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. 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 predictive-entropy UCT to expert ensembles. On the inverse folding task (CAMEO and PDB benchmarks), MCTD-ME outperforms single-expert and unguided baselines in both sequence recovery (AAR) and structural similarity (scTM), with gains increasing for longer proteins and benefiting from multi-expert guidance. More generally, the framework is model-agnostic and applicable beyond inverse folding, including de novo protein engineering and multi-objective molecular generation.

1 Introduction

Large generative models have demonstrated impressive capabilities across domains such as natural language processing [29] and molecular design [25]. 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 [21]. 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 [11], MCTS has shown success in domains requiring sequential decision-making, most notably in AlphaGo [28]. 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 [33] and RethinkMCTS [13] illustrate this potential in language and code generation, where search paths can be evaluated and adjusted.

Large language models (LLMs such as GPT [18, 19, 22–24]) 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 [15], while value-guided MCTS decoding (PPO-MCTS [16]) 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 [9] 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 2.

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 exploration. Central to our design are (i) a pLDDT-guided masking policy that focuses edits on structurally uncertain regions to improve biophysical fidelity, and (ii) a multi-expert selection mechanism that arbitrates among experts using cached uncertainty statistics, yielding stable and efficient search.

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 and guided by a multi-expert rollout mechanism, where each expert proposes candidate com-

pletions 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, an entropy-aware selection rule, extending the classic UCT formula with exploration bonuses based on expert disagreement (e.g., score variance or entropy), inspired by entropy-guided UCB from ERP [21].

Third, we demonstrate empirical gains on long-horizon generation tasks—specifically inverse protein folding for just one case example. Using the CAMEO 2022 benchmark [10] and a PDB date-split benchmark [4], we show that our framework consistently improves sequence recovery (AAR) and structural similarity (scTM) over strong DPLM-2 baselines [31]. Our approach yields higher-quality, more diverse solutions, and these improvements grow with larger search budgets—highlighting the scalability and deliberative nature of our framework. We observe that both pLDDT-guided masking and multi-expert selection contribute complementary gains, with larger benefits on longer proteins where uncertainty is spatially concentrated. 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

We review related work in protein diffusion models, MCTS in generative modeling and molecule design, discrete diffusion for planning and generation, and multi-objective and expert-guided Generation.

Protein and Drug Diffusion Models Recent advances apply diffusion models to protein design across both structure and sequence domains. RFdiffusion [32] introduced a guided 3D diffusion model for protein backbones, achieving state-of-the-art results. On the sequence side, DPLM [30] and its extension DPLM-2 [31] 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 [8] 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 [14] used a brute-force generate-and-test method by sampling large program sets and selecting those that passed tests. More structured methods followed: RethinkMCTS [13] performs MCTS over LLM reasoning steps, using execution feedback to refine intermediate thoughts. Tree-of-Thoughts [33] frames reasoning as a tree of self-evaluated partial solutions expanded via classical search (DFS/BFS), boosting performance on puzzles and planning. PG-TD [34] simulates lookahead in code generation by executing candidate programs during decoding. 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 like [27] further closed the performance gap with autoregressive models. In planning, Diffuser [9] 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 [17] similarly integrates reward-guided search with jumpy denoising for efficient inverse folding. 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 from its area of strength. This idea has been explored in reinforcement learning and bandit settings, where algorithms learn from multiple expert policies or oracles to improve decision-making [20]. 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 [5] or even heterogeneous model architectures [12] 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.

3 Preliminaries

We describe Diffusion-based generative model, MCTS, and protein design along with their mathematical notation in the subsequent sections and unify them under the Markov decision processes framework.

Markov decision processes. We formulate generative planning as a finite-horizon MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, H, P, R \rangle$ [26], 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 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. This formulation generalizes to multiple critics: given K evaluators $C_0, \ldots, C_{K-1}$, each scoring complete sequences, guidance becomes even more natural. Instead of aggregating into a scalar reward, we perform imitation-style rollouts: partial candidates are completed, scored individually by each expert, and used to inform node evaluations during tree search. This preserves the diversity of expert preferences and supports structured planning, which we realize through our Monte Carlo Tree Diffusion (MCTD) framework.

