FIRM: FROM INTERFACES TO REALIZATIONS IN FRAGMENT-BASED MOLECULAR GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Individually valid fragments still require compatible attachments to form valid molecules. Yet compatible assembly alone does not ensure that generated molecules follow the reference distribution. This motivates learning connection requirements separately from fragment identities, allowing the model to select among compatible fragment structures according to the learned molecular distribution. We introduce **FIRM**, the **F**ragment **I**nterface **R**ealization **M**odel, which generates typed connection requirements before selecting compatible fragment structures. FIRM uses two discrete flows. Interface Flow jointly models typed connections with a latent probabilistic circuit, while Conditional Realization Flow jointly refines fragment templates and attachment mappings through discrete denoising within the interface-defined compatible space. On MOSES and NPGen, conditioning on interfaces rather than untyped fragment topology reduces the empirical training-set conditional entropy of fragment templates and attachment mappings by approximately 61% and 49.5%, respectively. Generation-order comparisons show that interface-first generation improves success without structural correction by 3.25 and 2.67 percentage points over mixed-order generation. Additionally, FIRM more closely matches reference molecular feature distributions than the evaluated fragment-graph baselines. This formulation provides a foundation for controllable molecular design, allowing future methods to explore alternative fragment structures under specified connection requirements.

1 INTRODUCTION

Fragment-based molecular generation builds molecules by assembling reusable chemical substructures (Jin et al., 2018; 2020; Podda et al., 2020; Maziarz et al., 2022; Kong et al., 2022). To learn a molecular distribution, such models capture the statistical patterns of both the fragments and their modes of combination in the training data. Reusing fragments preserves their internal structures, but models must still specify inter-fragment attachment sites and bond types while satisfying atom-level valence constraints. These attachment choices affect both chemical validity and the structures of the resulting molecules. Moreover, valid substructures are not necessarily mutually compatible, and different connections among the same substructures can yield distinct molecules (Jin et al., 2018). Even with fixed fragment identities, multiple feasible attachment schemes may remain. Valence constraints can eliminate infeasible attachments but do not determine the probability assigned to each feasible scheme. Fitting the reference molecular distribution therefore requires jointly learning the probabilistic patterns governing fragment selection and connectivity.

Existing methods determine inter-fragment connectivity through assembly, stepwise generation, or structural reconstruction. Assembly and graph-growth methods select attachment sites and connect fragments during molecular construction (Jin et al., 2018; 2020; Maziarz et al., 2022). MSTG predicts connectivity from previously generated substructures and their tree context before selecting the next substructure (Wang et al., 2022). Coarse-to-fine methods reconstruct atom-level connectivity from fragment-level structure (Kong et al., 2022; Lee et al., 2026). MiCaM selects connection-aware motifs, while motif graphs and fragment strings encode attachments alongside fragment identity (Geng et al., 2023; Liu et al., 2026; Zhu et al., 2026; Noutahi et al., 2024). MAGNet first generates scaffolds and their connectivity, then assigns chemical features and resolves attachment positions. Yet its scaffolds already fix internal atom adjacency (Hetzel et al., 2025), motivating a representation that specifies connection requirements while leaving internal fragment structures unspecified.

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.

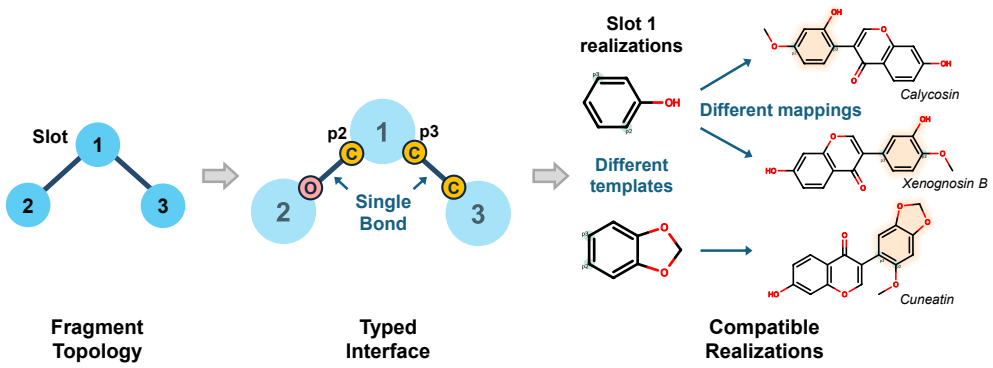


Figure 1: From fragment topology to compatible molecular realizations. Fragment topology specifies adjacency among anonymous slots. The typed interface adds attachment-port types, pairings, and bond types without fixing internal fragment graphs or atom-level attachment sites. Here, slot 1 has two carbon ports, p_2 and p_3, connected by single bonds to an oxygen port in slot 2 and a carbon port in slot 3. With the other slots fixed, the upper template yields Calycosin and Xenognosin B through different mappings from these ports to its attachment sites. The lower template yields Cuneatin under the same interface requirements. Shading highlights the slot 1 fragment in each molecule and its attachment sites in the template drawings. Port letters indicate elements; BRICS class indices are omitted for clarity.

We represent these requirements as an *anonymous typed molecular interface* (Figure 1). It specifies attachment-port types, pairings, and inter-fragment bond types without assigning fragment identities or atom-level attachment sites. Each slot is realized by a compatible template and a mapping from ports to attachment sites. Multiple templates and mappings can satisfy the same interface, allowing different molecular structures to share connection requirements.

FIRM, the **F**ragment **I**nterface **R**ealization **M**odel, learns this decomposition with two discrete flows followed by typed assembly (Figure 2). Interface Flow uses a latent probabilistic circuit to capture dependencies among port types, pairings, and bond types. The resulting interface determines the admissible templates and attachment mappings for each slot. Conditional Realization Flow learns their probabilities through discrete denoising, using the current assignments across the molecule so that fragment choices can adapt to one another. Both flows update categorical variables in parallel. Typed assembly connects the fragments at the assigned sites with the specified bond types.

Our experiments test whether connection requirements provide useful information before fragment structures are chosen. On MOSES and NPGen, interfaces reduce uncertainty over templates and attachment mappings beyond untyped topology, while leaving multiple realizations possible. Generating this information first also improves realization prediction and raises success without structural correction by 3.25 and 2.67 percentage points over mixed-order generation. Across four benchmarks, FIRM achieves closer molecular feature distribution matching than the evaluated fragment-graph baselines, showing that the interface formulation supports learning molecular distributions as well as compatible assembly.

Our contributions are:

- We formulate an anonymous typed molecular interface that specifies connection requirements while leaving internal fragment structures open to multiple compatible realizations.
- We develop FIRM, a two-stage discrete flow model that learns connection patterns with a latent probabilistic circuit and coordinates compatible fragment and attachment choices through conditional denoising.
- We show that interface conditioning reduces empirical training-set realization entropy by approximately 61% on MOSES and 49.5% on NPGen, while generating interfaces first improves realization prediction and generation success. Across four benchmarks, FIRM improves generation metrics and closely matches reference distributions.

2 Related Work

Fragment-based molecular generation and connectivity. BRICS defines reaction-motivated fragmentation and typed attachment points (Degen et al., 2008). JT-VAE generates junction trees and enumerates cluster assemblies (Jin et al., 2018). MSTG labels attachment positions and predicts connections conditioned on existing substructures (Wang et al., 2022). Hierarchical motif generation predicts attachment configurations between motif- and atom-level decisions, while MoLeR adds motifs during scaffold-conditioned graph construction (Jin et al., 2020; Maziarz et al., 2022). Coarse-to-fine methods reconstruct connectivity from fragment-level structure: PS-VAE predicts global connections among generated principal subgraphs, while FragFM uses a continuous latent code with a coarse fragment graph to score and match candidate atom-level edges (Kong et al., 2022; Lee et al., 2026).

Connection information can also be part of the generated representation. MiCaM jointly selects connection-aware motifs and their connections (Geng et al., 2023). Fragment language models serialize substructure additions, SAFE encodes fragment connectivity in sequence tokens, and motif graphs record motif identity, inter-motif bonds, and directional attachment positions (Podda et al., 2020; Noutahi et al., 2024; Liu et al., 2026; Zhu et al., 2026). MAGNet generates abstract scaffolds and their connectivity through shared join atoms, then assigns atom and bond types and resolves join positions. These scaffolds retain internal untyped atom adjacency (Hetzel et al., 2025).

Grammar-based models express structural constraints through production rules. MHG builds a molecular hypergraph grammar, DEG learns graph grammars from limited data, and HGR serializes higher-order molecular complexes as production-rule sequences (Kajino, 2019; Guo et al., 2022; Huang et al., 2026).

Discrete diffusion and flow matching. D3PM defines categorical corruption and denoising kernels, which DiGress adapts to graphs (Austin et al., 2021; Vignac et al., 2023). GDSS uses coupled stochastic differential equations, while DisCo and Cometh use continuous-time processes on discrete graph states (Jo et al., 2022; Xu et al., 2024; Siraudin et al., 2025). Discrete flow matching defines probability paths and finite-state dynamics between source and data distributions; DeFoG applies this framework to graphs (Gat et al., 2024; Qin et al., 2025). General discrete paths allow nonuniform sources and endpoint-dependent schedules (Shaul et al., 2025). In molecular generation, MELD studies how learned per-element corruption trajectories and schedules affect denoising (Seo et al., 2026).

3 Problem Formulation

FIRM generates an anonymous typed interface and then a compatible molecular realization. Interface generation is conditioned on a structural descriptor specifying the number of fragment slots and the cycle rank; slot topology is generated as part of the interface. A realization assigns each slot a compatible fragment template and a mapping from interface ports to atom-level attachment sites. Typed assembly restores the specified inter-fragment bonds.

For a molecular graph $G = (V_G, E_G)$ with atoms as vertices and bonds as edges, BRICS decomposition identifies bonds between fragments (Degen et al., 2008). Let $\mathcal{B}$ be the set of inter-fragment bond types. An endpoint type $s = (\ell, \tau, \bar{b}) \in \mathcal{S}$ encodes a BRICS label, attachment-atom type, and removed bond type. The allowed ordered connections form a reversal-closed set $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{B}$. The endpoint field $\bar{b}$ enforces bond-type consistency when an attachment port is matched to an atom-level site. The edge label b specifies the bond restored between two fragments, and a valid connection requires $\bar{b}_i = \bar{b}_j = b$. The descriptor $c = (N, \beta)$ gives $M = N - 1 + \beta$ connections among N slots.

An interface $I \in \mathcal{I}_c$ uses port signatures represented in the training fragment bank and contains typed ports and their pairings, but no fragment identities or atom-level sites. It is feasible when its slot graph is simple and connected, every port belongs to one connection, its numbers of slots and connections agree with c, and every typed connection satisfies $(s_i, s_j, b) \in \mathcal{R}$.

Given I, let $\mathcal{P}_v(I)$ be the required ports at slot v and $S(f)$ the complete set of atom-level attachment sites of template f. An attachment mapping is a type-preserving bijection $m : \mathcal{P}_v(I) \xrightarrow{\sim} S(f)$. Thus, distinct ports map to distinct sites and every site is used, although several sites may share an

attachment atom. Mappings related by an automorphism of the template feature graph form an orbit; throughout the main text, m in a realization state $a_v = (f, m)$ denotes a fixed canonical representative of that orbit (Appendix A.1). These orbits can group mappings that yield distinct stereochemical or isotopic realizations. Under the typed assembly rules, I specifies the connection constraints between slots. A realization therefore selects one locally compatible template and mapping state per slot:

$$\Omega_v(I) = \{(f, m) : (f, m) \text{ is compatible with } \mathcal{P}_v(I)\}, \qquad \Omega(I) = \prod_{v=1}^{N} \Omega_v(I). \tag{1}$$

A realization $A = (a_v)_{v=1}^{N}$ selects one state per slot. The product specifies the admissible assignments; it does not require their joint distribution to factorize across slots. The realization model learns these dependencies within the compatible space. Generation proceeds as

$$c \sim p(c), \qquad I \sim p_{\theta_1}(I \mid c), \qquad A \sim p_{\theta_2}(A \mid I), \qquad \widehat{G} = \text{Assemble}(I, A), \tag{2}$$

where $p(c)$ is empirical and $p_{\theta_2}(A \mid I)$ is supported on $\Omega(I)$.

4 FIRM: Fragment Interface Realization Model

FIRM implements Equation 2 through Interface Flow, Conditional Realization Flow, and typed assembly (Figure 2). Both flows alternate clean-state proposals with discrete bridge updates.

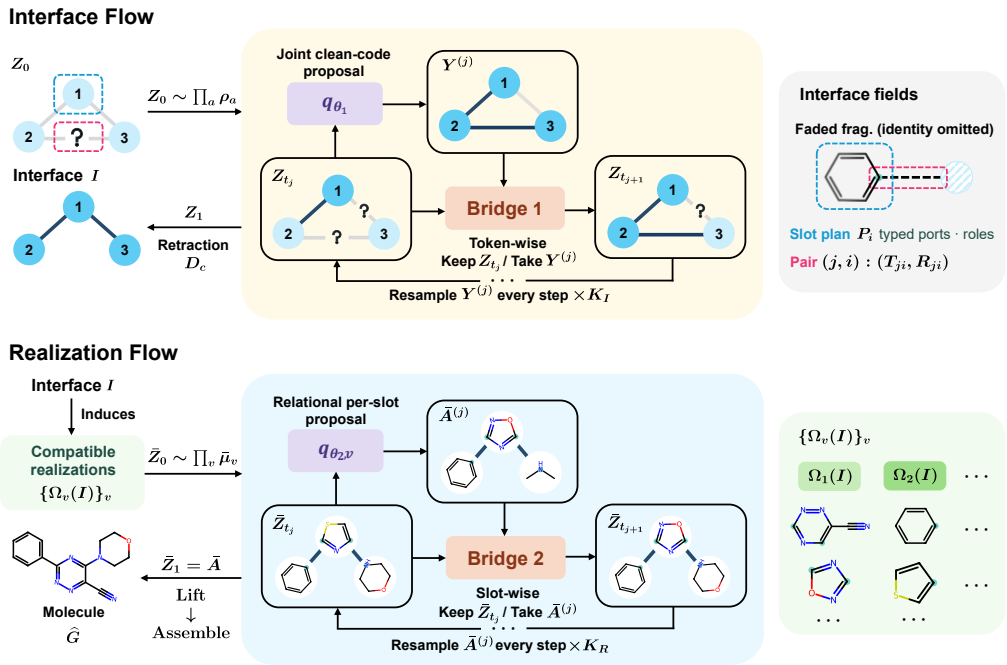


Figure 2: Overview of FIRM. Top: $Z_0 \sim \prod_a \rho_a$ initializes the code from sources ρ_a, where a indexes code coordinates (signature, port-role, pair-kind, and rule tokens); q_{θ_1} proposes a clean code $Y^{(j)}$ given (Z_{t_j}, t_j, c). The diagram shows successful retraction $I = D_c(Z_1)$; D_c may instead return $\perp$. Right: P_i encodes the required ports $\mathcal{P}_i(I)$ and code-specific roles; (T_{ji}, R_{ji}) encodes pair kind and connection rule; fragment identity remains unspecified. Bottom: I defines compatible template-and-mapping sets $\Omega_v(I)$. Fragment drawings are schematic; attachment mappings and compression are omitted. Bars denote compressed realization states; $\bar{\mu}_v$ is their per-slot source distribution. Given the shared context $(\bar{Z}_{t_j}, I, t_j)$, the per-slot predictors $q_{\theta_2, v}$ independently draw the components of $\bar{A}^{(j)}$; context is recomputed after each step. Bridges retain values or adopt proposals token-wise in K_I interface steps and slot-wise in K_R realization steps. The lift $L(A \mid \bar{A}, I)$ samples assignments A uniformly within $\bar{A}$; $\text{Assemble}(I, A)$ constructs $\widehat{G}$. Appendix D.1 shows generation trajectories.

4.1 INTERFACE FLOW

Interface Flow learns dependencies among port types, their assignments, and inter-slot connections with a joint clean-code predictor. For descriptor c, the canonical encoder gives

$$\mathcal{Y}_c^{\mathrm{can}} = E_c(\mathcal{I}_c) \subseteq \mathcal{Y}_c^{\mathrm{valid}} \subseteq \mathcal{Y}_c. \tag{3}$$

Here $\mathcal{Y}_c^{\mathrm{valid}}$ contains structurally valid codes, including noncanonical orderings. The code $Y = E_c(I)$ has slot blocks $Y_i = (P_i, \{T_{ji}, R_{ji}\}_{j<i})$: P_i specifies typed ports, T_{ji} the pair kind, and R_{ji} the ordered compatibility tuple for a connected pair, with $R_{ji} = 0$ for no edge. Reversal swaps endpoint types and preserves the bond type. Because one port plan can affect several pair coordinates, the predictor models dependencies within and across slot blocks.

A code coordinate a is one categorical token: a slot signature, a port-role entry, a pair kind, or a connection rule. Let ρ_a assign positive mass to every clean value of that token, and let α be nondecreasing with $\alpha(0) = 0$, $\alpha(1) = 1$, and $\alpha(s) < 1$ for all $0 \le s < 1$. The conditional reference path is

$$R_t(z \mid y) = \prod_a [(1 - \alpha(t))\rho_a(z_a) + \alpha(t)\delta_{y_a}(z_a)]. \tag{4}$$

The path runs from $\prod_a \rho_a$ to the clean interface code y. For $s < t$, the bridge preserves coordinates already equal to y_a and otherwise reveals y_a with probability $[\alpha(t) - \alpha(s)]/[1 - \alpha(s)]$. From a corrupted code Z_t, we predict $q_{\theta_1}(Y \mid Z_t, t, c)$ over $\mathcal{Y}_c$ using a balanced-tree latent probabilistic circuit, building on tractable and conditional circuit formulations (Choi et al., 2020; Shao et al., 2022). Its separator rank r controls dependence across hierarchical partitions; $r = 1$ factorizes the prediction across code blocks.

Writing $Y = E_c(I)$, with $(Y, c) \sim p_{\mathrm{data}}$, $t \sim \widetilde{\lambda}$, and $Z_t \sim R_t(\cdot \mid Y)$, training minimizes

$$\mathcal{L}_{\mathrm{int}}(\theta_1) = \mathbb{E}_{c,Y,t,Z_t}\left[-w(t) \log q_{\theta_1}(Y \mid Z_t, t, c)\right]. \tag{5}$$

Here $\widetilde{\lambda}$ is the training-time distribution, and $w(t)$ weights samples according to the target time measure (Appendix A.3).

During generation, a joint clean-code proposal is followed by parallel bridge updates. With the exact clean-code posterior, these updates match the joint reference marginal at each grid time; the circuit approximates this posterior during generation. Appendix A.3 gives the full construction and analysis.

Because $\mathcal{Y}_c$ includes infeasible codes, $D_c : \mathcal{Y}_c \to \mathcal{I}_c \cup \{\perp\}$ maps the terminal sample to a feasible interface or $\perp$, with $D_c(E_c(I)) = I$. It decodes structurally valid codes directly and otherwise maximizes agreement with the sampled code subject to the interface constraints or returns $\perp$. Conditional on successful decoding at fixed c, this composition induces $p_{\theta_1}(I \mid c)$ in Equation 2; $\perp$ marks an unsuccessful attempt.

