

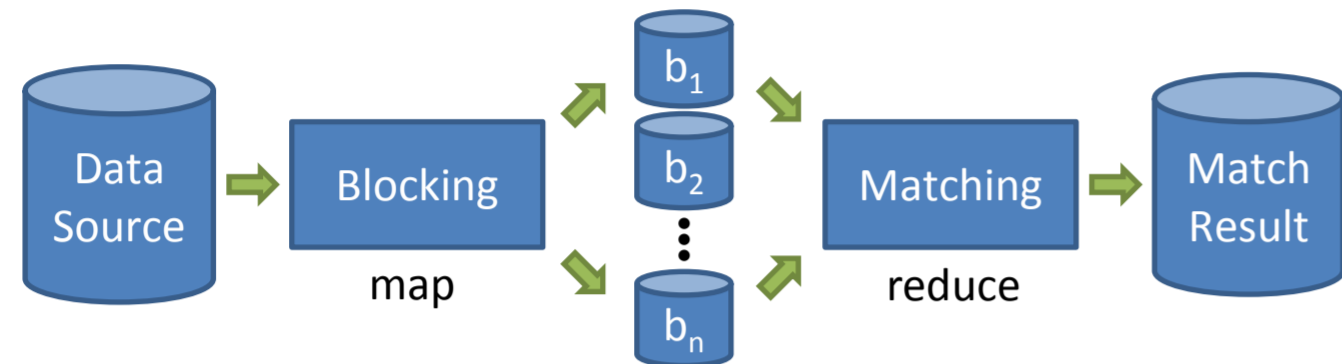
Block-based Load Balancing for Entity Resolution with MapReduce

Lars Kolb, Andreas Thor, Erhard Rahm
Database Group, University of Leipzig
<http://dbs.uni-leipzig.de>

Motivation

Entity Resolution

- Task of identifying entities referring to the same real-world object
- Application of similarity measures on pairs of input entities
 - Evaluation of Cartesian product leads to complexity of $O(n^2)$
 - Based on entity signatures (blocking keys), blocking techniques semantically group similar entities in blocks and restrict matching to entities of the same block



