A fast approach for parallel deduplication on multicore processors

Guilherme Dal Bianco, Renata Galante, Carlos A. Heuser

Overview

- General Blocking
- MD-Approach Overview
- MapReduce Implementation
- Evaluation
- Discussion

General Blocking

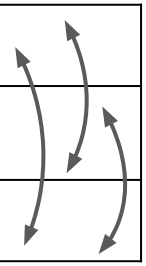
DiscID	DiscName	Genre	Year	...
1	From The Cradle - Eric Clapton	Blues	1994	...
2	Marvin Gaye - Here, My Dear	Soul	1975	...
3	The Beatles - A Hard Day's Night	Blues	1964	...
4	Eric Clapton - From the Cradle	Blues	1995	...
5	Beatles - A Hard Day's Night	Rock	1964	...
6	Curtis Mayfield - Curtis	Soul	1970	...
...	...	...	...	...

General Blocking - Blocking Key

DiscID	DiscName	Genre	Year	...
1	From The Cradle - Eric Clapton	Blues	1994	...
2	Marvin Gaye - Here, My Dear	Soul	1975	...
3	The Beatles - A Hard Day's Night	Blues	1964	...
4	Eric Clapton - From the Cradle	Blues	1995	...
5	Beatles - A Hard Day's Night	Rock	1964	...
6	Curtis Mayfield - Curtis	Soul	1970	...
...	...	...	...	...

General Blocking - Balance Problem

DiscID	DiscName	Genre	Year	...
1	From The Cradle - Eric Clapton	Blues	1994	...
3	The Beatles - A Hard Day's Night	Blues	1964	...
4	Eric Clapton - From the Cradle	Blues	1995	...
2	Marvin Gaye - Here, My Dear	Soul	1975	...
6	Curtis Mayfield - Curtis	Soul	1970	...
5	Beatles - A Hard Day's Night	Rock	1964	...



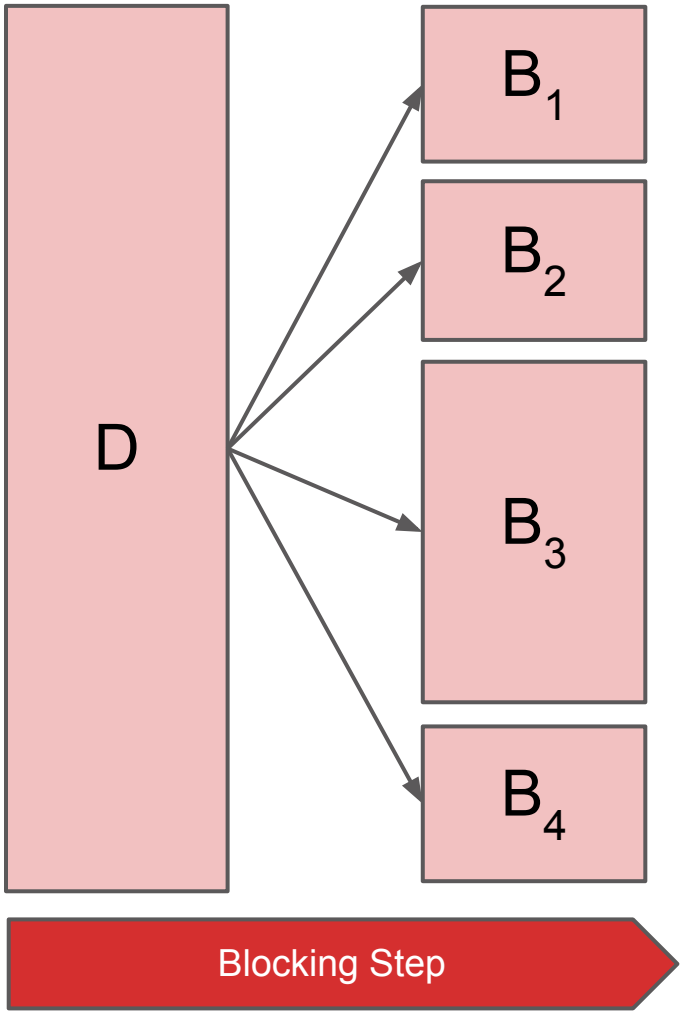
General Blocking - Keys Problem

DiscID	DiscName	Genre	Year	...
1	From The Cradle - Eric Clapton	Blues	1994	...
2	Marvin Gaye - Here, My Dear	Soul	1975	...
3	The Beatles - A Hard Day's Night	Blues	1964	...
4	Eric Clapton - From the Cradle	Blues	1995	...
5	Beatles - A Hard Day's Night	Rock	1964	...
6	Curtis Mayfield - Curtis	Soul	1970	...
...	...	...	...	...

