Meng, Huijun Wang, Bin Chen, and Jiayu Zhou. Molsearch: search-based multi-objective molecular generation and property optimization. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 4724–4732, 2022.

Kyle Swanson, Parker Walther, Jeremy Leitz, Souhrid Mukherjee, Joseph C Wu, Rabindra V Shivnaraine, and James Zou. Admet-ai: a machine learning admet platform for evaluation of large-scale chemical libraries. *Bioinformatics*, 40(7):btae416, 2024.

Roberto Todeschini and Viviana Consonni. *Handbook of molecular descriptors*. John Wiley & Sons, 2008.

Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, et al. Applications of machine learning in drug discovery and development. *Nature reviews Drug discovery*, 18(6):463–477, 2019.

Daniel F Veber, Stephen R Johnson, Hung-Yuan Cheng, Brian R Smith, Keith W Ward, and Kenneth D Kopple. Molecular properties that influence the oral bioavailability of drug candidates. *Journal of medicinal chemistry*, 45(12):2615–2623, 2002.

Travis T Wager, Xinjun Hou, Patrick R Verhoest, and Anabella Villalobos. Moving beyond rules: the development of a central nervous system multiparameter optimization (cns mpo) approach to enable alignment of druglike properties. *ACS chemical neuroscience*, 1(6):435, 2010.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Michael J Waring, John Arrowsmith, Andrew R Leach, Paul D Leeson, Sam Mandrell, Robert M Owen, Garry Pairaudeau, William D Pennie, Stephen D Pickett, Jibo Wang, et al. An analysis of the attrition of drug candidates from four major pharmaceutical companies. *Nature reviews Drug discovery*, 14(7):475–486, 2015.

Matthias Wirth, Vincent Zoete, Olivier Michielin, and Wolfgang HB Sauer. Swissbioisostere: a database of molecular replacements for ligand design. *Nucleic acids research*, 41(D1):D1137–D1143, 2013.

Jia Zhang, Tengfei Ma, Tianle Li, Daojian Zeng, Xieping Gao, and Xiangxiang Zeng. Agents on a tree: Pathwise coordination for multi-objective molecular optimization. *arXiv preprint arXiv:2606.00008*, 2026a.

Lisheng Zhang, Lilong Wang, Xiangyu Sun, Wei Tang, Haoyang Su, Yuehui Qian, Qikui Yang, Qingsong Li, Zhenyu Tang, Haoran Sun, et al. Molclaw: An autonomous agent with hierarchical skills for drug molecule evaluation, screening, and optimization. *bioRxiv*, pp. 2026–04, 2026b.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752, 2019.

Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Tingjun Hou, Jian Wu, et al. Sample-efficient multi-objective molecular optimization with gflownets. *Advances in neural information processing systems*, 36:79667–79684, 2023.

A Additional Experimental Details

A.1 Constrained ADMET-Rescue Tasks

Our main benchmark models lead optimization as the rescue of one or two ADMET liabilities while preserving the remaining molecular profile. Each task specifies target properties to improve and one-sided guardrails for properties that should not regress. A candidate may freely improve a guardrailed property, but it must remain within the corresponding acceptable range. All tasks additionally require chemical validity and seed-relative similarity $\mathrm{Sim}(m, m_0) \geq 0.50$.

BBBP, HIA, and Mutagenicity are evaluated with ADMET-AI (Swanson et al., 2024). QED, penalized logP (pLogP), molecular weight (MW), topological polar surface area (TPSA), and hydrogen-bond donors (HBD) are computed with RDKit (Landrum, 2016). Table 3 gives the exact constraints used in evaluation.

A.2 Task Construction and Threshold Rationale

We construct the benchmark so that each query isolates the intended optimization problem. Before optimization, a source must exhibit the target liability at the predictor decision boundary ($\mathrm{BBBP} < 0.5$, $\mathrm{HIA} < 0.5$, or $\mathrm{Mutagenicity} > 0.5$) while already satisfying every non-target guardrail. We additionally require $200 \leq \mathrm{MW} \leq 450$ for CNS-oriented tasks and $200 \leq \mathrm{MW} \leq 500$ otherwise.

