Aligned Diffusion Models for Retrosynthesis

Najwa Laabid [*1] **Severi Rissanen** [*1] **Markus Heinonen** [1] **Arno Solin** [1] **Vikas Garg** [1 2]

Abstract

Retrosynthesis, the task of identifying precursors for a given molecule, can be naturally framed as a conditional graph generation task, with diffusion models being a particularly promising approach. We show mathematically that permutation equivariant denoisers severely limit the expressiveness of graph diffusion models and thus their adaptation to retrosynthesis. To address this limitation, we relax the equivariance requirement such that it only applies to *aligned permutations* of the conditioning and the generated graphs obtained through atom mapping, resulting in a diffusion model with state-of-the-art results in template-free retrosynthesis.

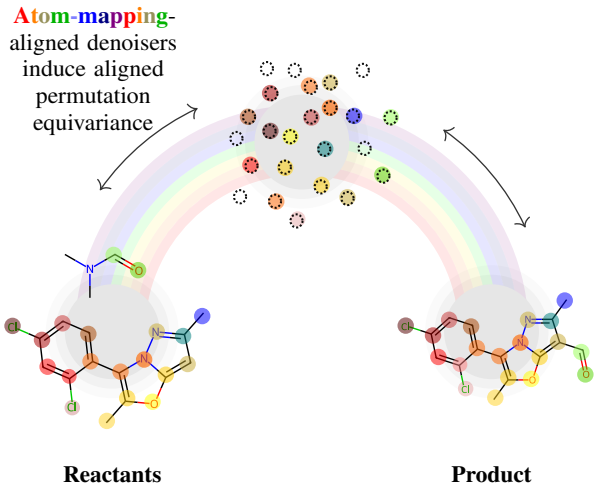


Figure 1: **Aligned permutation equivariance.** We use atom-mapping information (visualized by highlighting atoms with matching colors) as an input to the neural network denoiser to generalize permutation equivariance to *aligned* permutation equivariance between reactant and product graphs.

1. Introduction

Single-step retrosynthesis plays a pivotal role in chemistry, as it focuses on identifying appropriate precursors for a given target compound which can then be chained to form complex synthesis plans (Corey and Cheng, 1996). Approaches to retrosynthesis have steadily moved from relying on explicit expert-curated rules or simplifying assumptions (template-based (Corey and Wipke, 1969) and synthon-based (Shi et al., 2020; Somnath et al., 2021)), to adopting purely data-driven machine-learning models (template-free (Wan et al., 2022; Igashov et al., 2024)).

Diffusion-based generative models promise several advantages in template-free retrosynthesis. For instance, trading off generation speed and quality on a per sample basis (Song et al., 2021) and the possibility to add additional conditions even post-training (Song et al., 2021; Chung et al., 2023; Li et al., 2022) could be used by a multi-step retrosynthetic planner as additional controls. Furthermore, a simple adaptation of diffusion models to retrosynthesis holds the promise of all future advances diffusion model methodology being opened up to the retrosynthesis domain as well. To enable all of these advantages, however, we need to understand how to properly apply diffusion models in this graph-conditional setting.

In this paper: **1) We characterize a serious limitation of neural network expressivity** that occurs when using standard permutation equivariant denoiser architectures for graph diffusion models in retrosynthesis. **2) We introduce *aligned permutation equivariance***, where the reactants and products are aligned with atom-mapping information, and equivariance to permutations only holds when the alignment through atom mapping remains intact (see Fig. 1 with an example sketch). **3) We propose multiple methods to achieve this alignment** with standard GNN-based denoisers. **4) We show that our method achieves excellent top-k accuracy**, with state-of-the-art results in template free retrosynthesis.

[*]Equal contribution [1]Department of Computer Science, Aalto University, Espoo, Finland [2]YaiYai Ltd, Finland. Correspondence to: Najwa Laabid <najwa.laabid@aalto.fi>, Severi Rissanen <severi.rissanen@aalto.fi>.

Accepted as an extended abstract for the Geometry-grounded Representation Learning and Generative Modeling Workshop at the 41st International Conference on Machine Learning, ICML 2024, Vienna, Austria.

2. Related Work

There are three main types of retrosynthesis models (Liu et al., 2023): Template-based (Segler and Waller, 2017; Xie et al., 2023) and synthon-based (Yan et al., 2020; Shi et al., 2020; Wang et al., 2023) models use explicit chemical rules, also restricting their generality. Template-free methods (Wan et al., 2022; Seo et al., 2021) do not make these assumptions, and our work falls in this category. Recently, Igashov et al. (2024) proposed a Schrödinger-bridge inspired method for retrosynthesis sharing similarities with diffusion models. In another concurrent work (Wang et al., 2023), the authors propose to use a synthon-based hierarchical diffusion model for multi-stage generation.

A fundamental inductive bias with neural networks operating on graphs is *permutation equivariance* (Wu et al., 2021), which ensures that the network only uses topological connectivity information and does not process the order of nodes in an input graph $\mathbf{X}$. It is widely used in graph diffusion models to parameterize the reverse process (Niu et al., 2020; Vignac et al., 2023; Hoogeboom et al., 2022; Huang et al., 2022). Recently, Yan et al. (2023) showed that relaxing permutation equivariance in graph diffusion using absolute positional encodings can improve performance empirically.

3. Methods

We consider a database of N_{obs} known chemical reactions $\mathcal{D} = \{(\mathbf{X}_n, \mathbf{Y}_n, \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}})\}_{n=1}^{N_{\text{obs}}}$, where $\mathbf{X}_n$ are reactants, $\mathbf{Y}_n$ are products and $\mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}}$ are matrices defining *atom mappings* between products and reactants. The *retrosynthesis* task is: given that the data is sampled from an unknown distribution $p(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$, predict the valid reactant molecules $\mathbf{X} \sim p(\mathbf{X} \mid \mathbf{Y})$ for a given product molecule $\mathbf{Y}$. In our data encoding (also illustrated in Fig. A4 in App. A) we model the reactants and products as molecular graphs with nodes and edges representing atoms and bonds, respectively. Formally, we define the reactant graph $\mathbf{X}$ as a tuple $(\mathbf{X}^{\mathcal{N}}, \mathbf{X}^{E})$ of a node feature matrix $\mathbf{X}^{\mathcal{N}} \in \mathbf{R}^{N_X \times K_a}$ and an edge feature matrix $\mathbf{X}^{E} \in \mathbf{R}^{N_X \times N_X \times K_b}$, s.t. K_a and K_b are atom and edge feature dimensions respectively. The product graph $\mathbf{Y}$ is similarly defined as $(\mathbf{Y}^{\mathcal{N}}, \mathbf{Y}^{E})$ with a potentially different number of nodes N_Y. The node features are one-hot vectors over atom types $(\perp, \mathrm{C}, \mathrm{N}, \mathrm{O}, \mathrm{P}, \ldots)$ and edges are one-hot vectors over bond types $(\perp, 1, 2, 3)$. Both include an empty value $\perp$ that represents a missing node or edge. Usually, $N_X \geq N_Y$, as the data is defined to only include the main product molecule.

The atom mapping matrices $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}} \in \mathbf{R}^{N_X \times N_Y}$ give us additional information about how atoms that ended up in the product were reconfigured in the reaction, and are defined such that $\mathbf{P}_{i,j}^{\mathbf{Y}\to\mathbf{X}} = 1$ if the i^{th} atom of the reactant corresponds with the j^{th} atom of the product, and zero otherwise. Thus, $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^{\mathcal{N}}$ equals $\mathbf{X}^{\mathcal{N}}$ with the non-atom-mapped atoms zeroed out. Correspondingly, $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^{E}(\mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^{\top}$ equals $\mathbf{X}^{E}$ with edges to non-atom-mapped atoms zeroed out.

We propose to model the problem with a discrete variant of the standard denoising diffusion probabilistic model (DDPM, (Ho et al., 2020)) where the reactants $\mathbf{X}_0$ are diffused in a Markov chain $q(\mathbf{X}_t \mid \mathbf{X}_{t-1})$ for $t \in [1, T]$, and we learn how to transform noise back to reactants $p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y})$, while conditioning on the fixed product $\mathbf{Y}$. Specifically, we adapt discrete diffusion models (Sohl-Dickstein et al., 2015; Hoogeboom et al., 2022; Austin et al., 2021) to graphs, following Vignac et al. (2023).

We use a discrete diffusion framework, following (Austin et al., 2021; Vignac et al., 2023) with a Markov *forward* process on the nodes and edges of the graph

$$q(\mathbf{X}_{t+1} \mid \mathbf{X}_t) = \textstyle\prod_{i=1}^{N_X} q(\mathbf{X}_{t+1}^{\mathcal{N},i} \mid \mathbf{X}_t^{\mathcal{N},i}) \prod_{i,j=1}^{N_X} q(\mathbf{X}_{t+1}^{E,ij} \mid \mathbf{X}_t^{E,ij}), \tag{1}$$

to diffuse the reactant to noise, and a reverse process

$$p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t) = \textstyle\prod_{i=1}^{N_X} p_\theta(\mathbf{X}_{t-1}^{\mathcal{N},i} \mid \mathbf{X}_t, \mathbf{Y}) \prod_{i,j}^{N_X} p_\theta(\mathbf{X}_{t-1}^{E,ij} \mid \mathbf{X}_t, \mathbf{Y}), \tag{2}$$

defining our generative model. The full generative distribution is $p_\theta(\mathbf{X}_{0:T} \mid \mathbf{Y}) = p(\mathbf{X}_T) \prod_{t=1}^{T} p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y})$, where $p(\mathbf{X}_T)$ is a predefined prior such that $p(\mathbf{X}_T) = q(\mathbf{X}_T \mid \mathbf{X}_0)$. We use the neural network specifically to predict ground truth labels from noised samples, so that the neural net directly outputs a distribution $\tilde{p}_\theta\big(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}\big)$. Following (Vignac et al., 2023), train the model by minimizing the cross-entropy between denoised $\mathbf{X}_0$ probability vectors and the true $\mathbf{X}_0$. We adopt the absorbing-state formulation from Austin et al. (2021), where nodes and edges gradually transfer to the *absorbing state*, defined as the empty state $\perp$. For more details on the discrete diffusion formalism, see App. A. Throughout the paper, we denote the direct output of the neural network as $D_\theta(\mathbf{X}_t, \mathbf{Y}) = (D_\theta(\mathbf{X}_t, \mathbf{Y})^{\mathcal{N}}, D_\theta(\mathbf{X}_t, \mathbf{Y})^{E})$, where $D_\theta(\mathbf{X}_t, \mathbf{Y})^{\mathcal{N}} \in \mathbf{R}^{N_X \times K_a}$ and $D_\theta(\mathbf{X}_t, \mathbf{Y})^{E} \in \mathbf{R}^{N_X \times N_X \times K_b}$, s.t. we have a probability vector for each node and edge. An overview of the approach is shown in Fig. 2.

3.1. Theoretical results on permutation equivariance

In this section, we consider a data set $\mathcal{D} = \{(\mathbf{X}_n, \mathbf{Y}_n, \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}})\}_{n=1}^{N_{\text{obs}}}$, dubbed the 'identity reaction data', where for all data points $\mathbf{X}_n = \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}_n$. In the data, both sides of the reaction are equivalent, up to some permutation, as defined in the atom mapping matrix $\mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}}$. We use this toy scenario to reason about the expressiveness of our denoiser neural network. Furthermore, chemical reactions often induce limited changes to the precursor molecules (Zhong et al., 2022), motivating the identity reac-

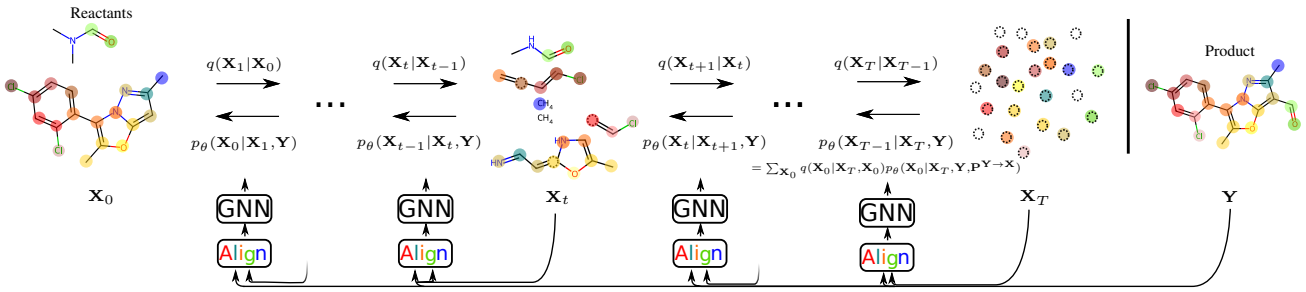


Figure 2: An overview of our approach to retrosynthetic graph diffusion. We adopt *absorbing state diffusion* (Austin et al., 2021), and use atom mapping information (highlighted with the matching colours) as an input to the neural network denoiser to align the product condition with the reactant data.

tion data as an important base case that the model should be able to handle.

In our first result, we show that the denoisers in standard graph diffusion models are constrained to output 'mean' solutions, such that the outputs for all nodes and all edges are equal in the early stages of generation. This is formalized in the following Theorem:

Theorem 1. *(The optimal permutation equivariant denoiser) Let $D_\theta(\mathbf{X}_T, \mathbf{Y})$ be permutation equivariant s.t. $D_\theta(\mathbf{P}\mathbf{X}_T, \mathbf{Y}) = \mathbf{P}D_\theta(\mathbf{X}_T, \mathbf{Y})$, and let $q(\mathbf{X}_T)$ be permutation invariant. The optimal solution with respect to the cross-entropy loss with the identity reaction data is, for all nodes i and j*

$$\begin{cases} D_\theta(\mathbf{X}_T, \mathbf{Y})^{\mathcal{N}}_{i,:} = \hat{\mathbf{y}}^{\mathcal{N}}, \hat{\mathbf{y}}^{\mathcal{N}}_k = \sum_i \mathbf{Y}_{i,k} / \sum_{i,k} \mathbf{Y}_{i,k}, \\ D_\theta(\mathbf{X}_T, \mathbf{Y})^{E}_{i,j,:} = \hat{\mathbf{y}}^{E}, \hat{\mathbf{y}}^{E}_k = \sum_{i,j} \mathbf{Y}_{i,j,k} / \sum_{i,j,k} \mathbf{Y}_{i,j,k}, \end{cases} \tag{3}$$

where $\hat{\mathbf{y}}^{\mathcal{N}}_k$ and $\hat{\mathbf{y}}^{E}_k$ are the marginal distributions of node and edge values in $\mathbf{Y}$.

