Monte Carlo Tree Diffusion with Multiple Experts for Protein Design

Xuefeng Liu[1]*, Mingxuan Cao[2]*, Songhao Jiang[1]†, Xiao Luo[3]†, Xiaotian Duan[1,4]†
Mengdi Wang[5], Tobin R. Sosnick[6], Jinbo Xu[3], Rick Stevens[1,4]
[1]Department of Computer Science, University of Chicago
[2]Data Science Institute, University of Chicago
[3]Toyota Technological Institute at Chicago
[4]Argonne National Laboratory
[5]AI Lab, Princeton University
[6] Department of Biochemistry and Molecular Biology, University of Chicago

Abstract

The goal of protein design is to generate amino acid sequences that fold into functional structures with desired properties. Prior methods combining autoregressive language models with Monte Carlo Tree Search (MCTS) struggle with long-range dependencies and suffer from an impractically large search space. We propose MCTD-ME, Monte Carlo Tree Diffusion with Multiple Experts, which integrates masked diffusion models with tree search to enable multi-token planning and efficient exploration under the guidance of multiple experts. Unlike autoregressive planners, MCTD-ME uses biophysical-fidelity-enhanced diffusion denoising as the rollout engine, jointly revising multiple positions and scaling to large sequence spaces. It further leverages experts of varying capacities to enrich exploration, guided by a pLDDT-based masking schedule that targets low-confidence regions while preserving reliable residues. We propose a novel multi-expert selection rule (P$\mathcal{H}$-UCT-ME) extends Shannon-entropy-based UCT to expert ensembles with mutual information. MCTD-ME achieves superior performance on the CAMEO and PDB benchmarks, excelling in protein design tasks such as inverse folding, folding, and conditional design challenges like motif scaffolding on lead optimization tasks. Our framework is model-agnostic, plug-and-play, and extensible to de novo protein engineering and beyond.

1 Introduction

Large generative models have demonstrated impressive capabilities across domains such as natural language processing [34] and molecular design [28]. Yet, conventional decoding strategies—greedy sampling, nucleus sampling, and beam search—remain fundamentally limited in tasks that require strategic planning or optimizing multiple objectives. In molecule or protein design, for example, a generative model must produce sequences that are not only syntactically valid but also meet biochemical, structural, and functional requirements [23]. Greedy decoding often gets stuck in suboptimal choices, and beam search—despite its popularity—offers little control over long-range dependencies or diverse trade-offs, especially in long-horizon domains like drug generation or code synthesis [3].

*Equal Contribution. Correspondence to: Xuefeng Liu <xuefeng@uchicago.edu>, Mingxuan Cao <caom@uchicago.edu>. † Equal contribution as second authors.

In contrast, planning algorithms such as Monte Carlo Tree Search (MCTS) offer adaptive search through a balance of exploration and exploitation. Since the introduction of UCT [13], MCTS has shown success in domains requiring sequential decision-making, most notably in AlphaGo [32]. Unlike static decoding, MCTS enables dynamic trajectory expansion, backtracking, and selective refinement, making it a promising mechanism to augment generative models with deliberative search. Recent frameworks such as Tree-of-Thoughts [39] and RethinkMCTS [15] illustrate this potential in language and code generation, where search paths can be evaluated and adjusted.

Large language models (LLMs such as GPT [25–27]) have recently been paired with MCTS to improve search in complex generation tasks. For instance, GPT-guided MCTS has enhanced search efficiency in symbolic regression [17], while value-guided MCTS decoding (PPO-MCTS [19]) improved the preferability of generated text over standard RLHF outputs. However, GPT-based MCTS approaches face notable pitfalls. Due to the autoregressive, token-by-token generation, even a single token error can derail an entire solution, and models often get stuck in suboptimal trajectories because their prior training biases lead to unbalanced exploration-exploitation. In other words, maintaining long-range coherence is challenging and it's difficult to revise parts of a candidate without regenerating the whole sequence.

Beyond these GPT-specific pitfalls, integrating MCTS into large-scale generative frameworks—particularly for biological design—faces major challenges as follows: First, unlike classical planning environments, we lack a tractable forward model to evaluate partially generated sequences. While recent approaches like Diffuser [11] show how diffusion models can operate as planners without explicit forward simulators, standard diffusion decoding is still largely open-loop. This limits its ability to adaptively revise candidates during generation. Second, real-world biological design tasks are intrinsically multi-objective. Protein design, for instance, requires satisfying multiple criteria, from structural compatibility to folding efficiency and evolutionary plausibility. Third, GPT-based generators make it inefficient to apply MCTS, especially for long-sequence tasks, resulting in unnecessarily large tree searches in both breadth and depth and inevitably exploring unpromising states. Finally, relying on a single GPT for exploration limits biophysical-fidelity, diversity, and constrains the search space due to biases from the pretraining dataset.

In this paper, we propose Monte Carlo Tree Diffusion with Multiple Experts (MCTD-ME), a novel planning framework that integrates diffusion-based generation with an MCTS search guided by a diverse ensemble of domain experts models. Motivated by the challenges of protein sequence design—where search spaces are vastly larger than small-molecule generation—we leverage diffusion models for their ability to efficiently denoise multiple positions in parallel, and use expert models to narrow the search space by injecting biologically grounded priors. Each node in our tree represents a partially denoised sequence (a partial plan), and expansion corresponds to one step of noise reduction. This structure imbues the diffusion process with MCTS's adaptive lookahead: evaluating, pruning, and refining paths to focus on promising candidates. In addition, we adopt an imitation-style rollout approach, where each expert proposes candidate completions based on the current partial sequence, thereby increasing diversity in the exploration space. These trajectories guide selection and backpropagation through the tree. Disagreement among experts encourages exploration; consensus promotes exploitation—enabling informed multi-objective planning at test time. We defer additional details on related work to Appendix A.6.

Summary of contributions: To our knowledge, this is the first work to incorporate an ensemble of expert into a diffusion-based planning framework, enabling multi-experts Monte Carlo tree search for complex generative problems. By unifying diffusion models with a multi-critic *evaluation* module and a multi-expert *proposal* module within MCTS, we introduce a general test-time strategy to steer generation toward goals that single decoders or static samplers struggle with—especially in large, structured domains like protein sequence design, where generation spaces are vast and require both parallel decoding and multi-perspective biophysical-fidelity-aware exploration.

Second, we propose several technical innovations to realize this integration:

1. A tree-structured diffusion process that expands and evaluates partial sequences through iterative *biophysical-fidelity-enhanced* denoising, achieved by masking low-pLDDT positions, the structurally uncertain regions, and guided by a multi-expert rollout mechanism, where each expert proposes candidate completions conditioned on the partial sequence, thereby enabling more diverse and meaningful exploration across a vast search space.
2. A novel P$\mathcal{H}$-UCT-ME selection algorithm, a multi-expert selection mechanism that arbitrates among experts using cached uncertainty statistics, yielding stable and efficient search,

extending the classic UCT formula with exploration bonuses based on expert disagreement (e.g., score variance or entropy), inspired by entropy-guided UCB [23].

Third, we demonstrate empirical gains on long-horizon generation tasks——including folding, inverse folding, and motif scaffolding — using the CAMEO 2022 benchmark [12] and a PDB date-split benchmark [5]. Across all settings, our framework consistently improves over strong baselines such as ESMFold [18], DPLM-2 [36], and ProteinMPNN [6]. Specifically, we achieve substantial RMSD reductions and TMscore enhancement in folding, higher sequence recovery (AAR) and structural fidelity (scTM) in inverse folding, and improved motif preservation in scaffolding—where our ensemble planner not only outperforms the baseline but also surpasses single-expert ablations of RFDiffusion [38], FoldFlow[4], and ProteinA [8]. These improvements are amplified on longer proteins and more challenging scaffolds, where uncertainty is concentrated and single models often fail.

Our results highlight several strengths: (i) pLDDT-guided masking focuses computation on low-confidence regions, improving efficiency and stability; (ii) multi-expert selection balances complementary strengths of diverse pretrained models, yielding robust gains even when some experts underperform individually; and (iii) the planner scales gracefully with larger search budgets, producing higher-quality and more diverse designs. Importantly, the method is model-agnostic—it can integrate any set of experts as black-box critics—making it broadly applicable across domains. Overall, this work advances the state-of-the-art in test-time guided generative planning, opening up promising directions for applications in drug discovery, program synthesis, and beyond.

2 Related works

Recent work has demonstrated the promise of diffusion models for protein and drug design, including structure-based backbones (RFdiffusion [37]), discrete sequence generation with versatile conditioning (DPLM and DPLM-2 [35, 36]), and property-guided atom–bond generation (DiffGui [10]), but these methods generally rely on one-shot decoding that lacks adaptive refinement. In parallel, MCTS has enhanced generative reasoning in domains such as code generation (AlphaCode [16], RethinkMCTS [15], Tree-of-Thoughts [39], PG-TD [42]), though primarily for autoregressive GPT-style models that struggle with long-range dependencies. Discrete diffusion provides a flexible backbone, as shown in text generation with absorbing-state noise (D3PM [1]) and recent masked-diffusion LMs that simplify and strengthen discrete denoising [31, 30], trajectory planning (Diffuser [11]), protein inverse folding (ProtInvTree [20]), and peptide optimization (PepTune [33]), yet existing methods typically depend on a single generator, limiting exploration. While mixture-of-experts strategies have been applied in RL and diffusion-based image generation [7, 14, 22], they remain static and one-shot during inference. Our approach overcomes these limitations by integrating multiple specialized expert generators within an MCTS-driven diffusion framework, enabling dynamic collaboration, adaptive search, and multi-objective optimization—significantly broadening exploration capacity and improving reliability in biomolecular generative design. (Additional details are provided in Appendix A.6.)