Table 3: Operational definitions of the five constrained lead-optimization tasks. Targets specify the required property change; guardrails bound only regression. All tasks additionally require chemical validity and $\mathrm{Sim}(m, m_0) \geq 0.50$.

Task	Source profile	Target(s)	Guardrails
CNS	low-BBBP CNS lead	BBBP $\uparrow \geq 0.7$	QED≥ 0.6, pLogP$\in [0, 5]$, MW≤ 450, HBD≤ 3, HIA≥ 0.5, Mut.≤ 0.3
Oral	low-HIA oral lead	HIA $\uparrow \geq 0.5$	QED≥ 0.6, pLogP$\in [0, 5]$, MW≤ 500, TPSA≤ 140, Mut.≤ 0.3
Safety	mutagenic lead	Mut. $\downarrow \leq 0.3$	QED≥ 0.6, pLogP$\in [-1, 5]$, MW≤ 500
CNS+Safety	mutagenic CNS lead	BBBP $\uparrow \geq 0.7$, Mut. $\downarrow \leq 0.3$	QED≥ 0.6, pLogP$\in [0, 5]$, MW≤ 450, HBD≤ 3, HIA≥ 0.5
Oral+Safety	mutagenic oral lead	HIA $\uparrow \geq 0.5$, Mut. $\downarrow \leq 0.3$	QED≥ 0.6, pLogP$\in [-3, 5]$, MW≤ 500, TPSA≤ 140

Thus, a nominal BBBP rescue, for example, does not begin from a molecule that simultaneously requires repair of QED or another protected property.

The rescue thresholds are deliberately separated from the source-liability boundary. BBBP sources begin below 0.5 but must reach at least 0.7, while mutagenic sources begin above 0.5 but must reach at most 0.3. Success therefore requires a non-trivial change rather than merely crossing the predictor decision boundary. Dual-target tasks require two liabilities simultaneously and consequently yield smaller eligible source pools: 190 CNS+Safety and 390 Oral+Safety sources, compared with 500 for each single-target task.

The remaining thresholds are absolute, literature-motivated ranges rather than source-relative tolerances. This makes success correspond to reaching an acceptable molecular profile instead of simply avoiding degradation from the starting molecule. Table 4 summarizes the rationale.

Table 4: Rationale for the property thresholds used in the constrained rescue benchmark. CNS-oriented constraints follow CNS multi-parameter design criteria, while oral constraints follow commonly used drug-likeness and oral-bioavailability ranges.

Constraint	Value	Rationale
QED	≥ 0.6	Lower bound of the high-QED range observed among approved drugs (Bickerton et al., 2012)
MW (CNS)	≤ 450	CNS multi-parameter optimization favors lower molecular weight (Wager et al., 2010)
HBD (CNS)	≤ 3	CNS multi-parameter optimization criterion (Wager et al., 2010)
MW (oral)	≤ 500	Rule-of-Five molecular-weight ceiling (Lipinski et al., 2001)
pLogP	≤ 5	Rule-of-Five lipophilicity ceiling (Lipinski et al., 2001)
TPSA (oral)	≤ 140	Common oral-bioavailability criterion (Veber et al., 2002)
BBBP	≥ 0.7	Requires a confident shift into the BBB-permeable region rather than merely crossing the 0.5 predictor boundary (Martins et al., 2012)
HIA	≥ 0.5	ADMET-AI decision threshold for intestinal absorption (Swanson et al., 2024)
Mutagenicity	≤ 0.3	Places successful candidates well below the predictor decision boundary (Mortelmans & Zeiger, 2000)
Similarity	≥ 0.50	Morgan-fingerprint Tanimoto constraint used as a permissive proxy for structural continuity to the original lead (Todeschini & Consonni, 2008)

Two task-specific choices are worth noting. For Oral+Safety, the pLogP lower bound is widened to -3 because the joint low-HIA/high-mutagenicity source population is substantially more polar; using the $[0, 5]$ range would remove nearly all otherwise eligible sources. Here pLogP denotes the synthetic-accessibility-penalized logP objective used by the molecular generators (Kusner et al., 2017), rather than an experimental logP assay.