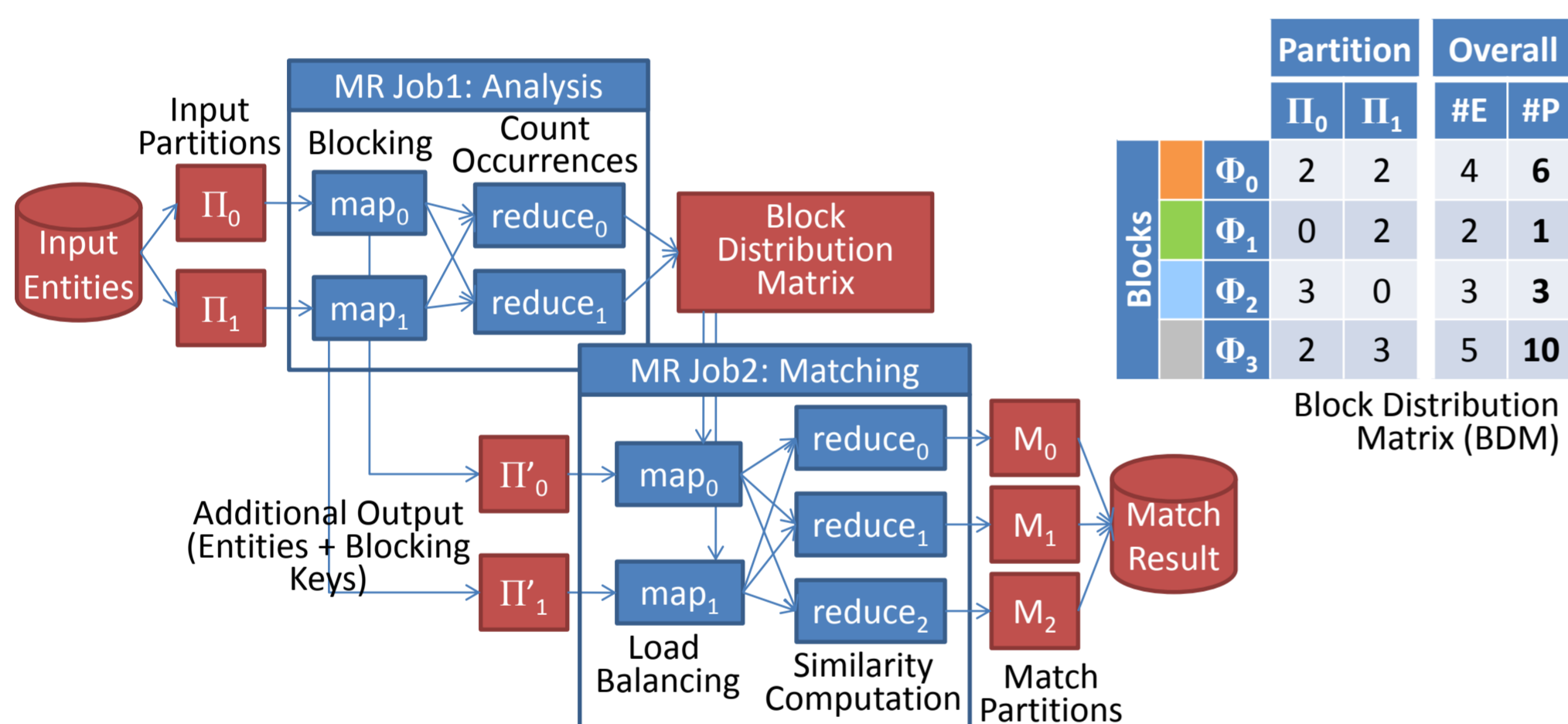
Basic approach

- Map – determine blocking key for every input entity and output (blockkey, entity) pair
- Part – partitioning by blocking key and block-wise redistribution to r reduce tasks
- Reduce – matching of entities of the same block

Goals

- Parallelization of time-intensive Blocking-based Entity Resolution with MapReduce
- Load balancing mechanism to evenly utilize available compute capacity ensuring effectiveness and scalability