The proof is given in App. B.1, and the Theorem is illustrated in **??**. Clearly, this is a severe limitation. Ideally, we would like the model to handle the copy-paste task in a single denoising step, leaving the capacity of the diffusion model for modelling the parts of the reaction that are more difficult to handle. To solve the issue, we propose to relax the standard permutation equivariance constraint to *aligned permutation equivariance*, verbally defined as: *if we permute $\mathbf{X}$ and/or $\mathbf{Y}$, and accordingly permute the atom mapping $P^{\mathbf{Y}\to\mathbf{X}}$ such that the matching between $\mathbf{Y}$ and $\mathbf{X}$ remains, the model output should be the same*. Formally, we use the atom-mapping permutation matrix $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$ as an input to the denoiser and consider denoisers that satisfy the following constraint: $D_\theta(\mathbf{R}\mathbf{X}, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \mathbf{R}D_\theta(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$, where $\mathbf{R}$ and $\mathbf{Q}$ are permutation matrices of shapes $(N_\mathbf{X} \times N_\mathbf{X})$ and $(N_\mathbf{Y} \times N_\mathbf{Y})$, respectively. With $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$ and $\mathbf{Y}$ as an input, an unconstrained denoiser can output the ground-truth permutation $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}Y$ for the identity reaction task.

The question arises: With the relaxation, do we lose the inductive bias provided permutation equivariance? In partivular, with permutation equivariant denoisers for graph diffusion, we know that if the prior for $\mathbf{X}_T$ is permutation invariant, then our distribution for $\mathbf{X}_0$ is permutation invariant as well (Niu et al., 2020; Vignac et al., 2023). In our second theoretical result, we show that a generalized permutation invariance also holds for aligned denoisers:

Theorem 2. *(Aligned denoisers induce aligned permutation invariant distributions) If the denoiser function D_θ has the aligned equivariance property and the prior $p(\mathbf{X}_T)$ is permutation invariant, then the generative distribution $p_\theta(\mathbf{X}_0 \mid \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$ has the corresponding property for any permutation matrices $\mathbf{R}$ and $\mathbf{Q}$:*

$$p_\theta(\mathbf{R}\mathbf{X}_0 \mid \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = p_\theta(\mathbf{X}_0 \mid \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}). \tag{4}$$

A proof is given in App. B.2. Informally, the theorem states that the chemical reaction has the same probability for all isomorphisms of the reactant and product graphs, as long as the atom mapping is not reassigned to different atoms. Thus, during training, we can use the permutations present in the data, and be sure that the model generalizes to other permutations. Initially, during sampling, we only have access to $\mathbf{Y}$ without atom mapping information. The theorem ensures we can assign atom mappings arbitrarily during sampling and still obtain effectively the same distribution over reactant graphs.

3.2. Methods for Alignment in GNNs

We show multiple simple methods to turn a regular, permutation equivariant denoiser into an aligned permutation equivariant denoiser. Here, we present the implementation details, and in App. B.3 we prove for each method that they belong to the class of aligned equivariant models. Fig. A5 visualizes the methods.

Atom-mapped positional encodings In the first method, we match pairs of atoms via the atom-mapping matrix

in both $\mathbf{X}_t$ and $\mathbf{Y}$ by adding a positional encoding vector to each unique atom pair. We first create the positional encodings based on the products with $\boldsymbol{\varphi} = g(\mathbf{Y})$ (g being some function) and copy them to the reactant input to the denoiser as $D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) = f_\theta([\mathbf{X}_t^{\mathcal{N}} \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\boldsymbol{\varphi}], \mathbf{X}_t^E, [\mathbf{Y}^{\mathcal{N}} \ \boldsymbol{\varphi}], \mathbf{Y}^E)$, where f_θ is the neural network that takes as inputs the augmented node features and regular edge features.

Direct skip connection In the second method, we modify the network to include a direct connection from the product to the reactant output: $D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) = \text{softmax}(f_\theta^{\text{logit}}(\mathbf{X}_t, \mathbf{Y}) + \lambda\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y})$ where $f_\theta^{\text{logit}}(\mathbf{X}_t, \mathbf{Y})$ are the logits at the last layer of the neural network for the nodes and edges, λ is a learnable parameter and $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y} = (\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^{\mathcal{N}}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^E(\mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^\top)$.

Aligning $\mathbf{Y}$ in the input In the third method, we align $\mathbf{Y}$ in the *input* by concatenating $\mathbf{X}$ and $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}$ along the feature dimension before passing it to the neural network with $D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) = f_\theta([\mathbf{X}_t^{\mathcal{N}} \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^{\mathcal{N}}], [\mathbf{X}_t^E \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}^E(\mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^\top])$.

4. Experiments

Data and preprocessing We use the reaction data benchmark data set USPTO-50k for our experiments. To generate precursor sets of various sizes, we set the reactant blocks to a fixed size of $N_Y + d$, where d is a hyperparameter we set empirically to 15, with a 'blank' node type for non-atoms. We use the graph transformer architecture introduced by (Dwivedi and Bresson, 2021) and used by (Vignac et al., 2023). More details in App. C.1, App. C.3 and App. C.5.

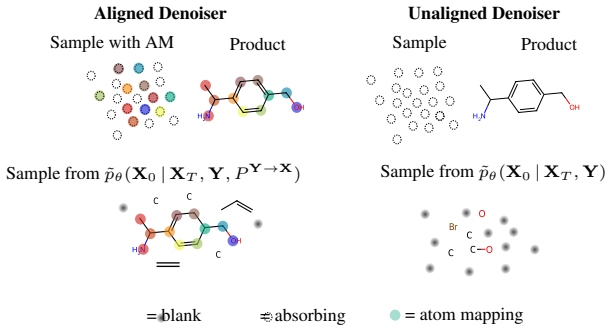
Results We experiment with 1) an equivariant model 2) a model with input alignment 3) a model with positional encoding alignment 4) a model with positional encoding alignment + output skip connection alignment. We sample 100 reactants for each condition in the test data set with $T = 100$, rank them based on the frequency and ELBO (details in App. C.2) and measure how often the ground-truth reactant was in the top-k samples. Quantitative results are in Table 1. While the unaligned, permutation equivariant model struggles to get any reactions correct, our best aligned model beats other template-free models in all top-k values, and even the template-based and synthon-based models in top-1.

We also visualize the output of an aligned denoiser vs. the output of the permutation equivariant denoiser in Fig. 1. The aligned model correctly copies the product structure to the output, whereas the unaligned model output does not have any structure.

Other results. In the Appendix, we also show that the model performs well with small step counts and show how to apply post-training conditioning mechanisms for diffusion models (inpainting and property-guided generation) for interactive applications and as additional controls for multi-step retrosynthesis.

Table 1: Top-k accuracy and MRR on the USPTO-50k test data set. For comparison to other models, including ones pretrained on larger data sets, see App. C.6.

	Method	k = 1 ↑	k = 3 ↑	k = 5 ↑	k = 10 ↑
Temp.	Retrosym (Coley et al., 2017b)	37.3	54.7	63.3	74.1
	GLN (Dai et al., 2019)	52.5	74.7	81.2	87.9
	LocalRetro (Chen and Jung, 2021)	**52.6**	**76.0**	**84.4**	**90.6**
Synthon	G2G (Shi et al., 2020)	48.9	67.6	72.5	75.5
	GraphRetro (Somnath et al., 2021)	**53.7**	68.3	72.2	75.5
	MEGAN (Sacha et al., 2021)	48.0	70.9	78.1	**85.4**
	RetroDiff (Wang et al., 2023)	**52.6**	71.2	**81.0**	83.3
Template-free	SCROP (Zheng et al., 2019)	43.7	60.0	65.2	68.7
	Tied Transformer (Kim et al., 2021)	47.1	67.1	73.1	76.3
	GTA_aug (Seo et al., 2021)	51.1	67.6	74.8	81.6
	Graph2SMILES (Tu and Coley, 2022)	52.9	66.5	70.0	72.9
	Retroformer (Wan et al., 2022)	52.9	68.2	72.5	76.4
	DualTF_aug (Sun et al., 2021)	53.6	70.7	74.6	77.0
Ours	Unaligned	4.1	6.5	7.8	9.8
	DiffAlign-input	44.1	65.9	72.2	78.7
	DiffAlign-PE	49.0	70.7	76.6	**81.8**
	DiffAlign-PE+skip	**54.7**	**73.3**	**77.8**	81.1



(a) Visualizing the output of an aligned denoiser by taking a sample from the one-step denoising distribution, and the output of an unaligned denoiser. Colours = atom mappings.

4.1. Conclusion

In this work, we studied an important aspect of the design space of conditional graph diffusion models: the equivariance of the denoiser. We showed that a permutation equivariant model converges to a 'mean' distribution for all components of the graph, which ultimately impedes the performance of the model. We propose aligned permutation equivariance to force the model to only consider permutations which maintain the intrinsic alignment between the conditioning and generated graphs. This modification allowed us to achieve state-of-the-art results in retrosynthesis among template-free methods, reaching a top-1 accuracy beyond that of template-based methods. Our wider vision is that our work opens up all the future advances in the diffusion paradigm itself to retrosynthesis, now that the fundamentals of how to apply diffusion models in this context are cleared.

References

J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc., 2021.

B. Boys, M. Girolami, J. Pidstrigach, S. Reich, A. Mosca, and O. D. Akyildiz. Tweedie moment projected diffusions for inverse problems. *arXiv preprint arXiv:2310.06721*, 2023.

S. Chen and Y. Jung. Deep retrosynthetic reaction prediction using local reactivity and global attention. *JACS Au*, 1(10):1612–1620, 2021.

H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations*, 2023.

C. W. Coley, R. Barzilay, T. S. Jaakkola, W. H. Green, and K. F. Jensen. Prediction of organic reaction outcomes using machine learning. *ACS Central Science*, 3(5):434–443, 2017a.

C. W. Coley, L. Rogers, W. H. Green, and K. F. Jensen. Computer-assisted retrosynthesis based on molecular similarity. *ACS Central Science*, 3(12):1237–1245, 2017b.

E. Corey and X.-M. Cheng. The logic of chemical synthesis. *Journal of the American Chemical Society*, 118(43):10678–10678, 1996.

E. J. Corey and W. T. Wipke. Computer-assisted design of complex organic syntheses: Pathways for molecular synthesis can be devised with a computer and equipment for graphical communication. *Science*, 166(3902):178–192, 1969.

H. Dai, C. Li, C. Coley, B. Dai, and L. Song. Retrosynthesis prediction with conditional graph logic network. In *Advances in Neural Information Processing Systems*, volume 32, pages 8872–8882. Curran Associates, Inc., 2019.

Z. Dou and Y. Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *International Conference on Learning Representations*, 2024.

V. P. Dwivedi and X. Bresson. A generalization of transformer networks to graphs. In *AAAI Workshop on Deep Learning on Graphs*, 2021.

P. Ertl and A. Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of Cheminformatics*, 1: 1–11, 2009.

M. A. Finzi, A. Boral, A. G. Wilson, F. Sha, and L. Zepeda-Nunez. User-defined event sampling and uncertainty quantification in diffusion models for physical dynamical systems. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10136–10152. PMLR, 2023.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc., 2021.

E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling. Equivariant diffusion for molecule generation in 3D. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 2022.

H. Huang, L. Sun, B. Du, Y. Fu, and W. Lv. Graphgdp: Generative diffusion processes for permutation invariant graph generation. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 201–210. IEEE Computer Society, 2022.

I. Igashov, A. Schneuing, M. Segler, M. M. Bronstein, and B. Correia. RetroBridge: Modeling retrosynthesis with Markov bridges. In *Conference on Learning Representations*, 2024.

E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2016.

Y. Jiang, W. Ying, F. Wu, Z. Huang, K. Kuang, and Z. Wang. Learning chemical rules of retrosynthesis with pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5113–5121, 2023.

W. Jin, C. Coley, R. Barzilay, and T. Jaakkola. Predicting organic reaction outcomes with Weisfeiler-Lehman network. In *Advances in Neural Information Processing Systems*, volume 30, pages 2607–2616. Curran Associates, Inc., 2017.

E. Kim, D. Lee, Y. Kwon, M. S. Park, and Y.-S. Choi. Valid, plausible, and diverse retrosynthesis using tied two-way transformers with latent variables. *Journal of Chemical Information and Modeling*, 61(1):123–133, 2021.

X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto. Diffusion-lm improves controllable text generation. In *Advances in Neural Information Processing Systems*, volume 35, pages 4328–4343. Curran Associates, Inc., 2022.

S. Liu, Z. Tu, M. Xu, Z. Zhang, L. Lin, R. Ying, J. Tang, P. Zhao, and D. Wu. FusionRetro: Molecule representation fusion via in-context learning for retrosynthetic planning. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22028–22041. PMLR, 2023.

D. M. Lowe. *Extraction of Chemical Structures and Reactions from the Literature*. PhD thesis, University of Cambridge, 2012.

C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon. Permutation invariant graph generation via score-based generative modeling. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4474–4484. PMLR, 2020.

X. Peng, Z. Zheng, W. Dai, N. Xiao, C. Li, J. Zou, and H. Xiong. Improving diffusion models for inverse problems using optimal posterior covariance. *arXiv preprint arXiv:2402.02149*, 2024.

M. Sacha, M. Błaz, P. Byrski, P. Dabrowski-Tumanski, M. Chrominski, R. Loska, P. Włodarczyk-Pruszynski, and S. Jastrzebski. Molecule edit graph attention network: modeling chemical reactions as sequences of graph edits. *Journal of Chemical Information and Modeling*, 61(7):3273–3284, 2021.

N. Schneider, N. Stiefl, and G. A. Landrum. What's what: The (nearly) definitive guide to reaction role assignment. *Journal of Chemical Information and Modeling*, 56(12):2336–2346, 2016.