4.2 CONDITIONAL REALIZATION FLOW

Given I, Conditional Realization Flow learns a distribution over compatible assignments $A \in \Omega(I)$. The predictor is conditioned on the interface and the current assignments at all slots. To handle templates with many attachment mappings, the flow operates on compressed states and samples a concrete assignment at the endpoint. For the canonical states in Section 3, ϕ_v retains fully enumerated states as singletons; in larger template-specific sets, observed states remain singletons and the rest are grouped by template. Thus,

$$\bar{a}_v = \phi_v(a_v), \qquad \bar{A} = (\bar{a}_v)_v, \qquad \bar{\Omega}_v(I) = \phi_v(\Omega_v(I)), \qquad \bar{\Omega}(I) = \prod_v \bar{\Omega}_v(I).$$

Conditional Realization Flow operates on $\bar{A}$. Let $\bar{\mu}_v$ be a source distribution with full support on $\bar{\Omega}_v(I)$. For a clean compressed state $\bar{a}_v$, the reference path is

$$R_t(z_v \mid \bar{a}_v) = \kappa_t(\bar{a}_v)\delta_{\bar{a}_v}(z_v) + [1 - \kappa_t(\bar{a}_v)]\bar{\mu}_v(z_v). \tag{6}$$

With slot index v suppressed, κ_t adapts the functional form of Shaul et al. (2025) to the source mass per concrete state represented by $\bar{a}$. It is nondecreasing and satisfies $\kappa_0(\bar{a}) = 0$ and $\kappa_1(\bar{a}) = 1$

for every $\bar{a}$. Each marginal $q_{\theta_2,v}(\bar{a}_v \mid \bar{Z}_t, I, t)$ is conditioned on the complete interface and joint compressed state, and can use all other assignments. Let $m_v(\bar{a}) = |\phi_v^{-1}(\bar{a})|$ denote the multiplicity of $\bar{a}$. Under the uniform lift, the per-slot denoising loss in the concrete state space is

$$\ell_{\mathrm{real},v} = -\log q_{\theta_2,v}(\bar{a}_v \mid \bar{Z}_t, I, t) + \log m_v(\bar{a}_v).$$

Training minimizes the expected mean slot loss over training molecules, a shared time $t \sim 0.1\delta_0 + 0.9\,\mathrm{Unif}(0,1)$, and $\bar{Z}_t \sim \prod_v R_t(\cdot \mid \bar{a}_v)$:

$$\mathcal{L}_{\mathrm{real}}(\theta_2) = \mathbb{E}\left[\frac{1}{N}\sum_{v=1}^{N} \ell_{\mathrm{real},v}\right]. \tag{7}$$

The denoising loss gives each molecule equal weight. During sampling, synchronous slot updates define a product kernel that approximates the oracle kernel based on the full joint posterior. The relational context is recomputed after each synchronous step. As the assignments evolve, each slot can therefore respond to the fragments currently selected elsewhere in the molecule. The interface fixes which assignments are compatible, while the learned context determines their relative probabilities during transport.

Let $\bar{p}_{\theta_2}(\bar{A} \mid I)$ denote the sampling chain's terminal joint law on compressed assignments. A uniform draw from each $\phi_v^{-1}(\bar{a}_v)$ restores a concrete assignment and induces

$$p_{\theta_2}(A \mid I) = \sum_{\bar{A} \in \bar{\Omega}(I)} \bar{p}_{\theta_2}(\bar{A} \mid I) L(A \mid \bar{A}, I).$$

For $\bar{A} = (\phi_v(a_v))_v$, exactly one term contributes, giving $p_{\theta_2}(A \mid I) = \bar{p}_{\theta_2}(\bar{A} \mid I) / \prod_v m_v(\bar{a}_v)$. The uniform lift preserves normalization and support on the mapping-orbit space $\Omega(I)$. Appendices A.5 and A.4 give the finite-step bound and lift derivation.

4.3 Typed assembly

Given an interface satisfying $\mathcal{R}$, typed assembly checks that the selected attachment mappings match its endpoint types. Assembly is deterministic given I and the canonical mappings in A. It copies the selected fragment into each slot and uses these mappings to locate the atoms at each end of an interface connection. It removes temporary port-capping hydrogen valence as the fragments are merged and restores the specified inter-fragment bonds. The resulting graph is checked by RDKit sanitization.

5 Experiments

5.1 Experimental setup

We evaluate FIRM on MOSES (Polykovskiy et al., 2020), NPGen (Lee et al., 2026), GuacaMol (Brown et al., 2019), and ZINC250K (Irwin & Shoichet, 2005), using 16 updates per flow and selective mixed-integer linear programming (MILP) retraction. Appendices B.1–B.2, A.7, and C.8 specify preprocessing, metrics, model settings, and budgets.

Baselines cover atom-level graphs (GraphINVENT, DiGress, DeFoG), fragment graphs (JT-VAE, MolGrow, FragFM), and fragment sequences (SAFE-GPT), spanning autoregressive, diffusion, and flow-based generation. FragFM is the closest fragment-graph comparison; it recovers atom-level connections by scoring and matching candidate edges. SAFE-GPT provides the fragment-sequence comparison. Baseline results follow FragFM's benchmark tables (Lee et al., 2026; Noutahi et al., 2024).

On MOSES and NPGen, a joint model tests generation order by generating interface and realization variables within one iterative process. Its interface-first (IF), realization-first (RF), and mixed (MIX) variants share data, state spaces, architecture, initialization, and budgets, varying only reveal order. FIRM uses two separately trained flows, generating and selectively retracting an interface before generating compatible realizations. Joint-model comparisons omit intermediate correction to isolate order effects, measuring successful assembly and sanitization with feasible interfaces and compatible realizations (Appendix C.2).

5.2 DISTRIBUTION LEARNING ACROSS FOUR BENCHMARKS

Table 1: MOSES distribution-learning results. FIRM results are means over three independent training runs with 25,000 generation attempts per run; baseline values are reproduced from Lee et al. (2026). Red and blue denote autoregressive and non-autoregressive baselines, respectively. Bold and underlining mark the best and second-best model results. Appendices B.3 and B.2 give standard deviations and protocols.

Model	Rep. level	Valid↑	Unique↑	Novel↑	Filters↑	FCD↓	SNN↑	Scaf↑
Training set	–	**100.0**	**100.0**	–	**100.0**	0.48	0.59	0.0
GraphINVENT (Mercado et al., 2021)	Atom (Graph)	96.4	99.8	–	95.0	1.22	0.54	12.7
JT-VAE (Jin et al., 2018)	Fragment (Graph)	**100.0**	**100.0**	99.9	97.8	1.00	0.53	10.0
SAFE-GPT (Noutahi et al., 2024)	Fragment (Sequence)	98.1	**100.0**	90.9	98.2	0.71	0.54	9.8
MolHF (Zhu et al., 2023)	Atom (Graph)	71.0	**100.0**	**100.0**	17.8	35.40	0.23	1.5
MolGrow (Kuznetsov & Polykovskiy, 2021)	Fragment (Graph)	**100.0**	**100.0**	99.8	82.7	6.39	0.42	7.7
DiGress (Vignac et al., 2023)	Atom (Graph)	85.7	**100.0**	95.0	97.1	1.19	0.52	14.8
DisCo (Xu et al., 2024)	Atom (Graph)	88.3	**100.0**	97.7	95.6	1.44	0.50	15.1
Cometh (Siraudin et al., 2025)	Atom (Graph)	90.5	**100.0**	96.4	97.2	1.44	0.51	15.9
Cometh-PC (Siraudin et al., 2025)	Atom (Graph)	90.5	99.9	92.6	99.1	1.27	0.54	**16.0**
DeFoG (Qin et al., 2025)	Atom (Graph)	92.8	99.9	92.1	98.9	1.95	0.55	14.4
FragFM (Lee et al., 2026)	Fragment (Graph)	99.8	**100.0**	87.1	99.1	0.58	**0.56**	10.9
FIRM (ours)	Fragment (Graph)	**100.0**	99.9	88.6	**99.4**	**0.54**	0.55	13.4

Table 2: NPGen distribution-learning results. FIRM results are means over three independent training runs with 30,000 generation attempts per run; baseline values are reproduced from Lee et al. (2026). Red and blue denote autoregressive and non-autoregressive baselines, respectively. Lower FCD and KL divergences indicate closer distribution matching. Bold and underlining mark the best and second-best model results. Appendices B.3 and B.2 give standard deviations and protocols.

Model	Rep. level	Valid↑	Unique↑	Novel↑	NP score KL div.↓	NPClassifier KL divergence ↓ Pathway	Superclass	Class	FCD↓
Training set	–	100.0	100.0	–	0.0006	0.0002	0.0028	0.0094	0.01
GraphAF (Shi et al., 2020)	Atom (Graph)	79.1	63.6	95.6	0.8546	0.9713	3.3907	6.6905	25.11
JT-VAE (Jin et al., 2018)	Fragment (Graph)	**100.0**	97.2	99.5	0.5437	0.1055	1.2895	2.5645	4.07
HierVAE (Jin et al., 2020)	Fragment (Graph)	**100.0**	81.5	97.7	0.3021	0.4230	0.5771	1.4073	8.95
SAFE-GPT (Noutahi et al., 2024)	Fragment (Sequence)	96.5	98.6	73.5	**0.0024**	**0.0054**	**0.0414**	**0.1722**	**0.15**
MolHF (Zhu et al., 2023)	Atom (Graph)	71.0	59.6	97.6	0.8831	1.8072	9.1608	10.3760	31.26
DiGress (Vignac et al., 2023)	Atom (Graph)	85.4	**99.7**	**99.9**	0.1957	0.0229	0.3370	1.0309	2.05
DeFoG (Qin et al., 2025)	Atom (Graph)	85.9	98.4	99.2	0.1550	0.1252	0.4134	1.3597	4.46
FragFM (Lee et al., 2026)	Fragment (Graph)	98.0	99.0	95.4	0.0374	0.0196	0.1482	0.3570	1.34
FIRM (ours)	Fragment (Graph)	**100.0**	99.0	88.4	0.0109	0.0094	0.0915	0.2878	0.32

On MOSES, FIRM has the lowest FCD among the compared models: 0.537 versus FragFM's 0.58, with slightly higher novelty (88.6% versus 87.1%; Table 1). Scaffold similarity improves from 10.9 to 13.4, and the filter pass rate from 99.1% to 99.4%. These results indicate closer ChemNet and scaffold matching with fewer filter failures. FIRM also leads in FCD on GuacaMol (85.8 to 86.67, higher is better) and ZINC250K (0.630 to 0.461) (Appendix B.4).

On NPGen, FIRM leads evaluated graph models in NP-score divergence, all three NPClassifier divergences, and FCD. Relative to FragFM (Lee et al., 2026), FIRM shows a novelty–FCD trade-off: FCD falls from 1.34 to 0.3191, while novelty falls by 7.0 percentage points (95.4% to 88.4%; Table 2). SAFE-GPT matches reference distributions more closely but has lower novelty (73.5% versus FIRM's 88.4%). To assess whether distributional gains arise from training-set matches, we further remove exact training-set matches from FIRM (12.0% of valid samples) and reevaluate the retained molecules. FCD rises to 0.3847 and NP-score divergence from 0.0109 to 0.0239, while LogP and TPSA remain close to the reference (Appendix C.6).

Selective retraction corrects infeasible interface codes before realization, maximizing agreement with sampled codes under interface constraints while retaining feasible interfaces. This raises validity from 96.87% to 100.0% on MOSES and from 88.84% to 100.0% on NPGen (Appendix C.4).

5.3 Interface information and realization uncertainty

We quantify how connection requirements inform fragment choices through training-set $\widehat{I}(A; I \mid S_0) = \widehat{H}(A \mid S_0) - \widehat{H}(A \mid I, S_0)$. Here, S_0 fixes fragment count, cycle rank, and exact anonymous untyped topology; A assigns templates and attachment-mapping orbits (Appendix C.3).

Table 3: Interface-conditioned realization uncertainty. Empirical training-set estimates in nats unless noted otherwise. Realization labels A pair fragment templates with mapping orbits under template feature-graph automorphisms.

	Realization entropy		Interface information		
Dataset	$\widehat{H}(A \mid S_0)$	$\widehat{H}(A \mid I, S_0)$	$\widehat{I}(A; I \mid S_0)$	bits	Molecules
MOSES	11.999	4.710	7.290	10.517	1,584,663
NPGen	9.296	4.695	4.602	6.639	526,852

Conditioning on interfaces lowers empirical training-set realization entropy from 11.999 to 4.710 nats on MOSES and from 9.296 to 4.695 on NPGen, reductions of approximately 61% and 49.5% relative to untyped topology (Table 3). Connection requirements therefore reduce uncertainty about fragment templates and attachment mappings while retaining multiple compatible realizations (Appendix C.5).

Interface information also improves probability estimates within the compatible set. With the candidate set and backbone held fixed, adding typed connections to port-signature composition reduces held-out NLL at every evaluated time. The reductions are 0.107 and 0.172 nats at $t = 0$ on MOSES and NPGen. This evaluation is restricted to targets supported by the fragment bank constructed from training data (Appendix C.1).

5.4 Effect of generation order

The predictive value of interface information raises a further question: whether making it available first improves generation. We compare reveal orders within a joint model that generates interface and realization variables in one process, without intermediate correction. IF achieves the highest success: $80.07 \pm 0.25\%$ on MOSES and $65.67 \pm 1.28\%$ on NPGen. Figure 3 shows its gains over RF and MIX.

Dataset	Contrast	Mean $\pm$ SD
MOSES	IF $-$ RF	$+4.13 \pm 0.30$
MOSES	IF $-$ MIX	$+3.25 \pm 0.43$
NPGen	IF $-$ RF	$+7.66 \pm 1.91$
NPGen	IF $-$ MIX	$+2.67 \pm 0.60$

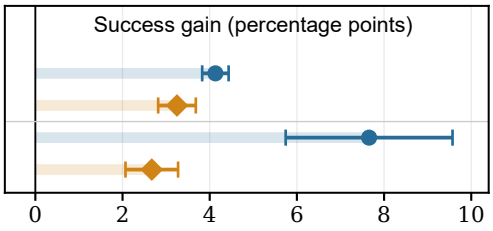


Figure 3: Success gains from interface-first joint generation. Training and sampling seeds are matched (42, 43, 44); IF has the highest success in every run. Table and plot rows align; markers and bars show means and sample SD over the three runs (percentage points).

IF produces fewer infeasible interfaces and incompatible realizations than MIX (Table 14). Establishing connection requirements first thus helps the model form feasible interfaces and choose fragments that satisfy them. These improvements support using the interface as an explicit intermediate generation target.

Even under IF, the joint model has lower validity and worse FCD than FIRM (Table 15). The remaining gap supports FIRM's staged design as a whole, which models interfaces separately, corrects infeasible proposals, and then generates realizations within the compatible space.

5.5 Component contributions to distribution learning

On MOSES, we compare Interface Flow's balanced tree ($r = 12$) with a parameter-matched Star and a block-factorized control ($r = 1$). The balanced tree links code blocks through a binary hierarchy of latent variables, whereas Star connects them all to one central latent variable. For realization, Source samples compatible states from smoothed training counts; Nodewise learns a 16-step flow from the interface and target-slot state; Relational also uses other slots' states (Appendix C.2).

Table 4: Component ablations on MOSES. Panel (a) reports clean-code denoising NLL per scored field at $t = 0$ and integrated over the reference path, and pre-retraction interface feasibility; panel (b) reports novelty (%), FCD, and scaffold similarity ($\times 100$). Nodewise and Relational results are averaged over three training runs; Source uses training counts without learned transport (seed 42).

(a) Interface Flow

Variant	NLL ($t = 0$)↓	Path-integrated NLL↓	Pre-retraction feas. (%)↑
Balanced tree ($r = 12$)	0.943	0.219	96.9
Star (parameter-matched)	1.255	0.280	96.7
Block-factorized ($r = 1$)	1.415	0.326	95.4

(b) Conditional Realization Flow

Variant	Novel↑	FCD↓	Scaf↑
Relational	88.6	0.537	13.4
Nodewise	91.3	0.638	11.8
Source	96.7	1.701	11.7

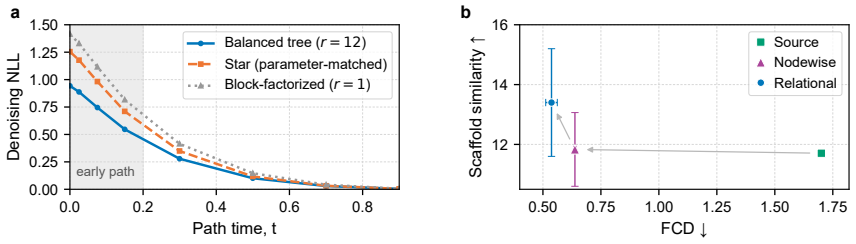


Figure 4: Component ablations on MOSES. (a) Denoising NLL per scored interface field. (b) Realization FCD and scaffold similarity ($\times 100$). Nodewise and Relational show three-run means $\pm$ sample SD; Source uses training counts and sampling seed 42 without learned transport.

The balanced tree lowers source-time NLL from Star's 1.255 to 0.943 and path-integrated NLL from 0.280 to 0.219, with similar feasibility (Table 4). Removing cross-block dependence raises NLL to 1.415 and lowers feasibility from 96.9% to 95.4%. The hierarchy gives its largest predictive gains early in the path, when most variables remain noisy (Figure 4a). Cross-block dependence also improves feasibility.

For conditional realization, Source has FCD 1.701, compared with 0.638 for Nodewise. Relational further lowers FCD to 0.537 and raises scaffold similarity from 11.8 to 13.4, with novelty falling from 91.3% to 88.6%. Conditional selection accounts for most of the FCD reduction; cross-slot context further improves FCD and scaffold similarity. Both learned variants also bring logP, QED, TPSA, and ring counts closer to the reference than Source (Appendix B.5, Figure 5).

6 Conclusion

FIRM separates connection requirements from fragment structures through two discrete flows. This separation reduces realization uncertainty and improves prediction and assembly when interfaces are generated first. Across four benchmarks, FIRM also improves molecular feature distribution matching over the evaluated fragment-graph baselines, with gains over FragFM extending to natural-product character on NPGen. The current model still relies on selective retraction and a training-derived fragment bank. Future work can generate feasible interfaces directly and expand compatible fragment choices under specified connection requirements.

AI USE STATEMENT

We used generative AI tools to assist with writing mathematical content, checking mathematical derivations for correctness, and translating parts of the manuscript. We also used these tools to improve manuscript readability, search for information, and summarize and analyze existing literature. All AI-assisted work was reviewed by the authors, and the authors take responsibility for the final content of this work, including its mathematical and other scientific claims, text, and artifacts.

ETHICS STATEMENT

This work studies unconditional molecular distribution learning and does not involve human participants, personal data, or animal experiments. Generated molecules are computational outputs and are not evidence of safety, efficacy, synthesizability, or biological activity. Molecular generators can support chemical design, but they can also be misused to propose hazardous compounds.

