ARROW FLOW MATCHING FOR REACTION MECHANISMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Chemists explain a reaction by drawing arrows. Each arrow moves one pair of valence electrons, and an elementary step made of such arrows creates no atom and no electron. We propose a generative model of these arrows. Its state is an integer bond–electron matrix, and its dynamics is a Markov chain trained by discrete flow matching. The central design choice is the *elementary event*, the change that one update of the chain applies to the state. The common choice is an edit of one entry of the matrix. We prove that a model which draws every entry independently conserves the electron count exactly only if it is deterministic in every entry, whatever its parameters. Arrow Flow Matching instead takes as its elementary event one *move* from a four-element alphabet, the matrix form of an arrow that chemists draw. Every move leaves the electron count unchanged, and the chain may stop only where a table of valid per-atom states allows it. Conservation therefore reduces to an algebraic identity and validity to a lookup on one softmax class, and both hold with probability one. As every update is an arrow, the trajectory of the model reads as a mechanism that a chemist can inspect. These guarantees cost almost no accuracy. Arrow Flow Matching is competitive with the most accurate model we train, and at its smallest width it already reaches a higher composite score than every other model at every width.

1 INTRODUCTION

Chemistry explains a reaction by arrows drawn on the reactants. Each arrow moves one pair of valence electrons, and an ordered collection of them turns reactants into products. We refer to this ordered sequence of elementary steps as the *mechanism*, and to one recorded such sequence as a *pathway*. Models that predict reaction outcomes are already in use. They plan syntheses for robotic platforms (Segler et al., 2018; Coley et al., 2019b; Mikulak-Klucznik et al., 2020), and they rank the reactions worth trying (Ahneman et al., 2018; Dai et al., 2019; Xu et al., 2025). A model of the mechanism gives more, the route the electrons take rather than the outcome alone (Ren et al., 2026; Das et al., 2026). An elementary step creates no atom, and it moves only valence electrons, between lone pairs and bonds. A model of a step must therefore conserve atoms and electrons exactly and return a chemically valid molecule. For machine learning, this is a discrete problem on a fixed atom set, with one global constraint, a sum over the state, and one local constraint per atom.

Three approaches to reaction prediction. The dominant approach translates a reactant string (Weininger, 1988) into a product string (Nam & Kim, 2016; Schwaller et al., 2018; 2019; Tu & Coley, 2022). These models are accurate and remain strong baselines. However, a string holds no inventory of atoms or electrons, and nothing stops an incorrect decoding from creating or losing either. A second family edits the molecular graph at a predicted reaction centre (Jin et al., 2017; Coley et al., 2019a; Sacha et al., 2021; Somnath et al., 2021; Chen & Jung, 2021), which preserves the atoms but not the electron count. A third family predicts the arrows themselves. They rank or generate the arrows of polar, pericyclic and radical steps (Kayala & Baldi, 2012; Miller et al., 2025). ELECTRO (Bradshaw et al., 2019) generates a reaction as a path of electron movements, and Reactron (Chen et al., 2025) emits the arrow-pushing diagram of each step. Recent language models write electron flow into an annotated string (Neukomm et al., 2025). The FlowER of Joung et al. (2025), here called *Continuous FlowER*, transports the reactant bond–electron matrix (Dugundji & Ugi, 1973) into the product matrix by continuous flow matching (Lipman et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2023). MAELLE (Nguyen et al., 2026) learns the rearrangement of electrons by discrete flow matching, but on electron-pair counts over a heavy-atom graph, not on the bond–electron matrix.

Three design axes. A model of an elementary step should return chemically valid molecules, conserve electrons, and be interpretable. Interpretability means that the internal states of the model follow the arrows by which a chemist reasons about the step, so that a chemist can check a prediction arrow by arrow. Every candidate of ELECTRO is chemically valid, as it only moves electrons over a structure that already exists. However, its arrows form a single chain that alternates breaking a bond with forming one. This rules out radical steps and steps in which several bonds change at once, and ELECTRO reaches a top-1 step accuracy of 0.662, against 0.872 for the best model in our comparison. Continuous FlowER conserves the total number of electrons along its trajectory, but its final matrix is not integer and has to be rounded. The rounding preserves the total but can break the symmetry of the matrix, create negative entries or overfill an atom, so part of what it returns is not a valid molecule. MAELLE recovers electron flows that agree with known mechanisms, but it guarantees neither conservation nor validity. No existing model satisfies all three requirements.

Contributions. We build two generative models of one elementary step on the same network. They differ only in the elementary event, and the second one satisfies all three requirements.

- We propose Discrete FlowER (§3, Appendix D), which applies discrete flow matching directly to integer bond–electron matrices. Every state it visits is an integer matrix, so it removes the rounding of Continuous FlowER, and it reaches the highest top-1 step accuracy in our comparison.
- We show that removing the rounding is not enough. Discrete FlowER edits one entry at a time, and its head draws the entries independently. We prove that such a head places all its mass on electron-conserving matrices only if it is deterministic in every entry (§4, Theorem 1).
- We propose Arrow Flow Matching (§5), whose elementary event is one of four moves, the matrix forms of a chemist's arrows. Each move conserves electrons by construction (Proposition 2), and the chain may stop only in a state that a per-atom table accepts, so every candidate is chemically valid (Proposition 3). Every update is an arrow, so the trajectory reads as a mechanism.
- Across five widths and eight baselines (§6), Arrow Flow Matching is the only model whose candidates are always chemically valid and always keep the reactant's electron, atom and proton counts. Its accuracy matches that of the most accurate model we train, near the ceiling that branching imposes on any deterministic model, and, already at its smallest width, its composite score is higher than that of every other model at every width.

Appendix A discusses the related work in detail.

2 The object and the constraint

Chemical background. A molecule is a labelled graph that records a count on every vertex and every edge. The vertices are atoms, each labelled by its element. The count of an edge is its *bond order* $\in \{1, 2, 3\}$, the number of electron *pairs* the two atoms share. The count of a vertex is its number of *lone-pair electrons*, which belong to that atom alone (Lewis, 1916). Electrons usually move in pairs[1], and chemists draw a step as curly arrows, each taking one pair from where it is to where it goes. A single electron is moved by a *fish-hook*, a curly arrow with one barb. An arrow has one tail and one tip, because the pair leaves one place and arrives at another at once. Finally, each atom has a limited capacity. Carbon, nitrogen and oxygen take at most four pairs in total, lone and shared together, and hydrogen at most one. This capacity constraint on a vertex is the *octet rule*. Each of these three facts becomes one piece of the construction. The counts become the entries of the bond–electron matrix below, the arrows become the move alphabet of §5, and the octet rule becomes the condition under which the chain may stop.

The bond–electron matrix. Fix the N mapped atoms of one step and let $a = (a_1, \ldots, a_N)$ record their elements. The atom map is a bijection between the reactant and product sides, and a is therefore the same vector on both and never changes. A set of molecules on these atoms is encoded by a symmetric $N \times N$ matrix B, the *bond–electron matrix* of Dugundji & Ugi (1973). Off the diagonal, B_{ij} is the bond order of the pair $\{i, j\}$. On it, B_{ii} is the number of lone-pair *electrons* on atom i. The units therefore differ on and off the diagonal, electrons on it and pairs off it. That is the convention of the bond–electron matrix and the source of every factor of two below. An off-diagonal *entry*

[1] An orbital holds at most two electrons (the Pauli exclusion principle), and stable molecules usually fill it.

is printed twice, at (i, j) and at (j, i), but it is one entry. Sums and products over the entries of a symmetric matrix x therefore run over $i \le j$, and x_{ij} is the entry at (i, j). Write B^r, B^p for the reactant and product matrices and set $\Delta := B^p - B^r$.

Per-atom counts and the electron count. The *bonding-pair count* β_i is the number of pairs that atom i shares with its neighbours. The *total pair count* c_i is the number of pairs at atom i, lone and shared together, and the octet rule caps it at four,

$$\beta_i(x) = \sum_{j \neq i} x_{ij}, \qquad c_i(x) = \tfrac{1}{2} x_{ii} + \beta_i(x), \qquad c_i(x) \le 4. \tag{1}$$

The duet rule of hydrogen caps it at one. Every hydrogen is an explicit, mapped atom with its own row, so β_i is a plain row sum (Remark 4). The *electron count* $\mathcal{E}$ is the global counterpart of β_i, a linear functional of the whole matrix given by the sum of its entries,

$$\mathcal{E}(x) := \sum_{i,j} x_{ij} = \sum_i x_{ii} + 2 \sum_{i<j} x_{ij}, \qquad \mathcal{E}(\Delta) = 0. \tag{2}$$

The condition $\mathcal{E}(\Delta) = 0$, that is $\mathcal{E}(B^p) = \mathcal{E}(B^r)$, is the physical law the model must respect.

The state space. Every entry of B is a small non-negative integer (Figure 3), so the matrices a model may visit form the *state space*

$$\mathcal{X} = \{x \text{ symmetric} : x_{ij} \in \mathcal{V}_{ij}\}, \qquad \mathcal{V}_{ii} = \{0, \dots, 14\}, \quad \mathcal{V}_{ij} = \{0, \dots, 3\} \ \ (i < j), \tag{3}$$

whose ranges are fixed by the representation and cover the data. Both models run on $\mathcal{X}$.

Problem definition. Given B^r, we require a conditional distribution $p_\theta(\cdot \mid B^r)$ that can be both *sampled* and *scored*, as a search over whole pathways needs a ranked list of candidates at every step. A candidate $\widehat{B}$ is useful only if it is a valid molecule, which amounts to four conditions. It must carry the reactant's multiset of heavy atoms (R1), its count of hydrogens (R2) and its electron count (R3). It must also be *chemically valid* (R4), meaning that every atom obeys the *valence rules* of its element, the octet rule (1) and its counterparts for the other elements. For a matrix model R1 and R2 are automatic, the atom map fixing the vertex set, and R3 is a condition on the matrix alone (Appendix C.1). A perfect score on all four checks (R1–R4) is therefore the target

$$\Pr\left[\mathcal{E}(\widehat{B}) = \mathcal{E}(B^r)\right] = \Pr\left[\widehat{B} \text{ is chemically valid}\right] = 1. \tag{4}$$

Training steps (B^r, B^p) have the empirical law π, and the element vector a is left implicit.

The conservation sheet and the physical sector. The first condition of (4) is one linear equation on the matrix. The second is not, as valency is a rule per element, but it is local. Whether atom i obeys the valence rules depends only on its *local state* $(a_i, x_{ii}, \beta_i(x))$, which consists of its element, its lone-pair electron count and its bonding-pair count. Let $\mathcal{T}$ be the table of local states that obey the valence rules. Each component of a local state ranges over a finite set, so $\mathcal{T}$ is finite (§5 constructs it). With $n := \mathcal{E}(B^r)$, the conserving matrices form the set $\mathcal{S}_n$, and the valid molecules among them form its subset $\mathcal{M}$,

$$\mathcal{S}_n = \{x \in \mathcal{X} : \mathcal{E}(x) = n\}, \qquad \mathcal{M} = \{x \in \mathcal{S}_n : (a_i, x_{ii}, \beta_i(x)) \in \mathcal{T} \text{ for every } i\}. \tag{5}$$

We call $\mathcal{S}_n$ the *conservation sheet* and $\mathcal{M}$ the *physical sector*. The sheet is the first condition of (4), and the sector adds the N local conditions of the second. Both endpoints of every training pair lie in $\mathcal{M}$. The construction of §5 rests on this decomposition into one global and N local conditions.

Appendix B collects the notation, and Appendix C expands this section (Figure 3).

3 Discrete flow matching over the matrix

Molecules are discrete, and every entry of a bond–electron matrix is an integer. A continuous flow leaves this domain and must round its endpoint, so the first model stays in $\mathcal{X}$. It applies discrete flow matching (Campbell et al., 2024; Gat et al., 2024) to $\mathcal{X}$ without modification. We call it **Discrete FlowER** and derive it in full in Appendix D. The process starts at the reactant B^r rather than at a random prior, so its path is a Markov bridge pinned at two data points, B^r at $t = 0$ and B^p at

$t = 1$. Each entry independently holds its reactant value with probability $1 - t$ and its product value otherwise. The schedule is linear in t, the only one used here,

$$p_t\big(x \mid B^r, B^p\big) = \prod_{i \le j} \Big[(1-t)\,\mathbb{1}[x_{ij} = B^r_{ij}] + t\,\mathbb{1}[x_{ij} = B^p_{ij}]\Big]. \tag{6}$$

The rate that generates the average of (6) over the data depends only on the posterior distribution of the product value of each entry (Appendix D.2), so the model predicts the product value of every entry. A *backbone network* maps the reactant, the current state and the time to one vector per atom, and an output layer turns these vectors into logits. We call the composition the *network* $\ell_\theta(x, t)$. It returns one logit vector ℓ_{ij} per entry, over the values $\mathcal{V}_{ij}$ of that entry. For logits ℓ over a finite class set $\mathcal{Y}$, we write $\mathrm{softmax}_{\mathcal{Y}}(\ell)$ for the distribution on $\mathcal{Y}$ that gives each class y a mass proportional to $\exp \ell_y$. The *head* is the distribution that these logits define over the next state. Here it draws each entry from its own softmax, and training is a plain cross-entropy at states drawn from the bridge,

$$p_\theta\big(x' \mid x, t\big) = \prod_{i \le j} p^{ij}_\theta\big(x'_{ij} \mid x, t\big), \qquad p^{ij}_\theta\big(\cdot \mid x, t\big) = \mathrm{softmax}_{\mathcal{V}_{ij}}\big(\ell_{ij}\big), \qquad \ell = \ell_\theta(x, t), \tag{7}$$

$$\mathcal{L}(\theta) = \mathbb{E}_{(B^r, B^p)\sim\pi}\, \mathbb{E}_{t\sim\mathcal{U}[0,1]}\, \mathbb{E}_{x_t \sim p_t(\cdot \mid B^r, B^p)} \Big[- \sum_{i \le j} \log p^{ij}_\theta\big(B^p_{ij} \mid x_t, t\big)\Big], \tag{8}$$

whose minimiser recovers the exact marginal generator. We call the smallest change of the state to which a head assigns a probability the *elementary event* of the model. The elementary event of Discrete FlowER is an *edit*, the change of one entry to a new value. Sampling is an Euler scheme over S updates. Each update resamples every entry from its endpoint distribution with a probability rising from $1/S$ to 1, and leaves it untouched otherwise, so one update applies several independent edits. The candidate $\widehat{B}$ is the per-entry arg-max of the head at the last state. Its *score* $s(\widehat{B})$ is the head's log-probability of that candidate, and it covers the last update only. Every intermediate state is an integer matrix, so decoding is the identity map and nothing is rounded, projected or repaired (Figure 5 in Appendix D.5). The head, however, never couples two entries.

4 The limitation of the entrywise head

Discrete FlowER stays in $\mathcal{X}$ and yet cannot meet (4), because it factorises over entries in its training path (6) and in its head (7). On the path, the electron count is correct only in expectation (Appendix E.2). Here we show the consequence for the head.

The head. Every conserving matrix lies on the conservation sheet $\mathcal{S}_n$ of (5), including B^r and B^p. The state space $\mathcal{X}$ is a *box*, a product of one set of values per entry. The sheet is a slice of it, cut out by one linear equation, and is not a box itself. The support of a product measure, by contrast, is always a box, the product of the supports of its factors.

Theorem 1 (A product measure cannot live on the conservation sheet)**.** *Let $\mu = \bigotimes_{i \le j} \mu_{ij}$ be a product measure on $\mathcal{X}$ with $\operatorname{supp}\mu \subseteq \mathcal{S}_n$ for some fixed n. Then every μ_{ij} is a point mass.*

The proof is given in Appendix E.1, and Figure 1(a,b) draws the two shapes it compares. The entrywise head (7) is a product measure at every state, and so is any update built from it. As soon as it is uncertain about the product value of even one entry, part of its mass therefore falls on matrices with the wrong electron count. The cause is the product form of (7), not the values of θ, so neither more data nor more training removes it. The same argument applies to validity, as the physical sector lies inside the sheet. The candidate $\widehat{B}$ of §3 is an arg-max and thus deterministic, so the theorem does not apply to it. It conserves electrons only when its errors on individual entries cancel, which (7) does not encourage. Figure 6 (Appendix E.4) draws a deterministic head that conserves and a spread one that does not.

Changing the elementary event. Theorem 1 identifies the component to change. The loss is a proper scoring rule, and the backbone network already conditions on the whole matrix, so neither is the cause. The remaining component is the elementary event. If every elementary event adds an increment A with $\mathcal{E}(A) = 0$, it keeps the state on the sheet, and the head can spread its mass over any subset of the sheet (Figure 1c). Section 5 constructs such an elementary event. An autoregressive head that draws the entries one after another, each conditioned on those already drawn, is not a

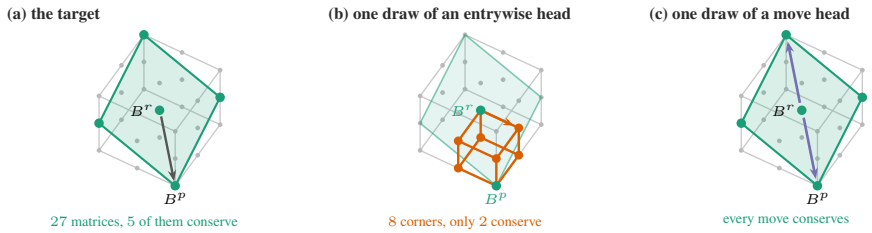


Figure 1: Theorem 1, drawn on three entries (Figure 5) with the rest of the matrix frozen. All 27 lattice points are matrices, and the five carrying the reactant's electron count form the rhombus $x_{11} + x_{44} + 2x_{14} = n$, the sheet $\mathcal{S}_n$ in (a). Panels (b) and (c) both run from B^r to B^p, and differ only in what a single draw of the head may change. In (b) every entry is drawn on its own, so one draw lands on any corner of the orange box, and six of the eight corners miss the rhombus. In (c) one draw is a whole move and lands back on the rhombus, and the two blue moves are the HOMOLYSIS and its inverse the COLLIGATION.

product measure and also avoids the theorem. It would need one pass of the network per entry, $N(N+1)/2$ passes for a single update, and would still have to force its last entries onto the sheet. A move of §5 costs one pass and is an arrow a chemist draws. Appendix E.3 explains why imposing the constraint only at decoding, by projecting or filtering the output, does not solve the problem.

5 Arrow Flow Matching

Arrow Flow Matching replaces the edit of Discrete FlowER by a *move* as its elementary event. An edit changes one entry and therefore always changes the electron count. A move changes several entries jointly. A pair move takes one pair from one entry, its tail, and places it at another entry, its tip, so the electron count stays the same. The move must also be named by few enough indices that the network can score every move of the alphabet in one fixed tensor of logits, the same shape at every state. The backbone network and the cross-entropy carry over from §3, while the bridge, the conditioning and the sampler are rebuilt on moves. Table 10 (Appendix H) compares both models.

Four move types. The alphabet has *four* move types τ, listed in Table 1. A move $y = (\tau, i, j)$ is named by its type and an ordered pair of distinct atoms, so the network scores a dense $(4, N, N)$ grid of classes. It changes at most three entries, x_{ii}, x_{jj} and $x_{ij} = x_{ji}$, by the increments δ_τ^{ii}, δ_τ^{jj} and δ_τ^{ij}, and maps a state x to $x + A_y$, where A_y is zero except for these increments,

$$\begin{bmatrix} (A_y)_{ii} & (A_y)_{ij} \\ (A_y)_{ji} & (A_y)_{jj} \end{bmatrix} = \begin{bmatrix} \delta_\tau^{ii} & \delta_\tau^{ij} \\ \delta_\tau^{ij} & \delta_\tau^{jj} \end{bmatrix}, \qquad \mathcal{E}(A_y) = \delta_\tau^{ii} + \delta_\tau^{jj} + 2\,\delta_\tau^{ij} = 0. \tag{9}$$

By (2), $\mathcal{E}(A_y)$ is the sum in the second identity, and the fifth column of Table 1 shows that it vanishes for every type. Figure 2 draws each row.

Table 1: The four move types.

type τ	what it does	δ_τ^{ii}	δ_τ^{jj}	δ_τ^{ij}	$\mathcal{E}(A_y)$	requires
LONE→BOND	lone pair on i becomes a bond to j	-2	0	$+1$	0	$x_{ii} \geq 2, x_{ij} \leq 2$
BOND→LONE	bond i–j breaks, pair onto j	0	$+2$	-1	0	$x_{ij} \geq 1$
HOMOLYSIS	bond i–j breaks, one electron each	$+1$	$+1$	-1	0	$x_{ij} \geq 1$
COLLIGATION	one electron on each of i, j pair up	-1	-1	$+1$	0	$x_{ii}, x_{jj} \geq 1, x_{ij} \leq 2$

The first two rows are pair moves, the arrows from (i, i) to (i, j) and from (i, j) to (j, j). Appendix F.1 shows that every arrow conserves $\mathcal{E}$ whichever two entries it joins, as one pair is worth two electrons wherever it sits. The last two rows are single-electron moves, each drawn as two fish-hooks. They are needed whenever a step changes a diagonal entry by an odd amount, which the pair moves cannot do. Radical chemistry does this (Appendix F.3). The two pair moves depend on the order of i and j. The two single-electron moves do not, so they are offered only for $i < j$, and each move is exactly one class. Appendix F establishes two facts. First, the third arrow type used by chemists is the *bond migration*, from the bond (i, j) to an adjacent bond at j. It is named by an atom *triple* and would make the class set ragged, a different set of classes at every state. It is also the composition of two halves, a BOND→LONE followed by a LONE→BOND at the shared atom, and is therefore not needed (Remark 7). Second, the alphabet is closed under inversion, so *no reachable state has an empty admissible set* (Remark 8).

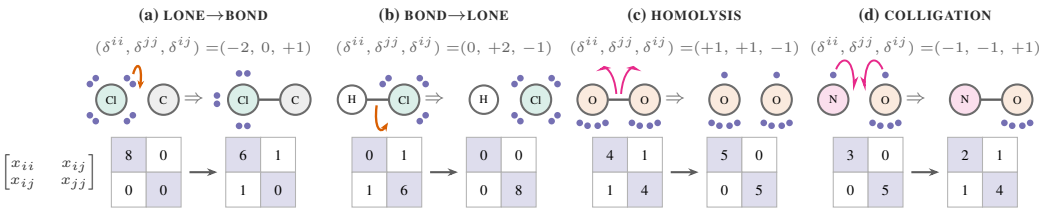


Figure 2: The move alphabet. Above, a chemist's arrows on the smallest fragment showing each type. Below, the matrix entries each type touches, before and after. An orange double-barbed arrow moves a pair, a magenta fish-hook moves a single electron, and a dot is one valence electron. Columns (a) and (b) are pair moves, which take a whole pair from one entry to another. Columns (c) and (d) are single-electron moves. In the matrices the shaded cells are the diagonal and count *electrons*. The plain cell is the bond and counts *pairs*, which the symmetric layout prints twice. So the diagonal changes by 2 where the off-diagonal changes by 1. Every column satisfies (9), so $\mathcal{E}$ is unchanged. Columns (a) and (b) are inverse to one another, and so are (c) and (d).

The two guarantees. The construction discharges the global condition of (5) by the *algebra* of the moves, and the local ones by a *gate* on the stop class.

Proposition 2 (Conservation)**.** *Let $(x_k)_{k\geq 0}$ be any chain every update of which is a move of the alphabet. Then $\mathcal{E}(x_k) = \mathcal{E}(x_0)$* surely*, not in expectation, and however the moves are chosen.*

The proof is an induction on k using the linearity of $\mathcal{E}$ and (9) (Appendix F.4). A move is *admissible* at x when the new state $x + A_y$ still lies in the state space $\mathcal{X}$ of (3) and the bonding-pair counts of its two atoms stay below the bound $\bar{b} = 12$ of the representation, which is above any capacity the valence rules impose,

$$\mathcal{A}(x) = \big\{y = (\tau, i, j) : \; x + A_y \in \mathcal{X}, \quad \beta_i(x + A_y) \leq \bar{b}, \quad \beta_j(x + A_y) \leq \bar{b}\big\}. \tag{10}$$

A move changes only the entries (i,i), (j,j) and (i,j) and the bonding-pair counts β_i and β_j, so the three clauses reduce to five scalar tests, one on each of these three entries and one on each of the two counts. Appendix G.1 works them through at one real reactant, and Figure 9 draws the whole of $\mathcal{A}(x)$ there.

Proposition 3 (Validity)**.** *Let a chain start at B^r, take moves in $\mathcal{A}(x_k)$, and be permitted to terminate only at a state x with $(a_i, x_{ii}, \beta_i(x)) \in \mathcal{T}$ for every i. Then it terminates in $\mathcal{M}$.*

The proof is in Appendix G.7. Proposition 3 turns validity from a check after generation into a *precondition of the stop class*. This is not repair, as no candidate is filtered, projected or rounded.

The gate on the stop class. The table $\mathcal{T}$ is fixed once, before any training, by listing every local state and keeping those that obey the valence rules of their element (Appendices G.3 and G.4), so it is not a modelling choice. At run time it is a boolean tensor indexed by (a_i, x_{ii}, β_i), so the gate is one lookup per atom. For almost every element, $\mathcal{T}$ keeps the local states whose total pair count c_i of (1) is at most a capacity C_{a_i} of the element (Appendix G.2), and the octet rule is the case $C_{a_i} = 4$. Only the stop class is gated, because the valence rules describe the molecules before and after a step, and Appendix G.5 shows why a gate on every move would block a chain of arrows.

The bridge over moves. The move bridge takes the place of the bridge (6). A state of (6) applies a random subset of the edits of a step, whereas a state of the move bridge applies a prefix of its ordered moves. Every such state lies on the sheet, and the network is trained on exactly the states that the sampler visits. *Decomposition.* We factorise Δ into moves the way a chemist factorises a step into curly arrows. Such a factorisation always exists, as every degree-preserving transformation of this kind decomposes into cyclic electron-pushing diagrams (Flamm et al., 2023). We expand each bond migration into its two halves (Remark 7) and take each odd-parity change by HOMOLYSIS or COLLIGATION, which writes Δ as a sum of m moves. *Ordering.* The m moves are a *multiset*, and every intermediate state must satisfy (10). We order them by a greedy admissible ordering, computed once offline (Appendix H.2). *The path.* The ordered moves $y_0, \ldots, y_{m-1}$ and the stop class $y_m := \varnothing$ give the bridge

$$x_0 = B^r, \qquad x_{k+1} = x_k + A_{y_k} \;\text{ with }\; y_k \in \mathcal{A}(x_k) \quad (k < m), \qquad x_m = B^p, \tag{11}$$

whose states are the *prefixes* of the ordered moves, the subscript k counting the moves applied so far. Every prefix is admissible whenever the greedy ordering succeeds, which it did for every step of

the data[2]. Training draws $k \sim \mathcal{U}\{0, \dots, m\}$, a uniform prefix length on a path of $m+1$ points. The move count k plays the role that t plays in (6), and the head is conditioned on it. An *identity step*, one with $m = 0$ in which the reactant is the product, contributes only $k = 0$ and marks the end of a pathway. Figure 8 in the appendix shows this paragraph on a real step.

The head and the loss. The backbone network of §3, with the move count in place of the time, turns the reactant and the current state into one vector z_i per atom (Appendix I.1). The output layer scores each of the four move types τ at the pair (i, j) by a small bilinear product of z_i and z_j, plus a bias learned for the pair's entry code, and reads one *stop* logit off the pooled molecule (Figure 12, Appendix I.3). The *class set* $\mathcal{Y}(x)$ at a state x, over which the softmax of §3 normalises, is $\mathcal{A}(x)$, joined by the stop class $\varnothing$ when $x \in \mathcal{M}$. The head and the loss take the form of (7) and (8),

$$p_\theta\big(\cdot \mid x, k\big) = \mathrm{softmax}_{\mathcal{Y}(x)}\big(\ell\big), \qquad \ell = \ell_\theta(x, k), \tag{12}$$

$$\mathcal{L}(\theta) = \mathbb{E}_{(B^r, B^p)\sim\pi}\, \mathbb{E}_{k\sim\mathcal{U}\{0,\dots,m\}}\Big[-\log p_\theta\big(y_k \mid x_k, k\big)\Big], \tag{13}$$

The two heads differ in two places only. The entrywise head (7) is a product of one softmax per entry. The move head (12) is a single softmax, over the classes admissible at the current state. Propositions 2 and 3 therefore hold for the *parameterisation*, not only for the sampler. An inadmissible move is not a move given low probability but a class the distribution does not contain.

The trajectory likelihood factorises over its moves, so training is again a plain cross-entropy, with the endpoint target of (8) replaced by the next class. Drawing k uniformly weights every class of a step by $1/(m+1)$, so (13) is a length-normalised cross-entropy on the ordered trajectory rather than a likelihood of it. The target is admissible by construction, with $y_k \in \mathcal{A}(x_k)$ for $k < m$ by the ordering and $B^p \in \mathcal{M}$, so it always carries finite mass and the loss is finite for every example. Stopping is an ordinary class of an ordinary softmax, competing with the moves, so *when to stop is learned exactly as what to do next is learned*. As in §3, the cross-entropy fits the whole distribution of the next class rather than its most likely value (Remark 6).