Load Balancing – Overview



Idea

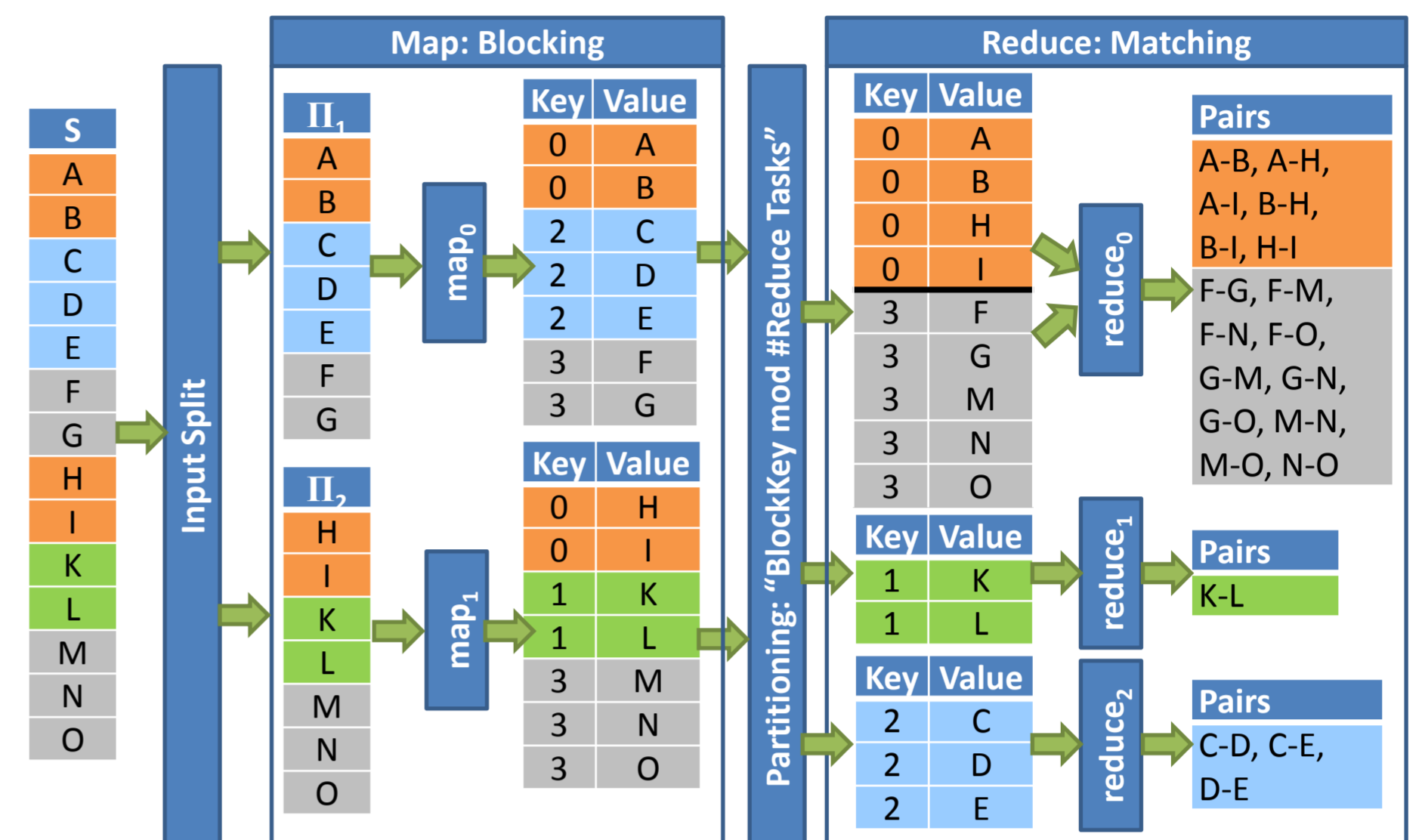
- ER processing in two MR jobs based on the same partitioning of the input data
 - Analysis job – computation of the BDM that specifies the number of entity pairs per block separated by input partitions
 - Match job – utilization of the BDM for load balancing strategies (e.g. BlockSplit) during the map phase & matching of entities in reduce phase

BlockSplit

- Generation of match tasks per block & distribution among r reduce tasks
- Large block Φ_k is split according to the input partitioning into m sub-blocks
 - m match tasks $k.i$ for matching all entities of i^{th} sub-block
 - $m(m-1)/2$ match tasks $k.i \times j$ that match Cartesian product of sub-blocks i and j
- Small block Φ_k is processed within single match task $k.*$
- Greedy load balancing – Sorting of match tasks in descending order by their size & assignment to fewest loaded reduce task (ignoring empty match tasks)

Example without Load Balancing

Basic approach ($m=2$ input partitions/map tasks, $r=3$ reduce tasks)



Problem

- Susceptible to severe load imbalances due to skewed block sizes
- Execution time dominated by a few tasks that process the largest block
- Large blocks prevent utilization of more than a few nodes

Example with Load Balancing (BlockSplit)