REPRODUCIBILITY STATEMENT

Appendix A provides mathematical derivations, model architectures, and training settings. Appendix B describes hardware, data preprocessing, and evaluation protocols; Appendix C details ablations and sampling costs. Both the main benchmarks on MOSES, NPGen, GuacaMol, and ZINC250K and the IF, MIX, and RF comparisons on MOSES and NPGen use three training seeds (42, 43, and 44) with matching sampling seeds, reporting means and sample standard deviations. Code and seed-42 generated molecules and evaluation results are available at `https://anonymous.4open.science/r/FIRM-iclr2027`.

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.

Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996. doi: 10.1021/jm9602928. URL https://doi.org/10.1021/jm9602928.

Nathan Brown, Marco Fiscato, Marwin H.S. Segler, and Alain C. Vaucher. GuacaMol: Benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling*, 59 (3):1096–1108, 2019. doi: 10.1021/acs.jcim.8b00839. URL https://doi.org/10.1021/acs.jcim.8b00839.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models, 2020. URL https://starai.cs.ucla.edu/papers/ProbCirc20.pdf.

Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the 27th International Conference on Machine Learning*, ICML'10, pp. 255–262. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/347.pdf.

Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem*, 3(10):1503–1507, 2008. doi: 10.1002/cmdc.200800178. URL https://doi.org/10.1002/cmdc.200800178.

Peter Ertl, Silvio Roggo, and Ansgar Schuffenhauer. Natural product-likeness score and its application for prioritization of compound libraries. *Journal of Chemical Information and Modeling*, 48(1):68–74, 2008. doi: 10.1021/ci700286x. URL https://doi.org/10.1021/ci700286x.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, pp. 133345–133385. Curran Associates, Inc., 2024. doi: 10.52202/079017-4239. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/f0d629a734b56a642701bba7bc8bb3ed-Paper-Conference.pdf.

Zijie Geng, Shufang Xie, Yingce Xia, Lijun Wu, Tao Qin, Jie Wang, Yongdong Zhang, Feng Wu, and Tie-Yan Liu. De novo molecular generation via connection-aware motif mining. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Q_Jexl8-qDi.

Samuel Genheden, Amol Thakkar, Veronika Chadimová, Jean-Louis Reymond, Ola Engkvist, and Esben Bjerrum. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. *Journal of Cheminformatics*, 12(1):70, 2020. doi: 10.1186/s13321-020-00472-1. URL https://doi.org/10.1186/s13321-020-00472-1.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Minghao Guo, Veronika Thost, Beichen Li, Payel Das, Jie Chen, and Wojciech Matusik. Data-efficient graph grammar learning for molecular generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=l4IHywGq6a.

Leon Hetzel, Johanna Sommer, Bastian Rieck, Fabian Theis, and Stephan Günnemann. MAGNet: Motif-agnostic generation of molecules from scaffolds. In *International Conference on Learning Representations*, volume 2025, pp. 27221–27245, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/43d1d3bdd92204c96fa4ac3c578f6a33-Paper-Conference.pdf.

Xiaoyang Hou, Tian Zhu, Milong Ren, Dongbo Bu, Xin Gao, Chunming Zhang, and Shiwei Sun. GGFlow: A graph flow matching method with efficient optimal transport. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=K8RlXtMgzo.

Yiming Huang, Yujie Zeng, Vijay Prakash Dwivedi, Simone Foti, Jianmin Wang, Jure Leskovec, and Tolga Birdal. Higher-order grammar representations for molecular generation and learning. In *Learning Meaningful Representations of Life (LMRL) Workshop at ICLR 2026*, 2026. URL https://openreview.net/forum?id=AHHCuMItmh.

John J. Irwin and Brian K. Shoichet. ZINC – a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modeling*, 45(1):177–182, 2005. doi: 10.1021/ci049714+. URL https://doi.org/10.1021/ci049714+.

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. doi: 10.1039/c8sc05372c. URL https://doi.org/10.1039/c8sc05372c.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2323–2332. PMLR, 2018. URL https://proceedings.mlr.press/v80/jin18a.html.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4839–4848. PMLR, 2020. URL https://proceedings.mlr.press/v119/jin20a.html.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10362–10383. PMLR, 2022. URL https://proceedings.mlr.press/v162/jo22a.html.

Jaehyeong Jo, Dongki Kim, and Sung Ju Hwang. Graph generation with diffusion mixture. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22371–22405. PMLR, 2024. URL https://proceedings.mlr.press/v235/jo24b.html.

Hiroshi Kajino. Molecular hypergraph grammar with its application to molecular optimization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3183–3191. PMLR, 2019. URL https://proceedings.mlr.press/v97/kajino19a.html.

Hyun Woo Kim, Mingxun Wang, Christopher A. Leber, Louis-Félix Nothias, Raphael Reher, Kyo Bin Kang, Justin J. J. van der Hooft, Pieter C. Dorrestein, William H. Gerwick, and Garrison W. Cottrell. NPClassifier: A deep neural network-based structural classification tool for natural products. *Journal of Natural Products*, 84(11):2795–2807, 2021. doi: 10.1021/acs.jnatprod.1c00399. URL https://doi.org/10.1021/acs.jnatprod.1c00399.

Xiangzhe Kong, Wenbing Huang, Zhixing Tan, and Yang Liu. Molecule generation by principal subgraph mining and assembling. In *Advances in Neural Information Processing Systems*, volume 35, pp. 2550–2563. Curran Associates, Inc., 2022. doi: 10.52202/068431-0185. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/1160792eab11de2bbaf9e71fce191e8c-Paper-Conference.pdf.

Maksim Kuznetsov and Daniil Polykovskiy. MolGrow: A graph normalizing flow for hierarchical molecular generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8226–8234, 2021. doi: 10.1609/aaai.v35i9.17001. URL https://ojs.aaai.org/index.php/AAAI/article/view/17001.

Youngchun Kwon, Dongseon Lee, Youn-Suk Choi, Kyoham Shin, and Seokho Kang. Compressed graph representation for scalable molecular graph generation. *Journal of Cheminformatics*, 12 (1):58, 2020. doi: 10.1186/s13321-020-00463-2. URL https://doi.org/10.1186/s13321-020-00463-2.

Joongwon Lee, Seonghwan Kim, Seokhyun Moon, Hyunwoo Kim, and Woo Youn Kim. FragFM: Hierarchical framework for efficient molecule generation via fragment-level discrete flow matching. In *International Conference on Learning Representations*, volume 2026, pp. 90722–90777, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/92f556b408cfac2943fb228207487967-Paper-Conference.pdf.

Xiao Qing Lewell, Duncan B Judd, Stephen P Watson, and Michael M Hann. RECAP retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *Journal of Chemical Information and Computer Sciences*, 38(3):511–522, 1998. doi: 10.1021/ci970429i.

Gang Liu, Jie Chen, Yihan Zhu, Michael Sun, Tengfei Luo, Nitesh V Chawla, and Meng Jiang. Graph diffusion transformers are in-context molecular designers. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lJ87GN5zJc.

Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3496–3508, 2024. doi: 10.1109/TPAMI.2023.3344758. URL https://doi.org/10.1109/TPAMI.2023.3344758.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. GraphDF: A discrete flow model for molecular graph generation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7192–7203. PMLR, 2021. URL https://proceedings.mlr.press/v139/luo21a.html.

Krzysztof Maziarz, Henry Richard Jackson-Flux, Pashmina Cameron, Finton Sirockin, Nadine Schneider, Nikolaus Stiefl, Marwin Segler, and Marc Brockschmidt. Learning to extend molecular scaffolds with structural motifs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ZTsoE8G3GG.

Rocío Mercado, Tobias Rastemo, Edvard Lindelöf, Günter Klambauer, Ola Engkvist, Hongming Chen, and Esben Jannik Bjerrum. Graph networks for molecular design. *Machine Learning: Science and Technology*, 2(2):025023, 2021. doi: 10.1088/2632-2153/abcf91. URL https://doi.org/10.1088/2632-2153/abcf91.

James S. Milne. Group theory (v3.15), 2020. URL https://www.jmilne.org/math/CourseNotes/GT315.pdf. Course notes. Available at www.jmilne.org/math/.

Emmanuel Noutahi, Cristian Gabellini, Michael Craig, Jonathan S. C. Lim, and Prudencio Tossou. Gotta be SAFE: a new framework for molecular design. *Digital Discovery*, 3(4):796–804, 2024. doi: 10.1039/d4dd00019f. URL https://doi.org/10.1039/d4dd00019f.

Marco Podda, Davide Bacciu, and Alessio Micheli. A deep generative model for fragment-based molecule generation. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2240–2250. PMLR, 2020. URL https://proceedings.mlr.press/v108/podda20a.html.

Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. MIT OpenCourseWare, 6.441 Information Theory, Spring 2016, 2016. URL https://ocw.mit.edu/courses/6-441-information-theory-spring-2016/5d8f16adc3385c9ff2975b121bd620e4_MIT6_441S16_course_notes.pdf. Course notes; preface dated 27 February 2015.

Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alán Aspuru-Guzik, and Alex Zhavoronkov. Molecular sets (MOSES): A benchmarking platform for molecular generation models. *Frontiers in Pharmacology*, 11:565644, 2020. doi: 10.3389/fphar.2020.565644. URL https://doi.org/10.3389/fphar.2020.565644.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 337–346, 2011. URL https://homes.cs.washington.edu/~pedrod/papers/uai11a.pdf.

Kristina Preuer, Philipp Renz, Thomas Unterthiner, Sepp Hochreiter, and Günter Klambauer. Fréchet ChemNet distance: A metric for generative models for molecules in drug discovery. *Journal of Chemical Information and Modeling*, 58(9):1736–1741, 2018. doi: 10.1021/acs.jcim.8b00234. URL https://doi.org/10.1021/acs.jcim.8b00234.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50269–50326. PMLR, 2025. URL https://proceedings.mlr.press/v267/qin25d.html.

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://doi.org/10.1021/ci100050t.

Hyunjin Seo, Taewon Kim, Sihyun Yu, and Sungsoo Ahn. Learning flexible forward trajectories for masked molecular diffusion. In *International Conference on Learning Representations*, volume 2026, pp. 95424–95448, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/9a17c1eb808cf012065e9db47b7ca80d-Paper-Conference.pdf.

Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting. Conditional sum-product networks: Modular probabilistic circuits via gate functions. *International Journal of Approximate Reasoning*, 140:298–313, 2022. doi: 10.1016/j.ijar.2021.10.011. URL https://www.sciencedirect.com/science/article/pii/S0888613X21001766.

Neta Shaul, Itai Gat, Marton Havasi, Daniel Severo, Anuroop Sriram, Peter Holderrieth, Brian Karrer, Yaron Lipman, and Ricky T. Q. Chen. Flow matching with general discrete paths: A kinetic-optimal perspective. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tcvMzR2NrP.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. GraphAF: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1esMkHYPr.

Antoine Siraudin, Fragkiskos D. Malliaros, and Christopher Morris. Cometh: A continuous-time discrete-state graph diffusion model. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=nuN1mRrrjX.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UaAD-Nu86WX.

Shuang Wang, Tao Song, Shugang Zhang, Mingjian Jiang, Zhiqiang Wei, and Zhen Li. Molecular substructure tree generative model for de novo drug design. *Briefings in Bioinformatics*, 23 (2):bbab592, 2022. doi: 10.1093/bib/bbab592. URL https://doi.org/10.1093/bib/bbab592.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4(1):66–82, 1960. doi: 10.1147/rd.41.0066. URL https://doi.org/10.1147/rd.41.0066.

Zhe Xu, Ruizhong Qiu, Yuzhong Chen, Huiyuan Chen, Xiran Fan, Menghai Pan, Zhichen Zeng, Mahashweta Das, and Hanghang Tong. Discrete-state continuous-time diffusion for graph generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 79704–79740. Curran Associates, Inc., 2024. doi: 10.52202/079017-2531. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/91813e5ddd9658b99be4c532e274b49c-Paper-Conference.pdf.

Qi Yan, Zhengyang Liang, Yang Song, Renjie Liao, and Lele Wang. SwinGNN: Rethinking permutation invariance in diffusion models for graph generation. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=abfi5plvQ4.

Yihan Zhu, Yuhan Liu, Weijiang Li, Tengfei Luo, and Meng Jiang. Controllable molecular generative foundation models. *arXiv preprint arXiv:2605.15354*, 2026. URL https://arxiv.org/abs/2605.15354.

Yiheng Zhu, Zhenqiu Ouyang, Ben Liao, Jialu Wu, Yixuan Wu, Chang-Yu Hsieh, Tingjun Hou, and Jian Wu. MolHF: A hierarchical normalizing flow for molecular graph generation. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 5002–5010. International Joint Conferences on Artificial Intelligence Organization, 2023. doi: 10.24963/ijcai.2023/556. URL https://doi.org/10.24963/ijcai.2023/556. Main Track.

SUPPLEMENTARY MATERIALS

Appendix A develops the mathematical details and implementation; Appendix B reports evaluation settings and additional benchmark results. Appendix C examines interface information, diversity, and model behavior, while Appendix D shows generation trajectories and molecular samples.

A Mathematical Details and Implementation

Interface Flow ends with a categorical code. Decoding produces a feasible interface or reports failure. On success, Conditional Realization Flow generates the terminal realization, which is lifted to concrete compatible states and passed to typed assembly.

A.1 Interface state space and canonical representation

Let $G = (V_G, E_G)$ denote a molecular graph with atoms as vertices and bonds as edges. When BRICS decomposition cleaves a bond between fragments, it yields two endpoint types and the removed bond type (Degen et al., 2008). An endpoint type is $s = (\ell, \tau, \bar{b}) \in \mathcal{S}$: ℓ is the BRICS label, τ is the attachment-atom type, and $\bar{b} \in \mathcal{B}$ is the removed bond type. The valid pairings form an ordered rule set $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{B}$ that is closed under reversal. The endpoint field $\bar{b}$ specifies the bond type associated with a local attachment choice, while the edge label b specifies the bond restored during assembly; valid edges require both endpoint fields to equal b. The structural descriptor is $c = (N, \beta)$, where N is the number of fragment slots and β is the cycle rank. It fixes $M = N - 1 + \beta$ edges between slots.

For fixed c, $\mathcal{I}_c$ contains the simple connected interfaces $I = (V, E, \{s_{v,k}\})$ with $|V| = N$ and $|E| = M$. Each endpoint port occurs in exactly one edge, and every edge carries a rule from $\mathcal{R}$. A root, a slot ordering, and a spanning tree are not part of I. Interfaces are identified up to type-preserving relabelling of slots and anonymous ports.

The encoder minimizes, over slot orders with connected prefixes, the lexicographic certificate that appends each slot signature and its typed edge labels to earlier slots. The first slot is the root; each later slot attaches to its earliest adjacent predecessor, defining the spanning tree. Tied orders yield the same typed code. The root has no UP port; at every other slot, one port is marked UP. Ports on tree edges are marked DOWN and UP according to their orientation, and the β non-tree edges use EXTRA ports. These roles belong to the code rather than the interface.

The port-plan alphabet is restricted to signatures supported by the training fragment bank, so every realization set $\Omega_v(I)$ is nonempty. For such an interface, let $\mathcal{P}_v(I)$ be the ports required at slot v and $S(f)$ the attachment sites of template f. A raw compatible mapping is a type-preserving bijection $m : \mathcal{P}_v(I) \xrightarrow{\sim} S(f)$. Attachment sites are template port slots, so distinct sites may refer to the same attachment atom. Let $\mathrm{Aut}(f)$ denote the permutation group on attachment sites induced by automorphisms of the template feature graph. Atom labels encode atomic number, formal charge, hybridization, and aromaticity; bond labels encode bond type, aromaticity, conjugation, and bond stereo. Typed port nodes connect to their attachment atoms. Atom chirality and isotope labels are omitted from this graph, so its mapping orbits can merge distinct stereochemical or isotopic realizations. The group acts on the compatible bijections by postcomposition (Milne, 2020, Chapter 4). Mappings in the same orbit produce one canonical template and mapping state; in the main text, m denotes its canonical representative. Thus $\Omega_v(I)$ contains pairs $(f, [m])$, written (f, m) for brevity. Since I fixes every cross-slot constraint, the resulting space of compatible orbit states factorizes as $\Omega(I) = \prod_{v \in V(I)} \Omega_v(I)$.

A.2 Canonical code and deterministic retraction

As in Equation 3, the canonical encoder writes $E_c(I) = Y \in \mathcal{Y}_c^{\mathrm{can}}$, where $\mathcal{Y}_c^{\mathrm{can}} = E_c(\mathcal{I}_c) \subseteq \mathcal{Y}_c^{\mathrm{valid}} \subseteq \mathcal{Y}_c$, with slot blocks $Y_i = (P_i, \{T_{ji}, R_{ji} : j < i\})$. The first component P_i is the typed port plan. For each $j < i$, the pair kind $T_{ji} \in \{\text{NONE}, \text{TREE}, \text{EXTRA}\}$ marks the relation between slots j and i. If they are connected, R_{ji} gives the ordered compatibility tuple; otherwise $R_{ji} = 0$ is the dedicated no-edge token. In clean-code prediction, this token is fixed when $T_{ji} = \text{NONE}$ and contributes no separate rule log-probability. During corruption, the stored rule coordinate, including its no-edge value, is corrupted like the other coordinates; the mask token is distinct from the no-edge token. Pair coordinates are stored only for $j < i$. Reversing the pair swaps its endpoint types and preserves the bond type, so the other orientation is recovered without another generated coordinate. Deterministic constraint masks enforce the local alphabet, endpoint inventory, and pair constraints.

Pre-retraction feasibility checks membership in $\mathcal{Y}_c^{\mathrm{valid}}$: the code must satisfy the structural interface constraints, while its ordering may still require canonicalization. Such codes are decoded directly.

Otherwise, $D_c : \mathcal{Y}_c \to \mathcal{I}_c \cup \{\perp\}$ projects the code onto the feasible set by solving a binary optimization problem over port plans and typed tree and extra edges. One plan is selected at each slot, its typed ports are matched to incident edges, every non-root slot receives one earlier tree parent, and exactly β extra edges are selected.

The primary objective counts code entries matching the sampled code. A deterministic secondary score refines this objective. It is computed from the raw code and a fixed variable ordering, with total magnitude below $1/2$, so it cannot compensate for the loss of one unit of primary agreement. The solver configuration, variable ordering, and any remaining tie resolution are fixed as part of D_c. Every successful projection is recanonicalized and checked before use.

Proposition 1 (Canonical decoding)**.** *For every $I \in \mathcal{I}_c$, $D_c(E_c(I)) = I$, and every successful output of D_c lies in $\mathcal{I}_c$. Moreover, for distributions p and q on $\mathcal{Y}_c$,*

$$\mathrm{TV}((D_c)_{\#}p, (D_c)_{\#}q) \le \mathrm{TV}(p, q). \tag{8}$$

Proof. The first two statements follow from direct decoding and the final feasibility check. The last statement is the data-processing inequality for total variation under the deterministic map D_c (Polyanskiy & Wu, 2016). □

Let $Q_{\theta_1,c}$ denote the terminal code law produced by Interface Flow for descriptor c. Its unconditional pushforward through D_c and its failure mass are

$$\widetilde{p}_{\theta_1}(\cdot \mid c) = (D_c)_{\#}Q_{\theta_1,c}, \qquad \varepsilon_{\theta_1}(c) = \widetilde{p}_{\theta_1}(\perp \mid c). \tag{9}$$

Whenever $\varepsilon_{\theta_1}(c) < 1$, the interface law in Equation 2 is the success-conditioned distribution

$$p_{\theta_1}(I \mid c) = \frac{\widetilde{p}_{\theta_1}(I \mid c)}{1 - \varepsilon_{\theta_1}(c)}, \qquad I \in \mathcal{I}_c, \tag{10}$$

or equivalently $\widetilde{p}_{\theta_1}(I \mid c) = [1 - \varepsilon_{\theta_1}(c)]p_{\theta_1}(I \mid c)$. Equation 8 concerns the unconditional laws on $\mathcal{I}_c \cup \{\perp\}$; the same contraction need not hold after conditioning on success.