Sampling, ranking and cost. Candidates are produced by running the chain forward, each chain carrying its own running matrix. After every move the state is re-encoded and (12) is formed, and the chain then either stops or applies one more move,

$$\begin{aligned} &x_0 = B^r, \qquad y_k \sim p_\theta\big(\cdot \mid x_k, k\big), \qquad x_{k+1} = x_k + A_{y_k} \;\; (k < \hat{m}), \\ &\widehat{B} = x_{\hat{m}}, \qquad s\big(\widehat{B}\big) = \sum_{k \le \hat{m}} \log p_\theta\big(y_k \mid x_k, k\big), \end{aligned} \tag{14}$$

where $\hat{m}$ is the first k with $y_k = \varnothing$. This is the bridge (11) with the classes drawn rather than given, hence the same symbols. The score $s(\widehat{B})$ is now the exact log-probability of the whole trajectory, not of the last update as in §3, so candidates can be ranked by score as well as by sampling frequency. The model also admits a *beam* decoder, which Discrete FlowER does not (Appendix J.1). A chain that has not stopped within S moves emits the last state at which it *was* allowed to stop, which always exists because $B^r \in \mathcal{M}$. The mask costs no throughput, as it is one boolean tensor of fixed shape (Appendix I.4).

6 Experiments

The experiments address five questions. **Q1.** Do the two guarantees hold in practice, exactly and at every width, and does any other model provide them? **Q2.** What is the trade-off between them and predictive accuracy? **Q3.** How many parameters does the construction need, since a constraint that has to be learned costs parameters and one built in should not? **Q4.** Does the mechanism a model computes give the arrows a chemist draws? **Q5.** Do the guarantees hold, and what do they cost in accuracy, when the test steps move away from the training steps?

Datasets. We use the mechanistic dataset released with Continuous FlowER (Joung et al., 2025), in which expert-curated mechanistic templates are applied to USPTO reactions to impute an ordered pathway of elementary steps for each recorded transformation. We take the released splits unchanged (Appendix K.1).

[2]The mechanistic dataset of Joung et al. (2025), described in §6 and Appendix K.1. The greedy ordering succeeded on every step of its training and test splits.

Table 2: All nine models at the smallest width, roughly 1.3M parameters, under sampling with frequency ranking. Every entry is the mean over 10 random seeds, and Table 16 in Appendix L.2 gives the standard deviations. Path@1 is the recursive pathway metric, and the last four columns are validity and electron, heavy-atom and proton conservation, each averaged over the candidate budget. The last column is the composite score of §6. In each column, the **1st**, **2nd** and **3rd** best entry is coloured, ties sharing a place. Figure 15 in Appendix L.2 plots the same comparison.

model	params	top-1	top-3	top-5	top-10	path@1	valid.	elec.	atom	prot.	score
Molecular Transformer	1.29 M	0.6446	0.7540	0.7758	0.7855	0.3770	0.9036	0.7303	0.7237	0.6939	0.6284
Graph2SMILES	2.36 M	0.7347	0.8549	0.8745	0.8841	0.6858	0.9584	0.8066	0.8036	0.7894	0.7883
MechT5	3.77 M	0.8414	0.9404	0.9474	0.9482	0.7152	0.9637	0.9628	0.9637	0.9637	0.8646
MechLLaMa	2.05 M	0.7895	0.8824	0.8887	0.8892	0.5084	0.8845	0.8830	0.8845	0.8845	0.7484
Reactron	1.48 M	0.8147	0.9198	0.9252	0.9254	0.7993	**0.9814**	**0.9803**	**0.9814**	**0.9814**	**0.8898**
ELECTRO	1.27 M	0.6622	0.7590	0.7653	0.7656	0.3159	**1.0000**	**0.9954**	**1.0000**	**1.0000**	0.6754
Continuous FlowER	1.30 M	**0.8432**	**0.9590**	**0.9682**	**0.9688**	**0.8609**	0.8144	0.8067	0.8144	0.8144	0.8323
Discrete FlowER (§3)	1.26 M	**0.8719**	**0.9732**	**0.9830**	**0.9847**	**0.9491**	**0.9767**	0.9553	**0.9767**	**0.9767**	**0.9408**
Arrow Flow Matching (§5)	1.27 M	**0.8690**	**0.9777**	**0.9851**	**0.9855**	**0.9336**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**0.9491**

Models. We re-implement nine models in one framework and train them on the same data with the same schedule family at comparable parameter counts (Table 12), so the numbers are ours. Four predict a product, the Molecular Transformer (Schwaller et al., 2019), Graph2SMILES (Tu & Coley, 2022), and MechT5 and MechLLaMa of Neukomm et al. (2025). Two predict the mechanism, ELECTRO (Bradshaw et al., 2019) and Reactron (Chen et al., 2025). Three are flow models over the bond–electron matrix, Continuous FlowER (Joung et al., 2025), Discrete FlowER (§3) and Arrow Flow Matching (§5). They share one backbone, one featurisation and every architecture hyper-parameter, so they differ only in the elementary event.

Metrics. *Step accuracy at* k is the fraction of steps whose recorded product is among the k highest-ranked candidates. *Pathway accuracy at* 1 applies the model recursively from a pathway's initial reactants and counts a pathway as correct when some branch reaches its recorded final product. *Validity* is the fraction of candidates satisfying R4, and *electron*, *atom* and *proton conservation* are the fractions keeping the reactant's valence-electron count, heavy-atom multiset and hydrogen count. A chemically invalid candidate fails all of R1–R4 rather than being dropped. The *composite score* is the geometric mean of top-1 step accuracy, top-1 pathway accuracy, validity and the mean of the three conservation checks. Appendix L.1 gives the ceiling that branching in the data sets on top-1 accuracy, the candidate budgets and the counting conventions.

Settings. All models are trained with AdamW for at most 100 epochs, and the checkpoint with the best validation loss is evaluated. Every model is trained at five widths from about 1.2M to 21M parameters. Appendix K and Table 12 give all settings. Every model is decoded by sampling with frequency ranking, the one protocol all nine share, and Appendix L reports the other four regimes.

Results. Appendix L.2 plots Table 2 (Figure 15) and the width sweep (Figure 16). **Q1.** The guarantees hold exactly for Arrow Flow Matching at every one of the five widths. ELECTRO is valid by construction, but it does not always conserve electrons. Continuous FlowER loses conservation in rounding its off-lattice endpoint, and Discrete FlowER in the arg-max over independently predicted entries. **Q2.** The guarantees cost almost no accuracy. Branching in the data caps the top-1 step accuracy of any model at 0.8840. Arrow Flow Matching comes within 0.017 of this ceiling and within 0.004 of the best top-1 step accuracy at every width. At the smallest width it has the best step accuracy at every $k \geq 3$ and the second best pathway accuracy. **Q3.** Its accuracy is flat over the $15\times$ width range, and its composite score at the smallest width beats every other model at every width.

Q4. Arrow Flow Matching computes a sequence of moves and ELECTRO an electron path. The FlowER models compute a chain of matrices, and an update of the chain is a set of arrows when it conserves electrons. Table 3 compares these arrows with those that chemists drew in PMechDB (Tavakoli et al., 2024) and RMechDB (Tavakoli et al., 2023) (Appendix N), and Figure 19 draws two examples. Only Arrow Flow Matching gives arrows on every step, and on every step whose product it predicts correctly its arrows are the chemist's. Its share of the chemist's arrows is its accuracy on these steps. Among 100 sampled mechanisms it draws the chemist's on 98.7% of the radical and 75.9% of the polar steps, whereas ELECTRO draws no radical one. Fine-tuning all four models on the curated steps, under 1% of the training data, raises the chemist's share of Arrow Flow Matching and ELECTRO, while the FlowER chains still rarely give the chemist's arrows.

Table 3: The mechanism of each model read as curly arrows at the smallest width, on 728 test steps whose arrows chemists drew (273 polar, 455 radical). *Account* is the share of steps on which the mechanism gives arrows. *Chemist* is the share on which the arrows of the top-1 candidate equal the chemist's. *First* and *drawn* come from 100 mechanisms sampled per step. *First* is the share on which the chemist's arrows score best among them, and *drawn* the share on which they are among them. A dash marks a model with no probability over arrows, which cannot be scored this way. Each model is given as trained and fine-tuned on the curated training steps (Appendix N). Values are means over three seeds, in %, coloured as in Table 2.

	polar steps								radical steps							
	account		chemist		first		drawn		account		chemist		first		drawn	
fine-tuned	no	yes	no	yes	no	yes	no	yes	no	yes	no	yes	no	yes	no	yes
ELECTRO	62.6	67.2	36.8	40.8	43.8	48.2	57.1	65.2	3.7	2.2	0.0	0.0	0.0	0.0	0.0	0.0
Continuous FlowER	43.0	40.5	0.0	0.0	–	–	–	–	31.3	29.4	16.0	15.4	–	–	–	–
Discrete FlowER	9.8	4.3	0.6	0.6	–	–	–	–	0.4	0.4	0.4	0.4	–	–	–	–
Arrow Flow Matching	100.0	100.0	45.2	47.3	46.5	49.6	75.9	81.0	100.0	100.0	79.3	80.5	78.9	80.4	98.7	99.3

Table 4: The four models of Table 3 under three shifts away from training, at the smallest width. The *move signature* of a step is the multiset of its moves, each written as its type and the elements of its atoms. *Unseen combination* scores a model retrained without a tenth of the signatures, drawn per decade of training frequency, on the test steps of these signatures, whose moves all occur in training. *Ten examples* scores the same model on the same steps after fine-tuning on ten training steps of each held-out signature. 1% *of the data* scores a model trained on 1% of the training split on test steps whose signature occurs in training. The score is the geometric mean of top-1, validity and the mean of the three conservation checks. Coloured as in Table 2.

	unseen combination				ten examples				1% of the data			
model	top-1	valid.	elec.	score	top-1	valid.	elec.	score	top-1	valid.	elec.	score
ELECTRO	0.324	1.000	0.987	0.686	0.319	1.000	0.981	0.682	0.526	1.000	0.990	0.806
Continuous FlowER	0.360	0.670	0.657	0.544	0.394	0.669	0.653	0.559	0.335	0.535	0.512	0.455
Discrete FlowER	0.493	0.936	0.885	0.752	0.554	0.929	0.871	0.777	0.755	0.914	0.828	0.848
Arrow Flow Matching	0.468	1.000	1.000	0.776	0.607	1.000	1.000	0.847	0.729	1.000	1.000	0.900

Q5. Three shifts move the test steps away from the training steps (Table 4, Appendix M). A held-out step combines known moves in a new way, the model sees ten examples of each such combination, or it is trained on 1% of the data. Discrete FlowER and Arrow Flow Matching share one recipe, so they differ only in the head. Every shift lowers the top-1 of every model, and the two discrete heads lose it alike. On an unseen combination, Discrete FlowER reaches 0.493 and Arrow Flow Matching 0.468. Arrow Flow Matching keeps validity and electron conservation at 1 under every shift. ELECTRO keeps validity by construction and electron conservation near 0.99, but it has the lowest top-1 on an unseen combination, 0.324. The two FlowER heads lose validity and conservation where data are scarce or the combination is new. With 1% of the data, the electron conservation of Discrete FlowER falls from 0.953 to 0.828 and that of Continuous FlowER to 0.512 (Table 17). The guarantees cost at most 0.03 in top-1, and the score of Table 4 is highest for Arrow Flow Matching under every shift, with a margin that grows as the data shrink (Figure 17c). With ten examples of each unseen combination, Arrow Flow Matching also learns faster (Figure 17b), 0.607 against 0.554 in top-1.

7 Conclusion and Limitations

A discrete flow-matching model of an elementary step that draws the entries of the bond–electron matrix independently places all its mass on electron-conserving matrices only if it is deterministic in every entry. Changing the elementary event removes this limitation. With one move of a four-element alphabet as the elementary event, Arrow Flow Matching makes conservation an algebraic identity and validity a table lookup on one softmax class, and every update is an arrow that a chemist can read. Several limitations point to future work. The move bridge needs an offline decomposition and ordering of every training step. Several orderings of one multiset of moves can reach the same product, so the score (14) is a lower bound on $\log p_\theta(\widehat{B} \mid B^r)$. Intermediate states need not be valid, as the table gates only the stop class, and non-local validity conditions, such as aromaticity, lie beyond any per-atom table. Three-centre changes would need the alphabet extended, and (9) is all that is required of new increments. Finally, the principle is not specific to chemistry. Whenever a hard constraint fixes a linear functional of a discrete state, an elementary event that leaves the functional unchanged provides the same guarantees, which we leave for future work.

Reproducibility statement

The construction is given in full in §3, §5 and Appendices D to H, together with the derivations that the main text compresses. The network and a summary algorithm are in Appendices I and D.6. The enumeration of $\mathcal{T}$ and the ordering procedure are in Appendices G.3 and H.2, and every proof is given in the appendix that expands the section of its statement. Appendix K lists the data splits, the architecture and optimiser hyper-parameters of all nine models at all five widths, the schedule and the decoding budgets. The dataset is the public mechanistic dataset of Joung et al. (2025). Appendix L reports every evaluation regime rather than the one in the main text.

AI use statement

Generative AI tools were used for coding assistance, editing prose, preparing figures and tables, and running the experiments. They were not used to generate research ideas, and they assisted in drafting and checking the proofs. All AI-assisted output was reviewed by the authors, and we take responsibility for the final content of this work.

References

Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari. Semantic probabilistic layers for neuro-symbolic learning. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2206.00426.

Derek T. Ahneman, Jesús G. Estrada, Shishi Lin, Spencer D. Dreher, and Abigail G. Doyle. Predicting reaction performance in C–N cross-coupling using machine learning. *Science*, 360(6385):186–190, 2018. URL https://doi.org/10.1126/science.aar5169.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.15571.

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. URL https://jmlr.org/papers/v26/23-1605.html.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021. URL https://papers.nips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

John Bradshaw, Matt J. Kusner, Brooks Paige, Marwin H. S. Segler, and José Miguel Hernández-Lobato. A generative model for electron paths. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1805.10970.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2205.14987.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/campbell24a.html.

Ricky T. Q. Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2302.03660.

Shuan Chen and Yousung Jung. Deep retrosynthetic reaction prediction using local reactivity and global attention. *JACS Au*, 1(10):1612–1620, 2021. URL https://doi.org/10.1021/jacsau.1c00246.

Shuan Chen, Kye Sung Park, Taewan Kim, Sunkyu Han, and Yousung Jung. Predicting chemical reaction outcomes based on electron movements using machine learning. *arXiv preprint arXiv:2503.10197*, 2025. URL https://arxiv.org/abs/2503.10197.

Jacob K. Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2402.03559.

Connor W. Coley, Regina Barzilay, Tommi S. Jaakkola, William H. Green, and Klavs F. Jensen. Prediction of organic reaction outcomes using machine learning. *ACS Central Science*, 3(5):434–443, 2017. URL https://doi.org/10.1021/acscentsci.7b00064.

Connor W. Coley, Wengong Jin, Luke Rogers, Timothy F. Jamison, Tommi S. Jaakkola, William H. Green, Regina Barzilay, and Klavs F. Jensen. A graph-convolutional neural network model for the prediction of

chemical reactivity. *Chemical Science*, 10(2):370–377, 2019a. URL https://doi.org/10.1039/C8SC04228D.

Connor W. Coley, Dale A. Thomas, III, Justin A. M. Lummiss, Jonathan N. Jaworski, Christopher P. Breen, Victor Schultz, Travis Hart, Joshua S. Fishman, Luke Rogers, Hanyu Gao, Robert W. Hicklin, Pieter P. Plehiers, Joshua Byington, John S. Piotti, William H. Green, A. John Hart, Timothy F. Jamison, and Klavs F. Jensen. A robotic platform for flow synthesis of organic compounds informed by AI planning. *Science*, 365 (6453):eaax1566, 2019b. URL https://doi.org/10.1126/science.aax1566.

Hanjun Dai, Chengtao Li, Connor Coley, Bo Dai, and Le Song. Retrosynthesis prediction with conditional graph logic network. In *Advances in Neural Information Processing Systems*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html.

Manajit Das, Ajnabiul Hoque, Mayank Baranwal, and Raghavan B. Sunoj. DeepMech: A machine learning framework for chemical reaction mechanism prediction. *Chemical Science*, 17(33):15745–15761, 2026. URL https://doi.org/10.1039/D6SC02809H.

Chenru Duan, Yuanqi Du, Haojun Jia, and Heather J. Kulik. Accurate transition state generation with an object-aware equivariant elementary reaction diffusion model. *Nature Computational Science*, 3(12):1045–1055, 2023. URL https://doi.org/10.1038/s43588-023-00563-7.

Chenru Duan, Guan-Horng Liu, Yuanqi Du, Tianrong Chen, Qiyuan Zhao, Haojun Jia, Carla P. Gomes, Evangelos A. Theodorou, and Heather J. Kulik. Optimal transport for generating transition states in chemical reactions. *Nature Machine Intelligence*, 7(4):615–626, 2025. URL https://doi.org/10.1038/s42256-025-01010-0.

James Dugundji and Ivar Ugi. An algebraic model of constitutional chemistry as a basis for chemical computer programs. In *Computers in Chemistry*, volume 39 of *Fortschritte der Chemischen Forschung*, pp. 19–64. Springer, 1973. URL https://doi.org/10.1007/BFb0051317.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Nic Fishman, Leo Klarner, Valentin De Bortoli, Emile Mathieu, and Michael John Hutchinson. Diffusion models for constrained domains. *Transactions on Machine Learning Research*, 2023. URL https://arxiv.org/abs/2304.05364.

Christoph Flamm, Stefan Müller, and Peter F. Stadler. Every atom-atom map can be explained by electron pushing diagrams. *arXiv preprint arXiv:2311.13492*, 2023. URL https://arxiv.org/abs/2311.13492.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f0d629a734b56a642701bba7bc8bb3ed-Abstract-Conference.html.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. URL https://doi.org/10.1198/016214506000001437.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2006.11239.

Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Annual Meeting of the Association for Computational Linguistics*, 2017. URL https://arxiv.org/abs/1704.07138.

Peter Holderrieth, Marton Havasi, Jason Yim, Neta Shaul, Itai Gat, Tommi Jaakkola, Brian Karrer, Ricky T. Q. Chen, and Yaron Lipman. Generator matching: Generative modeling with arbitrary Markov processes. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.20587.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, 2021. URL https://papers.nips.cc/paper/2021/hash/67d96d458abdef21792e6d8e590244e7-Abstract.html.

Ilia Igashov, Arne Schneuing, Marwin Segler, Michael Bronstein, and Bruno Correia. RetroBridge: Modeling retrosynthesis with Markov bridges. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2308.16212.

Wengong Jin, Connor W. Coley, Regina Barzilay, and Tommi Jaakkola. Predicting organic reaction outcomes with Weisfeiler-Lehman network. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1709.04555.

Joonyoung F. Joung, Mun Hong Fong, Nicholas Casetti, Jordan P. Liles, Ne S. Dassanayake, and Connor W. Coley. Electron flow matching for generative reaction mechanism prediction. *Nature*, 645(8079):115–123,

2025. URL https://doi.org/10.1038/s41586-025-09426-9.

Matthew A. Kayala and Pierre Baldi. ReactionPredictor: Prediction of complex chemical reactions at the mechanistic level using machine learning. *Journal of Chemical Information and Modeling*, 52(10):2526–2540, 2012. URL https://doi.org/10.1021/ci3003039.

Seonghwan Kim, Jeheon Woo, and Woo Youn Kim. Diffusion-based generative AI for exploring transition states from 2D molecular graphs. *Nature Communications*, 15(1):341, 2024. URL https://doi.org/10.1038/s41467-023-44629-6.

Greg Landrum et al. RDKit: Open-source cheminformatics. https://www.rdkit.org, 2024.

Gilbert N. Lewis. The atom and the molecule. *Journal of the American Chemical Society*, 38(4):762–785, 1916. URL https://doi.org/10.1021/ja02261a002.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024. URL https://arxiv.org/abs/2412.06264.

Guan-Horng Liu, Tianrong Chen, Evangelos A. Theodorou, and Molei Tao. Mirror diffusion models for constrained and watermarked generation. In *Advances in Neural Information Processing Systems*, 2023a. URL https://arxiv.org/abs/2310.01236.

Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A. Theodorou, Weili Nie, and Anima Anandkumar. I^2SB: Image-to-image Schrödinger bridge. In *International Conference on Machine Learning*, 2023b. URL https://arxiv.org/abs/2302.05872.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023c. URL https://arxiv.org/abs/2209.03003.

Aaron Lou and Stefano Ermon. Reflected diffusion models. In *International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2304.04740.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2310.16834.

Barbara Mikulak-Klucznik, Patrycja Gołębiowska, Alison A. Bayly, Oskar Popik, Tomasz Klucznik, Sara Szymkuć, Ewa P. Gajewska, Piotr Dittwald, Olga Staszewska-Krajewska, Wiktor Beker, Tomasz Badowski, Karl A. Scheidt, Karol Molga, Jacek Mlynarski, Milan Mrksich, and Bartosz A. Grzybowski. Computational planning of the synthesis of complex natural products. *Nature*, 588(7836):83–88, 2020. URL https://doi.org/10.1038/s41586-020-2855-y.

Ryan J. Miller, Alexander E. Dashuta, Brayden Rudisill, David Van Vranken, and Pierre Baldi. Mechanism-aware deep learning for polar reaction prediction. *Journal of the American Chemical Society*, 147(44):41168–41176, 2025. URL https://doi.org/10.1021/jacs.5c16838.

Juno Nam and Jurae Kim. Linking the neural machine translation and the prediction of organic chemistry reactions. *arXiv preprint arXiv:1612.09529*, 2016. URL https://arxiv.org/abs/1612.09529.

Théo A. Neukomm, Zlatko Jončev, and Philippe Schwaller. Teaching language models mechanistic explainability through MechSMILES. *arXiv preprint arXiv:2512.05722*, 2025. URL https://arxiv.org/abs/2512.05722.

Xuan-Vu Nguyen, Octavian Susanu, Daniel Armstrong, and Philippe Schwaller. Mechanistic reaction prediction via discrete flow matching on graph-structured electron occupation. *arXiv preprint arXiv:2608.27429*, 2026. URL https://arxiv.org/abs/2608.27429.

Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018. URL https://arxiv.org/abs/1804.06609.

Kaipai Ren, Kun Tang, Yujing Zhao, Lei Zhang, Jian Du, Qingwei Meng, and Qilei Liu. Reactive machine learning potential for accelerating transition state search in organic synthesis. *Nature Communications*, 17(1):6253, 2026. URL https://doi.org/10.1038/s41467-026-72945-0.

Victor Sabanza Gil, Andres M. Bran, Malte Franke, Rémi Schlama, Jeremy S. Luterbacher, and Philippe Schwaller. Holistic chemical evaluation reveals pitfalls in reaction prediction models. *arXiv preprint arXiv:2312.09004*, 2023. URL https://arxiv.org/abs/2312.09004.

Mikołaj Sacha, Mikołaj Błaż, Piotr Byrski, Paweł Dąbrowski-Tumański, Mikołaj Chromiński, Rafał Loska, Paweł Włodarczyk-Pruszyński, and Stanisław Jastrzębski. Molecule edit graph attention network: Modeling chemical reactions as sequences of graph edits. *Journal of Chemical Information and Modeling*, 61(7):3273–3284, 2021. URL https://doi.org/10.1021/acs.jcim.1c00537.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In

Advances in Neural Information Processing Systems, 2024. URL https://arxiv.org/abs/2406.07524.

Philippe Schwaller, Théophile Gaudin, Dávid Lányi, Costas Bekas, and Teodoro Laino. "Found in Translation": Predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chemical Science*, 9(28):6091–6098, 2018. URL https://doi.org/10.1039/C8SC02339E.

Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. Molecular Transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS Central Science*, 5(9):1572–1583, 2019. URL https://doi.org/10.1021/acscentsci.9b00576.

Marwin H. S. Segler, Mike Preuss, and Mark P. Waller. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature*, 555(7698):604–610, 2018. URL https://doi.org/10.1038/nature25978.

Neta Shaul, Itai Gat, Marton Havasi, Daniel Severo, Anuroop Sriram, Peter Holderrieth, Brian Karrer, Yaron Lipman, and Ricky T. Q. Chen. Flow matching with general discrete paths: A kinetic-optimal perspective. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2412.03487.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2406.04329.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion Schrödinger bridge matching. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.16852.

Vignesh Ram Somnath, Charlotte Bunne, Connor W. Coley, Andreas Krause, and Regina Barzilay. Learning graph models for retrosynthesis prediction. In *Advances in Neural Information Processing Systems*, 2021. URL https://arxiv.org/abs/2006.07038.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2011.13456.

Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to DNA sequence design. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.05841.

Mohammadamin Tavakoli, Yin Ting T. Chiu, Pierre Baldi, Ann Marie Carlton, and David Van Vranken. RMechDB: A public database of elementary radical reaction steps. *Journal of Chemical Information and Modeling*, 63(4):1114–1123, 2023. URL https://doi.org/10.1021/acs.jcim.2c01359.

Mohammadamin Tavakoli, Ryan J. Miller, Mirana Claire Angel, Michael A. Pfeiffer, Eugene S. Gutman, Aaron D. Mood, David Van Vranken, and Pierre Baldi. PMechDB: A public database of elementary polar reaction steps. *Journal of Chemical Information and Modeling*, 64(6):1975–1983, 2024. URL https://doi.org/10.1021/acs.jcim.3c01810.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2302.00482.

Zhengkai Tu and Connor 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. URL https://doi.org/10.1021/acs.jcim.2c00321.

Ping Tuo, Jiale Chen, and Ju Li. Flow matching for reaction pathway generation. *Nature Communications*, 17 (1):8769, 2026. URL https://doi.org/10.1038/s41467-026-75654-w.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.14734.

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

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023. URL https://arxiv.org/abs/2307.09702.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International Conference on Machine Learning*, 2018. URL https://proceedings.mlr.press/v80/xu18h.html.

Li-Cheng Xu, Miao-Jiong Tang, Junyi An, Fenglei Cao, and Yuan Qi. A unified pre-trained deep learning framework for cross-task reaction performance prediction and synthesis planning. *Nature Machine Intelligence*, 7(9):1561–1571, 2025. URL https://doi.org/10.1038/s42256-025-01098-4.

Robin Yadav, Qi Yan, Guy Wolf, Avishek Joey Bose, and Renjie Liao. Retro SynFlow: Discrete flow-matching for accurate and diverse single-step retrosynthesis. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.04439.

Appendix contents

A Related work **17**

B Notation **18**

C The encoding **20**

C.1 The task . . . 20

C.2 Three descriptions of one molecule . . . 20

D Discrete FlowER **21**

D.1 The state space and the path . . . 21

D.2 The generator, the marginal rate and the minimiser . . . 22

D.3 The parameterisation and the loss . . . 23

D.4 Relation to the evidence bound . . . 23

D.5 Sampling and decoding . . . 25

D.6 The algorithm . . . 25

E The limitation of the entrywise head **25**

E.1 Proof of Theorem 1 . . . 25

E.2 Conservation on the training path . . . 27

E.3 Repair at the end . . . 27

E.4 The two ends of the dichotomy . . . 28

F The move alphabet **28**

F.1 The arrow identity . . . 28

F.2 The bond migration . . . 29

F.3 Single-electron moves . . . 29

F.4 Proof of Proposition 2 . . . 30

G Admissibility and the table **31**

G.1 Admissible moves at a real reactant . . . 31

G.2 The closed form of the table . . . 32

G.3 Measurement of the capacities . . . 32

G.4 What the enumeration returns . . . 34

G.5 The jam . . . 34

G.6 The place where the table is checked . . . 35

G.7 Proof of Proposition 3 . . . 35

H The move bridge **36**

H.1 Decomposition . . . 36

H.2 Ordering . . . 36
H.3 The choice of discrete time . . . 37

I The shared network and the two heads **37**
I.1 The backbone . . . 37
I.2 The entrywise output layer . . . 37
I.3 The move head . . . 38
I.4 The mask on a GPU . . . 38

J Decoding **39**
J.1 The move decoder . . . 39
J.2 Decoders and ranking rules of every model . . . 40

K Experimental setup **40**
K.1 Data . . . 40
K.2 The evaluation pipeline . . . 41
K.3 The scope of the guarantee . . . 41
K.4 Training . . . 41
K.5 Decoding budgets . . . 41

L Further results **42**
L.1 The evaluation protocol . . . 42
L.2 The main comparison and the width sweep . . . 43
L.3 The other four regimes at the smallest width . . . 44
L.4 The width sweep under the other regimes . . . 44

M Generalisation **46**
M.1 The move-signature split . . . 46
M.2 Training and evaluation . . . 46
M.3 Results . . . 47

N Interpretability **48**

A Related work

This appendix places the paper between data-driven reaction prediction and generative modelling on discrete spaces.

Mechanistic reaction prediction. Data-driven reaction prediction divides according to what the model emits. Sequence models emit a product string, and the analogy with machine translation was proposed by Nam & Kim (2016) and Schwaller et al. (2018) and refined into the Molecular Transformer (Schwaller et al., 2019), with graph encoders replacing the reactant string later (Tu & Coley, 2022). Graph-edit models emit a change to the molecular graph, either by ranking candidate products generated at a predicted reaction centre (Jin et al., 2017; Coley et al., 2017; 2019a) or by generating the edits themselves as a sequence (Sacha et al., 2021). The same decomposition drives retrosynthesis models that predict a graph edit and then complete the fragments (Somnath et al., 2021; Chen & Jung, 2021), which together with search (Segler et al., 2018) is what a synthesis planner consumes. None of these represents electrons, and Sabanza Gil et al. (2023) show how easily aggregate accuracy hides the resulting failures. Mechanistic models instead emit the elementary steps. Kayala & Baldi (2012) rank polar, pericyclic and radical elementary reactions between approximate molecular orbitals. Miller et al. (2025) predict polar elementary steps from the PMechDB data, and their ArrowFinder model recovers the arrow-pushing mechanism that connects given reactants and products. ELECTRO (Bradshaw et al., 2019) generates a reaction as a sequence of electron-path steps and is the closest antecedent of our move alphabet, though its paths are linear and alternate removal with addition. Reactron (Chen et al., 2025) predicts a product by generating the arrow-pushing diagram of each mechanistic step, and Neukomm et al. (2025) teach a language model an annotated SMILES encoding of electron flow. That arrow-pushing is expressive enough to be a complete formalism, rather than a convenient notation, is a theorem. Every atom-to-atom map decomposes into cyclic electron-pushing diagrams (Flamm et al., 2023). Joung et al. (2025) assembled the mechanistic dataset on which all of these can now be measured.