Table 5: Supplementary single-target optimization performance under the same structural-similarity constraint. Val, Sim, Prop, and SR follow the definitions in Section 6.1. Best SR per task is shown in bold.

	BBBP				HIA				Mut.				QED				pLogP			
Method	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**	Val	Sim	Prop	**SR**
General-purpose LLMs																				
Llama-3.1-8B	57.0	44.8	28.8	18.4	57.6	35.2	43.4	22.2	67.6	41.0	37.0	12.4	56.2	36.0	24.6	9.8	64.8	44.6	31.8	19.0
Mistral-7B	59.2	53.2	31.8	26.2	62.0	49.2	45.2	33.6	57.6	43.0	38.8	27.0	55.2	44.8	20.4	13.8	59.6	46.6	26.0	19.6
Foundational chemistry LLMs																				
LlaSMol-CodeLlama	98.0	16.4	86.8	9.8	97.0	56.6	48.8	20.4	98.2	20.4	73.4	11.2	98.2	16.4	71.4	8.4	98.6	38.6	35.6	10.6
LlaSMol-Galactica	98.6	4.8	87.0	4.0	95.2	43.2	52.2	20.4	98.8	9.8	75.6	6.2	97.6	7.6	54.8	1.4	98.8	26.6	22.2	6.2
LlaSMol-Llama2	59.4	9.2	52.2	5.4	69.2	41.6	29.8	12.4	41.4	6.6	33.4	3.2	67.2	11.2	47.6	5.6	63.8	22.2	17.0	6.4
LlaSMol-Mistral	98.2	6.4	87.2	2.8	98.2	47.6	43.4	15.4	99.4	11.0	70.8	5.4	99.2	6.0	67.4	0.8	99.0	33.6	14.0	6.0
Lead-optimization LLMs																				
GeLLM^{3}O-P(4)-Llama	87.6	66.8	77.6	58.6	82.8	42.4	74.2	39.0	88.0	63.2	63.8	44.2	87.0	67.2	75.8	58.2	86.2	56.8	65.6	45.6
GeLLM^{3}O-P(4)-Mistral	92.6	80.4	83.4	72.0	83.4	64.6	75.2	57.6	91.4	78.2	71.0	60.6	92.4	82.6	80.8	72.2	91.4	79.8	81.6	71.6
GeLLM^{3}O-P(6)-Llama	90.2	71.0	79.4	61.6	85.0	46.0	77.8	42.6	89.8	65.0	75.8	53.6	92.0	71.6	79.2	62.2	91.0	59.8	69.0	49.6
GeLLM^{3}O-P(6)-Mistral	92.0	82.0	82.4	73.2	86.0	70.0	79.6	66.0	93.8	79.8	82.2	70.4	93.0	81.2	80.4	69.8	90.4	73.2	76.4	65.0
Lead-optimization agents																				
TRACE-Retrieve	100.0	100.0	97.2	97.2	100.0	100.0	89.2	89.2	100.0	100.0	95.6	95.6	100.0	100.0	97.6	97.6	100.0	100.0	93.4	93.4
TRACE-Parallel	100.0	100.0	98.2	98.2	100.0	100.0	93.8	93.8	100.0	100.0	98.6	98.6	100.0	100.0	99.8	**99.8**	100.0	100.0	99.2	99.2
MOLT (ours)	100.0	100.0	99.2	**99.2**	100.0	100.0	97.6	**97.6**	100.0	100.0	99.2	**99.2**	100.0	100.0	99.6	99.6	100.0	100.0	99.2	**99.2**

A.3 Supplementary Single-Target Benchmark

We additionally evaluate the simpler single-target setting used in prior work. The benchmark contains $2{,}500$ queries, with 500 each for BBBP, HIA, Mutagenicity, QED, and pLogP. BBBP, HIA, QED, and pLogP are increased, while Mutagenicity is decreased. Success requires a valid molecule that improves the target in the required direction while satisfying $\mathrm{Sim}(m, m_0) \geq 0.50$.

