A Computing supervision through tree hashing

In every decoding step $t$, we wish to compute for every tree $z_{\text{new}}$ in the frontier $F_{t+1}$ if $z_{\text{new}} \in Z_t^{\text{gold}}$. This is achieved using tree hashing. First, during preprocessing, for every height $t$, we compute the gold hashes $h_t^{\text{gold}}$, the hash values of every sub-tree of $z^{\text{gold}}$ of height $t$, in a recursive fashion using a Merkle tree hash (Merkle, 1987). Specifically, we define:

$$\text{hash}(z) = g(\text{label}(z), \text{hash}(z_l), \text{hash}(z_r))$$

Where $g$ is a simple hash function, $z_l, z_r$ are the left and right children of $z$, and $\text{label}(\cdot)$ gives the node type (such as $\sigma$ and $\Pi$).

During training, in each decoding step $t$, since the hash function is defined recursively, we can compute the frontier hashes using the hash values of the current beam. Then, for every frontier hash we can perform a lookup to check if $\text{hash}(z) \in h_t^{\text{gold}}$. Both the hash computation and lookup are done in parallel for all frontier trees using the GPU.

B Examples for Relational Algebra Trees

We show multiple examples of relation algebra trees along with the corresponding SQL query, for better understanding of the mapping between the two.
Figure 8: Unbalanced and balanced relational algebra trees for the utterance “How many flights arriving in Aberdeen city?”, where the corresponding SQL query is

\[
\text{SELECT COUNT(\* ) FROM flights JOIN airports ON flights.destairport = airports.airportcode WHERE airports.city = 'Aberdeen'.}
\]

Figure 9: Unbalanced and balanced relational algebra trees for the utterance “When is the first transcript released? List the date and details.”, where the corresponding SQL query is

\[
\text{SELECT transcripts.transcript_date , transcripts.other_details FROM transcripts ORDER BY transcripts.transcript_date ASC LIMIT 1.}
\]
Figure 10: Unbalanced and balanced relational algebra trees for the utterance “How many dog pets are raised by female students?”, where the corresponding SQL query is

```
SELECT COUNT(*) FROM student
JOIN has_pet ON student.stuid = has_pet.stuid
JOIN pets ON has_pet.petid = pets.petid
WHERE student.sex = 'F' AND pets.pettype = 'dog'.
```

Figure 11: Unbalanced and balanced relational algebra trees for the utterance “Find the number of distinct name of losers.”, where the corresponding SQL query is

```
SELECT COUNT(DISTINCT matches.loser_name) FROM matches.
```