Retro*: Learning Retrosynthetic Planning with Neural Guided A* Search

Binghong Chen, Chengtao Li (Galixir), Hanjun Dai (Google Brain), Le Song

Georgia Institute of Technology

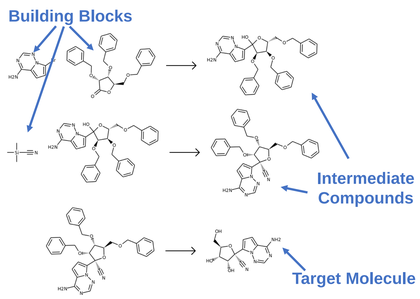
Background

Retrosynthesis Problem

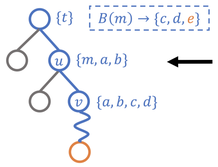
Fundamental problem in chemistry / drug discovery / material science.

Given a target molecule, the task is to figure out **a series of reactions** that lead to the synthesis of the molecule.

Criterion for `better routes` may vary:
- shorter with higher yields.
- more economically efficient.
- more environmentally friendly.
- …

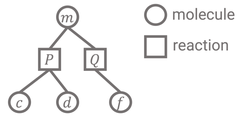


Existing Planners



Monte Carlo Tree Search
- Rollout time-consuming and comes with high variance.
- Sparsity in variance estimation.

Proof Number Search
- Formulation mismatch.
- Hand-designed criterion during search, hard to tune and generalize.



Problem Statement

One-step Retrosynthesis

Input: a target molecule t.

Output: k possible reactions that could lead to the synthesis of t in one step.

Note: In retrosynthesis planning, we assume a good one-step model is given.

$$B(\cdot): \quad t \rightarrow \{R_i, \mathcal{S}_i, c(R_i)\}_{i=1}^{k}$$

B : One-step retrosynthesis model
t : target molecule
R_i : reaction
$\mathcal{S}_i$: the set of predicted reactants
$c(R_i)$: cost of reaction

Retrosynthesis Planning

Input:
- a target molecule t
- a set of building blocks M
- a one-step retrosynthesis model B

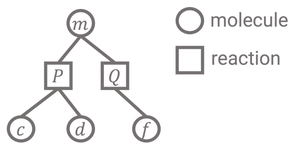
Output: a series of possible reactions predicted with B that start with molecules in M and ultimately lead to synthesis of t.

Note: We consider the following criteria:
- High quality
  - Routes should be chemically sound with high probability
  - Reactants and reactions should be of low costs
- Efficient
  - Shorter routes are preferred

We assume these can be captured by $c(\cdot)$.

AND-OR Tree Representation

Formulation
- Each molecule is encoded as an `OR` node (like m), requiring at least one of children to be solved.
- Each reaction is encoded as an `AND` node (like P), requiring all children to be solved.

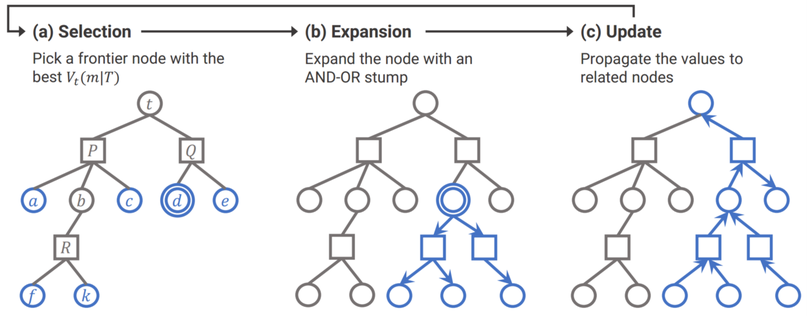


Example:

Reaction P: $molecule\ c + molecule\ d \rightarrow molecule\ m.$

Reaction Q: $molecule\ f \rightarrow molecule\ m.$

Algorithm Framework



Algorithm Details

Key Idea: Prioritize the synthesis of the molecules in the current *best plan*.

Definition of $V_t(m|T)$: under the current search tree T, the cost of the current best plan containing m for synthesizing target t.

Algorithm 1: $\texttt{Retro}^*(t)$

Initialize $T = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} \leftarrow \{t\}, \mathcal{E} \leftarrow \emptyset$;
while *route not found* **do**
  $m_{next} \leftarrow \text{argmax}_{m \in \mathcal{F}(T)} V_t(m)$; (a) Select the most promising frontier node
  $\{R_i, \mathcal{S}_i, c(R_i)\}_{i=1}^{k} \leftarrow B(m_{next})$; (b) Expand the node with one-step model
  for $i \leftarrow 1$ **to** k **do**
    Add R_i to T under m_{next};
    for $j \leftarrow 1$ **to** $|\mathcal{S}_i|$ **do**
      Add $\mathcal{S}_{ij}$ to T under R_i;
  Update $V_t(m)$ for m in $\mathcal{F}(T)$; (c) Update current estimate of V function
return route;

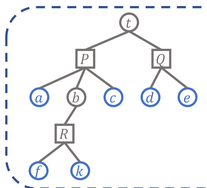
Computation of $V_t(m|T)$: using the structure of the AND-OR tree, we can decompose $V_t(m|T)$ into simpler components in a recursive fashion.

(1) Denote $V_m \equiv V_m(m|\emptyset)$, the cost of synthesizing m.