Flow matching in mechanistic reaction prediction. Continuous FlowER (Joung et al., 2025) is the direct ancestor of this paper, transporting the reactant bond–electron matrix to the product matrix by continuous flow matching, with a zero-sum training target and zero-sum noise, so that the drift conserves electrons exactly. The guarantee covers neither the rounding of the endpoint, whose consequences §1 lists, nor the N local per-atom conditions. Elsewhere in chemistry the same machinery appears with a different state space. Tuo et al. (2026) use conditional flow matching with deterministic optimal transport for transition states and reaction products in three dimensions. Duan et al. (2025) pose transition-state generation as an optimal-transport problem between reactant and product geometries, having previously solved it by an object-aware diffusion model (Duan et al., 2023), and Kim et al. (2024) generate transition-state geometries from 2D graphs by diffusion. On the retrosynthesis side, RetroBridge (Igashov et al., 2024) and RetroSynFlow (Yadav et al., 2025) make the same choice of coupling we do, pinning the path at two data points rather than at a prior. They work over molecular graphs and synthons rather than over an electron budget, and neither carries a conservation law. The present work differs in that the constraint is enforced by the *support of the elementary event* rather than by the geometry of the state space or the target of a regression. MAELLE (Nguyen et al., 2026) is the closest work in method. It also applies discrete flow matching to an integer state, builds its path from electron moves that an integer optimal-transport problem finds, and needs no annotated elementary steps. Its trajectories recover electron flows that agree with known mechanisms, which is the interpretability that §1 asks for. It differs from Arrow Flow Matching in five respects. First, its state is indexed by the heavy atoms only, and the hydrogens of each heavy atom are pooled into one count instead of being mapped atoms, and protonation and deprotonation are simplified. Second, its lone-pair entries count pairs rather than electrons, so it has no single-electron moves and no radical steps. Third, besides moves that transfer a pair from one entry to another, its elementary events include adding and deleting a pair, so its electron count is not fixed by construction. Fourth, no per-atom table restricts where its chain may stop, so validity is not guaranteed. Fifth, one update applies several elementary events, one per entry that fires, whereas one update of Arrow Flow Matching applies exactly one move. At the time of writing, no official implementation of MAELLE had been released, so it is not among the models of §6.

Flow matching. Continuous flow matching regresses a velocity field onto the time derivative of a conditional interpolation between a source and a target sample, so that a single simulation-free regression yields a probability path between two distributions (Lipman et al., 2023; Liu et al., 2023c;

Albergo & Vanden-Eijnden, 2023). The three constructions were introduced independently and were shown to be instances of one stochastic-interpolant framework (Albergo et al., 2025) which also contains score-based diffusion (Ho et al., 2020; Song et al., 2021) as a special case. The practical variants differ mainly in how the source and target are coupled, minibatch optimal transport straightening the paths and reducing the number of solver steps (Tong et al., 2024), and the recipe carries to Riemannian manifolds (Chen & Lipman, 2024) and to large-scale image synthesis (Esser et al., 2024). A recent tutorial (Lipman et al., 2024) collects the design choices. On a discrete space there is no velocity, and the field's role is taken by a rate matrix. Categorical diffusion supplied the first such models, multinomial diffusion (Hoogeboom et al., 2021) and the discrete-state denoising processes of Austin et al. (2021), whose absorbing-MASK and uniform corruptions are still the two default priors, and Campbell et al. (2022) put them in continuous time as Markov chains. Discrete flow matching proper (Campbell et al., 2024; Gat et al., 2024) then carries the conditional-to-marginal argument over. A bridge between two points induces a conditional rate, and the rate generating the marginal path is its posterior mean, which is (19). Subsequent work widens the design space with arbitrary discrete probability paths under a kinetic-optimality criterion (Shaul et al., 2025), score-entropy parameterisations (Lou et al., 2024), the simplified masked objectives of Sahoo et al. (2024) and Shi et al. (2024), and continuous relaxations onto the simplex (Stark et al., 2024). Generator matching (Holderrieth et al., 2025) then shows all of them, jump processes included, to be instances of one construction over an arbitrary Markov generator. Graph-structured discrete diffusion (Vignac et al., 2023) is the closest of these to our state space. Our Discrete FlowER (§3 and Appendix D) is this recipe instantiated on bond–electron matrices, written out in full precisely because §5 is the same recipe with one substitution. The novelty there is not a path or a loss but the *elementary event*, and Theorem 1 is the reason to change it.

Constraints in generative models. The usual devices are a penalty, a rejection filter, or a projection applied to the output. Diffusion and flow models have been adapted to constrained domains by changing the noising process, using reflected stochastic differential equations (Lou & Ermon, 2023), barrier and reflected metrics on manifolds (Fishman et al., 2023) or mirror maps into a dual space (Liu et al., 2023a), or by making sampling itself a constrained optimisation (Christopher et al., 2024). These handle convex sets and smooth manifolds well, but a level set of a linear functional intersected with a lattice is neither, and none makes the *intermediate* states of the process admissible. Neuro-symbolic layers instead compile a logical constraint into the output distribution, either as a differentiable penalty (Xu et al., 2018) or as a tractable circuit that renormalises over the satisfying assignments (Ahmed et al., 2022), which is exact at a cost that grows with the circuit size of the constraint. The device used here is the least expensive member of that family, a *masked support*, in which violating classes are removed before the softmax is normalised, as in lexically constrained and grammar-constrained decoding (Hokamp & Liu, 2017; Post & Vilar, 2018; Willard & Louf, 2023). It applies here because of the elementary event rather than the mask. That elementary event leaves $\mathcal{E}$ unchanged by construction, and the mask costs $O(1)$ per move and is provably non-empty (Remark 8).

B NOTATION

Every symbol is defined once, at its first use, and the symbols used in more than one place are collected here, in five groups. The last column justifies each symbol that serves more than one role. Calligraphic letters denote sets, with the electron count $\mathcal{E}$, the loss $\mathcal{L}$ and the uniform law $\mathcal{U}$ as the exceptions.

Atoms, entries and sets. Sums and products over the entries of a symmetric matrix run over $i \leq j$, so each off-diagonal entry is counted once.

symbol	meaning	remark
N	number of mapped atoms	
$i, j, l; i', j'$	atoms; a second pair of atoms, or any atom other than i, j	
k	moves applied so far (11)	also the list cut-off of the accuracy metrics, scopes that never meet
$a = (a_i)_i$	the element vector, over the elements of the data	

symbol	meaning	remark
x_{ij}; $\mathcal{V}_{ij}$	the entry (i, j), $i \le j$; its values, $\{0, \dots, 14\}$ if $i = j$ and $\{0, \dots, 3\}$ otherwise	
$\mathcal{X}$; μ	the state space (3); a product measure on it	
$\mathcal{T}$; (a, d, b); $\bar{b}$	per-atom table; a local state of element, lone-pair electrons and bonding pairs; $\bar{b} = 12$	
$\mathcal{S}_n$; $\mathcal{M}$	conservation sheet, physical sector (5), $n = \mathcal{E}(B^r)$	
$\mathcal{A}(x)$; $\mathcal{Y}(x)$	admissible moves at x (10); class set of the move head, $\mathcal{A}(x)$ with $\varnothing$ when $x \in \mathcal{M}$	

Matrices and per-atom quantities. A per-atom quantity carries the atom as a subscript, and a per-element quantity carries the element.

symbol	meaning	remark
B^r, B^p, $\widehat{B}$; Δ	reactant, product and candidate matrices; $\Delta = B^p - B^r$	
x, x'; ν, ν'	states; values of one entry	
β_i; c_i; q_i	bonding pairs (1); total pairs (1); formal charge (15)	
G_a; C_a; n_i^{H}	group valence; per-element capacity (31); implicit hydrogens	
E_{ij}	matrix of one pair at entry (i, j) (27)	
$\mathcal{E}$	electron count (2)	
$\mathbf{1}_{ij}$; $\mathbb{1}[\cdot]$	symmetric unit matrix at entry (i, j); indicator	

Moves. A move changes three entries, and δ with a superscript names these increments.

symbol	meaning	remark
τ; $y = (\tau, i, j)$; $\varnothing$	move type; a move, one class of the head; the stop class	
δ_τ^{ii}, δ_τ^{jj}, δ_τ^{ij}; A_y	increments of a type; the matrix a move adds (9)	
δf	change of f under one move or arrow	lower-case as $\delta x_{ii} = \delta_\tau^{ii}$, capital Δ kept for a step
δ_x, δ_ν	point mass at x or ν	
m	number of moves of a step	

Processes. A subscript t or k on a state counts the time of the entrywise bridge, x_t, or the moves of the move bridge, x_k, and the entries of such a state are written $(x_t)_{ij}$. A parenthesised superscript counts Euler updates $x^{(n)}$ or network layers $z^{(l)}$.

symbol	meaning	remark
π	empirical law of steps (B^r, B^p)	
t; x_t	time; state of the entrywise bridge, from B^r at $t = 0$ to B^p at $t = 1$	
p_t; ξ_{ij}; α, ω	path (6); its Bernoulli switch; B_{ij}^r, B_{ij}^p	
u_t, $\bar{u}_t$, u_t^θ; U_t	conditional, marginal and model rates; matrix of $\bar{u}_t$	
λ_t; ρ_n	rate $1/(1-t)$; resampling probability $1/(S-n)$	$\rho_n \in [0, 1]$, unlike a rate
$p_{1\mid t}$; v_t	posterior endpoint marginal; continuous velocity	
$\mathbb{Q}$, $\mathbb{P}_\theta$	path laws of App. D.4	
S; t_n	updates per candidate; n/S	Euler updates, or the cap on the number of moves
x_k; y_k; $\hat{m}$	the chain after k moves; class taken at x_k, $y_m = \varnothing$; sampled chain length	ordered move in (11), target in (13), drawn in (14)

Model. Both models share the names of the logits, the loss and the score.

symbol	meaning	remark
$p_\theta(\cdot \mid B^r)$	law of the emitted candidate	
$\mathrm{softmax}_{\mathcal{Y}}(\ell)$	distribution on a class set $\mathcal{Y}$ with mass proportional to $\exp \ell_y$	
$p_\theta(\cdot \mid x, t)$, p_θ^{ij}	entrywise head (7) and its factors	real second argument
$p_\theta(\cdot \mid x, k)$	move head (12)	integer second argument
ℓ_θ; ℓ_{ij}; $\mathcal{L}$; $s(\widehat{B})$	logits; logit vector of entry (i, j); loss; score of a candidate	one name per role in both models
$z_i^{(l)}$, z_i; R_{ij}; h	hidden state, last one; pair representation; attention head	
$d_{\mathrm{model}}, d_{\mathrm{head}}, d_{\mathrm{ff}}$	widths	
$\beta_1, \beta_2, \varepsilon$	AdamW constants	conventional, unrelated to β_i
$N_{\mathrm{batch}}, N_{\mathrm{beam}}$; σ	batch size, beam width; noise of Continuous FlowER	

C THE ENCODING

This appendix expands §2.

C.1 THE TASK

A reaction mechanism is an ordered sequence of elementary steps. One elementary step is a pair of molecule sets, a reactant side and a product side, together with an *atom map*, which is a bijection between the atoms of the two sides. As this map is a bijection, both sides carry one indexing $1, \ldots, N$, and no atom is created or destroyed. The nuclei stay put, and what changes is where the valence electrons sit. Given B^r, we require a conditional distribution $p_\theta(\cdot \mid B^r)$ from which candidates can be drawn and scored. A further requirement is the subject of this paper. A candidate is useful only if it is a *valid molecule*. The emitted matrix must be reconstructible into a chemical structure, and it must carry the same number of valence electrons as the reactant. A model that is accurate but occasionally emits a five-valent carbon has not solved this task but a relaxation of it. Of the conditions R1–R4 of §2, R1 and R2 are automatic, the atom map fixing the vertex set, and R3 is a condition on the matrix alone, as follows. Let G_a be the group valence of element a, i.e., its outer-shell electron count as a neutral free atom, so that $G_{\mathrm{C}} = 4$, $G_{\mathrm{O}} = 6$ and $G_{\mathrm{H}} = 1$. The formal charge of atom i is

$$q_i(x) = G_{a_i} - x_{ii} - \beta_i(x), \tag{15}$$

where the subtracted terms are the electrons assigned to atom i. Summing over i and using $x_{ij} = x_{ji}$ gives $\sum_i (G_{a_i} - q_i(x)) = \sum_i x_{ii} + 2\sum_{i<j} x_{ij} = \mathcal{E}(x)$, so the electron count of a candidate is fixed by its matrix alone.

C.2 THREE DESCRIPTIONS OF ONE MOLECULE

Figure 3 carries one reactant of the dataset through all three descriptions used in this paper at once, the string that the file stores, the structure that a chemist draws, and the matrix that the model is handed. They carry the same information, and the atom map is what makes passing between them a bijection rather than a matching problem. The off-diagonal entries are integers, as aromatic rings are resolved into alternating single and double bonds before the matrix is built. Panels (d) and (e) of the figure read the per-atom quantities of §2 off this matrix.

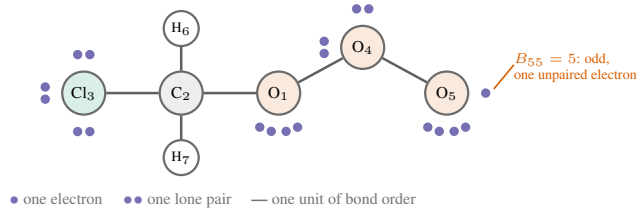
Remark 4 (The dataset makes every hydrogen its own atom)**.** On this dataset, all but a handful of hydrogens are written explicitly and carry an atom map (Appendix K.1), so each has a row of the matrix like any other atom. On data that leaves n_i^{H} hydrogens on atom i implicit, n_i^{H} is added to β_i in (1) and hence subtracted in (15). Then $\sum_i (G_{a_i} - q_i(x))$ becomes $\mathcal{E}(x) + \sum_i n_i^{\mathrm{H}}$, whose last term is fixed by the atom map.

(a) text, what the dataset stores

```
[O:1]([C:2]([Cl:3])([H:6])[H:7])[O:4][O:5]
```

Each `:i` is the atom map. It names the row and column the atom owns in (c), and is the *same* name on the product side, so no atom is created or destroyed.

(b) graph, the Lewis structure a chemist draws



(c) matrix, what the model sees, B^r

	O_1	C_2	Cl_3	O_4	O_5	H_6	H_7
O_1	4	1	0	1	0	0	0
C_2	1	0	1	0	0	1	1
Cl_3	0	1	6	0	0	0	0
O_4	1	0	0	4	1	0	0
O_5	0	0	0	1	5	0	0
H_6	0	1	0	0	0	0	0
H_7	0	1	0	0	0	0	0

diagonal: lone-pair *electrons*, counted once
off-diagonal: bond order, shared *pairs*, counted twice

(d) what (1), (2) and (15) read off that matrix

$\beta_{O_1} = B_{12} + B_{14} = 1 + 1 = 2$ and $\beta_{O_5} = B_{45} = 1$, the bonding pairs of two oxygens.

$q_i = G_{a_i} - B_{ii} - \sum_{j \neq i} B_{ij} = 0$ at every atom, so the molecule is neutral.

$\mathcal{E}(B^r) = \underbrace{19}_{\text{diagonal}} + 2 \cdot \underbrace{6}_{\text{bond orders}} = 31$, which agrees with the valence electrons of the free atoms, $3 \cdot 6 + 4 + 7 + 2 \cdot 1 = 31$ ✓.

(e) the total pair count c_i at each atom

O_1 $c_{O_1} = \frac{1}{2} \cdot 4 + 2 = 4$
C_2 $c_{C_2} = \frac{1}{2} \cdot 0 + 4 = 4$
Cl_3 $c_{Cl_3} = \frac{1}{2} \cdot 6 + 1 = 4$
O_4 $c_{O_4} = \frac{1}{2} \cdot 4 + 2 = 4$
O_5 $c_{O_5} = \frac{1}{2} \cdot 5 + 1 = 3\frac{1}{2}$
H_6 $c_{H_6} = \frac{1}{2} \cdot 0 + 1 = 1$
H_7 $c_{H_7} = \frac{1}{2} \cdot 0 + 1 = 1$

Figure 3: One molecule of the mechanism dataset, the alkyl trioxide radical $ClCH_2{-}O{-}O{-}O^{\bullet}$, written in the three ways used in this paper in (a) to (c), with the per-atom quantities read off its matrix in (d) and (e). The units on and off the diagonal deliberately differ. The shaded diagonal counts *electrons* and everything off it counts *pairs*, which is why $\mathcal{E}$ of (2) counts them once and twice. Panels (c) to (e) share one colour code, purple for the diagonal and pink for the bonds. In (e), one slot is one pair, coloured by the entry that holds it, and the half slot at O_5 is its unpaired electron. The octet rule is therefore not a statement about the matrix as a whole but a limit on the total pair count c_i of (1) at each atom, which involves only the entries that touch it. Every entry is a small non-negative integer, which is the observation on which §3 turns.

D Discrete FlowER

This appendix derives Discrete FlowER, which §3 summarises.

D.1 The state space and the path

State space and coupling. We keep the integrality that the data possess. The state space $\mathcal{X}$ of (3) is a finite set of size $4^{N(N-1)/2} \cdot 15^N$. Finiteness is what makes categorical distributions, rate matrices and exact likelihoods available (Austin et al., 2021; Campbell et al., 2022). The coupling is the one place where this construction differs from generative modelling as it is usually presented, as *the source is the conditioning object*. Generic discrete constructions have to choose a prior, because the target is unconditional and there is nothing else to start from. The usual choices are an absorbing MASK state (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024), whose reverse process is a masked language model, or a uniform draw over $\mathcal{X}$ (Hoogeboom et al., 2021; Campbell et al., 2022; Lou et al., 2024). Here the reactant is already a point of $\mathcal{X}$, and so is the product. The path is therefore a *Markov bridge* pinned at two data points, as in RetroBridge (Igashov et al., 2024) and RetroSynFlow (Yadav et al., 2025). The continuous setting does the same in data-to-data bridge constructions (Liu et al., 2023b; Shi et al., 2023). No prior appears anywhere below.

The choice of a jump process. Maximum likelihood needs a normalised density, and the decisive observation concerns the *data*. By (3), a normalised distribution over a *single entry of* B is a softmax over at most fifteen classes, which is writable, differentiable, exactly normalised and cheap to evaluate. The *joint* distribution over all $O(N^2)$ entries is the object that is unavailable. A product of per-entry softmaxes normalises trivially, but it asserts something false, as a bond order cannot be chosen independently of the lone-pair electron counts on the two atoms it touches, as (1) ties them together. Anything richer needs a normalising constant that sums over a set exponentially large in N,

or the entry-by-entry factorisation of §4. The solution is the one that generative modelling always uses (Ho et al., 2020; Song et al., 2021; Lipman et al., 2023). The joint distribution is not *written* but *defined* as the law of the output of a procedure. If a process $(x_t)_{t\in[0,1]}$ starts at something we already hold and ends distributed as B^p, then sampling means simulating it, and no normalising constant is formed. The model learns the *local* behaviour of the process, and a network can fit such an object. Two choices remain. *What it runs on.* We keep the lattice $\mathcal{X}$, so the state is an integer matrix at all times. No state lies between two adjacent points of $\mathcal{X}$, so such a process must sit at a point, wait, and *jump*. It is therefore a continuous-time Markov chain, and its local behaviour is a matrix of jump *rates* rather than a velocity (Campbell et al., 2022; Gat et al., 2024). *What path it takes.* The simplest choice is to move probability mass rather than a point, so that each entry holds its reactant value for a while and, at a random time, switches once and for all to its product value (Campbell et al., 2024; Gat et al., 2024).

The conditional path. In the path (6), t is *the probability that a given entry has already switched*, and δ_x below denotes the point mass at x. The path is the discrete counterpart of the straight-line interpolation of continuous flow matching (Lipman et al., 2023; Liu et al., 2023c; Albergo & Vanden-Eijnden, 2023). In a vector space one moves a *point* along a segment, whereas here one moves *probability mass* between the two atoms of a two-point measure, independently at each entry (Campbell et al., 2024; Gat et al., 2024). The path is sampled by

$$\xi_{ij} \overset{\text{iid}}{\sim} \text{Bernoulli}(t), \qquad (x_t)_{ij} = (1-\xi_{ij})B^r_{ij} + \xi_{ij}B^p_{ij} \;\; (i \le j), \qquad (x_t)_{ji} := (x_t)_{ij}, \tag{16}$$

where the draw is made on the upper triangle and mirrored, so symmetry is a property of the representation and never has to be projected. Substituting $t = 0$ and $t = 1$ gives $p_0 = \delta_{B^r}$ and $p_1 = \delta_{B^p}$, so both endpoints are hit *exactly*, and there is no residual noise scale to trade off. Every entry always holds a value in $\mathcal{V}_{ij}$, so $x_t \in \mathcal{X}$ for every t and every draw, and an intermediate state is an integer matrix like any other (Figure 5a).

D.2 The generator, the marginal rate and the minimiser

The path, the head and the loss all factorise over entries, so we work at one entry (i, j) and return to the matrix only where needed. The entry takes values $\nu \in \mathcal{V}_{ij}$. A distribution on $\mathcal{V}_{ij}$ is a vector indexed by ν, and δ_ν is the point mass at ν. We write $\alpha := B^r_{ij}$ and $\omega := B^p_{ij}$, so that the bridge (6) at this entry is $p_t = (1-t)\delta_\alpha + t\delta_\omega$.

Rates. A continuous-time Markov chain on a finite set is specified by its rates. We write $u_t(\cdot, \nu)$ for the vector of rates at which the chain leaves the current value ν for each other value. It must satisfy

$$\underbrace{u_t(\nu', \nu) \ge 0 \;(\nu' \ne \nu),}_{\text{non-negative}} \qquad \underbrace{\textstyle\sum_{\nu'} u_t(\nu', \nu) = 0,}_{\text{entries sum to zero}} \qquad \underbrace{\partial_t p_t = \textstyle\sum_\nu u_t(\cdot, \nu)\, p_t(\nu),}_{\text{Kolmogorov forward equation}} \tag{17}$$

which is the standard object of continuous-time discrete generative modelling (Campbell et al., 2022; 2024; Gat et al., 2024; Holderrieth et al., 2025). The second condition makes $-u_t(\nu, \nu)$ the total rate of leaving ν, and the third is the discrete continuity equation. A rate vector whose only target is ω is therefore a positive multiple of $\delta_\omega - \delta_\nu$, the point mass where the chain goes minus the point mass where it stands. On the matrix, a chain that changes one entry per jump moves from x to a state x' that differs from x at (i, j) only at the rate of that entry, and it never moves to a state that differs from x at two entries.

Proposition 5 (The generator of Discrete FlowER)**.** *Let* $\lambda_t = 1/(1-t)$*, and let* $p_{1|t}(\nu \mid x) := \Pr[B^p_{ij} = \nu \mid x_t = x]$ *be the* posterior endpoint marginal *of entry* (i, j)*.*

(i) The conditional rate

$$u_t\big(\cdot, \nu \mid B^r, B^p\big) = \lambda_t\big(\delta_\omega - \delta_\nu\big) \tag{18}$$

satisfies (17) *and generates the bridge* (6)*.*

(ii) The marginal rate

$$\bar{u}_t(x', x) := \mathbb{E}\big[u_t(x', x \mid B^r, B^p) \,\big|\, x_t = x\big] \tag{19}$$

generates the marginal path $p_t(x) = \sum_{B^r, B^p} \pi(B^r, B^p)\, p_t(x \mid B^r, B^p)$*, and at entry* (i, j) *it reads*

$$\bar{u}_t(\cdot, x) = \lambda_t\big(p_{1|t}(\cdot \mid x) - \delta_{x_{ij}}\big). \tag{20}$$

(iii) For fixed t and x, the expected cross-entropy $-\mathbb{E}[\log p(B^p_{ij}) \mid x_t = x]$ over distributions p on $\mathcal{V}_{ij}$ is minimised uniquely at $p = p_{1|t}(\cdot \mid x)$.

Proof. *(i)* For $\nu' \neq \nu$ the rate is $\lambda_t \delta_\omega(\nu') \geq 0$, and the entries of the vector sum to $\lambda_t(1-1) = 0$. The rate is linear in the point masses, and p_t sums to one, so

$$\sum_\nu u_t(\cdot, \nu)\, p_t(\nu) = \lambda_t\big(\delta_\omega - p_t\big) = \lambda_t(1-t)\big(\delta_\omega - \delta_\alpha\big) = \delta_\omega - \delta_\alpha = \partial_t p_t.$$

(ii) The marginal path is a mixture of conditional paths. We differentiate under the finite sum, apply (17) to each conditional path, and use Bayes' rule $\pi(B^r, B^p)\, p_t(x \mid B^r, B^p) = p_t(x)\, \pi(B^r, B^p \mid x_t = x)$. This gives

$$\partial_t p_t(x') = \sum_{B^r, B^p} \pi(B^r, B^p) \sum_x u_t(x', x \mid B^r, B^p)\, p_t(x \mid B^r, B^p) = \sum_x p_t(x)\, \bar{u}_t(x', x),$$

so $\bar{u}_t$ generates the marginal path. At entry (i, j) the conditional rate $\lambda_t(\delta_{B^p_{ij}} - \delta_{x_{ij}})$ is linear in $\delta_{B^p_{ij}}$, and the conditional mean of the point mass at a random value is the law of that value, $p_{1|t}(\cdot \mid x)$. This is (20). *(iii)* The expected cross-entropy is $-\sum_\nu p_{1|t}(\nu \mid x) \log p(\nu) = H(p_{1|t}) + \mathrm{KL}(p_{1|t} \,\|\, p)$. The entropy is free of p, and by Gibbs' inequality the divergence is non-negative and zero only at $p = p_{1|t}$. □

Part (ii) is the discrete instance of the conditional-to-marginal argument of flow matching (Lipman et al., 2023; Campbell et al., 2024; Gat et al., 2024), which generator matching (Holderrieth et al., 2025) shows to hold for an arbitrary Markov generator. Set beside the endpoint form of a continuous flow-matching velocity (Lipman et al., 2023; 2024), $v_t(x) = \lambda_t(\mathbb{E}[B^p \mid x_t{=}x] - x)$, the rate (20) has the same shape. A difference of *points* has become a difference of *distributions*, with the current value represented by the point mass $\delta_{x_{ij}}$. The reformulation is a substitution rather than a redesign, which is why a backbone network designed for one transfers unchanged to the other.

D.3 The parameterisation and the loss

By (20), knowing $p_{1|t}$ is the same as knowing the generator. The network therefore predicts the *endpoint* of every entry, and it is trained by maximum likelihood for that endpoint at states drawn from the bridge, by a plain cross-entropy, as in the clean-data prediction parameterisations of discrete diffusion (Austin et al., 2021; Campbell et al., 2024; Gat et al., 2024). This is the head (7) and the loss (8). In words, the product is corrupted back towards the reactant along the bridge, and the network is asked to name the product value of every entry. Nothing is simulated or estimated. Inside the expectation over t and x_t, the loss at entry (i, j) is the expected cross-entropy of Proposition 5(iii). Its minimiser is therefore $p^{ij}_{\theta^\star} = p_{1|t}$, and by (20) the model recovers the exact marginal generator.

Remark 6 (Why cross-entropy)**.** Cross-entropy is a *strictly proper scoring rule* (Gneiting & Raftery, 2007), so its optimum is the entire predictive distribution rather than a summary of it. We take a reactant that admits two products, with $\pi(B^p \mid B^r) = \frac{1}{2}\delta_{B^{p,\mathrm{I}}} + \frac{1}{2}\delta_{B^{p,\mathrm{II}}}$. At $t = 0$ the state carries no information about the branch, so at an entry where the two products differ the optimum of (8) is $p^{ij}_\theta = \frac{1}{2}\delta_{B^{p,\mathrm{I}}_{ij}} + \frac{1}{2}\delta_{B^{p,\mathrm{II}}_{ij}}$. A draw from it is one of the two product values. A squared error would instead fit their mean, which is neither of them and need not be an integer. Figure 4 draws the two optima.

D.4 Relation to the evidence bound

The cross-entropy (8) was written down directly, as a fit of the endpoint. It also serves the likelihood of the product, because it has the same minimiser as an upper bound on $-\log \mathbb{P}_\theta(B^p)$.