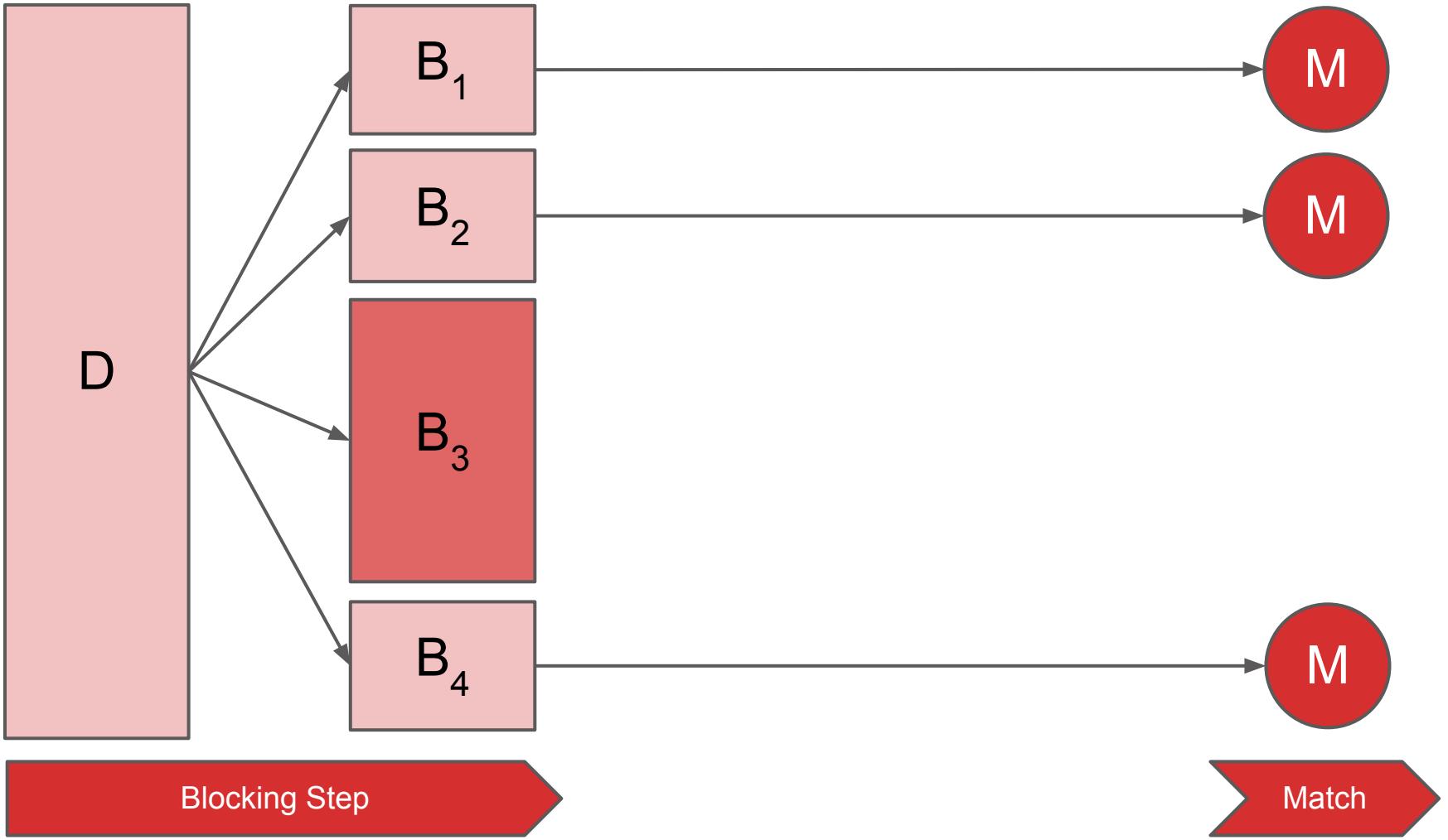
Blocking Functions & Multipass

- blocking functions are defined as followed:
  - bf_1(record) = {genre}
  - bf_2(record) = {year, genre}
  - bf_3(record) = {1^{st} 3 letters of genre, 1^{st} 3 digits of year}
- in a n-multipass several blocking functions are applied to each record
  - BFS = {$bf_{1,}$ $bf_{2,}$ ..., bf_n}

