Chain-of-Generation: Progressive Latent Diffusion for Text-Guided Molecular Design

Lingxiao Li
llingxia@umich.edu
University of Michigan

Haobo Zhang
haobozha@umich.edu
University of Michigan

Bin Chen
chenbi12@msu.edu
Michigan State University

Jiayu Zhou
jiayuz@umich.edu
University of Michigan

Abstract

Compositional prompts in text-to-molecule generation often specify several structural constraints at once, such as scaffolds, functional groups, substituents, and finer modifiers. Existing molecular latent diffusion models typically inject the full prompt once and apply it uniformly throughout denoising, which can cause later constraints to overwrite or distort earlier structural commitments. We propose **Chain-of-Generation (CoG)**, a training-free progressive latent diffusion strategy for text-guided molecular design. Technically, CoG realizes progressive generation as multi-stage latent denoising over cumulative molecular sub-prompts: it factorizes a prompt into chemistry-informed coarse-to-fine segments, conditions each stage on the cumulative sub-prompt, and continues denoising from the previous-stage latent using a fixed partial-denoising schedule. This turns one-shot conditional generation into a sequence of stage-wise denoising steps that help retain earlier structure while introducing later constraints. Across benchmark, out-of-distribution, and ablation settings, CoG improves structural prompt fidelity over one-shot conditioning on the tested latent graph-diffusion backbones.

1. Introduction

Text-conditioned molecular generation translates natural-language descriptions into molecular structures, letting a chemist specify scaffolds, functional groups, substituents, and finer modifiers without handcrafted constraint encodings. A central feature of this setting that is rarely treated as such is that prompts are *compositional*: a single description encodes several graph-level structural constraints at different chemical scales (a core scaffold, one or more functional groups attached to it, and fine-grained modifiers such as halogens, charges, or stereochemical configurations). Faithful generation requires the model to satisfy all of them jointly. Early work in this field primarily focused on retrieval-based approaches (Edwards et al., 2021; Zeng et al., 2022). Later sequence-to-sequence models (Edwards et al., 2022; Christofidellis et al., 2023) address this by token-level decoding over SMILES (Weininger, 1988), which linearizes the molecular graph and makes graph-level constraints implicit; in our experiments we observe that this can lead to invalid outputs or mode collapse.

Latent diffusion models (LDMs) for molecular graphs (Rombach et al., 2022; Zhu et al., 2024) provide an attractive alternative because their stochastic sampling process facilitates the creation of structurally diverse and novel molecules. In practice, however, current text-conditioned molecular LDMs encode the entire prompt once and inject the resulting embedding throughout the denoising trajectory, a *one-shot conditioning* regime. We observe that this regime struggles when prompts are compositional: as the number of structural constraints increases, generated molecules increasingly omit substructures, distort attachments, or trade earlier scaffolds for later modifiers. Figure 1 illustrates this failure mode which limits the capacity of generation.

There are three main challenges of one-shot conditioning that limit the capacity of compositional molecular generation. First, a full prompt is encoded once and applied uniformly during denoising, so the generation process has no explicit stage corresponding to an individual scaffold, functional group, substituent, or fine modifier. Second, as constraints accumulate, generated molecules can omit required components, distort attachments, or trade an earlier scaffold-level

Accepted to FoGen 2026: Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning, an ICML 2026 workshop (non-archival).

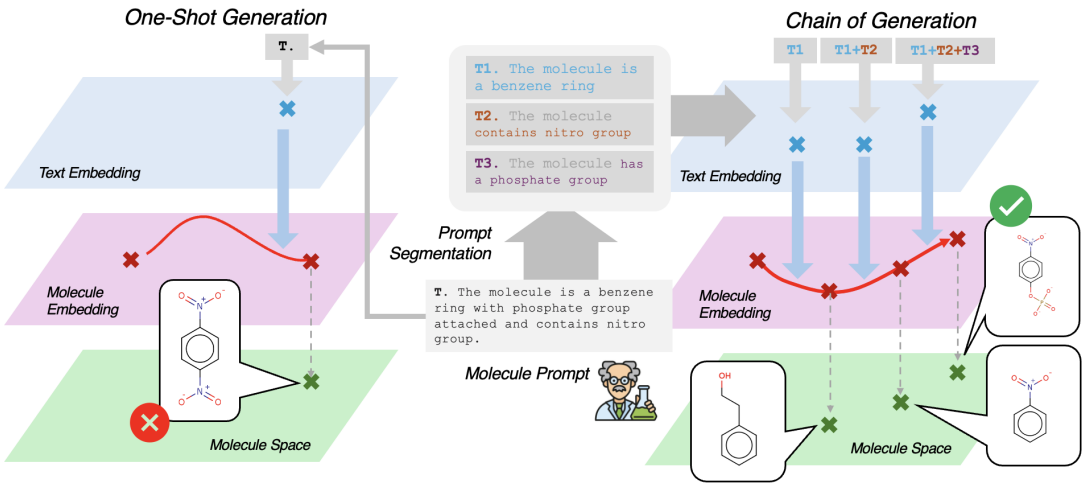


Figure 1. Overview of CoG. One-shot conditioning injects a single full-prompt embedding throughout the denoising trajectory. CoG first decomposes the prompt into chemistry-informed segments and then performs cumulative multi-stage denoising: each stage conditions on the cumulative sub-prompt $T_{1:k}$ and continues the partial denoising trajectory from the previous-stage latent.

structure for a later local constraint. Third, scaffold-level, functional-group-level, and fine-modifier-level requirements are forced into a single reverse trajectory, even though molecular descriptions naturally follow a coarse-to-fine organization.

We then propose **Chain-of-Generation** (CoG), a training-free progressive latent diffusion strategy for text-guided molecular design which has three CoG designs to address these challenges. (1) Chemistry-informed prompt factorization decomposes the prompt into meaningful molecular components. (2) Cumulative conditioning makes earlier components remain in the text condition when later components are introduced. (3) Progressive coarse-to-fine partial denoising follows the chemical order from scaffold to functional groups to fine modifiers, while later stages initialize from the previous-stage latent rather than restarting from noise. We perform extensive experiments on benchmark, out-of-distribution, and synthetic datasets, showing that CoG yields more faithful text-guided molecular generation that can be applied to existing latent graph-diffusion backbones.

2. Related Work

Text-conditioned molecular generation is an emerging task at the intersection of computational chemistry and multimodal learning. It aims to generate molecular structures from natural language prompts, enabling intuitive, goal-directed molecular design. Early works approached this as a *retrieval* problem, learning joint embeddings to align text and molecular representations (Edwards et al., 2021; Zeng et al., 2022). While effective, these methods are constrained to pre-existing molecules and cannot generate novel structures. To overcome this, generation-based models emerged, typically translating text to SMILES strings (Weininger, 1988). Models such as MolT5 (Edwards et al., 2022) and ChemT5 (Christofidellis et al., 2023) use encoder–decoder transformers to map prompts to molecules, benefiting from large pretrained language models. However, prior sequence-to-sequence approaches exhibit severe *mode collapse*: despite the large solution space implied by flexible prompts, they repeatedly converge to a single rigid output. A second limitation is *chemical invalidity*. Because SMILES impose an artificially defined string grammar, models must implicitly relearn these rules, often producing syntactically or chemically invalid molecules. An illustrative example is provided in Appendix G. Graph-based diffusion methods alleviate the issue of invalid SMILES generation by operating directly on molecular graphs. However, performing diffusion in the original discrete graph space remains problematic: permutation invariance introduces many isomorphic orderings, and training often relies on auxiliary reconstruction losses (Vignac et al., 2022; Simonovsky & Komodakis, 2018). To avoid these limitations, LDMs (Zhu et al., 2024)

encode graphs into a continuous latent space and perform text-conditioned diffusion there, preserving validity while enabling stochastic, conditional exploration for more diverse and novel molecules. Our work occupies a different slice of the design space. We do not propose a new molecular representation, encoder, or backbone diffusion model. Instead, we treat compositional prompt following as an inference-time progressive latent diffusion problem for text-guided molecular design and contribute the CoG inference strategy. The CoG inference strategy combines chemistry-informed prompt factorization, cumulative conditioning, and partial denoising from the previous-stage latent, and can be applied on top of existing latent graph-diffusion backbones without additional training.