3 Preliminaries

Markov decision processes. We formulate generative planning as a finite-horizon MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, H, P, R \rangle$ [29], where $\mathcal{S}$ is the set of states (e.g., partial or completed sequences), $\mathcal{A}$ is the action space (e.g., denoising steps or token insertions), and $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ defines stochastic transitions. The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ provides a scalar evaluation, often defined only on terminal states. The policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ defines the action distribution at each state, such as a denoising model in diffusion or a next-token model in autoregressive generation. An episode corresponds to constructing a sequence $y = (y_1, ..., y_H)$, with a termination action marking the end of generation. We denote the action-value function $Q^\pi(s, a)$ as the expected cumulative reward of taking action a in state s under policy π, and the value function $V^\pi(s)$ as the expected return from s.

Diffusion-based generative model. We briefly review the diffusion framework for sequence generation. Let $y_{0:H}$ denote a token sequence of length H. A forward noising process $q(y_t \mid y_{t-1})$, for $t = 1, \ldots, T$, gradually corrupts the sequence, such that y_T becomes maximally noisy (e.g., fully masked for discrete data or Gaussian noise for continuous). A reverse denoising model $p_\phi(y_{t-1} \mid y_t)$ is trained to recover y_{t-1} from y_t, enabling sampling via iterative denoising from $y_T \sim p(y_T)$ down

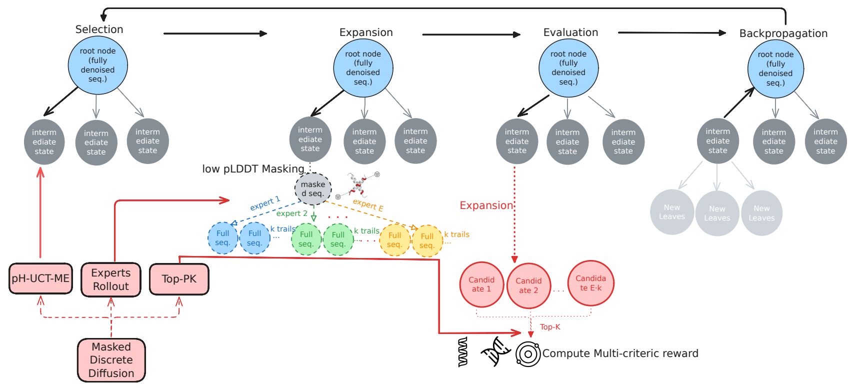

Figure 1: Overview of MCTD-ME in diffusion-based protein sequence generation. Nodes are partially denoised sequences expanded via masked diffusion and guided by multi-expert rollouts.

to $y_0 \sim p_\phi(y_0)$. Crucially, diffusion models naturally support guided generation. When an auxiliary score function $f(y)$ is available—e.g., a property predictor or constraint critic—generation can be biased toward desirable outputs. One classic approach is classifier guidance, where each reverse step is adjusted in the direction of $\nabla_{y_{t-1}} f(y_{t-1})$. Alternatively, reverse distribution can be reweighted as:

$$p_\phi^{\text{guide}}(y_{t-1} \mid y_t) \propto p_\phi(y_{t-1} \mid y_t) \cdot \exp\left\{\beta \cdot f(y_{t-1})\right\},$$

with β controlling guidance strength. In this work, we generalize the formulation to multiple experts setting: given K evaluators $\{C_k\}_{k=1}^{K}$, each scoring complete sequences generated by multiple experts $\{\pi^e\}_{e=1}^{E}$.

MCTS. Monte Carlo Tree Search (MCTS) is a planning algorithm that combines tree-based exploration with stochastic simulation to balance exploration and exploitation in large decision spaces. It proceeds in four stages: selection, expansion, evaluation, and backpropagation. Starting from the root (current state), the selection phase recursively chooses child nodes using an Upper Confidence Bound (UCT) criterion [13]:

$$\text{UCB} = Q(s,a) + c_p \cdot \sqrt{\frac{\log(N(s))}{N(s,a)}}, \tag{1}$$

where $Q(s,a)$ is the estimated reward, $N(s)$ and $N(s,a)$ are visit counts, and c_p controls exploration. When a leaf node is reached, expansion adds child nodes corresponding to unexplored actions. Then, these new nodes are evaluated by executing a rollout policy (e.g., sampling until termination or a predefined depth), producing a scalar reward. Finally, backpropagation propagates the reward up the visited path, updating $N(\cdot)$ and $Q(\cdot)$ statistics. Over repeated iterations, MCTS refines its value estimates and focuses the search on high-reward regions of the state space.

4 The MCTD-ME Algorithm

We propose Monte Carlo Tree Diffusion with Multi-Experts (MCTD-ME), which enhances protein generation by combining masked diffusion with an MCTS planner. An overview of the MCTD-ME workflow is shown in Fig 1. MCTD-ME structures diffusion as a tree, refining partially masked sequences to preserve scaffold integrity while systematically exploring uncertain regions. Search quality and diversity are further improved by incorporating multiple diffusion experts for diverse refinements and by using pLDDT-guided masking to target low-confidence residues, as illustrated in Fig 2. These design choices couple the planning strength of tree search with expert-aware diversification, yielding a principled framework for balancing exploration and exploitation in protein modeling. We present the algorithm in Algorithm 1 (deferred to Appendix A.1), with each rollout consisting of four key steps as following:

Selection. To navigate the space of protein sequences under the reverse *masked* diffusion framework, we define a tree where each node corresponds to a *complete* sequence (with no masked tokens) at a particular reverse step, and edges represent one reverse transition obtained by applying masked discrete diffusion to a low-confidence subset of tokens. At each iteration, we employ a novel selection algorithm, P$\mathcal{H}$-UCT-ME—a Policy(P)-guided entropy($\mathcal{H}$)-augmented variant of UCT—designed to identify the most promising path, where ME denotes Multiple Experts. Specifically, from a parent node s_t we choose the transition (i.e., reverse move to s_{t-1} via a particular mask set) that maximizes the following objective:

$$\text{P}\mathcal{H}\text{-UCT-ME}(s_t) \;=\; \arg\max_{a\in\mathcal{A}} \; \text{P}\mathcal{H}\text{-UCB-ME}((s_t,a))\,,$$

with

$$\text{P}\mathcal{H}\text{-UCB-ME}((s_t,a)) = Q(s_t,a) + c_p\,\frac{\sqrt{\log N(s_t)}}{1+N(s_t,a)}\cdot\left(\underbrace{w_{\text{ent}}\,\mathcal{U}_{\text{ent}}(s_t,a) \;+\; w_{\text{div}}\,\mathcal{U}_{\text{div}}(s_t,a)}_{\text{P}\mathcal{H}\text{-ME bonus (cached at expansion)}}\right).$$

Here, $Q(s_t,a)$ is the current value estimate, $N(\cdot)$ are visit counts, $c_p>0$ controls exploration, and $w_{\text{ent}}, w_{\text{div}} \geq 0$ weight the predictive-uncertainty and diversity bonuses, respectively. In our diffusion setting, the action prior is given by the reverse diffusion model $p_\phi(s_{t-1}\mid s_t)$ over masked edits, but—consistent with our implementation—we *do not* recompute its entropy during selection; instead we cache uncertainty statistics for each expanded child and reuse them during subsequent selections.

Multi-Expert P$\mathcal{H}$ (P$\mathcal{H}$-ME) bonus. Let $\mathcal{M}$ be the mask set chosen at s_t (e.g., via pLDDT), and let $\{p_\phi^{(e)}\}_{e=1}^{E}$ denote the E expert conditional distributions (softmax of logits) used during expansion. For the candidate child s_{t-1} obtained by action a, we define the ensemble-based uncertainty as

$$\mathcal{U}_{\text{ent}}(s_t,a) = \mathcal{H}\left(\frac{1}{E}\sum_{e=1}^{E} p_\phi^{(e)}(\,\cdot \mid s_{t-1},\mathcal{M})\right) - \frac{1}{E}\sum_{e=1}^{E}\mathcal{H}\Big(p_\phi^{(e)}(\,\cdot \mid s_{t-1},\mathcal{M})\Big)\,. \tag{2}$$

Here $\mathcal{H}(p) = -\sum_a p(a)\log p(a)$ denotes Shannon entropy. The first term is the entropy of the ensemble-averaged distribution $\bar{p}=\frac{1}{E}\sum_e p^{(e)}$, while the second is the average entropy of the individual experts. Their difference equals the mutual information (BALD score [9]), or Jensen–Shannon divergence between experts. Intuitively, it isolates epistemic uncertainty: if experts disagree, $\mathcal{H}(\bar{p})$ is large while the average $\mathcal{H}\left(p^{(e)}\right)$ remains small, giving a positive signal. If all experts are merely noisy but agree, the two terms cancel.

We also include a diversity measure, defined as the normalized Hamming distance between parent and child sequences:

$$\mathcal{U}_{\text{div}}(s_t,a) = \frac{1}{L}\sum_{i=1}^{L}\mathbf{1}\{y_i^{\text{child}} \neq y_i^{\text{parent}}\}. \tag{3}$$

In our multi-model setting, sequences may be batched and multi-channel (e.g., amino-acid tokens and structure tokens). We evaluate $\mathcal{U}_{\text{div}}$ over the task-relevant token subset: for *folding*, the difference is computed on structure tokens relative to the root structure; for *inverse folding*, on amino-acid tokens relative to the baseline sequence; and for *motif scaffolding*, on both channels (combined by concatenation or by averaging channel-wise normalized distances). When batching, we average the per-sequence $\mathcal{U}_{\text{div}}$ across the batch, thus the result remains in $[0,1]$. Both $\mathcal{U}_{\text{ent}}$ and $\mathcal{U}_{\text{div}}$ are computed once at expansion and cached for reuse during selection.