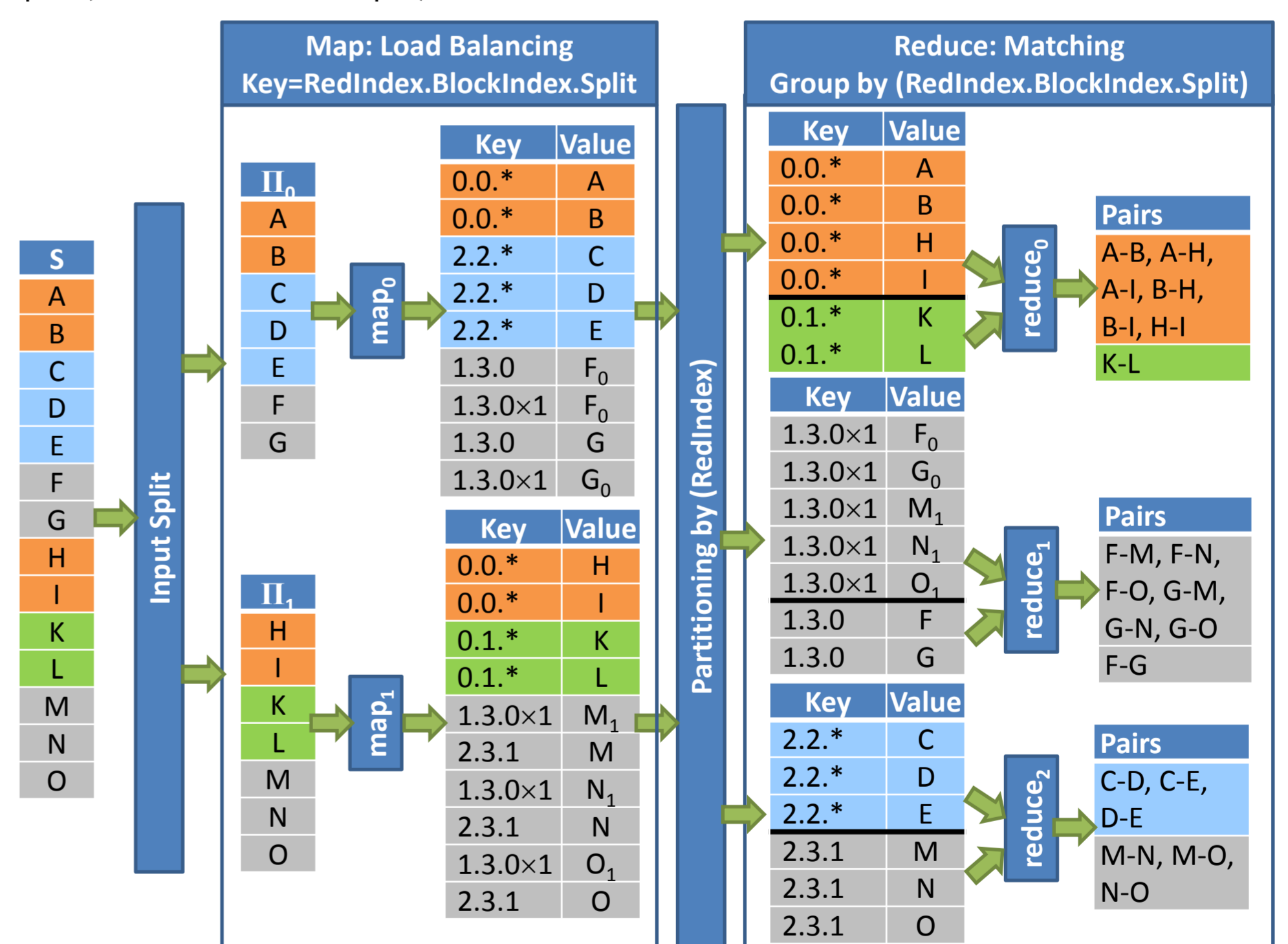
Analysis job

- Average workload per reduce task = $20/3 = 6.6$
- Large block Φ_3 ($\#P=10 > 6.6$) split in $m=2$ sub-blocks
- $\Phi_{3,0}, \Phi_{3,1} \rightarrow$ match tasks $3.0 \times 1, 3.0, 3.1$

