Algorithm 2: Best-First Aggregation and Greedy aggregation

Input: M reasoning chains where the reasoning steps of m-th chain are denoted as $z_{i=1}^{n^m}$.
Output: Aggregated chain $\overline{z}_{1...n}$.

- **Best-First Aggregation**
for *each chain* $m = 1, 2, ..., M$ **do**
  Compute the sum of edge weights for m-th chain as $V^m = \sum_{i=m}^{n^m} V_{i-1,i}$.
end
Get the best chain among M chains by performing $m^* = argmax_m \{V^m\}$
Assign the aggregated chain as the best chain, $\overline{z}_{1...n} := \left\{ z_{i=1}^{n^{m^*}} \right\}$
- **Greedy Aggregation**
$\overline{z}_1 := z_1^{m^*}$ where $m^* = argmax_m \{V_1^m\}$.
for *each aggregation step* $i = 2, ..., n$ **do**
  for *each chain* $m = 1, 2, ..., M$ **do**
    Collect $J^m = \left\{ j, sim\left(\overline{z}_{i-1}, z_j^m\right) > 0.7; j \in n^m \right\}$.
    Get $j^{*,m} = argmax_{j \in J^m} \left\{ V_{j,j+1}^m \right\}$
  end
  Get the best next reasoning step by performing: $\overline{z}_i = z_{j*+1}^m$ where $j* = argmax_{j \in \{j^{*,m}\}_{m=1}^M} \left\{ V_{j,j+1}^m \right\}$.
end
Obtain the aggregated chain $\overline{z}_{1...n}$.

D THOUGHT STRUCTURES AGGREGATION

After completing the reasoning in Heterogeneous Tree Structures, the aggregation process of BoT first extracts the best reasoning chain from each tree and then combines them using either the Best-First or Greedy aggregation method into a single reasoning chain. More details of these two aggregation methods can be accessed in the source code *examples/BoostingOfThought/BoT_aggregator.py*.

As shown in the first block of the algorithm 16, the Best-first aggregation is a straightforward approach for aggregation as it directly extracts the chain with the highest sum of edge weights. This method is fast and stable. It typically guarantees competitive performance as the subsequent experience is able to be generated by analyzing the best chain among obtained reasoning chains. However, it can only select existing chains without making effective adjustments. Greedy aggregation is more advanced as it combines reasoning steps from different chains to produce a new, better reasoning chain with the highest edge weights. The greedy aggregation procedure in algorithm 16 contains two steps. It first collects reasoning steps that are similar to the aggregated reasoning step z_{i-1}. Thus, the next aggregated reasoning step is selected from the next reasoning steps of this collected set by maximizing the edge weights. And, sim is the similarity function that uses LLMs to assess the percentage of identical words and mathematical numbers shared between two paragraphs. 0.7 is an empirical threshold obtained from experiments.

E INFLUENCE OF THE BAD FEEDBACK

The feedback obtained by evaluating the aggregated reasoning chain with LLMs may include analysis of limited usefulness and completely incorrect conclusions and error reports. This issue typically arises due to the nature of LLMs, which are language models and do not inherently verify the accuracy of the generated text. Additionally, the capabilities of LLMs, such as gpt-3.5-turbo, are constrained when used as validators for mathematical problems.

A direct example is presented in Table 7. The analysis report concludes that "The final result obtained in Step 3 is 80, which is mathematically equal to 24." Even worse, the experience further contains that "the reasoning chain is correct" and "No errors were found in the reasoning steps.". Using the prompt with this experience as the input in the first iteration, BoT is misled to generate wrong reasoning steps, and the corresponding aggregated chain can be seen at the beginning of Table 8. It is evident