3. Preliminary and Motivation

3.1. Preliminary

Latent diffusion models for molecular graphs (Rombach et al., 2022; Zhu et al., 2024) learn a bidirectional mapping between molecular graphs and a continuous latent space, then run a conditional diffusion process in that latent space and decode the result back to a molecular graph. We adopt a latent graph-diffusion backbone following prior molecular LDMs (Zhu et al., 2024): a variational graph auto-encoder (Kingma et al., 2013) whose encoder maps a molecular graph G into a latent Gaussian via a Graph Isomorphism Network (Xu et al., 2018), and whose graph decoder follows the HierVAE motif-based architecture (Jin et al., 2020). Encoder and decoder are trained jointly with the standard ELBO,

$$\mathcal{L}_{\text{elbo}} = -\mathbb{E}_{q(g|G)}\Big[\log p(G \mid \hat{G})\Big] + D_{\text{KL}}[q(g \mid G) \,\|\, p(g)]\,,$$

with $\hat{G} = D(g)$ the reconstructed graph and $p(g) = \mathcal{N}(0, I)$ the prior. The reparameterization trick (Kingma et al., 2013) provides a continuous latent g as the diffusion target.

Text-conditioned generation is then a conditional diffusion process in this latent space. Starting from g_0, the forward process adds Gaussian noise on a predefined schedule, $z_t = \sqrt{\bar{\alpha}_t}\, g_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$. A denoising network θ is trained to recover g_0 from z_t given the text embedding c by minimizing $\mathbb{E}_{t,c,g_0,\epsilon}\Big[\|\epsilon - \theta(z_t, t, c)\|^2\Big]$. At sampling time, the reverse process produces a clean latent under condition c, which is decoded into a molecular graph.

3.2. Motivation

Recent advances in latent diffusion models have shown promise in text-conditioned molecular generation, but complex prompts often remain challenging with one-shot conditioning: models sometimes fail to construct molecular structures that fully reflect all required semantics. To probe this issue, we investigate how diffusion models, as black-box generators, parse natural language descriptions and incrementally assemble molecular structures during denoising. We conduct a trajectory analysis by decoding partially denoised latents and visualizing intermediate outputs, enabling us to examine how structural features emerge and evolve throughout the generation process.

We begin with simple prompts (e.g., "`This molecule is made of a pyridine ring`") and gradually increase complexity by adding compositional elements. We see that when the prompt includes three or more structural keywords, model performance begins to diverge significantly. Figure 2 shows an example. The prompt is constructed from compositional vocabulary in our training set. In the visualization, colored regions correspond to substructures correctly generated in accordance with the prompt, while uncolored parts indicate incomplete alignment. Notably, as we increase the number of components in the prompt, latent diffusion cannot simultaneously handle all the requirements, indicating the limitation of one-shot conditioning. Additionally, the model tends to resolve larger, high-level scaffolds first, for example, the pyridine ring, which is designated as the core structure in the prompt. Then, as denoising proceeds, it decodes peripheral substructures piece by piece (e.g., attaching a nitro group) that serve as modifiers or substituents. This illustrates that the model progressively integrates semantic elements from the prompt, aligning more closely with the intended meaning over the course of denoising, a behavior rarely observed in sequence-to-sequence models. Motivated by this observation, we propose our Chain-of-Generation inference strategy in Section 4.3.

4. Molecule Chain-of-Generation

Motivated by the aforementioned observation of one-shot conditioning in Section 3.2, we propose a novel **Chain-of-Generation (CoG)** inference technique in this section, which decomposes each prompt into semantically meaningful segments and progressively guides the generation through these stages. Importantly, CoG operates entirely at sampling

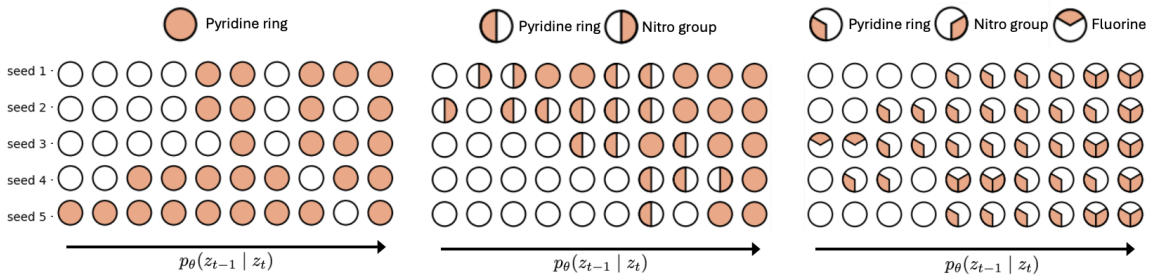


Figure 2. Decoded-trajectories for prompts of increasing compositional complexity. Each panel corresponds to one prompt; within each panel, rows are random seeds and columns are decoded intermediate states along the one-shot reverse trajectory. Circles reflect decoded intermediate states, and colors indicate substructures matching the corresponding prompt component. **Left**: ``The molecule is made of a pyridine ring.'' **Middle**: ``The molecule is a pyridine ring with a nitro substituent.'' **Right**: ``The molecule is made of a pyridine ring substituted with both a nitro group and fluorine atoms.'' Note that as the complexity of the prompt increases, one-shot conditioning may not be able to satisfy all the requirements. Moreover, latent diffusion ensembles each component progressively during generation.

time without requiring retraining. We first discuss three challenges of one-shot conditioning. We then propose three designs on latent diffusion to address those challenges. Equipped with that understanding, we introduce our CoG strategy.

4.1. Challenges of One-Shot Conditioning

Although recent LDMs have demonstrated promising capabilities in text-conditioned molecular generation, a key challenge remains in how language guidance is incorporated into the latent generation process. Existing methods encode the entire textual prompt once and use it to condition the entire generation process. However, this one-shot conditioning strategy struggles with complex or compositional prompts. In Section 3.2, empirical results are shown to illustrate the failure of one-shot conditioning. In this section, we further analyze the challenges of one-shot conditioning from three perspectives. **C1: No component-level conditioning handle.** In one-shot conditioning, the full prompt is encoded once and applied uniformly, so the sampler has no explicit stage at which a scaffold, functional group, substituent, or fine modifier is emphasized. This makes the generation process difficult to localize with respect to prompt components. **C2: Missing or distorted substructures.** When multiple structural requirements are specified, one-shot conditioning can produce molecules that satisfy the core scaffold but omit a peripheral substituent, or satisfy a local modifier while distorting an earlier scaffold-level structure. This is the main output-level failure as prompts become compositional. **C3: All-at-once conditioning across chemical scales.** A compositional molecular prompt contains constraints at different chemical granularities, but one-shot conditioning asks a single reverse trajectory to satisfy them simultaneously. This ignores the natural coarse-to-fine organization of molecular descriptions and leaves the generation order underspecified.

4.2. Progressive Latent Diffusion Design

In this section, we propose three designs to address the above three challenges, respectively. **D1: Chemistry-informed prompt factorization.** To address C1, CoG decomposes the prompt into molecular components whose granularity follows chemical descriptions, such as scaffold, functional group or substituent, and fine modifier. This gives each component an explicit conditioning stage instead of compressing all components into one prompt embedding. **D2: Cumulative conditioning.** To address C2, CoG conditions each stage on the cumulative sub-prompt, not only on the newest component. Earlier textual constraints therefore remain visible to the denoiser while later components are added. **D3: Progressive coarse-to-fine partial denoising.** To address C3, CoG follows the natural chemical progression from coarse scaffold to functional groups or substituents and then to fine modifiers. On the latent side, each later stage initializes from the previous-stage latent and runs partial denoising under the cumulative condition, rather than restarting from noise. Thus, “progressive” in CoG means both coarse-to-fine chemical conditioning and stage-wise latent denoising from the previous-stage result.

4.3. CoG: Progressive Denoising over Cumulative Sub-Prompts

We now instantiate D1–D3 as the CoG inference procedure. The modification is at *inference* only: we replace the standard one-shot reverse process under condition $E_{\text{text}}(T)$ by a sequence of partial reverse processes under cumulative sub-prompts, where each later stage initializes from the previous-stage latent and runs only the final portion of the reverse trajectory.

CoG differs from generic multi-stage prompt scheduling in that each stage is not an independent prompt restart: the text condition is cumulative, and each later stage initializes the partial reverse trajectory from the previous-stage latent rather than re-sampling from the prior. The stage index thus changes the text condition and the latent initialization.