The bound. Let $\mathbb{Q}$ and $\mathbb{P}_\theta$ be the path laws of two continuous-time Markov chains on $\mathcal{X}$ with the same initial law and with rate matrices $u^{\mathbb{Q}}_t$ and u^θ_t. The blackboard letters keep them apart from the heads. Their divergence is (Campbell et al., 2022)

$$\mathrm{KL}\left(\mathbb{Q} \,\|\, \mathbb{P}_\theta\right) = \mathbb{E}_{\mathbb{Q}}\left[\int_0^1 \sum_{x' \neq x_t} \left(u^\theta_t(x', x_t) - u^{\mathbb{Q}}_t(x', x_t) + u^{\mathbb{Q}}_t(x', x_t) \log \frac{u^{\mathbb{Q}}_t(x', x_t)}{u^\theta_t(x', x_t)}\right) dt\right]. \quad (21)$$

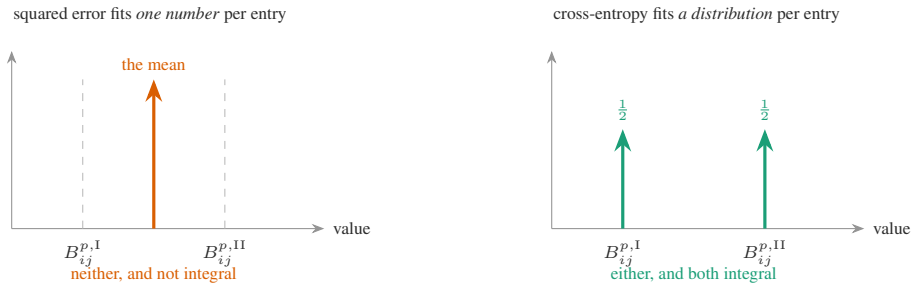


Figure 4: Why the objective matters, at one matrix entry of a reactant that admits two products. A squared-error optimum is the posterior mean and lands between the modes, whereas a cross-entropy optimum is the posterior itself and keeps them apart. Only the right-hand object can be sampled to give either product, and only the right-hand object is supported on integers.

The integrand compares the two chains at the current state x_t. Its first two terms compare their total rates of leaving x_t, and the logarithm compares where they jump. We take $\mathbb{Q}$ to be the bridge (18), whose final law is the point mass at B^p. The divergence between the final laws is then $-\log \mathbb{P}_\theta(B^p)$, and it cannot exceed the divergence between the path laws. Keeping the terms of (21) that depend on θ gives

$$-\log \mathbb{P}_\theta(B^p) \;\le\; \mathbb{E}_{\mathbb{Q}}\left[\int_0^1 \sum_{x' \neq x_t} \Big(u_t^\theta(x', x_t) - u_t^{\mathbb{Q}}(x', x_t) \log u_t^\theta(x', x_t)\Big) dt\right] + \text{const.} \tag{22}$$

The bound for Discrete FlowER. Two facts evaluate the sums over x'. First, the model changes one entry per jump. By (20), with p_θ^{ij} in place of $p_{1|t}$, it moves entry (i, j) to a value $\nu \neq x_{ij}$ at rate $\lambda_t p_\theta^{ij}(\nu \mid x, t)$. Each p_θ^{ij} normalises, so the total rate of leaving is

$$\sum_{x' \neq x_t} u_t^\theta(x', x_t) = \lambda_t \sum_{i \le j} \Big(1 - p_\theta^{ij}\big((x_t)_{ij} \mid x_t, t\big)\Big).$$

Second, the bridge makes one kind of jump only. It moves entry (i, j) to B_{ij}^p at rate λ_t, and only while that entry has not yet switched. Its sum therefore keeps one term per unswitched entry,

$$\sum_{x' \neq x_t} u_t^{\mathbb{Q}}(x', x_t) \log u_t^\theta(x', x_t) = \lambda_t \sum_{i \le j} \mathbb{1}\big[B_{ij}^p \neq (x_t)_{ij}\big] \Big(\log \lambda_t + \log p_\theta^{ij}\big(B_{ij}^p \mid x_t, t\big)\Big).$$

Under $\mathbb{Q}$ the state x_t is drawn from the bridge (6). Substituting both sums into (22) and dropping $\log \lambda_t$, which is free of θ, gives

$$-\log \mathbb{P}_\theta(B^p) \le \int_0^1 \lambda_t \, \mathbb{E}_{x_t}\Bigg[\sum_{i \le j} \Big(\underbrace{1 - p_\theta^{ij}\big((x_t)_{ij} \mid x_t, t\big)}_{\text{(i) rate of leaving } x_t} \underbrace{- \mathbb{1}\big[B_{ij}^p \neq (x_t)_{ij}\big] \log p_\theta^{ij}\big(B_{ij}^p \mid x_t, t\big)}_{\text{(ii) cross-entropy, unswitched}}\Big)\Bigg] dt + \text{const.}$$

The minimiser of the bound. Both terms carry θ, and neither may be dropped. Term (i) is the total rate at which the model leaves x_t, and it is the only term that rewards mass *at* the current value. Term (ii) is a cross-entropy against the endpoint, taken only at the entries that have not yet switched. We fix t, x_t and one entry (i, j), and we write $q := p_{1|t}(\cdot \mid x_t)$ for the posterior marginal there, p for the head and x_{ij} for the current value. We average the bracket over the posterior and substitute $1 - p(x_{ij}) = \sum_{\nu \neq x_{ij}} p(\nu)$. The bracket at this entry becomes

$$\sum_{\nu \neq x_{ij}} \Big(p(\nu) - q(\nu) \log p(\nu)\Big).$$

Each summand depends on one value $p(\nu)$ only and is convex in it, with derivative $1 - q(\nu)/p(\nu)$. The summands are therefore minimised separately, at $p(\nu) = q(\nu)$, which is feasible as

$\sum_{\nu \neq x_{ij}} q(\nu) \leq 1$. Normalisation then gives $p(x_{ij}) = 1 - \sum_{\nu \neq x_{ij}} q(\nu) = q(x_{ij})$. The bound is therefore minimised where (8) is, at $p_\theta^{ij} = p_{1|t}$ (Proposition 5). Term (i) is what makes this so. Term (ii) alone would be minimised by a head that puts *no* mass on the current value, which is never its target.

The bound and the implemented loss. The loss (8) reaches the same minimiser by another route. It weights all times equally rather than by λ_t, and it scores *every* entry rather than only the unswitched ones, so that keeping the current value is itself a target wherever $B_{ij}^p = (x_t)_{ij}$. The two objectives agree on what they fit and differ only in the variance and conditioning of the gradient estimator.

D.5 Sampling and decoding

We write U_t for the marginal rate matrix of (19), the array whose (x', x) entry is $\bar{u}_t(x', x)$. Over an update of length $1/S$ with U_t held fixed, the exact transition kernel is the matrix exponential $\exp(U_t/S) = I + U_t/S + O(S^{-2})$, and truncating it gives the Euler scheme, which is the discrete-space analogue of an ODE solver update and a first-order case of the tau-leaping used for continuous-time discrete models (Campbell et al., 2022; Gat et al., 2024). Entrywise, with S updates on $t_n = n/S$, the model rate (20) with p_θ^{ij} in place of $p_{1|t}$ gives the law of the next value of entry (i, j),

$$\begin{aligned} \mathrm{Law}\big(x_{ij}^{(n+1)}\big) &= \delta_{x_{ij}^{(n)}} + \tfrac{1}{S}\lambda_{t_n}\Big(p_\theta^{ij}\big(\cdot \mid x^{(n)}, t_n\big) - \delta_{x_{ij}^{(n)}}\Big) \\ &= \big(1 - \rho_n\big)\,\delta_{x_{ij}^{(n)}} + \rho_n\, p_\theta^{ij}\big(\cdot \mid x^{(n)}, t_n\big), \qquad \rho_n := \tfrac{1}{S}\lambda_{t_n} = \tfrac{1}{S-n}, \end{aligned} \tag{23}$$

a mixture of two distributions with weights that sum to one. The resampling probability ρ_n gets its own letter because, unlike the rate λ_t, it lies in $[0, 1]$. *Read as an algorithm, this resamples entry (i, j) from the endpoint distribution with probability ρ_n and leaves it untouched otherwise.* Since $\rho_0 = 1/S$ rises to $\rho_{S-1} = 1$, the early updates are conservative and the last one would resample every entry at once, the divergence of λ_t as $t \to 1$ being absorbed by the discretisation. We replace that last draw by its mode, so the chain is run for $S - 1$ updates and the emitted candidate is $\widehat{B}_{ij} = \arg\max_\nu p_\theta^{ij}(\nu \mid x^{(S-1)}, t_{S-1})$, whose log-probability under that head is read off at a *single* sample,

$$s\big(\widehat{B}\big) = \sum_{i \leq j} \log p_\theta^{ij}\big(\widehat{B}_{ij} \mid x^{(S-1)}, t_{S-1}\big) = \log p_\theta\big(\widehat{B} \mid x^{(S-1)}, t_{S-1}\big). \tag{24}$$

The score (24) covers the last update alone. It is neither $\log p_\theta(\widehat{B} \mid B^r)$, which would marginalise over the Euler paths that reach $x^{(S-1)}$, nor the probability of the emitted candidate under the arg-max rule, which is one. It is nevertheless the only score that a one-shot head provides, and we rank by it. Decoding is then the identity map, as $\widehat{B}$ is already an integer matrix. Continuous FlowER (Joung et al., 2025) instead ends its trajectory off the lattice and must round it, with the consequences listed in §1. Figure 5 contrasts the paths of the three flow models on one step.

D.6 The algorithm

Equations (3), (6), (7), (8) and (23), with the backbone and the entrywise output layer of Appendix I, are the whole of **Discrete FlowER**. Algorithm 1 collects its training and sampling in one place.

E The limitation of the entrywise head

This appendix expands §4.

E.1 Proof of Theorem 1

Theorem 1 (A product measure cannot live on the conservation sheet)**.** *Let $\mu = \bigotimes_{i \leq j} \mu_{ij}$ be a product measure on $\mathcal{X}$ with $\operatorname{supp}\mu \subseteq \mathcal{S}_n$ for some fixed n. Then every μ_{ij} is a point mass.*

Proof. The proof is by contradiction, and it has three stages.

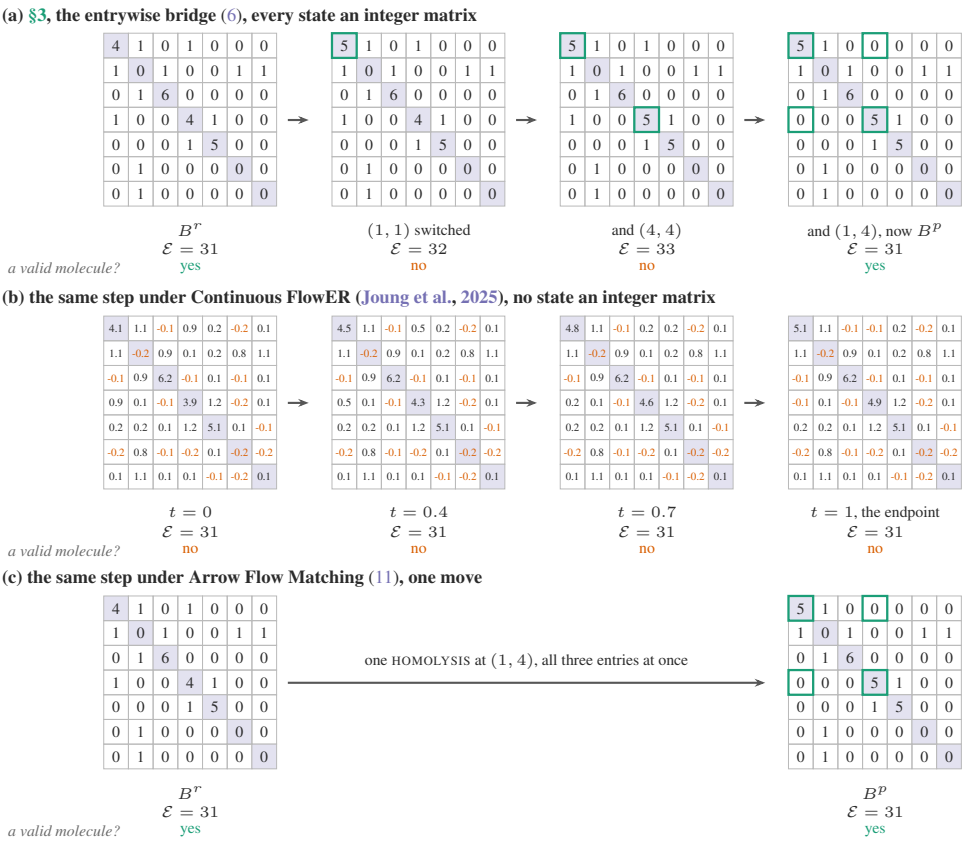


Figure 5: Three ways to move a bond–electron matrix from B^r to B^p, shown on the homolysis of the peroxide bond of Figure 3. Every row draws the same seven-by-seven matrix in the same columns. The diagonal cells are tinted, and a negative value is set in orange. **(a)** The entrywise bridge of Discrete FlowER. The three entries that the step changes switch one at a time, and the outlined cells are those that have already switched. The order drawn is one of six, the eight states that such orders reach are the corners of the box of Figure 1(b), and the middle two states drawn carry 32 and 33 electrons. **(b)** The path of Continuous FlowER, drawn schematically, with only the entries that the step changes moving. $\mathcal{E}$ is 31 at every t and every matrix is symmetric, but every entry is fractional and the endpoint is off the lattice. **(c)** The move bridge of Arrow Flow Matching. The same three increments form one HOMOLYSIS, so this step has no intermediate state and $\mathcal{E}$ stays at 31.

Stage 1, the support is a box. A configuration x has positive probability under μ exactly when each x_{ij} has positive probability under its own factor μ_{ij}, because the factors are independent. Hence

$$\operatorname{supp} \mu = \prod_{i \le j} \operatorname{supp} \mu_{ij}.$$

The right-hand side is a product of sets, one per entry, so the support is a box. Its defining feature is that the entries may be varied one at a time without leaving it.

Stage 2, two support points one entry apart. Suppose, for contradiction, that some entry (i, j) is not deterministic, so that $\operatorname{supp} \mu_{ij}$ contains two distinct values $\nu \neq \nu'$. We pick any $x \in \operatorname{supp} \mu$ with $x_{ij} = \nu$, which is possible because every other entry may take any value in its own support. Let x' agree with x everywhere except at (i, j), where $x'_{ij} = \nu'$. By Stage 1, the point x' also lies in $\operatorname{supp} \mu$. The box therefore contains two points that differ at (i, j) and nowhere else.

Stage 3, the two points have different electron counts. $\mathcal{E}$ is linear, so the difference of its values at two points depends only on where they differ,

$$\mathcal{E}(x') - \mathcal{E}(x) = \begin{cases} \nu' - \nu, & i = j, \\ 2(\nu' - \nu), & i < j, \end{cases}$$

by (2). Neither value is zero, as $\nu' \neq \nu$, so $\mathcal{E}(x') \neq \mathcal{E}(x)$. At most one of the two can equal n, so at least one of them lies outside $\mathcal{S}_n$. This contradicts $\operatorname{supp} \mu \subseteq \mathcal{S}_n$.

Algorithm 1 Discrete FlowER, training and sampling.

Training (one update).

draw a step $(B^r, B^p) \sim \pi$ and a time $t \sim \mathcal{U}[0,1]$

draw $\xi_{ij} \sim \text{Bernoulli}(t)$ independently for $i \le j$ and set $(x_t)_{ij} = (1-\xi_{ij})B^r_{ij} + \xi_{ij}B^p_{ij}$, mirrored to the lower triangle ▷ (16)

form the codes $\gamma, \tilde{\gamma}$ from (B^r, x_t), run the backbone, read off $p^{ij}_\theta(\cdot \mid x_t, t)$ ▷ (7)

take one optimiser update on $-\sum_{i\le j} \log p^{ij}_\theta\big(B^p_{ij} \mid x_t, t\big)$ ▷ (8)

Sampling (one candidate, $S-1$ draws and one arg-max).

$x^{(0)} \leftarrow B^r$

for $n = 0, \dots, S-2$ **do**

  $t_n \leftarrow n/S, \quad \rho_n \leftarrow 1/(S-n)$

  evaluate $p^{ij}_\theta(\cdot \mid x^{(n)}, t_n)$ for every $i \le j$

  for $i \le j$ **do**

    with probability ρ_n resample $x^{(n+1)}_{ij} \sim p^{ij}_\theta(\cdot \mid x^{(n)}, t_n)$, else $x^{(n+1)}_{ij} \leftarrow x^{(n)}_{ij}$ ▷ (23)

  end for

  mirror the upper triangle onto the lower one

end for

$\widehat{B}_{ij} \leftarrow \arg\max_\nu p^{ij}_\theta(\nu \mid x^{(S-1)}, t_{S-1})$ and $s(\widehat{B}) \leftarrow \sum_{i\le j} \log p^{ij}_\theta(\widehat{B}_{ij} \mid x^{(S-1)}, t_{S-1})$ ▷ (24)

return $\widehat{B}$, which is an integer matrix and needs no decoding

No entry can therefore carry two values, so every μ_{ij} is a point mass. □

E.2 Conservation on the training path

Discrete FlowER couples no two entries, neither in its path (6), nor in its head (7), nor in its sampler (23). On the path, this makes the electron count correct in expectation but not with probability one. From (16), $(x_t)_{ij} = B^r_{ij} + \xi_{ij}\Delta_{ij}$ with $\xi_{ij} \sim \text{Bernoulli}(t)$ independent, so by linearity of $\mathcal{E}$ and (2) we have $\mathcal{E}(x_t) - \mathcal{E}(B^r) = \sum_i \xi_{ii}\Delta_{ii} + 2\sum_{i<j} \xi_{ij}\Delta_{ij}$ and

$$\mathbb{E}\big[\mathcal{E}(x_t)\big] - \mathcal{E}(B^r) = t\Big(\sum_i \Delta_{ii} + 2\sum_{i<j}\Delta_{ij}\Big) = t\mathcal{E}(\Delta) = 0, \quad \text{linearity of expectation, then (2)} \tag{25}$$

$$\operatorname{Var}\big[\mathcal{E}(x_t)\big] = t(1-t)\Big(\sum_i \Delta^2_{ii} + 4\sum_{i<j}\Delta^2_{ij}\Big) > 0. \quad \text{independence, and } \operatorname{Var}[\xi_{ij}] = t(1-t)\text{; positive for } 0 < t < 1 \text{ and } \Delta \neq 0 \tag{26}$$

Only the *first moment* is correct. A six-entry step, with three bonds formed and three broken, has $|\Delta_{ij}| = 1$ at each of six off-diagonal entries, so that the bracket of (26) is $4 \cdot 6 = 24$ and the standard deviation at $t = \frac{1}{2}$ is $\sqrt{6} \approx 2.4$ electrons. Such a state conserves exactly when as many formed as broken entries have switched, which, averaged over t, happens with probability $\sum_\kappa \binom{3}{\kappa}^2 (2\kappa)!\,(6-2\kappa)!/7! = \frac{2304}{5040}$. About half of the states on which the network is trained therefore carry the wrong electron count. Figure 5(a) draws one trajectory of this kind on a three-entry step.

E.3 Repair at the end

Imposing the constraint only at decoding, by filtering or projecting the output, delivers a valid candidate at two costs. Filtering censors a distribution that was never shaped by the constraint, so the model spends parameters on candidates that it will never be allowed to emit, and the retained mass is not the conditional that the training objective fitted. Projection maps an inadmissible candidate onto an admissible one by a rule that the network never saw, so the emitted law is the model pushed through an untrained map, which is neither the fitted conditional nor a calibrated approximation to it. Neither device touches the intermediate states, which still carry the wrong electron count (Figure 5a). §5 removes both costs by moving the constraint into the support of the elementary event, so that it holds at every move.

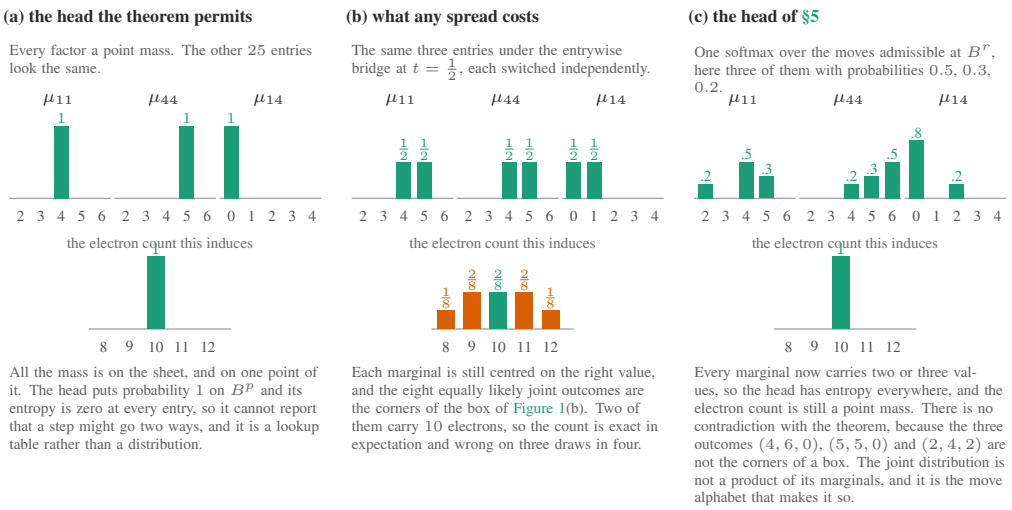


Figure 6: The two ends of the dichotomy that Theorem 1 forces, and the third possibility that changing the elementary event opens, on the three entries that the step of Figure 5 touches. The diagonal alphabets $\{0, \dots, 14\}$ are drawn truncated, and all three electron-count axes are on one scale. **(a)** The conclusion of the theorem drawn, where exact conservation is available to a factorised head at the price of the spread of every factor. **(b)** The entrywise bridge of Discrete FlowER at $t = \frac{1}{2}$. **(c)** The move head (12), whose support lies on $\mathcal{E} = 10$ while its marginals are spread out.

E.4 The two ends of the dichotomy

Theorem 1 is a dichotomy rather than a bound, so we draw both of its ends on the same axes. Figure 6 does this on the three entries that the step of Figure 5 touches. Their values are $x_{11} = 4$, $x_{44} = 4$ and $x_{14} = 1$ at the reactant and 5, 5 and 0 at the product, so the sheet within these three entries is $x_{11} + x_{44} + 2x_{14} = 10$. Panel (a) is the head that the theorem permits, and panel (b) is what any spread costs. Panel (c) is the head that §5 builds, which is neither of the two, as it is not a product measure at all.

F The move alphabet

This appendix expands the move alphabet of §5.

F.1 The arrow identity

The diagonal of the matrix counts electrons and everything off it counts pairs (Figure 3), so one pair changes a diagonal entry by two units and an off-diagonal entry by one. Let $\mathbf{1}_{ij}$ be the symmetric unit matrix at entry (i, j), a 1 there and zeros elsewhere, which off the diagonal means a 1 at (i, j) *and* at (j, i), as these are one entry and not two. By (2), $\mathcal{E}(\mathbf{1}_{ii}) = 1$ and $\mathcal{E}(\mathbf{1}_{ij}) = 2$ for $i < j$. The matrix E_{ij} of one pair placed at entry (i, j) is therefore

$$E_{ii} := 2\,\mathbf{1}_{ii}, \qquad E_{ij} := \mathbf{1}_{ij} \;\; (i < j), \qquad \text{so that} \qquad \mathcal{E}(E_{ij}) = 2 \;\; \text{for all } i \le j, \tag{27}$$

which says that one pair is worth two electrons *wherever it sits*. An arrow transfers one pair from a source entry (i, j) to a sink entry (i', j'),

$$A = E_{i'j'} - E_{ij} \qquad \Longrightarrow \qquad \mathcal{E}(A) = 2 - 2 = 0. \tag{28}$$

The implication follows from the linearity of $\mathcal{E}$ and from (27) at both entries. Conservation is therefore not a constraint imposed on the arrow. *It is an identity forced by the arrow having a tail and a head that carry the same quantum*, which is the mathematical content of the drawing convention of chemists. Nothing about which entries these are, which molecule it is, or how large N is enters the argument. The first two move types of §5 are arrows, namely LONE→BOND from (i, i) to (i, j) and BOND→LONE from (i, j) to (j, j), so for them (9) is (28) written out.

Figure 7 writes two steps of the reactant of Figure 3 out in this form. The first is an arrow, the heterolysis of the $C_2{-}Cl_3$ bond, so it is (28) with the bond entry as the source and the diagonal

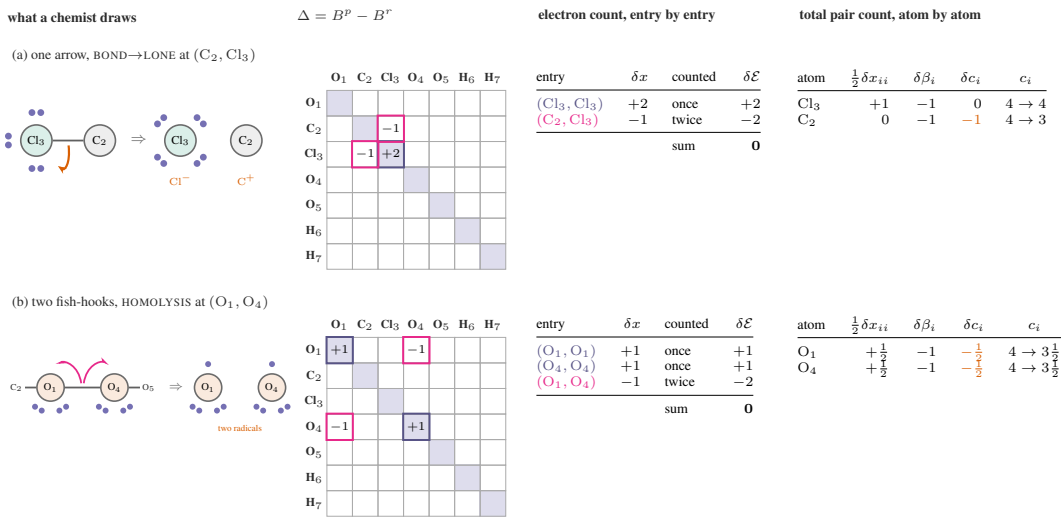


Figure 7: Two steps of the reactant of Figure 3, written out in the same four columns. (a) The heterolysis of the $C_2{-}Cl_3$ bond, one arrow that moves a pair. (b) The homolysis of the $O_1{-}O_4$ bond, two fish-hooks that move one electron each. From left to right, what a chemist draws, the matrix Δ, its electron count entry by entry, and the total pair count (1) atom by atom. Purple marks diagonal entries and pink marks bond entries, as in Figure 3, and Δ is zero at every entry left blank. The electron count sums to zero in both steps, and by (27) it would for any pair of entries. The total pair count does not. It falls at C_2 in (a) and at both oxygens in (b), and whether the new values are allowed is decided by the table $\mathcal{T}$, not by the algebra.

entry of chlorine as the sink. The second is a homolysis, which is not an arrow, as it sends one electron to each end of the broken bond. In both steps the entries that change carry different units. A bond entry counts pairs and a diagonal entry counts electrons, so the increments differ, and the electron count cancels because a pair is worth two electrons at every entry. The total pair counts c_i of (1) do not cancel. They pass from atom to atom, and keeping them within the capacities is the task of the table $\mathcal{T}$ of §5, not of the algebra. The pattern of the three entries that change is what §5 turns into the move alphabet.

F.2 The bond migration

Remark 7 (The bond migration is a composition)**.** The third arrow type of chemists is the *bond migration*, in which a pair moves from the bond (i, j) to an adjacent bond (j, l) that shares one atom with it. It is not in the alphabet, as it is named by an atom *triple* and would force the network to score a ragged set of candidates. It is also unnecessary, being the composition of two moves that are already present,

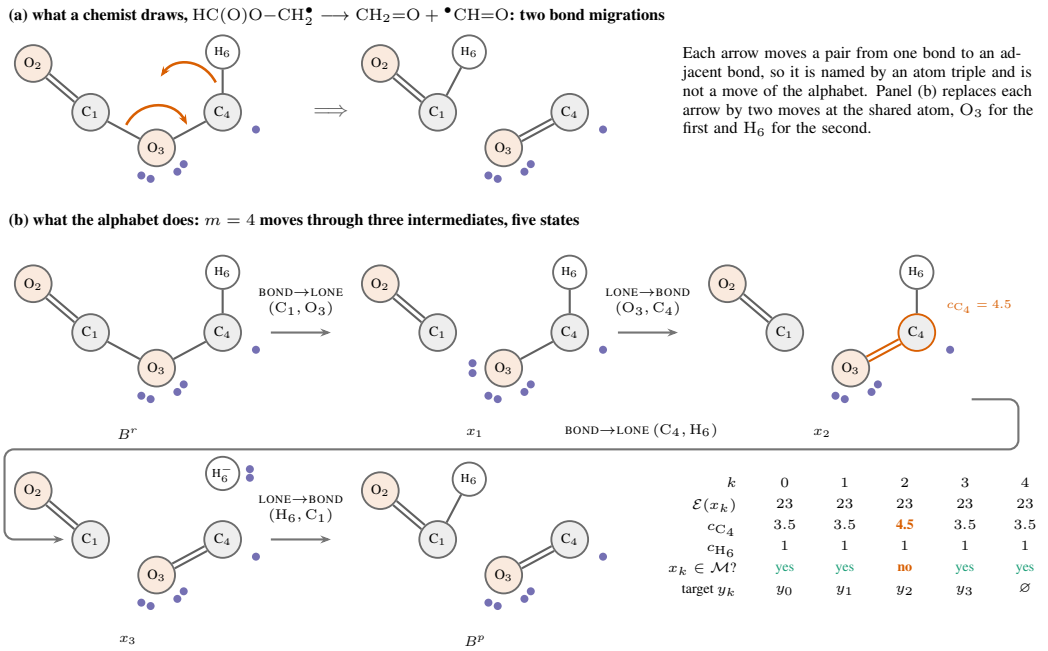
$$\underbrace{(i,j) \to (j,j)}_{\text{BOND}\to\text{LONE}} \text{ then } \underbrace{(j,j) \to (j,l)}_{\text{LONE}\to\text{BOND}} \quad\Longrightarrow\quad \delta x_{ij} = -1,\ \ \delta x_{jj} = 2 - 2 = 0,\ \ \delta x_{jl} = +1,$$

which is the migration. The price is that the intermediate state, in which the pair sits on j as a lone pair before moving on, has to be a state that the chain is allowed to occupy, and the cost in path length is one extra move per migration. The expansion also provides something that the alphabet was not designed for. Appendix G.5 shows that splitting the migration in two is exactly what unlocks the concerted steps, which as single arrows cannot be started at all.