P. Schwaller, T. Gaudin, D. Lanyi, C. Bekas, and T. Laino. "Found in Translation": predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chemical Science*, 9(28):6091–6098, 2018.

P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas, and A. A. Lee. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS Central Science*, 5 (9):1572–1583, 2019.

M. H. Segler and M. P. Waller. Neural-symbolic machine learning for retrosynthesis and reaction prediction. *Chemistry–A European Journal*, 23(25):5966–5971, 2017.

S.-W. Seo, Y. Y. Song, J. Y. Yang, S. Bae, H. Lee, J. Shin, S. J. Hwang, and E. Yang. GTA: Graph truncated attention for retrosynthesis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):531–539, 2021.

C. Shi, M. Xu, H. Guo, M. Zhang, and J. Tang. A graph to graphs framework for retrosynthesis prediction. In *International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8818–8827. PMLR, 2020.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265. PMLR, 2015.

V. R. Somnath, C. Bunne, C. Coley, A. Krause, and R. Barzilay. Learning graph models for retrosynthesis prediction. In *Advances in Neural Information Processing Systems*, volume 34, pages 9405–9415. Curran Associates, Inc., 2021.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

R. Sun, H. Dai, L. Li, S. Kearnes, and B. Dai. Towards understanding retrosynthesis by energy-based models. In *Advances in Neural Information Processing Systems*, volume 34, pages 10186–10194. Curran Associates, Inc., 2021.

I. V. Tetko, P. Karpov, R. Van Deursen, and G. Godin. State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis. *Nature Communications*, 11(1):5575, 2020.

B. M. Trost. The atom economy—a search for synthetic efficiency. *Science*, 254(5037):1471–1477, 1991.

Z. Tu and C. W. Coley. Permutation invariant graph-to-sequence model for template-free retrosynthesis and reaction prediction. *Journal of Chemical Information and Modeling*, 62(15):3503–3513, 2022.

C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard. DiGress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations*, 2023.

Y. Wan, C.-Y. Hsieh, B. Liao, and S. Zhang. Retroformer: Pushing the limits of end-to-end retrosynthesis transformer. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 22475–22490. PMLR, 2022.

Y. Wang, Y. Song, M. Xu, R. Wang, H. Zhou, and W. Ma. RetroDiff: Retrosynthesis as multi-stage distribution interpolation. *arXiv preprint arXiv:2311.14077*, 2023.

L. Wu, B. Trippe, C. Naesseth, D. Blei, and J. P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 31372–31403. Curran Associates, Inc., 2023.

Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.

S. Xie, R. Yan, J. Guo, Y. Xia, L. Wu, and T. Qin. Retrosynthesis prediction with local template retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5330–5338, 2023.

C. Yan, Q. Ding, P. Zhao, S. Zheng, J. YANG, Y. Yu, and J. Huang. RetroXpert: Decompose retrosynthesis prediction like a chemist. In *Advances in Neural Information Processing Systems*, volume 33, pages 11248–11258. Curran Associates, Inc., 2020.

Q. Yan, Z. Liang, Y. Song, R. Liao, and L. Wang. SwinGNN: Rethinking permutation invariance in diffusion models for graph generation. *arXiv preprint arXiv:2307.01646*, 2023.

S. Zheng, J. Rao, Z. Zhang, J. Xu, and Y. Yang. Predicting retrosynthetic reactions using self-corrected transformer neural networks. *Journal of Chemical Information and Modeling*, 60 (1):47–55, 2019.

Z. Zhong, J. Song, Z. Feng, T. Liu, L. Jia, S. Yao, M. Wu, T. Hou, and M. Song. Root-aligned SMILES: a tight representation for chemical reaction prediction. *Chemical Science*, 13(31): 9023–9034, 2022.

Appendices

This appendix is organized as follows. App. A provides additional details on the setup for conditional graph diffusion, including the transition matrices, noise schedule, and data encoding as graphs. App. B presents our theoretical results on aligned permutation equivariance and accompanying proofs. App. C includes additional details to replicate our experimental setup. App. D develops a method to apply arbitrary post-training conditioning with discrete diffusion models, and presents case studies showcasing the usefulness of post-training conditional inference in applications relevant to retrosynthesis. This includes generating samples with desired properties and refining the generation interactively through inpainting.

A. Details on Conditional Graph Diffusion

We start out with a review of the method as outlined in the main paper, and continue with additional details.

We propose to model the problem with a discrete variant of the standard denoising diffusion probabilistic model (DDPM, (Ho et al., 2020)) where the reactants $\mathbf{X}_0$ are diffused in a Markov chain $q(\mathbf{X}_t \mid \mathbf{X}_{t-1})$ for $t \in [1, T]$, and we learn how to transform noise back to reactants $p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y})$, while conditioning on the fixed product $\mathbf{Y}$. Specifically, we adapt discrete diffusion models (Sohl-Dickstein et al., 2015; Hoogeboom et al., 2022; Austin et al., 2021) to graphs, following Vignac et al. (2023).

We assume a Markov *forward* process

$$q(\mathbf{X}_{t+1} \mid \mathbf{X}_t) = \prod_{i=1}^{N_X} q(\mathbf{X}_{t+1}^{\mathcal{N},i} \mid \mathbf{X}_t^{\mathcal{N},i}) \prod_{i,j=1}^{N_X} q(\mathbf{X}_{t+1}^{E,ij} \mid \mathbf{X}_t^{E,ij}), \tag{5}$$

to diffuse the reactant to noise, and a reverse process

$$p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t) = \prod_{i=1}^{N_X} p_\theta(\mathbf{X}_{t-1}^{\mathcal{N},i} \mid \mathbf{X}_t, \mathbf{Y}) \prod_{i,j}^{N_X} p_\theta(\mathbf{X}_{t-1}^{E,ij} \mid \mathbf{X}_t, \mathbf{Y}), \tag{6}$$

defining our generative model. Here, θ represents neural network parameters. Note we implicitly always have time conditioning $p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, t)$, but we drop the explicit t for notational convenience. We also condition on the atom

Algorithm 1 Loss calculation

Input: product $\mathbf{Y}$, reactant $\mathbf{X}_0$, and optional permutation matrix $\mathbf{P}^{\mathbf{X}\to\mathbf{Y}}$ for alignment
$t \sim \text{Uniform}(\{0, \dots, T\})$
$\mathbf{X}_t \sim q(\mathbf{X}_t \mid \mathbf{X}_0)$
$\tilde{\mathbf{X}}_0 = D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$
Return Cross-Entropy$(\mathbf{X}_0, \tilde{\mathbf{X}}_0)$

Algorithm 2 Sampling

Input: product $\mathbf{Y}$
Choose (for alignment): $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}} \in \mathbb{R}^{N_\mathbf{X}\times N_\mathbf{Y}}$
$\mathbf{X}_T \propto p(\mathbf{X}_T)$
for $t = T$ **to** 1 **do**
  $\tilde{\mathbf{X}}_0 = D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$
  $\mathbf{X}^i_{t-1} \sim \sum_k q(\mathbf{X}^i_{t-1} \mid \mathbf{X}^i_t, \mathbf{X}^i_0)\tilde{\mathbf{X}}^i_0$
Return $\mathbf{X}_0$

mapping $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$ in the case of aligned models, but we will not include it in the notation in this section. The full generative distribution is $p_\theta(\mathbf{X}_{0:T} \mid \mathbf{Y}) = p(\mathbf{X}_T)\prod_{t=1}^T p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y})$, where $p(\mathbf{X}_T)$ is a predefined prior such that $p(\mathbf{X}_T) = q(\mathbf{X}_T \mid \mathbf{X}_0)$. Following (Hoogeboom et al., 2021; Austin et al., 2021), we use the neural network specifically to predict ground truth labels from noised samples, so that the neural net directly outputs a distribution $\tilde{p}_\theta\big(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}\big)$. The reverse process is then given by

$$p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}) = \textstyle\sum_{\mathbf{X}_0} q\big(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{X}_0\big)\tilde{p}_\theta\big(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}\big). \tag{7}$$

Throughout the paper, we denote the direct output of the neural network as $D_\theta(\mathbf{X}_t, \mathbf{Y}) = (D_\theta(\mathbf{X}_t, \mathbf{Y})^\mathcal{N}, D_\theta(\mathbf{X}_t, \mathbf{Y})^E)$, where $D_\theta(\mathbf{X}_t, \mathbf{Y})^\mathcal{N} \in \mathbf{R}^{N_X\times K_a}$ and $D_\theta(\mathbf{X}_t, \mathbf{Y})^E \in \mathbf{R}^{N_X\times N_X\times K_b}$, s.t. we have a probability vector for each node and edge.

The single-step transition for nodes (resp. for edges) is defined with a transition matrix $\mathbf{Q}^\mathcal{N}_t$ as

$$q(\mathbf{X}^{\mathcal{N},i}_t \mid \mathbf{X}^{\mathcal{N},i}_{t-1}) = \text{Cat}(\mathbf{X}^{\mathcal{N},i}_t; \mathbf{p} = \mathbf{X}^{\mathcal{N},i}_{t-1}\mathbf{Q}^\mathcal{N}_t). \tag{8}$$

To define $\mathbf{Q}^\mathcal{N}_t$ (resp. $\mathbf{Q}^E_t$), we adopt the absorbing-state formulation from Austin et al. (2021), where nodes and edges gradually transfer to the *absorbing state*, defined as the empty state $\perp$. Formally, $\mathbf{Q}_t = (1-\beta_t)I + \beta_t \mathbb{1}e_\perp^\top$, where β_t defines the *diffusion schedule* and $e_\perp$ is one-hot on the absorbing state. Then, the marginal $q(\mathbf{X}_t \mid \mathbf{X}_0)$ and conditional posterior $q(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{X}_0)$ also have a closed form for the absorbing state transitions. The prior $p(\mathbf{X}_T)$ is correspondingly chosen to be a delta distribution at a graph with no edges and nodes set to the $\perp$ state. The noise schedule β_t is defined using the mutual information criterion proposed in Austin et al. (2021).

We use the cross-entropy loss, as discussed in Austin et al. (2021) and Vignac et al. (2023):

$$-\mathbb{E}_{q(\mathbf{X}_0,\mathbf{Y})q(t)q(\mathbf{X}_t \mid \mathbf{X}_0)}[\log \tilde{p}_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})], \tag{9}$$

where $q(t)$ is a uniform distribution over $t \in \{1 \dots T\}$.

The training and sampling procedures with graph diffusion models are presented in Alg. 1 and Alg. 2, along with optional conditioning by $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$, as described in Sec. 3.2.

Our transition matrices To define $\mathbf{Q}^\mathcal{N}_t$ and $\mathbf{Q}^E_t$, we adopt the absorbing-state formulation from Austin et al. (2021), where nodes and edges gradually transfer to the *absorbing state* $\perp$. Formally, we give the generic form of the transition matrix $\mathbf{Q}_t$ for node input $\mathbf{X}^\mathcal{N} \in \mathbb{R}^{N_X\times N_X}$ [1]

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t \mathbb{1}e_\perp^\top, \tag{10}$$

where β_t defines the *diffusion schedule* and $e_\perp$ is one-hot on the absorbing state $\perp$. For completeness, we list the other two common transitions relevant to our application. The first is the uniform transition as proposed by Hoogeboom et al. (2021)

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t \frac{\mathbb{1}\mathbb{1}^\top}{K} \tag{11}$$

where β_t, I are as before and K is the number of element (edge or node) types, i.e. the number of input features for both nodes and edges. Vignac et al. (2023) also proposed a marginal transition matrix

$$\mathbf{Q}^\mathcal{N}_t = (1-\beta^t)\mathbf{I} + \beta^t \mathbb{1}\big(m^\mathcal{N}\big)^\top \quad \text{and} \quad \mathbf{Q}^E_t = (1-\beta^t)\mathbf{I} + \beta^t \mathbb{1}\big(m^E\big)^\top \tag{12}$$

[1] The only difference between $\mathbf{Q}^\mathcal{N}_t$ and $\mathbf{Q}^E_t$ for the absorbing-state and uniform transitions is the dimensions of I, e, $\mathbb{1}$ and the value of K. We therefore give a generic form for both and imply choosing the right dimensions for each case.

which they argued leads to faster convergence. In this case, $m^{\mathcal{N}} \in \mathbb{R}^{K_a}$ and $m^E \in \mathbb{R}^{K_b}$ are row vectors representing the marginal distributions for node and edge types respectively. We tested all three types of transition matrices in early experiments and noted the absorbing state model to be slightly better than the others. The marginal $q(\mathbf{X}_t \mid \mathbf{X}_0)$ and conditional posterior $q(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{X}_0)$ also have a closed form for all of these transition matrices.

Noise schedule We use the mutual information noise schedule proposed by Austin et al. (2021), which leads to

$$\frac{t}{T} = 1 - \frac{I(\mathbf{X}_t; \mathbf{X}_0)}{H(\mathbf{X}_0)} = \frac{H(\mathbf{X}_0, \mathbf{X}_t) - H(\mathbf{X}_t)}{H(\mathbf{X}_0)} = \frac{\sum_{\mathbf{X}_0, \mathbf{X}_t} p(\mathbf{X}_0) q(\mathbf{X}_t \mid \mathbf{X}_0) \log \frac{q(\mathbf{X}_t \mid \mathbf{X}_0)}{\sum_{\mathbf{X}_0'} p(\mathbf{X}_0') q(\mathbf{X}_t \mid \mathbf{X}_0')}}{\sum_{\mathbf{X}_0} p(\mathbf{X}_0) \log p(\mathbf{X}_0)} \tag{13}$$

For absorbing state diffusion, these equations lead to $\beta_t = \frac{1}{T-t+1}$. Similarly, the total transition probability to the absorbing state at time t has a simple form: $q(\mathbf{X}_t = \perp \mid \mathbf{X}_0) = \frac{t}{T}$.

