GFLOWNETS BUILD FEASIBLE DESIGNS FOR DIRECTED EVOLUTION CAMPAIGNS

Anonymous authors
Paper under double-blind review

ABSTRACT

In a directed evolution campaign the budget, not the size of the search space, is the binding constraint: a few hundred assays, a plate of variants a round, to improve an enzyme or a binder. When a hard feasibility constraint applies, the pipelines used today — a genetic algorithm, machine-learning-guided design (MLDE), CMA-ES over a continuous relaxation — cannot represent which sequences are buildable; they draw candidates and subsequently reject the illegal ones, which wastes budget on unmakeable variants and, when feasible designs are scarce, leaves the method unable to run at all. Building each design move by move under a mask that permits only legal continuations removes the bottleneck entirely, since every proposal is then feasible by construction. We present EvoGFN, a GFlowNet that constructs designs over the construction graph rooted at a campaign's parent, enforcing feasibility by masking each move, and test it at wet-lab budgets against those pipelines. Its mean fitness is the highest on every task in our synthetic suite, decisively where few sequences are feasible, though a CMA-ES baseline reaches the same ceiling on the task where both saturate; on the two empirical landscapes it leads narrowly. Throughout it proposes only feasible designs, where the pipelines it is compared against spend much of their budget on designs that cannot be built. The same advantage holds on a real, structure-derived constraint, a folding-stability predicate on the GB1 landscape thresholded on predicted $\Delta\Delta G$, where a proximal baseline matches it on fitness only at the tightest threshold. Controls separate mask from policy and trace the dependence on predictor quality: porting the mask to a plain genetic algorithm gives it full feasibility and a fitness gain, untraining our own policy costs it little except where the feasible set is small, and injecting noise into the fitness predictor that ranks candidates before any assay is spent erodes the lead only where that predictor was informative. The advantage narrows where the feasibility predicate is simple enough to admit an exact projection — a dynamic program that recovers the best design under an additive decoding score, though not under the true fitness — against which the learned sampler keeps a small edge on some tasks and ties on others. A masked sampler gains most where the constraint is strict, and where an exact decoder can be built it closes much of that gain. Where the constraint couples every position, a local mask reaches almost none of the legal designs, and we leave that regime as an open frontier.

1 INTRODUCTION

Directed evolution improves a protein by an iterated wet-lab search: propose a handful of variants, build and assay them, keep the best, and repeat. Each round is a plate synthesised and assayed at the bench within a mutation budget of the current best, and each round blocks the next, so a useful enzyme or binder has to be reached inside a few hundred measurements. From an optimization perspective the loop is active learning under a hard query budget — and much of the sequence space holds nothing that can be built, settled before fitness is ever measured.

A method that builds on the constraint, instead of paying draw after draw to rediscover which candidates are legal, reaches better variants in fewer rounds. When only some sequences can be built, a feasibility constraint decides which ones count as designs, and the campaign is scored by the best fitness it reaches inside that set. Such requirements are an inherent part of building real proteins: a synthesis route that must assemble the sequence, an adjacency rule two cysteines must

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.

satisfy, a folding classifier that must call the variant expressible. The pipelines a laboratory runs today meet these requirements the same way, and only after a candidate has been proposed: a genetic algorithm, machine-learning-guided design (Yang et al., 2019; Wittmann et al., 2021), CMA-ES over a continuous relaxation (Hansen & Ostermeier, 2001), and the proximal and adaptive methods that refine them all draw a candidate from a distribution over whole sequences and test feasibility only after the draw. On a constrained task that test fails often, and where the constraint is strict most of a plate can be variants that make no sense to build; each such failure spends one of a few hundred assays on a design the laboratory discards, and where feasible designs are scarce enough the pipeline can fail to fill a plate at all.

A constructive sampler solves this by construction: it never draws and rejects. It assembles a design one move at a time, filters each move by a mask, and emits no illegal object. GFlowNets (Bengio et al., 2021; Jain et al., 2022) fit the proposal step of a campaign well: the policy samples in proportion to reward, so a plate fills with a diverse batch of high-reward candidates instead of collapsing onto one, and the per-move mask makes every design it emits feasible by construction. We cast the proposal step of a directed-evolution campaign as a constrained sampling problem, feasibility enforced while the design is built, not by testing a finished draw and rejecting it, and ask whether a sampler built on it spends a fixed assay budget better than the pipelines protein engineers run today.

Here we present EvoGFN, a GFlowNet that constructs designs over the construction graph rooted at a campaign's parent, enforcing the feasibility constraint at every move so that every design it proposes is buildable and no assay is ever spent on a variant the laboratory would only discard. States are sequences within a mutation budget of the parent, moves are single substitutions, and the mask admits only the substitutions that keep the design feasible. The policy is deliberately small, a two-layer trunk of sixty-four hidden units refit from a few hundred assays, and we run it inside the protocols the pipelines above use, at the budgets those protocols specify: a plate a round, three rounds in one protocol and eight in the other, 396 and 384 assays respectively, rerooting on the best measurement between rounds. Every arm in every comparison received the same parent, constraint, rounds, and assay budget, so no result here is bought with extra experiments.

Each result below isolates a single claim. Our sampler leads every published pipeline on every task in our synthetic suite (§3.1), decisively where few sequences are feasible and narrowly on the two unconstrained landscapes, tying only where both reach the ceiling, and it spends every assay on a feasible design (a feasible fraction of 1.000 against 0.295 for a plain genetic algorithm on the ALDE protocol) where the pipelines with no way to fix an infeasible draw spend most of theirs on discards (§3.2). We run two controls to locate the win: feasibility by construction is a portable mechanism that lifts a plain genetic algorithm to full feasibility and higher fitness, while learning pays off where few sequences are feasible; given four times the proposals an untrained policy on the same masked graph catches up on the unconstrained landscapes (§3.4). An intervention on the fitness predictor traces this to predictor quality: degrading the predictor costs the sampler its edge in proportion to how well it predicted fitness (§3.5). The advantage carries to a real, structure-derived constraint, a folding-stability threshold on the GB1 landscape set on predicted $\Delta\Delta G$ (§3.3): decisive where the gate leaves a set large enough to search, and holding diversity where it is tightest, so the result is not an artefact of the Ehrlich generator. The advantage narrows where the constraint is simple enough for an exact projection: a dynamic program recovers the best design under an additive decoding score, though not the true fitness, and the learned sampler keeps only a small margin over it (§3.6). A masked sampler gains where the feasible set is small, and an exact decoder closes much of that gain where one can be built. Where the predicate couples every position — the regime our support study builds (Appendix D) — a local mask reaches almost none of the feasible set, and we leave effective sampling there as an open frontier.

2 A MASKED SAMPLER OVER THE CONSTRUCTION GRAPH

Construction graph. A campaign fixes a parent $a \in \mathcal{V}^L$ of length L, an alphabet of size v, and a mutation budget B. A state is a complete sequence s, with touched set $T(s) = \{p : s_p \neq a_p\}$ recording the positions already edited. Each substitution replaces the token at one untouched position; a stop action ends construction. A trajectory $a = s_0 \to \cdots \to s_k$ edits each position at most once. The resulting construction graph is a directed acyclic graph rooted at the parent, whose structurally reachable designs form the Hamming ball $\mathcal{B}_B(a) = \{x : d_H(x, a) \leq B\}$ of radius B about the

parent. A feasibility predicate φ defines the feasible set $F = \{x : \varphi(x)\}$ out of that ball. We train a GFlowNet (Bengio et al., 2021) over this graph with the sub-trajectory-balance objective (Madan et al., 2023), which drives the policy toward sampling designs in proportion to a reward built from the campaign's fitness predictor: predicted fitness, floored at 10^{-8} and raised to an exponent β that controls how sharply the target concentrates on the best designs. At $\beta = 0.1$, the floor retains appreciable reward: an infeasible or non-positive prediction still scores $10^{-0.8} \approx 0.16$, and the feasible rewards run from there up to one, giving the policy a comparatively flat training target. We select sub-trajectory balance, $\beta = 0.1$, $\lambda = 0.9$, 300 training steps per round and a width-64 trunk on a separate objective-selection task. The pre-declared rule selects the lowest mean regret over 100 seeds, with diversity breaking ties. We use this configuration throughout. The policy embeds each position's token in a 32-dimensional space, concatenates the embeddings across the L positions, and passes them through a two-layer 64-unit trunk; a learned state-flow head reads that trunk and the forward head scores the admissible substitutions, while the backward policy stays uniform over the parents each state permits.

Masking illegal moves. The mask builds feasibility into the transition function instead of requiring an extra repair step. At a state s it admits a substitution when the resulting sequence still satisfies φ, so every design the policy can emit is feasible and no draw is ever rejected. The constraint is paid in the mask, in computation, not in the assays a rejection loop spends on every draw it discards. This guarantee concerns feasibility, not coverage (Appendix B). A local mask tests φ on the next complete state, not on whether a feasible design can still be *completed* from it, so under edit-from-parent the set a masked policy can actually build — the *reachable* set — can be a strict subset of $F \cap \mathcal{B}_B(a)$. The campaigns here reroot between rounds, as both published protocols we implement do; Appendix B measures how much of that difference rerooting removes. The constrained comparisons below give every arm the same parent and mutation budget; the supports they can actually construct differ, which is what Appendix B measures.

Relation to completion-aware sampling. Masked and flow-based samplers already exploit the probability that a partly built object still admits a valid completion. Nie et al. (2026) identify that probability as the correction a constrained sampler needs, with local masking the special case that treats it as constant; Ghari et al. (2024) train a GFlowNet to locate the positions in a seed sequence that compromise a target property and edit them under a learned policy. What separates our setting is whether that probability is cheap. Under a formal grammar (Park et al., 2024) or a finite automaton (Dang et al., 2026), an incremental parser reads a partial string and says at once whether any completion satisfies the constraint; the mask is then exact, every feasible design stays reachable, and what those methods repair is the distribution over an already-correct support. No such parser exists here: ours is an edit Markov decision process whose intermediate states are not prefixes, and whose predicate is not read position by position. Deciding whether a feasible completion survives is correspondingly expensive, which we argue is NP-hard for the predicate family of Corollary 4. We therefore measure what the local mask gives up: on instances small enough to enumerate, the share of feasible designs it can still reach. We also check, as an exploratory test, whether the learned flow at a state tracks how many feasible designs remain reachable from it.

Baselines. We compare against six state-of-the-art pipelines drawn from the fields a directed evolution campaign borrows from — evolutionary search, black-box optimisation, and machine-learning-guided design: the genetic algorithm the benchmark ships with, at its own rates (Stanton et al., 2024); CMA-ES over a continuous relaxation, decoded greedily (Hansen & Ostermeier, 2001); AdaLead, FLEXS' recommended algorithm (Sinai et al., 2020); ALDE, at its authors' bench configuration (Yang et al., 2025); MLDE (Wittmann et al., 2021; Yang et al., 2019), compressed to the shared budget; and proximal exploration (Ren et al., 2022), a model-guided search that favours low-order mutants of the parent and selects along a frontier in mutation count and predicted fitness. We add two arms of our own, each adapted to get past a limitation that would otherwise make the comparison uninformative: a rejection-sampling genetic algorithm that reaches feasibility by discarding, which separates the feasibility effect from the fitness effect, and an exact-projection decoder used only in §3.6, which tests whether solving the constrained problem outright removes the learned sampler's advantage.

Tasks and protocol. The synthetic constrained tasks use Ehrlich functions (Stanton et al., 2024): closed-form, procedurally generated landscapes with a guaranteed optimum and pairwise-adjacency constraints. Furthermore, we use two published protocols. ALDE assays three plates of 132 variants (396 assays), with up to twenty-one substitutions per round; EvolvePro assays eight plates of 48 variants (384 assays), with up to four substitutions per round. Both protocols run on the same Ehrlich instance, so only the campaign shape differs between them. A second instance, the sparse Ehrlich task, uses the tightest transition density tested (0.15), four plates of ninety-six variants (384 assays), and a four-substitution budget. Its parent stays fixed, so no rerooting softens the cost of rejection. The two unconstrained tasks are the empirical four-site landscapes of protein G domain B1 and tryptophan synthase (Wu et al., 2016; Johnston et al., 2024), four-site libraries of 160,000 variants of which 149,361 and 159,129 respectively were measured, the rest imputed as zero fitness, and carrying no feasibility predicate; they are the control on everything we attribute to masking, since on them the reachable set is the whole library and every masking effect must vanish. Within each task, all arms receive the same parent, constraint, rounds and oracle budget. We compare arms seed by seed within a task and read the result with a two-sided sign test. A campaign's best fitness lands on one of only six values on the ALDE protocol, the closest two differing by 0.0625, so two arms either finish on exactly the same value or separate by a visible step; counting how often each one wins is an informative way to report that, where a mean would blur the ties. We do not pool fitness across landscapes of different scales: the one pooled comparison, against the exact-projection decoder in Appendix F, runs over Ehrlich cells sharing a common normalised scale. Demšar (2006) recommends the Wilcoxon signed-ranks test for comparing methods across heterogeneous datasets, where each dataset contributes a single paired observation. Our comparisons sit inside one landscape and pair over seeds on a common normalised scale, so we report one test per landscape and count wins, ties and losses with the sign test.

3 Results

3.1 Campaign outcomes against published pipelines

We compare the best fitness reached within the assay budget and the diversity of the high-scoring designs (Table 1). Following protein-design benchmarks (Sinai et al., 2020; Yang et al., 2025), we normalise maximum fitness by the attainable optimum (Appendix B). EvoGFN's mean leads on every task, widely where few sequences are feasible and reaching the optimum on every seed of the sparse Ehrlich task, and narrowly on the two unconstrained landscapes. Read as a paired two-sided sign test over 100 seeds, the lead is significant on the constrained tasks (each $p < 0.001$), except against `cmaes` on the sparse Ehrlich task, where it ties `cmaes` on 97 of 100 seeds. On the two unconstrained landscapes, the comparisons are significant at $p < 0.05$ except for TrpB against `mlde` ($p = 0.067$). The one baseline close everywhere is `cmaes`, which ties on 97 of the 99 seeds it completed on the sparse Ehrlich task and splits roughly $53/34/13$ on the two protocol tasks. The continuous relaxation it searches cannot express a constraint between positions, so its decoded design is rarely buildable. The repair we supply on its behalf — the literature specifies none — accepts substitutions in order of gain while the design stays legal, which reads the constraint much as our mask does. Appendix F reports the full win/tie/loss on `best` for every arm and task (Table 10). We measure diversity among each campaign's top designs, since measuring it over all assays would also credit scatter among low-fitness variants a campaign would not retain; by this measure EvoGFN is the most diverse wherever the feasible set is small — such lead disappears on the unconstrained landscapes.

Two qualifications matter for interpreting this comparison. First, supervised fitting requires *usable* measurements. Infeasible assays supply no fitness targets, and the resulting shortage of training data affects the pipelines differently. `mlde` requires 96 usable measurements before fitting. Across 100 seeds per task, it collects at most 22, 63 and 29 on the ALDE protocol, the EvolvePro protocol and the sparse Ehrlich task, respectively, and remains in its random-screening stage throughout all three constrained tasks. `alde` instead refits its campaign-level ensemble whenever finite observations are available, which occurs before the final round on all 100 seeds of each constrained task. EvoGFN leads it regardless. Second, the five tasks rest on four landscapes: the two protocol tasks share one Ehrlich instance, the sparse Ehrlich task uses a second from the same generator, and GB1 and TrpB are empirical four-site landscapes. Repeating seeds shows how reliably a result holds on a given

Table 1: Normalised maximum fitness and diversity, mean over 100 seeds, with standard deviations. *Max. fitness* is the highest true fitness assayed over a campaign as a fraction of the attainable optimum the audit constructs (Appendix B), so a value of 1.000 reaches it on every seed. *Diversity* is the mean pairwise Hamming distance among each campaign's ten highest-scoring designs, repeats not removed. Bold: best mean, ahead of every other arm at $p < 0.05$ (two-sided sign test over paired seeds, unadjusted). *`cmaes` over the 99 seeds it completed on the sparse Ehrlich task; †`pex` over the 91 on which it produced a feasible design.