#Comparisons	match tasks					
	0.*	3.0x1	2.*	3.1	1.*	3.0
Reduce task	0	1	2	2	0	1

Match job

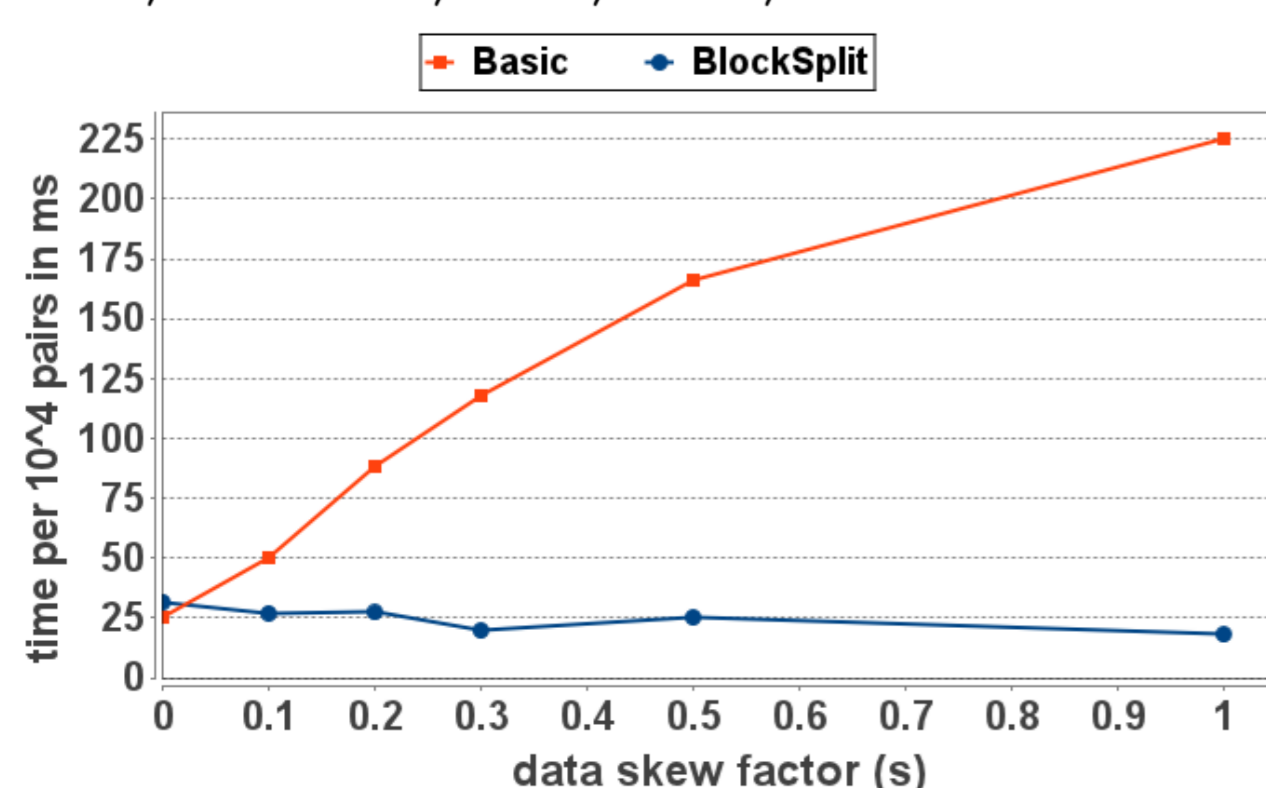
- Composite keys – `reduceTask.block.split`
- Replication of entities by map
- `part(reduceTask.block.split) = reduceTask`



Experimental Results (n =#dual core VMs, m =#map tasks, r =#reduce tasks)