Forward process posterior For transition matrices that factorize over dimensions, we have

$$q(\mathbf{X}_{t-1,i,:} \mid \mathbf{X}_{t,i,:}, \mathbf{X}_{0,i,:}) \sim \frac{\mathbf{X}_t \mathbf{Q}_t^\top \cdot \mathbf{X}_{0,i,:} \bar{\mathbf{Q}}_{t-1}}{\mathbf{X}_{0,i,:} \bar{\mathbf{Q}}_t \mathbf{X}^\top} \tag{14}$$

where $\mathbf{X}_{i,:}$ is the one-hot encoding of i[th] node/edge of the graph, in row vector format. The formula outputs the correct distribution

Variational lower-bound loss Diffusion models are commonly trained by minimizing the negative variational lower-bound on the model's likelihood (Ho et al., 2020). Austin et al. (2021) discuss the difference between optimizing the ELBO and cross-entropy losses and show that the two losses are equivalent for the absorbing-state transition. We choose to use cross-entropy, similar to Vignac et al. (2023), due to faster convergence during training. We include the formula for the ELBO in Eq. (15) for completeness.

$$\begin{aligned}
\mathcal{L}_{\text{vb}} = \mathbb{E}_{\mathbf{X}_0 \sim q(\mathbf{X}_0)} \Bigg[& \underbrace{\mathrm{KL}\big[q(\mathbf{X}_T \mid \mathbf{X}_0) \,\|\, p(\mathbf{X}_T)\big]}_{\mathcal{L}_T} \\
& + \underbrace{\sum_{t=2}^{T} \mathbb{E}_{\mathbf{X}_t \sim q(\mathbf{X}_t \mid \mathbf{X}_0)} \mathrm{KL}\big[q(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{X}_0) \,\|\, p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t)\big]}_{\mathcal{L}_{t-1}} \\
& - \underbrace{\mathbb{E}_{\mathbf{X}_1 \sim q(\mathbf{X}_1 \mid \mathbf{X}_0)} \log p_\theta(\mathbf{X}_0 \mid \mathbf{X}_1)}_{\mathcal{L}_0} \Bigg]
\end{aligned} \tag{15}$$

We also note that we use this quantity as part of the scoring function mentioned in Sec. 4.

Data encoding and atom-mapping We illustrate our graph encoding using atom-mapping and permutation matrices.

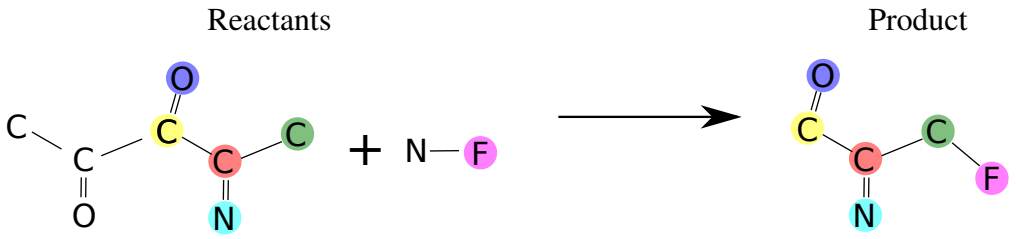
B. Theoretical Results on Aligned Permutation Equivariance

B.1. Proof that Permutation Equivariant Denoisers Do Not Implement the Identity Reaction

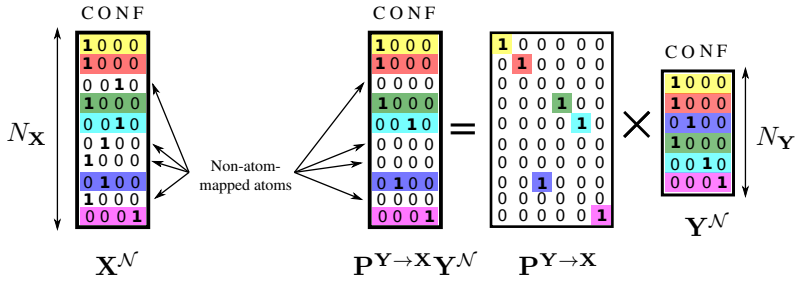
Definitions Let us consider a data set $\mathcal{D} = \{\mathbf{X}_n, \mathbf{Y}_n, \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}}\}_{n=1}^{N_{obs}}$, where for all data points, $\mathbf{X}_n = \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}} \mathbf{Y}_n$, that is, both sides of the reactions are equivalent, up to some permutation, as defined in the atom mapping matrix $\mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}}$. It is always possible to preprocess the data such that the rows of $\mathbf{Y}_n$ are permuted with $\mathbf{Y}_n \leftarrow \mathbf{P}_n^{\mathbf{Y}\to\mathbf{X}} \mathbf{Y}_n$ so that the resulting atom mapping between $\mathbf{Y}_n$ and $\mathbf{X}_n$ is always identity. For simplicity, we assume such a preprocessed data set in this section.

Let us assume that the one-step denoiser probability, $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_T, \mathbf{Y})$, is parameterized by the neural network $D_\theta(\mathbf{X}_T, \mathbf{Y}) \in \mathbb{R}^{N \times K}$ such that the probability factorises for the individual nodes and edges (so there is one output in the network for each node and each edge): $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_T, \mathbf{Y}) = \prod_i \sum_k \mathbf{X}_{0,i,k}^{\mathcal{N}} D_\theta(\mathbf{X}_T, \mathbf{Y})_{i,k}^{\mathcal{N}} \prod_{i,j} \sum_k \mathbf{X}_{0,i,j,k}^{E} D_\theta(\mathbf{X}_T, \mathbf{Y})_{i,j,k}^{E}$.

The ideal denoiser Clearly, the correct one-step denoiser $D_\theta(\mathbf{X}_T, \mathbf{Y}) = \mathbf{Y}$. This can be shown with the Bayes' rule by $q(\mathbf{X}_0 \mid \mathbf{X}_T, \mathbf{Y}) = \frac{q(\mathbf{X}_T \mid \mathbf{X}_0, \mathbf{Y}) q(\mathbf{X}_0 \mid \mathbf{Y})}{q(\mathbf{X}_T \mid \mathbf{Y})} = \frac{q(\mathbf{X}_T) q(\mathbf{X}_0 \mid \mathbf{Y})}{q(\mathbf{X}_T)} = q(\mathbf{X}_0 \mid \mathbf{Y})$. Because one $\mathbf{X}$ always matches with exactly one



Atom mapping from the nodes of $\mathbf{Y}$ to the nodes of $\mathbf{X}$



Atom mapping from the edges of $\mathbf{Y}$ to the edges of $\mathbf{X}$

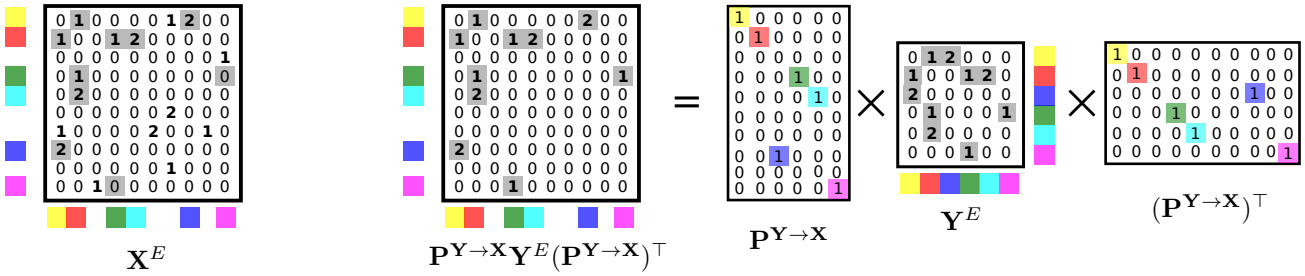


Figure A4: Illustrating atom mapping-based permutations and the tensor-based encoding of the reaction graph. The colours highlight the atom-mapped atoms and the elements that link them in the atom-mapping matrix $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$. $\mathbf{Y}^{\mathcal{N}}$ and $\mathbf{X}^{\mathcal{N}}$ are the one-hot encoded products and reactants, respectively. $\mathbf{Y}^{E}$ and $\mathbf{X}^{E}$ show the edge tensors, where 1=single bond, 2=double bond, 3=triple bond and 0=no bond. The grey boxes highlight links that connect to atom-mapped atoms in the visualization.

$\mathbf{Y}$ in the data, this is a delta distribution $q(\mathbf{X}_0 \,|\, \mathbf{Y}) = \prod_i \delta_{x_{0,i}, y_i}$, where we define $x_{0,i}$ and y_i as the value of the i:th node / edge. It is also easy to see that $D_\theta(\mathbf{X}_T, \mathbf{Y}) = \mathbf{Y}$ is the optimal solution for the cross-entropy loss:

$$
\begin{aligned}
&- \sum_{(\mathbf{X}_0, \mathbf{Y})} q(\mathbf{X}_0) \log p_\theta(\mathbf{X}_0 \,|\, \mathbf{X}_T, \mathbf{Y}) \propto - \sum_{\mathbf{Y}} \log p_\theta(\mathbf{Y} \,|\, \mathbf{X}_T, \mathbf{Y}) \\
&\qquad = - \sum_{\mathbf{Y}} \log \left[\prod_i \sum_k \mathbf{Y}^{N}_{0,i,k} D_\theta(\mathbf{X}_T, \mathbf{Y})^{\mathcal{N}}_{i,k} \prod_{i,j} \sum_k \mathbf{Y}^{E}_{0,i,j,k} D_\theta(\mathbf{X}_T, \mathbf{Y})^{E}_{i,j,k} \right] . \qquad (16)
\end{aligned}
$$

All of the sums $\sum_k \mathbf{Y}^{N}_{0,i,k} D_\theta(\mathbf{X}_T, \mathbf{Y})^{\mathcal{N}}_{i,k}$ and $\sum_k \mathbf{Y}^{E}_{0,i,j,k} D_\theta(\mathbf{X}_T, \mathbf{Y})^{E}_{i,j,k}$ are maximized for each i, j and $\mathbf{Y}$ if $D_\theta(\mathbf{X}_T, \mathbf{Y}) = \mathbf{Y}$. In this case, the loss goes to zero.

The following theorem states that if the neural net is permutation equivariant, it will converge to a 'mean' solution, where the output for each node and each edge is the marginal distribution of nodes and edges in the conditioning product molecule, instead of the global optimum $D_\theta(\mathbf{X}_T, \mathbf{Y}) = \mathbf{Y}$.

Theorem 3. *The optimal permutation equivariant denoiser Let $D_\theta(\mathbf{X}_T, \mathbf{Y})$ be permutation equivariant s.t. $D_\theta(\mathbf{P}\mathbf{X}_T, \mathbf{Y}) = \mathbf{P}D_\theta(\mathbf{X}_T, \mathbf{Y})$, and let $q(\mathbf{X}_T)$ be permutation invariant. The optimal solution with respect to the cross-entropy loss with the identity reaction data is, for all nodes i and j*

$$\begin{cases} D_\theta(\mathbf{X}_T, \mathbf{Y})^{\mathcal{N}}_{i,:} = \hat{\mathbf{y}}^{\mathcal{N}}, \hat{\mathbf{y}}^{\mathcal{N}}_k = \sum_i \mathbf{Y}_{i,k} / \sum_{i,k} \mathbf{Y}_{i,k}, \\ D_\theta(\mathbf{X}_T, \mathbf{Y})^{E}_{i,j,:} = \hat{\mathbf{y}}^{E}, \hat{\mathbf{y}}^{E}_k = \sum_{i,j} \mathbf{Y}_{i,j,k} / \sum_{i,j,k} \mathbf{Y}_{i,j,k}, \end{cases} \tag{17}$$

where $\hat{\mathbf{y}}^{\mathcal{N}}_k$ and $\hat{\mathbf{y}}^{E}_k$ are the marginal distributions of node and edge values in $\mathbf{Y}$.

Proof. **Nodes.** The cross-entropy denoising loss for the nodes can be written as

$$CE = -\sum_{(\mathbf{X}_0, \mathbf{Y})} \mathbb{E}_{q(\mathbf{X}_T \mid \mathbf{X}_0)} \sum_{i,k} \mathbf{X}^{\mathcal{N}}_{0,i,k} \log D_\theta(\mathbf{X}_T, \mathbf{Y})_{i,k} \tag{18}$$

$$= -\sum_{(\mathbf{X}_0, \mathbf{Y})} \mathbb{E}_{q(\mathbf{X}_T)} \sum_{i,k} \mathbf{X}^{\mathcal{N}}_{0,i,k} \log D_\theta(\mathbf{X}_T, \mathbf{Y})_{i,k} \tag{19}$$

$$= -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}_T)} \sum_{i,k} \mathbf{Y}^{\mathcal{N}}_{i,k} \log D_\theta(\mathbf{X}_T, \mathbf{Y})_{i,k}, \tag{20}$$

where the first equality is due to $q(\mathbf{X}_T \mid \mathbf{X}_0)$ containing no information about $\mathbf{X}_0$ at the end of the forward process, and the second equality is due to $\mathbf{X}_0 = \mathbf{Y}$ in the data. Since $q(\mathbf{X}_T)$ is permutation invariant, that is, all permuted versions $\mathbf{P}\mathbf{X}_T$ of $\mathbf{X}_T$ are equally probable, we can split the expectation into two parts $\mathbb{E}_{q(\mathbf{X}_T)}[\cdot] \propto \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{\mathbf{P}}[\cdot]$, where $\mathbf{X}'_T$ contain only graphs in distinct isomorphism classes, and $\sum_{\mathbf{P}}$ sums over all permutation matrices of size $N \times N$,

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{\mathbf{P}} \sum_{i,k} \mathbf{Y}^{\mathcal{N}}_{i,k} \log D_\theta(\mathbf{P}\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{i,k}. \tag{21}$$

Due to the permutation equivariance, $D_\theta(\mathbf{P}\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}} = \mathbf{P}D_\theta(\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}$, and $D_\theta(\mathbf{P}\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{i,k} = D_\theta(\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{\pi(i),k}$, where $\pi(i)$ denotes the index the index i is mapped to in the permutation $\mathbf{P}$. Thus,

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{\pi} \sum_{i,k} \mathbf{Y}^{\mathcal{N}}_{i,k} \log D_\theta(\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{\pi(i),k} \tag{22}$$

$$= -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{i,k} \sum_{\pi} \mathbf{Y}^{\mathcal{N}}_{\pi^{-1}(i),k} \log D_\theta(\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{i,k}, \tag{23}$$

where the equality is due to all permutations being in a symmetric position: What matters is the relative permutation between $\mathbf{Y}^{\mathcal{N}}$ and $D_\theta(\mathbf{X}'_T, \mathbf{Y})$. Now, $\sum_\pi \mathbf{Y}^{\mathcal{N}}_{\pi^{-1}(i),k} = \sum_\pi \mathbf{Y}^{\mathcal{N}}_{\pi(i),k} = C \sum_i \mathbf{Y}^{\mathcal{N}}_{i,k}$, because for each node index i, all the nodes in $\mathbf{Y}$ are included equally often due to symmetry. This is proportional to the marginal distribution $\hat{\mathbf{y}}^{\mathcal{N}}$ up to some constant, and thus we have:

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{i,k} \hat{\mathbf{y}}^{\mathcal{N}}_k \log D_\theta(\mathbf{X}'_T, \mathbf{Y})_{i,k}. \tag{24}$$

Clearly, the optimal value for each node output i is the empirical marginal distribution $D_\theta(\mathbf{X}'_T, \mathbf{Y})^N_{i,:} = (\hat{\mathbf{y}}^{\mathcal{N}})^\top$.