Equation 2 specifies the generation components, with the interface law conditioned on successful decoding at fixed c. If $\mathsf{E}_{\mathrm{int}}$ denotes this event, then

$$p(c, I \mid \mathsf{E}_{\mathrm{int}}) = \frac{p(c)[1 - \varepsilon_{\theta_1}(c)]p_{\theta_1}(I \mid c)}{\sum_{c'} p(c')[1 - \varepsilon_{\theta_1}(c')]}.$$

For successful molecular generation $\mathsf{E}_{\mathrm{mol}}$, including realization, assembly, and sanitization, write $u(c) = \Pr(\mathsf{E}_{\mathrm{mol}} \mid c)$. The retained outputs have descriptor law $p(c \mid \mathsf{E}_{\mathrm{mol}}) = p(c)u(c)/\sum_{c'} p(c')u(c')$. Evaluation counts every attempt, including failures, in the validity denominator.

A.3 Interface Flow

Each port plan P_i comprises a signature and a padded vector of port roles; it encodes the required port set $\mathcal{P}_i(I)$ together with code-specific roles. In Equation 4, a indexes individual signature tokens, port-role entries, and upper-triangular pair-type and rule tokens. These coordinates are corrupted independently; reverse pair entries are recovered deterministically. The probabilistic circuit jointly proposes the plan and pair coordinates grouped into each block in Figure 2.

For coordinate a, let $\mathcal{X}_a$ be its clean alphabet, and augment it with a mask symbol. The standard source law is $\rho_a(\textsc{mask}) = 0.9$ and $\rho_a(x) = 0.1/|\mathcal{X}_a|$ for $x \in \mathcal{X}_a$. The conditional reference path is Equation 4 with $\alpha(t) = t$. For $0 \le s < t \le 1$, set $r_{s,t} = [\alpha(t) - \alpha(s)]/[1 - \alpha(s)]$. The coordinate bridge keeps y_a fixed. From $z_a \ne y_a$, it stays at z_a with probability $1 - r_{s,t}$ and is revealed as y_a with probability $r_{s,t}$.

Proposition 2 (Interface transport)**.** *Throughout this proposition, c is fixed; conditioning on it is suppressed unless shown. For fixed y,*

$$R_s(\cdot \mid y)B_{s,t}(\cdot \mid \cdot, y) = R_t(\cdot \mid y). \tag{11}$$

Let ν_1 be the clean-code law, with $\mu_t(z) = \sum_y \nu_1(y)R_t(z \mid y)$ and $\pi_t(y \mid z) = \nu_1(y)R_t(z \mid y)/\mu_t(z)$. Mixing the bridge over π_s gives $K^{\star}_{s,t}(z, z') = \sum_y \pi_s(y \mid z)B_{s,t}(z' \mid z, y)$ and $\mu_s K^{\star}_{s,t} =$

μ_t. *The learned kernel is* $\widehat{K}_{s,t}(z, z') = \sum_y q_{\theta_1}(y \mid z, s, c) B_{s,t}(z' \mid z, y)$. *On a grid* $0 = t_0 < \cdots < t_K = 1$, *let* $\widehat{\mu}_{t_0} = \mu_0$ *and* $\widehat{\mu}_{t_{j+1}} = \widehat{\mu}_{t_j} \widehat{K}_{t_j, t_{j+1}}$. *Then*

$$\begin{aligned} \mathrm{TV}(\widehat{\mu}_1, \nu_1) &\le \sum_{j=0}^{K-1} \mathbb{E}_{Z \sim \widehat{\mu}_{t_j}} \left[\mathrm{TV}\big(q_{\theta_1}(\cdot \mid Z, t_j, c), \pi_{t_j}(\cdot \mid Z, c)\big) \right] \\ &\le \sum_{j=0}^{K-1} \sup_z \mathrm{TV}\big(q_{\theta_1}(\cdot \mid z, t_j, c), \pi_{t_j}(\cdot \mid z, c)\big) \,. \end{aligned} \tag{12}$$

Proof. Fix one coordinate and write $b = \rho_a(y_a)$. Starting from the distribution at time s, the bridge gives the following masses at $x \neq y_a$ and y_a, respectively:

$$\begin{aligned} &(1 - r_{s,t})(1 - \alpha(s))\rho_a(x) = (1 - \alpha(t))\rho_a(x), \\ &\alpha(s) + (1 - \alpha(s))b + r_{s,t}(1 - \alpha(s))(1 - b) = \alpha(t) + (1 - \alpha(t))b. \end{aligned}$$

This calculation includes the base mass $\rho_a(y_a)$. Independence across coordinates yields the product identity. Substituting $\mu_s(z)\pi_s(y \mid z) = \nu_1(y)R_s(z \mid y)$ reduces the result to the conditional bridge identity. For the learned chain, Markov contraction and data processing through the common bridge channel bound each step; telescoping gives the expectation bound, and each expectation is at most the corresponding supremum. □

A balanced latent probabilistic circuit over the N code blocks is organized as a binary tree, following the sum-product representation of tractable distributions (Poon & Domingos, 2011; Choi et al., 2020). Each tree node carries a latent separator $H_v \in [r]$. The common cardinality r is the separator rank: it bounds the dependence passed across the hierarchical cut represented by that node. At $r = 1$, every separator is deterministic and the probability head factorizes across code blocks. For context $h = (Z_t, t, c)$, internal nodes $\mathcal{T}_{\mathrm{int}}$, and leaves $\mathcal{T}_{\mathrm{leaf}}$, the circuit factorizes as

$$q_{\theta_1}(Y \mid h) = \sum_H \pi(H_{\mathrm{root}} \mid h) \prod_{v \in \mathcal{T}_{\mathrm{int}}} \psi_v(H_{v_L}, H_{v_R} \mid H_v, h) \prod_{i \in \mathcal{T}_{\mathrm{leaf}}} e_i(Y_i \mid H_i, h), \tag{13}$$

where π is the root distribution, ψ_v is a joint conditional over the two child separators, and e_i emits a code block. Padding leaves emit unit mass. Every internal conditional and observable emission is normalized on its masked alphabet, so bottom-up summation yields $\sum_Y q_{\theta_1}(Y \mid Z_t, t, c) = 1$. The same sum-product recursion evaluates the probability of a given clean code. The Star control in Section 5.5 replaces this hierarchy with one central latent separator connected to all code blocks and chooses its cardinality to match the parameter count of Balanced tree ($r = 12$).

For a clean code Y and descriptor c, define $\ell_{\theta_1}(t; Y, c) = \mathbb{E}_{Z_t \sim R_t(\cdot \mid Y)}[-\log q_{\theta_1}(Y \mid Z_t, t, c)]$. The theoretical objective uses the target time measure $\lambda = 0.1\delta_0 + 0.9\,\mathrm{Unif}(0, 1)$:

$$\mathcal{L}_{\mathrm{int}}(\theta_1) = \mathbb{E}_{(Y,c) \sim p_{\mathrm{data}}} \mathbb{E}_{t \sim \lambda}[\ell_{\theta_1}(t; Y, c)]. \tag{14}$$

The time expectation is estimated by importance sampling. Time is drawn from $\widetilde{\lambda} = 0.1\delta_0 + 0.9g(t)\,dt$, where $g(t) = 0.5\,\mathrm{Beta}(t; 1, 3) + 0.5$ and $\mathrm{Beta}(t; 1, 3)$ is the beta density on $(0, 1)$. The importance weight is $w = d\lambda / d\widetilde{\lambda}$, with $w(0) = 1$ for the point mass at the source and $w(t) = 1/g(t)$ for $t \in (0, 1)$. This gives

$$\mathbb{E}_{(Y,c) \sim p_{\mathrm{data}}} \mathbb{E}_{t \sim \widetilde{\lambda}} \mathbb{E}_{Z_t \sim R_t(\cdot \mid Y)} \left[w(t)(-\log q_{\theta_1}(Y \mid Z_t, t, c)) \right] = \mathcal{L}_{\mathrm{int}}(\theta_1), \tag{15}$$

using one joint clean-code log probability per graph.

Training groups molecules by graph size and discards incomplete batches for efficiency. This changes the relative weights of graph sizes compared with the full empirical distribution.

Algorithm 1 Interface Flow sampling

Require: structural descriptor c, grid $0 = t_0 < \cdots < t_{K_I} = 1$
1: sample $Z_{t_0} \sim \prod_a \rho_a$
2: **for** $j = 0, \ldots, K_I - 1$ **do**
3: sample a joint clean-code proposal $Y^{(j)} \sim q_{\theta_1}(\cdot \mid Z_{t_j}, t_j, c)$
4: update code coordinates with the conditional bridge
5: **end for**
6: **return** $D_c(Z_1)$ if decoding succeeds

A.4 Compressed realization states, source law, and concrete lift

Given the decoded interface I, Conditional Realization Flow operates on compressed compatible states. Let $\mathcal{O}_{v,f}$ be the set of compatible mapping orbits $[m]$ for template f. After quotienting by automorphisms, the map $\phi_v : \Omega_v(I) \to \bar{\Omega}_v(I)$ further compresses these canonical states. We enumerate $\mathcal{O}_{v,f}$ when it contains at most 512 mapping orbits, keeping every state as a singleton. Above this threshold, each orbit observed during training remains a singleton, while all unobserved orbits for the same template form one background class. Hence

$$a \sim_{\phi_v} a' \quad \Longleftrightarrow \quad \phi_v(a) = \phi_v(a'), \tag{16}$$

which means either $a = a'$ or both states are unobserved mapping orbits of the same template in its background class. The class multiplicity is

$$m_v(\bar{a}) = \left|\phi_v^{-1}(\bar{a})\right|. \tag{17}$$

It equals one for every explicit state and the number of represented unseen mapping orbits for a background state.

We estimate a source distribution with full support using training-split counts and smoothing parameter $\eta = 1$. For template f, let n_f be its count, $n_{f,\bar{a}}$ the count assigned to compressed state $\bar{a}$, $M_{v,f} = |\mathcal{O}_{v,f}| = \sum_{\bar{a}} m_{f,\bar{a}}$ the number of concrete mapping classes, and F_v the set of compatible templates. Here n is the total training count over the compatible set, $n_{\bar{a}}$ counts an explicit state, and $m_{f,\bar{a}}$ is its multiplicity within template f. The counts and multiplicities in this paragraph are restricted to the compatible set at the fixed slot v; their slot index is suppressed. Explicit and hierarchical smoothing are

$$\bar{\mu}_v(\bar{a}) = \begin{cases} \dfrac{n_{\bar{a}} + \eta}{n + \eta|\bar{\Omega}_v(I)|}, & \text{fully explicit set,} \\ \dfrac{n_f + \eta}{n + \eta|F_v|} \dfrac{n_{f,\bar{a}} + \eta m_{f,\bar{a}}/M_{v,f}}{n_f + \eta}, & \text{hybrid set, template } f. \end{cases} \tag{18}$$

The hierarchical form assigns one template pseudocount and distributes its mapping pseudocount mass uniformly over the full set of mapping-orbit states before aggregating them into compressed states.

The terminal lift is uniform on each preimage:

$$\begin{aligned} L_v(a \mid \bar{a}, I) &= \frac{\mathbf{1}\{a \in \phi_v^{-1}(\bar{a})\}}{m_v(\bar{a})}, \\ L(A \mid \bar{A}, I) &= \prod_v L_v(a_v \mid \bar{a}_v, I), \\ p(a) &= \frac{\bar{p}(\phi_v(a))}{m_v(\phi_v(a))}. \end{aligned} \tag{19}$$

Conditional on a compressed state, the lift assigns equal probability to the canonical mapping-orbit states in its preimage. Within a background class, these states are exchangeable under the source, model score, and transition kernel. A background-class draw samples a compatible bijection uniformly, canonicalizes it under $\mathrm{Aut}(f)$, and rejects the finite set of observed singleton orbits. For fixed f, $g \circ m = m$ implies $g = \mathrm{id}$ because m is surjective. Thus the action is free, and the orbit-stabilizer theorem gives $|[m]| = |\mathrm{Aut}(f)|$ (Milne, 2020, Corollary 4.8). Uniform sampling of

bijections therefore yields a uniform distribution over accepted mapping orbits in the background preimage. The preimages of ϕ_v partition the compatible set, and a normalized $\bar{p}$ therefore induces a normalized concrete law. Hence $-\log p(a) = -\log \bar{p}(\phi_v(a)) + \log m_v(\phi_v(a))$. Because the multiplicity is fixed by the compatible-state construction, this conversion leaves the training gradient unchanged while making the loss equal to the induced concrete-state negative log likelihood.

A.5 Conditional Realization Flow

Let $\bar{A} = (\bar{a}_v)_v$ and let $\bar{\nu}_I(\bar{A})$ be the clean compressed-assignment law. The source is $\bar{\mu}_I(z) = \prod_v \bar{\mu}_v(z_v)$. We adopt the kinetic-optimal functional form of Shaul et al. (2025), evaluated at the source probability assigned to each concrete state represented by a compressed state, $\widetilde{\mu}_v(\bar{a}) = \bar{\mu}_v(\bar{a})/m_v(\bar{a})$. In the main configuration,

$$\omega_v(\bar{a}) = \arccos\sqrt{\widetilde{\mu}_v(\bar{a})}, \qquad \kappa_t(\bar{a}) = 1 - \frac{\sin^2((1-t)\omega_v(\bar{a}))}{\sin^2 \omega_v(\bar{a})}, \tag{20}$$

where the slot index v in κ_t is suppressed, with numerical endpoint handling. For $\widetilde{\mu}_v(\bar{a}) = 1$, the continuous limit is $\kappa_t(\bar{a}) = 1 - (1-t)^2$. For every $\bar{a}$, the schedule is nondecreasing and satisfies $\kappa_0(\bar{a}) = 0$ and $\kappa_1(\bar{a}) = 1$. The path in Equation 6 therefore starts at $\bar{\mu}_v$ and ends at $\delta_{\bar{a}_v}$. Over slots, $R_t(z \mid \bar{A}) = \prod_v R_t(z_v \mid \bar{a}_v)$. The interval probability $r^v_{s,t}(\bar{a}) = [\kappa_t(\bar{a}) - \kappa_s(\bar{a})]/[1 - \kappa_s(\bar{a})]$ is valid, and its bridge transports the conditional path exactly for fixed $\bar{A}$.

Direct-sampling kernel and objective. The relational predictor outputs slot conditional laws $q_{\theta_2,v}(\bar{a}_v \mid z, I, s)$. Write $q_v(\cdot \mid z, I, s) = q_{\theta_2,v}(\cdot \mid z, I, s)$, abbreviating $q_v(y \mid z, I, s)$ to $q_v(y)$ only when z, I, s are fixed. Here $z \in \bar{\Omega}(I)$ is a joint state and $x, y \in \bar{\Omega}_v(I)$ are slot states. For one slot, proposing $\bar{a}^\star_v \sim q_v(\cdot \mid z, I, s)$ and revealing it with probability $r^v_{s,t}(\bar{a}^\star_v)$ gives

$$K_v(y \mid x) = \begin{cases} q_v(y) r^v_{s,t}(y), & y \neq x, \\ \sum_{\bar{b} \in \bar{\Omega}_v(I)} q_v(\bar{b})[1 - r^v_{s,t}(\bar{b})] + q_v(x) r^v_{s,t}(x), & y = x. \end{cases} \tag{21}$$

These entries are nonnegative and sum to one. After a shared relational forward pass, the joint update draws the next slot states conditionally independently from these kernels. The relational context is recomputed at each step. Training draws one time per molecule from $0.1\delta_0 + 0.9\,\mathrm{Unif}(0,1)$. Under the uniform lift, the per-slot denoising loss in the concrete state space is $\ell_{\mathrm{real},v} = -\log q_{\theta_2,v}(\bar{a}_v \mid \bar{Z}_t, I, t) + \log m_v(\bar{a}_v)$. For a molecule with N slots, the objective is

$$\mathcal{L}_{\mathrm{real}}(\theta_2) = \mathbb{E}\left[\frac{1}{N}\sum_{v=1}^{N} \ell_{\mathrm{real},v}\right], \tag{22}$$

which we estimate by averaging the per-molecule losses over the minibatch.

The realization variants in Section 5.5 use this parameterization. Source draws $\bar{a}_v \sim \bar{\mu}_v$ without a learned update. Nodewise uses $q^{\mathrm{node}}_{\theta_2,v}(\bar{a}_v \mid z_{t,v}, I, t)$, which excludes the other current slot states. Relational uses $q_{\theta_2,v}(\bar{a}_v \mid \bar{Z}_t, I, t)$ and recomputes their shared context after every synchronous update. Source → Nodewise adds learned transport; Nodewise → Relational adds cross-slot relational coupling.

Algorithm 2 Conditional Realization Flow and assembly

Require: interface I supported by the fragment bank, grid $0 = t_0 < \cdots < t_{K_R} = 1$
1: sample $\bar{Z}_{t_0,v} \sim \bar{\mu}_v$ for every slot
2: **for** $j = 0, \ldots, K_R - 1$ **do**
3: evaluate all $q_{\theta_2,v}(\cdot \mid \bar{Z}_{t_j}, I, t_j)$ with shared relational context
4: sample slot proposals $\bar{A}^{(j)}$ and apply their bridge updates in parallel
5: **end for**
6: set $\bar{A} = \bar{Z}_1$ and sample $A \sim L(\cdot \mid \bar{A}, I)$
7: **return** $\mathrm{Assemble}(I, A)$

Oracle transport. Define the path law, its posterior, and the oracle kernel by

$$\mu_t(z) = \sum_{\bar{A}} \bar{\nu}_I(\bar{A}) R_t(z \mid \bar{A}), \qquad \pi_t(\bar{A} \mid z, I) = \frac{\bar{\nu}_I(\bar{A}) R_t(z \mid \bar{A})}{\mu_t(z)}, \tag{23}$$

$$P^{\star}_{s,t}(z, z') = \sum_{\bar{A}} \pi_s(\bar{A} \mid z, I) \prod_v B^v_{s,t}(z'_v \mid z_v, \bar{a}_v). \tag{24}$$

For fixed interface I, the identity $\mu_s P^{\star}_{s,t} = \mu_t$ follows from $\mu_s(z)\pi_s(\bar{A} \mid z, I) = \bar{\nu}_I(\bar{A}) R_s(z \mid \bar{A})$ and conditional transport for each $\bar{A}$.

Dependence gap of product updates. Independent proposals within one update produce

$$Q_{s,t}(z, z') = \prod_v \sum_{\bar{a}_v} q_v(\bar{a}_v \mid z, I, s) B^v_{s,t}(z'_v \mid z_v, \bar{a}_v). \tag{25}$$

Let $\pi_{s,v}$ be the true posterior marginal and $\pi^{\otimes}_s = \prod_v \pi_{s,v}$.