This benchmark is constructed from the MuMOInstruct training partition and is included as a diagnostic comparison rather than as the main generalization benchmark. Because it contains only one optimization objective and no competing multi-property guardrails, the strongest lead-optimization methods approach ceiling performance.

The agentic methods are all close to ceiling in this setting, whereas the constrained multi-property benchmark in Table 1 produces much larger differences. We therefore use the latter as the primary evaluation in the main text.

A.4 Train/Test Separation

The main benchmark uses queries from the MuMOInstruct test partition, while transformation mining and action-policy training use the training partition. We prevent benchmark structure from influencing the construction of the learned policy: benchmark source molecules are excluded from the policy-training pool by canonical SMILES, leaving $7{,}129$ training queries over $6{,}079$ unique source molecules with zero exact overlap with the $2{,}063$ benchmark sources.

The transformation knowledge is also built only from training-partition transitions. It stores abstract SMIRKS transformations and their accumulated property effects rather than source molecules or benchmark products; test products are generated only at inference time.

MuMOInstruct itself is split at the instruction-instance level, so 21 of the $2{,}063$ benchmark source SMILES also appear somewhere in its raw training partition. These molecules are excluded from the action-policy training pool described above. We therefore treat the benchmark as a test-partition evaluation, but do not claim scaffold-disjointness between the underlying train and test distributions.

Finally, the same ADMET predictor is used to define the source liability and to evaluate whether the corresponding rescue target is satisfied. This shared oracle is part of the task definition rather than a train/test overlap, but the benchmark should be interpreted as predictor-based optimization rather than experimental validation.

A.5 Growth of Executable Transformation Knowledge

As exploratory experience grows, chemically equivalent edits are repeatedly merged into shared transformation rules. Table 6 shows four snapshots of this process. The number of observed transi-

tions grows eight-fold from 16K to 128K, while the deployed transformation set grows from 243 to 468 rules and accumulated support for recurring transformations increases.

Table 6: Growth of executable transformation knowledge with additional exploratory experience. Distinct MMP rules are canonicalized structural edits; high-confidence rules satisfy the support and effect-consistency criterion used in Section 5.1.

Observed transitions	16K	32K	64K	128K
Distinct MMP rules	7,454	13,143	23,132	40,336
High-confidence rules	97	186	368	686
High-confidence fraction	1.30%	1.42%	1.59%	1.70%
Deployed transformations	243	293	365	468
SUB/APP/REM (%)	87.5/5.4/7.0	89.1/4.6/6.3	90.4/3.9/5.7	91.3/3.6/5.2

Beyond growth, the deployed set at the final snapshot is well supported and multi-property: each transformation is backed by a median of 11 observations and records outcomes for 4.15 of the five properties on average, with 51% covering all five (Figure 4).