Edges With the exact same steps, we can get the equivalent of Eq. (21) for the edges:

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{P} \sum_{i,j,k} \mathbf{Y}^{E}_{i,j,k} \log D_\theta(P\mathbf{X}'_T, \mathbf{Y})^{E}_{i,j,k}. \tag{25}$$

The permutation equivariance property for the edges is now written as $D_\theta(\mathbf{P}\mathbf{X}'_T, \mathbf{Y})^E = \mathbf{P}D_\theta(\mathbf{X}'_T, \mathbf{Y})^E\mathbf{P}^\top$, and $D_\theta(\mathbf{P}\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{i,jk} = D_\theta(\mathbf{X}'_T, \mathbf{Y})^{\mathcal{N}}_{\pi(i),\pi(j),k}$. Thus,

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{\pi} \sum_{i,j,k} \mathbf{Y}^{E}_{i,j,k} \log D_\theta(\mathbf{X}'_T, \mathbf{Y})^{E}_{\pi(i),\pi(j),k} \tag{26}$$

$$= -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{i,j,k} \sum_{\pi} \mathbf{Y}^{E}_{\pi^{-1}(i),\pi^{-1}(j),k} \log D_\theta(\mathbf{X}'_T, \mathbf{Y})^{E}_{i,j,k}, \tag{27}$$

with the equality holding again due to symmetry. Now, for any pair of node indices i and j, the set of all permutations contains all pairs of node indices $(\pi(i), \pi(j))$ equally often due to symmetry. These pairs correspond to edges in $\mathbf{Y}^E$, and thus $\sum_\pi \mathbf{Y}^E_{\pi^{-1}(i),\pi^{-1}(j),k} = \sum_\pi \mathbf{Y}^E_{\pi(i),\pi(j),k} = D \sum_{i,j} \mathbf{Y}^E_{i,j,k}$, where D is a constant that counts how many times each edge pair appeared in the set of all permutations. This is again proportional to the marginal distribution over the edges $\hat{\mathbf{y}}^E$

$$CE \propto -\sum_{\mathbf{Y}} \mathbb{E}_{q(\mathbf{X}'_T)} \sum_{i,j,k} \hat{\mathbf{y}}^E_k \log D_\theta(\mathbf{X}'_T, \mathbf{Y})_{i,j,k}. \tag{28}$$

Again, the optimal value for each edge output (i, j) is $D_\theta(\mathbf{X}'_T, \mathbf{Y})^E_{i,j,:} = (\hat{\mathbf{y}}^E)^\top$.

□

B.2. Proof of the Generalized Distributional Invariance with Aligned Equivariance

We start by proving a useful lemma, and then continue and continue to the proof of the main theorem.

Lemma 1. *(An aligned denoiser induces aligned distribution equivariance for a single reverse step)*

If the denoiser function D_θ has the aligned equivariance property $D_\theta(\mathbf{RX}, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \mathbf{R}D_\theta(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$, then the conditional reverse distribution $p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$ has the property $p_\theta(\mathbf{RX}_{t-1} \mid \mathbf{RX}_t, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$.

Proof. First, let us denote the transition probabilities from t to $t-1$ with $F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$, where formally $F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^{\mathcal{N}}_{i,k} = \sum_{k'} q(\mathbf{X}^{\mathcal{N}}_{t-1,i,k} \mid \mathbf{X}^{\mathcal{N}}_{t,i,k}, \mathbf{X}^{\mathcal{N}}_{0,i,k}) D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^{\mathcal{N}}_{i,k'}$ and $F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^E_{i,j,k} = \sum_{k'} q(\mathbf{X}^E_{t-1,i,j,k} \mid \mathbf{X}^E_{t,i,j,k}, \mathbf{X}^E_{0,i,j,k}) D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^E_{i,j,k'}$. Clearly, since the values of F_θ depend only pointwise on the values of D_θ, F_θ is aligned permutation equivariant as well.

We continue by directly deriving the connection:

$$\begin{aligned}
&p_\theta(\mathbf{RX}_{t-1} \mid \mathbf{RX}_t, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) && (29)\\
&= \prod_i \sum_k (\mathbf{RX}_{t-1})^{\mathcal{N}}_{i,k} F_\theta(\mathbf{RX}_t, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top)_{i,k} \\
&\quad \times \prod_{i,j} \sum_k (\mathbf{RX}_{t-1})^E_{i,j,k} F_\theta(\mathbf{RX}_t, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top)^E_{i,j,k} && (30)\\
&= \prod_i \sum_k (\mathbf{X}_{t-1})^{\mathcal{N}}_{\pi(i),k} F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})_{\pi(i),k} \\
&\quad \times \prod_{i,j} \sum_k (\mathbf{X}_{t-1})^E_{\pi(i),\pi(j),k} F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^E_{\pi(i),\pi(j),k}, && (31)
\end{aligned}$$

where in the last line we used the aligned permutation equivariance definition, and the the effect of the permutation matrix $\mathbf{R}$ on index i was denoted as $\pi(i)$. Now, regardless of the permutation, the products contain all possible values i and pairs i, j exactly once. Thus, the expression remains equal if we replace $\pi(i)$ with just i:

$$\begin{aligned}
&p_\theta(\mathbf{RX}_{t-1} \mid \mathbf{RX}_t, \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) && (32)\\
&= \prod_i \sum_k (\mathbf{X}_{t-1})^{\mathcal{N}}_{i,k} F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})_{i,k} \prod_{i,j} \sum_k (\mathbf{X}_{t-1})^E_{i,j,k} F_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^E_{i,j,k} && (33)\\
&= p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}), && (34)
\end{aligned}$$

which concludes the proof. □

Theorem 4. *Aligned denoisers induce aligned permutation invariant distributions If the denoiser function D_θ has the aligned equivariance property and the prior $p(\mathbf{X}_T)$ is permutation invariant, then the generative distribution $p_\theta(\mathbf{X}_0 \mid \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$ has the corresponding property for any permutation matrices $\mathbf{R}$ and $\mathbf{Q}$:*

$$p_\theta(\mathbf{RX}_0 \mid \mathbf{QY}, \mathbf{RP}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = p_\theta(\mathbf{X}_0 \mid \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) \tag{35}$$

Proof. Let us assume that the result holds for some noisy data level t: $p_\theta(\mathbf{R}\mathbf{X}_t \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = p_\theta(\mathbf{X}_t \,|\, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$. We will then show that the same will hold for $\mathbf{X}_{t-1}$, which we can use to inductively show that the property holds for $\mathbf{X}_0$. We begin as follows:

$$p_\theta(\mathbf{R}\mathbf{X}_{t-1} \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \sum_{\mathbf{X}_t} p_\theta(\mathbf{R}\mathbf{X}_{t-1} \,|\, \mathbf{X}_t, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) p_\theta(\mathbf{X}_t \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) \tag{36}$$

$$= \sum_{\mathbf{X}_t} p_\theta(\mathbf{X}_{t-1} \,|\, \mathbf{R}^{-1}\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) p_\theta(\mathbf{R}^{-1}\mathbf{X}_t \,|\, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}). \tag{37}$$

where on the second line we used Lem. 1 and the assumption that the result holds for noise level t. The sum over $\mathbf{X}_t$ contains all possible graphs and all of their permutations. Thus, the exact value of $\mathbf{R}^{-1}$ does not affect the value of the final sum, as we simply go through the same permutations in a different order, and aggregate the permutations with the sum. Thus,

$$p_\theta(\mathbf{R}\mathbf{X}_{t-1} \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \sum_{\mathbf{X}_t} p_\theta(\mathbf{X}_{t-1} \,|\, \mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) p_\theta(\mathbf{X}_t \,|\, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) \tag{38}$$

$$= p_\theta(\mathbf{X}_{t-1} \,|\, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) \tag{39}$$

showing that if the result holds for level t, then it also holds for level $t-1$. We only need to show that it holds for level $\mathbf{X}_{T-1}$ to start the inductive chain:

$$p_\theta(\mathbf{R}\mathbf{X}_{T-1} \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \sum_{\mathbf{X}_T} p_\theta(\mathbf{R}\mathbf{X}_{T-1} \,|\, \mathbf{X}_T, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) p(\mathbf{X}_T) \tag{40}$$

$$= \sum_{\mathbf{X}_T} p_\theta(\mathbf{X}_{T-1} \,|\, \mathbf{R}^{-1}\mathbf{X}_T, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) p(\mathbf{R}^{-1}\mathbf{X}_T), \tag{41}$$

where on the second line we again used Lem. 1 and the permutation invariance of $p(\mathbf{X}_T)$. Again, the exact value of $\mathbf{R}^{-1}$ does not matter for the sum, since the sum goes through all possible permutations in any case. Thus we have

$$p_\theta(\mathbf{R}\mathbf{X}_{T-1} \,|\, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \sum_{\mathbf{X}_T} p_\theta(\mathbf{X}_{T-1} \,|\, \mathbf{X}_T, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) p(\mathbf{X}_T) \tag{42}$$

$$= p_\theta(\mathbf{X}_{T-1} \,|\, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}). \tag{43}$$

Thus, since the property holds for $\mathbf{X}_{T-1}$, it also holds for $\mathbf{X}_{T-2}, \ldots$, until $\mathbf{X}_0$. This concludes the proof. □

B.3. Proofs that Our Denoisers Are Aligned Permutation Equivariant

In this section, we show for each of the three alignment methods that the corresponding denoisers do indeed fall within the aligned permutation equivariance function class. Fig. A5 summarizes the different alignment methods.

Atom-mapped positional encodings We start by writing out one side of the aligned permutation equivariance condition, $D_\theta(\mathbf{R}\mathbf{X}_t, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top)$ for this particular function class, and directly show that it equals $\mathbf{P}D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$.

$$D_\theta(\mathbf{R}\mathbf{X}_t, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = f_\theta^{\mathbf{X}}(\begin{bmatrix}\mathbf{R}\mathbf{X}_t^N & \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top\mathbf{Q}\varphi\end{bmatrix}, \mathbf{R}\mathbf{X}^E\mathbf{R}^\top, \begin{bmatrix}\mathbf{Q}\mathbf{Y}^N & \mathbf{Q}\varphi\end{bmatrix}, \mathbf{Q}\mathbf{Y}^E\mathbf{Q}^\top) \tag{44}$$

$$= f_\theta^{\mathbf{X}}(\begin{bmatrix}\mathbf{R}\mathbf{X}_t^N & \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\varphi\end{bmatrix}, \mathbf{R}\mathbf{X}^E\mathbf{R}^\top, \begin{bmatrix}\mathbf{Q}\mathbf{Y}^N & \mathbf{Q}\varphi\end{bmatrix}, \mathbf{Q}\mathbf{Y}^E\mathbf{Q}^\top), \tag{45}$$

where f_θ itself is a function that is permutation equivariant for the combined $\mathbf{X}$ and $\mathbf{Y}$ graph as input. This means that the neural net itself gives an output for the entire combined graph, but we only consider the $\mathbf{X}$ subgraph as the denoiser output, denoted here as $f_\theta^{\mathbf{X}}$. For clarity, we can combine the reactant and product node features and adjacency matrices in the notation, and use the permutation equivariance property of our GNN:

$$f_\theta(\begin{bmatrix}\mathbf{R}\mathbf{X}_t^{\mathcal{N}} & \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\varphi \\ \mathbf{Q}\mathbf{Y}^{\mathcal{N}} & \mathbf{Q}\varphi\end{bmatrix}, \begin{bmatrix}\mathbf{R}\mathbf{X}_t^E\mathbf{R}^\top & 0 \\ 0 & \mathbf{Q}\mathbf{Y}^E\mathbf{Q}^\top\end{bmatrix}) \tag{46}$$

$$= \begin{bmatrix}\mathbf{R} & 0 \\ 0 & \mathbf{Q}\end{bmatrix} f_\theta(\begin{bmatrix}\mathbf{X}_t^{\mathcal{N}} & \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\varphi \\ \mathbf{Y}^{\mathcal{N}} & \varphi\end{bmatrix}, \begin{bmatrix}\mathbf{X}_t^E & 0 \\ 0 & \mathbf{Y}^E\end{bmatrix}) \quad \text{(permutation equivariance of base NN)} \tag{47}$$