	`protocol-alde`		`protocol-evolvepro`		`ehrlich-feas`		`gb1-anchor`		`trpb-anchor`	
Arm	max fitness	div	max fitness	div	max fitness	div	max fitness	div	max fitness	div
EvoGFN	**0.574**±0.113	**33.3**	**0.447**±0.108	**7.9**	1.000±0.000	**4.4**	**0.750**±0.119	2.8	0.811±0.087	2.9
`genetic`	0.244±0.032	2.0	0.253±0.043	3.7	0.418±0.138	0.5	0.564±0.078	2.1	0.730±0.038	2.2
`cmaes`	0.489±0.122	22.5	0.367±0.102	6.5	0.993±0.047*	2.7	0.585±0.125	**3.1**	0.665±0.126	**3.0**
`adalead`	0.154±0.053	5.7	0.241±0.064	3.6	0.357±0.078	0.1	0.534±0.095	1.4	0.687±0.050	1.2
`alde`	0.131±0.027	6.5	0.199±0.066	5.8	0.360±0.087	0.1	0.600±0.056	2.0	0.754±0.043	1.9
`mlde`	0.151±0.051	3.2	0.233±0.056	4.6	0.382±0.111	0.3	0.626±0.040	2.4	0.777±0.049	2.1
`pex`	0.258±0.063	2.9	0.278±0.065	2.8	0.408±0.134†	2.9	0.690±0.107	2.2	0.775±0.039	2.1

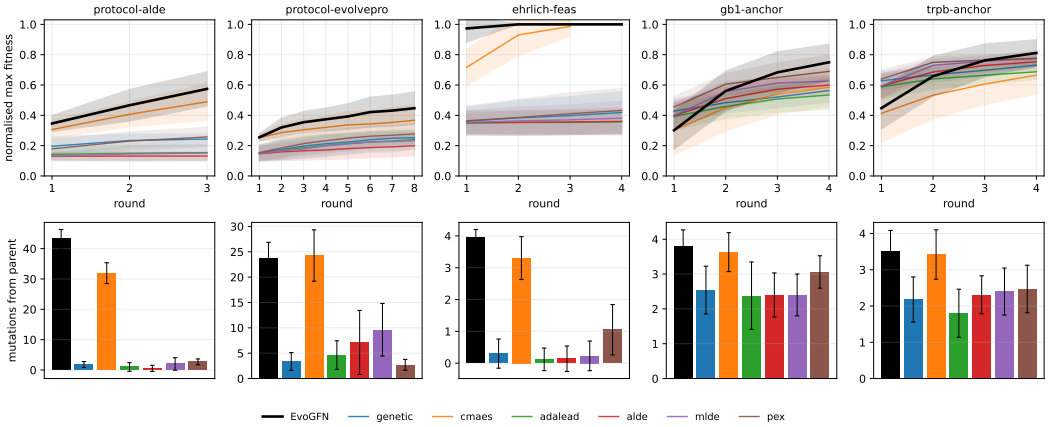


Figure 1: Fitness trajectory and the best design's edit distance. *Top:* best true fitness reached after each round as a fraction of the attainable optimum, mean ± s.d. over 100 seeds. *Bottom:* the edit distance of each method's best design from the parent, mean ± s.d. As in Table 1, EvoGFN leads throughout where few sequences are feasible and reaches the optimum by the second round on the sparse Ehrlich task, narrowing to the field on the two unconstrained landscapes, while its best designs sit farther from the parent, since every intermediate it builds is feasible and distance costs it no wasted assays.

landscape, but does not tell us whether it transfers to another. Figure 1 plots the same fitness as a round-by-round trajectory, beside each method's best-design edit distance.

GFNSeqEditor (Ghari et al., 2024), a published GFlowNet sequence editor, is the closest comparison and takes the opposite route: it ranks edits by property and leaves feasibility to chance rather than building it into the graph. At the authors' settings ($\delta = 0.4$, $\lambda = 0.1$, $\sigma = 0.001$) its assayed feasible fraction is only 0.368 on the ALDE protocol and 0.253 on the sparse Ehrlich task, and its near-deterministic editing exhausts the plate — proposing no design it has not already measured — on 72, 100 and 88% of seeds, so a clean comparison on `best` is undefined, and EvoGFN leads on every seed that does finish. Each remaining method fails the constraint in its own way: GFNSeqEditor and the rejection-genetic control can exhaust the plate; `mlde` can remain below its training threshold; and `cmaes` relies on the repair described above.

3.2 Feasibility and unbuildable designs

The masked construction graph admits only feasible designs, giving EvoGFN a feasible fraction of 1.000. The constraint excludes a substantial share of other methods' assays: a plain genetic algorithm spent 70.5% of its ALDE-protocol assays on designs the constraint rejects, and AdaLead, ALDE and MLDE spent 96.4%, 97.7% and 96.6% of theirs the same way, and an assay spent on a rejected design returns no fitness, so little of the budget becomes training data. `genetic-feasible`, a

Table 2: GB1 IgG-binding under a predicted folding-stability constraint, at two $\Delta\Delta G$ thresholds. *Normalised best* is binding fitness over the exact feasible optimum (8.76 and 6.27; the 20^4 library is enumerable, so the optimum is exact), *feasible fraction* is the share of assays spent on constructible designs, and *diversity* is the mean pairwise Hamming distance among each campaign's ten highest-scoring assayed designs, repeats not removed; all mean over 100 seeds. Bold: best mean, ahead of every other arm at $p < 0.05$ (two-sided sign test, unadjusted). `cmaes` produces no feasible design here (§3.3).

(a) Mild gate: 6 kcal/mol / 61% feasible.

Arm	Best	Feas.	Div.
EvoGFN	**0.795**±0.120	1.000	**2.70**
`genetic`	0.563±0.078	0.979	2.14
`adalead`	0.533±0.092	0.877	1.39
`alde`	0.601±0.057	0.962	2.01
`mlde`	0.620±0.049	0.932	2.38
`pex`	0.690±0.108	0.996	2.25
`pex-ensemble`	0.685±0.091	0.999	2.22
`alde-masked`	0.646±0.037	1.000	1.76
`mlde-masked`	0.643±0.040	1.000	1.60

(b) Strict gate: 3 kcal/mol / 6.6% feasible.

Arm	Best	Feas.	Div.
EvoGFN	0.901±0.051	1.000	2.29
`genetic`	0.765±0.080	0.667	1.88
`adalead`	0.678±0.078	0.491	1.23
`alde`	0.811±0.057	0.659	1.74
`mlde`	0.831±0.048	0.527	2.03
`pex`	0.908±0.025	0.793	2.28
`pex-ensemble`	0.913±0.022	0.756	2.27
`alde-masked`	0.896±0.035	1.000	1.52
`mlde-masked`	0.882±0.045	1.000	1.36

genetic algorithm that redraws until its candidates are legal, also reaches full feasibility, but pays for it in proposals: it attains 1.000 only by drawing $3{,}340$ candidates per campaign on the ALDE protocol and discarding $1{,}704$ of them, where a masked policy discards none. The same contrast appears in the edit distances of Figure 1: EvoGFN's best designs sit farther from the parent, since every intermediate it builds is feasible and distance costs it no wasted assays, while the rejection pipelines stay near the parent and stall at low fitness.

3.3 Beyond procedurally generated constraints: a real folding-stability gate

We next test whether the advantage extends beyond procedurally generated constraints, using a folding constraint derived from protein structure: a protein has to fold before it can be expressed and assayed, so a variant that does not fold yields no fitness for the campaign to read, and a threshold on its stability constrains which designs can be built and scored at all. On the empirical GB1 landscape (Wu et al., 2016), we classify a design as feasible when its predicted destabilisation relative to the wild type stays within an allowance, and the campaign optimises binding within the designs that pass. We compute ThermoMPNN's (Dieckhaus et al., 2024) predicted $\Delta\Delta G$ for each single substitution using the 1PGB structure, then sum these values over each variant's point mutations. Being predicted rather than measured, this structure-derived stability score inherits ThermoMPNN's error. Two thresholds bracket the regime: the mild GB1 folding gate, a net-unfold threshold at the domain's folding free energy of about 6 kcal/mol, leaves three-fifths of the library feasible; the strict GB1 folding gate, at 3 kcal/mol, leaves under a tenth.

Stability and binding are distinct features: across the measured library the predicted folding $\Delta\Delta G$ and the assayed binding fitness correlate at Spearman -0.34, so stability and function rank largely independently and the constraint removes designs on a ground the reward does not already encode. Some association remains — the four varied sites sit in the binding interface, where a destabilising substitution tends to cost binding — but the association is weak, short of the collinearity that would make the constraint a threshold on fitness. The wild type folds, so the campaign starts from a feasible parent.

Handed the gate, EvoGFN proposes only feasible designs at both thresholds, as the masked-pool arms do. On the mild GB1 folding gate, it also leads every published pipeline on binding: a normalised best of 0.795 against the next arm's 0.690, each margin significant at $p < 10^{-7}$, while holding the widest spread among its high-scoring designs (Table 2). CMA-ES is absent here: no repair we have applies to this constraint, so it produces no feasible design at all. The gate is not itself intractable — it is a sum of cached per-site stability changes against one threshold, additive over positions and enumerable over the whole library, so a feasible design can be found by a direct scan. Under the strict gate, only a fifteenth of the library remains feasible, and binding separates none of the leading arms: EvoGFN is matched by `pex` ($30/26/44$, $p = 0.13$), by `pex-ensemble` ($29/24/47$, $p = 0.051$)

Table 3: EvoGFN against the same policy left untrained. W/T/L on `best` from EvoGFN's side, 100 paired seeds, two-sided sign test. Both arms share the masked graph, constraint and assay budget; the untrained arm differs only in having zero training steps, and in the second column four times the proposal pool.

Task	Feasibility	vs Matched	vs 4× proposals
`protocol-alde`	sparse	28/62/10, p=0.005	41/45/14, p<0.001
`protocol-evolvepro`	sparse	35/52/13, p=0.002	35/49/16, p=0.011
`ehrlich-feas`	saturated	0/100/0	0/100/0
`gb1-anchor`	total	63/20/17, p<0.001	52/3/45, p=0.54
`trpb-anchor`	total	67/17/16, p<0.001	52/3/45, p=0.54

and by the masked-pool ALDE ($46/15/39$, $p = 0.52$). What separates them is how they spend the plate. `pex` places a fifth of its assays on designs the constraint rejects, and the masked-pool arms hold their plate feasible by narrowing onto fewer distinct designs — EvoGFN keeps a diversity of 2.29, where theirs falls to 1.52 and 1.36.

3.4 Separating masked construction from policy learning

Two controls separate masked construction from policy learning. First, we run the same policy with zero training steps, holding the masked graph, constraint and assay budget fixed. We run it at two proposal counts: the trained arm's own, and four times it, to test whether more sampling substitutes for training (Table 3). Learning improves performance where few sequences are feasible and the task is not already saturated: the trained policy led at matched proposals ($p = 0.005$ and $p = 0.002$) and led again at four times the proposals ($p < 0.001$ and $p = 0.011$). On GB1 and TrpB the same fourfold increase left no detectable advantage ($p = 0.54$ on both).

As a second control, we test whether the benefit of masking transfers to other samplers. We gave the same local mask to the genetic algorithm, so that it breeds only on the masked graph and its candidates are feasible by construction. Masking that base algorithm improved its `best` and not only its feasible fraction: the masked genetic algorithm beat the plain one $21/78/1$ on the ALDE protocol ($p = 1.1 \times 10^{-5}$) and $41/53/6$ on the EvolvePro protocol ($p = 1.8 \times 10^{-7}$). The benefit of masking is therefore not tied to the learned policy. Masked construction does not account for the full fitness difference: EvoGFN still beat the masked genetic algorithm $99/1/0$ and $73/26/1$, so the lead remains when both methods construct feasible designs on the same graph. Two further controls give a supervised screen the feasible candidates it otherwise never gets. Both draw their pool from a masked generator; one scores it with a deep ensemble and selects greedily, as MLDE does (`mlde-masked`), the other with a bootstrapped ensemble and Thompson sampling, as ALDE does (`alde-masked`). Plain MLDE differs from the first in its regression model as well. Both controls fit on feasible observations and still trail EvoGFN — $96/4/0$ and $92/8/0$ on the ALDE protocol, $67/30/3$ and $71/28/1$ on the EvolvePro protocol (all $p < 10^{-15}$) — so beating them is not an artefact of a supervised screen with too little feasible data to fit. Reinitialising the policy at each rerooting, instead of carrying it across, produced no detectable difference ($23/52/25$, $p = 0.885$; $25/53/22$, $p = 0.771$), so these experiments provide no evidence that carrying the policy across rerootings improves campaign performance (Appendix F.2).

3.5 Sensitivity to fitness predictor quality

The split in Table 3 follows the fitness predictor's accuracy. Where the predictor ranks candidates well, drawing more of them is already a good strategy, which is why an untrained policy catches up given enough draws. Measured each round on the selected batch, the correlation between predicted and measured fitness never exceeded $+0.12$ on the Ehrlich instance behind the two protocol tasks and sat near $+0.05$, while on GB1 and TrpB it rose across rounds to about $+0.47$, and on the sparse Ehrlich task reached $+0.94$ (Figure 2). Where it carries no signal, the policy's value is the range of designs it leaves for the next parent.

We tested this interpretation by degrading the fitness predictor on purpose. Each time the predictor is fitted, we corrupt its targets with Gaussian noise proportional to their spread, then compare against the same campaign run without noise over 100 paired seeds. On GB1, where the fitness predictor

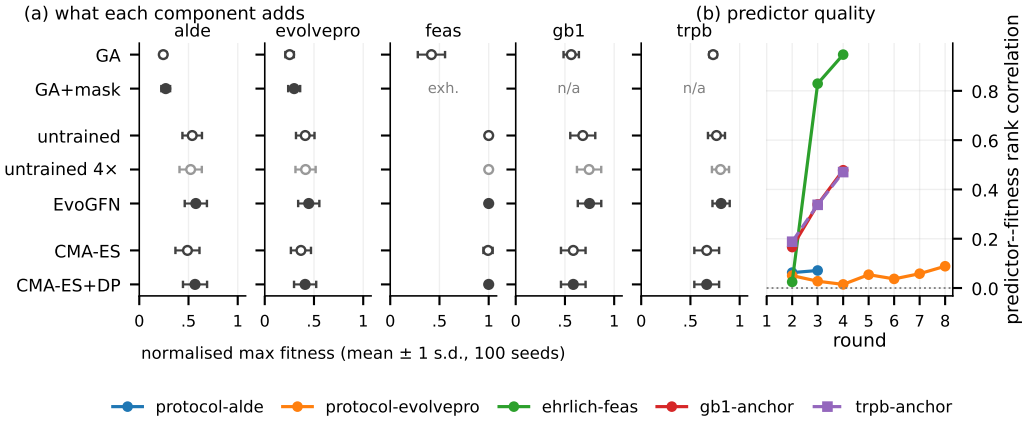


Figure 2: *(a)* Three matched comparisons per task; in each pair the filled marker is the variant under study and the open one its comparison: a genetic algorithm with and without the mask, our policy untrained at the trained arm's proposal count and at four times it, and CMA-ES with and without the exact-projection decoder. Normalised max fitness after the final round, mean ± 1 s.d. over 100 seeds; paired tests in Tables 3 and 10. `n/a`: no mask applies; `exh.`: the masked genetic algorithm exhausted its plate on all 100 seeds. *(b)* Predictor–fitness rank correlation on each round's plate (median over 100 seeds; round 1 has no fitted predictor).

is informative, the effect grows with the noise level: $\Delta\texttt{best}$ of -0.057 ($p = 0.18$) at noise 0.5, -0.354 ($p = 0.001$) at 1.0, and -1.081 ($p < 10^{-3}$) at 2.0. On the ALDE protocol, where the fitness predictor was already noise, the same intervention moved nothing measurable ($+0.005$, $p = 1.000$). The intervention shows that EvoGFN's performance depends on predictor quality where the predictor is informative. It does not tell us whether the loss comes through training or through screening, since the predictor drives both.