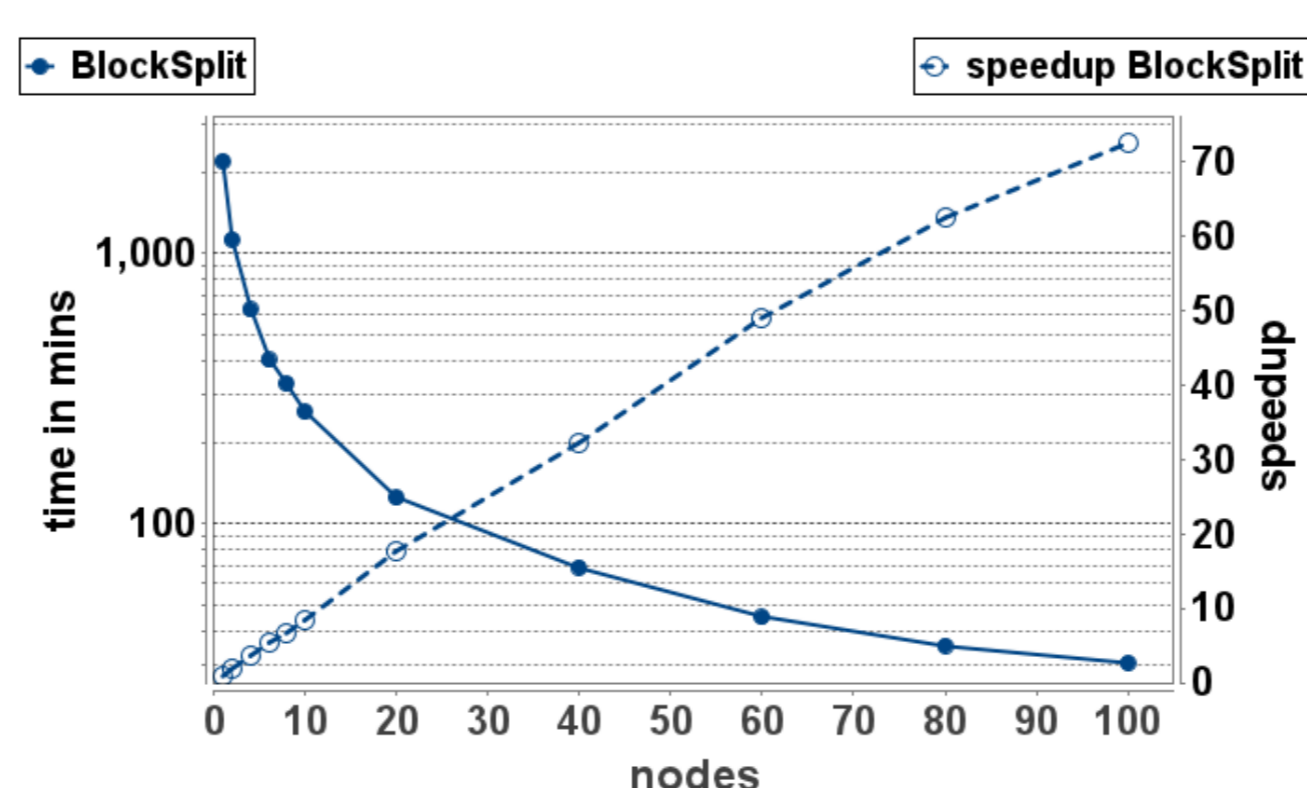
Robustness against data skew

- 100 blocks – size of k^{th} block is proportional to e^{-s^k}
- 114,000 entities, $n=10, m=20, r=100$



Scalability

- 114,000 entities $\rightarrow 3 \cdot 10^8$ comparisons
- $n \in [1, 100], m = 2 \cdot n, r = 10 \cdot n$



Related work

- L. Kolb, A. Thor, and E. Rahm. Parallel Sorted Neighborhood Blocking with MapReduce. BTW, 2011
- L. Kolb, A. Thor, and E. Rahm. Multi-pass Sorted Neighborhood Blocking with MapReduce. CSR, 2011
- L. Kolb, H. Köpcke, A. Thor, and E. Rahm. Learning-based Entity Resolution with MapReduce. CloudDB, 2011
- L. Kolb, A. Thor, and E. Rahm. Load Balancing for MapReduce-based Entity Resolution. ICDE, 2012 (to appear)