An action a is prioritized if it (i) has high estimated value $Q(s_t,a)$, (ii) corresponds to an under-explored branch (UCB exploration), (iii) induces strong ensemble disagreement ($\mathcal{U}_{\text{ent}}$ large), indicating information gain, and (iv) introduces sufficient novelty relative to the parent ($\mathcal{U}_{\text{div}}$ large). This encourages the search to balance reward exploitation with uncertainty- and diversity-driven exploration, while the reverse diffusion prior $p_\phi(s_{t-1}\,|\,s_t)$ maintains consistency with the denoising dynamics.

Remark 1 (single-expert case). When $E{=}1$, (2) vanishes ($\mathcal{U}_{\text{ent}}{=}0$). In MCTD-1, we therefore use the predictive (Shannon) entropy of the single expert over the masked sites as the uncertainty bonus:

$$\mathcal{U}_{\text{ent}}^{(1)}(s_t,a) = \frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathcal{H}\big(p_\phi(x_i \mid s_{t-1},\mathcal{M})\big),$$

which preserves the same units and interpretation (larger $\Rightarrow$ more uncertain edit). This value is computed at expansion and cached for reuse, akin to the entropy-guided UCB bonus used in ERP [23].

Dataset / Variant	ΔRMSD↑ (Å)	ΔTM-score↑	ΔReward (norm)↑	% Improved (RMSD)	% Improved (TM)	% Improved (Reward)
CAMEO (Folding evaluation)						
MCTD-ME-0 (Random)	$+1.046 \pm 0.675$	$+0.0032 \pm 0.0044$	$+0.0013 \pm 0.0029$	63.4%	63.4%	82.8%
Single-Expert (DPLM-2 150M)	$+4.104 \pm 0.509$	$+0.0391 \pm 0.0041$	$+0.0189 \pm 0.0035$	77.6%	77.6%	91%
Single-Expert (DPLM-2 650M)	$+4.767 \pm 0.528$	$+0.0451 \pm 0.0040$	$+0.0274 \pm 0.0028$	85.2%	85.2%	96%
Single-Expert (DPLM-2 3B)	$+4.402 \pm 0.508$	$+0.0401 \pm 0.0040$	$+0.0208 \pm 0.0033$	83.1%	82.5%	92.1%
MCTD-ME (Multi-Expert)	$+\mathbf{4.728 \pm 0.516}$	$+\mathbf{0.0462 \pm 0.0042}$	$+\mathbf{0.0274 \pm 0.0034}$	**88.0%**	**87.4%**	**97.6%**
PDB (Folding evaluation)						
MCTD-ME-0 (Random)	$+5.785 \pm 0.750$	$+0.0255 \pm 0.0055$	$+0.0153 \pm 0.0036$	90.2%	91.5%	88.2%
Single-Expert (DPLM-2 150M)	$+7.970 \pm 0.942$	$+0.0400 \pm 0.0069$	$+0.0240 \pm 0.0047$	90.3%	93.2%	96.1%
Single-Expert (DPLM-2 650M)	$+7.807 \pm 0.763$	$+0.0387 \pm 0.0058$	$+0.0232 \pm 0.0038$	93.8%	95.7%	94.4%
Single-Expert (DPLM-2 3B)	$+\mathbf{8.805 \pm 0.818}$	$+0.0416 \pm 0.0059$	$+0.0250 \pm 0.0040$	**95.4%**	92.4 %	96.6%
MCTD-ME (Multi-Expert)	$+8.419 \pm 0.925$	$+\mathbf{0.0435 \pm 0.0069}$	$+\mathbf{0.0261 \pm 0.0047}$	92.4%	**96.7%**	**96.7%**

Table 1: **Lead improvements in folding task.** We frame evaluation as inference-time lead optimization: the *baseline* is the one-shot ESMFold structure on the native sequence (the lead), and the results presented in the table are after MCTD-ME planning. All metrics are computed against the native backbone. $\{\Delta\text{RMSD}, \Delta\text{TM-SCORE}, \Delta\text{REWARD}\}$ denote the improvement over the initial lead for each metric. % indicates the percentage improvement over the lead after planning. Standard errors are computed across n targets (CAMEO: n=183; PDB: n=449). Boldface marks the best result within each dataset block.

Expansion. When P$\mathcal{H}$-UCT-ME selects a leaf node s_t, representing a complete sequence y_t, we expand it by generating candidate children via *masked diffusion*. Our expansion departs from standard MCTS in two important ways: (1) structure-aware masking via pLDDT scores to enhance biophysical fidelity, and (2) multi-expert imitation-guided rollouts to improve diversity and quality.

Progressive pLDDT Masking for Targeted Denoising. Rather than resampling all positions uniformly, we exploit predicted structural confidence (pLDDT) to target only low-confidence residues. Specifically, at reverse step t we define a mask $\mathcal{M}t$ by thresholding the pLDDT profile of y_t, masking unstable regions while preserving high-confidence subsequences. Crucially, the masking threshold decreases over diffusion steps, progressively reducing the fraction of masked residues as denoising proceeds. This "progressive masking" strategy focuses computation on the most uncertain regions early on, while stabilizing promising motifs in later stages:

$$y_{t-1} \sim p_\phi(y_{t-1} \mid \text{Mask}(y_t; \mathcal{M}_t)). \tag{4}$$

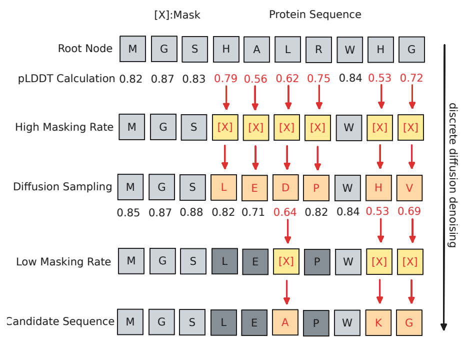

Figure 2: Masked discrete diffusion step: low-confidence sites are resampled while confident tokens stay fixed, progressively "unmasking" the sequence.

Mixture-over-experts view. Conceptually, expansion implements a guided sampler that averages proposal distributions from E pretrained experts and biases them by a scalar reward R:

$$p^{\text{plan}}(y_{t-1} \mid y_t) \;\propto \left[\sum_{e=1}^{E} w_e p^{(e)} \phi(y_{t-1} \mid y_t, \mathcal{M}_t)\right] \cdot \exp\{\beta, R(y_{t-1})\},$$

where $p_\phi^{(e)}$ is expert e's conditional, w_e are mixing weights, and β controls guidance. In practice we realize this by *sample–score–select* (below) rather than step-wise gradient injection.

Multi-Expert Guided Expansion via Rollouts. To enrich proposal diversity and imitate the training distribution, we leverage multiple experts during expansion. Given a masked sequence $\text{Mask}(y_t; \mathcal{M}_t)$, each expert $\pi^e \in \{1, \ldots, E\}$ performs k rollouts by sampling its conditional distribution $p_\phi^{(e)}(\cdot \mid y_t, \mathcal{M}_t)$. This yields a pooled candidate set

$$\mathcal{C}_t \;=\; \bigcup_{e=1}^{E} \, \bigcup_{r=1}^{k} \, y_{t-1}^{(e,r)}, \tag{5}$$

where $y_{t-1}^{(e,r)}$ is formed by filling masked sites and splicing back into y_t. We de-duplicate $\mathcal{C}_t$, compute a composite score $R(\cdot)$ (see Evaluation) per candidate, cache the results, and retain the top-K children by R. Each kept child stores two exploration bonuses computed once at expansion—ensemble surprisal $\mathcal{U}_{\text{ent}}$ and novelty $\mathcal{U}_{\text{div}}$—which are reused during P$\mathcal{H}$-UCT-ME selection.

Variant	AAR ↑			Normalized Reward ↑			scTM ↑		
	Baseline	Final	Δ	Baseline	Final	Δ	Baseline	Final	Δ
CAMEO Benchmark									
Single-Expert (150M, Baseline)	0.4425 ± 0.1190	–	–	0.3893 ± 0.1326	–	–	0.3701 ± 0.1528	–	–
MCTD-0 (Random)	–	0.4250 ± 0.009	-0.0175 ± 0.0145	–	0.3726 ± 0.008	-0.0167 ± 0.0130	–	0.3763 ± 0.014	$+0.0062 \pm 0.0560$
Single-Expert (650M)	–	0.4535 ± 0.1162	$+0.0110 \pm 0.0149$	–	0.3974 ± 0.1010	$+0.0081 \pm 0.0148$	–	0.3621 ± 0.1515	-0.0080 ± 0.0678
Single-Expert (ProteinMPNN)	–	0.4326 ± 0.1102	-0.0098 ± 0.0194	–	0.3786 ± 0.1026	-0.0107 ± 0.0304	–	0.3618 ± 0.1468	-0.0070 ± 0.0691
MCTD-ME (Multi-Expert)	–	$\mathbf{0.4667 \pm 0.1124}$	$\mathbf{+0.0237 \pm 0.0238}$	–	$\mathbf{0.4132 \pm 0.1119}$	$\mathbf{+0.0239 \pm 0.0461}$	–	$\mathbf{0.4204 \pm 0.1585}$	$\mathbf{+0.0503 \pm 0.0505}$
PDB Benchmark									
Single-Expert (150M, Baseline)	0.5037 ± 0.1196	–	–	0.4629 ± 0.0763	–	–	0.3804 ± 0.1575	–	–
MCTD-0 (Random)	–	0.4876 ± 0.1119	-0.0160 ± 0.0154	–	0.4591 ± 0.0751	-0.0032 ± 0.0157	–	0.3773 ± 0.1585	-0.0030 ± 0.0505
Single-Expert (650M)	–	0.5130 ± 0.1208	$+0.0109 \pm 0.0141$	–	0.4751 ± 0.0812	$+0.0122 \pm 0.0134$	–	0.3817 ± 0.1594	$+0.0022 \pm 0.0490$
Single-Expert (ProteinMPNN)	–	0.4882 ± 0.1112	-0.0155 ± 0.0163	–	0.4594 ± 0.0748	-0.0029 ± 0.0143	–	0.3774 ± 0.1575	-0.0030 ± 0.0596
MCTD-ME (Multi-Expert)	–	$\mathbf{0.5244 \pm 0.1350}$	$\mathbf{+0.0213 \pm 0.0139}$	–	$\mathbf{0.4828 \pm 0.0900}$	$\mathbf{+0.0199 \pm 0.0146}$	–	$\mathbf{0.3827 \pm 0.1549}$	$\mathbf{+0.0033 \pm 0.0410}$

Table 2: **Inverse folding results on CAMEO and PDB.** Baseline sequences are generated by one-shot DPLM-2 (150M) , which serves as the initialization for our lead-optimization procedure. *Baselines* may vary slightly across runs (see baseline-invariant results in Appendix **??**), while *Final* denotes the sequence after MCTD-ME refinement. Metrics reported include amino acid recovery (AAR), normalized reward, and scTM, with Δ indicating improvements over baseline. Means and uncertainties are reported as mean $\pm$ s.e.m. computed across targets (CAMEO: n=183; PDB: n=449). Boldface marks the best result within each dataset block.