Figure 8 carries a real test step through both readings. The step contains two migrations, and the figure shows the intermediate state that each expansion adds.

F.3 Single-electron moves

Pair coordinates would simplify the alphabet, halving every diagonal entry so that every arrow becomes a unit increment. This is legitimate only when every diagonal entry is even, and on this dataset it is not. Odd diagonal entries occur on hypervalent phosphorus and on organometallic centres, and some steps change a diagonal entry by an odd amount. These steps are not miscellaneous.



Training draws $k \sim \mathcal{U}\{0, \dots, 4\}$, forms x_k, and scores the class in the bottom row of the table under the masked softmax (12). Four of the five targets are moves and the fifth is $\varnothing$. In x_1 the pair of the broken $C_1{-}O_3$ bond waits on O_3 as a lone pair. Every target is admissible at its own state, so the loss is finite at every draw. Every state on the path carries $\mathcal{E} = 23$ electrons, by Proposition 2. In x_2 the carbon C_4 holds four and a half pairs for one move, so $\mathcal{T}$ offers no $\varnothing$ there and the chain moves on. x_3 carries the hydride, a lone pair on a hydrogen, with $c_{H_6} = 1$, which the table accepts. The spectator hydrogens H_5 on C_1 and H_7 on C_4, and the two lone pairs of O_2, are not drawn.

Figure 8: The bridge, on a multi-move step of the test split. (a) A chemist draws the step as two bond migrations. The pair of the $C_1{-}O_3$ bond moves to the $O_3{-}C_4$ bond, and the pair of the $C_4{-}H_6$ bond moves to a new $H_6{-}C_1$ bond. (b) Expanded by Remark 7 and ordered greedily (Appendix H.2), each migration becomes BOND⟶LONE followed by LONE⟶BOND at the shared atom. Each expansion adds one intermediate, x_1 with a lone pair on O_3 and x_3 with a hydride, a lone pair on H_6. The whole H_6 atom thereby crosses from C_4 to C_1.

Most of them are a single palladium–phosphorus bond breaking with one electron going to each atom, or the same bond forming from two such electrons. Most of the rest are the same two changes on an organic bond, such as O−O, N−O or C−Br. These are the last two rows of Table 1. A small remainder, 443 steps or 0.2% of the test split, combines an odd change with further changes in the same step. The decomposition writes such a step with several moves. It first reads the step by half-arrows alone, one HOMOLYSIS for each unit of bond order lost and one COLLIGATION for each unit gained. This covers hydrogen abstractions, radical additions and several metal–ligand bonds changing at once. A step that also moves an electron pair has no such reading. For it, the decomposition splits off the smallest set of at most four single-electron moves on the changed bonds that leaves every diagonal change even, and factorises the rest into pair moves. Their conservation is the same identity (9), which Figure 7(b) checks for HOMOLYSIS, and COLLIGATION is its reverse.

Remark 8 (The alphabet is closed under inversion)**.** Reversing LONE⟶BOND at (i, j) gives BOND⟶LONE at (j, i), and HOMOLYSIS and COLLIGATION are the reverse of each other. Every move can therefore be undone by another move of the alphabet, and that undo is admissible at the state which the move produced, as it merely restores a state that was admissible a moment ago. At B^r itself, which no move produced, the stop class is available, as $B^r \in \mathcal{M}$. Hence *no reachable state has an empty admissible set*, and the class set $\mathcal{Y}(x)$ of the masked softmax (12) is never empty, so the mask is safe to apply with no fallback branch.

F.4 Proof of Proposition 2

Proposition 2 (Conservation)**.** *Let $(x_k)_{k \geq 0}$ be any chain every update of which is a move of the alphabet. Then $\mathcal{E}(x_k) = \mathcal{E}(x_0)$ surely.*

Proof. By induction on k. The base case $k = 0$ is the identity $\mathcal{E}(x_0) = \mathcal{E}(x_0)$.

For the inductive step, we assume $\mathcal{E}(x_k) = \mathcal{E}(x_0)$. The next state is $x_{k+1} = x_k + A_y$ for some move $y = (\tau, i, j)$, where A_y is the symmetric matrix of (9). Since $\mathcal{E}$ is linear,

$$\mathcal{E}\big(x_{k+1}\big) = \mathcal{E}\big(x_k\big) + \mathcal{E}(A_y).$$

The matrix A_y is non-zero only at (i,i), (j,j) and the two copies of (i,j), so that $\mathcal{E}(A_y) = \delta_\tau^{ii} + \delta_\tau^{jj} + 2\delta_\tau^{ij}$, where the factor 2 counts the two copies. This sum is zero for each of the four types, which is (9). Hence $\mathcal{E}(x_{k+1}) = \mathcal{E}(x_k) = \mathcal{E}(x_0)$. The argument uses neither the probabilities of the moves nor the set $\mathcal{A}$, so conservation holds along every path and does not need admissibility. □

G ADMISSIBILITY AND THE TABLE

This appendix expands the admissible set and the table $\mathcal{T}$ of §5.

G.1 ADMISSIBLE MOVES AT A REAL REACTANT

Everything in (10) follows from one counting fact. We write $\delta f := f(x + A_y) - f(x)$ for the change of a quantity f under one move, or under one arrow in Figure 11, where the lower-case δ matches the increments δ_τ^{ii} and leaves Δ for a whole step. A move $y = (\tau, i, j)$ changes exactly three entries, and hence exactly five of the scalars that appear anywhere in (5),

$$\delta x_{ii} = \delta_\tau^{ii}, \quad \delta x_{jj} = \delta_\tau^{jj}, \quad \delta x_{ij} = \delta_\tau^{ij}, \quad \delta\beta_i = \delta\beta_j = \delta_\tau^{ij}, \quad \delta\beta_{i'} = 0 \ \ (i' \neq i, j). \tag{29}$$

The β increments are the row sums of (1), the only off-diagonal entry that the move touches in row i is x_{ij}, and the only one in row j is $x_{ji} = x_{ij}$. Every other atom of the molecule is untouched, whatever N is. This locality is what turns the three clauses of (10) into five scalar tests.

From three clauses to five tests. The first clause, $x + A_y \in \mathcal{X}$, asks every entry of $x + A_y$ to lie in its range $\mathcal{V}_{ij}$ of (3). The entries that the move does not touch are those of x, which already lies in $\mathcal{X}$, so only the first three lines of (29) need a test. By the fourth line, the second and third clauses are $\beta_i(x) + \delta_\tau^{ij} \le \bar{b}$ and $\beta_j(x) + \delta_\tau^{ij} \le \bar{b}$. They need no lower bound, as a bonding-pair count is a sum of entries that the first clause keeps non-negative. By the last line, no other atom needs a test. Written out,

$$y \in \mathcal{A}(x) \iff \underbrace{\begin{aligned} 0 &\le x_{ii} + \delta_\tau^{ii} \le 14, \\ 0 &\le x_{jj} + \delta_\tau^{jj} \le 14, \\ 0 &\le x_{ij} + \delta_\tau^{ij} \le 3, \end{aligned}}_{x + A_y \in \mathcal{X}} \qquad \underbrace{\beta_i(x) + \delta_\tau^{ij} \le \bar{b},}_{\beta_i(x + A_y) \le \bar{b}} \qquad \underbrace{\beta_j(x) + \delta_\tau^{ij} \le \bar{b}.}_{\beta_j(x + A_y) \le \bar{b}} \tag{30}$$

Each test reads one of the five scalars of (29), whatever the move type and N.

The tests at a real reactant. Everything that (30) needs is read off two per-atom vectors and one entry of the matrix. For the reactant B^r of Figure 3, the diagonal x_{ii} and the row sums β_i of (1) are

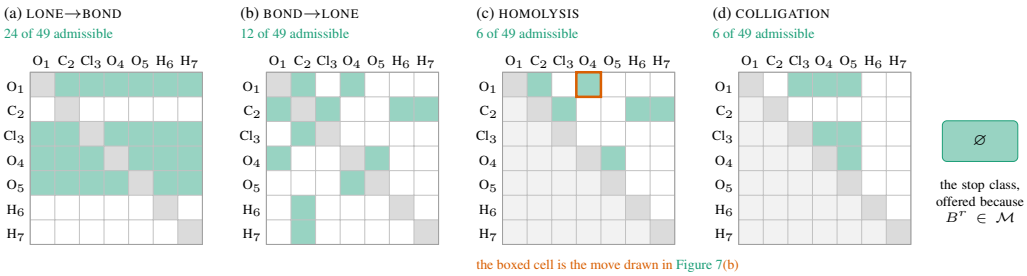
	O_1	C_2	Cl_3	O_4	O_5	H_6	H_7
x_{ii} (lone-pair electrons)	4	0	6	4	5	0	0
β_i (bonding pairs)	2	4	1	2	1	1	1

We take LONE→BOND at the ordered pair $(i, j) = (O_1, C_2)$, whose increments are $(\delta_\tau^{ii}, \delta_\tau^{jj}, \delta_\tau^{ij}) = (-2, 0, +1)$. The five tests, in the order of (30), are

$$\underbrace{0 \le x_{ii} - 2 = 2 \le 14,}_{4 \text{ — } O_1 \text{ has a lone pair to give}} \qquad \underbrace{0 \le x_{jj} + 0 = 0 \le 14,}_{0 \text{ — } C_2\text{'s diagonal is untouched}} \qquad \underbrace{0 \le x_{ij} + 1 = 2 \le 3,}_{1 \text{ — the bond may grow to a double}}$$

$$\underbrace{\beta_i + 1 = 3 \le 12,}_{2 \text{ — } O_1 \text{ gains a bonding pair}} \qquad \underbrace{\beta_j + 1 = 5 \le 12,}_{4 \text{ — } C_2 \text{ gains one too}}$$

where the value from which each test starts is written under its symbol. The first line is the clause $x + A_y \in \mathcal{X}$, and the second line holds the two bonding-pair clauses. All five tests pass, so the cell



How to read a panel. There is one panel per move type. The row is the first atom i of the move and the column is its second atom j, so a cell *is* a candidate move (τ, i, j). **Teal** marks that all five tests of (30) pass, so the move is offered. **White** marks that at least one test fails, so the class lies outside $\mathcal{Y}(x)$ and receives no mass. **Grey** marks what is not a move at all, namely the diagonal $i = j$, which no move indexes, and the lower triangle of HOMOLYSIS and COLLIGATION, which are symmetric in their two atoms and so are offered once per unordered pair. Across the four panels, 48 of the $4 \times 49 = 196$ cells are teal, and with the stop class this gives 49 admissible classes against 148 inadmissible ones.

Figure 9: The admissible set $\mathcal{A}(x)$ at the reactant of Figure 3, drawn in full, with every move that the model may consider at that one state. The tests are worked out in Appendix G.1. BOND→LONE and HOMOLYSIS need a bond to break, so their panels *are* the adjacency matrix of the molecule. LONE→BOND needs a lone pair to donate, so the rows of C_2, H_6 and H_7, which have none, are empty, whereas their *columns* are not, as these atoms are able to receive. COLLIGATION needs a spare electron at both ends, so it appears only among the four atoms whose diagonal is non-zero.

(O_1, C_2) of Figure 9(a) is admissible. We now take the *same* move at the reversed pair (C_2, O_1), where the first test reads $0 \le 0 - 2 = -2$ and fails, because C_2 has no lone pair to give away. One failure is enough, and the whole C_2 row of that panel is empty for the same reason. This is why the panels are not symmetric. A move is indexed by an *ordered* pair, and (i, j) asks something of i that (j, i) asks of j.

G.2 THE CLOSED FORM OF THE TABLE

We write a local state $(a_i, x_{ii}, \beta_i(x))$ of §2 as a triple (a, d, b) of an element a, a lone-pair electron count d and a bonding-pair count b. $\mathcal{T}$ is the set of local states that obey the valence rules. We obtain it by enumeration, against the implementation of these rules that the evaluation uses (Appendix K.2). The enumeration returns almost a single inequality, the octet rule $c_i \le 4$ of (1), with the constant 4 replaced by a per-element capacity C_a,

$$(a, d, b) \in \mathcal{T} \iff \tfrac{1}{2}d + b \le C_a \qquad \text{that is} \qquad \big(a_i, x_{ii}, \beta_i(x)\big) \in \mathcal{T} \iff c_i(x) \le C_{a_i}. \quad (31)$$

The N local clauses of (5) are therefore N integer comparisons against a lookup of one number, and $\mathcal{M}$ is a set of lattice points cut out by one linear equation and $O(N^2)$ linear inequalities. Read in this way, $\mathcal{T}$ leaves exactly one quantity that the algebra does not supply, namely the capacities C_a, one per element of the data. Chemistry answers $C_C = 4$ and $C_H = 1$, and if (4) were a statement about chemistry, this would end the matter, but it is not such a statement. Its second probability is decided by the evaluation, and a capacity asserted on paper would guarantee validity only against itself.

G.3 MEASUREMENT OF THE CAPACITIES

We fix one element a and ask which pairs (d, b) it may wear. The question is finite, as d runs over $\mathcal{V}_{ii} = \{0, \dots, 14\}$ and b over $\{0, \dots, \bar{b}\}$ with $\bar{b} = 12$. We therefore enumerate the whole grid, every element of the data with every such pair, and decide each of them by building the smallest molecule that wears it and seeing whether the evaluation gives it back unchanged. Figure 10(c) draws the five stages on one cell.

Stage 1, list the neighbourhoods. The number b says how many bonding pairs the atom holds in total, but *not* how they are shared out among its neighbours. Three pairs may be three single bonds, or one double and one single bond, or one triple bond, which is the same $b = 3$ but three different molecules. We write such a sharing-out as

$$b \;=\; o_1 + o_2 + \cdots + o_L, \qquad 1 \le o_j \le 3,$$

where L is how many neighbours the atom has and o_j is the bond order to neighbour j, capped at three because that is all the representation can express. These neighbourhoods are not interchangeable to the evaluation, and the model may produce any of them, so all of them must be tested.

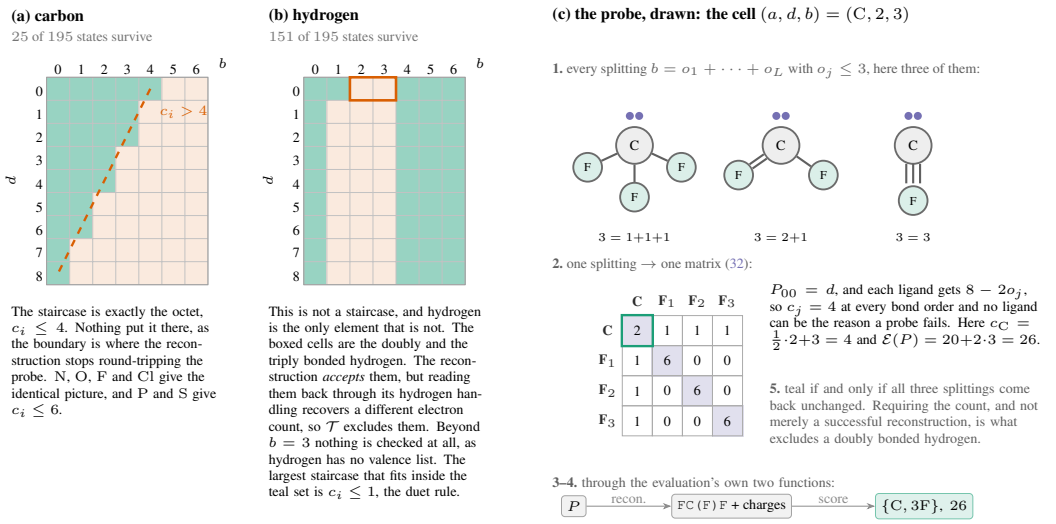


Figure 10: How the capacities C_a of (31) are measured. A cell is one local state, with $d = x_{ii}$ lone-pair electrons down the side and $b = \beta_i$ bonding pairs across the top, and teal means that the reconstruction and the valency check both returned the probe unchanged. (a) For carbon the enumeration returns exactly the staircase $\frac{1}{2}d + b \leq 4$, so $C_{\mathrm{C}} = 4$ is read off rather than assumed. Nitrogen, oxygen, fluorine and chlorine give the same picture, and phosphorus and sulphur give $C_a = 6$. (b) Hydrogen is the single element of the data whose teal set is not a staircase, and its shape is an artefact of the evaluation rather than a fact about chemistry (Appendix G.4). (c) The procedure of Appendix G.3 on one cell.

Stage 2, build one probe per splitting. A splitting is turned into a molecule by giving the atom exactly L neighbours, one per o_j, and nothing else. This is a bond–electron matrix like any other, of size $1 + L$,

$$\underbrace{P_{00} = d}_{\text{the lone pairs being tested}}, \qquad \underbrace{P_{0j} = P_{j0} = o_j}_{\text{the bond to neighbour } j}, \qquad \underbrace{P_{jj} = 8 - 2o_j}_{\text{the neighbour's own lone pairs}}. \tag{32}$$

Every neighbour is a fluorine, which constrains the probe least. It is univalent, so it cannot attach itself to a second atom and change the neighbourhood being tested, its capacity is a plain octet, and its own lone-pair electron count can be set to whatever the bond order demands. The last point is what $P_{jj} = 8 - 2o_j$ achieves. Reading c_j off (1) gives $c_j = \frac{1}{2}(8 - 2o_j) + o_j = 4$. A neighbour therefore sits at a full octet whether the bond to it is single, double or triple, so *no fluorine neighbour is ever the reason a probe is rejected*. That the verdict then carries over to the same local state in any other neighbourhood is an assumption, which Appendix K.3 audits.

Stages 3 and 4, run the evaluation and demand the probe back. We hand P to the two checks that the evaluation itself performs, and to no others. The first is the reconstruction, which turns a matrix into a molecule. The second is the valency check, which reads that molecule back and reports an atom multiset and a valence-electron count. Both checks must return what went in. Demanding the count, and not merely a successful reconstruction, is what makes the measurement mean something. The valency rules are stated only for the main-group elements that have them and accept everything else, metals and hydrogen included, so a doubly bonded hydrogen is accepted while the electron count recovered for it differs from the one that P specified, its formal charge having been read differently. Such a state would pass a naive stop mask and fail the metric, which is the gap that (4) forbids.

Stage 5, accept the cell only if every splitting survived. A cell records a property of the pair (d, b), and the model may realise that pair by any of the splittings of stage 1, so the cell is accepted only when all of them come back intact. One failure among them condemns the whole cell. Both quantifiers are necessary. A naive probe, which uses one splitting and no electron count, lets an untrained roll-out emit non-conserving candidates, and the procedure above does not. We roll out Arrow Flow Matching at its smallest width, untrained, on a fixed random sample of 19,772 test steps with ten candidates per step. With the naive probe 416 of the 197,720 candidates are non-conserving, and with the procedure above none is.

G.4 What the enumeration returns

The closed form (31) summarises almost all of what the enumeration accepts. For every element except hydrogen, the accepted set is *exactly* the staircase $\frac{1}{2}d + b \leq C_a$. The capacity is therefore well defined and the closed form is exact, and C_a takes only four values.

C_a	elements	why
4	C, N, O, F, Cl, Ge, Br, Sn, Pb	the octet (1), recovered and not assumed
5	Si	
6	P, S, As, Se, Sb, Te, I, Bi	the expanded shell of a third-row atom
∞	the remaining elements, every metal, and He, Li, Be, B, ...	no valence list is checked, so the only cap is the extent of the grid

The three finite groups are the elements for which the evaluation states a valency rule, and $C_a = \infty$ means that the whole grid is accepted, so for those elements the table imposes nothing beyond the ranges of the state space that (10) already enforces. Of the 12675 cells of the grid, 9580 are accepted. Over the test split, C, N, O, F, Cl and Br reach exactly $c_i = 4$, so the octet is attained and not only respected.

The exception and its cost. Hydrogen alone returns something that is not a staircase (Figure 10(b)). Here $b = 0$ is accepted at every d, $b = 1$ is accepted only at $d = 0$, $b \in \{2, 3\}$ is rejected everywhere, and $b \geq 4$ is accepted everywhere again. The rejections are real, being the doubly and triply bonded hydrogens of stage 4. The acceptances beyond them are not a chemical statement at all. The evaluation states no valency rule for hydrogen, so beyond the doubly and triply bonded cases it accepts every state. The largest staircase that fits inside the accepted set of hydrogen is $c_i \leq 1$, the duet rule, and no hydrogen of the data leaves it.

Remark 9 (The enumeration audits the algebra rather than replacing it)**.** The construction needs (31) together with one capacity per element, and the enumeration is how these capacities are *audited* against the evaluation that grades the output. The audit passes wherever the evaluation states a valency rule. Adopting $C_{\mathrm{H}} = 1$ for the one element where it has none would make $\mathcal{T}$ the closed form at every element. It would shrink $\mathcal{T}$ from 9580 local states to 9433, forbidding states that the evaluation accepts and that no molecule of the data contains. We ship the enumerated table because Proposition 3 is stated against (4), and the enumeration is what the evaluation supports. The algebraic table is a subset of it, and it is the choice to make if the guarantee is to be stated against chemistry instead.

G.5 The jam

Suppose the table were imposed at *every* move rather than only at termination. By (31), this means keeping $c_i \leq C_{a_i}$ at every atom throughout, so the question is what a move does to c. Substituting (29) into (1) answers it completely,

$$\delta c_i = \tfrac{1}{2}\delta_\tau^{ii} + \delta_\tau^{ij}, \qquad \delta c_j = \tfrac{1}{2}\delta_\tau^{jj} + \delta_\tau^{ij}, \qquad \delta c_{i'} = 0 \ \ (i' \neq i, j), \tag{33}$$

and reading the four rows of the alphabet off it gives its whole effect. LONE→BOND has $(\delta c_i, \delta c_j) = (0, +1)$, as the pair already touched i and now also touches j, BOND→LONE has $(-1, 0)$, and HOMOLYSIS and COLLIGATION have $(-\frac{1}{2}, -\frac{1}{2})$ and $(+\frac{1}{2}, +\frac{1}{2})$, the shared pair becoming half a pair each and back again. Summing these and using $\delta_\tau^{ii} + \delta_\tau^{jj} = -2\delta_\tau^{ij}$ from (9) gives $\delta c_i + \delta c_j = \delta_\tau^{ij}$, so a move that forms one unit of bond order raises $c_i + c_j$ by one and a move that breaks one lowers it by one. Moreover, *a move that raises c at an atom does not lower it there first*, as the two atoms move independently and a pair move raises c at one of them while leaving the other alone. An atom already at its capacity therefore blocks every move that would raise it, whatever the rest of the chain is trying to do. This is the octet rule acting as a *dynamical* selection rule. It can block a chain of arrows completely. We call a step a *jam* when no ordering of its arrows keeps every atom within its capacity. On the test split, about one step in twenty that a chemist draws with arrows alone is a jam.

Figure 11 works out the canonical instance, the Diels–Alder reaction. Six carbons close a ring, three bonds lose a pair and three gain one, and every carbon holds four pairs before and after the step. A chemist draws the step as three bond migrations. Each migration lowers c by one at one carbon and raises it by one at another, so every possible first arrow leaves some carbon with five pairs, and a

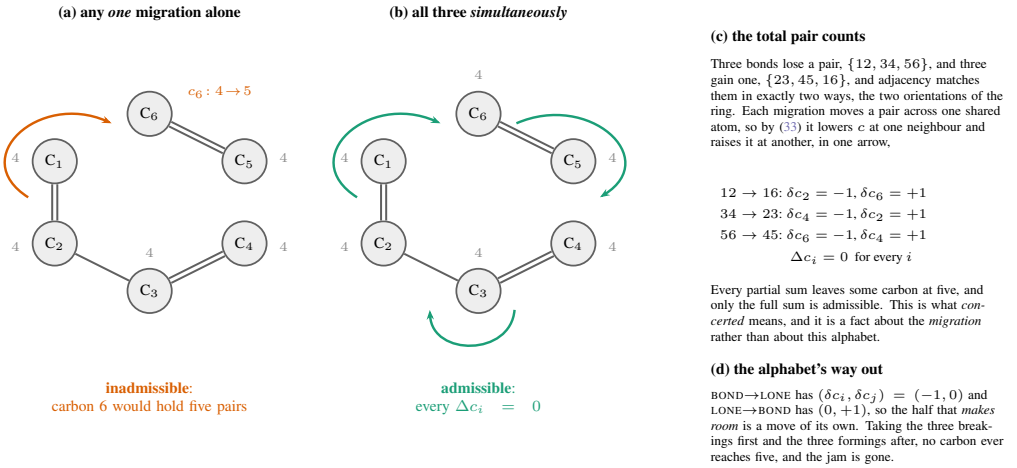


Figure 11: The Diels–Alder reaction is jammed *as arrows* and not as moves. Butadiene and ethylene close a six-membered ring, three bonds lose a pair and three gain one, and every carbon starts and ends with a full octet, $c_i = 4$. (a) Any single migration towards the product raises some carbon to five pairs before it has lost one. The same happens for all six migrations of both orientations, so there is no admissible *first* arrow and hence no admissible ordering of arrows at all. (b) The three migrations applied together change no total pair count anywhere, which is what *concerted* means. (c)–(d) The migration is not in the alphabet (Remark 7). Expanded into its two moves, its lowering half becomes a move that can be taken on its own, and the jam disappears.

chain of arrows gated at every move cannot start. The move alphabet removes this jam, because the two halves of a migration are separate moves (Remark 7). By (33), the BOND→LONE half lowers c at one atom and raises it nowhere, and the LONE→BOND half raises it at one atom. A chain can take the three halves that make room first and the three that fill it afterwards, so no carbon ever holds more than four pairs.

Arrow Flow Matching nevertheless gates the stop class alone, because the valence rules describe the molecules before and after a step. The chain moves freely in between, and each intermediate state shows where a pair sits between two arrows, such as a pair held as a lone pair on an atom or a hydrogen between its old and new partner.

G.6 The place where the table is checked

The construction gates the *stop* move alone with $\mathcal{T}$, as (10) and Proposition 3 do. Once the step is cut into moves, a gate on every move also becomes possible, as the jam of arrows disappears. The greedy procedure then finds an admissible ordering under it for every test step, so gating the stop class alone is a choice and not a necessity. §5 explains why we gate the stop class alone. The intermediate states that this choice allows are the ones a chemist draws between two arrows. Figure 8 passes through both, a hydride between its old and new partner and a carbon that holds four and a half pairs for one move.

G.7 Proof of Proposition 3

Proposition 3 (Validity)**.** *Let a chain start at B^r, take moves in $\mathcal{A}(x_k)$, and be permitted to terminate only at a state x with $(a_i, x_{ii}, \beta_i(x)) \in \mathcal{T}$ for every i. Then it terminates in $\mathcal{M}$.*

Proof. By (5), membership in $\mathcal{M}$ consists of three conditions at once. The terminal state must be a point of the state space $\mathcal{X}$, it must carry the electron count of the reactant (R3), and it must satisfy the N local conditions (R4). We check them in turn.

The state space. We show that $x_k \in \mathcal{X}$ for every k, by induction. The chain starts at $x_0 = B^r$, which lies in $\mathcal{X}$ because it is the bond–electron matrix of a real molecule. For the inductive step, the move y taken at x_k lies in $\mathcal{A}(x_k)$ by hypothesis, and the first clause of (10) is $x_k + A_y \in \mathcal{X}$. Hence $x_{k+1} \in \mathcal{X}$.

Table 10: The two constructions. Only the elementary event changes, and the backbone network, the bridge idea and the proper scoring rule are shared.

	Discrete FlowER (§3)	Arrow Flow Matching (§5)
state space	$\mathcal{X}$ (3)	$\mathcal{X}$ met with the sheet, stopping only in $\mathcal{M}$
elementary event	an edit, the change of one entry	a move, $\mathcal{E}(A_y) = 0$
head	$N(N+1)/2$ softmaxes, over $\mathcal{V}_{ij}$ of ≤ 15 classes each (7)	one softmax, over $\mathcal{Y}(x)$ of $\leq 4N^2{+}1$ classes (12)
loss	cross-entropy on the endpoint (8)	cross-entropy on the next move (13)
conservation	mean exact, variance > 0 for $0 < t < 1$ (26)	surely, every moment (Prop. 2)
validity	not guaranteed (§4)	surely (Prop. 3)
score; beam	factorised (24); no beam	exact trajectory (14); beam

The electron count. Every update is a move of the alphabet, so $\mathcal{E}(x_k) = \mathcal{E}(B^r)$ for every k by Proposition 2. This holds at the terminal state in particular.

The local conditions. The chain is permitted to terminate only at a state every atom of which has $(a_i, x_{ii}, \beta_i(x)) \in \mathcal{T}$. This is the hypothesis, and it is the N local clauses of (5).