Proposition 3 (Product-posterior approximation)**.** *For fixed z and I,*

$$\mathrm{TV}(P^{\star}_{s,t}(z, \cdot), Q_{s,t}(z, \cdot)) \le \mathrm{TV}\left(\pi_s(\cdot \mid z, I), \prod_v q_v(\cdot \mid z, I, s)\right). \tag{26}$$

$$\begin{aligned} \mathrm{KL}\left(\pi_s(\cdot \mid z, I) \middle\| \prod_v q_v(\cdot \mid z, I, s)\right) &= \mathrm{TC}(\pi_s(\cdot \mid z, I)) \\ &\quad + \sum_v \mathrm{KL}\left(\pi_{s,v}(\cdot \mid z, I) \| q_v(\cdot \mid z, I, s)\right), \end{aligned} \tag{27}$$

where $\mathrm{TC}(\pi_s(\cdot \mid z, I)) = \mathrm{KL}(\pi_s(\cdot \mid z, I) \| \pi^{\otimes}_s(\cdot \mid z, I))$.

Proof. Both kernels apply the same bridge channel to different proposal laws, so data processing gives the total-variation inequality. For the KL identity, add and subtract $\sum_v \log \pi_{s,v}(\bar{a}_v)$ inside the expectation under π_s. The first term is total correlation (Watanabe, 1960) and the remaining terms are marginal KL divergences. □

Finite-grid accumulation. For a grid $0 = t_0 < \cdots < t_K = 1$, let $\widehat{\mu}_{t_0} = \mu_0$ and $\widehat{\mu}_{t_{j+1}} = \widehat{\mu}_{t_j} Q_{t_j, t_{j+1}}$. Write $\Delta_j(z) = \mathrm{TC}(\pi_{t_j}(\cdot \mid z, I)) + \sum_v \mathrm{KL}(\pi_{t_j,v}(\cdot \mid z, I) \| q_v(\cdot \mid z, I, t_j))$. Markov contraction, telescoping under the learned intermediate laws, and Pinsker's inequality (Polyanskiy & Wu, 2016) give

$$\begin{aligned} \mathrm{TV}(\widehat{\mu}_1, \bar{\nu}_I) &\le \sum_{j=0}^{K-1} \mathbb{E}_{Z \sim \widehat{\mu}_{t_j}} \left[\sqrt{\frac{1}{2}\Delta_j(Z)}\right] \\ &\le \sum_{j=0}^{K-1} \sup_z \sqrt{\frac{1}{2}\Delta_j(z)}. \end{aligned} \tag{28}$$

This bound accumulates posterior error and dependence over a fixed grid. Convergence as the grid is refined requires additional control of these terms; the bound alone gives neither monotonic improvement with more steps nor an accuracy guarantee at 16 steps. Relating training NLL to endpoint error also requires control under the learned intermediate laws.

Exact marginals and joint dependence. Consider two binary slots whose clean law puts probability $1/2$ on 00 and $1/2$ on 11. From a fully uninformative source, the one-step joint posterior equals this clean law. Both posterior marginals are uniform Bernoulli. For a single update from the source to the clean endpoint, the oracle kernel produces only 00 and 11, while the factorized kernel assigns probability $1/4$ to each of $00, 01, 10, 11$. Their total-variation distance is $1/2$. Perfect slot marginals are insufficient for exact finite-step joint transport.

A.6 TYPED ASSEMBLY

After the terminal lift samples A, typed assembly computes $\widehat{G} = \mathrm{Assemble}(I, A)$ from an interface already validated against $\mathcal{R}$. It copies the selected fragment at each slot, maps interface ports to attachment atoms, checks that the mapped endpoint types agree with the interface, and restores the specified inter-fragment bonds. Temporary port-capping valence is removed as the atom graphs are merged. RDKit sanitization then checks valence and aromaticity. With the fragment templates and mapping representatives fixed, assembly maps (I, A) deterministically to a molecular graph or the failure symbol $\perp$.

A.7 MODEL SPECIFICATION AND TRAINING

Interface Flow uses a DiGress-style graph transformer (Vignac et al., 2023) followed by a balanced latent probabilistic circuit. Conditional Realization Flow combines a typed message-passing neural network (MPNN) (Gilmer et al., 2017) on the static interface with relational blocks that apply multi-head self-attention (Vaswani et al., 2017) and typed edge messages to the current slot states. Table 5 lists the model and training settings. Parameter counts vary with the training-derived vocabularies.

Table 5: Hyperparameters of FIRM. Parameter counts are listed in the order MOSES, NPGen, GuacaMol, and ZINC250K. Time-embedding frequencies specify the number of sinusoidal frequencies; candidate score scale multiplies the scaled dot-product logits.

Name	**Value**
Interface Flow	
Parameters (millions)	14.14 / 23.84 / 23.63 / 14.60
Backbone	DiGress-style graph transformer
Encoder layers	5
Node / edge / global width	256 / 128 / 256
Feed-forward expansion	4
Attention heads	8
Random-walk length	8
Time-embedding frequencies	128
Probability head	Balanced latent tree, rank 12
Conditional Realization Flow	
Parameters (millions)	5.62 / 5.78 / 5.83 / 5.64
Backbone	Typed MPNN + relational attention
Encoder layers	4 typed local + 3 relational
Node / hidden width	192 / 384
Attention heads	8
Time-embedding frequencies	16
Candidate score scale	4
Training setup	
Effective batch size	256
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Learning rate	3×10^{-4}
Schedule	Linear warm-up, cosine decay
Weight decay	10^{-4}
Dropout	0
Precision	BF16

The backbone configuration is shared across datasets; the Block-factorized ($r = 1$) control changes only the separator rank. NPGen Interface Flow accumulates two batches of 128 to reach the effective batch size of 256.

We select checkpoints by validation NLL, except that the MOSES realization model uses the final training checkpoint. Realization validation NLL is evaluated at the source time and averaged per slot. The experiment configurations specify the selected checkpoints and dataset-specific warm-up and clipping settings.

Each flow takes 16 updates. Interface Flow uses unit temperature and the linear finite-step bridge; Realization Flow uses the source-mass-dependent schedule in Appendix A.5. Selective MILP retraction precedes realization and typed assembly. Sampling uses no steering or reranking.

A.8 Computational cost and support

For fixed encoder widths, the current Interface Flow probability head evaluates a code's log likelihood in $O(Nr^3 + N^2r|\mathcal{R}| + Nr|\mathcal{P}| + N^3r)$ operations, where r is the separator rank and $\mathcal{P}$ is the port-plan alphabet. The N^3r term arises from dense pair-to-slot aggregation. Complete generation also includes the dense encoders, compatible-state construction, selective MILP, assembly, and sanitization.

The fragment bank and endpoint-type vocabulary are constructed from the training split. Generation recombines supported templates and compatible mappings within this fixed bank. Assembly is restricted to interfaces whose realization space is supported by the bank. Failed retraction, realization, assembly, or sanitization remains in the denominator as an unsuccessful generation attempt.

B Additional Results

B.1 Dataset splits and fragment statistics

All experiments used an NVIDIA GeForce RTX 5090 GPU (32 GB) and an Intel Core Ultra 9 285K CPU. Preprocessing retains all MOSES, NPGen, and ZINC250K molecules. It excludes 3 / 1 / 1 GuacaMol molecules from train / validation / test because an exact, port-consistent fragment-to-template atom map could not be found. Table 6 lists the retained molecules, fragment templates, and typed interfaces in each split. A fragment template includes its canonical core and typed attachment sites, including their atom locations. Interfaces are identified by their complete typed graph up to slot permutation. MOSES uses the original test split for validation and the scaffold-split test set for final evaluation. NPGen and GuacaMol use their predefined splits. ZINC250K follows the fixed 234,456 / 10,000 / 5,000 source partition.

Table 6: Retained molecules and distinct fragment templates and typed interfaces in each split.

Dataset	Split	Molecules	Fragments	Interfaces
MOSES	Train	1,584,663	51,879	126,563
	Validation	176,074	16,062	42,838
	Test	176,225	13,163	33,415
NPGen	Train	526,852	217,689	70,834
	Validation	32,928	24,592	12,762
	Test	98,786	59,427	27,568
GuacaMol	Train	1,273,101	229,163	273,105
	Validation	79,567	36,500	41,570
	Test	238,705	75,491	94,396
ZINC250K	Train	234,456	37,096	74,903
	Validation	10,000	5,620	6,921
	Test	5,000	3,557	3,824

The training port-signature vocabularies contain 368, 1,184, 1,468, and 453 entries for MOSES, NPGen, GuacaMol, and ZINC250K, respectively.

The interface-input prediction experiment in Appendix C.1 further requires targets supported by the training-derived candidates. Its test sets contain 171,810 MOSES and 67,839 NPGen molecules. We exclude 4,401 MOSES and 30,919 NPGen molecules because their targets are unsupported by the fragment bank, and a further 14 and 28 molecules, respectively, because of unsupported endpoint types. These prediction-specific restrictions do not change the unconditional benchmark reference sets.

B.2 METRICS AND BASELINES

All four benchmarks evaluate unconditional distribution learning. Molecules are sampled without a property target and compared with the benchmark-specific reference sets described below. For FIRM, validity is the fraction of all attempts that yield a molecule that passes RDKit sanitization after selective retraction when required, realization, and assembly. The denominator includes attempts that fail during retraction, realization, assembly, or sanitization. The other metrics use the preprocessing and denominators of the corresponding benchmark. Assembly outputs are canonical isomeric SMILES, retaining the stereochemical and isotope annotations present in the assembled molecule. Mapping-orbit equivalence in Appendix A.1 concerns the choice of attachment mappings. For NPGen uniqueness, novelty, and the exact-training-match analysis, generated SMILES are canonicalized with isomeric output enabled; training identities use the strings in the official training CSV without recanonicalization, following the benchmark protocol.

MOSES. We use the MOSES scaffold-split test set and, following the FragFM comparison protocol, make 25,000 generation attempts per run (Polykovskiy et al., 2020; Lee et al., 2026). In addition to validity, uniqueness, novelty, and the medicinal-chemistry filter rate, MOSES compares generated and reference molecules at three levels. FCD compares ChemNet feature distributions (Preuer et al., 2018). Following MOSES, SNN averages nearest-reference Tanimoto similarities between Morgan fingerprints, and Scaf compares scaffold-frequency vectors (Polykovskiy et al., 2020). The underlying representations are extended-connectivity fingerprints (Rogers & Hahn, 2010) and Bemis–Murcko molecular frameworks (Bemis & Murcko, 1996), respectively. Lower FCD is better; larger values are better for the remaining scores. For FIRM, uniqueness is computed over all 25,000 valid outputs per run, rather than the first 10,000; its three-run mean and sample standard deviation are $99.888\% \pm 0.038\%$.

NPGen. The natural-product benchmark follows the protocol introduced with FragFM and uses 30,000 generation attempts per run (Lee et al., 2026). Validity measures the fraction of successful attempts, uniqueness is the fraction of distinct molecules among valid outputs, and novelty is the fraction of these distinct molecules absent from training. FCD compares ChemNet feature distributions. Natural-product fidelity is evaluated through KL divergences for NP score and for the pathway, superclass, and class outputs of NPClassifier (Ertl et al., 2008; Kim et al., 2021). Lower values indicate smaller differences from the reference property and class distributions.

GuacaMol. We evaluate the GuacaMol distribution-learning benchmark (Brown et al., 2019), following FragFM in using 10,000 generated molecules per run and reporting joint validity, uniqueness, and novelty rates (Lee et al., 2026). KL and FCD are computed with the official GuacaMol v1 evaluator (GuacaMol 0.5.5). KL and FCD use the official benchmark's 10,000-molecule reference subsets sampled from the original GuacaMol v1 ChEMBL training set, not the held-out test split. The table reports validity, the valid-and-unique rate, and the valid-unique-novel rate, followed by normalized KL and FCD scores. All five quantities are shown on a 0–100 scale, with larger values indicating closer distributional similarity or a higher success rate. For FIRM, V.U. and V.U.N. are the numbers of distinct valid molecules and distinct valid molecules absent from the training set, respectively, each divided by the total number of generation attempts. Molecular identities follow the official GuacaMol canonicalization, with stereochemistry ignored. Unlike the FCD distances on the other three benchmarks, the GuacaMol FCD score increases with better distribution matching.

ZINC250K. ZINC250K contains 249,456 molecules selected from the ZINC database (Irwin & Shoichet, 2005), as described by FragFM (Lee et al., 2026). Following the FragFM comparison protocol, we generate 25,000 molecules per sampling seed. Our fixed training, validation, and test partition is specified above. Validity measures the fraction of successful generation attempts. NSPDK compares local graph-structure distributions through the neighborhood subgraph pairwise-distance kernel, whereas FCD compares ChemNet features (Costa & De Grave, 2010; Preuer et al., 2018). Both distances use the complete 5,000-molecule test set as the reference; smaller values are better.

Unless stated otherwise, FIRM uses 16 updates in each flow, unit temperature for Interface Flow, and no realization steering or reranking. The fragment bank is constructed from the training split. MOSES, NPGen, GuacaMol, and ZINC250K results are means with sample standard deviations over three independently trained runs.

Baseline scope. We retain FragFM's separation by generation strategy and representation. The autoregressive group includes sequential atom-level graph construction with GraphINVENT, GraphAF, and GraphDF, together with the fragment-level JT-VAE, HierVAE, and SAFE-GPT models (Mercado et al., 2021; Shi et al., 2020; Luo et al., 2021; Jin et al., 2018; 2020; Noutahi et al., 2024). The fragment models differ in how they carry connectivity: junction trees, hierarchical motifs, or a connected fragment sequence.

MolGrow and MolHF provide hierarchical normalizing-flow baselines. MolGrow constructs molecular graphs through recursive node splitting (Kuznetsov & Polykovskiy, 2021). FragFM lists it as Fragment (Graph) in its representation taxonomy; its construction uses graph splits rather than a predefined vocabulary of chemical fragments. GDSS, GSDM, GruM, SwinGNN, DiGress, DisCo, and Cometh represent diffusion-based graph generation, while DeFoG and GGFlow use flow matching (Kuznetsov & Polykovskiy, 2021; Zhu et al., 2023; Jo et al., 2022; Luo et al., 2024; Jo et al., 2024; Yan et al., 2024; Vignac et al., 2023; Xu et al., 2024; Siraudin et al., 2025; Qin et al., 2025; Hou et al., 2026). GuacaMol also includes MCTS and NAGVAE as search-based and latent-variable baselines (Jensen, 2019; Kwon et al., 2020). FragFM is the closest comparison to FIRM: it generates a fragment graph with discrete flow matching and then recovers atom-level connectivity through candidate edge scoring and matching (Lee et al., 2026). Baseline values and representation labels, including the GuacaMol MCTS and NAGVAE results, are reproduced from FragFM's comparison tables (Lee et al., 2026).

MiCaM provides a distribution-learning comparison on GuacaMol. Its published FCD and KL scores are 73.1 and 98.9 on our 0–100 scale, compared with 86.67 and 97.56 for FIRM (Table 9) (Geng et al., 2023). In this cross-study comparison, FIRM scores higher on FCD and MiCaM scores higher on KL. MiCaM reports uniqueness and novelty as separate rates, whereas our GuacaMol table reports joint valid-unique and valid-unique-novel rates. We compare the published FCD and KL scores for these two methods. On ZINC, MiCaM reports normalized GuacaMol scores, while our results use raw FCD and NSPDK distances.

MAGNet trains and evaluates its main distribution-learning model on ZINC using GuacaMol and MOSES evaluation tools and $10{,}000$ generated valid molecules (Hetzel et al., 2025). Its ZINC reference, valid-sample budget, and distance metrics differ from ours. Neither MiCaM nor MAGNet reports an NPGen experiment, which would require learning motif or scaffold resources from the NPGen training split.

Grammar-based methods cover molecular optimization and distribution-learning generation. MHG reports molecular optimization scores under specified oracle-query budgets, for goal-directed generation (Kajino, 2019). DEG evaluates low-data generation on acrylate, chain-extender, and isocyanate datasets, emphasizing class membership and synthesizability (Guo et al., 2022). HGR reports generation results on QM9, ZINC250K, MOSES, and GuacaMol, and separately introduces the ring-enriched RingDiv dataset (Huang et al., 2026).

B.3 Variation across runs

Table 7: MOSES results using 25,000 generation attempts per run. Entries are mean $\pm$ sample standard deviation over three independently trained runs; rates are percentages.

Model	Valid↑	Unique↑	Novel↑	Filters↑	FCD↓	SNN↑	Scaf↑
FIRM	100.0 ± 0.0	99.89 ± 0.04	88.6 ± 0.5	99.4 ± 0.0	0.537 ± 0.025	0.551 ± 0.001	13.4 ± 1.8

On NPGen, the scale and ordering of the reported divergences are stable across the three runs. Novelty remains lower than the value reported for FragFM in every NPGen run, so the tradeoff between distributional similarity and novelty persists beyond one training seed.

Table 8: NPGen results using 30,000 generation attempts per run. Entries are mean ± sample standard deviation over three independently trained runs; validity, uniqueness, and novelty are percentages.

Model	Val.↑	Unique↑	Novel↑	NP Score KL Div.↓	NP Class KL Div.↓ Pathway	Superclass	Class	FCD↓
FIRM	100.0 ± 0.0	99.0 ± 0.0	88.4 ± 0.4	0.0109 ± 0.0012	0.0094 ± 0.0028	0.0915 ± 0.0048	0.2878 ± 0.0040	0.3191 ± 0.0219

B.4 Additional benchmark datasets

Table 9: Molecular generation on GuacaMol using 10,000 generated molecules per run. FIRM results are reported as mean ± sample standard deviation over three independently trained runs; baseline values are reproduced from FragFM. Bold and underlining indicate the best and second-best results among the compared models. Blue denotes non-autoregressive baselines.

Model	Rep. level	Val.↑	V.U.↑	V.U.N.↑	KL score↑	FCD score↑
Training set	–	100.0	100.0	–	99.9	92.8
MCTS (Jensen, 2019)	Atom	**100.0**	**100.0**	95.4	82.2	1.5
NAGVAE (Kwon et al., 2020)	Atom	92.9	88.7	88.7	38.4	0.9
DiGress (Vignac et al., 2023)	Atom	85.2	85.2	85.1	92.9	68.0
DisCo (Xu et al., 2024)	Atom	86.6	86.6	86.5	92.6	59.7
Cometh (Siraudin et al., 2025)	Atom	94.4	94.4	93.5	94.1	67.4
Cometh-PC (Siraudin et al., 2025)	Atom	98.9	98.9	97.6	96.7	72.7
DeFoG (Qin et al., 2025)	Atom	99.0	99.0	**97.9**	**97.7**	73.8
FragFM (Lee et al., 2026)	Fragment	99.7	99.3	95.0	97.4	85.8
FIRM (ours)	Fragment	**100.00 ± 0.00**	99.98 ± 0.01	94.50 ± 0.27	97.56 ± 0.31	**86.67 ± 0.09**

On GuacaMol, FIRM achieves the highest FCD score among the compared methods and matches MCTS in validity. Among the models in Table 9, its KL score is second to DeFoG. Relative to FragFM, the FCD score increases from 85.8 to 86.67, while the valid-unique-novel rate decreases from 95.0 to 94.50. Both quantities have small standard deviations across the three runs.