Candidate Proposal Strategy. To retain diversity while prioritizing promising edits, we rank all expanded children and select the top-K candidates:

$$\{y_{i=1}^{K}\} = \text{Top-K}\left(\{y_{t-1}^{(j)}\}, K\right). \tag{6}$$

Here, j indexes all candidates pooled across experts and rollouts at step t; Top-K ranks by the composite score R and returns the best K. Each chosen child is inserted into the tree as a new node, annotated with its cached P$\mathcal{H}$-ME bonuses ($\mathcal{U}_{\text{ent}}, \mathcal{U}_{\text{div}}$) and expert rollout score. This beam-style expansion ensures that multiple high-quality hypotheses are pursued in parallel, enabling MCTD-ME to balance exploitation of strong candidates with exploration of diverse structural alternatives.

Evaluation. Whereas expansion uses experts to generate proposals, evaluation uses a panel of critics to score complete sequences. Because every node in the tree corresponds to a fully denoised sequence y_0, evaluation can be performed immediately without further decoding. Each sequence is scored by critics $C_1, \ldots, C_J$, producing

$$C_j(y_0) \coloneqq R_j(y_0), \quad j = 1, \ldots, J. \tag{7}$$

We normalize critic outputs to $[0, 1]$ and aggregate via a fixed convex combination

$$R(y_0) = \sum_{j=1}^{J} w_j, C_j(y_0), \qquad w_j \geq 0, \sum_{j=1}^{J} w_j = 1, \tag{8}$$

then cache $R(y_0)$ for re-use in selection/backprop. This separation—experts for proposals, critics for scores—implements a tractable, planning-based approximation to the guided reverse process sketched above.

Backpropagation. The expert rewards obtained at s_{t-1} are propagated back to update values along the path to the root. For each state–action pair (s_i, a_i) on the path, we update:

$$Q(s_i, a_i) \;\leftarrow\; \max\{Q(s_i, a_i),\; r_H\}, \tag{9}$$

where r_H is an aggregated expert score (e.g., the max across $\{R_e\}$). This max-based backup, consistent with our implementation, favors exploiting high-reward branches while maintaining the ability to explore alternative completions. Because each node is a complete sequence, long-range credit assignment is naturally aligned with the denoising trajectory.

Lead optimization→De Novo Generation. The framework, though implemented as complete-sequence lead optimization, naturally extends to de novo generation by treating the root as a fully masked sequence and progressively unmasking subsequences with similar reward-guided search dynamics and performance. (We defer further details to §A.5.)

5 Experiments

5.1 Experimental Configuration

Models, experts, and baselines. We instantiate MCTD-ME with masked *discrete* diffusion as the rollout engine and vary the *proposal experts* by task. Sequence experts are pre-trained DPLM-2 models [35] of different capacities (150M, 650M, 3B). For motif scaffolding we consider structure-aware experts (RFDiffusion [38], ProteinA [8], FoldFlow [4]) both as single-expert baselines and as members of a multi-expert ensemble. All experts propose from the *same* masked input; a single composite critic scores candidates. Baselines are task-specific: *Folding* (seq$\to$struct) uses one-shot ESMFold [18] on the native sequence (RMSD/TM vs. ground truth), while our planners propose structure-token edits via DPLM-2 conditioned on the sequence and compare directly to the reference backbone. *Inverse folding* (struct$\to$seq) uses one-shot DPLM-2 (150M) conditioned on backbone tokens; MCTD-ME performs masked amino-acid fills given the backbone. *Motif scaffolding* (motif $\to$scaffold) uses one-shot DPLM-2 (150M) co-generating sequence/structure with motif residues frozen; MCTD-ME edits scaffold positions (sequence and/or structure, depending on expert) while preserving the motif. In this study, we frame evaluation as *lead optimization*: starting from a one-shot initial *lead* and refine it. We compare MCTD-0 (random fill; no experts), MCTD-1 (single proposal expert; separate runs per expert), and MCTD-ME (multi-expert ensemble).

Datasets and splits. We evaluate on two benchmarks used: CAMEO 2022 [12] and a PDB date-split [5]. We follow the standard splits and filtering; statistics appear in Appendix A.2 (CAMEO: 183 targets, length $15/247.5/704$ min/avg/max; PDB date-split: a 449-target subset, length $2/242.2/511$). Ground-truth backbones and native sequences are available for both.

Critics and evaluation metrics. *All* planner variants share the *same* task-specific composite critic; for each target we report Baseline, Final, and Δ (Final–Baseline) and aggregate by mean $\pm$ s.e.m., mapping scalars to $[0,1]$ (pLDDT residue-averaged and divided by 100). The *folding* critic mixes TM-score with an inverted global Cα RMSD and, when available, mean pLDDT; the *inverse folding* critic amino-acid recovery (AAR) against the native sequence with a self-consistency TM-score (scTM) computed between the target backbone and the structure predicted from the designed sequence, plus a small biophysical bonus. For *motif scaffolding*, candidates must exactly preserve the motif, after which we score by scaffold pLDDT, a motif-aligned RMSD transformed to $[0,1]$ to emphasize sub-Å accuracy, and scTM, with a modest success bonus at strict thresholds. Full formulas and weights appear in Appendix A.3.

5.2 Protein Folding

We first evaluate MCTD-ME on the folding task: given an amino acid sequence, predict its 3D structure. We use the CAMEO 2022 [12] and a PDB date-split benchmark [5]. As the initial *lead*, we take the one-shot ESMFold structure of the native sequence and measure improvement against the true structure (backbone RMSD, TM-score). We compare three variants: (i) MCTD-0 (random) fills masked sites with random amino acids (no expert proposals); (ii) MCTD-1 (single-expert) uses one pretrained DPLM-2 model to generate rollouts; we report three single-expert variants separately, using capacities 150M, 650M, and 3B; and (iii) MCTD-ME (multi-expert) aggregates proposals from three experts (DPLM-2 150M, 650M, 3B). All variants share the same critic; only the proposal policy differs.

Results are shown in Table 1. MCTD-0 yields only slight improvements over baseline (near-random structure changes), whereas incorporating learned guidance dramatically reduces RMSD. On CAMEO, MCTD-ME attains the largest average RMSD reduction and TM gain (e.g., ΔRMSD $\approx +4.77$ Å, ΔTM $\approx +0.046$), marginally surpassing the best single-expert planner. Similar trends hold on the PDB split, where both the top single expert and MCTD-ME achieve large RMSD reductions, with MCTD-ME delivering the highest TM increase. We also observe a higher fraction of targets improved by MCTD-ME (e.g., 88.0% on CAMEO vs. 85.2% for the best single expert). Unless noted, we report *lead-optimization deltas*: ΔRMSD $=$ Lead $-$ Final (higher is better) and ΔTM $=$ Final $-$ Lead (higher is better), confirming that multi-expert proposals plus MCTS-based selection consistently refine the initial ESMFold lead.

5.3 Protein Inverse Folding

We evaluate MCTD-ME on structure-conditioned sequence design (CAMEO 2022 and a PDB date-split with known natives), reporting amino-acid recovery (AAR; fraction matching native) and self-consistency TM-score (scTM; TM between the target backbone and the fold of the designed sequence) following Wang et al. [36]. We ablate proposal experts: MCTD-0 (no experts; random fills), MCTD-1 (single proposal expert; separate runs with DPLM-2 150M, DPLM-2 650M, and ProteinMPNN [6]), and MCTD-ME (ensemble of those experts). All proposals are scored by the same critic with reward $R = 0.60\,\mathrm{AAR} + 0.35\,\mathrm{scTM} + 0.05\,B$ (small biophysical bonus B), and sequences are completed via masked-diffusion rollouts within MCTS.