3.6 Comparison with an exact constrained decoder

The advantage narrows sharply where the constrained problem admits an exact decoder. When the predicate's constraint graph over positions is a path, which is the case for the Ehrlich tasks, whose adjacency constraint the benchmark ships, the completion oracle becomes a dynamic program over the positions in order. We show that the reachable set under the mask it induces is exactly $F \cap \mathcal{B}_B(a)$ (Theorem 1), and that for a score additive over positions a single pass of that recursion recovers the best reachable design exactly (Corollary 3). We implemented that program and handed it to CMA-ES as an exact-projection decoder — an instrument, since it uses a structural fact about the benchmark's constraint that no published account of CMA-ES relies on. Against it the learned sampler kept only $+0.0315$ (bootstrap $[+0.0184, +0.0448]$) across four Ehrlich instances under both protocols (800 paired seeds), with 313 of them tied (Appendix F). The margin missed significance under the protocol with fewer, larger rounds ($p = 0.096$) and reached it under the one with more, smaller rounds ($p < 10^{-5}$). The decoder optimises a sum over positions while the Ehrlich fitness multiplies motif scores, which is one candidate explanation for the residual. Where the predicate couples every position instead, Appendix D measures how little of the feasible set a local mask reaches.

4 Discussion

A masked sampler pays off where the feasibility constraint is strict, so that draw-then-reject pipelines burn budget on discards, and where the fitness predictor fitted from a few hundred assays is too weak for an $\arg\max$ to be a good move — where keeping a spread of candidates beats committing to the single best-predicted one. Its advantage shrinks where the constraint is structured enough that an exact projection exists: there a dynamic program recovers the best design under an additive decoding score at a fraction of the cost, and the learned sampler keeps only a small margin over it.

We see three limitations. First, the constrained evidence rests mainly on one generator family, the Ehrlich functions. The structure-derived GB1 folding gate reproduces the result on an empirical

landscape, but from a single ThermoMPNN-predicted signal on a single landscape, so it corroborates without establishing performance across a range of biological constraints. Second, we compare two campaign shapes rather than varying rounds systematically, so the comparison confounds every way the two protocols differ. Last, the exact projection applies only where the constraint graph is a path; the appendices give the condition for a tractable exact mask. Within those limits the result stands: the gain comes from masked construction and within-round learning, and it tracks the quality of the fitness predictor.

Several directions follow from what these experiments leave untested. Every experiment here supplies the feasibility gate in advance. A campaign rarely knows it: a design fails to express or to fold, and that verdict arrives with the assay. Learning the gate from those successes and failures, and masking against what has been learned, is the natural next step, and a substantial piece of work in its own right — the mask would then be only as good as the classifier behind it. Predictor quality governs the size of the learning benefit, and the predictor used here is a deep ensemble refit each round without further tuning; improving its calibration and architecture is the most direct way to strengthen the method. What learned sampling is worth is itself partly open, and a factorial over trained and untrained policies, proposal pool size and predictor noise, on independent instances and reported at equal compute as well as equal assays, would separate policy learning from predictor quality. Coverage is the fourth gap: all our constrained tasks derive from one Ehrlich generator and one predicted stability gate, and testing several independent constraint families, holding the feasible share fixed while varying how positions couple, would show which of our results are properties of the constraint and which of the instance. Against the exact projection the sampler holds a small margin at roughly a hundred times the compute, so on predicates that admit such a decoder it is hard to justify. Finally, every campaign here optimises one fitness: a design is often scored on several axes at once — activity, stability, expression — and a plate may have to satisfy all of them, which the single reward we use cannot express; we leave that case to later work.

Reproducibility Statement

Each seeded campaign produces a record keyed by $(\text{task}, \text{arm}, \text{seed})$ and fingerprinted with the code that produced it, linking every reported number to its source. Comparisons use the same 100 paired seeds and the two-sided sign test defined in §2. The benchmark tasks are the public Ehrlich functions (Stanton et al., 2024) and the public GB1 and TrpB landscapes (Wu et al., 2016; Johnston et al., 2024), at the protocol budgets stated in §2. We give the soundness arguments in full in Appendix C, and the reachable-set audit in Appendix B was run against the benchmark's own reference implementation. The code and experimental result records will be made publicly available upon acceptance.

Ethics Statement

We evaluate sampling algorithms on synthetic benchmarks and public in-vitro protein-fitness data, with no human subjects, private data or new wet-lab experiments. The objects of study are sampling algorithms, evaluated on fixed public benchmarks, and no sequence produced here is intended for synthesis. We see no ethical concern requiring further review.

Use of Large Language Models

We used large language models throughout this work, under our direction: to write and refactor benchmark and analysis code; to locate related work, every citation of which we then read and verified; to draft and edit the manuscript; to develop and stress-test the technical arguments and proofs; to give feedback on experimental design and methodology; and to interpret results and audit them against the stored records, which surfaced errors we then corrected. The synthetic instances come from the published Ehrlich generator, and we used no language model for translation, for cleaning or reformatting the public landscape data, or for qualitative or thematic analysis, which this work does not involve. We set every design, hyperparameter and protocol, produced every number with our own code and verified it against the record store, so no claim entered the paper unchecked. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including text, claims and artefacts produced with the aid of generative AI.

REFERENCES

Olivier Bailleux and Pierre Marquis. DISTANCE-SAT: Complexity and algorithms. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pp. 642–647. AAAI Press, 1999.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. GFlowNet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023. URL https://jmlr.org/papers/v24/22-0364.html.

Jieyi Bi, Yining Ma, Jianan Zhou, Wen Song, Zhiguang Cao, Yaoxin Wu, and Jie Zhang. Learning to handle complex constraints for vehicle routing problems. In *Advances in Neural Information Processing Systems*, volume 37, pp. 93479–93509, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/a9d2a5fd12d34250c21b5e4fa8d906b0-Abstract-Conference.html. Introduces the PIP framework and the PIP-D variant, which learns the feasibility mask with an auxiliary decoder.

David H. Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *PMLR*, pp. 773–782, 2019.

Jean Cardinal, Erik D. Demaine, David Eppstein, Robert A. Hearn, and Andrew Winslow. Reconfiguration of satisfying assignments and subset sums: Easy to find, hard to connect. *Theoretical Computer Science*, 806:332–343, 2020. doi: 10.1016/j.tcs.2019.05.028.

Angelica Chen, Samuel Don Stanton, Frances Ding, Robert G. Alberstein, Andrew Martin Watkins, Richard Bonneau, Vladimir Gligorijević, Kyunghyun Cho, and Nathan C. Frey. Generalists vs. specialists: Evaluating LLMs on highly-constrained biophysical sequence optimization tasks. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 9029–9072, 2025. URL https://proceedings.mlr.press/v267/chen25bg.html.

Miruna Cretu, Charles Harris, Ilia Igashov, Arne Schneuing, Marwin Segler, Bruno Correia, Julien Roy, Emmanuel Bengio, and Pietro Liò. SynFlowNet: Design of diverse and novel molecules with synthesis constraints. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uvHmnahyp1.

Meihua Dang, Linxin Song, Honghua Zhang, Jieyu Zhao, Guy Van den Broeck, and Stefano Ermon. Mitigating bias in locally constrained decoding via tractable proposals. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Henry Dieckhaus, Michael Brocidiacono, Nicholas Z. Randolph, and Brian Kuhlman. Transfer learning to leverage larger datasets for improved prediction of protein stability changes. *Proceedings of the National Academy of Sciences*, 121(6):e2314853121, 2024. doi: 10.1073/pnas.2314853121.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. Grammar-constrained decoding for structured NLP tasks without finetuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10932–10952, 2023. doi: 10.18653/v1/2023.emnlp-main.674.

Pouya M. Ghari, Alex M. Tseng, Gökcen Eraslan, Romain Lopez, Tommaso Biancalani, Gabriele Scalia, and Ehsan Hajiramezanali. GFlowNet assisted biological sequence editing. In *Advances in Neural Information Processing Systems*, volume 37, pp. 106841–106869, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/c14760740573001c0d18d58879a6a305-Abstract-Conference.html.

Parikshit Gopalan, Phokion G. Kolaitis, Elitza N. Maneva, and Christos H. Papadimitriou. The connectivity of Boolean satisfiability: Computational and structural dichotomies. *SIAM Journal on Computing*, 38(6):2330–2355, 2009. doi: 10.1137/07070440X.

Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In *Advances in Neural Information Processing Systems*, volume 36, pp. 12489–12517, 2023.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398. The paper that introduces CMA-ES. Some of the sequence-design literature instead cites the later review, N. Hansen, "The CMA Evolution Strategy: A Comparing Review", in *Towards a New Evolutionary Computation*, Springer, 2006, pp. 75–102.

Moksh Jain, Emmanuel Bengio, Alex Hernández-García, Jarrid Rector-Brooks, Bonaventure F. P. Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. Biological sequence design with GFlowNets. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9786–9801, 2022. URL https://proceedings.mlr.press/v162/jain22a.html.

Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-García, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective GFlowNets. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14631–14653, 2023. URL https://proceedings.mlr.press/v202/jain23a.html.

Kadina E. Johnston, Patrick J. Almhjell, Ella J. Watkins-Dulaney, Grace Liu, Nicholas J. Porter, Jason Yang, and Frances H. Arnold. A combinatorially complete epistatic fitness landscape in an enzyme active site. *Proceedings of the National Academy of Sciences*, 121(32):e2400439121, 2024. doi: 10.1073/pnas.2400439121.

Michał Koziarski, Andrei Rekesh, Dmytro Shevchuk, Almer van der Sloot, Piotr Gaiński, Yoshua Bengio, Cheng-Hao Liu, Mike Tyers, and Robert A. Batey. RGFN: Synthesizable molecular generation using GFlowNets. In *Advances in Neural Information Processing Systems*, volume 37, pp. 46908–46955, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/53704142f230054140418ecd8857f391-Abstract-Conference.html.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning GFlowNets from partial episodes for improved convergence and stability. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23467–23483, 2023. URL https://proceedings.mlr.press/v202/madan23a.html.

Tao Meng, Sidi Lu, Nanyun Peng, and Kai-Wei Chang. Controllable text generation with neurally-decomposed oracle. In *Advances in Neural Information Processing Systems*, volume 35, pp. 28125–28139, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b40d5797756800c97f3d525c2e4c8357-Abstract-Conference.html.

Wenhua Nie, Zijie Meng, Kun Zou, Zheng Lin, Ziwei Li, Haoran Zheng, Jyh-Shing Roger Jang, and Hao Zhang. Future validity is the missing statistic: From impossibility to Φ-estimation for grammar-faithful speculative decoding, 2026.

Kanghee Park, Jiayu Wang, Taylor Berg-Kirkpatrick, Nadia Polikarpova, and Loris D'Antoni. Grammar-aligned decoding. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24547–24568, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2bdc2267c3d7d01523e2e17ac0a754f3-Abstract-Conference.html. Introduces ASAp, adaptive sampling with approximate expected futures.

Peter J. Ramadge and W. Murray Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal on Control and Optimization*, 25(1):206–230, 1987. doi: 10.1137/0325013.

Zhizhou Ren, Jiahan Li, Fan Ding, Yuan Zhou, Jianzhu Ma, and Jian Peng. Proximal exploration for model-guided protein sequence design. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 18520–18536, 2022.

Seonghwan Seo, Minsu Kim, Tony Shen, Martin Ester, Jinkyoo Park, Sungsoo Ahn, and Woo Youn Kim. Generative flows on synthetic pathway for drug design. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=pB1XSj2y4X. The method is RxnFlow.

Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric D. Kelsic. AdaLead: A simple and robust adaptive greedy search algorithm for sequence design, 2020. URL https://arxiv.org/abs/2010.02141. arXiv preprint, never published at a venue; the paper that introduces the FLEXS sandbox and proposes AdaLead as its baseline sequence-design algorithm.

Samuel Stanton, Robert Alberstein, Nathan Frey, Andrew Watkins, and Kyunghyun Cho. Closed-form test functions for biophysical sequence optimization algorithms. In *1st Machine Learning for Life and Material Sciences Workshop at ICML 2024*, 2024. The citation requested by the reference implementation, `holo-bench`; the archival version of this work is Chen et al., 2025.

Esmeralda S. Whitammer, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in GFlowNets. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5955–5967, 2022. doi: 10.52202/068431-0431. URL http://papers.nips.cc/paper_files/paper/2022/hash/27b51baca8377a0cf109f6ecc15a0f70-Abstract-Conference.html.

Brandon T. Willard and Rémi Louf. Efficient guided generation for large language models, 2023. Implemented in the Outlines library.

Bruce J. Wittmann, Yisong Yue, and Frances H. Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Systems*, 12(11):1026–1045.e7, 2021. doi: 10.1016/j.cels.2021.07.008.

W. M. Wonham and P. J. Ramadge. On the supremal controllable sublanguage of a given language. *SIAM Journal on Control and Optimization*, 25(3):637–659, 1987. doi: 10.1137/0325036.

Nicholas C. Wu, Lei Dai, C. Anders Olson, James O. Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 5:e16965, 2016. doi: 10.7554/eLife.16965.

Jason Yang, Ravi G. Lal, James C. Bowden, Raul Astudillo, Mikhail A. Hameedi, Sukhvinder Kaur, Matthew Hill, Yisong Yue, and Frances H. Arnold. Active learning-assisted directed evolution. *Nature Communications*, 16(1):714, 2025. doi: 10.1038/s41467-025-55987-8. URL https://www.nature.com/articles/s41467-025-55987-8. The ALDE protocol; the arm in this paper follows the configuration the authors took to the bench — one-hot encodings, a five-member bootstrapped DNN ensemble and Thompson sampling.

Kevin K. Yang, Zachary Wu, and Frances H. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature Methods*, 16(8):687–694, 2019. doi: 10.1038/s41592-019-0496-6.

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck, and Nanyun Peng. Adaptable logical control for large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 115563–115587,

2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/d15c16cf5619a2b1606da5fc88e3f1a9-Abstract-Conference.html. The method is Ctrl-G.

Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Grosse. Probabilistic inference in language models via twisted sequential Monte Carlo. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 60704–60748, 2024. URL https://proceedings.mlr.press/v235/zhao24c.html.

A Related work

A.1 GFlowNets

GFlowNets provide the sampler used here. Their objective targets a distribution proportional to reward, allowing a design round to sample several high-reward candidates. Bengio et al. (2021) cast the generation of a compositional object as a flow through a directed acyclic graph and turn flow-conservation constraints into a learning objective, in the way Bellman equations become temporal-difference learning, so that any global minimiser samples in proportion to the reward. The motivation is the one a design round needs — reinforcement learning converges to a single mode, Markov chain Monte Carlo mixes poorly between them, and a plate wants a diverse batch of high-reward candidates. The measure-theoretic generalisation (Bengio et al., 2023) supplies the state-graph formalism we use. Credit in flow matching and detailed balance propagates locally, and so slowly along long trajectories, which is why Whitammer et al. (2022) propose trajectory balance. Our selected configuration instead uses sub-trajectory balance (Madan et al., 2023), with a learned state-flow head and a forward-policy head reading the same two-layer trunk, and a uniform backward policy (Section 2). Their convergence metric is the unnormalised L_1 distance between the learned and the true reward-proportional distribution on an enumerable hypergrid. Appendix B moves that measurement onto a real landscape and finds it defined over the wrong set.

We contribute no new objective. We implement trajectory balance, detailed balance and sub-trajectory balance, and report the SubTB configuration that the selection rule of Section 2 picks.