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, simulation, and backpropagation. Starting from the root (current state), the selection phase recursively chooses child nodes using an Upper Confidence Bound (UCT) criterion [11]:

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

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,

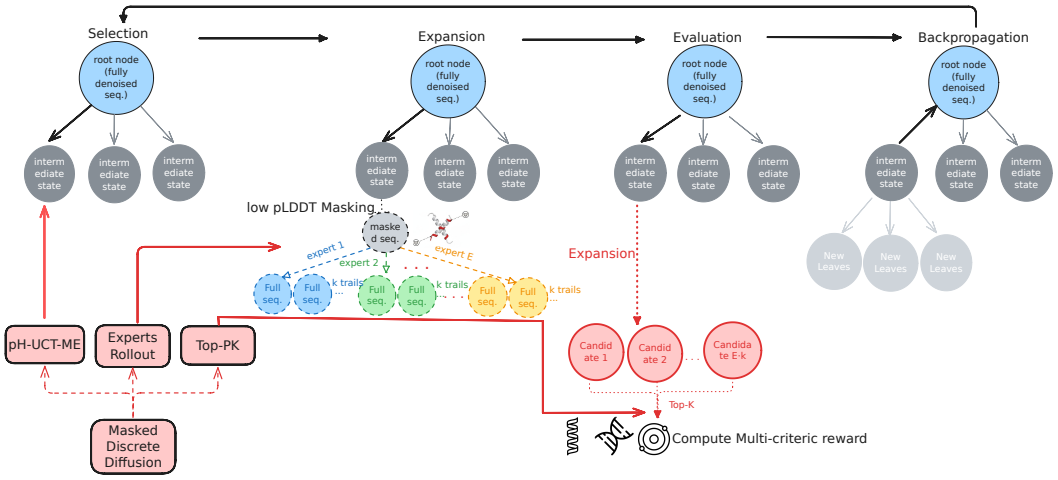

Figure 1: Overview of MCTD in diffusion-based protein sequence generation. Nodes are partially denoised sequences expanded via masked diffusion and guided by multi-expert rollouts.

simulation evaluates these new nodes 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. 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. 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, with each rollout consisting of four key steps.

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—an entropy-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:

$$\text{P}\mathcal{H}\text{-UCT-ME}(s_t) \;=\; \arg\max_{a\in\mathcal{A}} \; \text{P}\mathcal{H}\text{-UCB-ME}((s_t,a))\,, \tag{2}$$

with

$$\text{P}\mathcal{H}\text{-UCB-ME}((s_t,a)) = Q(s_t,a) + c_p\,\frac{\sqrt{\log N(s_t)}}{1+N(s_t,a)}\cdot\left(\underbrace{w_{\text{ent}}\,\mathcal{U}_{\text{ent}}(s_t,a) \;+\; w_{\text{div}}\,\mathcal{U}_{\text{div}}(s_t,a)}_{\text{P}\mathcal{H}\text{-ME bonus (cached at expansion)}}\right). \tag{3}$$

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 PH 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

$$\mathcal{U}_{\text{ent}}(s_t, a) = \mathcal{H}\left(\frac{1}{E}\sum_{e=1}^{E} p_\phi^{(e)}(\cdot \mid s_{t-1}, \mathcal{M})\right) - \frac{1}{E}\sum_{e=1}^{E} \mathcal{H}\Big(p_\phi^{(e)}(\cdot \mid s_{t-1}, \mathcal{M})\Big). \tag{4}$$

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 [7]), or Jensen–Shannon–style divergence between experts. Intuitively, it isolates epistemic uncertainty: if experts disagree, $\mathcal{H}[\bar{p}]$ is large while the average $\mathcal{H}[p^{(e)}]$ 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:

$$\mathcal{U}_{\text{div}}(s_t, a) = \frac{1}{L}\sum_{i=1}^{L} \mathbf{1}\{y_i^{\text{child}} \neq y_i^{\text{parent}}\}, \tag{5}$$

For our inverse folding case studies, this simply counts the fraction of amino acids that differ. Both $\mathcal{U}_{\text{ent}}$ and $\mathcal{U}_{\text{div}}$ are computed once at expansion and stored 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 $p_\phi(s_{t-1} \mid s_t)$ maintains consistency with the denoising dynamics.

