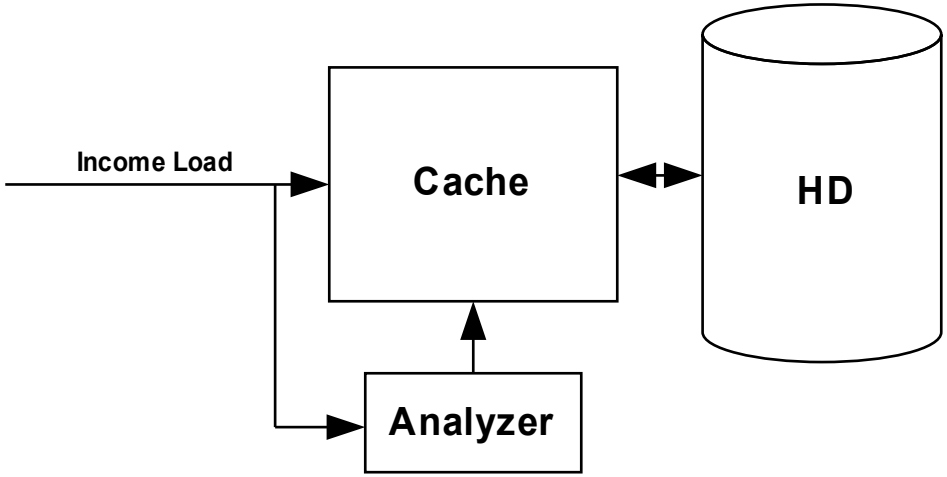
Fast Method for Dependency Mining

Mikhail Stepanov

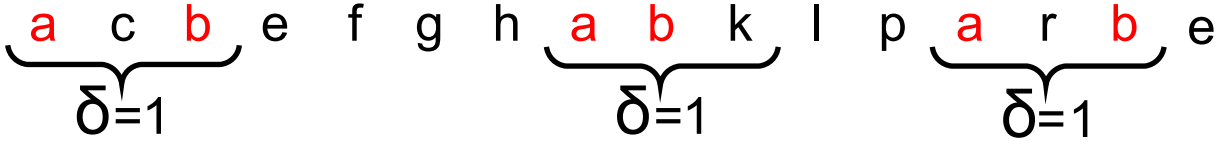
Use Case



- Analyzer detects dependent blocks by observing income load
- If some block was referenced cache prefetch depended blocks found by analyzer

Dependency of Blocks

- Let support(ab, δ) – the number of times when after reference to block a reference to block b is occurred with difference at most δ references
- Let support(a) – the number of reference to block a
- Block b is called dependent from block a if
  - support(a) ≥ m
  - $p(b \mid a) \geq p_{threshold}$, where $p(b \mid a) = \frac{support(ab, \delta)}{support(a)}$
- Proposed definition of dependency between pair of blocks could be easily generalized on the dependency between sequence of blocks, e.g. a1, a2,…,al-1 and some block, e.g. al.