A.2 GFlowNets for biological sequence design

The closest prior sampler also places a GFlowNet inside a design loop, but constructs sequences *de novo*, outside the setting examined by our audit. Jain et al. (2022) place a GFlowNet generator inside an active-learning loop, take the reward from an acquisition function over a proxy model carrying epistemic uncertainty, and build sequences *autoregressively*, token by token, so the state graph is a tree and the backward policy is trivial. That construction is the *de novo* setting of Theorem 2(a): an intermediate is a prefix, the local mask is the completion oracle, and the reachable set is the feasible set. Nothing in this paper's audit applies to it. Extending the line to several objectives at once, Jain et al. (2023) train a preference-conditional GFlowNet whose single policy covers a Pareto front, keeping token-by-token construction for sequences, and in the active-learning variant of the same work abandon that construction for a sequence of (location, token) edits applied to a parent — editing scales better to long sequences. That change of action space is what moves the setting from Theorem 2(a) to (b), and neither paper reports what it costs, because neither enumerates the set its sampler can construct.

We study the edit formulation, compare it with *de novo* construction theoretically (Theorem 2), and keep their Top-K and diversity definitions so the metrics stay comparable. We add the measurement neither paper can make on a synthetic grid: on an enumerable landscape the target distribution is computable, which is what lets the support it is defined on be separated from the support the sampler has.

A.3 Machine-learning-guided directed evolution

We compare our sampler with pipelines that generate candidates and rank them by predicted fitness. Yang et al. (2019) learn a sequence-to-function model from measured variants, propose the next

round, screen, and repeat, under a hard screening budget and on a landscape shaped by epistasis and ruggedness; Wittmann et al. (2021) instantiate the loop over combinatorial libraries as MLDE, a fixed training plate followed by a supervised handover. Both are single-objective and, at the proposal step, greedy: they rank and take the top predictions, the behaviour that collapses a batch onto one mode. We implement MLDE as a baseline rather than a foil. At small budgets on smooth landscapes it is expected to win. Its budget is nominal once a constraint bites, as it is for any pipeline that screens on usable assays: both published MLDE arms gate the handover on *usable* measurements and discard infeasible assays, so on the constrained instances tested here they accumulate far fewer training examples than their budget suggests, and plain MLDE never reaches its supervised stage (§3.1).

Conditioning by adaptive sampling (Brookes et al., 2019) reweights a generative model each round toward its own high-fitness estimates, replacing greedy top-k proposals; it leaves feasibility to the reward, so a hard constraint makes it discard as the pipelines above do, and we cite it without running it. Guided discrete diffusion (Gruver et al., 2023) designs directly in sequence space and steers a denoising model with classifier guidance, applying locality and developability constraints as terms that bias sampling: a design that violates one is discouraged, not excluded, so the guarantee a mask gives is unavailable to it. It also optimises several objectives at once, which the campaigns here do not, since we score one fitness and leave the multi-objective case to future work. Pretrained protein language models (Lin et al., 2023) enter the same proposal step, supplying either the candidates or the fitness estimate and scoring how typical a sequence is among natural proteins, a signal separate from whether a design is reachable under the campaign's constraint.

A.4 Benchmarks for constrained sequence optimisation

Stanton et al. (2024) argue that existing benchmarks either enumerate a small space exhaustively, fit an empirical fitness predictor whose off-manifold behaviour cannot be trusted, or run a slow simulation with no well-characterised solution, and propose Ehrlich functions instead: closed-form, procedurally generated, cheap to evaluate, and carrying an optimum guaranteed to exist. A design is scored by a product over spaced motifs of quantised maxima over placements, feasibility comes from the transition matrix of a discrete Markov process, and difficulty is tuned through length, motif count, motif size and the quantisation parameter. The archival version of that work (Chen et al., 2025) puts the suite to its intended use, pitting general-purpose language models against specialised solvers on the same constrained instances. Appendix B audits the reference implementation. Two of its properties matter for what follows — the reward is a product of quantised maxima and so not additive over positions, which is what Remark 3 turns on, and the feasibility predicate is a pairwise adjacency whose constraint graph is a path, which is why an exact projection exists for it at all.

The unconstrained control in our suite is an empirical landscape measured at almost every variant. Wu et al. (2016) assayed $149{,}361$ of the $160{,}000$ variants at four epistatically coupled positions of protein G domain B1, and that surface remains the standard near-complete protein fitness landscape; we impute the $10{,}639$ unmeasured variants as zero fitness (§2), which makes regret, mode coverage and distributional comparison exact against the imputed library rather than against the assay. Its lack of any feasibility constraint makes it the control in Section 3, where the reachable set is the whole Hamming ball, the entire library, and every effect this paper attributes to masking must vanish.

A.5 Constrained and grammar-aligned decoding

Constrained generation in language modelling addresses a related but distinct failure. Grammar-constrained decoding (Geng et al., 2023; Willard & Louf, 2023) masks the next-token distribution with an *incremental parser*, admitting a token exactly when some completion of the current prefix stays in the language; the mask is a co-accessibility oracle, computed by construction from the automaton or the parser state. Its support is therefore *exact*, and the pathology ASAp and its relatives repair (Park et al., 2024) is probability *distortion* within a correct support — renormalising a locally masked distribution changes the relative weights of strings the model would otherwise produce, so the constrained sampler is not the conditional of the unconstrained one. Appendix B measures the other failure, support *loss*, mass placed on a strict subset of the intended set, which arises because an intermediate in an edit MDP is not a prefix and no incremental viability parser exists for it. The two failure modes are disjoint: where the parser can be built, the one this paper is about does not arise.

A.6 Completion statistics and learned feasibility signals

One line of work estimates the probability that a partial object admits a valid completion and corrects generation by it, and reads a GFlowNet's flow as already such an estimator; we claim neither idea. Nie et al. (2026) name the conditional probability of a valid completion as the missing correction statistic, prove that the correctly constrained sampler is a Doob h-transform whose h is exactly that statistic, and show local masking is the constant-h case, then recover that statistic by one-step lookahead, by exact dynamic programming and by finite-language correction, each of which reads it off an explicit grammar rather than learning it. Ghari et al. (2024) go further in our own domain: reading a GFlowNet's flow as the total reward of the sequences extending a given prefix, they threshold it to gate edits in a sequence-editing GFlowNet for biological sequence design, the mechanism our own setting also exhibits — at a threshold we would take to zero. The same shape recurs outside the GFlowNet literature: a neural head trained to approximate an oracle's expectation over completions and used to reweight a base model (Meng et al., 2022), a distilled tractable model standing in for the constraint (Zhang et al., 2024), and twist functions learned for sequential Monte Carlo (Zhao et al., 2024). Where the exact mask is intractable, learning it with an auxiliary decoder has been proposed directly (Bi et al., 2024). Most recently, Dang et al. (2026) construct a tractable completion-aware proposal whose support is exact and which carries sequential Monte Carlo to the constrained distribution with far fewer particles, but for finite-automaton constraints, where the completion oracle is a table lookup and the reachable set equals the feasible one; the regime we occupy is the one that construction excludes, a non-regular endpoint predicate whose completion oracle is NP-hard (Corollary 4).

Two consequences for scope follow. That a flow-based sampler acquires the completion statistic without a second estimator is a property of the objective rather than a contribution of ours, and our own amortisation measurements do not support treating it as transferable across rerootings. A cheap exact oracle also exists for *regular* specifications: trimming the recognising automaton of its dead states leaves exactly the co-accessible ones, so the completion question is answered by a table lookup. An argument for learning the mask is therefore admissible only against a predicate for which no such automaton is small, the obligation Corollary 4 and Appendix D discharge with a predicate of induced width $L - 1$ rather than with the chain predicates the benchmark ships.

A.7 Supervisory control and reconfiguration

The vocabulary Appendix C uses is borrowed. Supervisory control of discrete-event systems (Ramadge & Wonham, 1987) supplies *supervisor* for a mask, *co-accessible* for a state from which a marked object is still attainable, *blocking* for a supervisor that can strand a trajectory, and *trim* for the restriction of a graph to accessible and co-accessible states, with the fact that a maximally permissive non-blocking supervisor exists and is unique (Wonham & Ramadge, 1987). Theorem 1 and Proposition 2 instantiate that on the mutation lattice, where the difficulty is not that the trim fails to exist but that computing it at sampling time is the whole problem. Methods that mask synthesis actions (Cretu et al., 2025; Seo et al., 2025; Koziarski et al., 2024) take the route itself as the specification and fall outside this claim.

The complexity of moving between admissible objects one step at a time is the subject of the reconfiguration literature, from which we take two facts and one warning. Gopalan et al. (2009) classify st-connectivity of Boolean solution graphs, PSPACE-complete in general because a witnessing walk may be exponentially long; Bailleux & Marquis (1999) show that deciding whether a satisfying assignment exists within a given Hamming distance of a point is NP-complete already for 2-CNF, Horn and reverse-Horn formulas, the independent hardness witness of Remark 4. The warning is that our lattice is more restricted than the standard setting — every position is edited at most once, so every trajectory is a geodesic of length at most L and the edit order is a polynomial certificate. The reachability question here is therefore in NP unconditionally and the PSPACE results do not transfer, in contrast to settings such as subset-sum reconfiguration (Cardinal et al., 2020) where they do; the restriction buys a lower complexity class and no tractability.

A.8 Position of this work

Table 4 compares these sequence-design lines on the questions studied here. Several constrain construction; the final column records which of them measure the set their constrained sampler can reach.

Table 4: Where the sequence-design lines stand on the two questions Appendices B and D ask. "Enumerable ground truth" means the target distribution can be computed exactly rather than estimated; "support reported" means the size of the reachable set is measured against the feasible set. Entries describe the cited implementations and configurations, not every variant in those literatures. Ehrlich functions are a benchmark rather than a method, so the sampling column does not apply to them.

	Enumerable ground truth	Feasibility enforced by masking	Samples $\propto$ reward	Support reported
Classical and ML-guided DE	no	no	no	no
GFlowNet-AL (Jain et al., 2022)	no	no	yes	no
MOGFN (Jain et al., 2023)	synthetic grid	no	yes	no
Ehrlich functions (Stanton et al., 2024)	yes	yes	—	no
EvoGFN (this work)	yes	yes	yes	yes

Two arms in our own tables are ours rather than the literature's, and we label them wherever they appear. The exact-projection instrument of Section 3 is CMA-ES (Hansen & Ostermeier, 2001) handed a constrained decoder written for this paper — one that exploits a structural property of the benchmark's predicate the benchmark does not expose; the early-fitting MLDE variant lowers the published training size so the supervised handover falls within this suite's budget. Neither is a published method, and neither appears in a table of published methods.

B THE REACHABLE SET AND THE INSTRUMENT THAT MEASURES IT

Comparing methods on a constrained sequence-design benchmark first requires identifying which designs each method can construct. A masked edit-based method does not draw from the feasible set the benchmark scores against; it draws from the set its own action space can construct, and the two need not coincide.

This appendix measures that discrepancy on the benchmark authors' own reference implementation. We separate the feasible set from the reachable set, label each failure by which of three causes it exhibits, and show that the discrepancy is a property of the mask a method is handed: changing the mask, or more cheaply the protocol, recovers much of what it had excluded.

B.1 THE FEASIBLE AND REACHABLE SETS

A benchmark scores against one set of sequences; a masked edit-based method draws from another. Section 2 names the two: the feasible set $F = \{x : \varphi(x)\}$ the benchmark scores against, and the reachable set R of endpoints of walks the mask permits, which is all the method can construct. Appendix C settles how they relate. The local mask M_φ, which tests the *next* state alone and never asks whether a feasible design can still be completed, gives $R(M_\varphi) \subseteq F \cap B_B(a)$, in general strictly; the completion-oracle mask M_C, the trim of the construction graph in the supervisory-control sense (Ramadge & Wonham, 1987), attains $R(M_C) = F \cap B_B(a)$ exactly.

That inclusion is what makes the choice of support a measurement decision. A benchmark scoring a masked sampler against a target normalised over $F \cap B_B(a)$, the cheap closed-form answer, charges it twice for every feasible design with no legal construction order: once for the mass it cannot place there, and again for the mass it places elsewhere. The distortion is a property of the support rather than of the sampler, and the sections below measure how much of $F \cap B_B(a)$ the local mask leaves out.

B.2 THE REFERENCE IMPLEMENTATION AND ITS INSTANCES

We ran the audit on `pytorch-holo`, the reference implementation of Ehrlich functions released by Stanton et al. (2024), in a separate process with its own interpreter and pinned dependencies. Holo generates the transition matrix, the motifs, the spacings, the wild type and the planted optimum, and we query its own `optimal_solution()` for the design that attains reward 1.0; our audit code dispatches on the landscape interface and runs against holo's instances unmodified.

Table 5: The audit on `pytorch-holo`'s own instances, 100 reference instances per shape unless noted. Column 2 is the fraction of instances on which the planted optimum has *no legal construction order under local masking* at any mutation budget; rates are Wilson score intervals at 95%. Columns 3 and 4 are the audited attainable fraction of the optimum under a fixed parent and under rerooting at the protocol's own cumulative budget, as Student-t intervals over instances. The three $L = 64$ rows draw from the *same* reference instance distribution and differ only in the protocol wrapped round them, which is what makes the repeated 0.760 a consistency check rather than a coincidence (Section B.6). †Both $L = 256$ audits are over 30 instances rather than 100, since the beam is the one expensive step, at 338 s per instance against 11–17 s at the smaller shapes; both are lower bounds, because the bracket closes on only 2 of those 30 draws. The unconstructibility rate in column 2 is over the full 100. The apparent absence of recovery at $L = 256$ should therefore not be read as a property of rerooting: where the audit pins on 2 draws in 30, the two bounds are what a mostly-unsuccessful beam constructed, and the gap between them carries correspondingly little information.

Shape	Per-round budget	No legal order under M_φ	Fixed-parent attainable	Rerooted attainable
$L = 32$, `diagnostic`	4	0.550 [0.450, 0.646]	0.678 [0.650, 0.706]	**0.909** [0.883, 0.934]
$L = 64$, `ehrlich-feas`	4	0.760 [0.666, 0.834]	0.728 [0.701, 0.754]	**0.941** [0.918, 0.963]
$L = 64$, `alde`	21	0.760 [0.666, 0.834]	0.948 [0.927, 0.969]	**0.971** [0.953, 0.988]
$L = 64$, `evolvepro`	4	0.760 [0.666, 0.834]	0.728 [0.701, 0.754]	**0.953** [0.933, 0.972]
$L = 256$, `large-space`	62	0.990 [0.944, 0.998]	0.443 [0.372, 0.514]†	0.466 [0.398, 0.535]†

We checked two prerequisites for interpreting the audit. First, the two implementations compute the same reward: scoring identical batches of 401 sequences (mixing holo's planted optimum, its `initial_solution` draws, uniform draws and mutants of the optimum at every partial-match level) through holo's `evaluate_true` and through ours gave zero feasibility mismatches and $\max|\Delta\text{reward}| = 0$ at each of $L = 32$, 64 and 256. Because the reward is quantised, any disagreement would have been a whole quantisation level rather than a rounding error, which makes this a sharp test. Second, the instance *distributions* differ substantially, so we report the audit on holo's instances and never mix it with ours: holo has no density knob (its bandwidth is $\lfloor 0.4v \rfloor$, giving transition density 0.858 at $v = 20$ against our 0.15–0.5), its chain is banded rather than i.i.d. so that token index is a metric, it always permits repeats, and its motifs are a contiguous walk of the chain whose planted optimum is a sequence of runs. No regret number should be quoted across the two generators.

The benchmark's default configuration also affects the interpretation of feasibility. Holo's bandwidth is a function of the alphabet size alone, so at its signature default `num_states`$= 5$ every token pair is permitted and no sequence is infeasible at all, and density never falls below about 0.75 at any alphabet size. Feasibility is vacuous at the defaults, and a feasible fraction measured there carries no information.