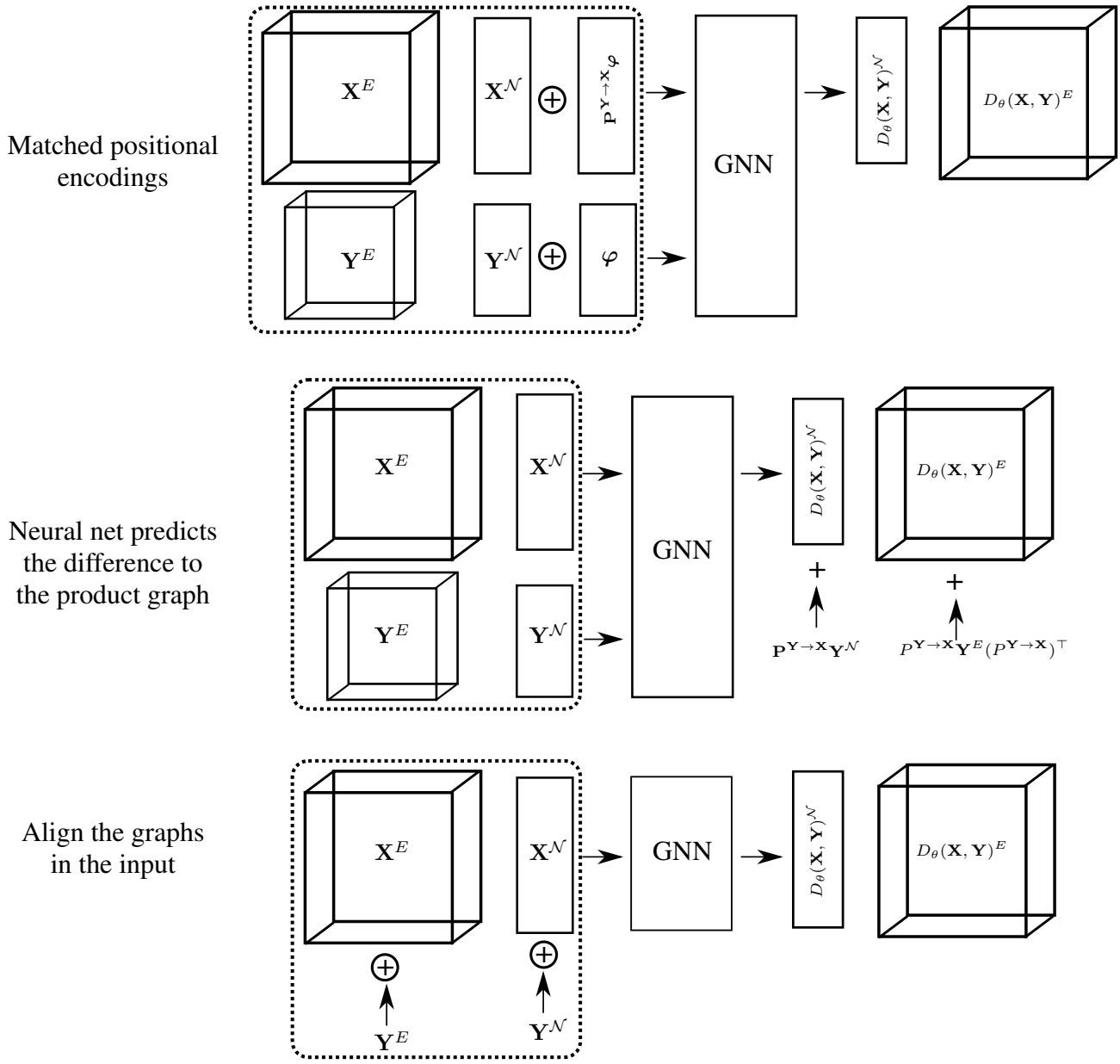


Figure A5: Different ways to align the graphs in the architecture. All of them can be combined. The $\oplus$ sign means concatenation along the feature dimension, and $+$ is the standard addition. All of the methods can be combined together.

Taking only the $\mathbf{X}$ part of the output and reverting to D_θ notation, we directly arrive at the result that $D_\theta(\mathbf{R}\mathbf{X}_t, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) = \mathbf{R}D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})$.

Directly adding $\mathbf{Y}$ to the output We again start by writing out an aligned equivariant input to the denoiser, with f_θ again denoting a network that is permutation equivariant with respect to the combined $\mathbf{X}$ and $\mathbf{Y}$ graph:

$$
\begin{aligned}
D_\theta(\mathbf{R}\mathbf{X}, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{X}\to\mathbf{Y}}\mathbf{Q}^\top) &= \text{softmax}(f_\theta^{\mathbf{X}}(\mathbf{R}\mathbf{X}, \mathbf{Q}\mathbf{Y}) + \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top\mathbf{Q}\mathbf{Y}) && (48)\\
&= \text{softmax}(\mathbf{R}f_\theta^{\mathbf{X}}(\mathbf{X}, \mathbf{Y}) + \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}) \quad \text{(permutation equivariance of base denoiser)} && (49)\\
&= \mathbf{R}\,\text{softmax}(f_\theta^{\mathbf{X}}(\mathbf{X}, \mathbf{Y}) + \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}) && (50)\\
&= \mathbf{R}D_\theta(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{X}\to\mathbf{Y}}) && (51)
\end{aligned}
$$

where we were able to move the permutation outside the softmax since the softmax is applied on each node and edge separately.

Aligning $\mathbf{Y}$ and $\mathbf{X}$ at the input to the model Let's denote by $[\mathbf{X}\ \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}]$ concatenation along the feature dimension for both the nodes and edges of the graphs. Recall then that the definition of aligning the graphs in the input is $D_\theta(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) = f_\theta([\mathbf{X}\ \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}])$, where f_θ is a permutation equivariant denoiser. Writing out the aligned equivariance condition

$$
\begin{aligned}
D_\theta(\mathbf{R}\mathbf{X}, \mathbf{Q}\mathbf{Y}, \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top) &= f_\theta([\mathbf{R}\mathbf{X}\ \ \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Q}^\top\mathbf{Q}\mathbf{Y}]) = f_\theta([\mathbf{R}\mathbf{X}\ \ \mathbf{R}\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}]) && (52)\\
&= f_\theta(\mathbf{R}[\mathbf{X}\ \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}]) = \mathbf{R}f_\theta([\mathbf{X}\ \ \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}\mathbf{Y}]) && (53)\\
&= D_\theta(\mathbf{X}, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}}) && (54)
\end{aligned}
$$

which shows that this method results in aligned equivariance as well.

B.4. A Single-layer Graph Transformer with Orthogonal Atom-mapped Positional Encodings is Able to Implement the Identity Reaction Solution for Nodes

Here, we show that a single-layer Graph Transformer neural net can model the identity reaction for the nodes given orthogonal atom-mapped positional encodings (e.g., the graph Laplacian eigenvector-based ones). In particular it is possible to find a θ such that $D_\theta(\mathbf{X}_t, \mathbf{Y}, \mathbf{P}^{\mathbf{Y}\to\mathbf{X}})^{\mathcal{N}} = \mathbf{Y}^{\mathcal{N}}$. We hope that this section can serve as an intuitive motivation for why matched positional encodings help in copying over the structure from the product to the reactant side.

Recall that we have N atoms on both sides of the reaction. Let us map the atom mapping indices to basis vectors in an orthogonal basis $\varphi = [\varphi_1, \varphi_2, \ldots, \varphi_N]^\top$. In practice, the node input to the neural net on the reactant side is now $\mathbf{X}_t^* = [\mathbf{X}_t^{\mathcal{N}}, \varphi]$, and the node input on the product side is $\mathbf{Y}^* = [\mathbf{Y}^{\mathcal{N}}, \varphi]$.

Now, a single-layer Graph Transformer looks as follows (as defined in the Vignac et al. (2023) codebase):

$$\mathbf{N} = [(\mathbf{X}_t^{*\mathcal{N}})^\top, (\mathbf{Y}^{*\mathcal{N}})^\top]^\top \quad \text{(Concatenate rows)} \tag{55}$$

$$\mathbf{E} = \begin{bmatrix} \mathbf{X}_t^E & 0 \\ 0 & \mathbf{Y}^E \end{bmatrix} \quad \text{(Create joined graph)} \tag{56}$$

$$\mathbf{N}_1 = MLP_N(\mathbf{N}) \quad \text{(Applied with respect to last dimension)} \tag{57}$$

$$\mathbf{E}_1 = \frac{1}{2}(MLP_E(\mathbf{E}) + MLP_E(\mathbf{E})^\top) \quad \text{(Symmetrize the input)} \tag{58}$$

$$t_1 = MLP_t(t) \tag{59}$$

$$\mathbf{Q}, \mathbf{K}, \mathbf{V} = \mathbf{W}_Q\mathbf{N}_1, \mathbf{W}_K\mathbf{N}_1, \mathbf{W}_V\mathbf{N}_1 \quad \text{(One attention head for simplicity)} \tag{60}$$

$$\mathbf{A}_1 = \mathbf{Q}\mathbf{K}^\top \tag{61}$$

$$\mathbf{A}_2 = \mathbf{A}_1 * (\mathbf{W}_{\mathbf{E}_2}\mathbf{E}_1 + 1) + \mathbf{W}_{\mathbf{E}_3}\mathbf{E}_1 \tag{62}$$

$$\mathbf{A}_3 = \text{softmax}(\mathbf{A}_3) \tag{63}$$

$$\mathbf{N}_2 = \mathbf{A}\mathbf{V}/\sqrt{d_f} \quad (d_f \text{ is the embedding dim of V}) \tag{64}$$

$$\mathbf{N}_3 = \mathbf{N}_2 * (\mathbf{W}_{t_2}t_1 + 1) + \mathbf{W}_{t_3}t_1 \tag{65}$$

$$\mathbf{E}_2 = \mathbf{W}(\mathbf{A}_2 * (\mathbf{W}_{t_4}t_1 + 1) + W_{t_5}t_1) \tag{66}$$

$$\begin{aligned} t_4 = {} & MLP_{t_4}(\mathbf{W}t_2 + \mathbf{W}[\min(\mathbf{N}_3), \max(\mathbf{N}_3), \text{mean}(\mathbf{N}_3), \text{std}(\mathbf{N}_3)] \\ & + \mathbf{W}[\min(\mathbf{N}_3), \max(\mathbf{N}_3), \text{mean}(\mathbf{N}_3), \text{std}(\mathbf{N}_3)] \end{aligned} \tag{67}$$

$$\mathbf{E}_3 = MLP_{\mathbf{E}_2}(\mathbf{E}_2) + \mathbf{E} \tag{68}$$

$$\mathbf{E}_{out} = \frac{1}{2}(\mathbf{E}_3 + \mathbf{E}_3^\top) \quad \text{(Symmetrize the output)} \tag{69}$$

$$\mathbf{N}_{out} = MLP_{N_3}(\mathbf{N}_3) + \mathbf{N} \tag{70}$$

$$t_{out} = t_4 + t \tag{71}$$

Now, for purposes of illustration, we can define most linear layers to be zero layers: $MLP_E, MLP_t, \mathbf{W}_{E_2}, \mathbf{W}_{E_3}, \mathbf{W}_{t_2}, \mathbf{W}_{t_3}, \mathbf{W}_{t_4}, \mathbf{W}_{t_5} = 0$. In addition to this, we define MLP_N to be an identity transform. $\mathbf{W}_Q$ and $\mathbf{W}_K$ are both chosen as picking out the $\mathbf{U}$ columns of $\mathbf{N}_1$, with additional overall scaling by some constant α. $\mathbf{W}_V$ is chosen to pick out the product node labels: $\mathbf{W}_V\mathbf{N}_1 = \begin{bmatrix} 0 \\ \mathbf{Y}^{\mathcal{N}} \end{bmatrix}$. Now, we can easily see how the Graph Transformer can obtain the optimal denoising solution for the nodes. Consider an input $\mathbf{N} = [(\mathbf{P}\mathbf{X}_t^*)^\top, (\mathbf{Y})^\top]^\top$, where the reactant side is permuted. The output of the network should be $\mathbf{P}\mathbf{Y}$. Focusing on the parts of the network that compute the node features:

$$\mathbf{A}_1 = \alpha^2 \begin{bmatrix} \mathbf{P}\varphi\varphi^\top\mathbf{P}^\top & \mathbf{P}\varphi\varphi^\top \\ \varphi\varphi^\top\mathbf{P}^\top & \varphi\varphi^\top \end{bmatrix} = \alpha^2 \begin{bmatrix} \mathbf{I} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{I} \end{bmatrix}, \tag{72}$$

$$\mathbf{A}_2 = \mathbf{A}_1, \tag{73}$$

$$\mathbf{A}_3 \approx \text{softmax}(\mathbf{A}_2) = \frac{1}{2}\begin{bmatrix} \mathbf{I} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{I} \end{bmatrix} \quad (\text{If } \alpha \gg 1), \tag{74}$$

$$\mathbf{N}_2 = \frac{\mathbf{A}_3\mathbf{V}}{\sqrt{d_F}} = \begin{bmatrix} \mathbf{I} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{I} \end{bmatrix} \begin{bmatrix} 0 \\ Y_N \end{bmatrix} /(2\sqrt{d_f}) = \begin{bmatrix} \mathbf{P}\mathbf{Y}_N \\ \mathbf{Y}_N \end{bmatrix} /(2\sqrt{d_f}). \tag{75}$$

Here, we used the fact that we chose the positional embeddings to be an orthogonal basis, and $\varphi\varphi^\top = \mathbf{I}$, as well as the fact that $\mathbf{P}\mathbf{P}^\top = \mathbf{I}$ for any permutation matrices. The term $\frac{1}{2}$ in the third equation came from the fact that each row of $\mathbf{A}_1$ contains two non-zero values that are also equal. The probability gets divided between the two of them in the softmax if the logits are scaled large enough, and the approximation becomes arbitrarily accurate.

From now on, since we are interested in the denoising output only for the reactant side, we drop out the reactant side $\mathbf{Y}^{\mathcal{N}}$ and only focus on the $\mathbf{P}\mathbf{Y}^{\mathcal{N}}$ part. We choose the final $MLP_{\mathbf{N}_2}$ to scale the output by some factor $\beta \gg 1$:

$$\mathbf{N}_{out} = \beta\mathbf{N}_2 + \mathbf{N} \tag{76}$$

$$n_\theta(\mathbf{P}\mathbf{X}_t^*, \mathbf{Y}^*)^{\mathcal{N}} = \text{softmax}(\beta\mathbf{P}\mathbf{Y}^{\mathcal{N}} + \mathbf{N}) \approx \mathbf{P}\mathbf{Y}^{\mathcal{N}}. \tag{77}$$

Here, the approximation can be made arbitrarily accurate by scaling β to a higher value, since the logits will become more and more peaked towards the values where $P\mathbf{Y}^{\mathcal{N}}$ equals one instead of zero. This showcases how the attention mechanism in the Graph Transformer pairs with atom mapping-based orthogonal positional encodings to achieve the identity function from products to reactants.