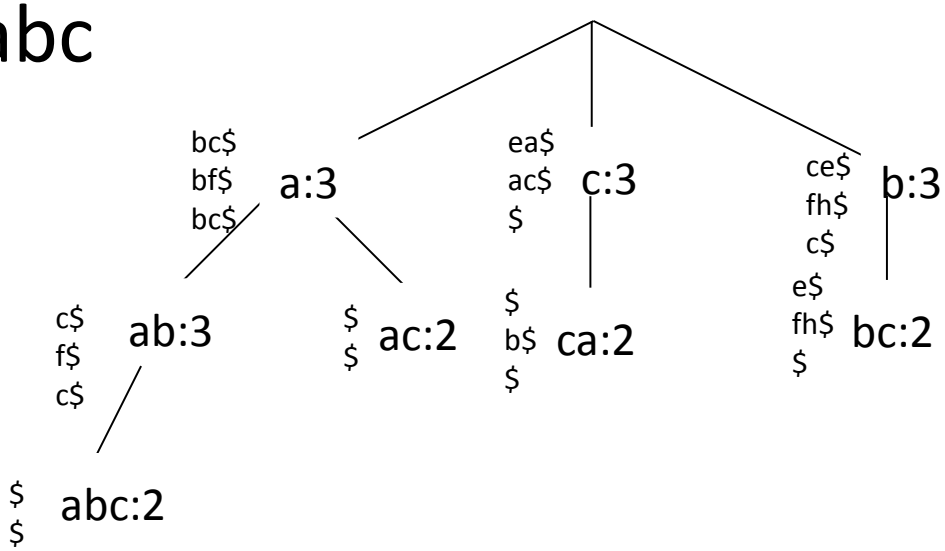


Basic Method (C-Miner)

Frequent Sequence Search

abceabfhcabc

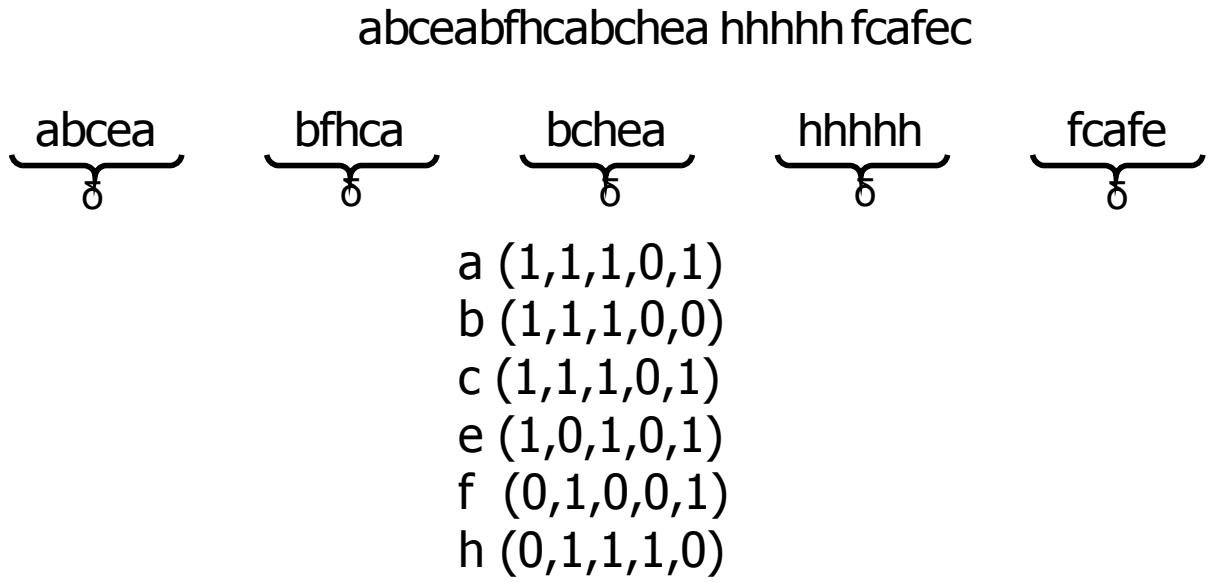
δ =1, m=2
a: bc$, bf$, bc$
b: ce$, fh$, c$
c: ea$, ab$, $



Frequent Subsequences = {ab, abc, ac, ca, a, b, c}

- Number of children in each node could reach number of unique reference in a sequence (n). Then complexity of algorithm is O(nδ)
- Size of stored data structure could exceed the sequence length

Proposed Approach



- Source sequence is split on the subsequences of equal length δ
- Each unique block is assigned activity vector
  - Length – the number of subsequences of length δ

Expected Results

- Implementation of LRU cache simulator
- Implementation of C-Miner algorithm
- Implementation of proposed approach
- Estimation of rules quality obtained by proposed approach
- Performance evaluation of LRU with rules obtained by
  - C-Miner
  - Proposed approach

Q&A