Table 10: ZINC250K results using 25,000 generation attempts per run. FIRM reports means and sample standard deviations over three independently trained runs (seeds 42, 43, and 44); baseline values are reproduced from FragFM. Bold and underlining indicate the best and second-best results among the compared models. Red/blue denote autoregressive/non-autoregressive baselines.

Model	Rep. level	Valid↑	NSPDK↓	FCD↓
Training set	–	–	0.0001	0.062
GraphAF (Shi et al., 2020)	Atom (Graph)	67.92	0.0432	16.128
GraphDF (Luo et al., 2021)	Atom (Graph)	89.72	0.1737	33.899
MolHF (Zhu et al., 2023)	Atom (Graph)	94.75	0.0709	22.230
GDSS (Jo et al., 2022)	Atom (Graph)	97.12	0.0192	14.032
GSDM (Luo et al., 2024)	Atom (Graph)	92.57	0.0168	12.435
GruM (Jo et al., 2024)	Atom (Graph)	98.32	0.0023	2.235
SwinGNN (Yan et al., 2024)	Atom (Graph)	86.16	0.0047	4.398
DiGress (Vignac et al., 2023)	Atom (Graph)	94.98	0.0021	3.482
GGFlow (Hou et al., 2026)	Atom (Graph)	99.63	0.0010	1.455
FragFM (Lee et al., 2026)	Fragment (Graph)	99.81	**0.0002**	0.630
FIRM (ours)	Fragment (Graph)	**100.00 ± 0.00**	0.00026 ± 0.00003	**0.461 ± 0.014**

FIRM has the highest validity and the lowest FCD among the compared methods on ZINC250K. Its FCD is 0.461, compared with 0.630 for FragFM; FragFM remains better on NSPDK, 0.0002 versus 0.00026. FIRM more closely matches ChemNet feature distributions, while FragFM has a slightly smaller distance in local graph structure.

B.5 Fragmentation and realization ablations

Fragmentation scheme. The main experiments use BRICS decomposition. We also train FIRM with an atom-preserving adaptation of RECAP, denoted RECAP-AP (Lewell et al., 1998). RDKit's standard RECAP decomposition recursively applies reaction rules to produce a fragment hierarchy; some rules omit atoms from the products, such as the oxygen in an ether linkage. RECAP-AP uses these reaction rules to identify bonds to cut in the original molecular graph and retains every original atom in the resulting fragments. Thus, the ether oxygen remains among the fragments, allowing assembly to reconstruct the original linkage. The adaptation changes how the fragments are extracted while retaining the RECAP reaction-center patterns. The model architecture, optimization budget, sampler, and MOSES evaluator follow the same settings as the BRICS experiments.

Table 11: Fragmentation ablation on MOSES. Values are three-run means with 25,000 attempts per run; rates are percentages and Scaf is scaled by 100.

Fragmentation	Valid↑	Novel↑	Filters↑	FCD↓	SNN↑	Scaf↑
BRICS	100.0	88.6	99.4	0.537	0.551	13.4
RECAP-AP	100.0	84.8	99.5	0.507	0.564	13.4

FIRM achieves full validity with selective interface correction under both fragmentation schemes. RECAP-AP has lower FCD and higher SNN, while filter rate and scaffold similarity remain close. Its novelty is lower. The fragmentation choice mainly shifts feature-level fidelity and novelty.

Realization transport. Figure 5 complements the aggregate Realization Flow results in Table 4 with four property marginals. The Nodewise distributions are generally closer to the reference than Source. The larger shift from Source to Nodewise indicates that learned conditional transport accounts for most of the improvement over the source distribution. The Relational distributions also closely track the reference, alongside the further FCD improvement from 0.638 to 0.537.

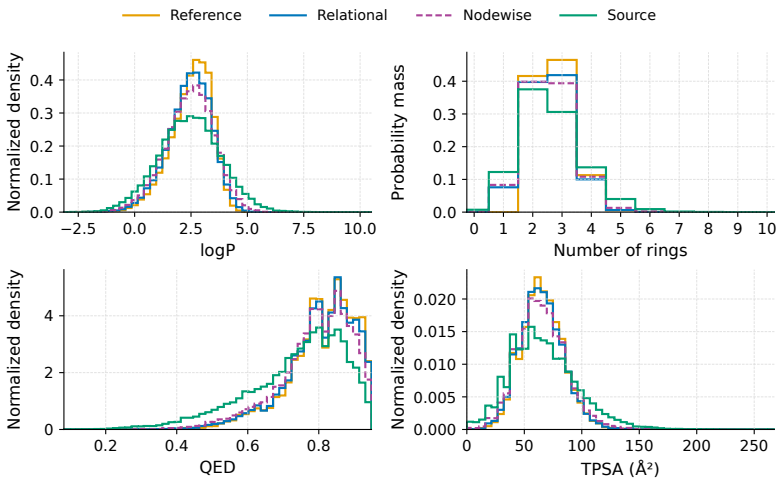


Figure 5: Molecular properties under learned realization on MOSES. Each group contains 25,000 molecules. Generated groups use sampling seed 42 and the same Interface Flow samples; Nodewise and Relational use their seed-42 training runs. The reference is a deterministic subset of distinct scaffold-test molecules. LogP, QED, and TPSA use shared density bins; ring counts show probability mass. Learned transport brings the compatible source closer to the reference.

C Additional Analyses

C.1 Interface inputs and realization prediction

We compare realization predictors on held-out molecules with fixed compatible candidates, complementing the entropy estimates in Table 3.

All four predictors share the structural descriptor, time, target-slot signature and current state, identical compatible candidates, a descriptor MLP, and a target-slot embedding. Figure 6 plots all four views relative to Local ports; the gaps between successive curves show the gains from other-slot signatures, typed connections, and other-slot current states (Table 12). Local ports bypasses cross-slot attention and edge messages; Signature composition uses attention without typed connections. Input restrictions are identical in training and evaluation.

Table 12: Nested inputs for target-slot realization prediction. A check indicates that the input is available. Parameter counts are matched across views: 5.658 million on MOSES and 5.814 million on NPGen. Other settings follow Conditional Realization Flow (Table 5).

Predictor view	Other-slot signatures	Typed connections	Other-slot current states
Local ports	–	–	–
Signature composition	✓	–	–
Full interface	✓	✓	–
Full interface + states	✓	✓	✓

Using seed 42, we train MOSES predictors for 60,000 updates and NPGen predictors for 100,000. At each $t \in \{0, 0.25, 0.5, 0.75, 0.9\}$, we average losses over target slots and three shared noise realizations, then equally over molecules. Paired 95% bootstrap intervals resample these molecule averages with the trained models fixed.

For a compressed candidate class of multiplicity m, concrete-state NLL is $-\log q + \log m$.

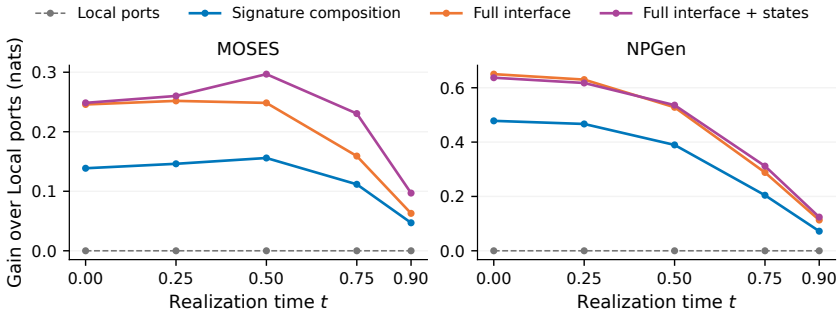


Figure 6: Predictive value of interface information. All four curves show the reduction in realization NLL relative to Local ports (Table 12); its curve is zero by definition. Vertical gaps between successive curves give the gains from other-slot signatures, typed connections, and other-slot current states.

Typed connections improve NLL beyond signature composition at every measured time (Figure 6). At $t = 0.75$, adding the other slots' current states reduces NLL by 0.0714 nats on MOSES and 0.0234 on NPGen. At $t = 0$, the corresponding gain is 0.0027 and -0.0131, respectively; the NPGen gain remains negative at $t = 0.25$ (-0.0124). The benefit of other slot states peaks later in the path.

C.2 Reveal order and model components

Reveal-order specialization under matched masks. Joint states contain a port plan (signature and port roles) and compressed realization per slot, plus a connection state (no edge, or tree/extra status and oriented rule) per unordered slot pair. All orders use a five-layer node-edge-global graph transformer with separate port-plan, connection, and realization heads. The realization head predicts a signature, then a state from its complete training-derived bank.

Shared random priorities order coordinates within each family for IF and RF, and across families for MIX. IF reveals interface coordinates first; RF reveals realizations first. After update k of 16, all orders have revealed the nearest integer to $k/16$ times the coordinate count. Revealed values are sampled from the current prediction and remain fixed.

Each dataset uses three training runs per order, with seeds 42, 43, and 44. Orders share initialization, data and masking randomness within each seed, with batch size 256 and AdamW. The training loss averages NLL over hidden coordinates within each molecule, then over molecules. Prediction and generation use fixed-step checkpoints: 100,000 steps for MOSES and 35,000 for NPGen.

Prediction queries share visible values and masks; the neutral query randomly reveals approximately half of each coordinate family.

We use 172,798 MOSES and 22,638 NPGen validation molecules, matched by original index across all training seeds and reveal orders, and selected separately from the interface-input experiment. Probe masks use seed 42, keeping visible values and masks identical across models. Metrics are computed separately for each trained model, then summarized as the mean and sample standard deviation over three training runs ($n = 3$).

For evaluation, Table 13 pools NLL and accuracy over hidden coordinates across molecules and reports the fraction of molecules with fully compatible slot assignments. The $I \to A$ query predicts compressed realization classes; $A \to I$ predicts interface coordinates; the neutral query pools both. A class with m mapping orbits has $\mathrm{NLL}_{\mathrm{orbit}} = \mathrm{NLL}_{\mathrm{class}} + \log m$ under the uniform lift, so paired model differences are unchanged.

NLL rankings agree across all three training seeds: IF performs best for interface-to-realization prediction, RF for inverse prediction, and MIX under neutral masking.

Table 13: Matched-mask evaluation: mean ± sample SD over three training seeds. Accuracy and fully compatible molecule rates are percentages; NLL is per hidden coordinate. Realization metrics use compressed candidate classes. Nonzero SDs below display precision are bounded.

		MOSES			NPGen		
Query	Metric	IF	MIX	RF	IF	MIX	RF
$I \to A$	NLL ↓	2.0286 ± 0.0002	2.0390 ± 0.0005	4.4982 ± 0.0111	2.3412 ± 0.0018	2.3568 ± 0.0008	5.2067 ± 0.0504
	Acc. (%) ↑	56.29 ± 0.03	56.08 ± 0.02	19.42 ± 0.43	55.65 ± 0.02	55.57 ± 0.03	19.99 ± 0.26
	Compat. (%) ↑	99.98 ± <0.01	99.94 ± 0.01	0.43 ± 0.51	99.63 ± 0.03	98.60 ± 0.21	4.48 ± 3.38
$A \to I$	NLL ↓	1.6944 ± 0.0417	0.0171 ± 0.0004	0.0081 ± 0.0001	1.3142 ± 0.0071	0.0344 ± 0.0006	0.0154 ± 0.0012
	Acc. (%) ↑	54.86 ± 0.44	99.42 ± 0.02	99.72 ± <0.01	67.81 ± 0.16	98.83 ± 0.01	99.50 ± 0.01
	Compat. (%) ↑	0.08 ± 0.01	99.87 ± 0.01	99.97 ± 0.01	7.44 ± <0.01	98.02 ± 0.11	99.43 ± 0.09
Neutral	NLL ↓	0.7404 ± 0.0126	0.5732 ± 0.0003	1.0327 ± 0.0075	0.6047 ± 0.0047	0.5012 ± 0.0012	0.8736 ± 0.0061
	Acc. (%) ↑	81.68 ± 0.26	86.70 ± <0.01	76.30 ± 0.18	86.79 ± 0.16	89.93 ± 0.02	82.28 ± 0.14
	Compat. (%) ↑	71.96 ± 0.86	93.22 ± 0.14	42.75 ± 0.65	73.54 ± 0.80	90.70 ± 0.64	42.37 ± 0.57

The compatibility rates sharpen the directional contrast beyond NLL: IF recovers compatible assignments when the interface is visible, while RF does so when realizations are visible. Under neutral masking, MIX has the lowest average prediction loss across both coordinate families. These fixed-mask probes diagnose specialization; the full-generation comparison below also includes errors accumulated during sampling.

Controlled reveal-order generation without correction. Each trained model is evaluated with its matching sampling seed: $42 \to 42$, $43 \to 43$, or $44 \to 44$, without correction. Each run uses 25,000 MOSES or 30,000 NPGen attempts per order. Within each run, orders share generation seed,

sample index, fragment count, cycle rank, sampling-batch seed, and model-call budget. Success requires interface feasibility, realization compatibility, assembly, and RDKit sanitization. Metrics and paired order differences are computed per run, then summarized by their mean and sample standard deviation; this variation includes both training and sampling randomness.

Table 14: First-failure categories and success without correction, as percentages of all attempts: mean ± sample SD ($n = 3$). Assembly-stage failures are zero in every run. Each order totals 75,000 MOSES or 90,000 NPGen attempts.

Dataset	Order	Interface failure	Incompatibility	Success
MOSES	IF	19.77 ± 0.26	0.16 ± 0.02	80.07 ± 0.25
	MIX	21.82 ± 0.25	1.36 ± 0.13	76.82 ± 0.36
	RF	23.43 ± 0.07	0.62 ± 0.10	75.94 ± 0.16
NPGen	IF	34.20 ± 1.29	0.13 ± 0.01	65.67 ± 1.28
	MIX	35.45 ± 0.85	1.55 ± 0.10	63.00 ± 0.76
	RF	41.25 ± 0.55	0.74 ± 0.12	58.01 ± 0.65

The mean 3.25-percentage-point IF–MIX success gap on MOSES comprises 2.05 points from fewer interface failures and 1.20 from fewer realization incompatibilities. On NPGen, the mean gap is 2.67 points, with corresponding reductions of 1.25 and 1.42 points. No assembly-stage failures occur in the selected runs.

Table 15: Generation without correction over three matched training/sampling seeds ($n = 3$). Entries in both panels show mean and sample SD within each cell. Success, uniqueness, novelty and filters are percentages; MOSES uniqueness is Unique@10k. Distributional metrics are averaged across runs. Lower FCD and divergences are better; other metrics are higher-is-better.

(a) MOSES

Order	Success%	Unique%	Novel%	Filters%	FCD
IF	80.07 ± 0.25	98.71 ± 0.09	96.35 ± 0.07	98.21 ± 0.11	0.992 ± 0.011
MIX	76.82 ± 0.36	99.16 ± 0.07	96.00 ± 0.19	98.40 ± 0.08	0.858 ± 0.027
RF	75.94 ± 0.16	99.17 ± 0.11	95.51 ± 0.22	98.51 ± 0.03	0.821 ± 0.005

(b) NPGen

Order	Success%	Unique%	Novel%	NP KLD	Path.	Super.	Class	FCD
IF	65.67 ± 1.28	95.29 ± 0.20	83.93 ± 0.25	0.051 ± 0.007	0.037 ± 0.001	0.283 ± 0.021	0.863 ± 0.054	2.111 ± 0.064
MIX	63.00 ± 0.76	95.75 ± 0.33	83.20 ± 0.20	0.047 ± 0.009	0.034 ± 0.003	0.274 ± 0.015	0.843 ± 0.043	2.166 ± 0.050
RF	58.01 ± 0.65	96.16 ± 0.27	81.92 ± 0.45	0.036 ± 0.004	0.038 ± 0.003	0.294 ± 0.015	0.856 ± 0.030	2.468 ± 0.106

The distributional metrics favor different orders (Table 15): RF has lower FCD and a higher filter rate on MOSES, while RF or MIX minimizes several NPGen divergences. IF has the highest generation success on both datasets.

Interface model. FIRM's interface model is a balanced latent probabilistic circuit with separator rank 12. The Star and block-factorized comparisons in Table 4 change the dependency topology or separator rank while keeping the encoder inputs and canonical code alphabet fixed.

For variant h and $n_{\mathrm{val}} = 10{,}000$ interfaces sampled without replacement from the validation split, let d_k count the scored clean-code fields in canonical code $Y^{(k)}$. Draw $Z_t^{(k)} \sim R_t(\cdot \mid Y^{(k)})$, and write $q_{\theta_1}^{(h)}$ for the variant's clean-code predictor. The clean-code denoising NLL per scored field at

path time t averages over this corruption:

$$\ell_h(t) = -\frac{1}{n_{\text{val}}} \sum_{k=1}^{n_{\text{val}}} \frac{1}{d_k} \mathbb{E}_{Z_t^{(k)} \sim R_t(\cdot \mid Y^{(k)})} \left[\log q_{\theta_1}^{(h)} \left(Y^{(k)} \mid Z_t^{(k)}, t, c^{(k)} \right) \right]. \tag{29}$$

Here $d_k = N_k + \binom{N_k}{2} + |E_k|$ counts one complete port plan per slot, one pair kind per unordered slot pair, and one rule per connection. The fixed no-edge rule contributes no separate clean-code field. This likelihood-normalization count differs from the number of coordinates a in Equation 4: corruption evolves signature and port-role entries within each plan, and stored no-edge rule tokens, separately. At $t = 0$, the clean-code denoising NLL per scored field is $\ell_h(0)$. For intervals b that partition $[0, 1]$, the path-integrated NLL is $\sum_b |b| \, \bar{\ell}_{h,b}$, where $\bar{\ell}_{h,b}$ is the conditional mean in interval b. Figure 4 reports these means for seven nonzero-time intervals.

Realization model. Source draws compatible realizations from smoothed training counts without learned transport. Nodewise learns their selection from the interface and target-slot state; Relational also uses the other slots' states. The learned variants use training seeds 42, 43, and 44, paired with Interface Flow samples from the same seed. Figure 4 reports means and sample standard deviations. The reduction in FCD from Source to Nodewise shows the contribution of learned selection, with a further improvement from relational context.

C.3 Interface structure and direct feasibility

Interface multiplicity within coarse-state classes. We analyze 1,584,663 MOSES and 526,852 NPGen molecules. As in Section 5.3, S_0 fixes the anonymous untyped fragment topology, N, and cycle rank; S_1 also fixes fragment templates and their typed attachment sites, leaving their assignment to connections unspecified. A coarse-state class is ambiguous if it contains multiple interfaces. Its effective interface multiplicity is $\exp(\widehat{H}(I \mid S))$, using unsmoothed empirical conditional entropy.

Table 16: Empirical interface ambiguity within S_0 and S_1 coarse-state classes. The effective number of interfaces is $\exp(\widehat{H}(I \mid S))$; information is $\widehat{I}(A; I \mid S)$.