B.3 No legal construction order under local masking

An unbounded mutation budget isolates the effect of construction order. For the design that holo constructs and verifies at reward 1.0, we asked whether *any* ordering of its substitutions is legal under M_φ in a single-substitution environment where each position is mutated at most once. The endpoint is feasible by construction, since $\varphi(x^\star)$ holds on every instance, so a negative answer isolates the ordering obstruction from everything else.

Table 5 reports the rate over 100 reference instances per shape, with Wilson score intervals. Seeds are `range(100)`; a seed fixes holo's instance *and* its wild type, so a draw is a draw of the whole problem.

At Stanton et al.'s own base configuration ($L = 256$, $v = 20$, $c = 4$, $k = 8$, $q = 4$) the sequence holo constructs and verifies at 1.0 had no legal construction order under local masking on 99 of 100 instances, and the rate fell with sequence length, to 0.760 at $L = 64$ and 0.550 $[0.450, 0.646]$ at $L = 32$. A sign test against a coin flip gave $p = 1.6 \times 10^{-28}$ at $L = 256$ and $p = 1.8 \times 10^{-7}$ at $L = 64$; at $L = 32$ it gave $p = 0.37$, so the obstruction there strikes about half the instances and the interval covers both sides of a half. The effect is a length effect, and it must not be quoted as a single number across shapes.

Separately, and needing no interval, at each task's *own* per-round budget the planted optimum was constructible on *none* of the 500 instance–protocol rows.

B.4 Recovery of attainable fitness under rerooting

Nothing in that result says the excluded designs are unreachable. Corollary 2 of Appendix C says the opposite: for every $x \in F \cap B_B(a)$ a legal construction order exists under *some* exact mask, because M_C admits every ordering of a design's substitutions. The only defensible caption on Table 5 is therefore "*no order exists under local masking*"; written as "unreachable", the sentence would attribute to the problem a property that belongs to the mask, and it would invite the reply that a better mask should be used, to which the answer is that we used one, and the recovery it produced is reported below.

The fourth column of Table 5 reports a weaker statement than that corollary, since it holds the local mask fixed and changes the protocol instead. Letting the campaign *reroot* between rounds, which is what the protocols in this literature already do, lifts the attainable fraction of the optimum to 0.909 at $L = 32$ (4 rounds of 4) from 0.678 under a fixed parent, and to 0.941, 0.971 and 0.953 at $L = 64$ across the three protocols from 0.728, 0.948 and 0.728 respectively, of which `alde` runs 3 rounds of 21 and `evolvepro` 8 rounds of 4. Both halves of the claim were swept at 100 instances per shape, on code we did not write: the planted optimum of a benchmark instance typically had no legal construction order under the local mask, and a protocol that rerooted recovered 0.94–0.97 of it against 0.728–0.948 under a fixed parent.

Rerooting recovers reachability by resetting the once-per-position restriction. A fixed-parent round mutates each position at most once, so a design whose only legal construction order would set a position to an enabling token and later to its final one is unreachable within a single round; rerooting builds a fresh environment each round and thereby resets that once-per-position restriction, which is what lets the chain route through an intermediate token an adjacency requires. What that enlarges is the set a masked campaign can *construct*; columns 3 and 4 report attainable fitness under the respective protocols, with bounds where the beam does not close, and neither column measures the size of those sets. Whether the enlargement is decisive for realised fitness depends on a constraint that forces such a detour; the folding gate of §3.3 would be the natural setting to test that, and we leave the ablation to future work.

Together, these measurements clarify what performance on this benchmark family measures. The gap is large, it is present in the reference implementation rather than in our port of it, and it is an artefact of a mask that can be replaced without touching the constraint, the landscape or the budget. A figure reported against F under M_φ therefore mixes a method's competence with the mask's incompleteness, and a figure reported against $R(M_\varphi)$ under a rerooting protocol does not. The `protocol-alde` and `protocol-evolvepro` campaigns of Section 3 reroot between rounds, the configuration column 4 measures at 0.94–0.97 of the optimum; the sparse feasibility diagnostic and the folding-gate tasks keep a fixed parent.

B.5 A taxonomy of causes

Failure to find the optimum can have three distinct causes:

(i) **Budget.** The design lies outside $B_B(a)$ at the budget the protocol actually spends. Nothing to do with masking.

(ii) **Ordering.** The design lies in $F \cap B_B(a)$, but no monotone construction order is legal under M_φ. This is the failure Section B.3 measures, and by Corollary 2 it is a property of the local mask alone.

(iii) **Intractability.** No sound and complete mask is evaluable in polynomial time, so the exact mask is out of reach and only incomplete or approximate ones remain, M_φ among them. This is the regime Appendix C characterises and the one in which some approximation of M_C has to be accepted.

Each cause calls for a different remedy — a larger budget, a better mask or a learned one — so a failed search has to be labelled before it can be interpreted. Table 6 labels the four MAIN Ehrlich tasks of our suite, each at its own cumulative budget, with the cause it exhibits.

Table 6: Cause taxonomy on the four MAIN Ehrlich tasks, at each task's own cumulative budget (rounds × per-round budget). $\varphi(x^\star)$ is the endpoint test on the planted optimum; "in budget" is membership of $F \cap B_B(a)$ at that cumulative budget; "constructible" is the existence of a legal order under M_φ. Printing the budget column and the constructibility column separately is what separates cause (i) from cause (ii); at *unbounded* budget all four rows are cause (ii), and reporting only that would have overstated the ordering effect.

Task	$\varphi(x^\star)$	$d_H(a, x^\star)$	Cumulative budget	In budget	Constructible
`protocol-alde`	True	61	63	True	False
`large-space`	True	248	248	True	False
`ehrlich-feas`	True	62	4	False	False
`protocol-evolvepro`	True	61	32	False	False

The four tasks split two against two. On the sparse Ehrlich task and the EvolvePro protocol, the planted optimum lies outside the budget the protocol spends, at $d_H(a, x^\star) = 62$ and 61 against cumulative budgets of 4 and 32 — cause (i), with masking not implicated. On the ALDE protocol and `large-space` the optimum is feasible *and* within budget, at 61 against 63 and at 248 against 248, and still has no legal construction order under the local mask, which is cause (ii); on those two an exact mask demonstrably recovers it.

The audit does not exhibit cause (iii): every predicate shipped here has a path constraint graph, so a sound tractable mask exists in principle. Cause (iii) needs a predicate of high induced width, which Appendix C constructs.

B.6 Checks and scope

Three checks control the instrument. Whether the planted optimum admits a legal construction order at unbounded budget is a property of the instance and not of the schedule wrapped round it: the three $L = 64$ shapes of Table 5 differ only in protocol, and their unconstructibility rates came out identical at 0.760, per draw. The rerooted recovery, by contrast, is a statement about what a campaign builds across rounds, and it did vary with the protocol, giving 0.941, 0.971 and 0.953 on those same instances. Either quantity behaving the other way round would indicate a bug. The sweep also reproduced figures obtained earlier by a different route, and corrected two subsidiary readings of them: the audit brackets on about a quarter of the small-shape instances, not only at $L = 256$, and the repaired-fraction floor is 0.594 over 400 sub-L=256 draws, not the 0.67 that five draws had suggested.

Two distinctions limit what the audit covers. When designs are built by appending tokens, an intermediate state is a prefix and a chain language is prefix-closed, so the local mask *is* the completion oracle and $R = F$ exactly. The gap appears under edit-from-parent action spaces, where an intermediate is a complete hybrid of parent and target: one and the same φ gives $R = F$ *de novo* and $R \subsetneq F \cap \mathcal{B}_B(a)$ under editing (Theorem 2). Ehrlich is therefore sound for whole-sequence optimisers and unsound for edit-based ones, which is most of what it is used to benchmark. Separately, methods that mask over synthesis actions (Cretu et al., 2025; Seo et al., 2025; Koziarski et al., 2024) take the route itself as the specification, so a molecule with no legal construction order is correctly excluded for them and nothing here applies. The claim bites where a mask enforces an endpoint property and the construction order is an artefact of how the sampler builds it.

C The condition

A mask forbids every construction step that would leave the legal set, and it is then credited with delivering exactly the designs the constraint admits. That credit rests on a question no local test can answer. A mask reads the constraint off the state in hand, a verdict on the object as it now stands; safety asks whether a feasible design can still be completed from here without spending more than the budget allows. The two questions are one when an intermediate is a genuine prefix with its remaining coordinates still free, and they come apart on the construction graph of a rooted campaign, where an intermediate is a full hybrid of parent and target whose untouched positions are pinned to the parent. Appendix B measured the distance that mismatch opens between the designs a constraint admits and the designs a constrained sampler can build; this appendix says when the distance vanishes.

The answer is not a property of the constraint alone. Taking the construction graph as the setting, the condition splits into two independent properties: one of the predicate and its variable set (*extension tractability*, P1), one of the action space and its state encoding (*local soundness*, P2). The two environments we ship are the two one-sided ways of failing it. Where both hold at bounded width, the exact mask is cheap and a learned sampler buys neither support nor, for additive scores, optimisation; where P1 fails, the dynamic program that makes it cheap no longer applies, and for the budget band of Corollary 4 no exact mask can be evaluated in polynomial time unless $\mathsf{P} = \mathsf{NP}$. We use the vocabulary of supervisory control (Ramadge & Wonham, 1987): a mask is a *supervisor*, a state from which some *marked* design is still attainable is *co-accessible*, a supervisor that can strand a trajectory is *blocking*, and the *trim* of a graph is its restriction to accessible, co-accessible states. The results below establish that the trim exists and is unique, that computing it is the whole difficulty, and that the shipped mask is a strict subgraph of it.

C.1 The construction graph

Definition 1 (Mutation lattice)**.** Fix an alphabet $\mathcal{V}$ with $|\mathcal{V}| = v$, a length L, a *parent* $a \in \mathcal{V}^L$, and a mutation budget $B \le L$. A *state* is a complete sequence $s \in \mathcal{V}^L$, with *touched set* $T(s) = \{p : s_p \neq a_p\}$; write $s[p \to t]$ for s with position p set to t. The *structurally available* actions at s are

$$\mathcal{A}(s) \;=\; \big\{(p,t) \;:\; s_p = a_p,\; t \neq a_p,\; |T(s)| < B\big\},$$

together with a distinguished terminating action. A *trajectory* runs $a = s_0 \to \cdots \to s_k$ and terminates; s_k is its *design*. A predicate φ gives the *feasible set* $F = \{x : \varphi(x)\}$, and $\mathcal{B}_B(a) = \{x : d_H(x,a) \le B\}$ is the Hamming ball of radius B. We assume throughout that $\varphi(a)$: the parent is a design that has already been made.

Two features of Definition 1 carry the weight of everything that follows, and the action space fixes both; φ fixes neither. States are *complete* sequences: an intermediate is not a partial object with free coordinates but a full hybrid of parent and target, every unedited position pinned to the parent's token. The requirement $s_p = a_p$ then admits at most one substitution per position, so the lattice is the poset of sets $T \subseteq [L]$ with $|T| \le B$, decorated at each touched position with a token $\neq a_p$.

Lemma 1 (Memorylessness and grading)**.** *For any state s reachable from a by structurally available substitutions: (i) $T(s)$ and $\mathcal{A}(s)$ are determined by s alone, independently of the order in which s was built; (ii) every trajectory from a to s has length exactly $|T(s)|$, and exactly $|T(s)|!$ of them are structurally legal; (iii) the structurally reachable designs are $\mathcal{B}_B(a)$.*

Proof. $T(s)$ is read off s, and $\mathcal{A}(s)$ is defined from s and $|T(s)|$; this gives (i). A substitution at p makes $s_p \neq a_p$ permanently, since every later substitution requires its position to still agree with a, so touched positions are never revisited and $|T|$ rises by one per step. A trajectory reaching s therefore substitutes each $p \in T(s)$ once and nothing else, and any of the $|T(s)|!$ orders is legal because every proper prefix has rank $< |T(s)| \le B$; this gives (ii), and (iii) follows by applying it to each x with $d_H(x,a) \le B$. □

Every trajectory is therefore a *geodesic* in the Hamming graph, and the sampler is Markov on complete sequences with no path memory.

C.2 Masks, stop rules, and the two sets

Definition 2 (Mask and stop rule)**.** A *mask* assigns to each state a subset $M(s) \subseteq \mathcal{A}(s)$; a *stop rule* is a set σ of states *entitled* to terminate. The controlled graph $G(M)$ has an edge $s \to s[p \to t]$ for each $(p,t) \in M(s)$. A trajectory may terminate at $s \in \sigma$ and *must* terminate when $M(s) = \emptyset$ (a *forced stop*: a policy needs at least one action to normalise over). Masks are ordered pointwise; $M_\top(s) = \mathcal{A}(s)$ is the maximum and $\sigma_\top = \mathcal{V}^L$ the weakest stop rule. The *local* mask is $M_\varphi(s) = \{(p,t) \in \mathcal{A}(s) : \varphi(s[p \to t])\}$ and the *terminal* stop rule is $\sigma_\varphi = F$.

Definition 3 (Reachable set)**.** $R(M,\sigma)$ is the set of designs of trajectories in $G(M)$ that start at a and terminate, whether by entitlement or by force.

Definition 4 (Soundness and completeness)**.** (M,σ) is *sound* if $R(M,\sigma) \subseteq F \cap \mathcal{B}_B(a)$ and *complete* if $R(M,\sigma) \supseteq F \cap \mathcal{B}_B(a)$.

Soundness promises that every emitted design is admissible, completeness that no admissible design is excluded. Both are properties of the pair (M, σ); φ fixes neither, and the two are independent: each environment we ship holds exactly one of them.

Proposition 1 (The sandwich)**.** $R(M_\varphi, \sigma_\top) \subseteq F \cap \mathcal{B}_B(a) \subseteq R(M_\top, \sigma_\varphi)$*, both strict in general. The mask that checks φ at every intermediate is sound and incomplete; the mask that defers φ to the terminal is complete and unsound.*

Proof. Left. $\varphi(a)$ holds and every M_φ-edge enters a state satisfying φ, so by induction every state on a M_φ-trajectory is feasible, and lies in $\mathcal{B}_B(a)$ by Lemma 1(iii). This holds for every stop rule, entitled or forced, hence for $\sigma_\top$. *Right.* For $x \in F \cap \mathcal{B}_B(a)$, Lemma 1(iii) gives a structural trajectory to x, every edge of which lies in $M_\top$, and $x \in \sigma_\varphi$ entitles the stop. *Strictness.* Take $L = 2$, $\mathcal{V} = \{0, 1\}$, $a = 00$, $B = 2$, and let φ be an adjacency predicate given by a transition relation E — the predicate we actually ship. For $E = \{(0,0), (1,1)\}$, $F \cap \mathcal{B}_2(a) = \{00, 11\}$, but the substitutions out of 00 give 01 and 10, both infeasible, so $R(M_\varphi, \sigma_\top) = \{00\}$: the design 11 is feasible, in budget, and has no legal construction order. For $E = \{(0,0), (0,1), (1,0)\}$, $F \cap \mathcal{B}_2(a) = \{00, 01, 10\}$, but under $M_\top$ the trajectory $00 \to 10 \to 11$ exhausts the budget at 11, where σ_φ entitles no stop and nothing is available; it is force-stopped and emits the infeasible 11. □

Remark 1 (The two hypotheses of the left inclusion). The left inclusion used exactly two facts. Should $\varphi(a)$ fail, a is itself an infeasible terminal under $\sigma_\top$. Should the mask apply a test only *necessary* for $\varphi(s[p \to t])$, short of equivalent to it, a trajectory can pass every local test and then be force-stopped — which is what happens once stopping short of the budget is not entitled, that is, under `allow_stop_before_max=False` — on a design outside F. In the default configuration studied here (complete states, φ evaluated on the whole of each child, free stopping) neither escape is open, and the local check *is* sufficient for soundness. What the forced stop can still do there is emit a design carrying *fewer* substitutions than the protocol requires, a failure of grade: the design is feasible but under-mutated, a different defect from emitting a design outside F. Completeness occupies the remainder of this section.