Notation. We use k to index the CoG stage and t for the diffusion timestep within a stage. Let T denote the original prompt. A segmentation operator $\mathcal{S}$ produces an ordered list of chemistry-informed segments

$$\mathcal{S}(T) \;=\; (T_1, T_2, \ldots, T_K),$$

where each T_k is a textual fragment describing a structural element at a particular chemical scale (scaffold, functional group, substituent, fine-grained modifier). We write the cumulative sub-prompt and its text-encoder embedding as

$$T_{1:k} \;=\; T_1 + T_2 + \cdots + T_k, \qquad c_k \;=\; E_{\text{text}}(T_{1:k}).$$

Let $\mathcal{R}_\theta$ be the trained backbone's conditional reverse sampler, D the graph decoder, and S the total number of reverse steps. We let z_k denote the latent generated at the end of stage k.

Chemistry-informed prompt factorization. $\mathcal{S}$ orders segments coarse-to-fine: T_1 captures the molecular scaffold or core ring system, $T_2, \ldots, T_{K-1}$ describe functional groups and substituents attached to it, and T_K describes fine-grained modifiers (specific atoms, charges, stereochemical configurations). Each T_k is an *atomic* segment, while $T_{1:k} = T_1 + T_2 + \cdots + T_k$ is the cumulative sub-prompt that the model conditions on at stage k. For example, given T = "`The molecule is a benzene ring with a phosphate group attached and contains a nitro group`", a coarse-to-fine factorization is T_1: "`The molecule is a benzene ring.`", T_2: "`The molecule contains a nitro group.`", and T_3: "`The molecule has a phosphate group attached.`" Manual factorization is feasible for small studies; we automate it with an LLM under a system prompt that names the granularity levels (Appendix C). Different LLMs are compared in Table 3; Grok v3 is used as the default segmenter in our main experiments.

Partial denoising under cumulative conditioning. Let $\mathcal{R}_\theta^{s\to 0}(z; c)$ denote the trained backbone's conditional reverse sampler, initialized from latent z at timestep s and conditioned on the continuous text embedding c, integrated down to $t = 0$. CoG uses this sampler unchanged. At stage $k = 1$, CoG runs the full reverse trajectory from random noise:

$$z_1 \;=\; \mathcal{R}_\theta^{S\to 0}(\epsilon;\, c_1)\,, \qquad \epsilon \sim \mathcal{N}(0, I).$$

For each subsequent stage $k \geq 2$, CoG initializes the reverse sampler from the previous-stage latent and runs only the final portion of the trajectory under the cumulative condition:

$$z_k \;=\; \mathcal{R}_\theta^{s_{\text{start}}\to 0}(z_{k-1};\, c_k)\,, \tag{1}$$

where s_{start} is a fixed partial-denoising start. After the final stage, the molecular graph is obtained by decoding $\hat{G} = D(z_K)$. In practice we set s_{start} to roughly 20% of the total reverse schedule (i.e., $s_{\text{start}} \approx 0.2\,S$).

Interpretation. CoG can be viewed as domain-informed progressive denoising. Instead of solving the full molecular prompt in a single conditional denoising process, it solves a sequence of increasingly constrained conditional sampling problems indexed by k. The stage order follows molecular composition, from scaffold-level constraints to functional groups and finer modifiers. The partial-denoising start s_{start} in Eq. (1) controls how much of the reverse trajectory is run at each later stage: a smaller s_{start} keeps the sample close to z_{k-1} and helps retain earlier structure, while a larger s_{start} allows more revision. Cumulative conditioning $c_k = E_{\text{text}}(T_{1:k})$ keeps earlier constraints active; together with a small s_{start} this is designed to retain previously introduced structural commitments. The ordering is a chemistry-informed design choice motivated by how molecular descriptions often organize structures around a scaffold and attached groups

Algorithm 1 Chain-of-Generation (CoG) sampling

Require: Prompt T; segmentation operator $\mathcal{S}$; text encoder E_{text}; backbone reverse sampler $\mathcal{R}_\theta$; decoder D; partial-denoising start s_{start}; total reverse steps S.

$(T_1, \ldots, T_K) \leftarrow \mathcal{S}(T)$ {chemistry-informed coarse-to-fine factorization}
for $k = 1, 2, \ldots, K$ **do**
  $T_{1:k} \leftarrow T_1 + \cdots + T_k$ {cumulative sub-prompt}
  $c_k \leftarrow E_{\text{text}}(T_{1:k})$
  if $k = 1$ **then**
    $\epsilon \sim \mathcal{N}(0, I)$
    $z_1 \leftarrow \mathcal{R}_\theta^{S \to 0}(\epsilon;\, c_1)$ {full reverse trajectory from prior}
  else
    $z_k \leftarrow \mathcal{R}_\theta^{s_{\text{start}} \to 0}(z_{k-1};\, c_k)$ {partial denoising from previous-stage latent}
  end if
end for
$\hat{G} \leftarrow D(z_K)$
return $\hat{G}$

(scaffold first, then substituents, then fine modifiers). Decomposition-based reasoning in LLMs (Wei et al., 2022; Yao et al., 2023) provides a loose analogy for staged sub-task solving, but our ordering is grounded in molecular composition rather than in language-reasoning structure. The ablation study compares this ordering against fine-to-coarse and non-cumulative variants.

5. Experiments

We conduct extensive experiments to evaluate the proposed technique. Section 5.1 motivates the graph-level structural-fidelity profile used throughout the paper: we explain why string-level, retrieval-style, and distribution-level metrics answer different evaluation questions, and define Validity, HM, Q-Cov, and Q-Nov as proxy measures of structural prompt fidelity. Section 5.2 evaluates CoG both in-domain on ChEBI-20 and PubChem and out-of-distribution on PubChemSTM. The benchmark tables compare SMILES-based methods and two molecular latent diffusion backbones with and without CoG; paired + CoG rows isolate the gain from the CoG inference strategy. Section 5.3 studies prompt planning and ablations on controlled compositional prompts, testing whether coarse-to-fine ordering and cumulative conditioning are both necessary.

5.1. Metrics

Text-guided molecular generation is not exact string prediction: a prompt can correspond to multiple valid molecular graphs (Figure 3). We do not use string-level metrics such as exact SMILES match, BLEU, or Levenshtein as primary scores, because they evaluate a particular serialization rather than the molecular graph; graph-equivalent or structurally close molecules can receive very different strings. Text2Mol-style scores (Edwards et al., 2021) measure text–molecule retrieval alignment, while distribution-level statistics such as Fréchet ChemNet Distance and the GuacaMol (Brown et al., 2019) and MOSES (Polykovskiy et al., 2020) suites are more appropriate for unconditional or set-level molecular generation. Our setting evaluates paired prompt-reference examples, so we report a graph-level *structural-fidelity profile* using MACCS fingerprints (Durant et al., 2002; O'Hagan & Kell, 2015) and Tanimoto similarity computed with RDKit (Bento et al., 2020): MACCS is a 166-bit predefined keyset of substructure indicators computed from the molecular graph, hence invariant to SMILES enumeration and canonicalization. These alternatives answer useful but different evaluation questions; we treat our four numbers as a structural-fidelity profile rather than a complete measure of prompt correctness, in line with multi-axis evaluation practice in generative modeling that separates fidelity, coverage, and diversity (Sajjadi et al., 2018; Naeem et al., 2020).

Reported metrics. Validity is the fraction of generations that decode to a chemically valid graph. HM is the harmonic mean of a broad structural-agreement rate (MACCS/Tanimoto similarity to the reference above 0.5) and a qualified structural-agreement rate (similarity in $(0.5, 0.8)$); it summarizes how often the generated graph hits both a low-bar and a higher-bar level of structural fidelity. Q-Cov measures how often valid generations fall into the qualified band.