Variant	Motif RMSD (Å) ↓			scTM ↑			Reward (norm) ↑		
	Baseline	Final	Δ	Baseline	Final	Δ	Baseline	Final	Δ
Baseline (DPLM-2 150M one-shot)	**6.575 Å ± 1.628 Å**	–	–	**0.417 ± 0.213**	–	–	**0.393 ± 0.183**	–	–
Single-Expert (DPLM-2 650M)	–	5.988 Å ± 0.943 Å	+0.583 Å ± 0.545 Å	–	0.449 ± 0.048	+0.030 ± 0.022	–	0.417 ± 0.162	+0.024 ± 0.013
Single-Expert (RFDiffusion)	–	5.105 Å ± 0.659 Å	+1.470 Å ± 0.876 Å	–	0.315 ± 0.050	−0.086 ± 0.027	–	0.397 ± 0.160	+0.004 ± 0.017
Single-Expert (FoldFlow)	–	7.478 Å ± 1.278 Å	−0.903 Å ± 0.800 Å	–	0.333 ± 0.044	−0.076 ± 0.028	–	0.379 ± 0.157	−0.015 ± 0.007
Single-Expert (ProteinA)	–	4.917 Å ± 1.148 Å	+1.658 Å ± 1.011 Å	–	0.323 ± 0.054	−0.079 ± 0.020	–	0.405 ± 0.159	+0.012 ± 0.010
MCTD-ME (Multi-Expert)	–	**5.300 Å ± 0.675 Å**	**+1.275 Å ± 0.453 Å**	–	**0.442 ± 0.049**	**+0.025 ± 0.020**	–	**0.481 ± 0.158**	**+0.088 ± 0.028**

Table 3: **Motif scaffolding results (mean ± s.e.m.; n=17 motifs).** The first row reports the *Baseline* from one-shot DPLM-2 (150M). Variant rows show the identical baseline values (left columns), the *Final* performance after MCTD-ME refinement, and the improvement Δ (for RMSD, Δ=Baseline−Final in Å; for scTM/Reward, Δ=Final−Baseline). The multi-expert planner yields consistent gains in motif preservation (lower RMSD) and fold quality (higher scTM and reward).

As shown in Table 2, MCTD-ME consistently improves structural fidelity and overall reward: on both CAMEO and PDB it delivers the largest scTM and normalized-reward gains over baselines, while AAR is *maintained* on CAMEO and *increases* on PDB. Random edits (MCTD-0) degrade AAR and yield only small structural gains, confirming the need for guided proposals. Single-expert planners track their proposal models—e.g., 150M preserves its baseline; 650M improves both scTM and AAR—while ProteinMPNN alone can trade off AAR for scTM. The ensemble's advantage grows on harder cases (e.g., longer proteins), reflecting benefits from expert diversity under a common critic. See Appendix A.4 (Fig. 3) for AAR, reward, and scTM improvements across protein-length bins.

5.4 Motif Scaffolding

Finally, we test conditional generation-motif scaffolding: given a functional motif (its amino acid sequence and 3D structure), generate the remaining scaffold sequence and structure around it. Following prior work (e.g., FrameFlow [40]), we use a PDB-sourced benchmark (EvoDiff's ∼24 curated motifs[41]). The baseline model is DPLM-2 (150M) run in conditional mode to co-generate scaffold sequence/structure given the motif. Our planner uses the same MCTS budget and masking schedule as in other tasks (Appendix A.8): at each expansion, only scaffold positions are masked and proposed. We ablate proposal experts: single-expert MCTD-ME with DPLM-2-650M, RFDiffusion [37], ProteinA [8], or FoldFlow [4]; and the multi-expert ensemble using all four. Across all variants, candidates are scored by the same composite critic, and the motif residues are hard-frozen throughout generation. Metrics are: motif-RMSD (RMSD of the designed scaffold after rigidly aligning on the motif), scTM (TM-score between the designed full backbone and native), and mean pLDDT over scaffold residues.

Table 3 summarizes averages over 17 motifs. The MCTD-ME achieves the strongest overall performance—consistently lower motif-RMSD, higher scTM, and higher reward—improving 82.4% of motifs. Single-expert planners show heterogeneous behavior: ProteinA often attains the largest RMSD drops on long scaffolds but can reduce scTM; RFDiffusion lowers RMSD yet sometimes trades off fold consistency; FoldFlow underperforms on average; DPLM-2-650M yields balanced but modest gains. These instance-level trade-offs are mitigated by the ensemble: aggregating diverse proposal experts under a shared critic yields robust improvements across motifs, rather than excelling only on specific cases. Overall, MCTD-ME substantially outperforms one-shot conditional baselines and single-expert search on motif scaffolding, while strictly preserving the motif and improving scaffold quality across complementary metrics.

6 Conclusion

We presented MCTD-ME, a model-agnostic planner that reframes reverse diffusion as Monte Carlo tree search, enabling parallel multi-token revisions and principled exploration guided by multiple proposal experts. A pLDDT-driven masking schedule focuses denoising on structurally uncertain regions, improving biophysical fidelity, while our P$\mathcal{H}$-UCT-ME selection reuses cached uncertainty to efficiently arbitrate among experts without recomputing entropies. Across folding, inverse folding, and motif scaffolding, MCTD-ME consistently outperforms single-expert and one-shot baselines—raising AAR and scTM and lowering RMSD—with gains that scale with test-time compute and are robust to the choice of experts. Although the method currently relies on surrogate structure predictors and incurs some search overhead as a limitation, it provides a unified, plug-and-play test-time mechanism that reliably improves design quality. Future work will integrate learned critics and apply further systems optimizations to enhance both accuracy and speed.

References

[1] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Maximillian Sintorn, Mohammad Norouzi, Nicholas Roberts, Jascha Sohl-Dickstein, and Rif A. Saurous. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17852–17866, 2021.

[2] Doojin Baek, Sungjin Ahn, Hyeonseo Cho, Jaesik Yoon, and Yoshua Bengio. Monte carlo tree diffusion for system 2 planning. *arXiv:2502.07202*, 2025. URL https://arxiv.org/abs/2502.07202.

[3] Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. MolGPT: Molecular generation using a transformer-decoder model. *Journal of Chemical Information and Modeling*, 62(9):2064–2076, 2021.

[4] Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Chenghao Liu, Andrei Nica, Maksym Korablyov, Michael M. Bronstein, and Alexander Tong. SE(3)-stochastic flow matching for protein backbone generation. In *International Conference on Learning Representations (ICLR)*, 2024.

[5] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

[6] Justin Dauparas, Ivan Anishchenko, Nathan Bennett, Huafeng Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, Alex Courbet, R. J. de Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, Sam Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and David Baker. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022. doi: 10.1126/science.add2187.

[7] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Remix-dit: Mixing diffusion transformers for multi-expert denoising. *arXiv preprint arXiv:2412.05628*, 2024.

[8] Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025.

[9] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv*, 1112.5745, 2011.

[10] Qiaoyu Hu, Changzhi Sun, Huan He, Jiazheng Xu, Danlin Liu, Wenqing Zhang, Sumeng Shi, Kai Zhang, and Honglin Li. Target-aware 3d molecular generation based on guided equivariant diffusion. *Nature Communications*, 16:7928, 2025. doi: 10.1038/s41467-025-63245-0.

[11] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[12] Bowen Jing, Ezra Erives, Peter Pao-Huang, Gabriele Corso, Bonnie Berger, and Tommi Jaakkola. Eigenfold: Generative protein structure prediction with diffusion models. *arXiv preprint arXiv:2304.02198*, 2023. ML for Drug Discovery Workshop, ICLR.

[13] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pages 282–293. Springer, 2006.

[14] Yunsung Lee, Jin-Young Kim, Hyojun Go, Myeongho Jeong, Shinhyeok Oh, and Seungtaek Choi. Multi-architecture multi-expert diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13427–13436, 2024.

[15] Yanjun Li, Jianing Wang, Yuxi Li, Zhiqiang Liu, Pengfei Liu, Peng Gao, and Zhaoxiang Zhang. Rethinkmcts: Refining erroneous thoughts in monte carlo tree search for code generation. *arXiv preprint arXiv:2409.09584*, 2024.

[16] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

[17] Yuntao Li, Yuhang Li, Zhenkun Shi, Mingdong Li, Yuxing Wang, and Jianbo Li. Discovering mathematical formulas from data via gpt-guided monte carlo tree search. *arXiv preprint arXiv:2401.14424*, 2024. URL https://arxiv.org/abs/2401.14424.

[18] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, William Lu, Nikita Smetanin, Robert Verkuil, Olga Kabeli, Abel Dos Santos Costa, and et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 2024. doi: 10.1126/science.adq7360.

[19] Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. Don't throw away your value model! Generating more preferable text with Value-Guided Monte-Carlo Tree Search decoding. *arXiv preprint arXiv:2309.15028*, 2023.

[20] Mengdi Liu, Xiaoxue Cheng, Zhangyang Gao, Hong Chang, Cheng Tan, Shiguang Shan, and Xilin Chen. Protinvtree: Deliberate protein inverse folding with reward-guided tree search, 2025. URL https://arxiv.org/abs/2506.00925.

[21] Xuefeng Liu, Takuma Yoneda, Rick L Stevens, Matthew R Walter, and Yuxin Chen. Blending imitation and reinforcement learning for robust policy improvement. *arXiv preprint arXiv:2310.01737*, 2023.

[22] Xuefeng Liu, Takuma Yoneda, Chaoqi Wang, Matthew R Walter, and Yuxin Chen. Active policy improvement from multiple black-box oracles. In *International Conference on Machine Learning*, pages 22091–22116. PMLR, 2023.

[23] Xuefeng Liu, Chih-chan Tien, Peng Ding, Songhao Jiang, and Rick L. Stevens. Entropy-reinforced planning with large language models for drug discovery. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[24] Xuefeng Liu, Fangfang Xia, Rick Stevens, and Yuxin Chen. Contextual active model selection. *Advances in Neural Information Processing Systems*, 37:49914–49956, 2024.

[25] Xuefeng Liu, Songhao Jiang, Siyu Chen, Zhuoran Yang, Yuxin Chen, Ian Foster, and Rick Stevens. Drugimprovergpt: A large language model for drug optimization with fine-tuning via structured policy optimization. *arXiv preprint arXiv:2502.07237*, 2025.

[26] Xuefeng Liu, Songhao Jiang, Ian Foster, Jinbo Xu, and Rick Stevens. Scaffoldgpt: A scaffold-based gpt model for drug optimization. *arXiv preprint arXiv:2502.06891*, 2025.

[27] Xuefeng Liu, Songhao Jiang, Bo Li, and Rick Stevens. Controllablegpt: A ground-up designed controllable gpt for molecule optimization. *arXiv preprint arXiv:2502.10631*, 2025.