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.

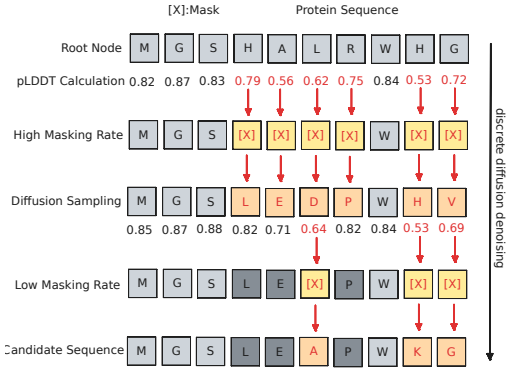

Figure 2: Masked discrete diffusion step: low-confidence sites (mask) are resampled while confident tokens remain fixed, yielding a progressively "unmasked" sequence.

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:

$$y_{t-1} \sim p_\phi(y_{t-1} \mid \text{Mask}(y_t; \mathcal{M}_t)). \tag{6}$$

This masked input encourages structural refinement and prevents overwriting already stable subsequences, reducing unnecessary sampling noise and improving convergence.

Multi-Expert Guided Expansion via Rollouts. To enrich proposal diversity and better imitate the training distribution, we leverage multiple pretrained experts during expansion. Given a masked

sequence Mask$(y_t; \mathcal{M}_t)$, each expert $e \in \{1, \ldots, E\}$ performs k rollouts by sampling its conditional distribution $p_\phi^{(e)}(\cdot \mid y_t, \mathcal{M}_t)$. This yields a pooled candidate set

$$\mathcal{C}_t \;=\; \bigcup_{e=1}^{E} \bigcup_{r=1}^{k} y_{t-1}^{(e,r)}, \tag{7}$$

where $y_{t-1}^{(e,r)}$ denotes a complete sequence obtained by filling the masked sites and splicing them back into y_t. All candidates in $\mathcal{C}_t$ are de-duplicated and scored once with the composite reward (see Evaluation). We then retain the top-K children, each annotated with two exploration bonuses computed at expansion: an *ensemble surprisal* term $\mathcal{U}_{\text{ent}}$ (capturing expert disagreement on the masked sites) and a *novelty* term $\mathcal{U}_{\text{div}}$ (the normalized Hamming distance to the parent). These cached quantities are reused during P$\mathcal{H}$-UCT-ME selection, ensuring that search is guided by both expert-informed proposal diversity and explicit uncertainty signals.

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

$$\{y_{i=1}^{K}\} = \text{Top-K}\left(\{y_{t-1}^{(j)}\}, K\right). \tag{8}$$

Here, j indexes every candidate child pooled across experts and rollouts at step t; Top-K ranks them by the composite score 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 scores. 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. Unlike expansion, where multiple *experts* serve as proposal generators, the evaluation stage relies on a set of *critics* to score complete sequences. Because every node in the tree corresponds to a fully denoised sequence y_0, thus evaluation can be performed immediately without further decoding. Each sequence is scored by a panel of *critics* $\{C_1, \ldots, C_J\}$, producing values

$$C_j(y_0) \;:=\; R_j(y_0), \quad j = 1, \ldots, J, \tag{9}$$

where the C_j represent evaluation functions such as sequence recovery (AAR), structural similarity (scTM), or biophysical quality metrics, J here denotes the number of *critics*. In practice, we aggregate these scores into a single scalar via a fixed weighted sum

$$R(y_0) \;=\; \sum_{j=1}^{J} w_j \, C_j(y_0), \qquad w_j \geq 0, \; \sum_{j=1}^{J} w_j = 1, \tag{10}$$

This design matches the diffusion context, where evaluation is only meaningful at complete denoised sequences.