Dataset	Coarse state	Molecules	Ambiguous fraction	Effective no. of interfaces	$\widehat{I}(A; I \mid S)$
MOSES	S_0	1,584,663	0.993250	1469.166	10.517 bits
MOSES	S_1	1,584,663	0.003212	1.002220	0.000269 bits
NPGen	S_0	526,852	0.902145	99.701	6.639 bits
NPGen	S_1	526,852	0.000757	1.000520	0.000214 bits

Fixing fragment templates and typed attachment sites reduces effective interface multiplicity from 1469.17 to 1.0022 on MOSES and from 99.70 to 1.0005 on NPGen (Table 16). For the training molecules, little ambiguity about I remains once both are fixed. The remaining ambiguous cases show that template identity alone cannot recover every interface, and these measurements do not guarantee unique recovery for new template combinations. Table 3 instead measures uncertainty in realization A after fixing I and S_0.

After fixing S_1, the conditional-information entries in Table 16 show that observing A adds little information about I. Table 3 asks the reverse question: with only S_0 fixed, how much does I reduce uncertainty about A? We use S_1 for this analysis only, not as an input to interface generation.

Long-tailed interfaces and novel combinations. The MOSES training split contains 126,563 complete interfaces, 108 untyped fragment graphs, and 368 local attachment patterns (port signatures). The corresponding NPGen counts are 70,834, 4,267, and 1,184.

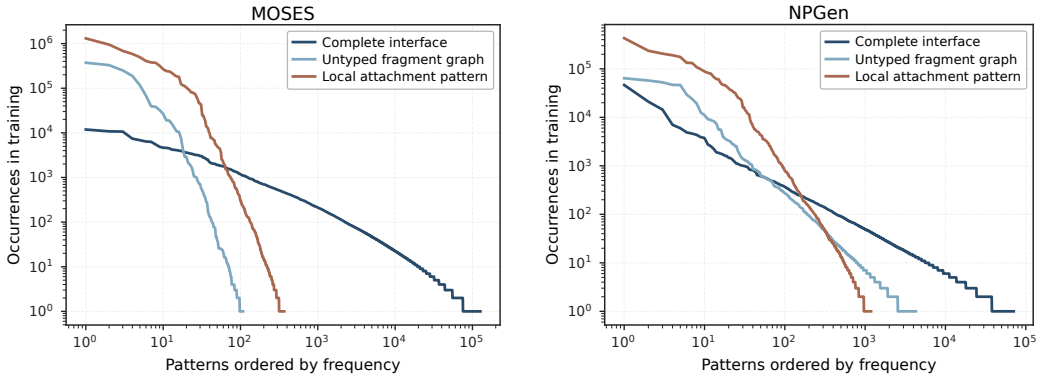


Figure 7: Training-set frequency curves for complete typed interfaces, untyped fragment graphs, and local attachment patterns. Patterns are ordered independently from most to least frequent.

Complete interfaces have a longer frequency tail than fragment graphs or local attachment patterns: singletons account for 40.7% of distinct MOSES interfaces and 46.8% of NPGen interfaces. Many complete interfaces combine the same local patterns.

Unseen complete interfaces occur in 6.4% of MOSES and 6.2% of NPGen test molecules, compared with 14.0% and 33.1% of generated molecules (Figure 8). These generated interfaces combine local attachment patterns observed in training; Section C.4 examines retraction's contribution.

Figure 9 groups generated molecules by their final interface's training frequency and reports the fraction of initial proposals requiring correction.

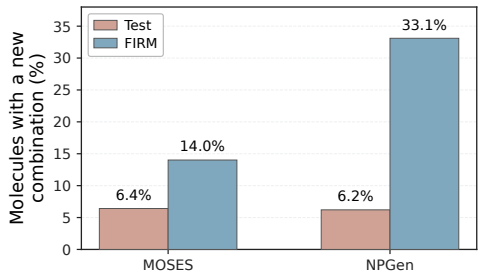


Figure 8: Test and generated fractions with unseen complete interfaces assembled from local attachment patterns observed in training.

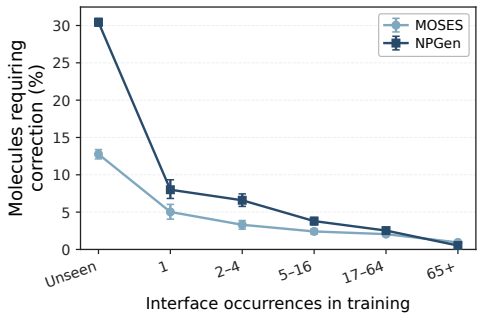


Figure 9: Correction rates by final interface training frequency. Error bars are molecule-level bootstrap 95% intervals.

Correction and compatible realization support. Correction is most frequent for rare final interfaces. From unseen interfaces to those observed at least 65 times, the rate falls from 12.77% to 0.94% on MOSES and from 30.43% to 0.52% on NPGen.

Figure 10 further groups the results by the final interface's compatible-space size $|\Omega(I)| = \prod_v |\Omega_v(I)|$, using test-set quartiles.

For unseen MOSES interfaces, pre-retraction feasibility rises from 73.2% in the smallest compatible-space quartile to 91.0% in the largest. On NPGen, the middle quartiles reach 77.5% and 78.0%, but the largest falls to 63.4%. Feasibility varies more consistently with interface frequency: common interfaces approach full feasibility in every size quartile on both datasets.

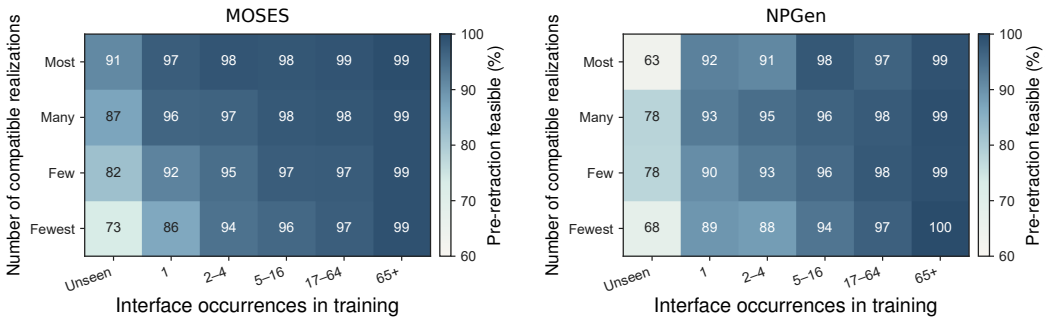


Figure 10: Pre-retraction interface feasibility across complete-interface frequency and compatible-space size. Cell values are percentages.

C.4 Compositional generation and selective retraction

We compare FIRM before and after selective retraction across three trained runs, each with 25,000 MOSES or 30,000 NPGen proposals. Feasible proposals retain their interface and realization, with code ordering canonicalized as needed; infeasible proposals fail without retraction and enter realization after correction. Unlike the joint models in Section C.2, this comparison uses FIRM's two-stage sampler.

Here, an unseen-interface generation combines an interface absent from training with previously observed node signatures, connection rules, and selected realization templates.

Table 17: Paired outcomes (% of all attempts; mean $\pm$ sample SD over three trained runs). The unseen-interface total is partitioned into direct and retraction-added outcomes.

Successful assembly	MOSES	NPGen
All interfaces: direct	96.87 ± 0.09	88.84 ± 0.46
All interfaces: added by retraction	3.13 ± 0.09	11.16 ± 0.46
Unseen interfaces: total	14.01 ± 0.70	33.10 ± 0.39
Direct (no retraction)	12.21 ± 0.56	23.03 ± 0.69
Added by retraction	1.80 ± 0.16	10.07 ± 0.31

Direct assembly accounts for most unseen-interface outcomes; retraction adds the remainder, with a larger contribution on NPGen.

Across runs, 10,511 MOSES and 29,788 NPGen generations combine observed local components into unseen interfaces, spanning 10,222 and 28,956 distinct interfaces. Requiring each signature, rule, and selected template to occur in at least 10 training molecules leaves 9,831 and 12,201 generations; requiring 100 occurrences leaves 8,660 and 6,302. Recombination includes frequent components, with NPGen counts falling more sharply as the threshold rises.

Retraction corrects 2,345 of 75,000 MOSES proposals and 10,046 of 90,000 NPGen proposals; all then assemble successfully. Figure 11 counts edits to signature, port-role, pair-kind, and rule entries, counting pair-kind and rule edits separately per unordered slot pair. Median counts are 1 and 2; 89.38% and 77.51% of corrected proposals change at most two coordinates, with maxima of 6 and 67. Signature changes occur in 1,497 MOSES and 7,401 NPGen proposals.

Larger interfaces are less often directly feasible (Figure 11, right). In the seed-42 NPGen step-budget experiment of Section C.8, 16 Interface Flow updates require correction for 1.56% of interfaces with 1 to 4 nodes and 61.73% of those with at least 19 nodes. Most corrections change only a few interface coordinates, although a small tail requires more extensive changes, especially on NPGen. Direct assembly succeeds for most sampled interfaces, while retraction handles an increasing share at larger sizes.

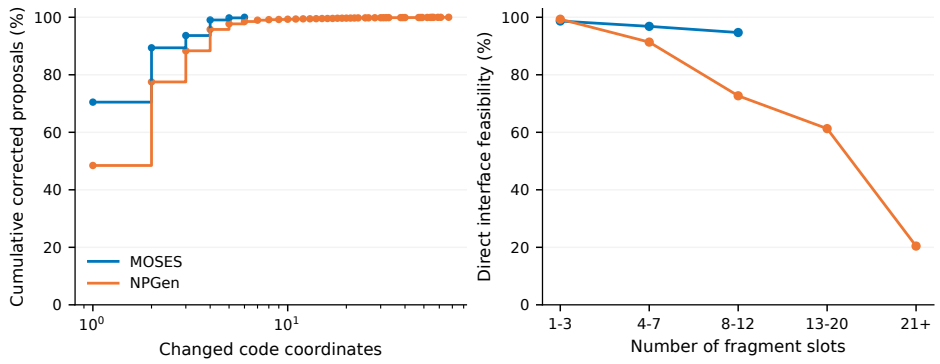


Figure 11: Extent of retraction and dependence on interface size. Left: empirical cumulative distributions of coordinate edit counts among corrected proposals. Right: direct feasibility within fragment-count strata, pooled across three trained runs.

C.5 Diversity within compatible spaces

Diversity in fingerprint space. We measure variation among molecules sharing an interface using 512-bit radius-2 Morgan fingerprints (Rogers & Hahn, 2010). For Figure 12, we select eligible interfaces separately for seeds 42, 43, and 44 and sample ten distinct realizations per interface. The covariance decomposition gives each interface equal weight.

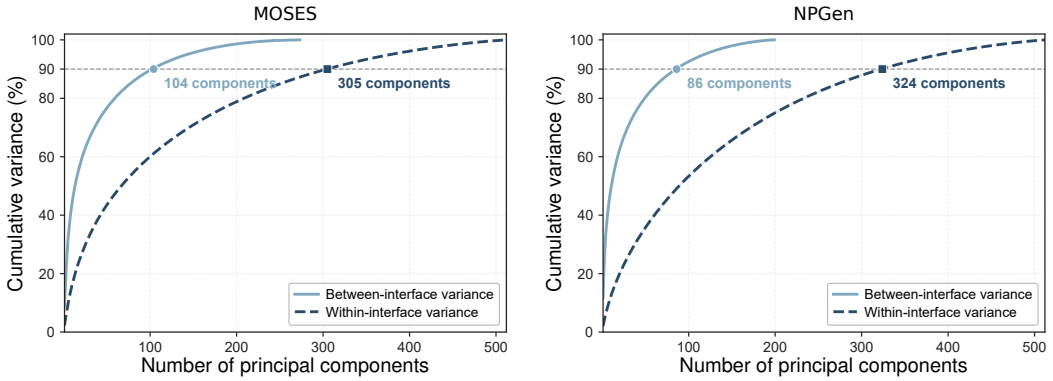


Figure 12: Cumulative fingerprint variance for differences across interfaces and among realizations of the same interface on MOSES (left) and NPGen (right). Curves are means over three generation seeds, and ribbons span the seedwise minimum and maximum. Annotations give the median across runs of the number of principal components explaining 90% of each variance component.

Between-interface variation is concentrated in fewer principal components. Explaining 90% requires 99 to 104 components between interfaces versus 304 to 306 within interfaces on MOSES, and 84 to 87 versus 319 to 326 on NPGen. Between-interface differences account for 25.5% to 26.7% and 28.1% to 29.5% of total variance, respectively; the pooled within-interface variance is distributed across more principal components. Fixing an interface thus leaves substantial variation in the realizations, spread over many fingerprint directions. The interface constrains how fragments can connect, while compatible fragment choices retain chemical variation within that structure.

We also compare the participation-ratio dimension of molecules generated under one interface with that of an equal-size random molecular group. For covariance eigenvalues λ_j, the participation-ratio dimension is $d_{\mathrm{PR}} = (\sum_j \lambda_j)^2 / \sum_j \lambda_j^2$. We retain groups with at least 10 distinct fingerprints, sample up to 32 per group, and compare against equally sized groups drawn from the pooled generated

fingerprint set. The reported ratios are averaged over three matched draws. The analysis includes 787 MOSES interface-seed groups and 592 NPGen groups.

The median dimension within an interface is 84.3% of that in matched random groups on MOSES and 91.9% on NPGen. The 10th–90th percentile intervals are 73.9%–92.9% and 71.7%–103.8%, respectively. Matching group sizes makes this comparison less sensitive to the number of sampled realizations. The retained dimension shows that compatibility constraints leave room for variation along many fingerprint directions, even though they restrict the available fragment choices. This measures the spread of variance across directions, rather than the total amount of variance.

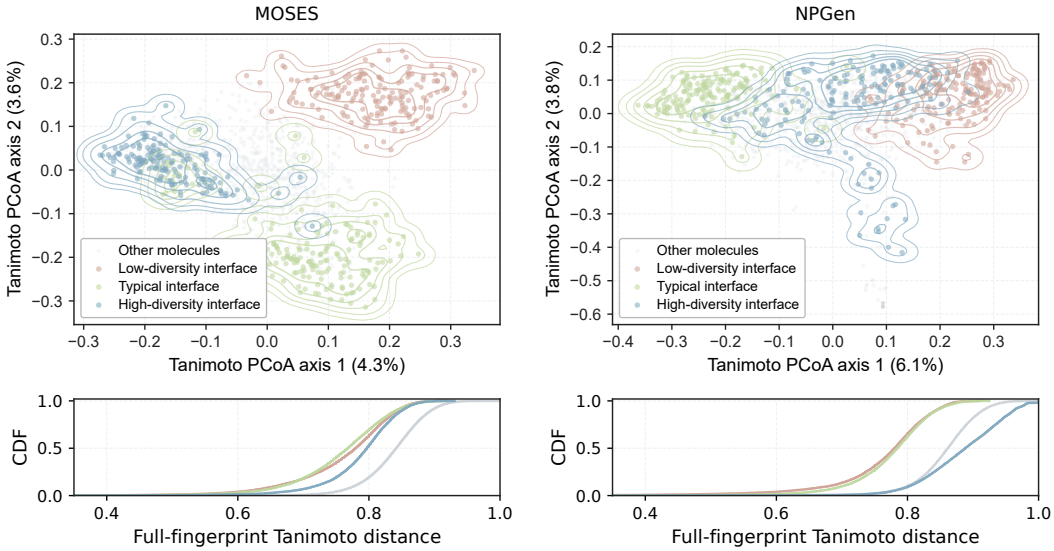


Figure 13: Classical PCoA projections of Tanimoto distances (upper panels) for interfaces near the 20th, 50th, and 80th percentiles of normalized participation-ratio dimension $d_{\mathrm{PR}}/\min(n-1, 512)$. Unlike the matched-group comparison above, selection here uses $n \leq 64$ distinct fingerprints per interface. Each selected interface is displayed using 160 distinct FIRM molecules; gray points sample 700 other generated molecules. Contours show projected density estimates. Percentages on the axes give each coordinate's share of the total positive PCoA eigenvalue mass. Lower panels show empirical cumulative distributions of all within-group pairwise Tanimoto distances, computed from the same molecules' complete 512-bit fingerprints; the gray curves use pairwise distances among the 700 background molecules.

Molecules sharing an interface spread across the PCoA projections and partly overlap the background sample (Figure 13). The displayed coordinates capture 7.9% of positive eigenvalue mass on MOSES and 9.9% on NPGen. The paired distance distributions use all 512 fingerprint bits and reveal variation hidden by those two coordinates: the three selected interfaces differ in their within-group spread, especially on NPGen. These curves describe the displayed samples; the covariance analyses above assess variation across a much larger set of interfaces.

Fragment selection. Figure 14 compares fragment use with the test set by training frequency. Figure 15 reports selection within each slot's compatible set: a template at frequency percentile $u \in [0, 1]$ has relative rarity $100(1-u)$, with midranks for ties. Each interval gives its fraction of slot assignments.

FIRM selects singleton fragments 3.96 times as often as the test set on MOSES and 4.67 times on NPGen; relative use is near one for fragments observed at least five times. Beyond 5% relative rarity, selection mass changes from 10.48% to 10.86% on MOSES and from 6.62% to 13.11% on NPGen. The shift toward rarer compatible fragments is larger on NPGen. The two comparisons describe different aspects of fragment choice: a fragment that is rare in the full training set need not be rare among the candidates compatible with a particular slot. On MOSES, singleton enrichment accompanies little change in the reported relative-rarity mass. On NPGen, both comparisons show

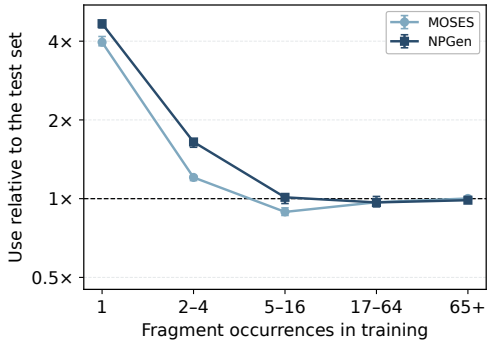


Figure 14: Fragment use relative to the test set by training-frequency bin. Points are three-seed means; error bars span the seedwise minimum and maximum.

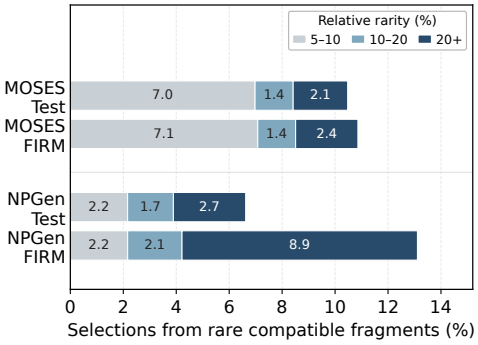
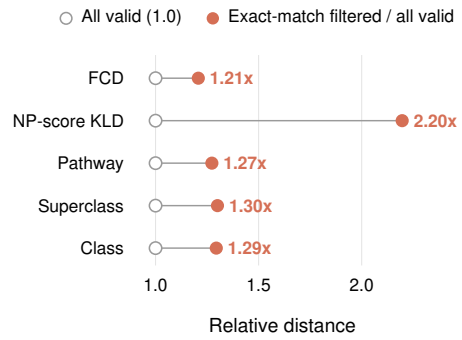


Figure 15: Absolute selection mass in relative-rarity intervals, where rarity is defined within each slot's compatible fragment set.

greater use of rare fragments, indicating that the shift also occurs within the choices allowed by compatibility.

