Frequent Itemsets

Hadoop Results

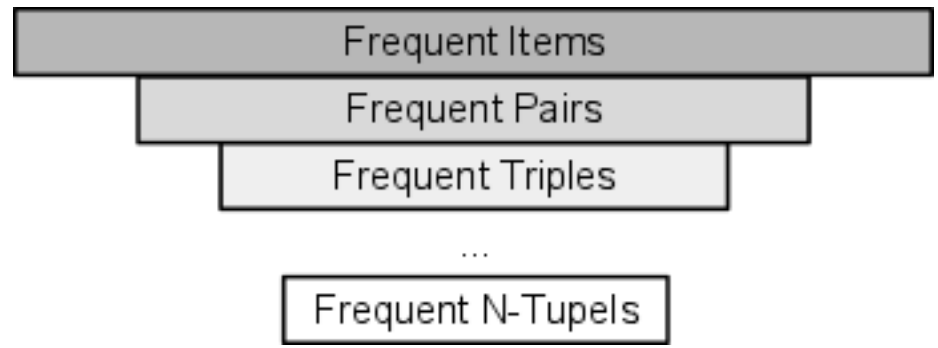
Overview

- Recap
- Current map/reduce implementation
  - filtering input data
- Evaluation on DBpedia Infobox data
- Next Steps

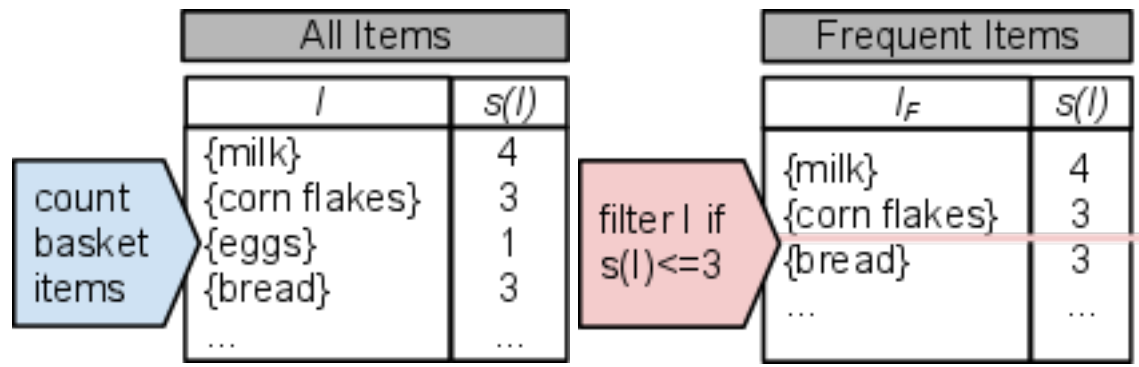
Recap

Find items that frequently occur together.

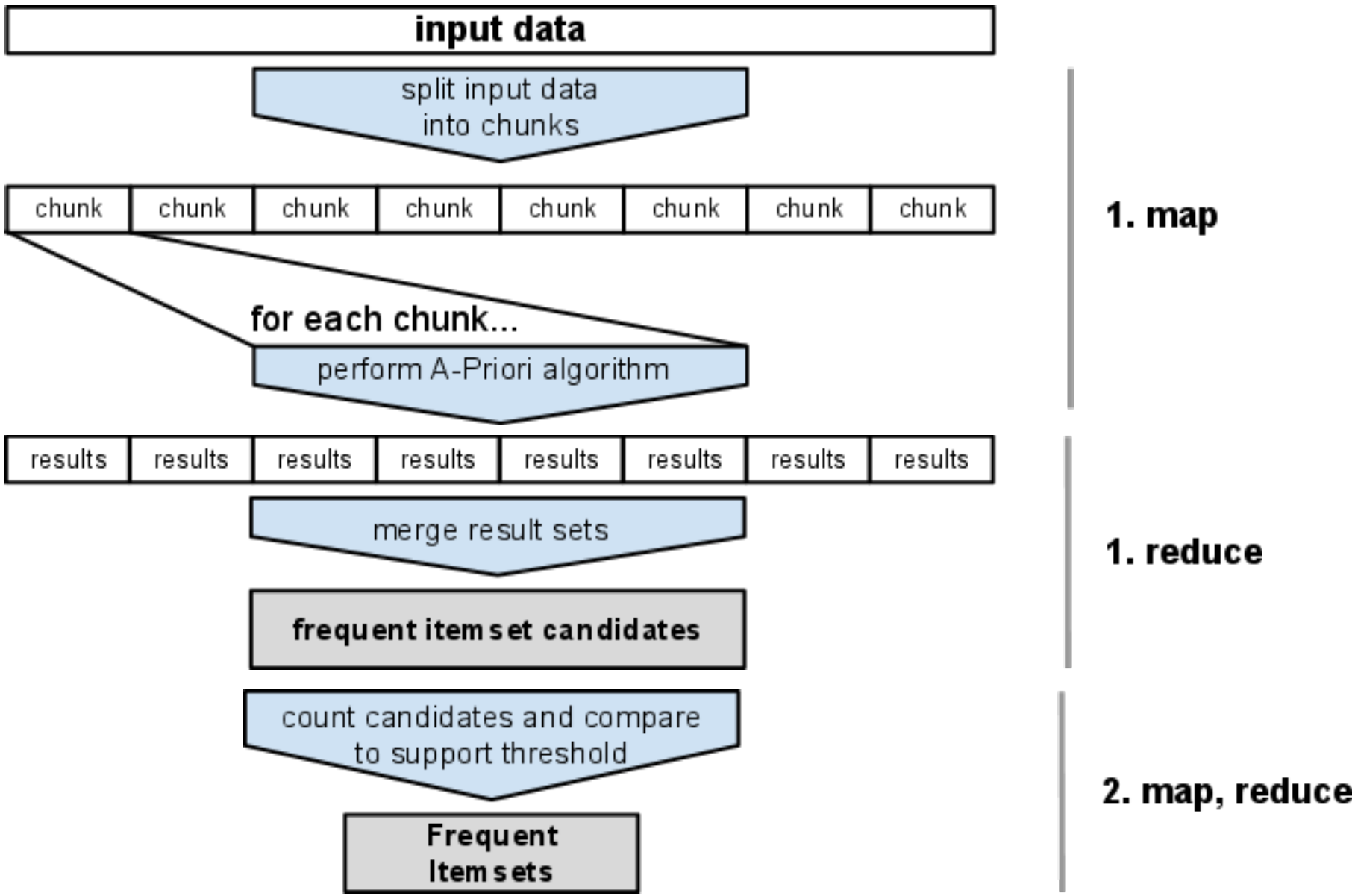
Monotonicity:
All subsets of a frequent Itemset must be frequent too.