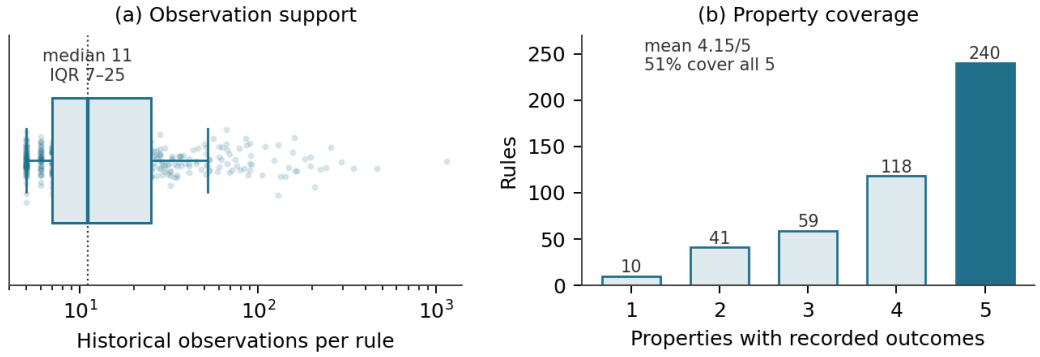


Figure 4: Deployed transformation set at the final experience snapshot (128K transitions). **(a)** Historical observation support per transformation. **(b)** Number of properties with recorded outcomes.

A.6 TANIMOTO SIMILARITY AND SCAFFOLD RETENTION

The main benchmark uses Morgan-fingerprint Tanimoto similarity $\mathrm{Sim}(m, m_0) \geq 0.50$ as a permissive structural-continuity constraint. This threshold does not mathematically guarantee scaffold identity, so we additionally evaluate whether it preserves the molecular core in practice using Bemis–Murcko frameworks (Bemis & Murcko, 1996).

For each successful MOLT output, we extract the Bemis–Murcko framework of the optimized molecule and its starting lead and record whether the two frameworks are identical. This provides a direct scaffold-level check that is complementary to fingerprint similarity.

Table 7: Bemis–Murcko framework retention among successful MOLT outputs at different seed-relative Tanimoto similarities.

Seed-relative similarity	Successful molecules	Framework retained (%)
$[0.50, 0.60)$	348	34.2
$[0.60, 0.70)$	457	63.9
≥ 0.70	697	87.1
All ≥ 0.50	1502	67.8

Framework retention increases monotonically with seed-relative similarity, confirming that Morgan-fingerprint Tanimoto similarity tracks Bemis–Murcko scaffold preservation in practice. The 0.50

threshold therefore serves as a permissive structural-continuity constraint rather than an exact scaffold-preservation requirement.

Retaining alternative branches also keeps the optimization structurally closer to the lead throughout the episode, not only at the endpoint. Figure 5 tracks the seed-relative Tanimoto similarity of successive selected outputs: planning over retained branches drifts far less than single-trajectory or local selection.

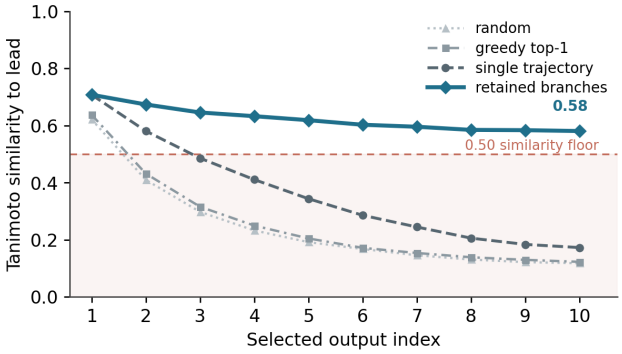


Figure 5: Seed-relative Tanimoto similarity over successive selected outputs, averaged over queries still active at each index. Planning over retained branches remains substantially closer to the original lead than the single-trajectory and local-selection baselines.

A.7 Action-Selection Shortlist Size

The context-conditioned policy returns a shortlist of K transformations to the planner at each molecule. As K increases, coverage (Hit@K) rises monotonically while precision declines, so the harmonic mean of Hit@K and Precision@K peaks at $K = 5$ for the learned policy (Figure 6). We therefore pass a shortlist of $K = 5$ transformations, which balances covering a task-satisfying edit against keeping the branching factor of the search small.