(2) Define *reaction number* $rn(\cdot|T)$: minimum estimated cost needed for a molecule/reaction to happen in the current tree.

$$rn(R|T) = c(R) + \sum_{m \in ch(R)} rn(m|T)$$

$$rn(m|T) = \begin{cases} V_m, & m \in \mathcal{F}(T) \\ \min_{R \in ch(m)} rn(R|T), & \text{otherwise} \end{cases}$$



Example for (2): $rn(t|T) = \min\big(rn(P|T), rn(Q|T)\big)$
$rn(Q|T) = c(Q) + V_d + V_e$

Example for (3): $V_t(f|T) = \underbrace{c(P) + c(R)}_{g_t(m|T)} + \underbrace{V_a + V_c + V_f + V_k}_{h_t(m|T)}$
(A^* algorithm!)

(3) Compute $V_t(m|T)$ with $rn(\cdot|T)$.

$$V_t(m|T) = \sum_{r \in \mathcal{A}(m|T) \cap \mathcal{V}^r(T)} c(r) + \sum_{m' \in \mathcal{V}^m(T), pr(m') \in \mathcal{A}(m|T)} rn(m'|T)$$

Estimating V_m from Planning Solutions

Dataset: $\mathcal{R}_{train} = \{rt_i = (m_i, v_i, R_i, B(m_i))\}$ each tuple contains target molecule m_i, best synthesis cost v_i, expert reaction R_i, and one-step retrosynthesis candidates $B(m_i)$.

Optimize

$$\min_{V_{(\cdot)}} \quad \mathbb{E}_{rt_i \sim \mathcal{R}_{train}} \Big[\mathcal{L}_{reg}(rt_i) + \lambda \mathbb{E}_{R_j \sim B(m_i) \setminus \{R_i\}} \left[\mathcal{L}_{con}(rt_i, R_j) \right] \Big]$$

Regression loss: $\mathcal{L}_{reg}(rt_i) = (V_{m_i} - v_i)^2$

Consistency loss:

$$\mathcal{L}_{con}(rt_i, R_j) = \max \left\{ 0, v_i + \epsilon - c(R_j) - \sum_{m' \in \mathcal{S}_j} V_{m'} \right\}$$

Guarantees on finding the optimal solution

Theorem 1 *Assuming V_m or its lowerbound is known for all encountered molecules m, Algorithm 1 is guaranteed to return an optimal solution, if the halting condition is changed to "the total costs of a found route is no larger than $\text{argmin}_{m \in \mathcal{F}(T)} V_t(m)$".*

Proof
Similar to A^* admissibility proof.

Remark
0 is the lowerbound of V_m for any molecule m if cost is defined as the negative log-likelihood.

Experiments

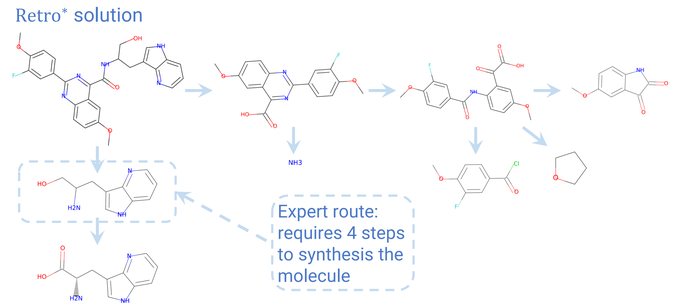


Figure: Sample solution route produced by $\texttt{Retro}^*$. Expert route requires 3 more steps to synthesize one molecule in the route.

Setting:
To create the retrosynthesis dataset, we use reactions in USPTO to build a knowledge graph from which we extract synthesis routes and split them into train/validation/test set. The available molecule list is obtained from the *eMolecules* database.

One-step model:
We trained a template-based MLP model for one-step retrosynthesis. The model learns from training set reactions and predicts top-50 templates for each product, as well as their likelihood. The templates are then applied to the product to get corresponding reactions.

Algorithm	Retro*	Retro*-0	DFPN-E	MCTS	Greedy DFS
Success rate	86.84%	79.47%	55.26%	33.68%	22.63%
Time	156.58	208.58	279.67	380.02	388.15
Shorter routes	50	52	59	30	11
Better routes	112	102	25	18	26

Performance Table: The number of shorter and better routes are obtained from the comparison against the expert routes, in terms of length and total costs.

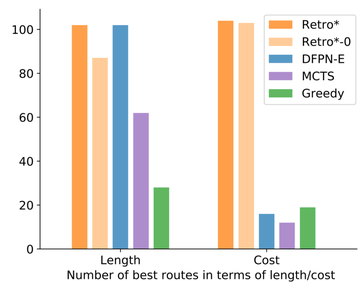


Figure: Counts of the best solutions among all algorithms in terms of length/cost.

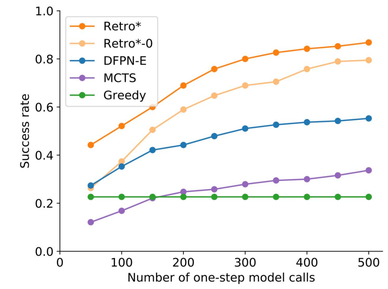


Figure: Influence of time on performance.

Baselines:
- **Greedy** - greedy Depth First Search: prioritize the reaction with the highest likelihood.
- **MCTS** - Monte-Carlo Tree Search (*Segler et al.*, 2018).
- **DFPN-E** - a variant of Proof Number Search (*Kishimoto et al.*, 2019).
- **Retro*-0** - obtained by setting V_m to 0 (ablation study).

Evaluation:
- Time: number of calls to the one-step model ($\approx 0.3s$ per call, occupying > 99% time).
- Solution quality: total costs of reactions / number of reactions (length).

Mailto: binghong@gatech.edu