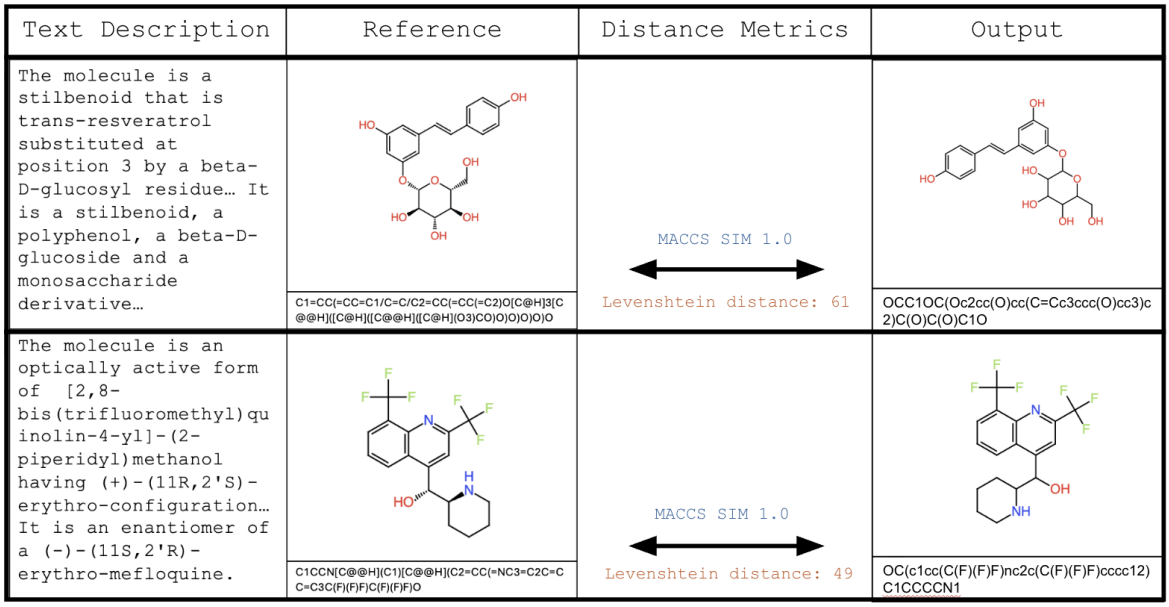


Figure 3. Two examples where nearly identical molecular graphs are represented by drastically different SMILES strings. NLP-based string metrics (e.g., Levenshtein) often assign a large distance to such pairs even though the graphs are similar. MACCS/Tanimoto is less sensitive to SMILES serialization and better reflects graph-level similarity in these examples; it is a structural-fidelity proxy, not a complete measure of prompt correctness.

Q-Nov combines the qualified-band rate with pairwise structural diversity among broadly agreeing generations; it is not novelty against the training set. Reporting these four together as a structural-fidelity profile, rather than a single prompt-correctness score, mirrors evaluation practice with competing desiderata: F-measure-style harmonic means conservatively balance two quantities (Christen et al., 2023), and generative precision–recall metrics separate fidelity from coverage because one-dimensional scores can hide different failure modes (Sajjadi et al., 2018). Formulas with descriptive variables are given in Appendix D; computational cost is reported in Appendix E.

5.2. Benchmark Evaluation

We evaluate the CoG inference strategy on benchmark datasets. Specifically, we conduct experiments on PubChem (Liu et al., 2023b) and ChEBI-20 (Edwards et al., 2021), restricting all training, validation, and test molecules to fewer than 30 atoms to control molecular complexity. Following the CoG sampling procedure, we segment each prompt into coarse, intermediate, and fine-grained structural levels when applicable: core scaffold, functional groups or substituents, and finer modifiers such as atoms, charges, or stereochemical descriptors. We use Grok v3 as the default segmenter.

We evaluate CoG on top of two molecular latent diffusion backbones: 3M-Diffusion (Zhu et al., 2024) and GraphLDM (described in Appendix A). CoG is applied identically to each backbone. Table 1 reports both backbones with and without CoG. CoG raises HM on both backbones for both ChEBI-20 and PubChem, with per-metric deltas shown in the table. Q-Cov increases consistently in all four settings, and validity remains 100% throughout. The four-row split separates the effect of the CoG sampler from differences between the two backbones: CoG improves each backbone by a similar margin. For broader context, GraphLDM + CoG yields the highest HM on both datasets among the SMILES-based and graph-based baselines we report (Pei et al., 2023; Christofidellis et al., 2023; Edwards et al., 2022; Zhu et al., 2024; Chang & Ye, 2024).

Q-Nov is mostly stable, with a small decrease in the GraphLDM PubChem setting. We interpret this as a mild constraint satisfaction diversity trade-off. A detailed analysis stratified by prompt segmentation is provided in Appendix F.

Out-of-distribution check. We also evaluate compositional generalization by training on ChEBI-20 and testing on PubChemSTM (Liu et al., 2023a), whose molecules differ in scale and structural diversity. Table 2 shows that CoG

Table 1. Benchmark structural fidelity on ChEBI-20 and PubChem (%). + CoG rows apply CoG to the backbone immediately above; superscript deltas show gains from adding CoG. Bold main values mark the best value within each dataset, and bold deltas mark positive CoG gains. GraphLDM + CoG achieves the best HM on both benchmarks.

Method	**ChEBI-20**				**PubChem**			
	HM	Q-Cov	Q-Nov	Validity	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods								
MolT5 small	73.62	24.25	4.96	78.45	48.52	10.88	1.04	72.93
MolT5 large	60.82	20.35	2.23	98.08	54.22	16.27	1.45	94.15
ChemT5 small	68.76	23.88	3.25	96.99	63.69	20.05	2.47	91.91
ChemT5 base	70.78	25.08	3.45	97.25	66.03	20.77	2.75	90.65
BioT5 base	60.56	20.88	4.57	99.98	52.93	16.78	2.89	100
BioT5 plus	49.29	15.44	2.47	100	53.24	17.11	3.08	100
LDMol	12.39	1.79	0.14	99.34	40.42	4.37	0.57	98.21
Graph-based methods								
3M-Diffusion	71.80	40.32	9.88	100	69.68	40.07	10.78	100
+ CoG	75.38 $^{+3.58}$	42.83 $^{+2.51}$	10.27 $^{+0.39}$	100	74.11 $^{+4.43}$	42.84 $^{+2.77}$	10.80 $^{+0.02}$	100
GraphLDM	79.74	43.05	11.02	100	74.54	40.88	**12.56**	100
+ CoG	**84.97** $^{+5.23}$	**45.63** $^{+2.58}$	**11.10** $^{+0.08}$	100	**78.00** $^{+3.46}$	**43.36** $^{+2.48}$	11.95 $^{-0.61}$	100

improves HM, Q-Cov, and Q-Nov on both tested backbones, suggesting that the sampler-side gain is not limited to in-domain benchmark prompts.

Table 2. Out-of-distribution evaluation on PubChemSTM with models trained on ChEBI-20 (%). + CoG rows apply CoG to the backbone immediately above; superscript deltas show gains from adding CoG. CoG improves every reported metric for both backbones under distribution shift.

Method	HM	Q-Cov	Q-Nov
3M-Diffusion	92.70	34.34	8.56
+ CoG	97.61 $^{+4.91}$	36.97 $^{+2.63}$	8.69 $^{+0.13}$
GraphLDM	90.60	34.55	9.70
+ CoG	98.44 $^{+7.84}$	38.79 $^{+4.24}$	10.26 $^{+0.56}$

Unlike the in-domain setting, where Q-Nov can slightly decrease, the OOD check improves Q-Nov together with HM and Q-Cov, suggesting that the sampler-side gain is not simply a byproduct of reducing novelty.

5.3. Prompt Planning and Ablations

We use a controlled synthetic set of 100 compositional prompts, each designed to contain exactly three structural components. The purpose is to isolate two sampler-side choices: the ordering of prompt components and whether later stages use cumulative conditioning. We denote the coarse, intermediate, and fine components by T_l, T_m, and T_s, corresponding to scaffold, functional groups or substituents, and fine modifiers; the cumulative prompts $T_{l:m}$ and T correspond to $T_{1:2}$ and $T_{1:3}$ in the main CoG notation.

We compare four strategies across three LLM segmenters (GPT-4o (Hurst et al., 2024), Gemini 2.5 Pro (Google DeepMind, 2024), Grok v3 (xAI, 2024)): one-shot full prompt, fine-to-coarse cumulative prompting, coarse-to-fine non-cumulative prompting, and coarse-to-fine cumulative prompting. The last strategy is CoG.

All staged variants use the same backbone and the same number of stages, only the prompt order and whether later stages use cumulative conditioning are changed. Thus, Table 3 isolates the two sampler-side design choices rather than differences in backbone, segmenter, or schedule.