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:

$$Q(s_i, a_i) \;\leftarrow\; \max\{Q(s_i, a_i),\; r_H\}, \tag{11}$$

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.

5 Experiments

5.1 Protein Inverse Folding

We evaluate Monte Carlo Tree Diffusion with Multiple Experts (MCTD-ME) on the task of protein inverse folding, i.e. structure-conditioned sequence design. The main purpose of inverse folding is to find the amino acid sequence such that its folding matches a given backbone structure. In this

Variant	AAR			Normalized Reward			scTM		
CAMEO Benchmark									
	Baseline	Final	Δ	Baseline	Final	Δ	Baseline	Final	Δ
MCTD-ME-0 (Random)	0.438	0.381	−0.057	0.425	0.429	+0.004	0.362	0.458	+0.096
Single-Expert (150M, Exp. 1)	0.427	0.427	+0.000	0.290	0.290	+0.000	0.371	0.371	+0.000
Single-Expert (650M, Exp. 0)	0.442	0.458	+0.016	0.431	0.447	+0.016	0.372	0.387	+0.014
Single-Expert (3B, Exp. 2)	0.443	0.429	−0.014	0.432	0.422	−0.011	0.373	0.369	−0.005
MCTD-ME (Multi-Expert)	**0.439**	**0.451**	**+0.012**	**0.427**	**0.465**	**+0.038**	**0.365**	**0.449**	**+0.084**
PDB Benchmark									
MCTD-ME-0 (Random)	0.430	0.400	−0.030	0.110	0.100	−0.010	0.360	0.380	+0.020
MCTD-ME-1 (650M)	0.490	0.520	+0.030	0.330	0.360	+0.030	0.390	0.420	+0.030
MCTD-ME-1 (150M)	0.460	0.460	+0.000	0.290	0.290	+0.000	0.370	0.370	+0.000
MCTD-ME-1 (3B)	0.500	0.420	−0.080	0.340	0.280	−0.060	0.400	0.370	−0.030
MCTD-ME (Multi-Expert)	**0.500**	**0.560**	**+0.060**	**0.350**	**0.410**	**+0.060**	**0.880**	**0.910**	**+0.030**

Table 1: **Main results on CAMEO and PDB.** Baseline and final values for AAR, normalized reward, and scTM are shown, with Δ indicating improvements. Multi-expert MCTD consistently outperforms single-expert and unguided baselines, especially on PDB targets.

work, we focus on this domain where ground truth sequences are known, leaving other applications (e.g. de novo structure generation) for future work. The evaluation uses the CAMEO 2022 dataset [10] and PDB data split ([4]), a standard benchmark of recently solved protein structures used in prior work. For each backbone structure in CAMEO, the goal is to generate an amino acid sequence that folds to that structure. We adopt the same metrics as DPLM-2 [31]: (1) Amino Acid Recovery (AAR) – the percentage of residues in the designed sequence that exactly match the native sequence (sequence recovery), and (2) self-consistency TM-score (scTM) – the TM-score measuring structural consistency between the target backbone and the structure predicted from the designed sequence. Higher AAR indicates better sequence accuracy for the given fold, while scTM reflects how well the designed sequence is likely to fold into the target structure. Following DPLM-2's protocol, we report both metrics for the CAMEO test set. (For reference, co-generation baselines like MultiFlow [4] and ESM3 [6] attain 32–47% AAR on CAMEO, while the state-of-the-art diffusion model DPLM-2 (3B) achieves 52% AAR and 0.89 scTM)

5.1.1 Baselines and Variants

We conduct an ablation study to quantify the impact of multi-expert guidance. In particular, we compare three variants of our approach:

MCTD-ME-0 (no experts): A control variant that performs the diffusion tree search with no expert guidance. The forward diffusion model's proposals are accepted at random, yielding essentially a random-fill of the sequence (no reward-driven pruning).

MCTD-ME-1 (single expert): Our MCTS-based diffusion planner using a single expert model as the critic. Here we use the DPLM-2 150M, 650M, 3B models seperately as the sole experts for scoring candidates (chosen for its strong performance and manageable runtime). This represents a single-criterion search.