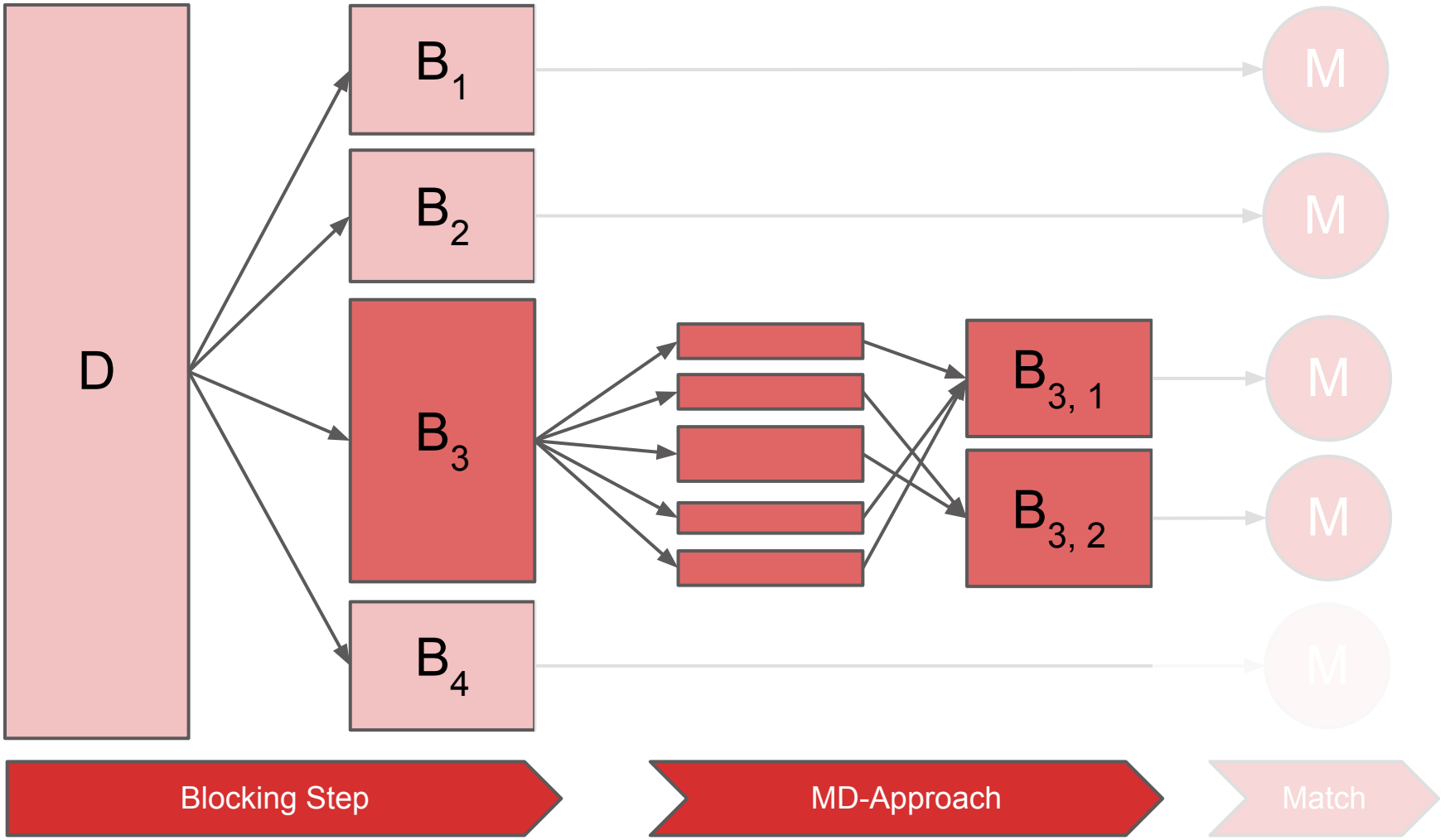
MD-Approach - Idea



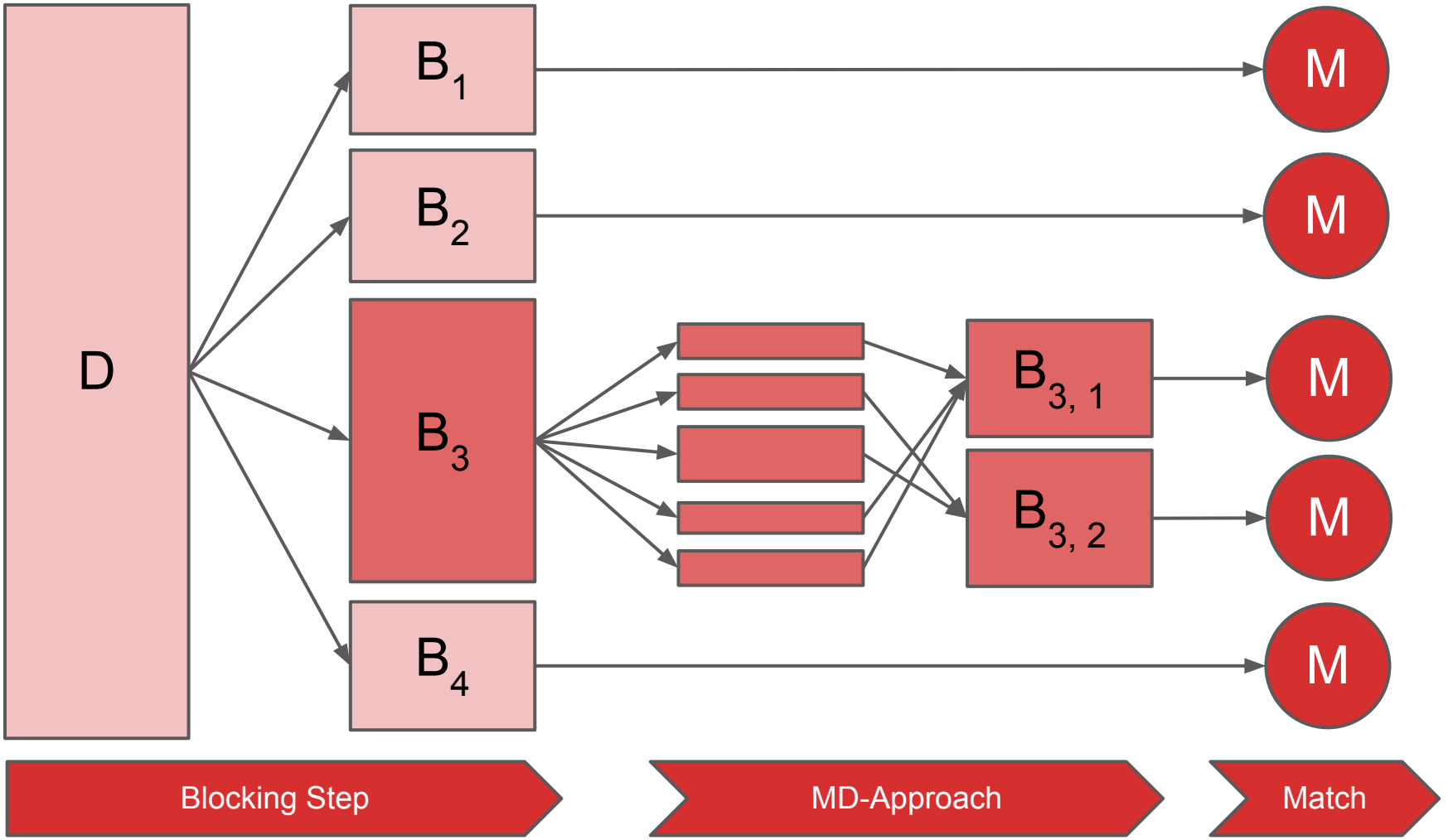
MD-Approach - Idea



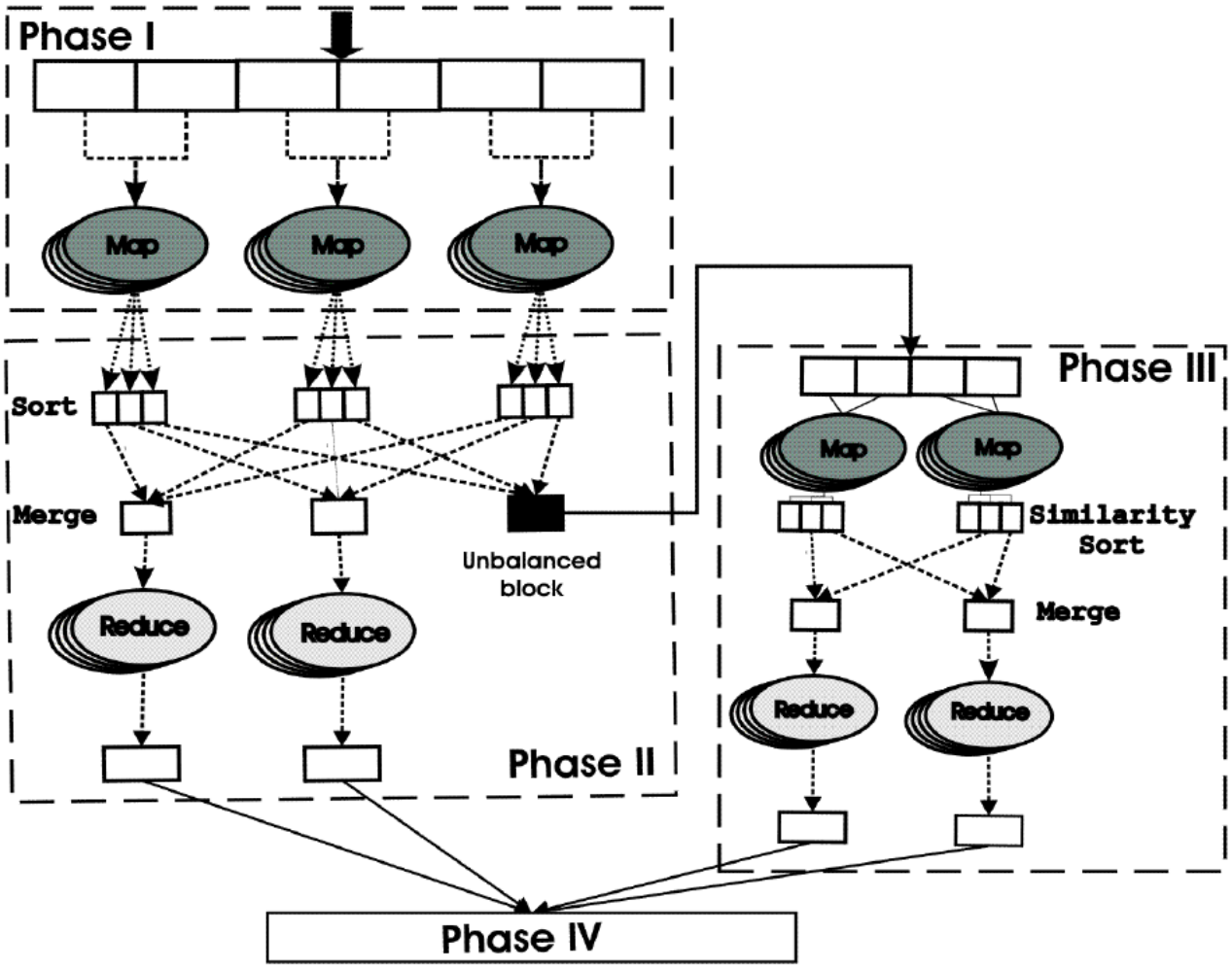
MD-Approach - Idea