[28] Joshua Meyers, Benedek Fabian, and Nathan Brown. De novo molecular design and generative models. *Drug Discovery Today*, 26(11):2707–2715, 2021. ISSN 1359-6446. doi: https://doi.org/10.1016/j.drudis.2021.05.019. URL https://www.sciencedirect.com/science/article/pii/S1359644621002531.

[29] Martin L Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[30] Samyak Sahoo, Ruisi Li, Long Wang, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*, 2024.

[31] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024.

[32] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[33] Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. Peptune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. *arXiv preprint arXiv:2412.17780*, 2024.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[35] Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners, 2024. URL `https://arxiv.org/abs/2402.18567`.

[36] Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Dplm-2: A multimodal diffusion protein language model, 2024. URL `https://arxiv.org/abs/2410.13782`.

[37] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, Basile I M Wicky, Nao Hanikel, Samuel J Pellock, Alexis Courbet, William Sheffler, Jipeng Wang, Preetha Venkatesh, Ian Sappington, Sara V Torres, Alisa Lauko, Valentin De Bortoli, Emile Mathieu, Sergey Ovchinnikov, Regina Barzilay, Tommi S Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. De novo design of protein structure and function with RFdiffusion. *Nature*, 620 (7976):1089–1100, 2023.

[38] Joseph L. Watson, David Juergens, Nathaniel R. Bennett, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620:1089–1100, 2023. doi: 10.1038/s41586-023-06415-8.

[39] Shunyu Yao, Dianjie Zhan, Harry Shafran, Joseph Allaway, Ahmet Gulcehre, Kyunghyun Chung, Rewon Anil, Jonathan Chi, Chen Li, and Jason Wei. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

[40] Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297*, 2023.

[41] Jason Yim, Andrew Campbell, Emile Mathieu, Andrew Y.K. Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S. Veeling, Frank Noé, et al. Improved motif-scaffolding with se(3) flow matching. *arXiv preprint arXiv:2401.04082*, 2024.

[42] Shun Zhang, Zhenfang Chen, Yikang Shen, Mingyu Ding, Joshua B Tenenbaum, and Chuang Gan. Planning with large language models for code generation. *arXiv preprint arXiv:2303.05510*, 2023.

A Appendix

A.1 Algorithm details.

Lead optimiaztion In this algorithm, we use MCTD-ME to tackle the lead optimization challenge. Beginning with an initial lead—a fully denoised baseline sequence—we execute MCTS with the following procedure: (i) selection uses a P$\mathcal{H}$-UCT-ME–style rule that combines value (Q), exploration, and cached uncertainty bonuses (ensemble surprisal ($\mathcal{U}_{\text{ent}}$) and novelty ($\mathcal{U}_{\text{div}}$)); (ii) expansion proposes edits only on pLDDT-masked uncertain sites and performs multi-expert rollouts (E generators), producing candidates scored once via a composite critic (cached to avoid re-evaluation); (iii) we keep the Top-K children by composite reward, attach their uncertainty bonuses, and (iv) backpropagate the candidate's value to update ancestors (max/sum backup). This yields a planner that routes among experts, focuses edits where structure is uncertain, reuses evaluations via caching, and optimizes a task-aware composite objective.

Algorithm 1 Monte Carlo Tree Diffusion with Multiple Experts (MCTD-ME)

```
Require:
    $root \leftarrow$ fully denoised baseline node $y_0$ (initial lead)
    $c_p$: exploration constant                                   // for P$\mathcal{H}$-UCT-ME
    $K$: number of children per expansion
    $E$: number of expert models
    $R$: rollouts per expert
    $T$: total simulations
    $\mathcal{E} = \{\pi^1, \dots, \pi^E\}$: generator experts
    $\mathcal{C}$: set of critic functions
    $cache \leftarrow$ empty dictionary for evaluated sequences
 1: for $i = 1$ to $T$ do
 2:     $node \leftarrow root$
        ▷ /* Selection via P$\mathcal{H}$-UCT-ME*/
 3:     while $node.children \neq \emptyset$ do
 4:         $node \leftarrow$ P$\mathcal{H}$-UCT-ME$(node.children)$   // uses $Q$, UCB, and cached $(\mathcal{U}_{\text{ent}}, \mathcal{U}_{\text{div}})$
        ▷ /* Expansion via pLDDT Masking and Multi-Expert Rollouts */
 5:     $\mathcal{M} \leftarrow$ GetMaskSet$(node.sequence)$          // Based on pLDDT
 6:     $\mathcal{Y} \leftarrow \emptyset$                             // candidate children
 7:     for $e \in \mathcal{E}$ do                                     // each generator expert
 8:         for $r = 1$ to $R$ do                                      // rollouts
 9:             $y_0 \leftarrow$ MaskedDiffusion$(node.sequence, \mathcal{M}, e)$
10:             if $y_0 \notin cache$ then
11:                 $score \leftarrow$ EvalComposite$(y_0; \mathcal{C})$   // weighted sum of critics
12:                 $cache[y_0] \leftarrow score$
13:             $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{y_0\}$
14:     $\mathcal{Y}_{\text{top}} \leftarrow$ TopK$(\mathcal{Y}, K;\ cache)$     // sort by composite reward
15:     for $y' \in \mathcal{Y}_{\text{top}}$ do
16:         $(\mathcal{U}_{\text{ent}}(y'), \mathcal{U}_{\text{div}}(y')) \leftarrow$ ComputeBonuses$(y',\ node.sequence,\ \mathcal{M})$   // ensemble
    surprisal & novelty
17:         AddChild$(node, y', \mathcal{U}_{\text{ent}}(y'), \mathcal{U}_{\text{div}}(y'))$   // create new node with full seq
        ▷ /* Evaluation of New Children (full-sequence) */
18:     for $y' \in \mathcal{Y}_{\text{top}}$ do
19:         $v \leftarrow cache[y']$                  // may hit cache; same composite reward as above
            ▷ /* Backpropagation */
20:         Backpropagate$(y',\ v)$                    // supports max or sum backup
21: return Top-$k$ sequences by $cache[\cdot]$
```

De novo generation We start from an all-mask root. At each iteration, *Selection* (P$\mathcal{H}$-UCT-ME) walks to a leaf. *Expansion* proposes partial fills only at low-pLDDT sites (high-pLDDT positions are frozen). Then *Evaluation* performs short rollouts from each new child (e.g., a few masked-diffusion steps / fast fold+critic passes) to estimate a terminal composite value; results are cached. Finally, we *Backpropagate* those values. This keeps the planner focused on uncertain regions while using cheap rollouts to decide which partials to keep; see Algorithm 2 for pseudocode.

Algorithm 2 De Novo MCTD with Multiple Experts (MCTD-ME -DeNovo)

```
Require:
    $root \leftarrow$ all-mask sequence of length $L$                          // no residues fixed
    $c_p$: exploration constant                                              // for P$\mathcal{H}$-UCT-ME
    $K$: children kept per expansion
    $E$: number of generator experts
    $S$: simulation depth (rollout steps)
    $T$: total simulations
    $\mathcal{E} = \{\pi^1, \dots, \pi^E\}$: generator experts
    $\mathcal{C}$: critic set (structure/biophys surrogates)
    $cache \leftarrow$ empty dictionary for (partial or full) sequence scores
 1: for $i = 1$ to $T$ do
 2:     $node \leftarrow root$
        ▷ /* 1) Selection via P$\mathcal{H}$-UCT-ME*/
 3:     while $node.children \neq \emptyset$ do
 4:         $node \leftarrow$ P$\mathcal{H}$-UCT-ME$(node.children)$        // uses $Q$, UCB, cached $(\mathcal{U}_{\text{ent}}, \mathcal{U}_{\text{div}})$
        ▷ /* 2) Expansion: edit only low-pLDDT sites; freeze high-pLDDT */
 5:     $\hat{p} \leftarrow$ PREDICTPLDDT$(node)$
 6:     $\mathcal{M} \leftarrow$ EDITABLEMASK$(\hat{p})$                       // low-confidence = editable
 7:     $\mathcal{Y} \leftarrow \emptyset$                                       // new children (partials)
 8:     for $e \in \mathcal{E}$ do                                              // each generator expert
 9:         $y' \leftarrow$ MASKEDDIFFUSION$(node.sequence, \mathcal{M}, e)$
10:         $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{y'\}$
        ▷ /* 3) Evaluation (Rollouts): estimate child values */
11:     for $y' \in \mathcal{Y}$ do
12:         if $y' \notin cache$ then
13:             $v \leftarrow$ ROLLOUTSIM$(y', S, \mathcal{E}, \mathcal{C})$
14:                 // e.g., $S$ short masked-diffusion steps with fast fold + EVALCOMPOSITE
15:             $cache[y'] \leftarrow v$
16:         else
17:             $v \leftarrow cache[y']$
18:         $(\mathcal{U}_{\text{ent}}(y'), \mathcal{U}_{\text{div}}(y')) \leftarrow$ COMPUTEBONUSES$(y', node.sequence, \mathcal{M})$
19:         ADDCHILD$(node, y', \mathcal{U}_{\text{ent}}(y'), \mathcal{U}_{\text{div}}(y'), Q{=}v)$
20:     $children\_topK \leftarrow$ TOPK$(node.children, K; cache)$            // prune by rollout value
21:     PRUNETO$(node, children\_topK)$
        ▷ /* 4) Backpropagation */
22:     for $y' \in children\_topK$ do
23:         BACKPROPAGATE$(y', cache[y'])$                                       // max or sum backup
24: return Top-$k$ (near-)complete sequences by $cache[\cdot]$
```

A.2 Dataset details.