MCTD-ME-E (ensemble of experts): The full multi-expert MCTD using an ensemble of three DPLM-2 models of varying scales – 150M, 650M, and 3B parameters. These experts provide a diverse set of evaluations, which are combined into a composite reward signal.

All variants use the same MCTS sampling pipeline described in Section 4. At each diffusion step (node expansion), the planner simulates rollouts to incrementally denoise the sequence. In our implementation, each expert performs 2 rollouts, and with 3 experts this yields around 6 candidate expansions per node. To limit branching, we retain the Top-2 candidates according to the multi-critic reward at each step. The MCTS selection policy uses a UCT-style formula adapted for multi-expert value estimates with an exploration constant $c_p = 1.414$. We also incorporate an entropy-aware bonus as described (Section 4.3), so that nodes with high disagreement among experts are explored more. All expert models are pre-trained (no further fine-tuning); MCTD operates as a test-time algorithm that plugs these experts into the search.

Reward Formulation: The multi-objective reward R used to guide MCTD-E is a weighted sum of the expert criteria. Specifically, we define $R = 0.6\text{AAR} + 0.35\text{scTM} + 0.05B$, where B is a small bonus from simple biophysical properties (average hydrophobicity, amino acid composition frequency, and net charge). The 60/35/5 weighting was chosen to prioritize sequence recovery while still strongly rewarding structural fidelity, plus a minor regularization to favor realistic amino acid

distributions. This composite reward design is in line with prior multi-critic strategies. In our setting, AAR and scTM are computed after a full sequence is generated (i.e. at a leaf node of the search); partial sequences during tree expansion receive interim estimates (for example, we estimate scTM for a partial sequence by filling missing residues with the most likely amino acids before structure prediction, and likewise compute a partial AAR by comparing the filled-in sequence to the native). These approximations guide the search towards promising candidates even before the sequence is complete.

5.1.2 Results

Table 1 summarizes the inverse folding results on the CAMEO dataset. Overall, MCTD-ME-E (multi-expert) achieves the best performance, outperforming both the single-expert and no-expert variants by a significant margin. MCTD-ME-E attains an average AAR of 44.5%, higher than the 43.2% achieved by the strongest single-expert run (650M, expert 0). The normalized reward of MCTD-ME-E designs averages 0.303, also exceeding the 0.295 of the 650M expert alone. By contrast, the MCTD-ME-1 (150M expert, expert 1) variant essentially matches its baseline (42.7% AAR, 0.290 reward), while the MCTD-ME-1 (3B expert, expert 2) variant actually degrades performance (37.0% AAR, 0.254 reward). This indicates that our MCTS planner by itself introduces little overhead or degradation when only one expert is used – essentially reproducing the single model's capabilities – but the ensemble of experts (MCTD-ME-E) yields a further boost: +1.3 percentage points in sequence recovery and a notable gain in normalized reward relative to the best single expert. This suggests that the ensemble is successfully steering the generation to find sequences that satisfy multiple criteria better than any single model alone. As expected, MCTD-ME-0 (random fill) performs near chance-level – we observed 38.7% AAR (roughly the random identity baseline for 20 amino acids) and very low reward (0.272), confirming that unguided diffusion yields essentially arbitrary sequences with poor folding to the target. These serve as a sanity check that improvements of MCTD-ME-1 and MCTD-ME-E are indeed due to learned guidance rather than artifact.