C.3 The completion oracle and the trim

Proposition 1 locates the failure at the state. Evaluating φ on the child it is about to enter, a local mask decides a property of a total assignment whose untouched positions still carry the parent's token, where safety asks whether some feasible design agrees with the child wherever the child has been edited and stays inside the budget. No computation on the child alone resolves the second question, because the child pins coordinates a completion is free to set and, in the other direction, holds the parent at coordinates a completion must move. The object that asks it directly is the following.

Definition 5 (Budgeted completion oracle)**.** $C(s) = 1$ iff there is $x \in F$ with $x_p = s_p$ for every $p \in T(s)$ and $|T(x)| \leq B$. Put $M_C(s) = \{(p, t) \in \mathcal{A}(s) : C(s[p \to t])\}$.

C is the indicator of co-accessibility, and what M_C does *not* require is worth stating plainly: the child need not satisfy φ at all. An intermediate hybrid may be infeasible, provided a feasible completion of it survives.

Theorem 1. $R(M_C, \sigma_\varphi) = F \cap \mathcal{B}_B(a)$*, exactly.*

Proof. (⊇) Let $x \in F \cap \mathcal{B}_B(a)$ with $T(x) = T$, $k = |T| \leq B$. Fix *any* ordering $p_1, \ldots, p_k$ of T and set $s_i = s_{i-1}[p_i \to x_{p_i}]$, $s_0 = a$. Each step is structurally available (p_i untouched in s_{i-1}; $x_{p_i} \neq a_{p_i}$ as $p_i \in T$; $|T(s_{i-1})| = i - 1 < k \leq B$), and each child satisfies C with witness x itself, since x agrees with s_i on $T(s_i) \subseteq T$ and $|T(x)| \leq B$. So every edge lies in M_C, and $\varphi(x)$ entitles the stop at $s_k = x$. *(⊆)* Let x be the design of an M_C-trajectory. Every state on it satisfies C (children by definition of M_C, the root because $C(a) = 1$ with witness a under $\varphi(a)$) and lies in $\mathcal{B}_B(a)$. If the stop was entitled then $\varphi(x)$. Otherwise $M_C(x) = \emptyset$; take a witness $y \in F$ for $C(x) = 1$. If $y \neq x$ they differ at some $p \notin T(x)$, so $x_p = a_p$, $y_p \neq a_p$, hence $p \in T(y) \setminus T(x)$ and $|T(x)| < |T(y)| \leq B$. Then $(p, y_p) \in \mathcal{A}(x)$ and $C(x[p \to y_p]) = 1$ with witness y again, so $(p, y_p) \in M_C(x)$ — contradiction. Hence $y = x$ and $\varphi(x)$. □

Proposition 2 (Maximal permissiveness of M_C)**.** M_C *is sound and complete, and it contains every sound mask on the states that mask reaches: if (M, σ) is sound, then $M(s) \subseteq M_C(s)$ for every state*

s reachable from a in $G(M)$. Hence the restriction of $G(M_C)$ to the states reachable from a is the trim of the structural graph with respect to the marked set $F \cap \mathcal{B}_B(a)$, and M_C is the maximally permissive non-blocking supervisor.

Proof. Soundness and completeness are Theorem 1. Let (M, σ) be sound and s reachable from a in $G(M)$, and suppose $M(s)$ admits an edge into s' with $C(s') = 0$. Then s' is reachable as well, every design below s' agrees with s' on $T(s')$ and lies in $\mathcal{B}_B(a)$, so none is feasible, and every trajectory through s' terminates, since the rank rises and is bounded by B. Hence $R(M, \sigma) \not\subseteq F$, contradicting soundness. Edges at states that M never reaches from a leave $R(M, \sigma)$ unchanged, so soundness places no condition on them, and the containment is stated on reachable states for that reason. □

Corollary 1 (Trajectory count under M_C)**.** *Under M_C every ordering of a design's substitutions is legal, so a design with k substitutions has exactly $k!$ trajectories from a.*

Proof. The $\supseteq$ half of Theorem 1 fixed an arbitrary ordering and used one witness throughout; combine with Lemma 1(ii). □

The witness used there was the target itself, and the positions it still needed were exactly the untouched ones. That is why the once-per-position restriction, the one that looks binding, costs nothing once the mask is correct. The converse is easy to get wrong. Under M_φ, $k!$ is only an *upper* bound: a backward pass that correctly restricts to feasible parents admits at most k of them, so any exact P_B, or any L_1 distance derived from a trajectory count, must not assume $k!$ under intermediate masking.

Corollary 2 (Dependence of unreachability on the mask)**.** *For every $x \in F \cap \mathcal{B}_B(a)$ there is a sound mask under which x has a legal construction order, namely M_C. Hence "x has no legal construction order at any budget" is a statement about the local mask M_φ, never about the action space.*

For this reason Appendix B reported *no order exists under local masking*; the stronger word *unreachable* would conflate three causes otherwise conflated. A design may fail to be emitted because (i) $d_H(x, a) > B$, which no mask repairs; (ii) $x \in F \cap \mathcal{B}_B(a)$ but $x \notin R(M_\varphi, \cdot)$, an ordering obstruction that Corollary 2 removes by changing the mask; or (iii) M_C is not computable within the sampling budget, which is the only one of the three that is a property of the problem. The audit measured (ii), while the condition concerns (iii).

C.4 Extension tractability and local soundness

A natural summary of the condition would be that "φ factorises along the construction order". That summary is wrong, and our own predicate refutes it. Pairwise adjacency does *not* factorise along the edit order, since an edit at p is constrained by tokens later edits will overwrite, which is exactly the pathology of Proposition 1; yet its completion oracle is a polynomial dynamic program. Two properties are being conflated. What factorises is the *obstruction*, and factorisation and locality are separate hypotheses.

Definition 6 (P1: extension tractability)**.** Suppose φ is given as a conjunction of constraints, each on a subset of the positions, and let G_φ have one vertex per position and an edge between two positions that occur together in some constraint of that conjunction. Then φ has *extension tractability of width* w if G_φ has treewidth w.

Proposition 3. *If φ has width w and a tree decomposition of that width is given, then $C(s)$ is computable in time $O(L\, v^{\,w+1})$.*

Proof sketch. Freeze each $p \in T(s)$ to s_p by a unary constraint and leave each untouched p free with unary cost 1 on tokens $\neq a_p$ and 0 on a_p; the cost of an assignment satisfying φ then counts exactly the further substitutions, so $C(s) = 1$ iff the least such cost is at most $B - |T(s)|$. Min-sum message passing over the decomposition computes that least cost exactly: each of the $O(L)$ bags tabulates, for each of its v^{w+1} assignments, the least cost of its subtree, and a node with several children adds their tables entrywise, so no budget counter is carried and no convolution over budgets is needed at join nodes. □

P1 concerns φ and its variable set alone; it says nothing about how designs are constructed. Our adjacency predicate has G_φ a path, so $w = 1$ and C costs $O(Lv^2)$. Bounded width is sufficient for a cheap oracle and not necessary, because the width belongs to the conjunction chosen to represent φ: the single additive threshold of the GB1 folding gate (§3.3) couples every position, yet its completion test sorts the per-position decreases and takes the best $B - |T(s)|$ of them. What P1 asks for is bounded *width*, not bounded arity: 2-CNF has arity two, unbounded width, and a hard budgeted completion problem (Remark 4).

Definition 7 (P2: local soundness)**.** The action space satisfies *local soundness* for φ if $M_\varphi = M_C$: for every reachable s and structurally available (p, t), $\varphi(s[p \to t]) \Leftrightarrow C(s[p \to t])$. The cheap local test *is* the completion oracle.

P2 is not a property of φ but of what an intermediate state *is*, which the action space and the state encoding fix between them. The result that follows makes the distinction precise for one and the same predicate.

Theorem 2 (Scoping)**.** *Let φ be the pairwise adjacency predicate of a transition relation E in which every token has an E-successor and an E-predecessor, and let F be its feasible set. Then (a) under* de novo *left-to-right generation — states are prefixes, an action appends a token, terminals are the length-L strings — the local mask "the appended token is E-compatible with its predecessor" satisfies P2, and the reachable set is F exactly; while (b) under* edit-from-parent *on the lattice of Definition 1, P2 fails and $R(M_\varphi, \sigma_\top) \subsetneq F \cap \mathcal{B}_B(a)$ in general. Since φ, E and F are the same in (a) and (b), the gap is a property of the action space, not of the predicate.*

Proof. (a) A prefix satisfying every adjacency inside it is co-accessible: as every token has an E-successor it extends greedily to a length-L member of F. So the completion oracle is identically 1 on locally valid prefixes and 0 elsewhere, which is what the local mask computes; each $x \in F$ is generated along its own prefix chain, and a locally valid full-length string is in F. (b) is the first witness in the proof of Proposition 1, whose relation $E = \{(0, 0), (1, 1)\}$ has a successor and a predecessor for every token, so both parts speak about the same φ. □

What decides whether a setting is affected is the nature of an intermediate. Under (a) it is a *partial* assignment whose remaining coordinates are free, so local validity implies co-accessibility; under (b) it is a *total* assignment whose untouched coordinates are pinned to the parent, so passing the local test is sufficient for a feasible completion — the child witnesses its own — without being necessary for one. What (a) has and (b) lacks is prefix-closure; factorisation is not the distinction.

Remark 2 (Two settings outside the scope). Neither grammar-constrained decoding nor synthesis-aware generation is affected, for the reasons given in §A.5 and §A.7: an incremental parser is a co-accessibility oracle, so its support is exact and the pathology it leaves is distortion within that support (Park et al., 2024), while a mask over synthesis actions (Cretu et al., 2025; Seo et al., 2025; Koziarski et al., 2024) takes reachability as the specification. The gap is a defect only where the specification is F and the mask delivers $R(M_\varphi, \cdot)$.

C.5 Consequences for a learned sampler

Where P1 holds at bounded width the exact mask is polynomial in L and v, and the remaining question is what a learned sampler is left to supply. For our chain predicate that question admits a sharp answer, because the ordering obstruction of Theorem 2(b) is itself a binary constraint on adjacent positions.

Lemma 2 (Pairwise obstruction)**.** *Let φ be the adjacency predicate of E, with $\varphi(a)$. Then $x \in R(M_\varphi, \sigma_\top)$ iff (i) $\varphi(x)$, (ii) $d_H(x, a) \le B$, and (iii) $E(x_p, a_{p+1}) \vee E(a_p, x_{p+1})$ for every p with $p, p+1 \in T(x)$. Membership is decidable in $O(L)$ time.*

Proof. A substitution disturbs only the two adjacent pairs it sits between and the parent is feasible, so feasibility of an intermediate is the conjunction over adjacent pairs, each of which changes value only when one of its positions is substituted. A pair with an untouched position passes through only the parent and destination combinations, feasible by $\varphi(a)$ and $\varphi(x)$. A pair with both positions substituted additionally passes through exactly one mixed combination, decided by which is substituted first: (x_p, a_{p+1}) if the left one is, (a_p, x_{p+1}) if the right one is. Necessity of (iii) follows. For sufficiency,

read (iii) as a constraint on the relative order of p and $p+1$: if both disjuncts hold the pair is unconstrained, and if exactly one holds it orients the edge $\{p, p+1\}$. The constrained graph is a path, every orientation of a path is acyclic, so a topological order exists; realising it substitutes each position once, keeps every intermediate feasible, and uses $|T(x)| \leq B$ substitutions, after which free stopping emits x. □

Corollary 3 (No support advantage; for additive scores, no optimisation advantage)**.** *Suppose φ has width w (P1). (a) M_C is computable in $O(Lv^{w+1}B)$ per state with $R(M_C, \sigma_\varphi) = F \cap \mathcal{B}_B(a)$; since every sampler on this lattice has support inside $\mathcal{B}_B(a)$ and the target support is $F \cap \mathcal{B}_B(a)$, no sampler, learned or otherwise, attains a support that this mask does not already attain, and at bounded width it is computed in time polynomial in L and v. (b) If moreover φ is the chain predicate of Lemma 2* and the score is additive over positions*, $f(x) = \sum_p f_p(x_p)$, then $\max\{f(x) : x \in R(M_\varphi, \sigma_\top)\}$ and its argmax are computed exactly by a Viterbi dynamic program over states (position, token, substitutions used) in $O(Lv^2B)$.*

Proof. (a) is Proposition 3 with Theorem 1. For (b), Lemma 2 characterises $R(M_\varphi, \sigma_\top)$ by conditions on adjacent position pairs — φ itself, and the obstruction (iii), which depends only on (x_p, x_{p+1}) and the parent — plus the cardinality condition $d_H(x, a) \leq B$. So the reachable set is a regular language over positions intersected with a Hamming cardinality constraint, and maximising an additive score over it is a shortest-path computation on the product of the token automaton with a counter. □

Remark 3 (Scope of Corollary 3(b)). Part (b) is proved for *additive* scores only, and the restriction is not cosmetic: the Ehrlich reward used throughout this paper is a product of quantised maxima over motif placements and is not additive. An exact projection of this kind decodes an additive score exactly and does not cover the objective the campaign is scored on.

We read Corollary 3 as a prediction. Where P1 holds, the score is additive, and the decision is taken once, a learned sampler has nothing to buy, so any measured advantage must come from the residual: a non-additive reward or a batch-level objective. Section 3 measures that residual and finds it where the corollary requires it to be.

Corollary 4 (A predicate for which extension tractability fails)**.** *Let φ be the* integer-weighted budget band*: each substitution (p, t) carries an integer weight $w(p, t)$, and $\varphi(x)$ holds iff the total weight of x's substitutions lies in a fixed interval $[\ell, h]$. Its constraint graph couples every position, so P1 fails maximally — the width is $L - 1$ and Proposition 3 degenerates — while P2's cheap half holds: the local test is exactly φ evaluated on the child. Nevertheless*

- *(a) $M_\varphi \subseteq M_C$, strictly on some instances, and deciding C is* NP*-hard. At an* infeasible *child — one whose running total has left the band — C asks whether some assignment of the untouched positions brings the total back inside it, which is a subset-sum question over the untouched weights, subject to the residual budget. The local test refuses every such child; M_C admits exactly those from which a completion survives.*
- *(b) Deciding whether a legal construction* order *exists, i.e. membership of $R(M_\varphi, \sigma_\top)$, is* NP*-complete: membership in* NP *by Remark 5, and hardness by reduction from* PARTITION.

Hence, unless P = NP*, no mask for this predicate that is evaluable in polynomial time is both sound and complete (Proposition 5); in particular the exact mask M_C of Theorem 1, which part (a) shows is* NP*-hard to evaluate, cannot be computed at sampling time. Every tractable mask is unsound, incomplete, or approximate.*

The two halves are independent and both are needed. Part (a) forecloses the exact mask; part (b) says that even the incomplete mask's own support is not efficiently characterisable, so the shortfall cannot be audited away by a cleverer test. Part (a) also answers the objection that a predicate whose local mask is exact should have nothing to hide: exactness of the local test is not co-accessibility, and here the two come apart on the first move. Appendix D reports the size of the gap on the shipped band instance, where M_C reaches every feasible in-budget design by Theorem 1 while M_φ reaches a vanishing fraction of them.