We evaluate MCTD-ME on two protein sequence benchmarks at the *chain* level. As illustrated in Table 4 (1) **CAMEO2022** comprises 183 targets released in 2022; we use the reference backbones and evaluate inverse folding on the corresponding chains. (2) **PDB_date** is a date-split benchmark of 7,855 proteins constructed from the Protein Data Bank (PDB); we hold out targets by deposition date and evaluate inverse folding on their chains. In our experiments we use a *date-split subset of 449 chains* from PDB_date for compute efficiency; unless otherwise noted, length statistics below are for the full benchmark. For both sets, sequence lengths are computed after simple canonicalization (upper-casing and removal of non-canonical/unknown residues).

Dataset	# Proteins	Min	Mean	Max
CAMEO2022	183	15	247.5	704
PDB_date (full)	7,855	2	242.2	511

Table 4: **Length statistics (in residues)** of amino-acid sequences in each benchmark. Counts refer to the number of evaluated chains. *Note:* Experiments use a date-split *subset* of 449 PDB_date chains; its length distribution is similar to the full set.

A.3 Reward definitions and normalization.

Terminology (metrics). TM-score: standard topology-similarity score between two backbones (range $[0, 1]$; higher is better).

scTM: "sequence-conditioned TM-score" — TM-score between the *target* backbone and the backbone *predicted from the designed sequence* (higher is better).

RMSD: global Cα root-mean-square deviation in Ångströms (lower is better).

Motif RMSD: Cα RMSD computed *only* on the predefined motif/epitope after aligning on the motif (lower is better).

pLDDT: per-residue confidence from a structure predictor (e.g., AlphaFold-type models), originally reported on a $[0, 100]$ scale; we divide by 100 to normalize to $[0, 1]$.

AAR: amino-acid recovery — fraction of positions where the designed sequence exactly matches the native (range $[0, 1]$).

Composite reward: weighted combination of normalized terms (e.g., TM-score, transformed RMSD, pLDDT, scTM, biophysical checks), constructed to lie in $[0, 1]$.

All reward components are normalized to the $[0, 1]$ range; pLDDT is averaged over residues and divided by 100 We use the composite reward score as follows:

Folding (sequence→structure):

$$R_{\text{fold}} = \alpha\,\text{TM} + \beta\,\big(1 - \min(\text{RMSD}/10,\ 1)\big) + \gamma\,\overline{\text{pLDDT}},$$

with defaults $(\alpha, \beta, \gamma) = (0.60, 0.40, 0)$. If pLDDT is available, set $\gamma = 0.05$ and renormalize (α, β) so $\alpha{+}\beta{+}\gamma{=}1$. RMSD is global C$\alpha$ RMSD in Å.

Inverse folding (structure→sequence):

$$R_{\text{inv}} = 0.60\,\text{AAR} + 0.35\,\text{scTM} + 0.05\,B,$$

where AAR is residue-wise recovery versus the native sequence, scTM is the TM-score between the target backbone and the fold predicted from the designed sequence, and B is a small biophysical bonus (e.g., motif/chemistry checks).

Motif scaffolding (motif→scaffold): Candidates that do *not* exactly preserve the motif receive $R_{\text{motif}}{=}0$. Otherwise,

$$R_{\text{motif}} = 0.40\,\overline{\text{pLDDT}}_{\text{scaf}}{+}0.30\,g(\text{RMSD}_{\text{motif}}){+}0.30\,\text{scTM}{+}0.20\,\mathbf{1}\big[\text{RMSD}_{\text{motif}}{<}1{\wedge}\text{scTM}{>}0.8\big],$$

where $\overline{\text{pLDDT}}_{\text{scaf}}$ is mean pLDDT on non-motif residues, and the motif-aligned Cα RMSD is mapped via

$$g(x) = \begin{cases} \max(0,\ 1 - x/2), & x < 1, \\ \max(0,\ 0.2 - x/10), & x \geq 1. \end{cases}$$

We use a broad inverted normalization for *global* folding RMSD and a sharper piecewise map (plus a success bonus) for *local* motif RMSD to prioritize precise interface preservation.

A.4 Length-binned improvements.

Here we stratify inverse-folding gains by protein length to assess scalability. Figure 3 reports mean improvements (Final–Baseline) for AAR, normalized reward, and scTM across five bins: <100, 100–200, 200–300, 300–400, and >400 residues. The multi-expert planner outperforms the single-expert baseline in every bin, with especially pronounced margins on long proteins: scTM gains remain positive and grow with length (single-expert even turns slightly negative in the 300–400 bin), reward boosts are largest for >400, and AAR gains—while modest—are consistently higher for the ensemble (peak around 200–300). These trends indicate that expert diversity is most beneficial as sequence–structure complexity increases. AAR is a strict token-identity metric and thus less sensitive to small but structurally meaningful edits, whereas scTM/reward are smoother structural objectives that respond strongly to localized corrections. In addition, our critic emphasizes structural fidelity and the pLDDT-guided masking focuses changes on low-confidence regions, encouraging fold-improving (not necessarily native-identical) substitutions—producing larger scTM/reward deltas than AAR, especially for longer, more complex proteins.

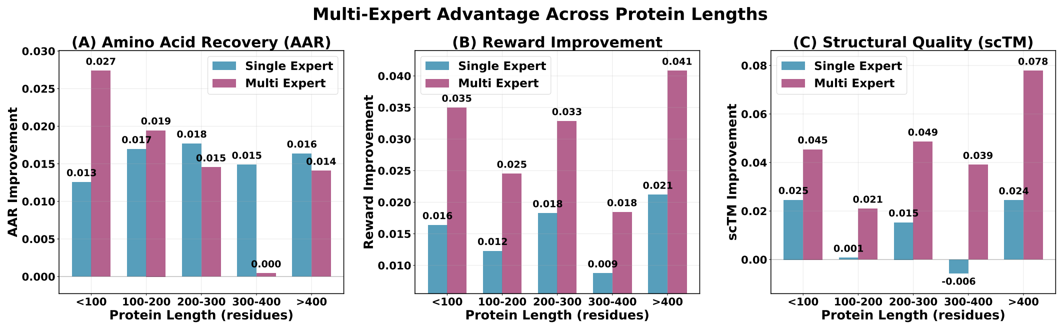

Figure 3: Length-binned improvements for inverse folding: mean ΔAAR, Δ normalized reward, and ΔscTM (Final$-$Baseline) for *single-expert* vs. *multi-expert* planners. Larger values indicate better recovery and structural consistency; the multi-expert planner yields consistent gains across bins, with larger margins on long proteins.

A.5 Lead-optimization vs. De novo generation.

While our current implementation treats root node as a complete sequence—effectively a lead-optimization style search—the framework naturally generalizes to fully de novo generation. In this setting, the root node could be a fully masked sequence, with subsequent nodes representing partially unmasked intermediates obtained through pLDDT-guided masking. Evaluation would then involve a simulation stage for reward estimation at each partial completion. Conceptually, this is equivalent to our guided unmasking scheme, differing only in granularity: entropy and disagreement are currently assessed at the level of entire candidate sequences, but under de novo generation they would be localized to the partially unmasked subsequence. As a result, performance is expected to be qualitatively similar, with the search dynamics adapting seamlessly from complete-sequence optimization to progressive de novo exploration.

A.6 Related works.

Protein and Molecular Diffusion Models. Recent advances apply diffusion models to protein design across both structure and sequence domains. RFdiffusion [37] introduced a guided 3D diffusion model for protein backbones. On the sequence side, DPLM [35] and its extension DPLM-2 [36] demonstrated that discrete diffusion pretraining on large-scale protein data enables high-quality sequence generation and versatile conditioning (e.g., inverse folding, property steering). Most recently, DiffGui [10] proposed a target-aware, E(3)-equivariant diffusion framework for joint atom and bond generation, incorporating multi-property guidance (e.g., binding affinity, drug-likeness) to generate realistic, high-affinity 3D molecules. However, these diffusion approaches generally rely on one-shot or open-loop decoding, making it difficult to incorporate adaptive search or revision during generation—highlighting the need for planning methods such as MCTS.

Monte Carlo Tree Search in Generative Modeling. Monte Carlo Tree Search (MCTS) has improved generative decoding in tasks requiring complex reasoning or program synthesis. In code generation, AlphaCode [16] used a brute-force generate-and-test method by sampling large program sets and selecting those that passed tests. More structured methods followed: RethinkMCTS [15] performs MCTS over LLM reasoning steps, using execution feedback to refine intermediate thoughts. Tree-of-Thoughts [39] frames reasoning as a tree of self-evaluated partial solutions expanded via classical search (DFS/BFS), boosting performance on puzzles and planning. PG-TD [42] simulates lookahead in code generation by executing candidate programs during decoding. ERP [23] achieves state-of-the-art performance by leveraging MCTS with a GPT model, with applications in both drug discovery and code generation. Across domains, MCTS-style planning consistently enhances generation quality and reliability. Yet, most of these approaches are tied to autoregressive GPT-style models, which remain token-by-token and struggle with long-range dependencies—suggesting that discrete diffusion could provide a more flexible backbone for planning.

Discrete Diffusion for Planning and Generation. While originally developed for continuous domains, diffusion models have been successfully adapted to discrete sequence generation and planning. D3PM [1] introduced absorbing-state (masking) noise for discrete diffusion, achieving competitive text generation. Follow-ups diffusion language models like MD4 [31] further closed the performance gap with autoregressive models. In planning, Diffuser [11] modeled offline RL as trajectory denoising, generating full state-action paths guided by value functions. To enable adaptive inference, Monte Carlo Tree Diffusion (MCTD) [2] reimagines diffusion as tree search, branching multiple denoising outcomes per step and scoring partial trajectories via heuristics. In protein design, ProtInvTree [20] similarly integrates reward-guided search with jumpy denoising for efficient inverse folding. More recently, PepTune [33] proposes multi-objective discrete diffusion for therapeutic peptide SMILES, pairing masked diffusion with an MCTS-style guidance strategy to negotiate tradeoffs among binding, solubility, and membrane permeability objectives. Still, existing methods optimize toward a single diffusion generator, which limits their ability to effective search space exploration in biological and chemical design.