The three conditions together are membership in $\mathcal{M}$. □

H THE MOVE BRIDGE

This appendix expands the bridge over moves of §5. Table 10 sets Discrete FlowER and Arrow Flow Matching side by side.

H.1 DECOMPOSITION

Read as a picture, Δ marks every entry it touches with a sign. An off-diagonal $\Delta_{ij} < 0$ is a bond *broken* and $\Delta_{ij} > 0$ a bond *formed*, whereas a diagonal $\Delta_{ii} < 0$ is an atom *donating* a lone pair and $\Delta_{ii} > 0$ an atom *accepting* one. Cutting Δ into arrows means threading these marks together. The rule is *alternation*. An arrow takes a pair out of one place and puts it into the next, so along a chain of atoms the bonds must read break, form, break, form, and so on. The marks are grouped into walks

$$i_0 \xrightarrow{\pm} i_1 \xrightarrow{\mp} i_2 \xrightarrow{\pm} \cdots$$

whose consecutive bond changes alternate in sign. The two ends of a walk are settled by the diagonal. A walk that *begins* with a formation must start at an atom donating a lone pair, as something has to supply the new bond. A walk that *ends* with a break must end at an atom accepting one, as the broken pair has to land somewhere. A walk that closes into a cycle has no ends, so it neither donates nor accepts. This is the pericyclic case, and the Diels–Alder reaction of Appendix G.5 is one such six-atom cycle. Every consecutive break-and-form pair along a walk is one curly arrow. Leftover donations matched to leftover acceptances with no bond change between them are lone-pair transfers, and they become single arrows of their own. Each arrow is then rewritten in the alphabet. An arrow into or out of a diagonal is already one of the two pair moves, an arrow from bond to bond is a migration and expands into the two moves of Remark 7, and a step of odd parity is written with the single-electron moves of Appendix F.3. That such a decomposition always exists is a theorem about degree-preserving transformations (Flamm et al., 2023).

H.2 ORDERING

The m moves of a step are a *multiset* rather than a sequence, and their sum is Δ whatever order they are applied in. Not every order is usable, however, because every intermediate matrix must satisfy (10), and a step that both forms and breaks the same bond can drive its order to four on the way if the two moves are taken in the wrong order. We order them greedily, at each point taking any remaining move that is admissible at the current state. This carries no guarantee of success, because a step may contain two moves acting on the same entry in opposite directions, so the counts need not evolve monotonically. It has not failed on the training or the test split. It finds an admissible order also for the rare steps whose unordered factorisation would produce a bond of order four. The

ordering is deterministic per step, so it is computed once, offline, and cached with the rest of the featurisation. Figure 8 in Appendix F.2 shows this subsection and the preceding one on one step, with the decomposition, the order, the states of the path and the target at each k.

H.3 The choice of discrete time

A continuous-time version of §5 is available. It gives each move a rate and integrates the path-space divergence (21) against a bridge whose unapplied moves fire at rate λ_t. This is strictly more machinery for the same content, and it costs three things. First, the weight that the time change puts on a training state, λ_t times the number $m - k_t$ of moves still to be applied, has unbounded variance as $t \to 1$. Second, stopping becomes a limit rather than a class of the softmax. Third, there is no per-candidate likelihood to rank by without integrating. The prefix length of (11) avoids all three.

I The shared network and the two heads

Discrete FlowER and Arrow Flow Matching share one backbone network and differ in the output layer that turns its per-atom vectors into logits.

I.1 The backbone

The network must see both where the bridge started and where it currently stands, as the difference is what identifies the entries that have already moved. Both are small integers, so we pack them into a single index per entry and embed them with a lookup table,

$$\gamma_{ij} = 4 \cdot B^r_{ij} + (x_t)_{ij} \in \{0, \dots, 15\} \quad (i \neq j), \qquad \gamma_{ii} := 0,$$
$$\tilde{\gamma}_i = 15 \cdot B^r_{ii} + (x_t)_{ii} \in \{0, \dots, 224\},$$

where 4 and 15 are the sizes of $\mathcal{V}_{ij}$ off and on the diagonal. The diagonal is removed from the pairwise code, whose alphabet of $15 \times 15 = 225$ values would overflow the sixteen bond codes, and it is carried per atom instead, which also reduces its storage from N^2 to N. With $z_i^{(l)} \in \mathbb{R}^{d_{\text{model}}}$ the hidden state of atom i at layer l, d_{model} the width, $\zeta(t)$ a sinusoidal time embedding and $\text{emb}^{\text{pair}}, \text{emb}^{\text{el}}, \text{emb}^{\text{dia}}$ learned lookup tables, each of which is an affine map of a one-hot code, we have

$$R_{ij} = \text{GELU}\big(\text{emb}^{\text{pair}}[\gamma_{ij}]\big), \qquad z_i^{(0)} = \sqrt{d_{\text{model}}}\Big(\text{emb}^{\text{el}}[a_i] + \text{emb}^{\text{dia}}[\tilde{\gamma}_i]\Big),$$

and, for attention head h,

$$\text{bias}^{(l)}_{h,ij} = \big\langle \mathsf{Q}^{(l)}_{h,i} + \mathsf{Q}^{(l)}_{h,0},\ [R_{ij}; \zeta(t)]_h \big\rangle, \qquad z^{(l+1)} = \text{Attn}^{(l)}\big(z^{(l)}; \text{bias}^{(l)}\big),$$

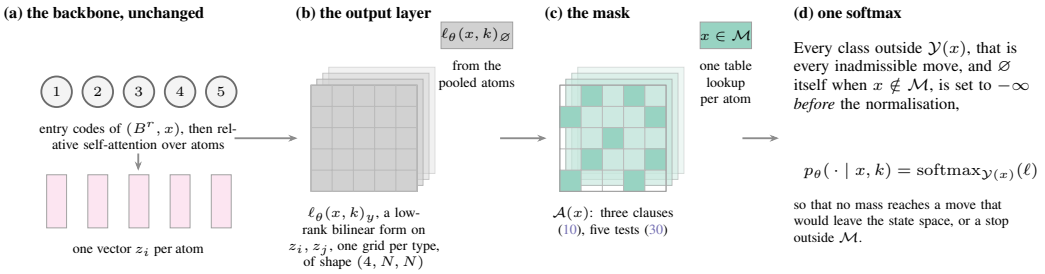
where $\mathsf{Q}^{(l)}_{h,i}$ is the query of atom i in head h, $\mathsf{Q}^{(l)}_{h,0}$ is a learned query shared by all atoms and $[\cdot]_h$ is the slice of head h, so that the pairwise code enters as a relative attention bias rather than as an $O(N^2 d_{\text{model}})$ tensor. Sans-serif letters denote learned projections. In Arrow Flow Matching the move count k enters through the same sinusoidal embedding, so that $\zeta(k)$ replaces $\zeta(t)$ wherever it appears, and x_k replaces x_t in the codes.

I.2 The entrywise output layer

The logit vectors ℓ_{ij} of (7) come from two output layers, $\ell_{ij} = \ell^{\text{bond}}_{ij,\cdot}$ off the diagonal and $\ell_{ii} = \ell^{\text{lone}}_{i,\cdot}$ on it, where

$$\ell^{\text{bond}}_{ij,\nu} = \frac{\big\langle \mathsf{Q}_{i,\nu}, \mathsf{K}_{j,\nu} \big\rangle + \big\langle \mathsf{Q}_{j,\nu}, \mathsf{K}_{i,\nu} \big\rangle}{\sqrt{d_{\text{head}}}} + \Gamma^{\text{bond}}[\gamma_{ij}]_\nu + \psi^{\text{bond}}_\nu(t), \qquad \nu \in \mathcal{V}_{ij} \ \ (i < j),$$
$$\ell^{\text{lone}}_{i,\nu} = \big(W_o z_i\big)_\nu + \Gamma^{\text{lone}}[\tilde{\gamma}_i]_\nu + \psi^{\text{lone}}_\nu(t), \qquad \nu \in \mathcal{V}_{ii},$$

where $\mathsf{Q}_{i,\nu}, \mathsf{K}_{i,\nu}$ are low-rank projections of the last hidden state z_i, $\Gamma^{\text{bond}}, \Gamma^{\text{lone}}$ are learned bias tables and $\psi^{\text{bond}}_\nu, \psi^{\text{lone}}_\nu$ are learned time biases. The bond logit is a *symmetrised* bilinear form, so that $\ell^{\text{bond}}_{ij,\nu} = \ell^{\text{bond}}_{ji,\nu}$ for all i, j, ν. Symmetry is therefore a property of the parameterisation, and it is never imposed afterwards on a raw asymmetric field.



Nothing in (a) changes from the entrywise formulation, and only (b)–(d) do. The class set has no ragged shape and no dynamic size, as it is the fixed grid of (b) with the boolean tensor of (c) laid over it, so the mask costs less than the backbone (Appendix I.4).

Figure 12: The head. There are four grids of move logits, one per type and each indexed by an ordered atom pair, plus one stop logit from the pooled molecule, then a boolean tensor of the same shape plus one boolean for stopping, and finally a single softmax over what survives.

Copy initialisation. Most entries do not change during a step, so the correct output is overwhelmingly "keep the current value". With $\Lambda > 0$, setting $\Gamma^{\mathrm{bond}}[\gamma]_\nu|_{\mathrm{init}} = \Lambda\,\mathbb{1}[\nu = \gamma \bmod 4]$ and $\Gamma^{\mathrm{lone}}[\tilde{\gamma}]_\nu|_{\mathrm{init}} = \Lambda\,\mathbb{1}[\nu = \tilde{\gamma} \bmod 15]$ makes the network do that at initialisation, as by construction of the codes $\gamma_{ij} \bmod 4 = (x_t)_{ij}$ and $\tilde{\gamma}_i \bmod 15 = (x_t)_{ii}$, so the logit at the current value exceeds the others by Λ and $p_\theta^{ij}(x_{ij})|_{\mathrm{init}} = \exp\Lambda/(\exp\Lambda + 3) \approx 0.999$ for $\Lambda = 8$ and $i < j$. The rationale is quantitative, since the identity map is correct for all but $O(1)$ of the $N(N+1)/2$ entries, while an Adam update at learning rate η moves a logit by at most $O(\eta)$, so discovering it from a random start costs $O(\Lambda/\eta)$ updates spent on learning to copy.

I.3 The move head

The move head replaces the two output layers of Appendix I.2. Figure 12 draws the four stages of (12). The backbone produces one vector per atom. Four low-rank bilinear grids, one per move type, give the move logits, and the pooled molecule gives the stop logit. The mask (10) and the stop gate $x \in \mathcal{M}$ then fix the class set $\mathcal{Y}(x)$, and a single softmax normalises over it. No mass reaches a move that would leave the state space, or a stop at a state outside $\mathcal{M}$. By Remark 8 the class set is never empty, so (12) is always a well-defined distribution. Figure 9 shows stage (c) at a real state.

I.4 The mask on a GPU

The state-dependent alphabet costs no throughput, because every move is named by a type and an ordered atom pair. The class set is a fixed $(4, N, N)$ grid plus one class, the same shape as the output of the entrywise output layer, and the mask is a boolean tensor of that shape. The softmax over $\mathcal{Y}(x)$ in (12) is computed by setting every class outside $\mathcal{Y}(x)$ to $-\infty$ *before* the normalisation, as in constrained decoding for language (Hokamp & Liu, 2017; Post & Vilar, 2018; Willard & Louf, 2023). Four of the five tests of (30) read a *per-atom* quantity, so they are vectors of length N broadcast along the axis on which they do not depend. The remaining test, on x_{ij}, is the only one whose quantity varies over both atom axes. The sign of δ_τ^{ij} fixes the direction of its inequality, so it reduces to a choice between two boolean grids, room to grow a bond or a pair to give up, and no integer tensor of size $[N_{\mathrm{batch}}, 4, N, N]$ is ever materialised. The stop gate reads $\mathcal{T}$, stored as a flat boolean tensor indexed by (a_i, x_{ii}, β_i) with a few thousand entries resident on the device. The only admissible local state of the padding element is the all-zero one that padded atoms hold, so padding never blocks the stop class. The stages are the following.

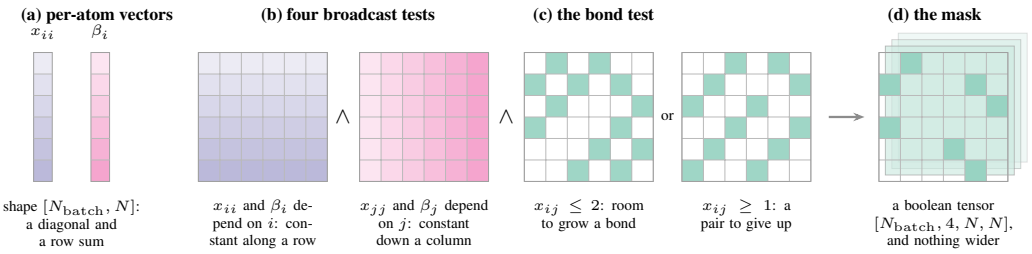


Figure 13: How the mask of Figure 9 is computed. The grids are schematic, with $N = 6$, and the shading shows the axis on which each factor depends. The mask costs less than the backbone network, which is quadratic in N and shared with Discrete FlowER.

stage	shape	cost
backbone network (unchanged)	$[N_{\text{batch}}, N, d_{\text{model}}]$, N^2 attention	dominant, unchanged
move logits, bilinear per type	$[N_{\text{batch}}, 4, N, N]$	one `matmul`
stop logit, from pooled atoms	$[N_{\text{batch}}]$	negligible
per-atom x_{ii} and β_i	$[N_{\text{batch}}, N]$	a diagonal and a row sum
move mask, the five tests (30)	$[N_{\text{batch}}, 4, N, N]$	four of the five broadcast over one axis
stop mask, one lookup per atom	$[N_{\text{batch}}, N]$	an `index_select`, then `all`
state update	$[N_{\text{batch}}, N, N]$	four scatter-adds

Building $\mathcal{A}(x)$ thus costs one diagonal, one row sum, four broadcast comparisons and one select, and the roll-out has no dynamic shapes and no host synchronisation. Figure 13 draws the broadcast structure.

J Decoding

This appendix expands the sampling of §5 and records which decoders and ranking rules each model supports.

J.1 The move decoder

The beam decoder, and where it applies. The model of §5 admits a *beam* decoder and Discrete FlowER does not, for the reason that a beam always needs. A beam extends partial hypotheses and compares their partial scores, so the output must factorise as an ordered chain of conditionals, each of which is conditioned on the prefix already committed to. Here it does, as (14) is that chain rule, x_k *is* the prefix, and truncating the sum at k gives a partial score on which every hypothesis of the same length is comparable. In (23) nothing is committed. The head is re-formed from scratch at each of the S Euler updates, the whole matrix is resampled, and the emitted candidate is read off the last update alone. The intermediate distributions are therefore not the conditionals of any factorisation of $p_\theta(\widehat{B} \mid B^r)$, so there is no prefix, no partial score and nothing that a beam could extend. Figure 14 draws both decoders. The rest of this subsection records the one way in which a beam can break a guarantee, and how the roll-out budget is handled.

A beam wider than the alphabet. There is one way in which a decoder can break a guarantee that the parameterisation holds. A beam of width N_{beam} asks each hypothesis for its N_{beam} best continuations, and a small molecule can have fewer admissible classes than that. A top-N_{beam} selection over a masked row then fills the surplus slots with classes whose log-probability is $-\infty$, which are the moves that the mask was built to forbid. Applying one of them takes the state out of the state space, and the candidate does not reconstruct. Before the fix, such slots lowered the validity of beam candidates below one on the small molecules, and the defect is invisible under sampling, where a Gumbel arg-max over $-\infty$ can never select a masked class. The fix is to read the surplus slots off the running *score* rather than off the class index. A slot whose cumulative log-probability is not finite holds no trajectory, so it takes no move and emits its last stoppable state. *The mask makes*

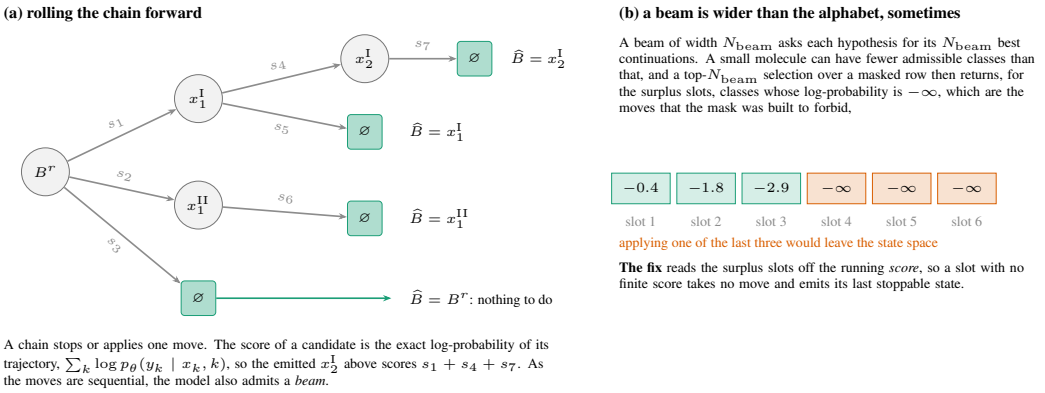


Figure 14: Sampling. The edge scores $s_1, \ldots, s_7$, which are the log-probabilities of the classes taken, are symbols rather than measurements. Every edge leaving a state is a class of one masked softmax, and the score of a candidate is a plain sum along its trajectory. Panel (b) shows the beam slots that the fix of Appendix J.1 handles.

the guarantee true of the distribution, and the decoder has to be careful not to sample outside its support.

The roll-out budget. The chain is capped at $S = 12$ moves, well above the 2.26 moves of an average test step. A hypothesis that has not stopped by then emits the last state at which it was allowed to stop, which always exists because the chain starts at $B^r \in \mathcal{M}$. The guarantee therefore holds for every budget, including $S = 0$, and the cap is a compute knob rather than a correctness assumption. The same fallback serves the residue of states inside $\mathcal{M}$ that the evaluation rejects (Appendix K.3). Figure 14(a) draws a roll-out.

J.2 Decoders and ranking rules of every model

A candidate list is produced by a *decoder*, which is beam search or sampling, and ordered by a *ranking rule*, which is the trajectory score, the marginal probability over trajectories, or the sampling frequency. A beam needs a sequential factorisation, as Appendix J.1 explains, and the score and the marginal probability need a log-probability per trajectory. Continuous FlowER (Joung et al., 2025) has neither. Its ODE is a one-shot map from noise, and every draw leaves with weight one. Discrete FlowER has the score (24) but no beam, as its entrywise head is one-shot as well. Arrow Flow Matching and the six sequence and graph baselines have both. Table 11 lists which of the five evaluation regimes of Appendix L.1 each model can fill. Sampling with frequency ranking is the only regime that every model fills, and it is the protocol of §6.

Table 11: The evaluation regimes that each model can fill. A dash marks a regime that the model cannot fill.

model	beam by score	sampling by score	sampling by frequency	beam by marginal	beam branch-aware
sequence and graph baselines	✓	✓	✓	✓	✓
Continuous FlowER	—	—	✓	—	—
Discrete FlowER	—	✓	✓	—	—
Arrow Flow Matching	✓	✓	✓	✓	✓

K Experimental setup

K.1 Data

We use the mechanistic dataset of Joung et al. (2025) exactly as released, with no filtering, deduplication or resplitting of our own. Its splits hold 1,933,750 training, 21,723 validation and 218,997 test elementary steps, and 31,189 test pathways, of which 3000 seeded ones are rolled out for the recursive pathway metric. All but six hydrogens of the test split are mapped explicitly (Remark 4), and aromatic rings are Kekulé-resolved before the matrix is built.

K.2 The evaluation pipeline

The chemical validity of a candidate, in the sense of (4), is decided throughout by one fixed implementation of the valency rules, `RDKit` (Landrum et al., 2024) writing and rereading a SMILES string (Weininger, 1988). The capacities of $\mathcal{T}$ are enumerated with the same implementation in Appendix G.3. Which implementation it is matters nowhere in the construction. The requirement is that one implementation does both jobs.

The enumeration tightens the table at stages 1 and 4 of Appendix G.3. It also exposed a defect that no table can fix, as it is not local at all. The reconstruction rebuilds a molecule by copying the reactant as a template, deleting its bonds, and installing the bonds that the predicted matrix specifies. The template was produced by resolving aromatic rings into alternating single and double bonds, but the reconstruction leaves its *atoms* still flagged as aromatic. A bond–electron matrix knows nothing of aromaticity and is a Kekulé structure, and those flags belong to the bonds of the template rather than to the ones now being installed. Left in place, the writer emits an aromatic-looking string joined by explicit single and double bonds, and re-reading that string fails whenever the step changed a ring. The consequence is that a state can satisfy every local condition of (5) and still fail the read-back half of (4), so *the local characterisation of the sector is false against the reconstruction as it stood*. On the untrained roll-out of Appendix G.3, 9,573 of 197,720 candidates (4.8%) that reconstruct correctly fail in this way, and clearing the flags removes all of them. The products of the test split rarely open an aromatic ring, and only 5 of them fail. The fix is one line and changes no molecule that already reconstructed, so it can only raise a validity number. On Discrete FlowER it raises validity from 0.9754 to 0.9761 and moves top-1 accuracy by 10^{-4}, which is why it went unnoticed. For a model whose claim is exactly one, it decides whether the claim holds. All numbers reported in this paper are measured after the correction, for every model.

K.3 The scope of the guarantee

This subsection states what the validity of 1 in Table 2 rests on. Proposition 3 guarantees that every candidate lies in $\mathcal{M}$, where every atom has a local state in $\mathcal{T}$. The evaluation checks R4 with one implementation of the valence rules, and $\mathcal{T}$ is enumerated against exactly this implementation (Appendix G.3). A state in $\mathcal{M}$ can therefore fail R4 only through a condition that involves several atoms at once. This is rare. Of 400,000 beam candidates of a trained model, 32 lay in $\mathcal{M}$ and failed R4, a share of 8×10^{-5}. In each of them the evaluation perceived a ring as aromatic and could not assign it alternating single and double bonds. This is a condition on the whole ring, and no per-atom table can see a ring. For such a candidate the decoder returns the last state of the same chain at which it was allowed to stop (Appendix J.1). This state was visited by the chain, but it replaces the candidate, and it should be read as a replacement.

K.4 Training

Every model is trained for at most 100 epochs with AdamW, gradient clipping at 1.0, and early stopping on validation loss with patience 25. The learning-rate schedule is a linear warmup followed by cosine decay to a floor of 0.1 of the peak rate. The inverse-square-root Noam decay used by the originating papers is replaced because its rate collapses to a small fraction of the peak within a few warmup lengths and keeps falling, which under-trained the transformer baselines. Optimiser defaults follow the AdamW convention $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$. The transformer-family models (Continuous and Discrete FlowER, Arrow Flow Matching, the Molecular Transformer, Graph2SMILES) override $\beta_2 = 0.998$ and $\varepsilon = 10^{-9}$, per their published settings. The checkpoint with the best validation loss is the one evaluated. Each model is trained at five widths, and every reported parameter count is measured on the instantiated model rather than being nominal.

K.5 Decoding budgets

Every model emits at most ten candidates per step. Arrow Flow Matching rolls out at most $S = 12$ moves per hypothesis, and Continuous FlowER integrates its ODE for 10 Euler updates, as does the entrywise chain of §3 with $S = 10$. The "marginal" ranking rule divides the log-probability of each trajectory by the GNMT length penalty $((5+\text{length})/6)^{0.6}$ before summing over the trajectories that

reached the same product, and ELECTRO normalises its own scores and therefore uses the exponent 0.

Table 12: Architecture and optimiser hyper-parameters at the largest width. The width knob names the quantity that is varied across the five widths, and the ladder is given below the table. The column S is the number of updates in which a candidate is generated, namely the Euler updates of the ODE solver of Continuous FlowER and of the chain (23) of Discrete FlowER, and the number of moves of (14) for Arrow Flow Matching, with a dash where a model has no such parameter. The quantity σ is the noise scale of Continuous FlowER.

model	architecture at the base width	S	width knob	peak lr	warmup	batch
Molecular Transformer	6+6 layers, d_{model}=256, 8 heads, d_{ff}=2304	—	d_{model}, d_{ff}	1×10^{-3}	1000	256
Graph2SMILES	6+6 layers, d_{model}=384, 8 heads, d_{ff}=2304	—	d_{model}, d_{ff}	4×10^{-4}	2000	256
MechT5	2 enc. + 8 dec. layers, d_{model}=512, 6 heads, d_{ff}=768	—	d_{model}, d_{ff}	3×10^{-4}	1000	512
MechLLaMa	6 layers, d_{model}=512, 4 heads, d_{ff}=1088, RoPE	—	d_{model}, d_{ff}	3×10^{-4}	1000	512
Reactron	3 message-passing + 3 attention layers, d_{model}=608	—	d_{model}	2×10^{-4}	2000	128
ELECTRO	4 GGNN layers, d_{model}=1024, path length ≤ 10	—	d_{model}	1×10^{-4}	2000	128
Continuous FlowER	6 layers, d_{model}=512, 16 heads, d_{ff}=2048, σ=0.15	10	d_{model}, d_{ff}	3×10^{-4}	4000	96
Discrete FlowER	the same backbone, entrywise head	10	d_{model}, d_{ff}	3×10^{-4}	4000	96
Arrow Flow Matching	the same backbone, masked move head	12	d_{model}, d_{ff}	3×10^{-4}	4000	96

The width ladder. The five widths nano $<$ micro $<$ mini $<$ small $<$ base are set by the knobs above so that every model spans roughly 1.2M to 21M parameters. The three flow models share one ladder, $(d_{\text{model}}, d_{\text{ff}}) \in \{(128, 512), (128, 1536), (256, 1024), (384, 1536), (512, 2048)\}$, so their curves may be read against each other directly. The sequence models use $(64, 512) \ldots (256, 2304)$ for the Molecular Transformer and $(128, 768) \ldots (384, 2304)$ for Graph2SMILES, the two mechanistic language models $(128, 256) \ldots (512, 768)$ and $(128, 320) \ldots (512, 1088)$, and the graph models $d_{\text{model}} \in \{256, 384, 512, 768, 1024\}$ for ELECTRO, and $d_{\text{model}} \in \{160, 256, 384, 512, 608\}$ for Reactron. Two adjustments were forced by numerical stability rather than chosen by us. Reactron diverges at its default peak rate once its width exceeds about four times the published one, so its two largest widths use 2×10^{-4} with a 2000-step warmup, and Continuous FlowER's ODE produces non-finite states for embedding widths whose per-head width is not a multiple of eight, so its micro point keeps $d_{\text{model}} = 128$ and grows d_{ff} instead of taking an intermediate width.

L Further results

This appendix expands the results of §6.

L.1 The evaluation protocol

Candidate lists and how they are read. Every model emits a ranked list of at most ten candidates. Each of them is rebuilt into a molecule, duplicates are merged, and a short list is padded out with failures. A recorded product counts as found when it appears among the top k, compared as a canonical atom-map-free string. Two quantities appear in Figure 15 but not in Table 2. *Pathway accuracy at k* is the recursive metric of §6 read at $k > 1$. *Teacher-forced pathway accuracy at k* is the stricter conjunction in which every step of a pathway is given its recorded reactant and must rank the recorded product within k. The composite score is a geometric mean rather than an arithmetic one, so that a value near zero in one factor cannot be offset by the others.

The five evaluation regimes. A regime pairs a decoder with a ranking rule and a notion of the right answer. Table 11 lists which regimes each model can fill, and sampling with frequency ranking is the only one that all nine fill. Any other choice would either grant Arrow Flow Matching a decoder

that its two closest competitors do not possess, or drop a model from the comparison. The five regimes below score the same trained models on the same test split. They differ in how candidates are generated, how they are ordered and what counts as the right answer. The model and its weights never change. Appendices L.3 and L.4 give what Arrow Flow Matching scores under the other four regimes, and record that its guarantees and accuracy are insensitive to the choice.

Beam. Beam search over the decoder of the model itself, with each distinct product ranked by the single highest-scoring trajectory that reached it. The other beam regimes are read against it.

Sampling, by score. Candidates drawn independently from the model rather than searched for, then ranked exactly as above. Read against the first regime, this isolates what the search itself is worth.

Sampling, by frequency. The same draws as in the previous regime, ranked instead by how many of them landed on each product. The scores of the model are discarded, which is the only rule available to a decoder that computes none, and therefore the only regime in which all nine models are comparable. This is the regime of §6.

Beam, by marginal probability. The same candidates as in the first regime, reordered. A product is reachable by many trajectories, most visibly by the orderings of one and the same multiset of moves. Its probability is the sum over these trajectories rather than the largest one. Taking the largest discards that mass, and it discards more of it from the products with more routes in. Each trajectory is first divided by a length penalty, since a score that sums one negative term per decision otherwise prefers short trajectories for no reason beyond their being short. That the sum is the right quantity does not make it the better ranking, and measured here it is not uniformly one. It pays where a decoder really does reach one product by many distinct routes, and it costs where a beam instead holds near-duplicates of one route, whose mass it then double-counts.

Beam, branch-aware ground truth. The predictions and the ordering of the first regime, scored differently. The ground truth of the dataset is set-valued, as 20.2% of test steps sit on a reactant that some other step carries to a different product, and the default metric reads it as if it were single-valued, so a deterministic decoder is marked wrong for answering a branch point with the wrong one of the answers of the data itself. Here a step counts as correct when the model returns a product the split records for that reactant. R1–R4 are properties of a molecule and not of the ground truth, so those panels repeat the first regime exactly, and only the accuracy panels and the score change.