Table 3 shows that the coarse-to-fine cumulative strategy is the only staged strategy that remains competitive with or improves over one-shot conditioning across segmenters. Fine-to-coarse prompting uses the same number of stages but performs worse, suggesting that scaffold-level constraints should be established before local substituent constraints. The non-cumulative variant also follows the coarse-to-fine order, but drops earlier prompt components from later conditions and degrades sharply across segmenters. This indicates that the previous-stage latent alone is not sufficient; cumulative text conditioning is needed to keep earlier constraints visible to the denoiser.

Table 3. Prompt-order and cumulative-conditioning ablation on 100 synthetic compositional prompts (%). One-shot is a non-staged reference; bold numbers mark the best staged strategy within each segmenter. CoG dominates fine-to-coarse and non-cumulative controls, showing that both ordering and cumulative conditioning matter. Validity is 100% for all graph-based runs and is omitted.

Strategy	HM	Q-Cov	Q-Nov
Reference			
One-shot full prompt (T)	87.40	31.69	10.03
GPT-4o segmenter			
Fine-to-coarse cumulative ($T_s \to T_{s:m} \to T$)	77.34	28.00	7.64
Coarse-to-fine non-cumulative ($T_l \to T_m \to T_s$)	30.18	10.77	3.27
CoG: coarse-to-fine cumulative ($T_l \to T_{l:m} \to T$)	**87.32**	**31.07**	**8.42**
Gemini 2.5 Pro segmenter			
Fine-to-coarse cumulative ($T_s \to T_{s:m} \to T$)	71.69	25.85	7.13
Coarse-to-fine non-cumulative ($T_l \to T_m \to T_s$)	14.01	4.62	1.22
CoG: coarse-to-fine cumulative ($T_l \to T_{l:m} \to T$)	**96.73**	**35.38**	**10.16**
Grok v3 segmenter			
Fine-to-coarse cumulative ($T_s \to T_{s:m} \to T$)	79.78	28.92	7.85
Coarse-to-fine non-cumulative ($T_l \to T_m \to T_s$)	16.00	6.15	1.78
CoG: coarse-to-fine cumulative ($T_l \to T_{l:m} \to T$)	**100.00**	**36.62**	**10.00**

Turning to prompt segmentation quality across different LLMs, we find that Grok v3 generally provides the most reliable results, correctly identifying the primary scaffold, functional groups, and fine-grained details in complex prompts. In contrast, GPT-4o occasionally mixes core and peripheral elements, sometimes overemphasizing atomic-level information too early or revisiting the scaffold at inappropriate stages (c.f. Appendix B for comparison).

6. Conclusion

We presented **Chain-of-Generation (CoG)**, a training-free multi-stage latent diffusion strategy for compositional text-conditioned molecular generation. The CoG inference strategy has three components: a chemistry-informed prompt factorization, cumulative conditioning across stages, and partial denoising from the previous-stage latent. Across benchmark, out-of-distribution, and ablation settings, CoG improves HM and Q-Cov over one-shot conditioning on the tested latent graph-diffusion backbones, with 100% observed validity throughout. This strategy enhances semantic fidelity and substructure alignment by explicitly linking prompt components to generated molecular motifs and structural features.

Limitations. CoG focuses on compositional structural prompts and does not address property-driven molecular optimization, binding-affinity constraints, or hard physicochemical thresholds. However, we view structural fidelity as a prerequisite for reliable downstream drug discovery applications, as molecular structure fundamentally encodes functional and physicochemical properties. Our evaluation depends on the quality of prompt segmentation, and metrics are graph-level structural-fidelity proxies rather than real chemist-level semantic judgments.

References

Beltagy, I., Lo, K., and Cohan, A. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.

Bento, A. P., Hersey, A., Félix, E., Landrum, G., Gaulton, A., Atkinson, F., Bellis, L. J., De Veij, M., and Leach, A. R. An open source chemical structure curation pipeline using rdkit. *Journal of Cheminformatics*, 12(1):51, 2020.

Brown, N., Fiscato, M., Segler, M. H., and Vaucher, A. C. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.

Chang, J. and Ye, J. C. Ldmol: Text-to-molecule diffusion model with structurally informative latent space. *arXiv preprint arXiv:2405.17829*, 2024.

Christen, P., Hand, D. J., and Kirielle, N. A review of the f-measure: its history, properties, criticism, and alternatives. *ACM Computing Surveys*, 56(3):1–24, 2023.

Christofidellis, D., Giannone, G., Born, J., Winther, O., Laino, T., and Manica, M. Unifying molecular and textual representations via multi-task language modelling. In *International Conference on Machine Learning*, pp. 6140–6157. PMLR, 2023.

Durant, J. L., Leland, B. A., Henry, D. R., and Nourse, J. G. Reoptimization of mdl keys for use in drug discovery. *Journal of chemical information and computer sciences*, 42(6):1273–1280, 2002.

Edwards, C., Zhai, C., and Ji, H. Text2mol: Cross-modal molecule retrieval with natural language queries. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 595–607, 2021.

Edwards, C., Lai, T., Ros, K., Honke, G., Cho, K., and Ji, H. Translation between molecules and natural language. *arXiv preprint arXiv:2204.11817*, 2022.

Google DeepMind. Gemini 2.5 Pro, 2024. https://deepmind.google/technologies/gemini/.

Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Jin, W., Barzilay, R., and Jaakkola, T. Hierarchical generation of molecular graphs using structural motifs. In *International conference on machine learning*, pp. 4839–4848. PMLR, 2020.

Kingma, D. P., Welling, M., et al. Auto-encoding variational bayes, 2013.

Liu, S., Nie, W., Wang, C., Lu, J., Qiao, Z., Liu, L., Tang, J., Xiao, C., and Anandkumar, A. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nature Machine Intelligence*, 5(12):1447–1457, 2023a.

Liu, Z., Li, S., Luo, Y., Fei, H., Cao, Y., Kawaguchi, K., Wang, X., and Chua, T.-S. Molca: Molecular graph-language modeling with cross-modal projector and uni-modal adapter. *arXiv preprint arXiv:2310.12798*, 2023b.

Naeem, M. F., Oh, S. J., Uh, Y., Choi, Y., and Yoo, J. Reliable fidelity and diversity metrics for generative models. In *International conference on machine learning*, pp. 7176–7185. PMLR, 2020.

O'Hagan, S. and Kell, D. B. Understanding the foundations of the structural similarities between marketed drugs and endogenous human metabolites. *Frontiers in pharmacology*, 6:105, 2015.

Pei, Q., Zhang, W., Zhu, J., Wu, K., Gao, K., Wu, L., Xia, Y., and Yan, R. Biot5: Enriching cross-modal integration in biology with chemical knowledge and natural language associations. *arXiv preprint arXiv:2310.07276*, 2023.

Polykovskiy, D., Zhebrak, A., Sanchez-Lengeling, B., Golovanov, S., Tatanov, O., Belyaev, S., Kurbanov, R., Artamonov, A., Aladinskiy, V., Veselov, M., et al. Molecular sets (moses): a benchmarking platform for molecular generation models. *Frontiers in pharmacology*, 11:565644, 2020.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Sajjadi, M. S., Bachem, O., Lucic, M., Bousquet, O., and Gelly, S. Assessing generative models via precision and recall. *Advances in neural information processing systems*, 31, 2018.

Simonovsky, M. and Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pp. 412–422. Springer, 2018.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Weininger, D. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

xAI. Grok v3, 2024. https://x.ai/blog/grok-3.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Zeng, Z., Yao, Y., Liu, Z., and Sun, M. A deep-learning system bridging molecule structure and biomedical text with comprehension comparable to human professionals. *Nature communications*, 13(1):862, 2022.

Zhu, H., Xiao, T., and Honavar, V. G. 3m-diffusion: Latent multi-modal diffusion for language-guided molecular structure generation. In *First Conference on Language Modeling*, 2024.

A. Construction of the GraphLDM Backbone

This appendix describes how GraphLDM, the second backbone used in our experiments (Section 5.2), is constructed. GraphLDM extends a molecular graph latent diffusion model with a contrastive alignment step between the molecular latent and the corresponding text embedding. This construction is independent of CoG, and the CoG sampling procedure of Algorithm 1 is unchanged regardless of which backbone is used.