Multi-Experts Guided Generation. Multi-expert systems leverage several generative models, or 'experts,' in tandem, allowing each to contribute its strengths while combining their state-wise expertise. This idea has been explored in reinforcement learning [21, 22] and bandit [24] settings, where algorithms learn from multiple expert policies or oracles to improve decision-making. In generative modeling, mixture-of-experts strategies similarly combine specialized models to better cover complex data distributions. For example, recent diffusion-based image generators use multiple specialized denoising models across different noise levels [7] or even heterogeneous model architectures [14] to boost output quality. However, these multi-expert approaches remain largely static and one-shot during inference, without an adaptive search process. To our knowledge, no prior diffusion model integrates multiple experts into a planning loop for generative design. By incorporating a team of expert generators within an MCTS-driven diffusion framework, our work enables dynamic collaboration among experts to satisfy multiple design criteria and explore a broader solution space—an innovative combination that extends diffusion-based generation to more complex, multi-objective scenarios.

A.7 Case study on CAMEO data and motif scaffolding.

Figure 4 shows how MCTD-ME refines a CAMEO target (7dz2_C). The light-gray ribbon is the *baseline lead* (one-shot structure). Over it, the MCTD-ME design is drawn only where an improvement over the lead is made and is colored by per-residue closeness to the native backbone after alignment: *red* = *very close*, *yellow* = *closer but not as close*. (Regions without improvement are left in gray and reflect the baseline.) We observe broad reddening across helices and connecting loops, indicating tighter agreement with the target while preserving segments the lead already modeled well. This aligns with our mechanism: pLDDT-aware masking concentrates edits on low-confidence sites, and PH-UCT-ME promotes diverse, high-value proposals from multiple experts, yielding higher fold consistency without over-editing stable regions.

Figure 5 visualizes how MCTD-ME refines the *scaffold* around a fixed functional motif. The motif is shown in blue and is hard-frozen throughout generation. The surrounding gray ribbon is the baseline scaffold (lead), while the colored overlay shows MCTD-ME edits *only where improvement over the*

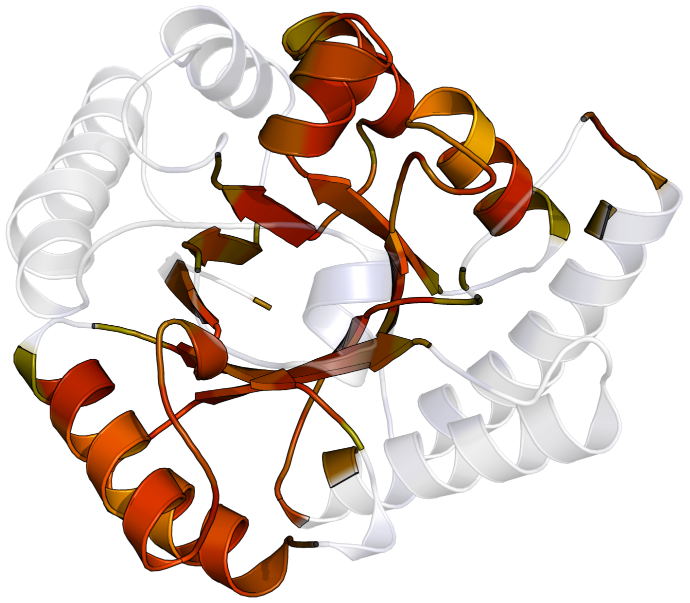

Figure 4: **CAMEO 7dz2_C refinement.** Light gray: baseline lead. Colored overlay: MCTD-ME edits where improvement is achieved, shaded by closeness to the native backbone (red = very close; yellow = closer but not as close). Gray segments indicate no change from the lead. The predominance of red/yellow across many regions visualizes how MCTD-ME moves the lead toward the native geometry via pLDDT-aware masking and multi-expert selection.

lead is achieved, shaded by closeness to the native structure after motif-aligned superposition (*red = very close*, *yellow = closer but not as close*). We observe concentrated reddening along the β-strand and adjacent loop, indicating better register and loop geometry at the motif–scaffold interface, while unchanged regions remain gray—highlighting that MCTD-ME preserves already adequate geometry and focuses edits where they matter.

A.8 Setup and hyperparameters.

Component	**Symbol**	**Setting(s) used**
Rollouts per expert	k_{roll}	**3** (fixed for reported runs)
Children kept per expansion	K	**3** (beam size per expansion)
Forward steps (tree depth)	max_depth	**5** (default; held fixed unless noted)
Exploration constant (P$\mathcal{H}$-UCT-ME)	c_p	**1.414** (UCB1 constant)
Candidates per expansion (random mode)	N_{cand}	**6** (used only in random_no_expert)
Backup rule	—	**max** (value backup along the path)
Number of experts (multi_expert)	E	**3** (unless otherwise noted)
Modes compared	—	random_no_expert, single_expert, multi_expert
Single-expert ID in plots	—	**single_expert_0** (for direct comparisons)
Diffusion reverse steps	max_iter	**150**
Diffusion temperature	T	**1.0**
Decoding strategy (all modes)	—	deterministic unmasking, argmax sampling

Table 5: **Search setup and hyperparameters.** We keep decoding and evaluation identical across methods. Hyperparameters are fixed for the main comparisons; only the presence/number of experts differs by mode.

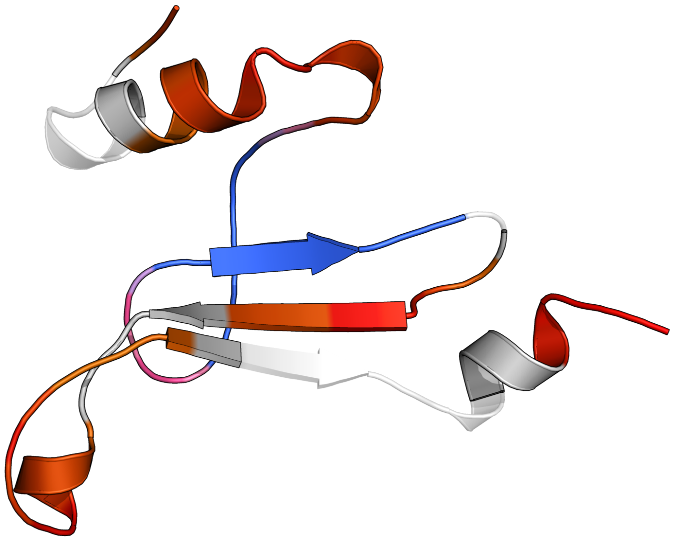

Figure 5: **Motif scaffolding (PDB 7mrx).** Blue segment: fixed input motif; gray ribbon: baseline scaffold (lead). Colored overlay: MCTD-ME 's improved scaffold residues (red = very close to native; yellow = closer), shown only where the scaffold improves relative to the lead after motif-aligned superposition. The localized reddening near the motif indicates targeted refinement of the interface without over-editing stable regions.

For equitable assessment, we compare all variants under matched search and decoding budgets. Unless otherwise specified, every method uses the same DPLM-2 decoding configuration (deterministic unmasking with argmax sampling) and identical reward evaluation. In each table/figure, competing methods share the same rollout budget and selection rule; only the availability of diffusion *experts* differs across modes (random_no_expert, single_expert, multi_expert). We conducted only light tuning around a small grid and then fixed a single setting for all reported comparisons.

A.9 Runtime and efficiency.

The enhanced performance comes at the cost of additional computation. MCTD-ME must evaluate multiple experts across many tree expansions, whereas a single diffusion model decoding is much faster. In our current implementation, generating one sequence for a CAMEO structure with MCTD-ME-E takes on the order of a few minutes (wall-clock) on a single GPU – roughly an order of magnitude slower than a single DPLM-2 pass. We emphasize that our focus here is on proof-of-concept effectiveness; optimization of the search procedure is left for future work. Techniques such as parallelizing rollouts, caching expert evaluations, or pruning low-reward branches more aggressively could greatly speed up MCTD-ME without sacrificing quality.

Experiments were run on a SLURM-managed cluster with one NVIDIA A100 GPU per job, four CPU cores and 32 GB RAM. All results were obtained with PyTorch mixed precision and identical inference code across modes; wall-time numbers in Table 6 are measured end-to-end per target (including expert rollouts and critic scoring).

A.10 Generality.

Although our evaluation focused on the inverse folding problem, the MCTD-ME framework is inherently general and model-agnostic. Any diffusion-based generator can be plugged into the tree

Mode	**Min (s)**	**Mean (s)**	**Max (s)**
Random (no expert)	89.6	645.3	5235.6
Single-expert (650M)	154.9	345.7	1024.8
Single-expert (150M)	257.6	502.9	1791.6
Single-expert (3B)	148.6	451.7	1052.0
Multi-expert (3$\times$)	314.0	800.2	2187.2

Table 6: **Per-target wall-clock time** (seconds) aggregated over CAMEO2022 and PDB_date runs. "Single-expert (650M/150M/3B)" correspond to our three DPLM-2 capacities (IDs 0/1/2). Times include all steps (masking, diffusion rollouts, scoring, and tree updates).

search, and any collection of expert predictors can shape the reward. This flexibility opens up many possibilities: for instance, MCTD-ME could be applied to de novo protein design by incorporating structural-validity scores and evolutionary metrics as critics, or extended to other domains such as molecule generation using pharmacophoric and ADMET property predictors. Our results on protein inverse folding demonstrate the promise of combining diffusion models with MCTS guided by multiple critics for multi-objective generative design. We believe this diffusion+MCTS approach offers a powerful new paradigm for generative modeling, and extending MCTD-ME to more tasks with richer reward functions is an exciting direction for future work.