C. Experimental Setup

C.1. Data: USPTO Data Sets

All open-source data sets available for reaction modelling are derived in some form from the patent mining work of Lowe (2012). We distinguish 5 subsets used in previous work: 15k, 50k, MIT, Stereo, and full (original data set). Table A2 provides key information about the subsets.

Table A2: UPSTO-50K subsets used in retrosynthesis

Subset	Introduced by	# of reactions	Preprocessiong & Data split (script)
Full	Lowe (2012)	1 808 938	Dai et al. (2019)
Stereo	Schwaller et al. (2018)	1 002 970	Schwaller et al. (2019)
MIT	Jin et al. (2017)	479 035	-
50k	Schneider et al. (2016)	50 016	Dai et al. (2019)

15k is proposed by Coley et al. (2017a). The subset includes reactions covered by the 1.7k most common templates. All molecules appearing in the reaction are included to model the involvement of reagents and solvents despite not contributing with atoms to the product.

50k is preprocessed by Schneider et al. (2016). The goal of the analysis is to assign roles (reactant, reagent, solvent) to different participants in a reaction through atom mapping. This effort led to the creation of an atom-mapped and classified subset of around 50k reactions, which is used nowadays as a benchmark for retrosynthesis tasks. It is not clear how said subset was selected.

MIT is used by Jin et al. (2017). The preprocessing is described as 'removing duplicate and erroneous reactions' with no further explanation of what qualifies as an erroneous reaction. The output of this filtering is a data set of 49k reactions (from an original set of 1.8M reactions).

Stereo is proposed by Schwaller et al. (2018). The authors apply a more flexible filtering strategy compared to USPTO-MIT. Their data set only discards 800k reactions from the original data set because they are duplicates or they could not be canonicalized by RDKit. In addition, the data set only considers single-product reactions (92% of the full data set), as opposed to splitting multi-product reactions. The preprocessing steps include removing reagents (molecules with no atoms appearing in the product), removing hydrogen atoms from molecules, discarding atom-mapping information and canonicalizing molecules. In addition, since the original method applied to this subset is a language model, tokenization is performed on the atoms.

Full is preprocessed by Dai et al. (2019). The processing includes removing duplicate reactions, splitting reactions with multiple products into multiple reactions with one product, removing reactant molecules appearing unchanged on the product side, removing all reactions with bad atom-mapping (i.e., when the sorted mapping between products and reactants is not one-to-one), and removing bad products (missing mapping, or not parsed by Rdkit).

Our choice Similar to many other works on retrosynthesis, we use 50k as the main data set to evaluate our method.

C.2. Notes on Our Sampling and Ranking Procedures

Duplicate removal Removing duplicates from the set of generated precursors is a common methodology in retrosynthesis, albeit often not discussed explicitly in papers. The benefit of duplicate removal is to ensure that an incorrect molecule that is nevertheless judged as the best one according to the ranking scheme does not fill up all of the top-k positions after ranking. While this does not affect top-1 scores, not removing the duplicates would degrade the other top-k scores significantly.

Choice of scoring function Specifically, we use the following formula for approximating the likelihood of the sample

under the model.

$$s(\mathbf{X}) = (1-\lambda)\frac{\text{count}(\mathbf{X})}{\sum_{\mathbf{X}'}\text{count}(\mathbf{X}')} + \lambda \cdot \frac{e^{\text{elbo}(\mathbf{X})}}{\sum_{\mathbf{X}'} e^{\text{elbo}(\mathbf{X}')}}, \tag{78}$$

where count(.) returns the number of occurrences of sample X in the set of generated samples (by default, 100), elbo(.) computes the variational lower bound of the specific sample under the model, and λ is a weighting hyperparameter. The sums are taken over the set of generated samples. Intuitively, the idea is to provide an estimate of the likelihood from two routes. The ELBO is an estimate (a lower bound) of $\log p_\theta(\mathbf{X})$, and exponentiating and normalizing gives an estimate of the probability distribution. Since the same reactants are often repeated in a set of 100 samples, the counts can be used as a more direct proxy, although they inherently require a relatively large amount of samples to limit the variance of this estimate.

We set the value of λ to be 0.9, although we find that the top-k scores are not at all sensitive to variation in the exact value, as long as it is below 1, and the count information is used. Thus, the counts seem more important than the ELBO, which may be due to the lower bound nature of the ELBO or stochasticity in estimating its value. More accurate likelihood estimation schemes for diffusion models, such as exact likelihood values using the probability flow ODE (Song et al., 2021), could be a valuable direction for future research in the context of retrosynthetic diffusion models.

C.3. Details on stereochemistry

Our model does not explicitly consider changes in stereochemistry in the reaction, but instead, we use the atom mappings implicitly assigned to the samples by the model to transfer the chiral tags from the products to the reactants. The initial choice of $\mathbf{P}^{\mathbf{Y}\to\mathbf{X}}$ at the start of sampling can be considered to be the atom mapping of the generated reactants, given that the model has been trained on correct atom mappings.

For the chiral tags, we take the ground-truth SMILES for the product molecules from the dataset and assign the corresponding chiral tag to the corresponding atom mapping on the generated reactants. For cis/trans isomerism, we use the `Chem.rdchem.BondDir` bond field in rdkit molecules and transfer them to the reactant side based on the atom mapping of the pair of atoms at the start and end of the bond.

Note that when using rdkit, transferring chirality requires some special care: The chiral tags `Chem.ChiralType.CHI_TETRAHEDRAL_CCW` and `Chem.ChiralType.CHI_TETRAHEDRAL_CW` are defined in the context of the order in which the bonds are attached to the chiral atom in the molecule data structure. Thus, the chiral tag sometimes has to be flipped to retain the correct stereochemistry, based on whether the order of the bonds is different on the reactant molecule data structure and the product molecule data structure.

C.4. Details of the Evaluation Procedure

Top-k scores We evaluate the top-k scores by ranking the list of generated and deduplicated samples and calculating the percentage of products for which the ground-truth reactants are in the first k elements in the list.

Mean reciprocal rank We formally define the MRR as $\text{MRR} = \mathbb{E}_{p(r)}[r^{-1}]$. Verbally, it is the expected value of the inverse of the amount of reactant suggestions that the model makes before encountering the ground truth, and as such measures how early on is the correct reactant encountered in the ranked samples. It also incorporates the intuition that the difference between obtaining the correct reactants in, say, the 9th or the 10th position, is not as significant as the difference between the 1st and 2nd positions.

While we do not have direct access to the entire $p(r)$ just based on the top-k scores, we can estimate it with a uniform distribution assumption on r within the different top-k ranges. Formally, we define four sets $S_1, S_2, S_3, S_4 = \{1\}, \{2,3\}, \{4,5\}, \{6,7,8,9,10\}$ in which $p(r)$ is assumed to be uniform, and $s(r) \in \{1,2,3,4\}$ denotes the group that rank r belongs to. Top-k is denoted as top_k, where $k \in \{1,3,5,10\}$ in our case. Note that they are also equal to the cumulative distribution of $p(r)$ until k. We thus define $\hat{p}(r) = \frac{\text{top}_{\max(G_{s(r)})} - \text{top}_{\max(G_{s(r)-1})}}{|G_{s(r)}|}$ and $\hat{p}(1) = \text{top}_1$. For the case where the ground truth was not in the top-10, we assume it is not recovered and place the rest of the probability mass on $p(\infty)$. Our MRR estimate is then defined as

$$\widehat{\text{MRR}} = \sum_{r=1}^{10} \hat{p}(r)\frac{1}{r}. \tag{79}$$

In cases where we do not have top-k values for all {1,3,5,10} (such as the Augmented Transformer in Table 1 for top-3), we

assume that $\hat{p}(r)$ is constant in the wider interval between the preceding and following top-ks (2–5 in the case of the top-3 missing).

C.5. Neural Network Architecture, Hyperparameters, and Compute Resources

We discuss here the neural network architecture and hyper-parameters we choose. Our denoiser is implemented as a Graph Transformer (Dwivedi and Bresson, 2021), based on the implementation of Vignac et al. (2023) with additional graph-level level features added to the input of the model. See Vignac et al. (2023) for an in-depth discussion of the neural network.

In all of our models, we use 9 Graph Transformer layers. When using Laplacian positional encodings, we get the 20 eigenvectors of the Graph Laplacian matrix with the largest eigenvalues and assign to each node a 20-dimensional feature vector.

We use a maximum of 15 'blank' nodes, in practice meaning that the models have the capacity to add 15 additional atoms on the reactant side. In another detail, following Vignac et al. (2023), we weigh the edge components in the cross-entropy loss by a factor of 5 compared to the node components.

We used a batch size of 16 for the models where the expanded graph containing X and Y as subgraphs is given as input. These models were trained for approximately 600 epochs with a single A100/V100/AMD MI250x GPU. For the model where alignment is done by concatenating $\mathbf{Y}$ along the feature dimension in the input, the attention map sizes were smaller and we could fit a larger batch of 32 with a single V100 GPU. This model was trained for 600 epochs. The training time for all of our models was approximately three days. In early experiments and developing the model, we trained or partially trained multiple models that did not make it to the main paper. Sampling 100 samples for one product with $T = 100$ from the model takes roughly 60 seconds with the current version of our code with an AMD MI250x GPU, and 100 samples with $T = 10$ takes correspondingly about 6 seconds. It is likely that the inference could be optimized, increasing the sample throughput.

The reported models were chosen based on evaluating different checkpoints with 10 diffusion steps on the validation set for different checkpoints and chose the best checkpoint based on the MRR score.

C.6. Comparison to Other Baselines

Table A3: Top-k accuracy and MRR on the USPTO-50k test data set—extended comparison, including models with pretraining on larger data sets, and Retrobridge (Igashov et al., 2024), where the evaluation is done with a relaxed metric that does not consider charges or stereochemistry.

	Method	$\mathbf{k=1}\uparrow$	$\mathbf{k=3}\uparrow$	$\mathbf{k=5}\uparrow$	$\mathbf{k=10}\uparrow$	$\widehat{\mathbf{MRR}}\uparrow$
Pre-trained	RSMILES (Zhong et al., 2022)	56.3	**79.2**	**86.2**	**91.0**	0.680
	PMSR (Jiang et al., 2023)	**62.0**	78.4	82.9	86.8	**0.704**
Temp.	Retrosym (Coley et al., 2017b)	37.3	54.7	63.3	74.1	0.480
	GLN (Dai et al., 2019)	52.5	74.7	81.2	87.9	0.641
	LocalRetro (Chen and Jung, 2021)	**52.6**	**76.0**	**84.4**	**90.6**	**0.650**
Synthon	GraphRetro (Somnath et al., 2021)	**53.7**	68.3	72.2	75.5	0.611
	RetroDiff (Wang et al., 2023)	52.6	**71.2**	**81.0**	83.3	**0.629**
	MEGAN (Sacha et al., 2021)	48.0	70.9	78.1	**85.4**	0.601
	G2G (Shi et al., 2020)	48.9	67.6	72.5	75.5	0.582
Template-free	SCROP (Zheng et al., 2019)	43.7	60.0	65.2	68.7	0.521
	Tied Transformer (Kim et al., 2021)	47.1	67.1	73.1	76.3	0.572
	Aug. Transformer (Tetko et al., 2020)	48.3	-	73.4	77.4	0.569
	Retrobridge (*) (Igashov et al., 2024)	50.3	74.0	80.3	85.1	0.622
	GTA_aug (Seo et al., 2021)	51.1	67.6	74.8	81.6	0.605
	Graph2SMILES (Tu and Coley, 2022)	52.9	66.5	70.0	72.9	0.597
	Retroformer (Wan et al., 2022)	52.9	68.2	72.5	76.4	0.608
	DualTF_aug (Sun et al., 2021)	53.6	70.7	74.6	77.0	0.619
Ours	Unaligned	4.1	6.5	7.8	9.8	0.056
	DiffAlign-input	44.1	65.9	72.2	78.7	0.554
	DiffAlign-PE	49.0	70.7	76.6	81.8	0.601
	DiffAlign-PE+skip	54.7	73.3	77.8	81.1	**0.639**

We present other results in the literature not directly comparable to our model.

Methods with a different evaluation procedure

Despite Igashov et al. (2024)'s RetroBridge being closely related to a diffusion model, we cannot include it in a straightforward comparison because it discards atom charges from the ground truth smiles during evaluation. Specifically, the model uses only atom types as node features and compares the generated samples to the smiles reconstructed from the ground truth data through the same encoding (i.e. without charges too) [2].

Methods with pretraining Zhong et al. (2022) and Jiang et al. (2023) pre-train their models on the USPTO-Full and Pistachio data sets, respectively, and as such the results are not directly comparable to models trained on the standard USPTO-50k benchmark. Pretraining with diffusion models is an interesting direction for future research, but we consider it outside the scope of our work. Furthermore, comparison between models with different pretraining datasets and pretraining strategies has the danger of complicating comparisons, given that relative increases in performance could be explained by the model, the pretraining strategy, or the pretraining dataset. As such, we believe that standardized benchmarks like USPTO-50k are necessary when researching modelling strategies.