The ceiling drawn at $k = 1$. The dashed segment at $k = 1$ in every accuracy panel is the highest top-1 value that a deterministic decoder can reach in that panel. A decoder is a function of its input, so where the data record several products for one reactant it can satisfy only one of them. On this split, 12,367 of the 31,189 test pathways carry a step whose mapped reactant is carried to a different product by some other pathway, and 20.2% of test steps sit on such a reactant, which caps the top-1 step accuracy at 0.8840 and the teacher-forced conjunction at 0.6035. The recursive pathway metric is capped at 1.0000, as it needs only its endpoint to match. The bound constrains $k = 1$ only. A top-2 number may exceed it, and a *sampling* decoder may exceed it even at $k = 1$, by answering two occurrences of one reactant differently, which is inconsistency rather than accuracy.

The budget. Every regime is given the same budget of ten candidates per test step, a beam of ten in the beam regimes and ten independent draws in the sampled ones, and duplicates are merged afterwards, so that no model is charged for candidates that its competitors were not allowed either. Appendix K.5 gives the per-model decoding budgets.

L.2 The main comparison and the width sweep

Figure 15 plots the comparison of Table 2, and Figure 16 repeats the comparison at all five widths, both under the sampling decoder with frequency ranking of §6.

Table 16 repeats Table 2 with the sample standard deviation of every entry. Each entry is the mean over 10 runs at the nano width. A run is one pair of a training seed and an evaluation seed. The training seed fixes the randomness of training and hence the weights of the checkpoint. The evaluation seed fixes the candidates that the sampler draws from these weights and the subsample of test pathways on which path@1 is measured, so the spread covers the whole measurement and not the decoder alone. Every run uses the schedule of Appendix K.4, the decoding budgets of Appendix K.5, and the evaluation protocol of Appendix L.1. The spread of Arrow Flow Matching is at most 0.0013

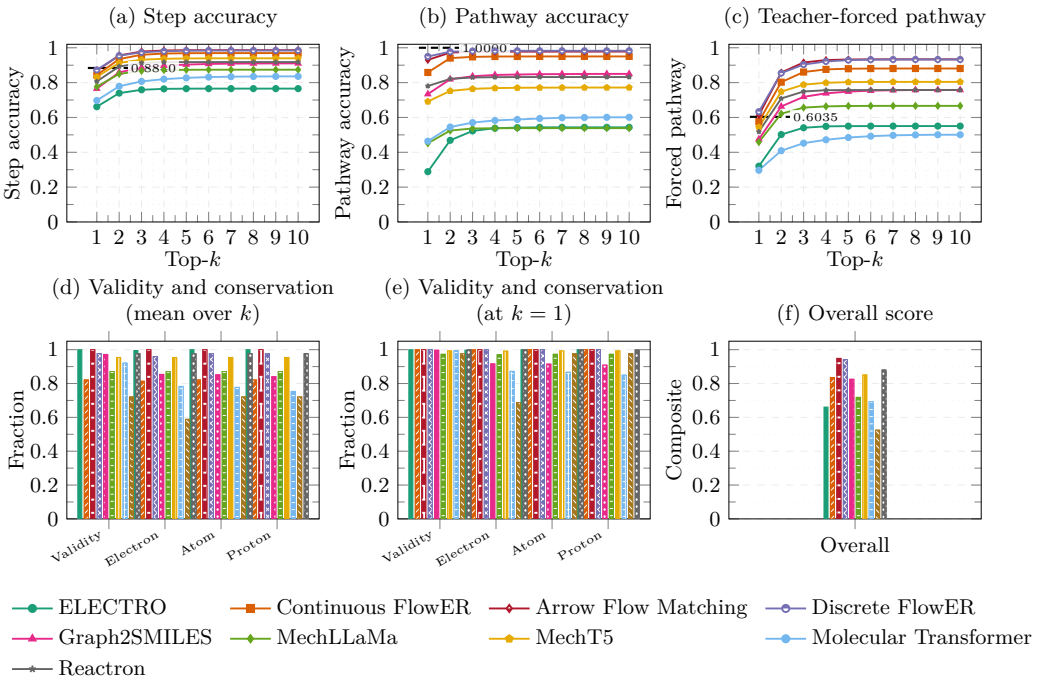


Figure 15: All nine models at the smallest width, the nano rung of the ladder at roughly 1.3M parameters, under the one protocol that every model can fill, with candidates drawn by sampling, ranked by frequency and scored against the recorded product. The key gives each model its short name. **(a)–(c)** Accuracy against the candidate budget k, with the dashed segment at $k = 1$ marking the ceiling that branching in the data imposes on a deterministic decoder. **(d)–(e)** Validity and the three conservation checks, averaged over k and at $k = 1$. **(f)** The composite score. Arrow Flow Matching is the only model at 1.0000 on all four checks of (d) while matching the best accuracy curves, and its composite score in (f) is the highest in the study, at this width and at every other. ELECTRO reaches validity 1 by construction but falls well below on accuracy, and Continuous FlowER is the weakest of the nine on validity. The figure shows one independent run of every model, so its numbers differ from the means over 10 runs in Table 2.

on every accuracy column except path@1 and zero on all four checks, and the differences between the top-1 step accuracies of the three flow models exceed their standard deviations.

L.3 The other four regimes at the smallest width

Figures 22–25 repeat Figure 15 under the four regimes that the main text does not report. The four guarantee panels of Arrow Flow Matching read exactly 1 in every one of them, as validity and conservation are properties of the emitted molecule that no decoder or ranking rule can change. Its accuracy is insensitive to the choice too. The beam decoder, which its two closest competitors lack, adds about half a point at $k = 10$ and nothing at $k = 1$.

L.4 The width sweep under the other regimes

Figures 26–29 repeat Figure 16 under the same four regimes, and the picture is unchanged. The validity and the electron, atom and proton conservation of Arrow Flow Matching are flat at exactly 1 across the whole 15× range of capacity under every regime, its accuracy is flat as well, and its composite score at the smallest width stays above every point that any other model reaches at any width.

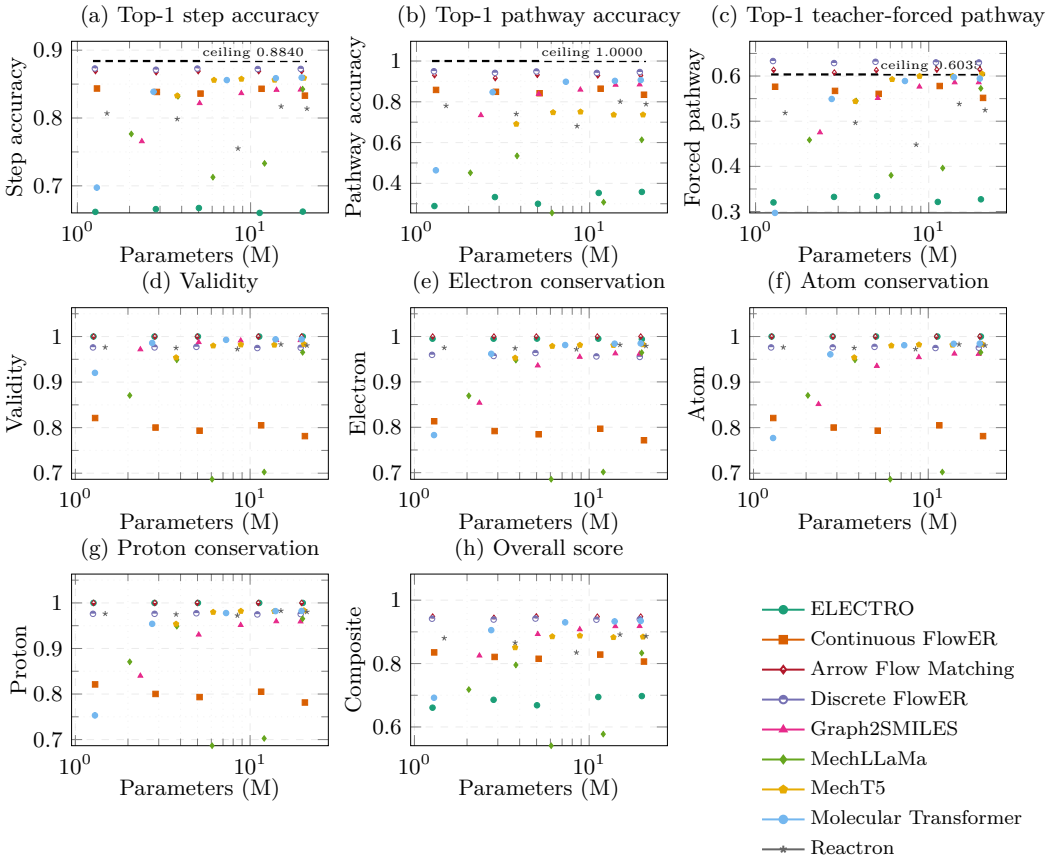


Figure 16: Every model at five widths spanning 1.2–21M parameters, under sampling with frequency ranking and scored against the recorded product. Panels **(d)–(g)** are validity and the three conservation checks, and panel **(h)** is the composite score. The four guarantee panels of Arrow Flow Matching are exactly 1 across the whole range, and its accuracy panels are flat. The validity and the electron conservation of Continuous FlowER *fall* as it grows. ELECTRO is the only other model that reaches 1 on a conservation panel.

M GENERALISATION

This appendix describes the three shifts of Q5 in §6 and reports their full results.

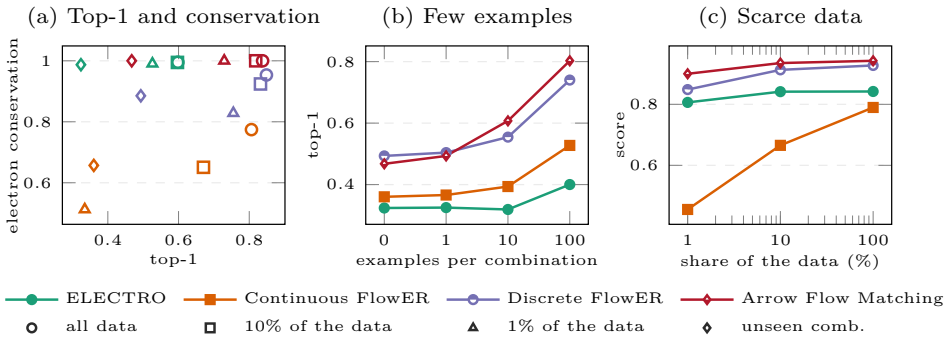


Figure 17: The models of Table 4 under the shifts, one point or line per model. (a) Top-1 against electron conservation. Arrow Flow Matching keeps full conservation under every shift. (b) Top-1 on unseen combinations against the examples of each. (c) Score against the share of the training data. Arrow Flow Matching has the highest score at every share.

M.1 THE MOVE-SIGNATURE SPLIT

The move signature of a step is the multiset of the moves of its decomposition (Appendix H), each move written as its type and the elements of its two atoms. The two atoms of a homolysis or a colligation are unordered, so their elements are sorted. Signatures are grouped by the decade of their number of training steps, and a tenth of each group is held out in a seeded order. A signature is held out only if each of its moves still occurs in a kept training signature, and a signature of more than 1% of the training steps is always kept, so that no single signature dominates. Every training and validation step with a held-out signature is removed. The test split gives two parts of 10,000 steps each, drawn from the steps with at least one move. The *seen part* holds steps with a kept signature, and the *held-out part* steps with a held-out signature. Figure 18 and Table 13 show that the split is meaningful. Rare and common signatures are both held out, and every held-out step is a new combination of moves that each occur in training. Most held-out steps are one move away from a training signature. The reactants of both parts are about as similar to the training reactants, so the held-out part differs from training in its mechanism and not in its molecules.

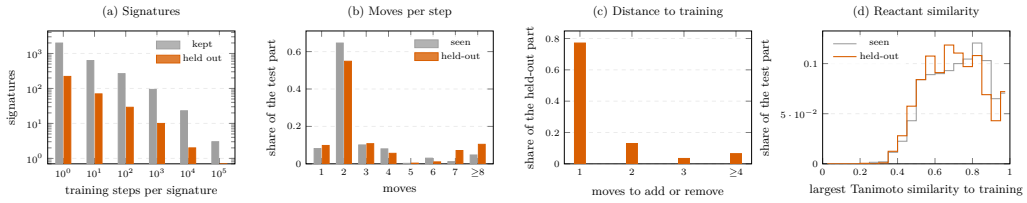


Figure 18: The move-signature split. (a) Training signatures by their number of training steps, kept and held out. (b) Moves per step in the seen and the held-out part of the test split. (c) For each held-out test step, the fewest moves to add or remove to turn its signature into a training signature. (d) The largest Tanimoto similarity (Morgan fingerprints, radius 2) of 2,000 reactants of each part to the training reactants.

M.2 TRAINING AND EVALUATION

Every model is trained at the smallest width. Discrete FlowER and Arrow Flow Matching take the recipe of Arrow Flow Matching, data-parallel on four GPUs with a batch of 192 per GPU, a peak learning rate of 8.5×10^{-4} and 30 epochs. ELECTRO and Continuous FlowER keep their own settings (Table 12) for the same number of epochs. For ten examples, each model retrained on the

Table 13: Counts of the move-signature split. The effective number of signatures is the exponential of the entropy of the signature distribution of a part. A reactant occurs in training when its canonical SMILES without atom map is a training reactant.

quantity	value
training steps	1,933,750
signatures with at least one move	3,333
held-out signatures	332
training steps removed	88,027 (4.6%)
validation steps removed	995 of 21,723
test steps with a held-out signature	10,095
steps predicted, seen and held-out part	10,000 and 10,000
distinct signatures, seen and held-out part	465 and 144
effective number of signatures, seen and held-out part	59 and 21
held-out steps whose every move occurs in a kept training signature	100.0%
reactant occurs in training, seen and held-out part	4.2% and 4.6%
median largest similarity to training, seen and held-out part	0.729 and 0.700

split is fine-tuned on $k \in \{1, 10, 100\}$ training steps of each held-out signature together with 20,000 replayed seen steps, for 20 epochs at a constant learning rate of 10^{-4}. The checkpoint is chosen by the loss on the validation steps with a held-out signature, which are disjoint from the test steps. For 1% of the data, the four models train on the same seeded 1% and 10% of the training split for 300 and 100 epochs. Discrete FlowER and Arrow Flow Matching keep the effective batch of their recipe on one GPU through gradient accumulation. Neither shift uses early stopping, and the lowest validation loss chooses the checkpoint. Every model is decoded as in Table 2 and scored on both parts of the move-signature split. The full-data checkpoints are those of Table 2.

M.3 Results

Table 14 gives the models retrained on the move-signature split next to their full-data counterparts. *Retention* is the held-out top-1 as a share of the seen top-1. Held-out steps are harder for every model even when their signatures are seen in training, and the shift adds a drop of a similar size to both discrete heads. Many reactants react in several recorded ways, and 39% of the held-out steps have such a reactant. *Any product* counts a candidate as correct when it is any product recorded for the step's reactant. Under this reading, most held-out errors are another recorded reaction of the same reactant.

Table 14: Models trained on the full training split and retrained on the move-signature split, on the seen and the held-out part. *Ret.* is the retention and *any* the any-product top-1, both on the held-out part. The last four columns are validity and electron, heavy-atom and proton conservation on the held-out part, as in Table 2. Coloured as in Table 2.

	trained on the full split				trained on the move-signature split							
model	seen	held-out	ret.	any	seen	held-out	ret.	any	valid.	elec.	atom	prot.
ELECTRO	0.598	0.440	0.736	0.496	0.605	0.324	**0.535**	0.427	**1.000**	**0.987**	**1.000**	**1.000**
Continuous FlowER	**0.806**	**0.736**	**0.912**	**0.930**	**0.775**	**0.360**	0.465	**0.657**	**0.670**	0.657	**0.670**	**0.670**
Discrete FlowER	**0.849**	**0.763**	**0.899**	**0.976**	**0.852**	**0.493**	**0.579**	**0.846**	**0.936**	**0.885**	**0.936**	**0.936**
Arrow Flow Matching	**0.838**	**0.854**	**1.019**	**0.968**	**0.854**	**0.468**	**0.548**	**0.824**	**1.000**	**1.000**	**1.000**	**1.000**

Figure 30 follows the fine-tuning on k examples. On all held-out steps, Arrow Flow Matching gains more from ten and a hundred examples than Discrete FlowER. On the held-out steps whose reactant has one recorded product, where no choice between recorded reactions is involved, the two discrete heads adapt at almost the same rate. The gain of Arrow Flow Matching thus lies mostly in choosing, among the recorded reactions of a reactant, the one that the new examples show. Table 17 gives the models trained on a share of the training split. The entrywise head reaches a slightly higher top-1, and its validity and conservation fall as the share shrinks, while those of Arrow Flow Matching stay exact.

Table 15: Top-1 step accuracy of the four models whose mechanism gives arrows, at the smallest width, as trained and fine-tuned on the curated training steps. The random test steps are the random part of the analysis subset. The curated test steps come from PMechDB (PM) and RMechDB (RS, RC), and the last column pools them. Each column gives its number of steps. Coloured as in Table 2.

	random (19,772)		PM (294)		RS (346)		RC (136)		curated (776)	
fine-tuned	no	yes	no	yes	no	yes	no	yes	no	yes
ELECTRO	0.666	0.660	**0.357**	**0.418**	0.000	0.000	0.000	0.000	0.135	0.159
Continuous FlowER	**0.836**	**0.833**	0.323	0.337	**0.749**	**0.746**	**0.551**	**0.544**	**0.553**	**0.555**
Discrete FlowER	**0.868**	**0.865**	**0.435**	**0.514**	**0.815**	**0.824**	**0.735**	**0.824**	**0.657**	**0.706**
Arrow Flow Matching	**0.867**	**0.864**	**0.442**	**0.476**	**0.792**	**0.812**	**0.772**	**0.757**	**0.656**	**0.675**

N INTERPRETABILITY

This appendix describes how the experiments of Q4 in §6 were performed, reports the fine-tuning, and gives two further analyses of the mechanism that Arrow Flow Matching computes. All results are for the smallest width.

Arrows drawn by chemists. The test split contains 294 steps taken from the curated part of PMechDB (Tavakoli et al., 2024) and 482 taken from RMechDB (Tavakoli et al., 2023). Each database step carries the arrows a chemist drew. We carry these arrows onto the atoms of the matching test step and keep the step when they give its product. This keeps 273 polar and 455 radical steps. Two half-arrows that leave one bond are a HOMOLYSIS, and two that reach one bond are a COLLIGATION, so a radical step is compared in the moves of §5. Two sets of arrows are equal when they are equal after atoms the reactant cannot tell apart, such as the hydrogens of a methyl group, are identified.

Reading a mechanism as arrows. The moves of Arrow Flow Matching are arrows. A BOND→LONE move onto an atom followed by a LONE→BOND move from that atom is the single bond-shift arrow a chemist draws (Appendix F.2). The electron path of ELECTRO is a chain of arrows that alternately form and break bonds. For the two FlowER models, each update of the chain that produced the top-1 candidate is factorised into arrows as the move bridge factorises a step (Appendix H). The chain gives arrows only when every update conserves electrons. Figure 19 draws two steps of Arrow Flow Matching beside the arrows of the chemist.

Sampling mechanisms. For the columns *first* and *drawn* of Table 3, each model draws 100 mechanisms for every test step with its own sampler. Mechanisms with equal arrows, after the atoms the reactant cannot tell apart are identified, count as one, and each keeps the best score it was drawn with. The chemist's mechanism is *first* when it is drawn and no other mechanism scores higher. The FlowER models give no probability over arrows, so they take no part.

Fine-tuning. The curated steps are 16,928 of the 1,933,750 training steps. Each of the four models continues from its checkpoint at the smallest width with a fresh optimiser. It is trained on the curated training steps of both databases together with 20,000 other training steps drawn at random, so that it keeps seeing its original task. A tenth of the curated steps is held out for validation, taken in whole mechanisms. Two steps belong to one mechanism when the product of one is the reactant of the other or when they share their reactant, so no held-out step has a neighbour in training. One recipe serves all four models, a learning rate of 10^{-4} with 200 warmup updates and a cosine decay, batches of 96 steps and at most 30 epochs. The checkpoint with the best top-1 step accuracy on the held-out steps is kept, and no test step is used. The fine-tuned model is decoded and read as arrows exactly as the model it started from.

Table 15 gives the top-1 step accuracy before and after fine-tuning. Fine-tuning costs every model at most 0.006 on the random test steps, and it raises the accuracy of every model on the curated test steps. Discrete FlowER gains most and ends 0.031 above Arrow Flow Matching on the curated steps. This small margin is the whole price of the arrows. The chain of Discrete FlowER gives arrows on under 5% of the polar steps and under 1% of the radical ones, and its candidates are valid and conserve electrons less often (Table 2). Every candidate of Arrow Flow Matching is valid, conserves every count, and comes with the arrows of the mechanism that produced it.

Editing the first arrow. A mechanism built arrow by arrow can be edited and continued. On 5000 random test steps with at least one move, the first move of Arrow Flow Matching is replaced by its

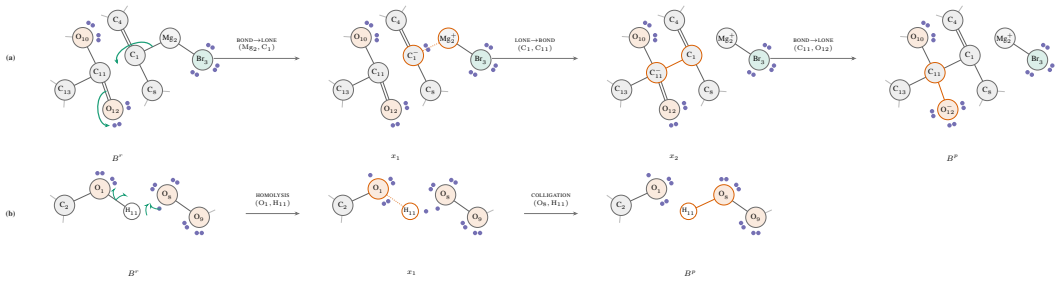


Figure 19: Two test steps whose arrows chemists drew, one per row. The first panel is the reactant with the chemist's arrows (green), and the following panels are the top-1 trajectory of Arrow Flow Matching, one panel per move. On both steps the arrows of the model equal the chemist's. (a) A polar step with two arrows of the chemist. The first, from the carbon–magnesium bond to the carbonyl carbon, is a bond shift, which the model makes as a BOND→LONE move and a LONE→BOND move from the shared carbon. The second, from the carbonyl bond onto the oxygen, is the third move. (b) A radical step. The chemist's four half-arrows are the HOMOLYSIS of the O–H bond and the COLLIGATION of the hydrogen with the other oxygen radical, the two moves of the model. Atoms and bonds that change are orange, a bond that breaks is dotted, and grey stubs are bonds to atoms not drawn.

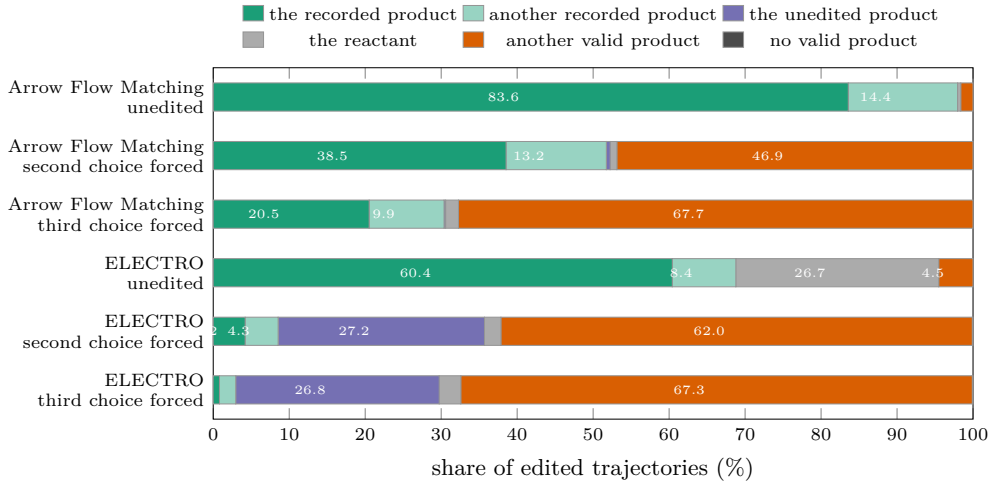


Figure 20: Where a mechanism ends when its first decision is forced to the second or third choice of the model, on 5000 random test steps. Each trajectory counts in the first class it fits, in the order of the key. For Arrow Flow Matching the first decision is a move, and for ELECTRO it is the atom its electron path starts from.

second or third most probable admissible move, and the rest of the trajectory is decoded greedily. For ELECTRO the atom its electron path starts from is replaced in the same way. Figure 20 shows where the edited mechanisms end. With the second choice forced, an edited trajectory of Arrow Flow Matching ends in another product that the data record for the same reactant on 13.2% of the steps, against 4.3% for ELECTRO, and every edited trajectory of Arrow Flow Matching ends in a valid molecule. Figure 31 draws one edited mechanism.

Where the uncertainty sits. Before each decision the move head is a distribution over the admissible moves and the stop, and its entropy measures how unsure the model is about the next arrow. We read it along the top-1 trajectory of each random test step, together with its share on the moves between atoms of the reaction centre, the atoms whose row the recorded step changes. For Discrete FlowER the analogue is the summed entropy of its entry heads along the chain behind its top-1 candidate. Figure 32 shows that on reactants the data resolve to several products the uncertainty of Arrow Flow Matching sits on the first decision and on the reaction centre, while the later moves are nearly certain. Figure 21 draws one such step.

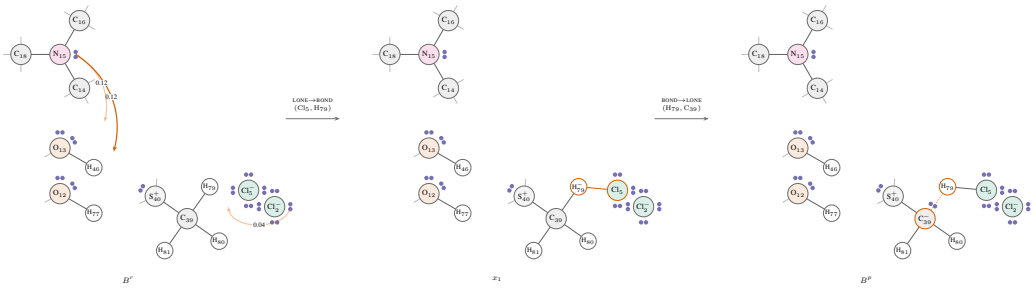


Figure 21: A test step whose reactant the data resolve to several products. The first panel draws on the reactant the three most probable first moves of Arrow Flow Matching, the chosen one solid and the others faded, each labelled with its probability under the move head. The following panels are the top-1 trajectory, whose product is the recorded one. The entropies of its decisions are 3.09, 0.00 and 0.00 nats, so the uncertainty sits on the first arrow. Atoms and bonds that change are orange, and grey stubs are bonds to atoms not drawn.

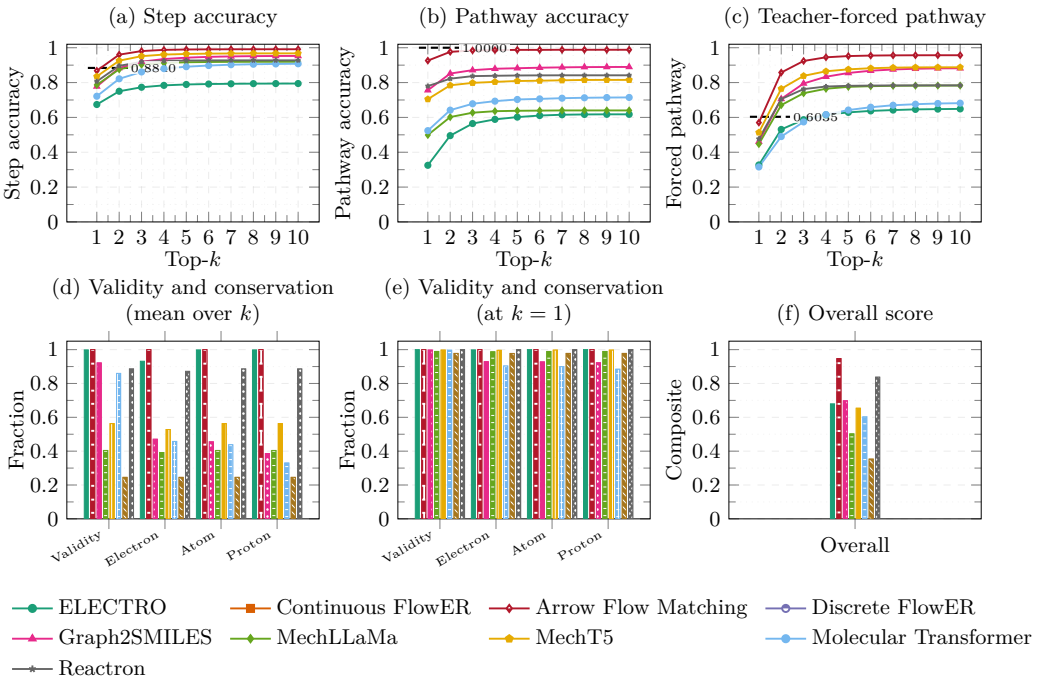


Figure 22: The comparison at the smallest width under beam search ranked by trajectory score, panel by panel as in Figure 15. Continuous and Discrete FlowER are absent (Table 11).