MD-Approach - Idea

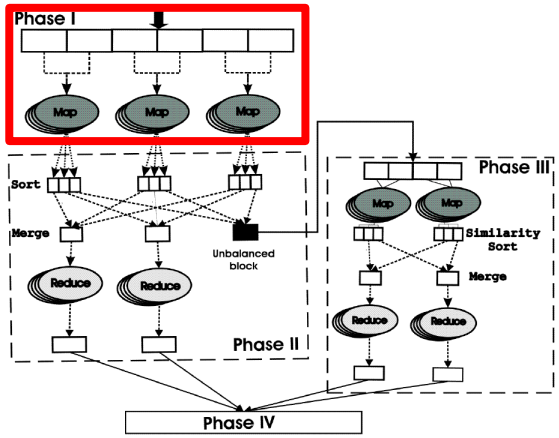


MD-Approach - MapReduce Overview



Map-Reduce Implementation
Phase I - First Blocking Step

- create dataset segments
- only map phase
- emits key-value pair
  - generated blocking key as key, e.g. bf(record) = {1st 3 letters of genre, 1st 3 digits of year}
  - record as value

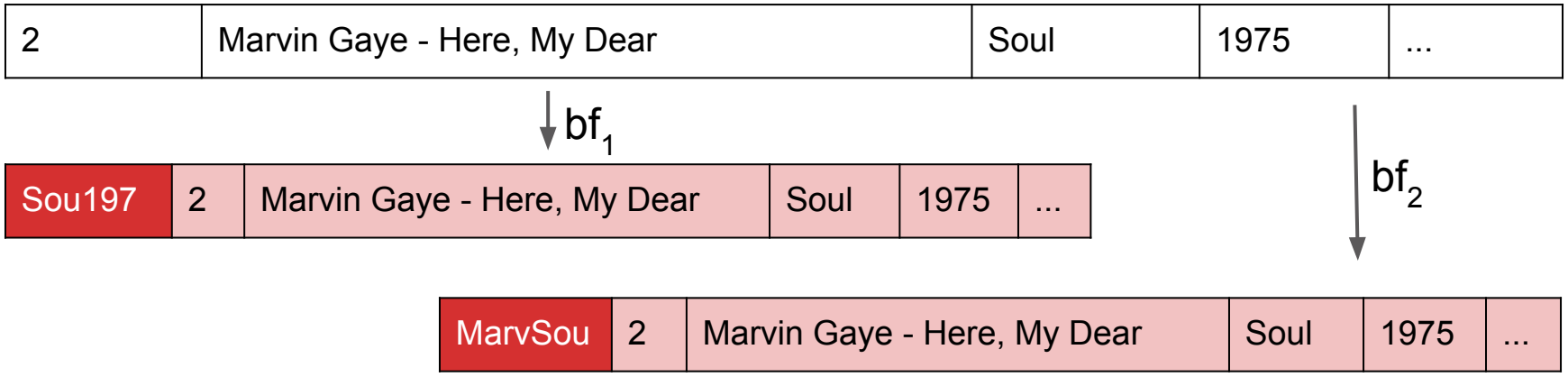


2	Marvin Gaye - Here, My Dear	Soul	1975	...

Sou197	2	Marvin Gaye - Here, My Dear	Soul	1975	...

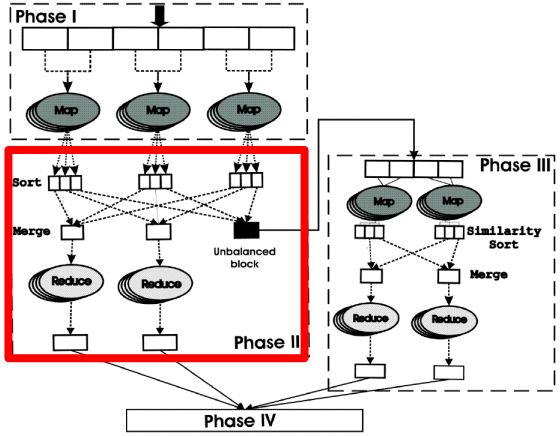
Map-Reduce Implementation Phase I - First Blocking Step