Remark 4 (Scope of the band hardness, and a second witness). The band is *weakly* NP-hard, in that a pseudo-polynomial dynamic program over the counter exists, so that small integer weights are easy and the hardness is claimed only for weights written in binary. Two features of the modelled

setting make that dynamic program expensive in practice: weights recorded at realistic precision (energies in hundredths of a kilocalorie) span wide ranges, and two simultaneous bands (charge and hydrophobicity) force it to track both totals together, at a cost that grows with the product of their ranges. Neither strengthens the hardness beyond the weak sense. A second witness shows that the budget, not the weight encoding, is what makes completion hard: deciding whether a satisfying assignment lies within Hamming distance B of a *given* point is NP-complete already for 2-CNF, Horn and reverse-Horn formulas (Bailleux & Marquis, 1999), predicates whose *unbudgeted* completion problem is polynomial. That result is stated for an arbitrary reference point, where ours is feasible by the standing assumption $\varphi(a)$, which makes a its own witness and $C(a) = 1$ immediate; it therefore speaks to the budgeted completion problem in general and not to C at the root here.

The three options left open by Corollary 4 correspond to the two environments and the one method this paper studies: $M_\top$ accepts unsoundness and is worthless on its own (Proposition 1); M_φ accepts incompleteness and pays the support loss the audit measured; and the third is to approximate C. Building that third regime deliberately, Appendix D places a learned sampler where approximating C is unavoidable, alongside the exact bounded-depth masks swept there, which approximate it by another route. It is also the only predicate here whose completion oracle we cannot compute: the contact families of width two and three are not chains but remain within reach of Proposition 3, and the folding gate couples every position yet answers the completion question by sorting the per-position decreases (§3.3), so cause (iii) is instantiated in exactly one of them.

Remark 5 (Membership in NP). The effect of the once-per-position restriction on the complexity is worth stating, because the reconfiguration literature invites the wrong guess. General solution-graph st-connectivity is PSPACE-complete for unrestricted CNF (Gopalan et al., 2009), since a witnessing walk may be exponentially long. Here, however, every trajectory is a geodesic of length at most L (Lemma 1), so that the substitution order serves as a polynomial certificate: for a predicate evaluable in polynomial time, deciding $x \in R(M_\varphi, \cdot)$ is in NP, and C is in NP with the completion as witness. The restricted decision problem is therefore in NP, which does not make it tractable. For the same reason $R(M_\varphi, \cdot)$ is contained in the parent's connected component of the feasible subgraph of the Hamming graph without being equal to it in general, the two coinciding only under extra structure, which Appendix D makes precise for the predicates used there.

D THE INTRACTABLE REGIME

Every constrained task measured up to this point sits on the tractable side of the condition of Appendix C: the predicate's constraint graph is a path, the completion oracle is a dynamic program, and a sound and complete mask therefore exists in principle even where the shipped one is sound but incomplete. Cause (iii) of Appendix B, no exact mask computable at sampling time, was consequently instantiated nowhere in our suite, so the half of the condition that predicts where an exact mask is unavailable had no witness. This section builds one and measures the single quantity such a witness can settle: how much of the feasible set a local mask can still construct as the predicate's constraint graph thickens from a path to a complete graph.

D.1 FOUR TASKS ON ONE LANDSCAPE

We define four tasks on the same landscape, parent and protocol, varying only the feasibility predicate, so that the local test and the completion question come apart by degrees rather than at a single adversarial point. Three are contact families, a set of pairwise constraints over position pairs built by adding skips to a path, so that the induced width of the constraint graph is 1, 2 and 3 while the predicate stays the same kind of object at every task. The fourth is the integer-weighted budget band of Corollary 4, in which each substitution carries an integer weight and a design is feasible when its total lies in a calibrated interval, so that its constraint graph couples every position and its induced width is one below the sequence length. The width-one task is the shape the standard benchmark's own predicate has (Stanton et al., 2024), which makes the family a dial rooted at the shipped case rather than a pair of unrelated problems.

The instance is chosen so that nothing else can cause support loss. Ten positions over four tokens leave 4^{10} sequences, so both sets are enumerated in full; the instance's own transition constraint is set vacuous, so the declared predicate is the only feasibility rule; and the tasks are unbudgeted and

Table 7: The support study. Four tasks on one landscape, one parent and one protocol, differing only in the predicate. $|F|$ and $|R|$ are exact enumerations over the 4^{10} sequences, R being the set reachable under the local mask M_φ; the completion oracle mask reaches F exactly at every task (Theorem 1). Calibration targets the same feasible share at every task (Section D.2), and the achieved shares are 4.34%, 1.43%, 2.02% and 3.79% of the 4^{10} sequences, so the last column is read against unequal densities.

Task	Induced width	Path	$\lvert F\rvert$	$\lvert R(M_\varphi)\rvert$	$\lvert R\rvert/\lvert F\rvert$
`support-w1`	1	yes	45,493	39,843	0.876
`support-w2`	2	no	15,017	11,923	0.794
`support-w3`	3	no	21,150	13,879	0.656
`support-band`	9	no	39,753	18	0.0005

fixed-parent, so the support loss is induced by the local mask alone, ordering obstructions included, and not by cause (i) of Appendix B.

D.2 Equalising the feasible share

Comparing constraint structures requires controlling the feasible share, which differed substantially in our initial parameterisation. Holding the per-contact density fixed at 0.6 across tasks left $21{,}188$ designs feasible at width 1 against 61 at width 3, because a per-contact density compounds over the contact count and the contact count is what the width dial raises; the resulting reachable-to-feasible ratios were 0.667, 0.596, 1.000 and 1.000, a family varying its own difficulty rather than its structure. We therefore target the same *feasible share* across tasks instead, each task solving for the per-contact density that admits 2% of the sequence space; the shares actually achieved are 4.34%, 1.43%, 2.02% and 3.79%, and calibrate the band by exact convolution of the ten per-position weight distributions rather than by choosing a constant interval. Large integer weights spread the distribution of totals, so a narrow band is genuinely selective; that same range is what leaves the band's hardness weak in the sense of Remark 4.

D.3 Support loss as the constraint graph thickens

Table 7 reports the exact sizes of the feasible and reachable sets at each task. Along the contact family the ratio falls from 0.876 at width 1 to 0.794 at width 2 to 0.656 at width 3, a decline of 0.22 over two comparable steps; at the band the same ratio is 0.0005, 18 designs reachable of $39{,}753$ feasible, against $13{,}879$ of $21{,}150$ at the widest contact task. By Theorem 1 the completion-oracle mask reaches all $39{,}753$, so the gap lies between two masks on one predicate, not between a method and a bound.

The band's support loss is already visible at the first move. From the parent, 30 single substitutions are structurally available; the local mask admits the 4 whose resulting total stays inside the band and refuses the other 26, although a feasible in-budget completion of each exists, since a later substitution of the untouched positions can bring the running total back inside. What the mask cannot see at such a child is a subset-sum question over the untouched weights, the content of Corollary 4(a). The local test is exactly the predicate evaluated on the child, so the cheap half of local soundness (Definition 7) holds, and the two masks come apart on the opening move all the same. Corollary 2 applies here unchanged, so the $39{,}735$ designs the local mask cannot build are not unreachable; they have no legal construction order under that mask, and a mask under which they do is the object that cannot be computed.

D.4 Lookahead depth required by an exact mask

Between the local mask and the completion oracle sits a family of exact, untrained masks, and sweeping it prices the approximation Appendix C says must be made. A *depth-k lookahead* mask admits a move when the resulting state is feasible, or when some sequence of at most k further substitutions from it reaches a feasible design, so that depth 0 is M_φ, depth L is M_C, and the intermediate depths interpolate between the two masks of Proposition 1 without any training, and so without the confounds a trained arm carries. What makes the family a sweep rather than a list is its cost: a mask seeing k moves ahead enumerates both which of the u untouched positions to fill and

Table 8: The lookahead sweep on the codebook family. A depth-k mask admits a move when the resulting state is legal or when some sequence of at most k further substitutions reaches a legal design, so depth 0 is M_φ and depth 3 recovers F at every task. Entries are exact counts of the designs reachable under the mask, of $|F| = 800$, with the share of the feasible set in parentheses. Evaluating the mask enumerates position subsets as well as token assignments, up to $\sum_{j \le k} \binom{u}{j} v^j$ continuations per move over u untouched positions; early exit reduces the realised count.

Task	Depth 0	Depth 1	Depth 2	Depth 3
`support-sep2`	1	1	288 (0.360)	800 (1.000)
`support-sep3`	1	1	234 (0.292)	800 (1.000)
`support-sep4`	1	1	1 (0.001)	800 (1.000)

what to put in them, so it tests up to $\sum_{j \le k} \binom{u}{j} v^j$ continuations at every move — after one edit of a ten-position task over four tokens, $5{,}989$ at depth 3 — and the depth a predicate demands is the exponent in the price of soundness.

On the budget band the sweep is uninformative: depth 0 reaches the 18 designs of Table 7, but depth 1 already reaches $39{,}748$ of the $39{,}753$ feasible ones, because weights spanning $[-16, 16]$ against a narrow band make one substitution large enough to carry the running total back inside. Two independently drawn bands behave the same way, depth 1 reaching $28{,}741$ of $28{,}788$, since a state offers 30 distinct substitutions against two constraints and a move repairing both is almost always among them. That one move suffices is a property of these band instances and not of their size: a codebook holding only 0^{10} and 1^{10} is equally enumerable and admits nothing beyond the parent below depth nine. What the sweep shows here is that the shipped band leaves little for a deeper mask to recover, and the PARTITION reduction of Corollary 4 is a statement about the general case, untouched by this.

A predicate whose depth requirement is fixed by construction avoids that coupling, and the least distance between legal designs supplies one. We declared three further tasks on the same landscape, parent and protocol, whose predicate is a *codebook*: a set of designs any two of which differ in at least d positions, grown greedily around the parent from a fixed random stream, with the parent admitted first so the campaign is legal at its own starting point. Any two codewords differ in at least d positions and every state between them is illegal, so a mask seeing fewer than $d - 1$ moves ahead admits nothing beyond the parent; the separation therefore sets a lower bound on the depth at which support can appear rather than fixing it. The codebook holds 800 designs at every task, so the dial moves the depth requirement and leaves the size of the target alone.

Depth 2 is the column that carries the result: it separates the three tasks, while depth 3 recovers all 800 designs everywhere. The depth required is therefore a controlled property of the predicate and not of the instance's size, and because every task is fully recovered at depth 3, the dial moves where support appears and not how much of it exists.

One feature of the construction bounds how far that reading goes, and the same table shows it. The separation is enforced as a *minimum*, while random codewords over ten positions and four tokens sit at a typical distance of about 7.5, so the parameter controls the floor on the distance between legal designs rather than the typical distance between them. Depth 2 separates the tasks and depth 3 recovers all 800 designs at each of them; which pairs a shallow mask crosses first, and why the depth-3 neighbourhood already covers the codebook, would take a path analysis we do not run.

D.5 A LEARNED FLOW IN PLACE OF THE SEARCH

A learned flow could stand in for that search. Where the reward is one on legal designs and zero elsewhere, a state from which no legal design is reachable carries zero flow. Among states from which some are, a terminal's reward is divided over the several paths that reach it, so the flow is not the count of reachable legal designs and need not order states by it; whether the two track each other is an empirical question, and we measure it below. If they do, a policy trained to that reward holds, implicitly, an ordering of states by their completion prospects, the question the local mask cannot answer. Which objective supplies that flow matters: trajectory balance constrains one scalar per trajectory and sees signal only at the terminal, on a reward that is zero almost everywhere, so

Table 9: The fitted flow against the exact reachable count, on the codebook tasks. ρ is the Spearman correlation between the trained flow's completion term $\log \tilde{F}(s) = \log F(s) - \log R(s)$ and $\log N(s)$, the log number of legal designs reachable from s, over the live states ($N > 0$); the last two columns give the mean $\log \tilde{F}$ over live and over dead ($N = 0$) states. All mean $\pm$ s.d. over 100 seeds. `-L20` and `-L30` lengthen the separation-4 codeword to widen the range of reachable counts; each task is read over tens of thousands of enumerated and policy-visited states, most of them dead.

Task	L	$\rho(\log \tilde{F}, \log N)$	$\overline{\log \tilde{F}}$ live	$\overline{\log \tilde{F}}$ dead
`support-sep2`	10	0.80 ± 0.17	1.7 ± 1.0	-0.9 ± 0.2
`support-sep3`	10	0.82 ± 0.15	1.7 ± 1.2	-0.8 ± 0.3
`support-sep4`	10	0.87 ± 0.00	1.4 ± 0.1	-0.9 ± 0.1
`support-sep4-L20`	20	0.90 ± 0.00	9.9 ± 0.1	-1.2 ± 0.0
`support-sep4-L30`	30	0.92 ± 0.00	20.1 ± 0.1	-1.0 ± 0.0

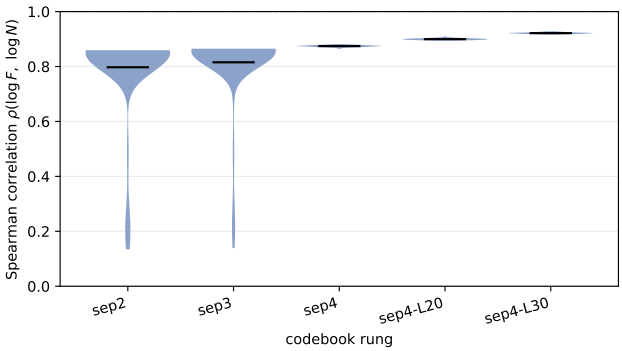


Figure 3: The rank correlation of Table 9 across all 100 seeds, one violin per task. It tightens from 0.80 on the shortest codeword to 0.92 on the longest, and holds across seeds on each task.

the flow at an intermediate state is fitted only against the trajectories that happen to pass through it, whereas the flow-as-oracle reading needs credit assigned *along* the trajectory — which is what forward-looking detailed balance provides, reparameterising $\log F(s) = \log R(s) + \log \tilde{F}(s)$ so the network learns the completion term $\log \tilde{F}$ directly.

We trained the forward-looking flow on each task and compared its fitted completion term $\log \tilde{F}(s) = \log F(s) - \log R(s)$ — the quantity the network parametrises, read off directly — against $\log N(s)$, the log of the exact number of legal designs reachable from that state, which the enumerable instance supplies. We evaluate the parent, all states within two substitutions of it, and states visited by the trained policy. This includes both live states ($N > 0$) and dead states ($N = 0$). To widen the range of reachable counts, we evaluate the three original tasks and two longer separation-4 codewords, at lengths 20 and 30. Table 9 gives the result: the rank correlation between fitted flow and exact reachable count is 0.80 on the shortest task and rises to 0.92 as the codeword lengthens and the range of counts grows, and the states from which some legal design survives carry far more flow than those from which none does, a separation that widens from about 2.6 to more than 21 nats.

Read as an ordering, the forward-looking flow tracks the reachable count (Figure 3): it ranks a state by how many legal designs lie beyond it and separates the live states from the dead, with ρ rising to 0.92 as the range of counts widens. This is a rank association only. It does not establish that the flow is calibrated to the reachable mass, nor that it allocates outgoing flow correctly across branches, and because a state's depth drives both quantities the association may in part reflect depth rather than completion structure; separating them would need a within-depth analysis and a held-out set of states the policy did not visit. The flow is therefore an exploratory ordering signal, and whether it could set a safe threshold for a mask is untested.

D.6 A POLICY-GUIDED BEAM AGAINST EXHAUSTIVE LOOKAHEAD

A useful ranking could reduce the number of branches explored at a given search depth. We test the ranking the trained sampler already provides: at each state we keep the eight actions the forward policy scores highest, which is the policy's own ordering and not the fitted completion term of Table 9, and gate the survivors on the same depth-2 test as Table 8. The parent offers 30 distinct substitutions, and the ranked slots also cover parent-token and stopping actions, so the width is a fixed count of actions and not a fixed share of the legal ones.

The beam reached 8 to 11 designs across the tasks where exhaustive depth 2 reached 234 to 288. A policy-guided beam therefore recovers far fewer feasible designs than exhaustive lookahead at the same depth. Whether the learned ranking beats an equally narrow random beam is untested, and repeated pruning over overlapping branches makes the retained share of actions an unreliable baseline for what a random beam would reach.

