Retrosynthesis Planning via Worst-path Policy Optimisation in Tree-structured MDPs

Mianchu Wang[1] and Giovanni Montana[1,2]

[1]University of Warwick [2]The Alan Turing Institute

UNIVERSITY OF WARWICK

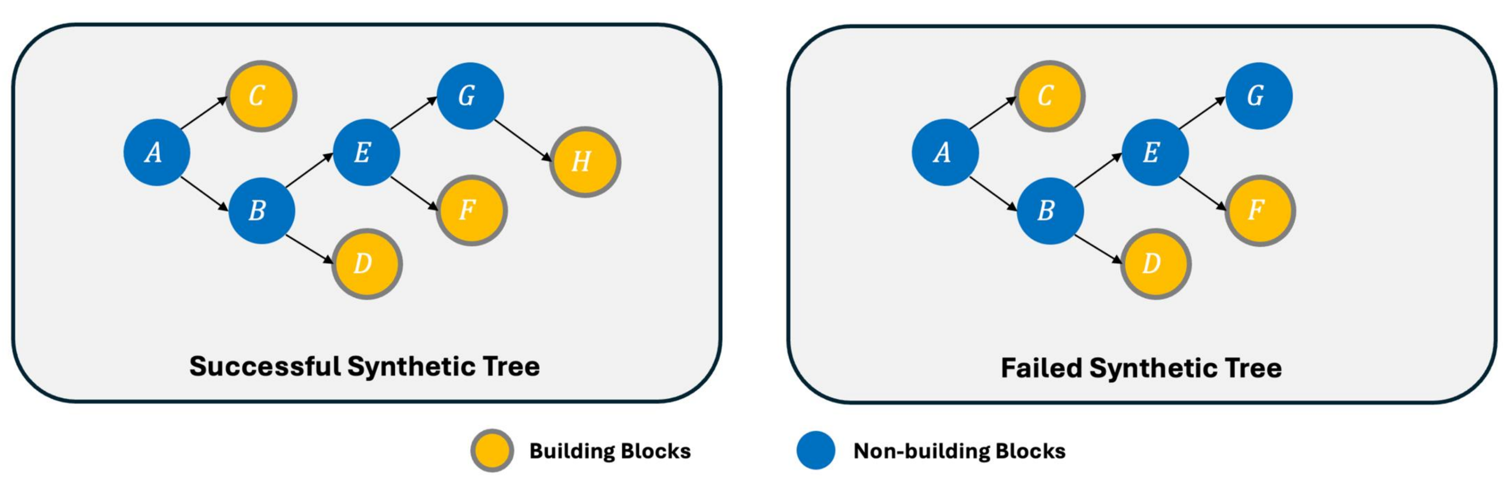
NEURAL INFORMATION PROCESSING SYSTEMS

A Brief Introduction

Retrosynthesis planning aims to decompose target molecules into available building blocks, forming a synthetic tree where each internal node represents an intermediate compound and each leaf ideally corresponds to a purchasable reactant. However, this tree becomes invalid if any leaf node is not a valid building block, making the planning process vulnerable to the “weakest link” in the synthetic route. Existing methods often optimise for average performance across branches, failing to account for this worst-case sensitivity.

In this paper, we reframe retrosynthesis as a worst-path optimisation problem within tree-structured Markov Decision Processes (MDPs). We prove that this formulation admits a unique optimal solution and provides monotonic improvement guarantees. Building on this insight, we introduce Interactive Retrosynthesis Planning (InterRetro), a method that interacts with the tree MDP, learns a value function for worst-path outcomes, and improves its policy through self-imitation, preferentially reinforcing past decisions with high estimated advantage.

We formulate the retrosynthesis planning problem into tree-structured MDPs.



Examples illustrating the tree MDP formulation.

As shown in the figure, each non-leaf node represents a molecule that is decomposed into one or more reactants. **Left tree**: A successful synthetic route for target molecule A. It contains 4 root-to-leaf paths:

$$P(\tau) = \{ABD, ABEF, ABEGH, AC\}.$$

Since all leaf nodes are building blocks, each path receives a value of γ^T, where T is the path length. The tree's overall value is

$$\min_{p \in P(\tau)} \{\gamma^2, \gamma^3, \gamma^4, \gamma\} = \gamma^4,$$

determined by the longest path. **Right tree**: A failed synthesis attempt for molecule A. One of its paths, $ABEG$, terminates at G, which is not a building block. This gives path $ABEG$ a value of 0, making the tree's overall value

$$\min_{p \in P(\tau)} \{\gamma^2, \gamma^3, 0, \gamma\} = 0,$$

illustrating why a single failing path invalidates the entire route.

We develop a weighted self-imitation algorithm to optimise a worst-path objective.

Given a molecule s as the root node, a Q-function estimates the expected worst-path return when first expanding s with reaction a and subsequently following policy π:

$$Q^\pi(s, a) = \mathbb{E}_{\tau\sim\pi}\left[\min_{p\sim P(\tau)} \sum_{t=0}^{T} \gamma^t r(s_t) \mid s_0 = s, a_0 = a\right]$$

Here, s_0 and a_0 denote the root molecule and the initial reaction, respectively. Similarly, the value function $V^\pi(s)$ represents the expected worst-path return when all subsequent reactions follow policy π:

$$V^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\min_{p\sim P(\tau)} \sum_{t=0}^{T} \gamma^t r(s_t) \mid s_0 = s\right]$$

With these definitions in place, we can express the advantage function, which quantifies the relative benefit of applying reaction a to molecule s compared to following the policy:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$$

A positive advantage indicates that reaction a leads to better outcomes than the policy's average behaviour.