- multi-passing
  - set of n several blocking functions
    - BFS = {bf_1, bf_2, ..., bf_n}
  - for each record emit **at once**:
    - $<k_{bf1} : record_1> ... <k_{bf1} : record_n>$
      $<k_{...} : record_1> ... <k_{...} : record_n>$
      $<k_{bfn} : record_1> ... <k_{bfn} : record_n>$



Map-Reduce Implementation Phase II - Sort Blocks & Match

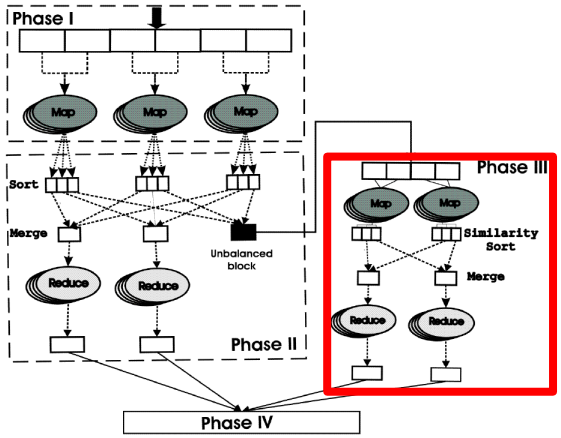
- identify unbalanced blocks
  - compare the record count of each block with a threshold
  - use reduce function until a certain threshold is reached



- reduce step (match step)
  - receives all records with the same key (here same block)
  - nested-loop pairwise comparing
  - outputs pairs of similar records

Map-Reduce Implementation Phase III - Second Blocking Step

- only unbalanced blocks
- map: expand blocking key from first blocking step
  - e.g. bf_1(record) = {1st 3 letters of genre, 1st 3 digits of year} → bf_1'(record) = {all letters of genre, all digits of year}
  - creates very fine granular blocks



Blu199	1	From The Cradle - Eric Clapton	Blues	1994	...
Blu199	4	Eric Clapton - From the Cradle	Blues	1995	...

Blues1994	1	From The Cradle - Eric Clapton	Blues	1994	...

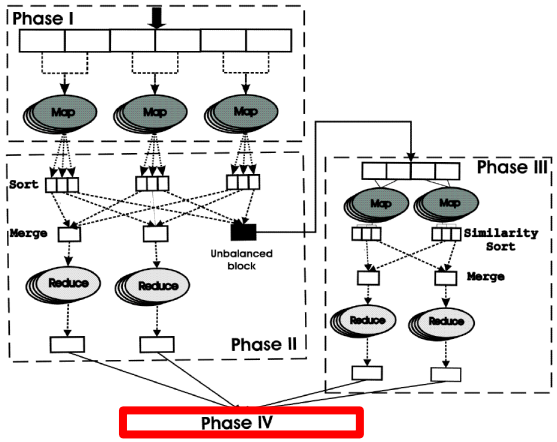
Blues1995	4	Eric Clapton - From the Cradle	Blues	1995	...

Map-Reduce Implementation
Phase III - Second Blocking Step

- to avoid loss of true positives use 'sliding window approach'
  - create an index structure for fine-grained keys after map phase
  - compare with k-nearest neighbors
  - if the similarity is high enough merge records with very similar keys to bigger blocks again
- reduce step (match) is same as in Phase II

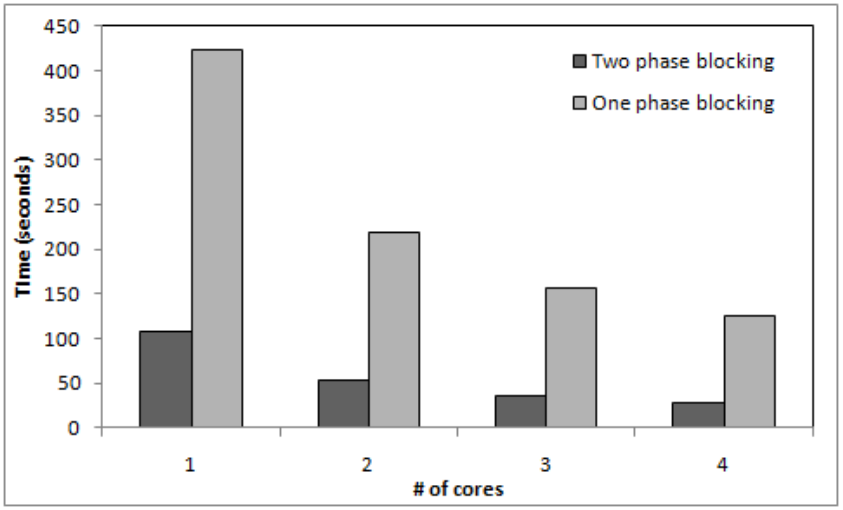
Map-Reduce Implementation
Phase IV - Merge Pairs