Importantly, MCTD-ME-E not only improves average performance but also reduces failure cases. We found that MCTD-ME-E produced valid, foldable sequences for nearly all test structures, whereas single-expert runs occasionally returned sequences with poor recovery or structural inconsistencies. The multi-expert approach mitigates this by combining perspectives: the smaller 150M expert, for instance, tends to propose more diverse amino acid choices (higher entropy), while the larger 3B expert strongly evaluates overall fitness. The MCTS search benefits from this diversity–exploitation mix, as reflected in the higher overall reward of MCTD-ME-E designs. We also observed that the advantage of multiple experts grows with problem complexity. On longer proteins (length >300), MCTD-ME-E's sequence recovery was several percentage points higher than MCTD-ME-1 on average, whereas on short proteins (<100 residues) both methods performed similarly. This trend aligns with the intuition that larger search spaces and more complex structures gain more from an ensemble of critics guiding the search. In sum, multi-expert MCTD consistently outperforms both single-expert planning and one-shot diffusion decoding on this task, achieving state-of-the-art results on the CAMEO inverse folding benchmark (to our knowledge). See Fig 3 for length-binned improvements in AAR, reward, and scTM.

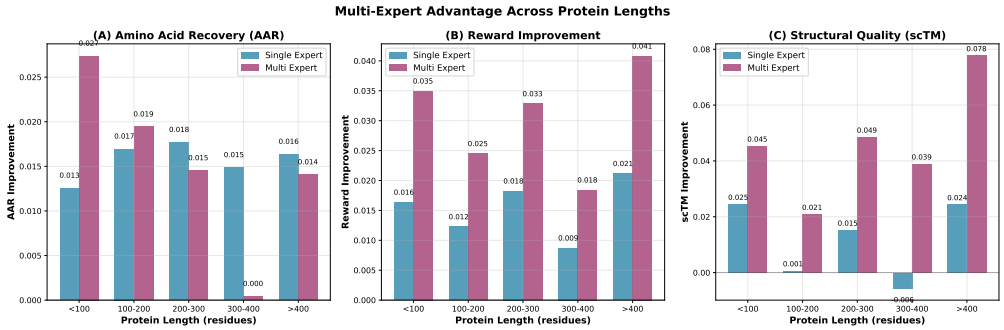

Figure 3: Comparison of three evaluation metrics (AAR, normalized reward, and scTM) across different model variants. Higher values indicate better sequence recovery and structural consistency, showing the advantage of multi-expert MCTD.

6 CONCLUSION

In conclusion, we introduced MCTD-ME, which combines diffusion-based generative planning with MCTS and multi-expert guidance. The method is novel in reframing diffusion decoding as tree search for parallel multi-token exploration and in integrating multiple expert evaluators with a pLDDT-guided masking schedule to improve biophysical fidelity. Our selection policy (P$\mathcal{H}$-UCT-ME) efficiently arbitrates among experts using cached uncertainty statistics, enabling stable and low-overhead rollouts. Empirically, MCTD-ME outperforms state-of-the-art baselines on inverse folding, scales effectively with additional computation, and provides a general template for diffusion-plus-search workflows in structured generative tasks.

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] 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.

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

[6] Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, and et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pages 2024–07, 2024.

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

[8] 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.

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

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] Xuefeng Liu, Songhao Jiang, Ian Foster, Jinbo Xu, and Rick L Stevens. Scaffold-driven gpt model for drug optimization. In *ICML 2025 Generative AI and Biology (GenBio) Workshop*, .

[19] Xuefeng Liu, Songhao Jiang, Bo Li, and Rick L Stevens. A ground-up designed controllable gpt for molecule optimization. In *ICML 2025 Generative AI and Biology (GenBio) Workshop*, .

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

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

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

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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

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

Require:
$root \leftarrow$ fully denoised baseline node y_0
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
$\mathcal{E} = \{\pi^1, \ldots, \pi^E\}$: generator experts
$\mathcal{C}$: set of critic functions
$cache \leftarrow$ empty dictionary for evaluated sequences