Setup. The latent vector g that the diffusion model denoises is sampled, via the reparameterization trick, from the variational posterior of the molecular graph encoder, and serves as the diffusion target during training and the diffusion output at inference. To couple this latent space to the corresponding text-embedding space, GraphLDM adds a contrastive objective that runs after the encoder/decoder have been pre-trained.

We perform contrastive learning between the resampled latent g and the text embedding extracted from SciBERT (Beltagy et al., 2019). The graph encoder is frozen during this stage so that the latent representation used as the diffusion target remains fixed. Given a batch $\mathcal{B}$ of molecule–text pairs, contrastive learning is applied *after* sampling g:

$$\mathcal{L}_{\text{con}} = -\frac{1}{|\mathcal{B}|} \sum_{(G_i,T_i)\in\mathcal{B}} \log \frac{\exp\left(\cos(\mathbf{g}_i, \mathbf{c}_i)/\tau\right)}{\sum_{j=1}^{|\mathcal{B}|} \exp\left(\cos(\mathbf{g}_i, \mathbf{c}_j)/\tau\right)},$$

where $\mathbf{g}_i = E_g(G_i)$ and $\mathbf{c}_i = E_t(T_i)$ are the molecular-graph and text-prompt representations, $\cos(\cdot)$ is cosine similarity, and τ is the temperature. We refer to the resulting backbone as GraphLDM. CoG is applied identically on top of GraphLDM and on top of 3M-Diffusion; results for both backbones are reported in Table 1, Table 2, and Section F.

B. Visualization and Segmentation across LLMs

As illustrated in Figure 4, GPT-4o occasionally fails to follow the correct order, resulting in the CoG process prematurely focusing on small decorative components during the early coarse stage, and subsequently struggling to reorganize the intermediate results to accommodate the core scaffold in the finer stages. In contrast, Grok v3 produces more stable coarse-to-fine decompositions on the same prompts.

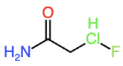
Primary structure?	GPT 4o	Gemini 2.5 Pro	Grok v3
	P		
	I		
	F	F	

Figure 4. Segmentation results across different LLMs

Table 4. Comparison of system prompt designs across different LLMs (%). The **validity** metric is omitted: all graph-based methods achieve 100% observed validity in our setting.

LLM	Prompting Strategy	HM	Q-Cov	Q-Nov
	One-shot (T)	87.40	31.69	10.03
One piece	GPT-4o One-shot (T_l)	24.20	9.23	2.43
	GPT-4o One-shot$^-$ (T_l)	24.22	9.23	2.34
	Gemini 2.5 Pro One-shot (T_l)	32.23	12.31	3.18
	Gemini 2.5 Pro One-shot$^-$ (T_l)	24.20	9.23	2.43
	Grok v3 One-shot (T_l)	32.27	12.31	3.29
	Grok v3 One-shot$^-$ (T_l)	40.32	15.38	4.39
Two pieces	GPT-4o One-shot (T_l) + CoG ($T_{l:m}$)	68.48	24.62	6.68
	GPT-4o One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$)	64.11	23.38	6.37
	Gemini 2.5 Pro One-shot (T_l) + CoG ($T_{l:m}$)	77.36	30.15	8.13
	Gemini 2.5 Pro One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$)	56.03	20.92	5.79
	Grok v3 One-shot (T_l) + CoG ($T_{l:m}$)	88.29	31.08	8.07
	Grok v3 One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$)	51.24	19.08	5.21
Three pieces	GPT-4o One-shot (T_l) + CoG ($T_{l:m}$) + CoG	87.32	31.07	8.42
	GPT-4o One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$) + CoG (T)	91.91	33.54	9.38
	Gemini 2.5 Pro One-shot (T_l) + CoG ($T_{l:m}$) + CoG (T)	96.73	35.38	**10.16**
	Gemini 2.5 Pro One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$) + CoG (T)	93.51	34.15	9.28
	Grok v3 One-shot (T_l) + CoG ($T_{l:m}$) + CoG (T)	**100.00**	**36.62**	10
	Grok v3 One-shot$^-$ (T_l) + CoG$^-$ ($T_{l:m}$) + CoG (T)	98.34	35.69	9.70

C. System Prompts Design

Prompt segmentation matters for CoG because the sampler conditions on the cumulative sub-prompts produced by the segmenter. The effectiveness of CoG depends on whether the LLM can decompose a molecular description into components in a coarse-to-fine order. When this decomposition is correct, the sampler benefits from cumulative conditioning; when it is wrong (e.g. a peripheral substituent is placed before the scaffold), errors propagate. The system prompt given to the LLM affects this decomposition quality.

In this section, we compare two designs of system prompts: one with detailed component definitions and one without them ($^-$). As shown in Table 4, Grok v3 with the detailed component definitions shows a stable performance. The performance gap across different LLMs can be attributed to their varying abilities to understand the text description and accurately segment it into a coarse-to-fine hierarchy.

```
System Prompt without detailed instruction:

You are a highly capable AI tasked with generating four text outputs (A only, AB, C
only, BC) by segmenting a single molecule's text description provided as {DESCRIPTION}.
 The description includes three components (A, B, C) in a syntactic form such as "The
molecule contains..." or "The molecule is made of...". Your task is to analyze the
description, identify components A, B, and C based on their roles, and generate four
outputs with modified text descriptions containing the specified component subsets.

Output Generation Rules:

A only: Include only the A component in the text description.
AB: Include both A and B components in the text description.
C only: Include only the C component in the text description.
BC: Include both B and C components in the text description.

Instructions:

Read the text description {DESCRIPTION} to identify A, B, and C based on their
significance in the molecule.
Write the new description in a consistent format, e.g., "The molecule is made of [
component(s)]", ensuring grammatical correctness.
If components are ambiguous, prioritize the most significant part as A, the next as B,
```

```
 and the least significant as C.
```

```
System Prompt with detailed instruction:

You are a highly capable AI tasked with generating four text outputs (A only, AB, C
only, BC) by segmenting a single molecule's text description provided as {DESCRIPTION}.
 The description includes three components (A, B, C) in a syntactic form such as "The
molecule contains..." or "The molecule is made of...". Your task is to analyze the
description, extract the components A, B, and C based on their semantic roles, and
generate four outputs with modified text descriptions containing the specified
component subsets. The output should include only the modified text descriptions.

Component Definitions:

A: The main structural core or primary ring system (e.g., benzene ring, naphthalene
ring, pyrimidine ring) or the most central atom in smaller molecules (e.g., carbon,
nitrogen, phosphorus).
B: Medium-sized functional groups or secondary ring systems directly attached to the
core (e.g., amide group, phosphate group, nitro group, cyclohexane ring).
C: Small atomic or functional group modifiers attached to A or B (e.g., fluorine,
chlorine, bromine, iodine, hydroxyl group).

Output Generation Rules:

From heavy to light:
A only: Include only the A component in the text description.
AB: Include both A and B components in the text description.
From light to heavy:
C only: Include only the C component in the text description.
BC: Include both B and C components in the text description.

For the input description:
Read the text description {DESCRIPTION} to identify A, B, and C based on their
structural significance, regardless of the syntactic structure.
Write the new description in a consistent format, e.g., "The molecule is made of [
component(s)]", ensuring grammatical correctness.
If a component appears ambiguous, prioritize the largest ring system or core as A, the
 next significant group as B, and the smallest modifier as C.
```

```
System Prompt on real datasets:

You are a highly capable AI tasked with analyzing a molecule's text description
provided as {DESCRIPTION} and outputting a single number representing the count of
distinct components, along with increment of textual segmentations. The description may
 include phrases like "The molecule is made of..." or similar syntactic forms
describing molecular components. Your task is to segment the description into
meaningful components based on their semantic roles and provide the reasoning for your
 segmentation.

Component Identification Guidelines

To determine the number of components, consider the following candidate definitions as
 a flexible framework. Use your reasoning to adapt these definitions or create new
ones based on the specific description provided:

Core Structure: The primary scaffold or backbone (e.g., glutamyl-alanine, tyrosine,
glycerone) counts as 1 component.
Functional Groups: Each distinct functional group (e.g., hydroxy, methyl, oxo, amino)
is counted separately, even if multiple instances exist (e.g., three hydroxy groups =
three components).
Modifiers: Each distinct modifier (e.g., ionic forms, complex substituents like
methylthioethyl, stereochemical configurations) is counted separately.
Stereochemistry: Enantiomeric or stereochemical configurations (e.g., L-enantiomer, (2
S,4S)-configuration) are counted as modifiers if explicitly described as part of the
```

```
molecule's structure.
Chemical Properties: Descriptive attributes of the molecule, such as acidity, basicity,
 or physical state (e.g., solid, liquid, gas).
You are free to refine or reinterpret these categories based on the description's
context.

Output Format:
For a molecule with n components, the output line includes:
The integer n (total component count).
A tab.
A fluent sentence for part 1 (core).
A tab.
A fluent sentence for parts 1+2 (core + first modifier/functional group).
A tab.
A fluent sentence for parts 1+2+3, and so on, up to all n parts.
Sentences are concise, chemically accurate, and cumulative, incorporating all prior
components in each subsequent sentence.

Component Counting: Each core, functional group, and modifier was counted separately,
without grouping similar substituents.

Example:
Input: "The molecule is a 1,8-naphthyridine derivative that is 4-oxo-1,4-dihydro-1,8-
naphthyridine-3-carboxylic acid bearing additional 2,4-difluorophenyl, fluoro and 3-
aminopyrrolidin-1-yl substituents at positions 1, 6 and 7 respectively. It is a 1,8-
naphthyridine derivative, an amino acid, a monocarboxylic acid, an organofluorine
compound, an aminopyrrolidine, a tertiary amino compound, a primary amino compound and
 a quinolone antibiotic. It is a conjugate base of a 1-[6-carboxy-8-(2,4-
difluorophenyl)-3-fluoro-5-oxo-5,8-dihydro-1,8-naphthyridin-2-yl]pyrrolidin-3-aminium."


Output:
Number: 5
Segmentations: The molecule is a heterocycle with a naphthyridine core.
The molecule is a heterocycle with a naphthyridine core, substituted by a 2,4-
difluorophenyl group at position 1.
The molecule is a heterocycle with a naphthyridine core, substituted by a 2,4-
difluorophenyl group at position 1 and a fluoro group at position 6.
The molecule is a heterocycle with a naphthyridine core, substituted by a 2,4-
difluorophenyl group at position 1, a fluoro group at position 6, and a 3-
aminopyrrolidin-1-yl group at position 7.
The molecule is a heterocycle with a naphthyridine core, substituted by a 2,4-
difluorophenyl group at position 1, a fluoro group at position 6, a 3-aminopyrrolidin
-1-yl group at position 7, and a carboxy group at position 3 to form a quinolone
antibiotic.

Constraints:
Ensure the segmentation is concise, chemically informed, fluent, and directly tied to
the description. If the description is ambiguous, make reasonable assumptions.
```