D.7 SCOPE OF THE SUPPORT STUDY

We do not claim the method works in this regime. The regime is built, the price of an exact mask in it is measured, and whether a learned flow can supply the missing oracle is left open: a beam of eight actions ranked by the forward policy recovers far less of the legal set than exhaustive search at the same depth, which agrees with the unsoundness Proposition 1 and the literature on terminally unrestricted flows both predict. A sampler that is not locally masked may learn to order the legal set without thereby supplying a mask that reaches it.

A follow-up would need to turn this ordering into a sampler that covers the legal set. It would also need to work at separation 4, where the exact alternative enumerates position subsets as well as tokens, not only on the two tasks where a depth-2 search recovers a third of the set. Ideally it would also run on an instance that is not enumerable, since the diagnostics that make the measurements here exact are the same property that lets a shallow lookahead succeed, and a method claim wants the regime where neither the enumeration nor the shallow search is available.

Two further limits bound the construction itself. First, the band's hardness is weak in the sense of Remark 4, and the shipped instance's weights lie in $[-16, 16]$, where a pseudo-polynomial dynamic program decides the oracle quickly: the instance demonstrates the support gap, the reduction from PARTITION demonstrates that no polynomial algorithm decides the general case, and neither statement carries the other. Second, all four tasks run on one Ehrlich landscape from a single generator family, so the scope caveat of Appendix B applies here in the stronger form that the family has one instance: the widths are a property of predicates we wrote, and how a contact family of a given width behaves on a different landscape, or under a predicate drawn from a different generator, is not measured.

E HARDNESS OF THE BUDGET BAND

Corollary 4 rests on the claim that, under an integer-weighted budget band, deciding whether a design admits a legal construction order is NP-complete. We checked that claim exhaustively before relying on it, over 1,977 PARTITION instances in both directions without a single disagreement. Exhaustive agreement on small instances is evidence rather than proof, and the reduction is short enough to give in full.

The predicate. Fixing a sequence of length L over an alphabet of size v, an integer weight $w(p, t) \in \mathbb{Z}$ is attached to each position–token pair, and a design x is feasible when its total $W(x) = \sum_p w(p, x_p)$ lies in a band $[\ell, h]$. Because every position enters one running total, the constraint graph over positions is complete and the induced width is $L - 1$, so Proposition 3 degenerates and the dynamic program it supplies is of no use here.

Theorem 3 (Hardness of monotone reachability under a budget band)**.** *Deciding, for a budget band φ, a parent a with $\varphi(a)$, and a target x with $\varphi(x)$, whether some ordering of the substitutions carrying a to x keeps every intermediate feasible is NP-complete.*

Proof. Membership in NP follows from Remark 5: positions on which a and x agree are never substituted and those on which they differ are substituted exactly once, so an ordering of the at most L differing positions is a certificate verifiable in $O(L \cdot |\varphi|)$.

For hardness we reduce from PARTITION. Given positive integers $a_1, \ldots, a_n$ with $\sum_i a_i = 2S$, we construct an instance on $L = n + 1$ positions over a binary alphabet. Every parent token carries weight 0, substituting position $i \leq n$ contributes a_i to the total, substituting the additional position z contributes $-S$, and the band is $[0, S]$. The parent is the design with nothing substituted, whose total is 0 and which is therefore feasible; the target is the design with every position substituted, whose total is $2S - S = S$ and which is feasible likewise. The construction is plainly polynomial.

Suppose first that a legal ordering exists, and let T denote the running total immediately before z is substituted. Every a_i is strictly positive, so the total increases monotonically until z is placed, and since each intermediate must lie in the band we have $T \leq h = S$. Substituting z then yields $T - S$, which must satisfy $T - S \geq \ell = 0$ and hence $T \geq S$. Combining the two inequalities gives $T = S$ exactly — so the positions substituted before z index a subset of $\{a_1, \ldots, a_n\}$ summing to S, and the PARTITION instance is a yes-instance.

Conversely, let $I \subseteq \{1, \ldots, n\}$ satisfy $\sum_{i \in I} a_i = S$. Substituting the elements of I in any order drives the total from 0 up to S through partial sums of positive integers bounded by S, each of which lies in $[0, S]$; substituting z next returns the total to 0; and substituting the remaining elements, whose weights also sum to S, drives it from 0 back up to S through intermediates bounded in the same way. Every intermediate is therefore feasible, and a legal ordering exists.

The two directions together give a polynomial-time many-one reduction from PARTITION, which is NP-complete, and hardness follows. □

The theorem above settles the construction-*order* problem, which is Corollary 4(b). The completion oracle of Corollary 4(a) is a different decision — whether a partial state admits any feasible in-budget completion at all — and we give it its own reduction, so that the claim "deciding C is NP-hard" carries a proof of its own; Proposition 5 then reuses its instance to rule out any polynomial mask that is both sound and complete.

Proposition 4 (Hardness of the completion oracle)**.** *For the integer-weighted budget band, deciding $C(s)$ — whether a state s within budget of the parent admits a feasible completion agreeing with it on $T(s)$ and within the mutation budget — is NP-complete.*

Proof. Membership in NP is immediate: a completion x is a certificate of length L, and checking $\varphi(x)$, agreement with s on $T(s)$ and $|T(x)| \leq B$ is linear. For hardness we reduce from SUBSET-SUM. Given positive integers $w_1, \ldots, w_n$ and a target t with $0 < t < \sum_i w_i$, build a band instance on $L = n + 1$ positions over $\{0, 1\}$ with parent $a = 0^L$. Every token 0 carries weight 0, a distinguished position z carries weight $w(z, 1) = t$, and each position $i \in \{1, \ldots, n\}$ carries weight $w(i, 1) = -w_i$; the band is the singleton $[\ell, h] = [0, 0]$ and the budget is $B = n + 1$. The parent has total 0 and is feasible, as the standing assumption requires. Consider the state $s = a[z \to 1]$, which substitutes only z: its total is $t > 0$, so s is infeasible, and $|T(s)| = 1$ leaves n of the budget. A completion of s substitutes a subset $P \subseteq \{1, \ldots, n\}$ of the untouched positions, reaching total $t - \sum_{i \in P} w_i$, and is feasible exactly when that total is 0, that is when $\sum_{i \in P} w_i = t$, the cardinality bound $|P| \leq n$ never binding. Hence $C(s) = 1$ if and only if the SUBSET-SUM instance is a yes-instance. The construction is polynomial and SUBSET-SUM is NP-complete, so deciding C is NP-hard, and with membership NP-complete. The reduction is on the completion question directly, and is independent of the ordering reduction of Theorem 3. □

Proposition 5 (No polynomial sound and complete mask)**.** *For the integer-weighted budget band, unless P = NP, no mask M evaluable in polynomial time is, on every instance and with some stop rule σ, both sound and complete.*

Proof. Take the instance of Proposition 4. Its parent $a = 0^L$ is feasible, and a design other than a is feasible exactly when it substitutes z together with some $P \subseteq \{1, \ldots, n\}$ with $\sum_{i \in P} w_i = t$, since without z every substitution lowers the total below the band. A substitution is never undone, so a trajectory that leaves a terminates, by entitlement or by force, at a design other than a. On a no-instance all such designs are infeasible, and a sound pair must therefore have $M(a) = \emptyset$: any

Table 10: EvoGFN against published methods. Win/tie/loss on `best`, 100 paired seeds per cell. The first three tasks carry a feasibility constraint; GB1 and TrpB are four-site landscapes (Wu et al., 2016; Johnston et al., 2024) with no constraint, and are the control on everything attributed to masking. *One `cmaes` seed exhausted its pool at 288 of 384 assays and returned no finite value, so that cell is over the 99 seeds both arms completed. †`pex` produced no feasible design on 9 of the 100 seeds of the sparse Ehrlich task, so that cell is over the 91 it did.

Task	Constraint	`genetic`	`cmaes`	`adalead`	`alde`	`mlde`	`pex`
`protocol-alde`	strict	100/0/0	53/34/13	100/0/0	100/0/0	100/0/0	99/1/0
`protocol-evolvepro`	strict	94/6/0	53/35/12	90/10/0	97/3/0	95/5/0	84/14/2
`ehrlich-feas`	strict	100/0/0	2/97/0*	100/0/0	100/0/0	100/0/0	91/0/0†
`gb1-anchor`	*none*	93/0/7	82/1/17	93/0/7	93/1/6	86/1/13	63/1/36
`trpb-anchor`	*none*	80/2/18	82/0/18	88/2/10	75/0/25	58/3/39	59/3/38

action at a would put an infeasible design into $R(M, \sigma)$. On a yes-instance some design other than a lies in $F \cap \mathcal{B}_B(a)$, and a complete pair must reach it, which takes an action at a, so $M(a) \neq \emptyset$. Evaluating M at a thus decides SUBSET-SUM, in polynomial time if M is evaluable in polynomial time. Maximal permissiveness is never used, so the conclusion covers every sound and complete mask. □

Scope of the result. PARTITION is NP-complete only in the weak sense — a dynamic program over the attainable totals decides it in time polynomial in the numeric value of the weights, which is exponential in their encoding length — and the reduction inherits that weakness, so instances whose weights are small admit a pseudo-polynomial procedure and are easy in practice. Remark 4 states what keeps that weakness from making the construction an artefact of unrealistically large numbers: physical quantities of the kind a band would express carry range in their own right, two simultaneous bands force the pseudo-polynomial program to track both totals at a cost growing with the product of their ranges, and the distance-satisfiability results of Bailleux & Marquis (1999) witness the same hardness, for an arbitrary reference point, for Krom, Horn and reverse-Horn formulas, whose *unbudgeted* completion problem is polynomial. The hardness therefore survives the choice of predicate, and none of these considerations strengthens it beyond the weak sense.

F EXTENDED RESULTS

This appendix gives the per-instance counts behind Section 3: the full win/tie/loss tables, the margin against the exact-projection instrument and what it cost to obtain, and the control that tests amortisation across rerootings.

F.1 FULL WIN/TIE/LOSS AGAINST THE PUBLISHED PIPELINES

Table 10 reports the paired win/tie/loss counts on `best` corresponding to the normalised means in Table 1, with 100 seeds per cell.

The margin against the instrument. Table 11 reports the paired difference in `best` between the selected EvoGFN and the exact-projection instrument, on 100 paired seeds per instance. EvoGFN has a positive mean margin in all eight cells — four Ehrlich instances under two protocols — with significance on three and a pooled interval excluding zero. The pooled margin is $+0.0315$; 313 of 800 seeds tie exactly, and five of the eight per-cell intervals include zero. The interval resamples seeds within these fixed cells, so it describes this set of instances and protocols, not a population of new landscapes. The margin is larger under the protocol with more, smaller rounds: pooled within protocol it was $+0.0167$ (CI $[-0.0008, +0.0342]$, $p = 0.096$) under three rounds of 21 and $+0.0462$ (CI $[+0.0272, +0.0659]$, $p < 10^{-5}$) under eight of 4. The two protocols differ in per-round budget as well as in round count, so neither is isolated as the cause.

Cost. Per campaign on the ALDE protocol, averaged over 100 seeds, EvoGFN spent 38,400 proxy evaluations and 332.8 CPU-seconds against the instrument's 0 and 3.1; on the EvolvePro protocol, 134,400 and 300.3 against 0 and 1.5. Across the eight instances the CPU ratio runs from 108× to

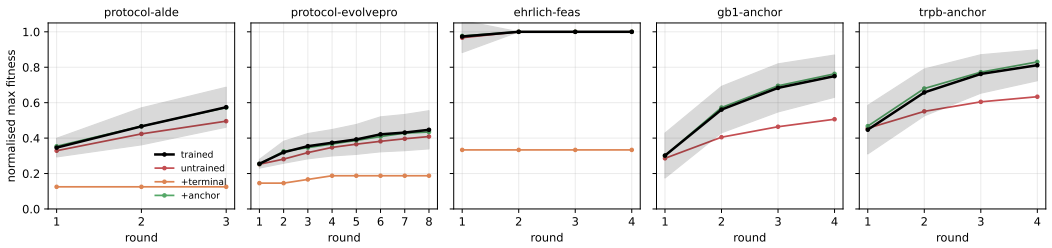


Figure 4: EvoGFN against its own ablations: the same policy left untrained, a mask variant (`+terminal` defers feasibility to the terminal state) and a policy variant (`+anchor` conditions the policy on the parent it edits from). Best-so-far normalised fitness against round, mean over 100 seeds, the trained arm with a ± 1 s.d. band. At matched proposal counts, the trained policy outperforms the untrained control on both protocol tasks and the empirical landscapes. Both reach the fitness ceiling on the feasibility stress test, the sparse Ehrlich task. The `+terminal` variant collapses where few sequences are feasible: deferring the check to the terminal state widens what the sampler may explore, and at these budgets it fails to locate feasible endpoints within it. `+anchor` tracks the trained arm throughout, so telling the policy which parent it edits from barely moves realised fitness on these tasks.

Table 11: Margin against the exact-projection instrument. Paired difference in `best` (EvoGFN minus instrument), 100 paired seeds per instance, bootstrap 95% CI over seeds; p is a two-sided sign test over non-tied seeds. Positive favours EvoGFN. The instrument is not a published baseline (§3.6).

Instance	Rounds × mutation budget	Mean Δ	95% CI	W/T/L	p
ALDE protocol: 3 rounds, 21 substitutions per round					
`protocol-alde`	3×21	$+0.0081$	$[-0.0225, +0.0387]$	32/43/25	0.43
`replicate-alde-i3`	3×21	$+0.0106$	$[-0.0275, +0.0500]$	26/48/26	1.00
`replicate-alde-i5`	3×21	$+0.0200$	$[-0.0181, +0.0575]$	36/36/28	0.38
`replicate-alde-i7`	3×21	$+0.0281$	$[-0.0056, +0.0619]$	32/47/21	0.17
pooled, ALDE	3×21	$+0.0167$	$[-0.0008, +0.0342]$	126/174/100	0.096
EvolvePro protocol: 8 rounds, 4 substitutions per round					
`protocol-evolvepro`	8×4	$+0.0369$	$[+0.0075, +0.0663]$	39/42/19	0.012
`replicate-evolvepro-i3`	8×4	$+0.0413$	$[-0.0050, +0.0881]$	42/29/29	0.15
`replicate-evolvepro-i5`	8×4	$+0.0587$	$[+0.0169, +0.0988]$	44/34/22	0.009
`replicate-evolvepro-i7`	8×4	$+0.0481$	$[+0.0119, +0.0844]$	45/34/21	0.004
pooled, EvolvePro	8×4	$+0.0462$	$[+0.0272, +0.0659]$	170/139/91	1.2×10^{-6}
pooled ($n = 800$)	—	$\mathbf{+0.0315}$	$[+0.0184, +0.0448]$	296/313/191	2.2×10^{-6}

$245\times$. The assay budget is fixed and identical across arms, so the projection's zero proxy calls buy it no extra experiments; the ratio is training and proposal compute, measured as CPU time for the whole campaign.

F.2 Amortisation of the policy across rounds

Amortisation could explain the advantage: carrying a trained policy across parent moves would allow later rounds to reuse earlier learning, a motivation for learned samplers in the GFlowNet literature. We tested it by re-initialising the policy at every rerooting, spending the same per-round training budget from a fresh start, and detected no difference: $23/52/25$ ($p = 0.885$) on the ALDE protocol and $25/53/22$ ($p = 0.771$) on the EvolvePro protocol. These results provide no evidence of a campaign-level benefit from retaining the policy between rounds; they do not establish whether useful information survives rerooting. Three explanations are consistent with the null and we do not separate them: the parent moves far enough each round (17.50 ± 3.69 of 21 positions on the ALDE protocol) that the reachable set is essentially new; the fitness predictor, refitted each round regardless, may carry whatever cross-round information exists; and three to eight rounds may be too few for transfer to pay against the cost of stale initialisation.