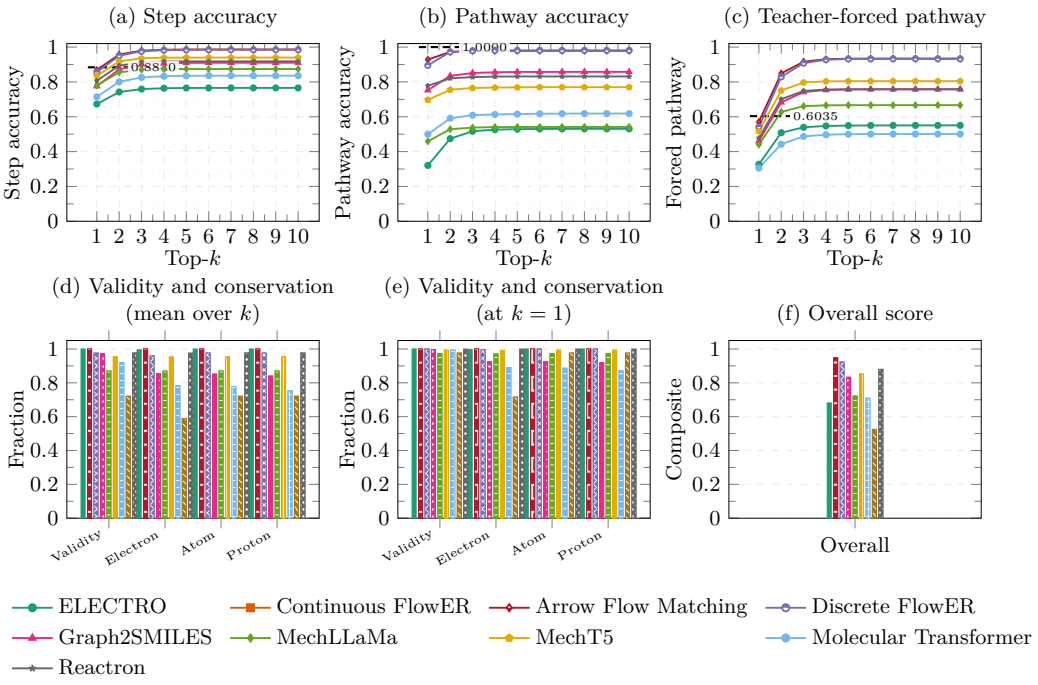


Figure 23: The comparison at the smallest width under sampling ranked by trajectory score. Continuous FlowER is absent (Table 11). Read against Figure 15, this isolates what the ranking rule is worth at a fixed set of draws.

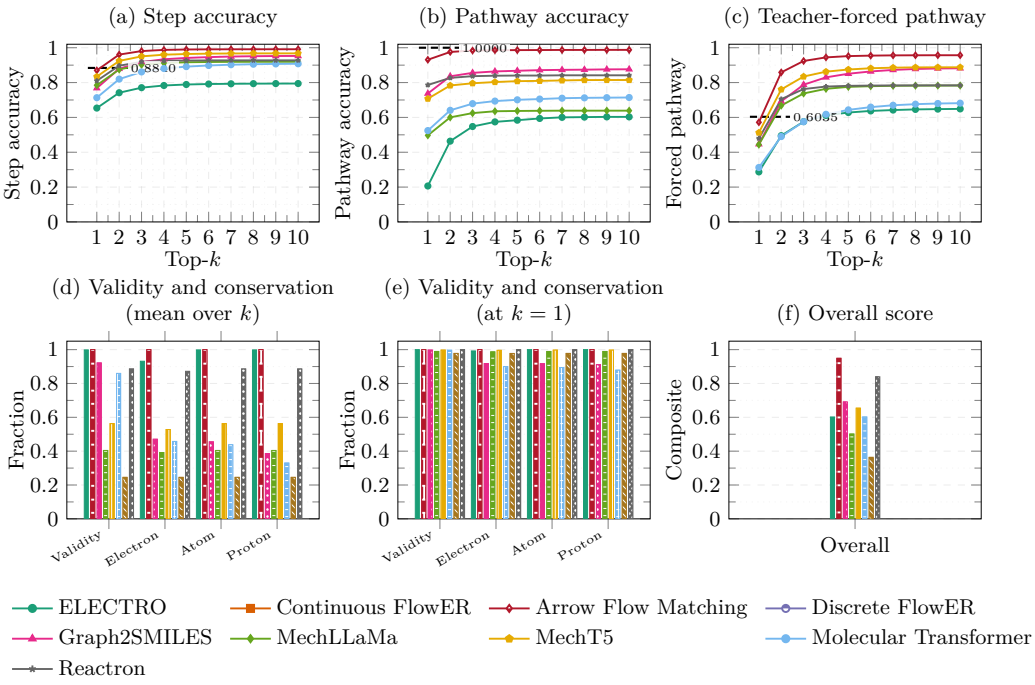


Figure 24: The comparison at the smallest width under beam search ranked by marginal probability over trajectories. Read against Figure 22, this isolates the ranking rule at a fixed set of candidates.

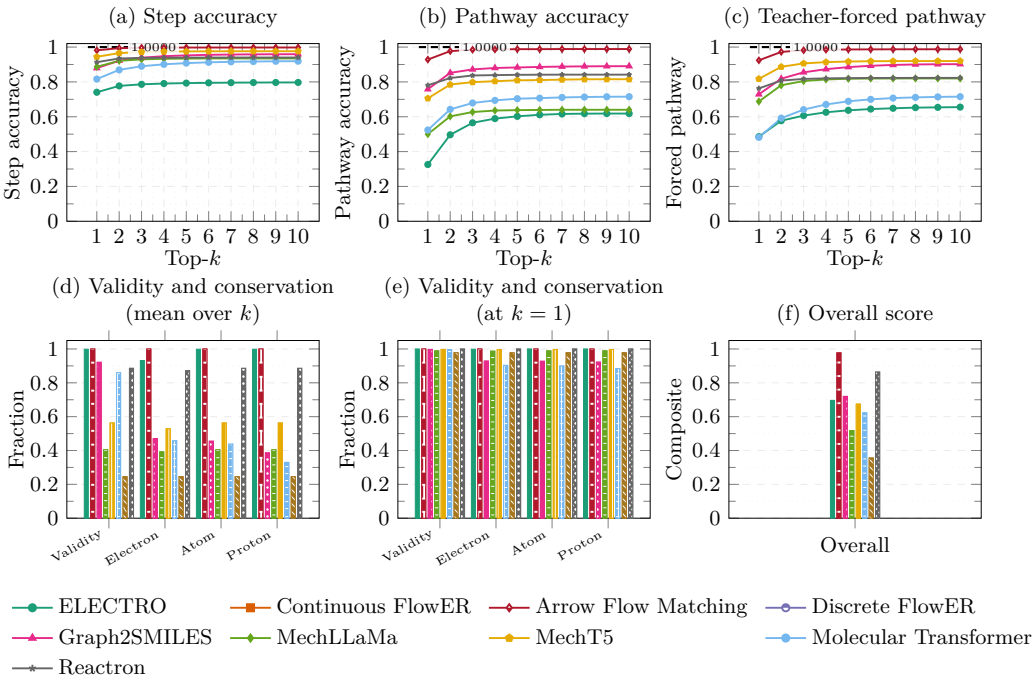


Figure 25: The comparison at the smallest width under beam search scored against the branch-aware ground truth. The predictions and the ordering are those of Figure 22, and only what counts as correct changes, so the validity and conservation panels repeat that figure exactly.

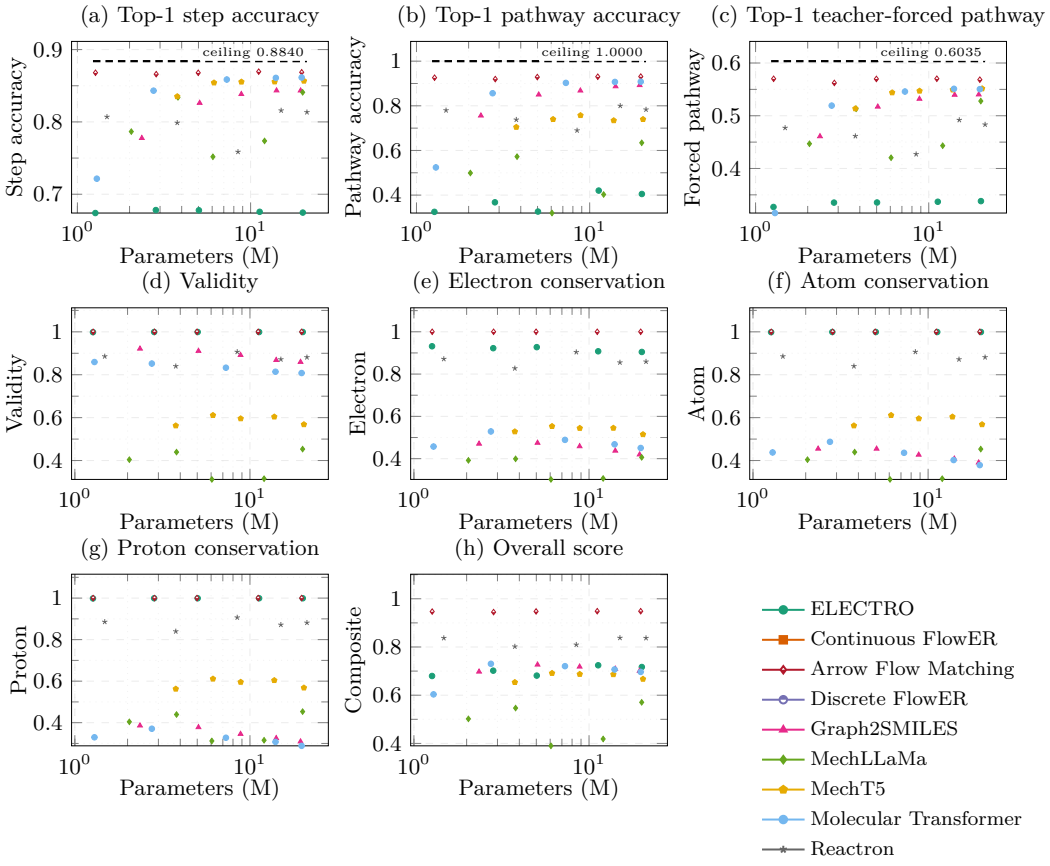


Figure 26: The width sweep under beam search ranked by trajectory score.

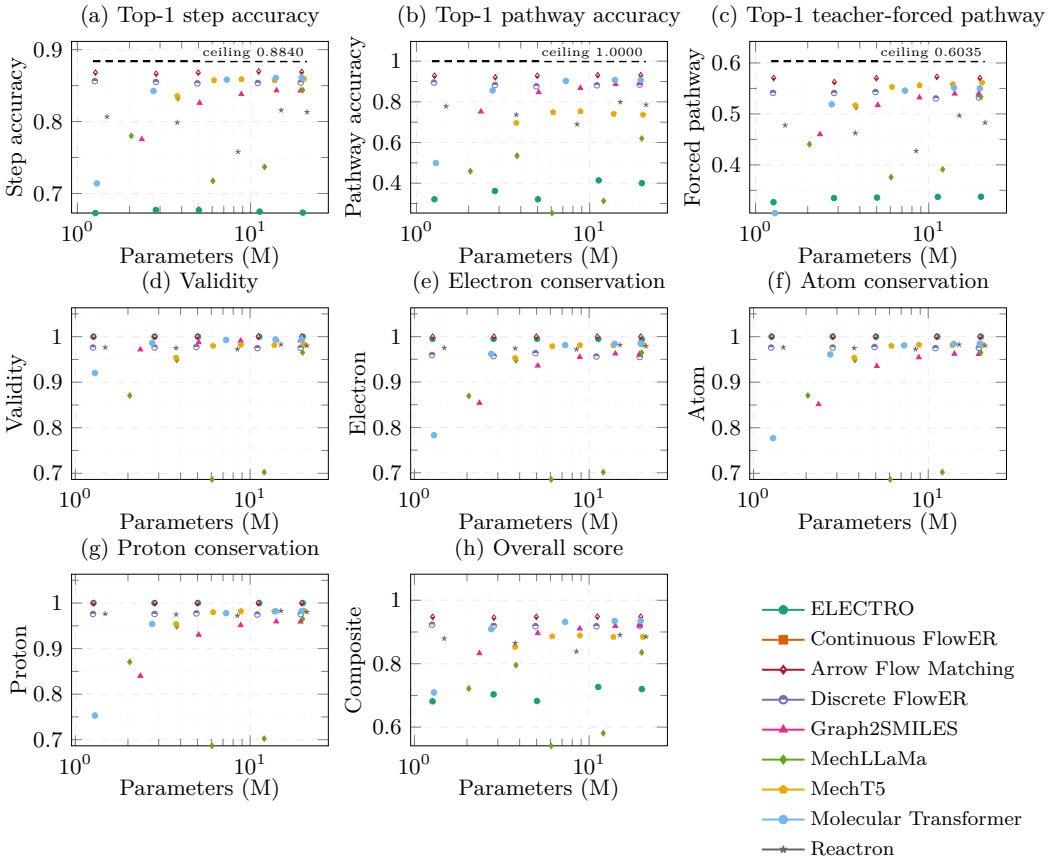


Figure 27: The width sweep under sampling ranked by trajectory score.

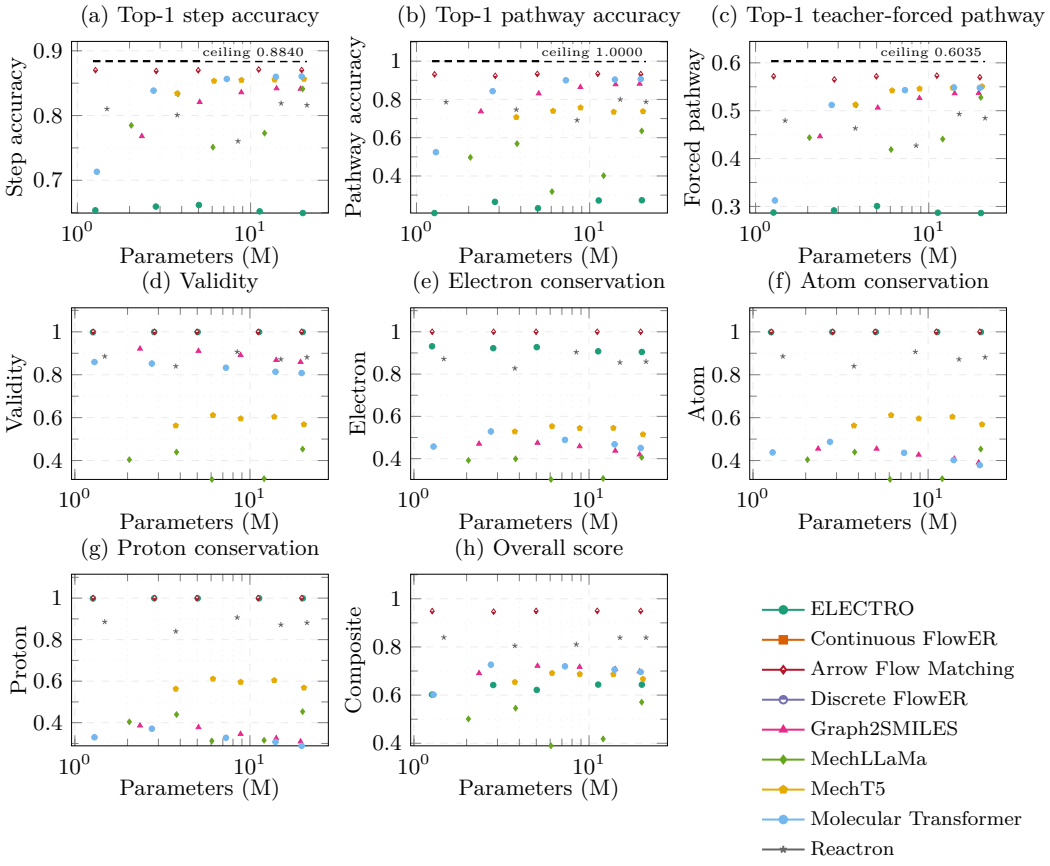


Figure 28: The width sweep under beam search ranked by marginal probability over trajectories.

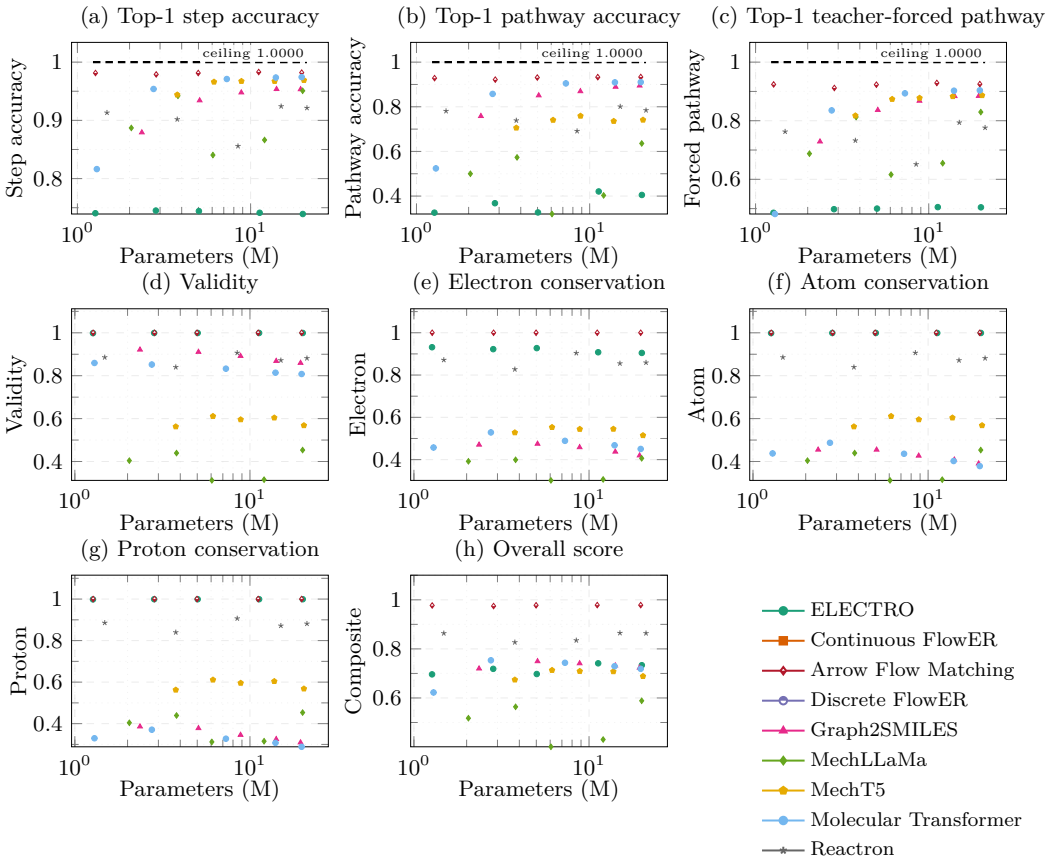


Figure 29: The width sweep under beam search scored against the branch-aware ground truth.

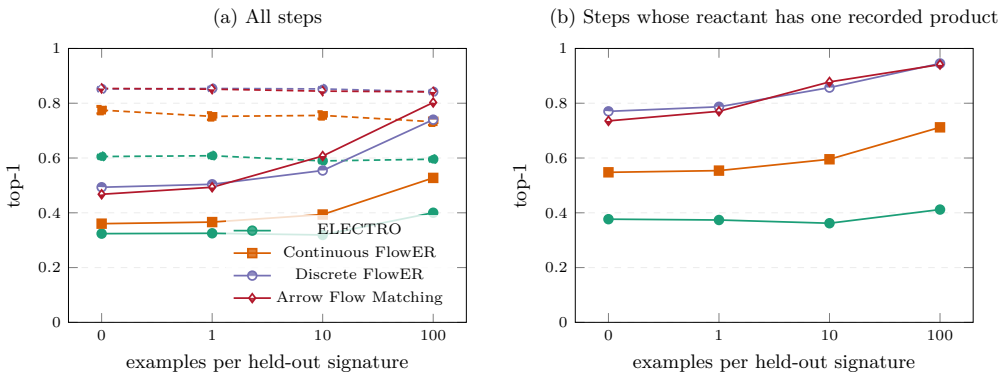


Figure 30: Top-1 after fine-tuning on k training steps of each held-out signature, on the held-out part (solid) and the seen part (dashed). (a) All steps. (b) The held-out steps whose reactant has one recorded product.

Table 16: Table 2 with the sample standard deviation over the 10 runs as a subscript. The colouring ranks the means, as in Table 2.

model	params	top-1	top-3	top-5	top-10	path@1	valid.	elec.	atom	prot.	score
Molecular Transformer	1.29 M	0.6446 ±0.0469	0.7540 ±0.0503	0.7758 ±0.0495	0.7855 ±0.0488	0.3770 ±0.0776	0.9036 ±0.0223	0.7303 ±0.0536	0.7237 ±0.0550	0.6939 ±0.0598	0.6284 ±0.0602
Graph2SMILES	2.36 M	0.7347 ±0.0132	0.8549 ±0.0128	0.8745 ±0.0116	0.8841 ±0.0106	0.6858 ±0.0240	0.9584 ±0.0044	0.8066 ±0.0183	0.8036 ±0.0186	0.7894 ±0.0201	0.7883 ±0.0158
MechT5	3.77 M	0.8414 ±0.0005	0.9404 ±0.0004	0.9474 ±0.0003	0.9482 ±0.0004	0.7152 ±0.0126	0.9637 ±0.0012	0.9628 ±0.0012	0.9637 ±0.0012	0.9637 ±0.0012	0.8646 ±0.0036
MechLLaMa	2.05 M	0.7895 ±0.0127	0.8824 ±0.0146	0.8887 ±0.0145	0.8892 ±0.0145	0.5084 ±0.0232	0.8845 ±0.0135	0.8830 ±0.0138	0.8845 ±0.0135	0.8845 ±0.0135	0.7484 ±0.0166
Reactron	1.48 M	0.8147 ±0.0031	0.9198 ±0.0050	0.9252 ±0.0052	0.9254 ±0.0052	0.7993 ±0.0113	**0.9814** ±0.0017	**0.9803** ±0.0019	**0.9814** ±0.0017	**0.9814** ±0.0017	**0.8898** ±0.0046
ELECTRO	1.27 M	0.6622 ±0.0015	0.7590 ±0.0011	0.7653 ±0.0011	0.7656 ±0.0011	0.3159 ±0.0348	**1.0000** ±0.0000	**0.9954** ±0.0001	**1.0000** ±0.0000	**1.0000** ±0.0000	0.6754 ±0.0190
Continuous FlowER	1.30 M	**0.8432** ±0.0018	**0.9590** ±0.0004	**0.9682** ±0.0002	**0.9688** ±0.0002	**0.8609** ±0.0075	0.8144 ±0.0009	0.8067 ±0.0009	0.8144 ±0.0009	0.8144 ±0.0009	0.8323 ±0.0019
Discrete FlowER (§3)	1.26 M	**0.8719** ±0.0008	**0.9732** ±0.0007	**0.9830** ±0.0007	**0.9847** ±0.0008	**0.9491** ±0.0086	**0.9767** ±0.0017	0.9553 ±0.0017	**0.9767** ±0.0017	**0.9767** ±0.0017	**0.9408** ±0.0030
Arrow Flow Matching (§5)	1.27 M	**0.8690** ±0.0013	**0.9777** ±0.0011	**0.9851** ±0.0006	**0.9855** ±0.0006	**0.9336** ±0.0062	**1.0000** ±0.0000	**1.0000** ±0.0000	**1.0000** ±0.0000	**1.0000** ±0.0000	**0.9491** ±0.0018

Table 17: Models trained on 1%, 10% and all of the training split. *One product* keeps the held-out steps whose reactant has one recorded product. The last four columns are validity and electron, heavy-atom and proton conservation on the seen part, as in Table 2. Coloured as in Table 2.

		top-1			seen part			
model	data (%)	seen	held-out	one product	valid.	elec.	atom	prot.
ELECTRO	1	0.526	0.404	0.388	**1.000**	**0.990**	**1.000**	**1.000**
Continuous FlowER	1	0.335	0.297	0.357	0.535	0.512	0.535	0.535
Discrete FlowER	1	0.755	0.621	0.812	0.914	0.828	0.914	0.914
Arrow Flow Matching	1	0.729	0.586	0.721	**1.000**	**1.000**	**1.000**	**1.000**
ELECTRO	10	0.597	0.428	0.444	**1.000**	**0.995**	**1.000**	**1.000**
Continuous FlowER	10	0.670	0.626	0.701	0.666	0.651	0.666	0.666
Discrete FlowER	10	**0.832**	0.704	**0.949**	**0.963**	0.924	**0.963**	**0.963**
Arrow Flow Matching	10	0.818	**0.801**	0.928	**1.000**	**1.000**	**1.000**	**1.000**
ELECTRO	100	0.598	0.440	0.461	**1.000**	**0.995**	**1.000**	**1.000**
Continuous FlowER	100	0.806	0.736	0.916	0.783	0.774	0.783	0.783
Discrete FlowER	100	**0.849**	**0.763**	**0.974**	**0.973**	0.953	**0.973**	**0.973**
Arrow Flow Matching	100	**0.838**	**0.854**	**0.966**	**1.000**	**1.000**	**1.000**	**1.000**

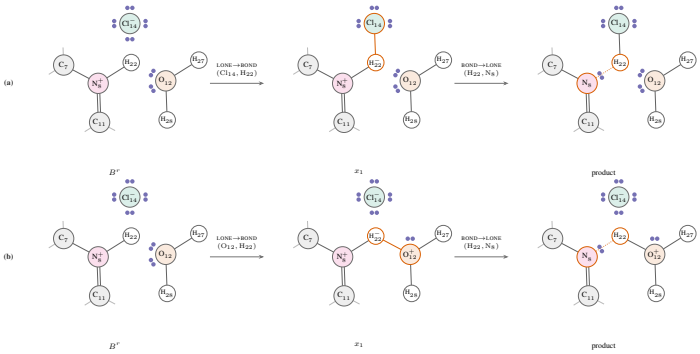


Figure 31: A test step whose reactant the data resolve to two products. (a) The greedy trajectory of Arrow Flow Matching reaches the recorded product. (b) With its first move replaced by the second most probable admissible move, the model completes the edited mechanism to the other recorded product. Atoms and bonds that change are orange, and grey stubs are bonds to atoms not drawn.

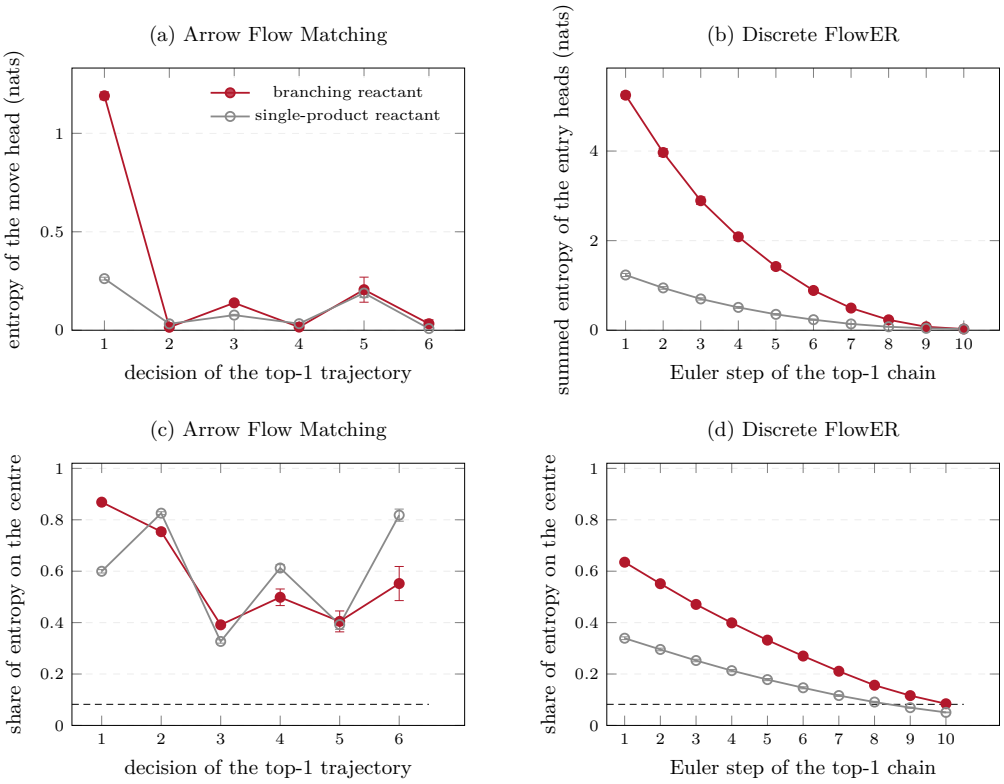


Figure 32: Uncertainty along the chain that produced the top-1 candidate, on 19,772 random test steps, with 95% bootstrap intervals, for reactants with several recorded products and with one. (a) Entropy of the move head of Arrow Flow Matching before each decision. (b) Summed entropy of the entry heads of Discrete FlowER at each Euler step. (c, d) The share of the entropy on the moves or entries that touch the reaction centre, which puts both models on one scale. The dashed line is the share for uncertainty spread evenly over all atom pairs.