D. Evaluation Metrics

Let $G^\star$ denote the reference molecule for a prompt and $\{G_i\}_{i=1}^n$ the set of n molecules generated for that prompt. Let

$$s_i \;=\; \mathrm{Tan}_{\mathrm{MACCS}}(G_i,\, G^\star)$$

be the MACCS/Tanimoto structural similarity between G_i and $G^\star$, and let $V_i \in \{0, 1\}$ indicate whether G_i decodes to a chemically valid graph (RDKit rules).

We define a broad structural-agreement rate, a qualified structural-agreement rate, and a pairwise-distance term over the broad set:

$$R_{\mathrm{basic}} \;=\; \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[s_i > 0.5], \qquad R_{\mathrm{qual}} \;=\; \frac{1}{n}\,\big|\{i : 0.5 < s_i < 0.8\}\big|,$$

$$\mathcal{B} \;=\; \{i : s_i > 0.5\}, \qquad D_{\mathcal{B}} \;=\; \frac{2}{|\mathcal{B}|(|\mathcal{B}|-1)} \sum_{\substack{i<j \\ i,j\in\mathcal{B}}} \big(1 - \mathrm{Tan}_{\mathrm{MACCS}}(G_i, G_j)\big).$$

The four reported numbers are then

$$\text{Validity} \;=\; \frac{1}{n}\sum_{i=1}^{n} V_i, \qquad \text{HM} \;=\; \frac{2\,R_{\text{basic}}\,R_{\text{qual}}}{R_{\text{basic}} + R_{\text{qual}}},$$

$$\text{Q-Cov} \;=\; \text{Validity} \times R_{\text{qual}}, \qquad \text{Q-Nov} \;=\; R_{\text{qual}} \times D_{\mathcal{B}}.$$

Validity and HM are quality terms; Q-Cov tracks how often valid generations land in the qualified band; Q-Nov tracks pairwise structural diversity within the broad set, weighted by the qualified-band rate. Q-Nov is therefore a qualified-rate-weighted diversity score over structurally agreeing generations, not novelty against the training set. If a denominator is zero (e.g., $\mathcal{B}$ is empty so $D_{\mathcal{B}}$ is undefined, or $R_{\text{basic}} + R_{\text{qual}} = 0$ so HM is undefined), the corresponding score is set to 0.

On the qualified band $0.5 < s_i < 0.8$. The qualified band is used to count non-trivial structural variants that retain substantial graph-level agreement with the reference. It should not be read as saying that outputs with $s_i \geq 0.8$ are incorrect: such high-similarity outputs still contribute to the broad agreement term R_{basic}, and therefore to HM. The metrics should be interpreted jointly as graph-level structural-fidelity proxies, not as a complete measure of prompt correctness.

E. Computational Cost

We report computational cost so that practitioners can budget the trade-off; we do not claim it is negligible in absolute terms.

Diffusion-sampling cost. All diffusion experiments run on a single NVIDIA RTX A5000 GPU. In the one-shot setting, generating 500 samples with the latent graph-diffusion backbone takes approximately 40 seconds. Each additional CoG stage adds approximately 25 seconds (about 0.05 seconds per sample). Our default CoG configuration uses three stages (Section 5.3), so total diffusion-sampling time is approximately 90 seconds for 500 samples on the same GPU.

LLM segmentation latency (caveat). The numbers above measure diffusion sampling only and do *not* include the one-time prompt-segmentation latency of the LLM segmenter (e.g., a Grok v3, GPT-4o, or Gemini 2.5 Pro API call). API latency depends on the LLM provider, prompt length, and network conditions, and is not measured in our experiments. Segmentation is performed once per prompt and then cached, so its cost is amortized across all samples generated from the same prompt.

CoG itself remains training-free and integrates with existing latent graph-diffusion pipelines as an inference-time wrapper.

F. Performance on Benchmark Subsets

To apply CoG on benchmark datasets, we segment each prompt into coarse, intermediate, and fine-grained structural levels when applicable: core scaffold, functional groups or substituents, and finer modifiers such as atoms, charges, or stereochemical descriptors. We use Grok v3 for this segmentation, with the reasoning prompt in Section C. As shown in Figure 5 and Figure 6, most prompts on these benchmarks can be segmented into two or three pieces. Results across different segmentation subsets are summarized in Tables 5 to 11. In all subset tables, + CoG rows apply CoG to the backbone immediately above; superscript deltas show gains from adding CoG.

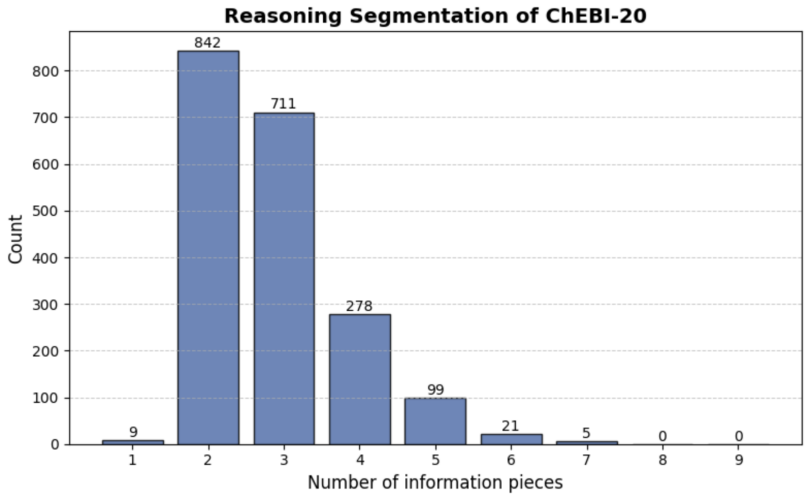


Figure 5. Distribution of number of information pieces after reasoning segmentation on ChEBI-20