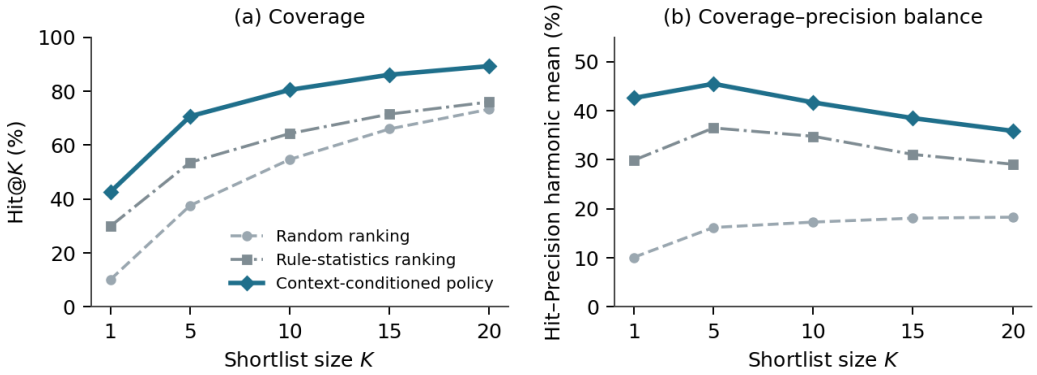


Figure 6: Context-conditioned action selection across shortlist sizes. **(a)** Hit@K, the fraction of eligible queries whose shortlist contains at least one successful transformation. **(b)** Harmonic mean of Hit@K and Precision@K, which peaks at $K = 5$ for the learned policy. Results are macro-averaged over five tasks and three seeds.

A.8 Generalization to ChEMBL

We evaluate database-level transfer on 100 leads sampled from ChEMBL that are disjoint from the MuMOInstruct training and benchmark sources and satisfy the same source precheck as the main benchmark. Eligible leads are available for the CNS/BBBP and Safety/Mutagenicity tasks, with 50

queries from each. All methods are evaluated on the same sources using the same success criteria as in the main experiments.

Table 8: Generalization to 100 ChEMBL leads from two constrained single-target tasks. Val, Sim, Prop, and SR follow the definitions used in Table 1. Best SR is shown in bold.

Method	Val	Sim	Prop	**SR**
GeLLM^{3}O-C-P(10)-Mistral	94.0	87.0	15.0	12.0
TRACE-Parallel	100.0	100.0	45.0	45.0
MOLT (top-1 action)	100.0	56.0	99.0	56.0
MOLT (single trajectory)	100.0	74.0	92.0	68.0
MOLT (ours)	100.0	97.0	84.0	**81.0**

MOLT reaches an SR of 81.0 on the ChEMBL leads, compared with 45.0 for TRACE-Parallel and 12.0 for the strongest evaluated generative baseline in this setting. Its SR is also close to the 81.8 obtained on the same two task families in the main benchmark, providing evidence that the learned transformation knowledge and decision process transfer beyond the source database.

A.9 Computational Cost

MOLT does not use a neural molecular generator at inference time. Candidate edits are produced by deterministic rule application, after which the action policy and LLM controller select among the resulting alternatives. Table 9 compares the resulting computational profile with TRACE on the multi-property benchmark.

Table 9: Per-query computational profile on the multi-property benchmark using NVIDIA A5000 GPUs. Wall-clock values are measured in our reruns.

Method	Controller	GPUs	Neural gen./query	Wall-clock/query
MOLT (ours)	local 7B LLM	1	0	seconds
TRACE-Online	GPT-4o-mini	4	≤ 18	$\approx$ 30s
TRACE-Parallel	GPT-4o-mini	4	≤ 72	$\approx$ 30s

On identical queries, MOLT runs in seconds per query on one GPU, whereas both TRACE variants require roughly 30 seconds per query on four GPUs. TRACE-Parallel additionally performs up to 72 neural molecular generations per query, while MOLT applies explicit transformations with RDKit and performs no neural molecular generation. The TRACE wall-clock measurements exclude its per-step GPT-4o-mini planner calls, so the reported timing comparison is conservative.