Our objective is to find a policy that maximises the worst-path objective. We achieve this through an iterative procedure: in each iteration i, we aim to find an improved policy π^{i+1} by imitating advantageous state-action pairs (s, a) experienced under policy π^i. The advantage $A^{\pi^i}(s, a)$ quantifies this, and the learning objective for π^{i+1} is formulated as:

$$J(\pi^{i+1}) = \mathbb{E}_{s\sim d_{\pi^i}(\cdot), a\sim\pi^i(\cdot \mid s)}\left[\exp\left(\beta A^{\pi^i}(s, a)\right) \log \pi^{i+1}(a \mid s)\right]$$

where $\beta > 0$ is the advantage coefficient controlling the strength of advantage weighting, and d_{π^i} is the state distribution induced by policy π^i. In this case, reactions with higher advantages receive higher weights, guiding the policy toward better-than-average reactions.

We propose InterRetro, a search-free approach to multi-step retrosynthesis planning.

Algorithm 1 Interactive retrosynthesis planning (InterRetro).

Input: pre-trained one-step policy π_θ, value function V_ϕ, training set $\mathcal{D}$, replay buffer $\mathcal{B}$.

```
def EXPLORE(π_θ, m):
  tree ← Tree(root = m)
  q ← {m}
  step ← 0
  while q ≠ ∅ and step < max_steps do
      s ← q.pop()
      a, S_r ← π_θ.get_reactants(s)
      tree.add_branch(s, a, S_r)

      # Add non-building blocks
      q ← q ∪ { s' ∈ S_r | s' ∉ S_bb }

      step ← step + 1
  end while
  return tree
```

```
def INTERRETRO(π_θ, m):
  for i = 1, . . . , I do
      while D is not empty do
          m ← D.pop()
          tree ← EXPLORE(π_θ, m)
          brs ← {}
          for each subtree τ ∈ tree do
              if τ is successful then
                  brs ← brs ∪ ALLBRANCHES(τ)
              end if
          end for
          B.append(brs)
          branches ← B.sample()
          V_φ.update(branches)              ▷ Eq. 15
          π_θ.update(V_φ, branches)         ▷ Eq. 16
      end while
  end for
```

We demonstrate that InterRetro is SOTA.

- success rate (achieving up to 100% on benchmark datasets),
- route length (reducing steps by 4.9%), and
- sample efficiency (reaching 92% of full performance with only 10% of training data).

Table 1: Performance evaluation on three benchmarks. The evaluation metrics include the success rate under different test molecules with different budgets of model calls, which are direct generation (DG), 100, 200 and 500 model calls. The DG columns are single-step model's results without search.

		Retro*-190				ChEMBL-1000				GDB17-1000			
Single-step	**Search**	DG	100	200	500	DG	100	200	500	DG	100	200	500
Template	MCTS	20.00	43.68	47.37	62.63	32.00	45.60	68.80	71.90	3.00	3.20	3.70	4.50
Template	Retro*	20.00	38.42	58.42	75.26	32.00	69.10	72.00	74.70	3.00	5.40	6.60	7.50
LocalRetro	MCTS	22.10	44.21	57.36	62.10	47.30	62.70	69.10	75.00	4.60	14.00	16.70	20.30
LocalRetro	Retro*	22.10	58.94	64.73	73.68	47.30	74.80	80.40	82.40	4.60	18.90	22.20	28.80
MEGAN	Retro*	8.42	60.52	62.10	73.15	38.00	71.70	75.40	79.00	6.20	37.60	45.70	57.20
Graph2Edits	Retro*	16.84	41.05	50.00	56.31	47.10	68.70	78.80	80.70	5.90	18.20	24.00	32.20
Self-improve	Retro*	–	67.37	83.16	94.74	–	–	–	81.10	–	–	–	15.00
PDVN	Retro*	–	93.68	97.37	98.95	–	–	–	83.50	–	–	–	26.90
RetroCaptioner	Retro*	5.26	68.94	72.63	85.26	3.90	72.60	76.50	78.70	3.20	56.20	68.20	75.20
DreamRetroer		32.10	78.94	88.42	90.52	31.10	78.10	81.70	83.10	4.20	27.36	28.97	33.20
InterRetro	MCTS	**95.78**	89.47	98.94	**100.00**	**93.10**	78.40	89.30	97.50	**89.00**	80.80	96.10	**99.50**
InterRetro	Retro*	**95.78**	**96.31**	**100.00**	**100.00**	**93.10**	**91.40**	**96.20**	**98.20**	**89.00**	**83.80**	**96.50**	97.20

References

Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. *Retro*: Learning retrosynthetic planning with neural guided A* search*. ICML 2020.

Guoqing Liu, Di Xue, Shufang Xie, Yingce Xia, Austin Tripp, Krzysztof Maziarz, Marwin Segler, Tao Qin, Zongzhang Zhang, and Tie-Yan Liu. *Retrosynthetic planning with dual value networks.* ICML 2023.

Due to a visa delay, the first author is unable to attend the conference in person. If you have any questions about the project, please feel free to contact him at mianchu.wang@outlook.com or via WeChat by scanning the QR code.