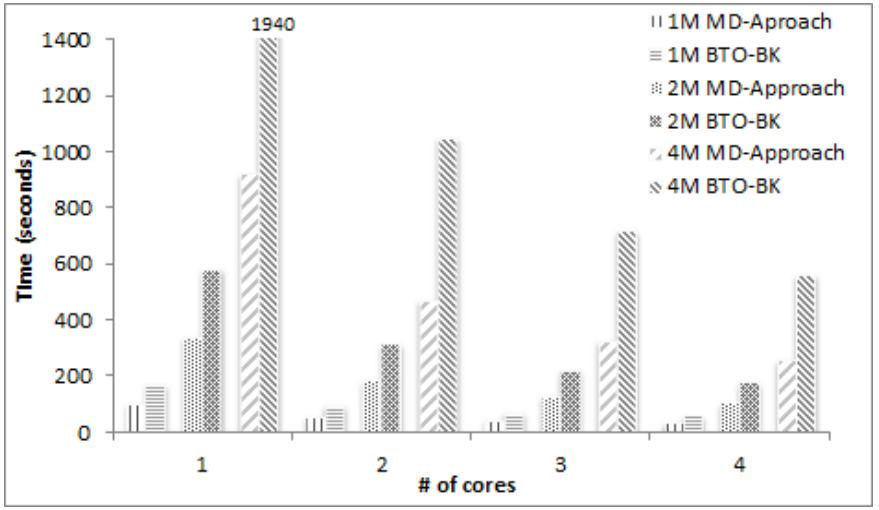
- short map-reduce operations to clean output file
  - identify and remove replicated pairs
  - multipass generates duplicates of detected records



Evaluation

- Phoenix MR framework was used for implementation - shared memory-architecture
- synthetic dataset generated by *Febrl* (1M, 2M, 4M, each with 10% duplicates)
- compared with BTO-BK
- used different similarity metrics for different approaches





Relevance for the seminar

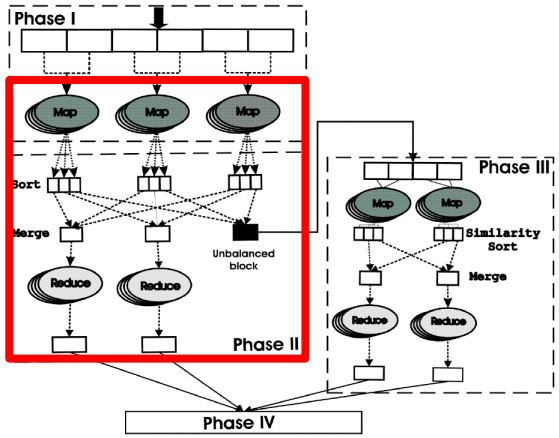
- interesting and intuitive main idea
- due to weaknesses in English language, sometimes hard to understand
- the MR-specific implementation details are very rare
- the mapping from a shared-memory (Phoenix) onto a shared-nothing (Hadoop, Stratosphere) architecture will be challenging
- to sum best things up:
  - single-run multi-pass
  - load balancing through re-blocking

Sources

1. Dal Bianco, Guilherme, Renata Galante, and Carlos A. Heuser. A fast approach for parallel deduplication on multicore processors. In Proceedings of the ACM Symposium on Applied Computing, 2011.

Map-Reduce Implementation First MR-Step

- map-step
  - emits (blocking-key, value)
- identify unbalanced blocks
- reduce-step (balanced blocks only)
  - similarity function
  - arithmetic average
  - find duplicate by threshold



Map-Reduce Implementation Second MR-Step

- map-step
  - emits expanded blocking-key
- "sliding window sort" (binary search)
- reduce-step
  - same as in First MR-Step