for $i = 1$ to T **do**
  $node \leftarrow root$
  ▷ **/* Selection via P$\mathcal{H}$-UCT-ME*/**
  while $node.children \neq \emptyset$ **do**
    $node \leftarrow$ 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 */**
  $\mathcal{M} \leftarrow$ GetMaskSet$(node.sequence)$ // Based on pLDDT
  $\mathcal{Y} \leftarrow \emptyset$ // candidate children
  for $e \in \mathcal{E}$ **do** // each generator expert
    for $r = 1$ to R **do** // rollouts
      $y_0 \leftarrow$ MaskedDiffusion$(node.sequence, \mathcal{M}, e)$
      if $y_0 \notin cache$ **then**
        $score \leftarrow$ EvalComposite$(y_0; \mathcal{C})$ // weighted sum of critics
        $cache[y_0] \leftarrow score$
      $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{y_0\}$
  $\mathcal{Y}_{\text{top}} \leftarrow$ TopK$(\mathcal{Y}, K; cache)$ // sort by composite reward
  for $y' \in \mathcal{Y}_{\text{top}}$ **do**
    $(\mathcal{U}_{\text{ent}}(y'), \mathcal{U}_{\text{div}}(y')) \leftarrow$ ComputeBonuses$(y', node.sequence, \mathcal{M})$ // ensemble surprisal & novelty
    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) */**
  for $y' \in \mathcal{Y}_{\text{top}}$ **do**
    $v \leftarrow cache[y']$ // may hit cache; same composite reward as above
    ▷ **/* Backpropagation */**
    Backpropagate(y', v) // supports max or sum backup
return Top-k sequences by $cache[\cdot]$

A.2 Dataset Details

We evaluate MCTD-ME on two protein sequence benchmarks at the *chain* level. (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 2: **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.

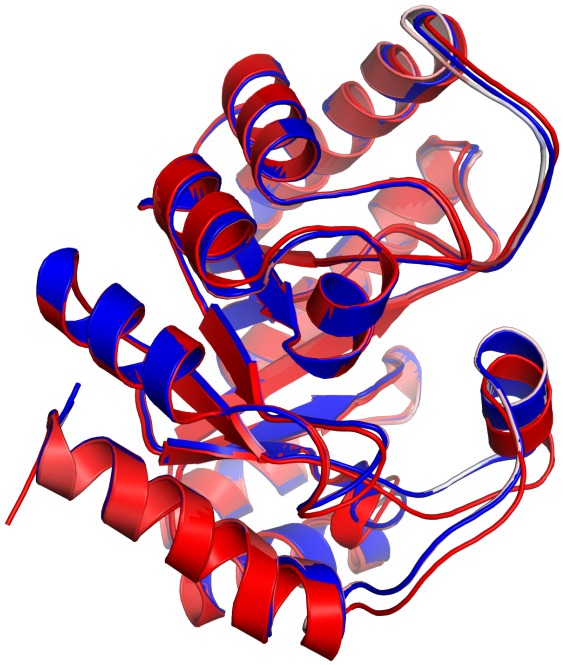

Figure 4: **CAMEO 7dz2_C overlay.** Red: target reference. Blue: baseline (starting point). Multicolor: MCTD design, colored per residue from red (near reference) to blue (far), after structural alignment. The shift from blue toward red across many regions visualizes how MCTD moves the baseline toward the reference via pLDDT-aware masking and multi-expert guided rollouts.

A.3 Case Study on CAMEO data

Fig. 4 overlays three structures to illustrate how MCTD-ME steers sequences toward the target fold. The red ribbon is the CAMEO reference (ground truth), the blue ribbon is the structure predicted from the baseline sequence, and the multicolor ribbon is our MCTD-ME design. For the multicolor model, residues are shaded on a red $\rightarrow$ blue gradient proportional to per-residue deviation from the reference (red = near, blue = far), making local improvements visually apparent along the chain. Relative to the blue baseline, the MCTD-ME design shows widespread "reddening," especially across helices and connecting loops, indicating closer alignment to the target geometry while preserving already well-formed secondary structure. This behavior matches the algorithm's intent: pLDDT-aware masking focuses edits on uncertain segments, while multi-expert rollouts—selected via P$\mathcal{H}$-UCT-ME—propose diverse fixes that are promoted during search, yielding higher structural fidelity (e.g., scTM) without over-editing stable regions.

A.4 Setup and Hyperparameters

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.

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 3: **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.

A.5 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 4 are measured end-to-end per target (including expert rollouts and critic scoring).

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×)	314.0	800.2	2187.2

Table 4: **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).

A.6 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 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.