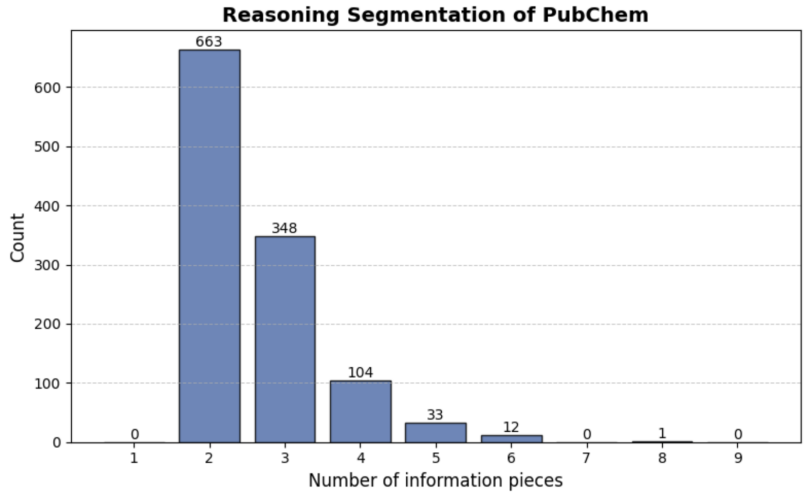


Figure 6. Distribution of number of information pieces after reasoning segmentation on PubChem

Table 5. Performance on the ChEBI-20 subset where each textual prompt contains five pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	69.47	24.49	7.07	66.33
MolT5 large	49.98	20.41	3.04	97.96
ChemT5 small	68.30	30.61	5.85	94.9
ChemT5 base	54.41	22.45	4.46	95.92
BioT5 base	70.33	34.29	13.22	100
BioT5 plus	60.24	26.73	7.32	100
LDMol	3.30	42.86	11.97	100
Graph-based methods				
3M-Diffusion	68.87	44.29	12.84	100
+CoG	77.48 $^{+8.61}$	53.27 $^{+8.98}$	14.61 $^{+1.77}$	100
GraphLDM	88.10	58.16	**17.49**	100
+CoG	**90.58** $^{+2.48}$	**59.39** $^{+1.23}$	16.98 $^{-0.51}$	100

Table 6. Performance on the ChEBI-20 subset where each textual prompt contains four pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	79.31	27.79	7.84	68.95
MolT5 large	71.01	29.24	4.39	97.11
ChemT5 small	69.14	27.8	4.79	96.03
ChemT5 base	71.07	28.88	5.19	96.39
BioT5 base	72.51	31.19	9.62	100
BioT5 plus	53.24	19.71	4.31	100
LDMol	11.45	41.73	11.43	100
Graph-based methods				
3M-Diffusion	70.49	43.68	11.63	100
+CoG	74.23 $^{+3.74}$	45.85 $^{+2.17}$	12.06 $^{+0.43}$	100
GraphLDM	82.09	50.40	**14.80**	100
+CoG	**85.08** $^{+2.99}$	**51.55** $^{+1.15}$	13.81 $^{-0.99}$	100

Table 7. Performance on the ChEBI-20 subset where each textual prompt contains three pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	78.31	26.06	5.47	77.61
MolT5 large	61.55	20.28	2.29	98.03
ChemT5 small	69.15	23.66	3.26	97.04
ChemT5 base	72.91	25.92	3.6	97.18
BioT5 base	62.33	20.90	4.20	100
BioT5 plus	51.10	15.71	2.47	100
LDMol	8.95	41.15	10.43	100
Graph-based methods				
3M-Diffusion	71.78	40.23	10.06	100
+CoG	75.94 $^{+4.16}$	42.90 $^{+2.67}$	10.30 $^{+0.24}$	100
GraphLDM	81.28	43.44	**14.80**	100
+CoG	**84.04** $^{+2.76}$	**45.38** $^{+1.94}$	11.44 $^{-3.36}$	100

Table 8. Performance on the ChEBI-20 subset where each textual prompt contains two pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	68.25	21.52	3.33	83.71
MolT5 large	58.12	17.47	1.38	98.45
ChemT5 small	68.35	22	2.43	97.5
ChemT5 base	70.79	23.43	2.64	97.74
BioT5 base	53.98	15.91	2.21	99.97
BioT5 plus	45.19	12.49	1.30	100
LDMol	16.65	**42.40**	**9.88**	100
Graph-based methods				
3M-Diffusion	72.58	38.83	8.81	100
+CoG	75.04 $^{+2.46}$	40.57 $^{+1.74}$	9.16 $^{+0.35}$	100
GraphLDM	76.69	38.53	9.10	100
+CoG	**82.18** $^{+5.49}$	42.28 $^{+3.75}$	9.23 $^{+0.13}$	100

Table 9. Performance on the PubChem subset where each textual prompt contains four pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	49.31	14.56	1.01	81.55
MolT5 large	41.06	13.6	0.85	99.03
ChemT5 small	51.71	17.48	1.95	94.17
ChemT5 base	56.52	20.39	2.82	97.09
BioT5 base	39.09	12.82	1.95	100
BioT5 plus	30.51	9.51	1.24	100
LDMol	13.68	41.55	10.92	100
Graph-based methods				
3M-Diffusion	76.34	49.51	13.04	100
+CoG	**80.92** $^{+4.58}$	**52.43** $^{+2.92}$	14.35 $^{+1.31}$	100
GraphLDM	69.61	41.17	12.96	100
+CoG	76.03 $^{+6.42}$	48.16 $^{+6.99}$	**14.69** $^{+1.73}$	100

Table 10. Performance on the PubChem subset where each textual prompt contains three pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	49.04	12.1	0.98	80.4
MolT5 large	40.42	11.23	0.65	97.69
ChemT5 small	56.94	17.58	1.79	97.12
ChemT5 base	64.13	20.46	2.21	95.39
BioT5 base	44.12	12.85	1.71	100
BioT5 plus	40.74	11.56	1.39	100
LDMol	21.08	39.08	9.99	100
Graph-based methods				
3M-Diffusion	71.37	41.67	11.31	100
+CoG	72.56 $^{+1.19}$	42.77 $^{+1.10}$	10.70 $^{-0.61}$	100
GraphLDM	75.76	43.40	**13.66**	100
+CoG	**78.35** $^{+2.59}$	**44.84** $^{+1.44}$	12.25 $^{-1.41}$	100

Table 11. Performance on the PubChem subset where each textual prompt contains two pieces of information. All values are reported as percentages.

Method	HM	Q-Cov	Q-Nov	Validity
SMILES-based methods				
MolT5 small	48.13	9.67	1.08	67.67
MolT5 large	63.50	19.33	1.96	91.54
ChemT5 small	69.19	21.75	2.9	88.82
ChemT5 base	68.51	21	3.03	87.16
BioT5 base	59.71	19.46	3.66	100
BioT5 plus	63.33	21.21	4.25	100
LDMol	54.72	34.95	8.87	100
Graph-based methods				
3M-Diffusion	67.76	37.76	10.15	100
+CoG	73.86 $^{+6.10}$	41.39 $^{+3.63}$	10.30 $^{+0.15}$	100
GraphLDM	74.66	39.52	**11.92**	100
+CoG	**78.12** $^{+3.46}$	**41.84** $^{+2.32}$	11.37 $^{-0.55}$	100

G. Visualization of Generated Samples

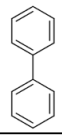
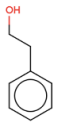
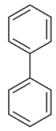
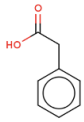
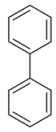
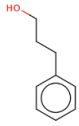
	ChemT5-base augment	MolT5-large	GraphLDM
seed 1	C1=CC=C2C=C1 ✗	 ✓	 ✓
seed 2	C1=CC=C2C=C1 ✗	 ✓	 ✓
seed 3	C1=CC=C2C=C1 ✗	 ✓	 ✓

Figure 7. Sequence-to-sequence methods struggle to produce valid and diverse outputs when given prompts such as "The molecule is a simple structure that contains a benzene ring." In particular, ChemT5 and MolT5 exhibit severe mode collapse: despite the large solution space implied by flexible prompts, they repeatedly converge to a single rigid output. A second limitation lies in chemical invalidity, as exemplified by the ChemT5-base augment model.