COMP9313 Project Report

In this project, I have implemented the following methods for optimization.

Method 1: Prefix filtering

This method is used to minimize the number of items emitted from the mappers.

$P = |record|$-$\lceil |record|{*}t \rceil + 1$

Method 2: Compute the length of shared tokens

This method is used to minimize the number of computing items emitted from method 1.

Firstly, calculated the Jaccard Similarity:

$sim(r, s) = |r \cap s| / |r \cup s|$

If $sim(r, s) \geq t, I = |r \cap s| \geq |r \cup s|{*}t \geq \max(|r|, |s|)\,{*}t$

Given a record r, we can compute the prefix length as $P = |r|$-$I + 1$,

r and s is a candidate pair, they must share at least one token in the first ($|r|$-$I + 1$) tokens.

If the record r=(A,B,C,D) and P=2, the mapper emits (A,r) and (B,r).

step 1: remove doc id by flat map drop(1)

step 2: count all doc id frequency such as wordcount.scala

step 3: sort file context except id by its number (e.g. 980>600) and then its word count frequency

step 4: map all context and group by key by perfix filter method

step 5: Finding “similar” id pairs (it size >1) by sim(r, s) >= τ, l = |r intersect s| >= |r union s| * τ >= max(|r|, |s|) * τ

step 6: filter result by bigger and equal threshold and remove duplicate results