Appendix A MapReduce algorithms

In the appendix, we include the algorithms that are used in the running examples of the paper. More specifically, Algorithm 1 refers to the wordcount example in Section 2.1.

Algorithm 1 Wordcount example

map(Long key, String value):
1: for all word w ∈ value do
2: emit(w, “1”);
3: end for

reduce(String key, Iterator values):
4: int count = 0;
5: for all value ∈ values do
6: count += parseInt(value);
7: end for
8: emit(key, count);

In Section 3.1 we described the calculation of the positive goal by applying a single join following Algorithm 2.

Although we mentioned, in Section 3.1, that duplicate elimination should take place as soon as possible in order to minimize overhead, the description of the algorithm was deferred to this appendix. Duplicate elimination can be performed as described in Algorithm 3. Practically, the Map function emits every inferred literal as the key, with an empty value. The MapReduce framework performs grouping/sorting resulting in one group (of duplicates) for each unique literal. Each group of duplicates consists of the unique literal as the key and a set of empty values (with values being eventually ignored). The actual duplicate elimination takes place during the reduce phase since for each group of duplicates, we emit the (unique) inferred literal once, using the key, while ignoring the values.

Finally, the calculation of the final goal as described in Section 3.2 follows Algorithm 4.
Algorithm 2 Single join

\[
\text{map}(\text{Long } key, \text{ String value}): \\
\quad \text{if value.predicate == "a" then} \\
\quad \quad \text{emit}(\text{value.Z}, \{\text{value.predicate}, \text{value.X}\}); \\
\quad \text{else if value.predicate == "b" then} \\
\quad \quad \text{emit}(\text{value.Z}, \{\text{value.predicate}, \text{value.Y}\}); \\
\quad \text{end if}
\]

\[
\text{reduce(String key, Iterator values)}: \\
\quad \text{List } a\text{List} = \emptyset, b\text{List} = \emptyset; \\
\quad \text{for all value } \in \text{values do} \\
\quad \quad \text{if value.predicate == "a" then} \\
\quad \quad \quad \text{aList.add(value.X);} \\
\quad \quad \text{else if value.predicate == "b" then} \\
\quad \quad \quad \text{bList.add(value.Y);} \\
\quad \quad \text{end if} \\
\quad \text{end for} \\
\quad \text{for all } a \in a\text{List do} \\
\quad \quad \text{for all } b \in b\text{List do} \\
\quad \quad \quad \text{emit("ab(a.X, key.Z, b.Y)", ");} \\
\quad \quad \text{end for} \\
\quad \text{end for}
\]

Algorithm 3 Duplicate elimination

\[
\text{map}(\text{Long } key, \text{ String value}): \\
\quad \text{emit(value, "")}; \\
\text{reduce(String key, Iterator values)}: \\
\quad \text{emit(key, "")};
\]
Algorithm 4 Anti-join

\[
\text{map}(\text{Long } \text{key}, \text{String } \text{value}):\quad \triangleright \text{key: position in document (irrelevant)}
\]

1. if \(\text{value.predicate} \text{== "ab"} \) then \(\triangleright \text{value: document line (literal)}\)
2. emit({\text{value.X, value.Z}}, {\text{value.predicate, value.Y}});
3. else if \(\text{value.predicate} \text{== "c"} \) then
4. emit({\text{value.X, value.Z}}, \text{value.predicate});
5. end if

\[
\text{reduce}(\text{String } \text{key}, \text{Iterator } \text{values}):\quad \triangleright \text{key: matching argument}
\]

6. List \(\text{ab_list} = \emptyset\); \(\triangleright \text{values: literals for matching}\)
7. for all \(\text{value} \in \text{values} \) do
8. if \(\text{value.predicate} \text{== "ab"} \) then
9. \(\text{ab_list.add(value.Y)}\);
10. else if \(\text{value.predicate} \text{== "c"} \) then
11. return; \(\triangleright \text{matched by predicate c}\)
12. end if
13. end for
14. for all \(\text{ab} \in \text{ab_list} \) do
15. emit("abc(key.X,key.Z,ab.Y)","");
16. end for