C.6 NPGen fidelity beyond exact training matches

We remove exact training matches from 30,000 valid NPGen samples for each of seeds 42, 43, and 44, excluding 3,484, 3,626, and 3,694 samples, respectively, and reevaluate the retained molecules.



Metric	All valid	Filtered	Ratio
FCD	0.3191	0.3847	1.21
NP-score KLD	0.0109	0.0239	2.20
Pathway	0.0094	0.0119	1.27
Superclass	0.0915	0.1189	1.30
Class	0.2878	0.3726	1.29

Figure 16: NPGen distribution distances after removing exact training-set matches. Left: mean filtered-to-unfiltered ratios across the three runs. Right: absolute mean distances and mean seedwise ratios; ratios need not equal quotients of the rounded means. Lower absolute values indicate closer distribution matching.

All distances and divergences increase after filtering (Figure 16). FCD rises from 0.3191 to 0.3847 (1.21 times); pathway, superclass, and class divergences rise by factors of 1.27 to 1.30. NP-score KLD shows the largest relative increase, from 0.0109 to 0.0239.

The filtered logP and TPSA distributions remain close to the reference (Figure 17), with medians of 3.37 versus 3.28 for logP and 89.17 versus 83.91 for TPSA.

Retained unique molecules have mean ECFP4 Tanimoto similarity 0.5475 to their nearest training molecule. Among those with a nonempty Bemis–Murcko scaffold, 58.40% have a scaffold absent from training.

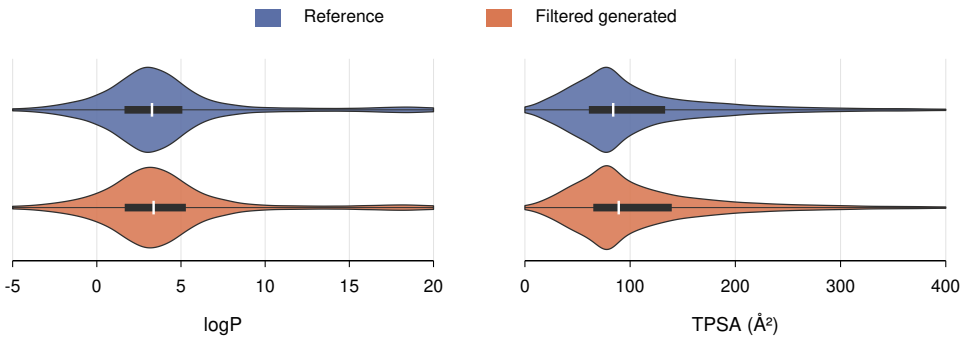


Figure 17: LogP and TPSA distributions for the NPGen test set and generated molecules retained after exact-match filtering. Distributions use unique reference molecules and unique retained molecules per run, with equal weight across runs. Dark bars show interquartile ranges, and white marks show medians.

C.7 Retrosynthetic planner compatibility

We evaluate FIRM samples from seed 42 and test molecules on all four benchmarks with AiZynthFinder 4.4.1 (Genheden et al., 2020), using the USPTO expansion and filter policies with ZINC stock. Each group contains 25,000 MOSES, 30,000 NPGen, 10,000 GuacaMol, or 5,000 ZINC250K molecules; the MOSES reference uses the scaffold test split. Monte Carlo tree search uses at most 100 iterations, 120 seconds, and six transformations; the target is excluded from stock. Figure 18 summarizes the search outcomes. A route succeeds only if it preserves the target identity.

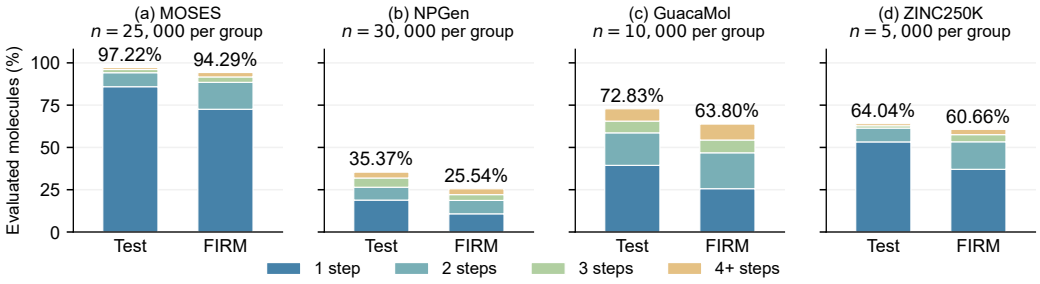


Figure 18: Retrosynthetic search outcomes for test molecules and FIRM samples (seed 42) under the same search settings. Each panel compares equally sized groups. Stacked segments show route-length fractions among all evaluated molecules; their sum is the search success rate, labeled above each bar.

Search success rates for FIRM and test molecules are 94.29% versus 97.22% on MOSES, 25.54% versus 35.37% on NPGen, 63.80% versus 72.83% on GuacaMol, and 60.66% versus 64.04% on ZINC250K. FIRM has a lower success rate on every benchmark, with the largest gap on NPGen (9.84 percentage points). Search success depends on the reaction policies, available stock, and search budget.

Planner cuts and interface boundaries. For 22,993 MOSES FIRM molecules with mapped first-step planner cuts, we measure the fraction of cut bonds coinciding with interface boundaries and average over molecules in each interface-frequency bin. Overlap ranges from 88.9% to 92.1%: unseen interfaces reach 89.5%, versus 88.9% for interfaces observed at least 65 times. Both use BRICS boundary types observed in training.

C.8 FINITE-STEP BEHAVIOR AND COMPUTATIONAL COST

Interface Flow step budget. For each $K_I \in \{8, 16, 32, 64\}$, we draw 30,000 proposals on MOSES, GuacaMol, ZINC250K, and NPGen. Figure 19 compares correction rates and the effect of Realization Flow budgets on updates, time, and NPGen FCD.

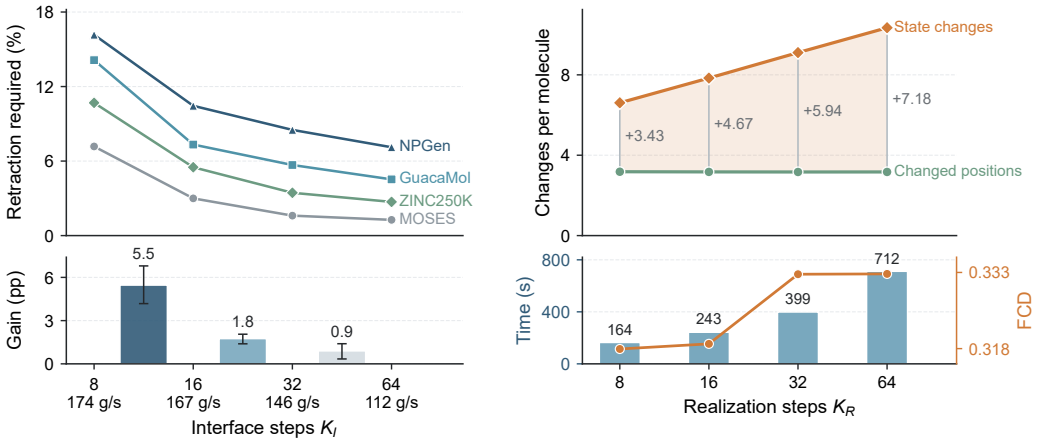


Figure 19: Finite-step behavior with 30,000 samples per setting. The left panel reports the fraction requiring retraction across four Interface Flow budgets. Lower bars give the mean reduction for each adjacent step interval, whiskers span the dataset-specific reductions, and the second row of tick labels gives NPGen throughput. The right panel shows NPGen changed slots and repeated changes across Realization Flow budgets. Its lower chart reports realization-stage wall-clock time and FCD on separate axes.

We evaluate each budget with seed 42, using batches of 256. Interface timing covers sampling and feasibility checks, excluding loading, output, and retraction. Realization timing starts from fixed interfaces and includes loading, candidate preparation, sampling, assembly, sanitization, and output, but excludes benchmark scoring.

Most of the reduction occurs between 8 and 16 Interface Flow updates on all four datasets. Averaged across datasets, this change lowers the fraction requiring retraction by 5.5 percentage points; the next two doublings lower it by another 1.8 and 0.9 points. On NPGen, 16 updates reduce the fraction from 16.17% to 10.45% while retaining 95.6% of the eight-update throughput. Increasing the budget from 16 to 64 updates lowers the fraction to 7.11% and throughput from 166.7 to 112.4 graphs per second. Sections C.3 and C.4 examine how correction varies with interface frequency and size.

Realization Flow update concentration. Across Realization Flow budgets, the mean number of slots changed at least once per molecule remains between 3.17 and 3.18. Additional changes to those slots rise from 3.43 at 8 updates to 4.67, 5.94, and 7.18. From $K_R = 16$ to 64, repeated changes increase by 54% and realization-stage wall-clock time by 193%. FCD is nearly unchanged at 8 and 16 updates (0.3179 and 0.3189) but rises to 0.3327 at both 32 and 64. In this evaluation, larger budgets revisit the same slots more often without lowering FCD.

D MOLECULAR VISUALIZATIONS

D.1 EVOLUTION THROUGH THE TWO FLOWS

Figures 20 and 21 show seed-42 trajectories for MOSES and NPGen molecules containing aromatic heterocycles and an oxygenated pyrone ring with a methoxyphenyl group, respectively. Both final molecules are absent from their training sets even when stereochemistry is ignored, and both interface proposals decode directly to feasible interfaces.

The upper row shows Interface Flow in a fixed slot layout, with node numbers identifying the same slots throughout. It displays signature, pair-kind, and connection-rule coordinates; port-role coordinates also evolve.

The lower row assembles Conditional Realization Flow states under the terminal interface. Transport changes fragment assignments while preserving interface compatibility, which holds at the source. Common substructures share an orientation to make changes easier to follow.

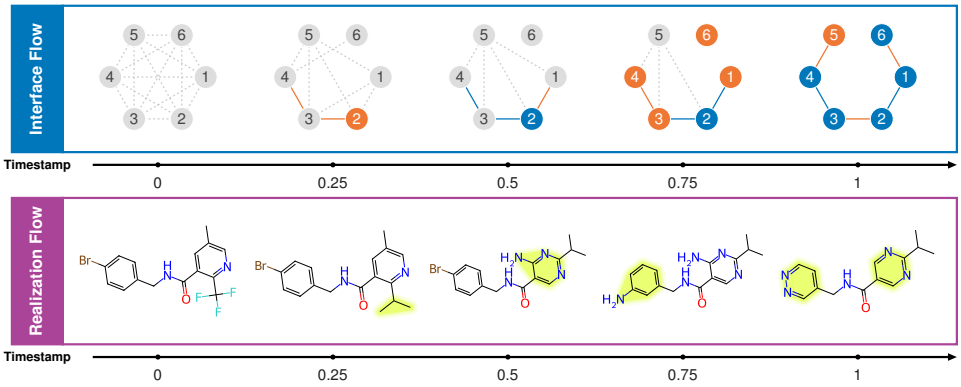


Figure 20: MOSES generation through Interface Flow and Conditional Realization Flow. Colors are relative to the preceding displayed snapshot: orange marks interface changes, blue unchanged revealed states, gray masks, and yellow changed fragments; dashed edges mark deleted connections.

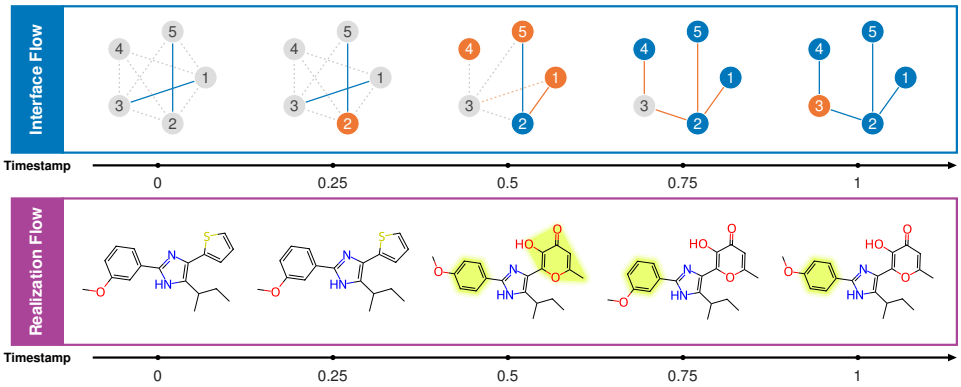


Figure 21: An NPGen generation trajectory with an oxygenated pyrone ring and a methoxyphenyl group. Normalized times and colors follow Figure 20. Fragment assignments change under a fixed terminal interface.

Each stage spans 16 updates on a separate normalized time axis from 0 to 1. Colors compare successive displayed snapshots. In Interface Flow, gray denotes masked states; orange marks newly revealed or changed node signatures, connection types, or connection rules, and blue marks unchanged revealed states. Pale orange dashed lines mark deleted connections. In Realization Flow, bright yellow highlights fragments whose template or attachment mapping changed. Changes are not highlighted at the start of either stage.

D.2 Unconditional molecular samples

Generated molecules were pooled across three independently trained runs, canonicalized, deduplicated with RDKit, and sampled at random. NPGen samples were also stratified by heavy-atom count. All structures use the same rendering settings.

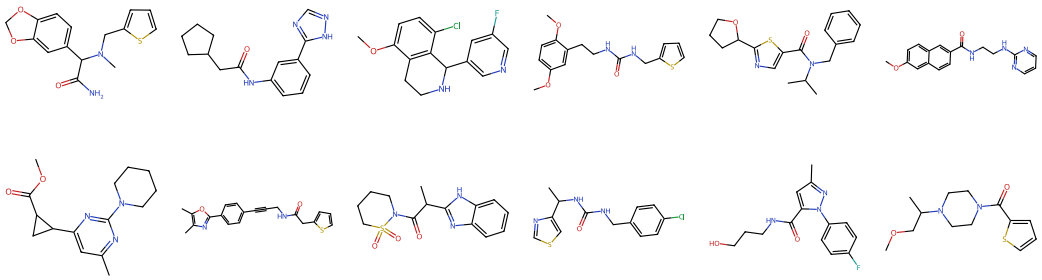


Figure 22: Twelve randomly selected molecules generated by FIRM on MOSES. The examples vary in ring arrangements and connecting groups.

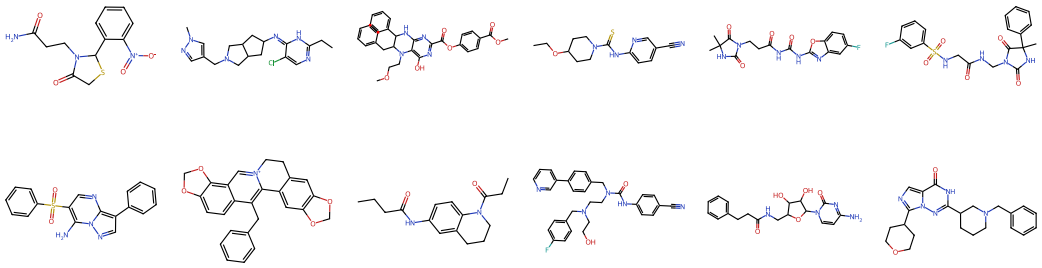


Figure 23: Twelve randomly selected molecules generated by FIRM on GuacaMol. The examples include linked and fused ring systems as well as extended chains.

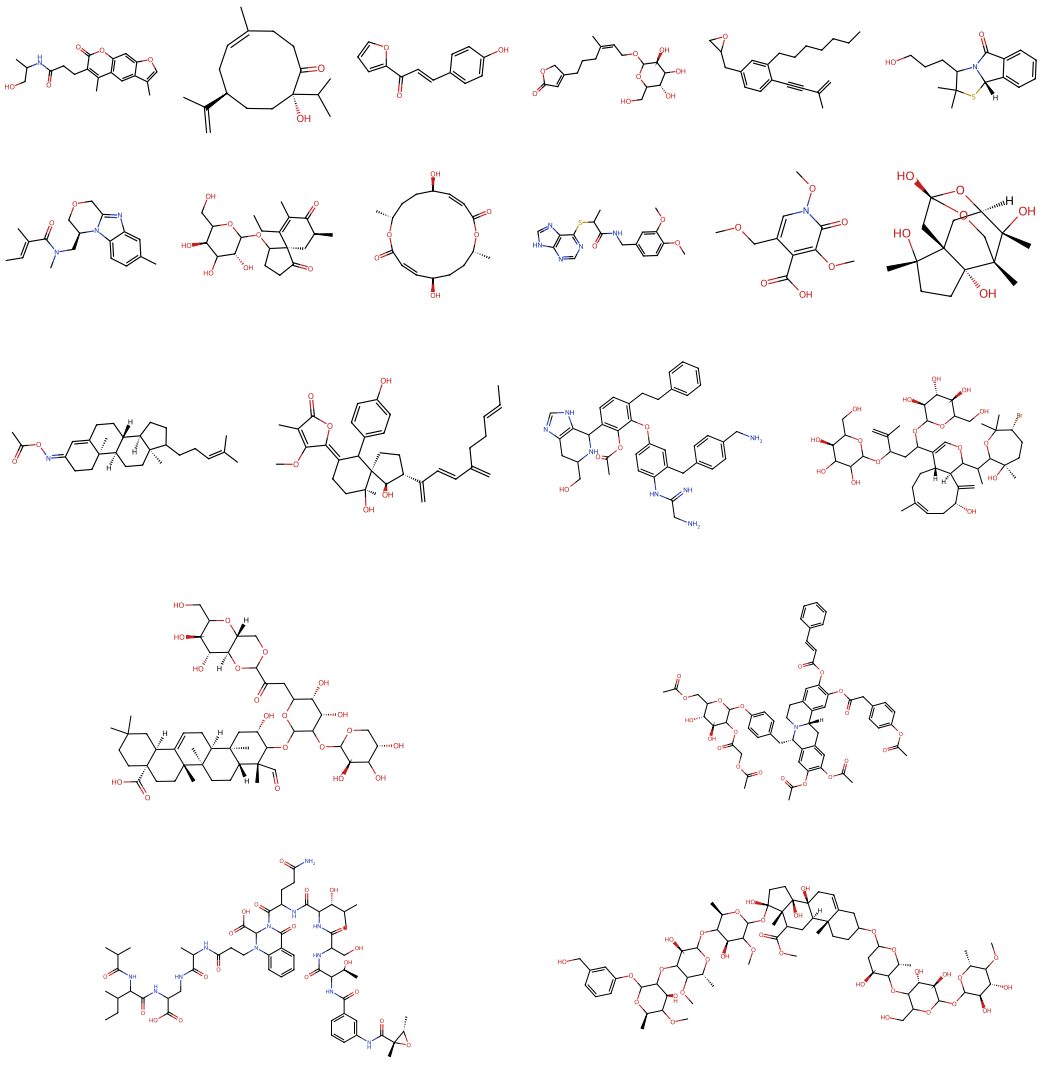


Figure 24: Molecules generated by FIRM on NPGen, grouped by heavy-atom count. The first two rows contain molecules with fewer than 30 heavy atoms; later rows cover counts from 30 to 100. The examples include compact ring systems and larger structures with multiple rings and side chains.