D. Adding Post-Training Conditioning to Discrete Diffusion Models

In this section, we show a method to add additional controls and conditions on the used discrete diffusion model post-training. While the notation is from the point of view of our retrosynthesis model, the method here applies in general to any discrete diffusion model. We start by writing the Bayes' rule for an additional condition y (e.g., a specified level of drug-likeness or synthesizability, or an inpainting mask)

$$p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y) \propto p(y \mid \mathbf{X}_{t-1}, \mathbf{X}_t, \mathbf{Y}) p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}) \tag{80}$$

$$= p(y \mid \mathbf{X}_{t-1}, \mathbf{Y}) p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}) \tag{81}$$

where the second equation was due to the Markovian structure of the generative process ($\mathbf{X}_{t-1}$ d-separates y and $\mathbf{X}_t$). Now, we can take the log and interpret the probabilities as tensors $\mathbf{P}_\theta(y \mid \mathbf{X}_{t-1}, \mathbf{Y})$ and $\mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y})$ defined in the same space as the one-hot valued tensors $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$. We get:

$$\log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y) \propto \log \mathbf{P}(y \mid \mathbf{X}_{t-1}, \mathbf{Y}) + \log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}) \tag{82}$$

Similarly to (Vignac et al., 2023), we can now Taylor expand $\log \mathbf{P}_\theta(y \mid \mathbf{X}_{t-1}, \mathbf{Y})$ around $\mathbf{X}_t$ with

$$\log \mathbf{P}(y \mid \mathbf{X}_{t-1}, \mathbf{Y}) \approx \log \mathbf{P}(y \mid \mathbf{X}_t, \mathbf{Y}) + \nabla_{\mathbf{X}'_t} \log \mathbf{P}(y \mid \mathbf{X}'_t, \mathbf{Y})|_{\mathbf{X}'_t = \mathbf{X}_t} (\mathbf{X}_{t-1} - \mathbf{X}_t) \tag{83}$$

Given that we are interested in the distribution w.r.t. $\mathbf{X}_{t-1}$, the $\mathbf{X}_t$ terms are constant when we plug them in to Eq. (82), resulting in

$$\log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y) \propto \nabla_{\mathbf{X}'_t} \log \mathbf{P}(y \mid \mathbf{X}'_t, \mathbf{Y})|_{\mathbf{X}'_t = \mathbf{X}_t} \mathbf{X}_{t-1} + \log \mathbf{P}_\theta(\mathbf{X}_{t-1} | \mathbf{X}_t, \mathbf{Y}) \tag{84}$$

Simplifying the notation to assume taking the gradient at $\mathbf{X}_t$, we can also write

$$\log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y) \propto \nabla_{\mathbf{X}_t} \log \mathbf{P}_\theta(y \mid \mathbf{X}_t, \mathbf{Y}) \mathbf{X}_{t-1} + \log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}) \tag{85}$$

In practice, the equation means that given $\log p_\theta(y \mid \mathbf{X}_t, \mathbf{Y})$, we get $p_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y)$ by adding the input gradient of $\log p_\theta(y \mid \mathbf{X}_t, \mathbf{Y})$ to the logits given by the regular reverse transition and re-normalizing.

It would be possible to approximate $\log p_\theta(y \mid \mathbf{X}_t, \mathbf{Y})$ by training an additional classifier, leading to so-called classifier guidance (Song et al., 2021). However, if we have direct access to $p(y \mid \mathbf{X}_0)$ (e.g., a synthesizability model), there exists a wide range of methods developed for continuous diffusion models that provide different levels of approximations of the true conditional distribution (Chung et al., 2023; Dou and Song, 2024; Finzi et al., 2023; Boys et al., 2023; Wu et al., 2023; Peng et al., 2024). Here, we show an approach similar to (Chung et al., 2023) for discrete diffusion models by using the gradients with respect to the denoiser $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$

$$p(y \mid \mathbf{X}_t, \mathbf{Y}) = \sum_{\mathbf{X}_0} q(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}) p(y \mid \mathbf{X}_0) \tag{86}$$

$$\approx \sum_{\mathbf{X}_0} p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}) p(y \mid \mathbf{X}_0), \tag{87}$$

[2] This can be seen in the code shared by Igashov et al. (2024): `https://github.com/igashov/RetroBridge`

Algorithm 3 Sampling with atom-count guidance

Input: Product $\mathbf{Y}$
Choose: $P^{\mathbf{Y}\to\mathbf{X}} \in \mathbb{R}^{N_\mathbf{X}\times N_\mathbf{Y}}$
$\mathbf{X}_T \propto p(\mathbf{X}_T)$
for $t = T$ **to** 1 **do**
  $\tilde{\mathbf{X}}_0 = D_\theta(\mathbf{X}_t, \mathbf{Y}, P^{\mathbf{Y}\to\mathbf{X}})$ ▷ *Denoising output*
  $\tilde{\mathbf{X}}^i_{t-1} = \sum_k q(\mathbf{X}^i_{t-1} \mid \mathbf{X}^i_t, \mathbf{X}^i_0)\tilde{\mathbf{X}}^i_0$ ▷ *Regular reverse transition probabilities*
  $\mathbf{X}^i_{t-1} \sim \text{softmax}(\log \tilde{\mathbf{X}}_{t-1} + \gamma\nabla_{\mathbf{X}_t} \log\sigma(\frac{\sum_{i\in S}\tilde{\mathbf{X}}_{0,i,d}-a}{b}))$ ▷ *Renormalize*
Return $\mathbf{X}_0$

where $q(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$ is the true denoising distribution. This results in the following update step:

$$\log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}, y) \propto \nabla_{\mathbf{X}_t} \log\left(\mathbb{E}_{p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})} p(y \mid \mathbf{X}_0)\right) \mathbf{X}_{t-1} + \log \mathbf{P}_\theta(\mathbf{X}_{t-1} \mid \mathbf{X}_t, \mathbf{Y}). \tag{88}$$

Summing over all possible graphs $\mathbf{X}_0$ is prohibitive, however. Instead, we could sample $\mathbf{X}_0$ from $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$ with the Gumbel-Softmax trick (Jang et al., 2016) and evaluate $\log p(y \mid \mathbf{X}_0)$. As long as $\log p(y \mid \mathbf{X}_0)$ is differentiable, we can then just use automatic differentiation to get our estimate of $\nabla_{\mathbf{X}_t} \log \mathbf{P}_\theta(y \mid \mathbf{X}_t, \mathbf{Y})$. Another, more simplified approach that avoids sampling from $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$ is to relax the definition of the likelihood function to directly condition on the continuous-valued probability vector $\mathbf{P}_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$ instead of the discrete-valued $\mathbf{X}_0$. For simplicity, in the next section, we adopt this approach, but the full method with Gumbel-Softmax is not significantly more difficult to implement.

D.1. Toy Synthesisability Model: Controlling Atom Economy

One obvious use-case for posthoc conditioning in the retrosynthesis context is to increase the probability of the generated reactants being synthesisable, using some pre-trained synthesisability model. Synthesisability of the generated reactants is crucial in the multi-step retrosynthesis context since otherwise, the generated branch of the search path is a dead end. The idea is that if we can improve the probability of synthesisability of the generated reactants, this should result in a much smaller search space for the multi-step model. To showcase the idea, we use a toy synthesisability model based on total the count of atoms in the reactants. The size of the reactants is indeed a feature used in synthesisability models (Ertl and Schuffenhauer, 2009), with the motivation that smaller reactants are more likely to be synthesisable than large and complex ones. The total count of atoms in a reaction is also referred to as the *atom economy* (Trost, 1991) of the reaction. It is the efficiency of a chemical process in converting all involved atoms into the desired products and is a key factor in creating synthesis routes that are efficient with respect to the raw materials. The ability to control the atom economy of a single-step retrosynthesis model thus has the potential to improve the efficiency of synthesis routes, in contrast to simply finding correct synthesis routes.

The model of synthesizability that we use is

$$\tilde{\mathbf{X}}_0 = p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y}), \tag{89}$$

$$p(y = \text{synthesizable} \mid \tilde{\mathbf{X}}_0) = \sigma(\frac{\sum_{i\in S}\tilde{\mathbf{X}}_{0,i,d} - a}{b})^\gamma, \tag{90}$$

where S is the set of non-atom-mapped nodes, a, b and γ are constants that define the synthesizability model and d refers to the dummy node index. The intuition is that the more nodes are classified as dummy nodes (non-atoms), the fewer atoms we have in total, leaving the atom economy higher. Note that $\sum_{i\in S}\tilde{\mathbf{X}}_{0,i,d}$ is the expected amount of dummy nodes from $p_\theta(\mathbf{X}_0 \mid \mathbf{X}_t, \mathbf{Y})$. We set a to half the amount of dummy nodes and b to one-quarter of the amount of dummy nodes. It turns out that this leaves γ as a useful parameter to tune the sharpness of the conditioning. The gradient estimate is then given by

$$\nabla_{\mathbf{X}_t} \log \mathbf{P}_\theta(y \mid \mathbf{X}_t, \mathbf{Y}) = \gamma\nabla_{\mathbf{X}_t} \log\sigma(\frac{\sum_{i\in S}\tilde{\mathbf{X}}_{0,i,d} - a}{b}), \tag{91}$$

which can be directly calculated with automatic differentiation. The full algorithm is detailed in Alg. 3. Figure A6 shows the effect of increasing γ on an example product.

D.2. A Case Study on Interactive Generation With Inpainting

We replicate BHC's green synthesis of Ibuprofen hypothetically using our model interactively with inpainting. Fig. A7 compares the output of our model to the ground truth synthesis. Below we explain the synthesis steps in detail:

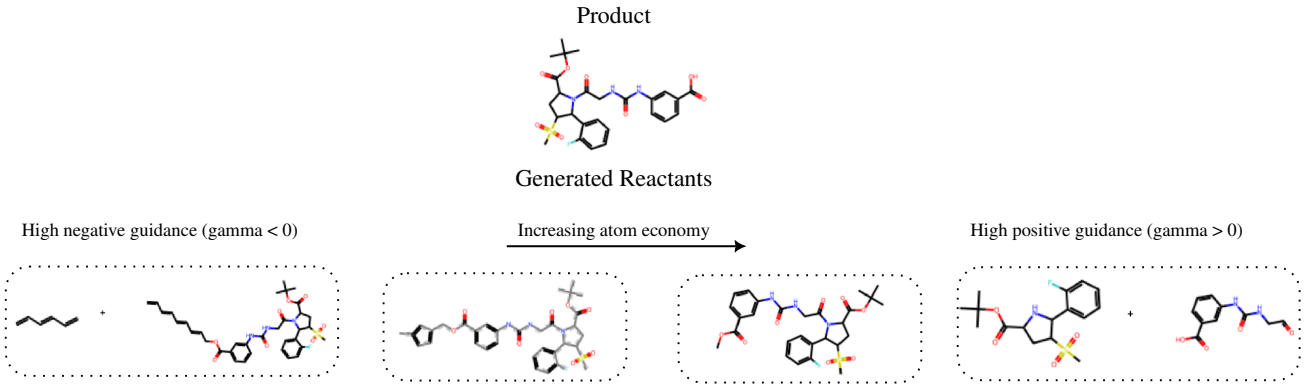


Figure A6: The effect of increasing the γ parameter. With negative γ, the model outputs large reactants, and with positive γ, the model outputs smaller ones, allowing for fine-grained control over the *atom economy* of the generated step.

1. The retrosynthesis path starts with carbonylation, a simple and well-known synthetic reaction which adds a 'CO' structure to a compound. The practitioner tries basic generation but then notices that the suggested reactant is not promising. They then suggest a partial structure of the reactants which could lead to a more sensible path. Our retrosynthetic model can complete the reactants with this information.

2. Next, our model proposes hydrogenation, which is a sensible suggestion in this case. The data does not consider explicitly individual H2 molecules, meaning that they are inferred from the context.

3. In the third step, regular generation again seems off, but the practitioner notices that an acylation reaction (a reaction with the C(=O)C group) might lead to reactants that are readily available. The model is able to complete the rest of the reaction after knowing that C(=O)C is present. These steps match the synthesis plan proposed by BHC.

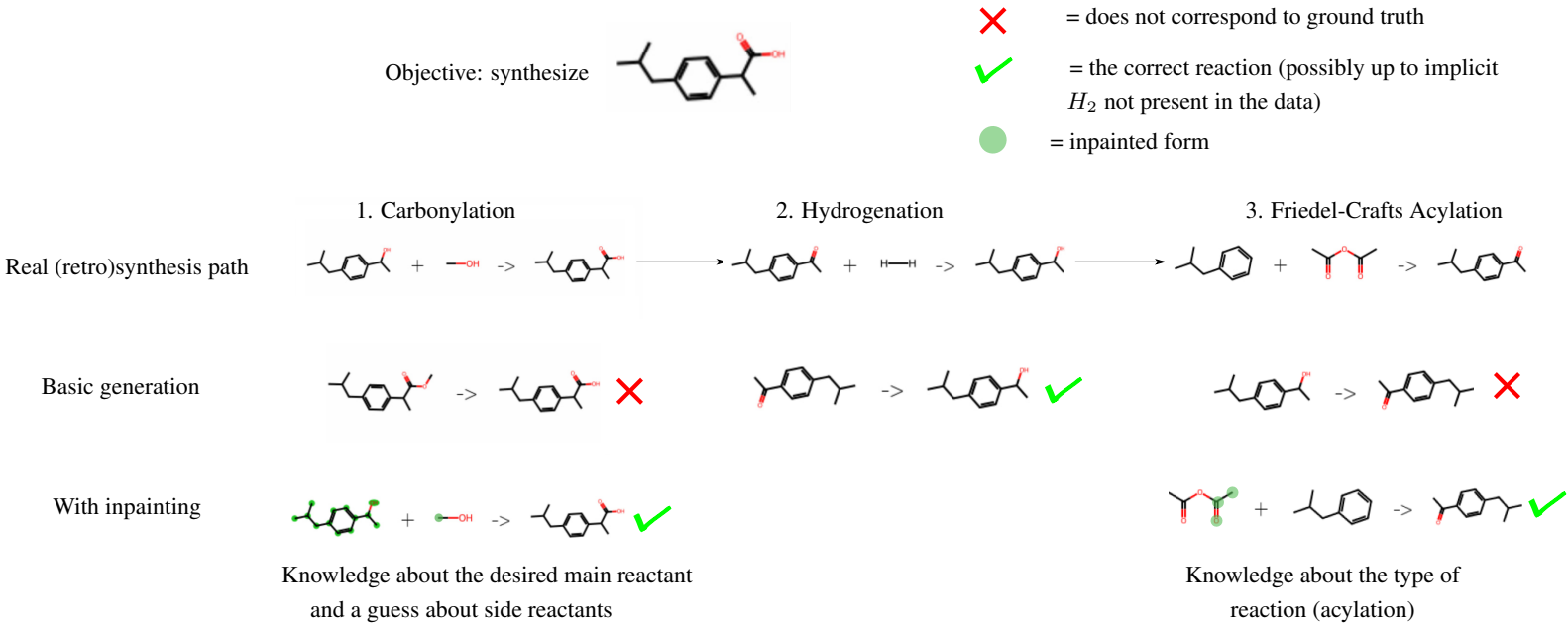


Figure A7: Replicating BHC's green synthesis of Ibuprofen using our model interactively, and comparing to the known synthesis path.