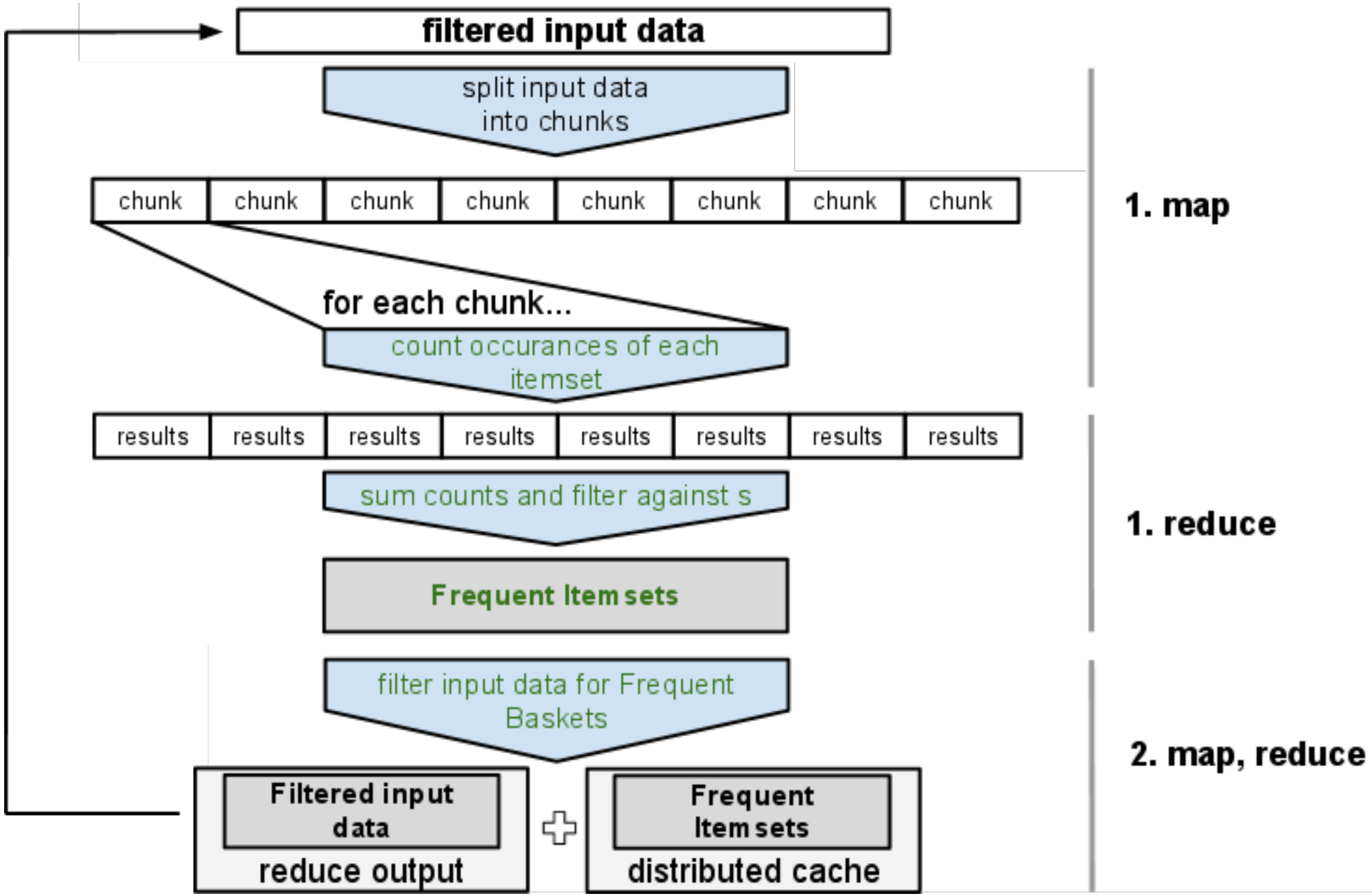
A-Priori Algorithm:
Count Items, filter infrequent
Count Pairs, filter infrequent
...



Theoretical SON Algorithm



current implementation



Filtering Input Baskets

Idea

Filter baskets that do not contain a sufficient number of frequent itemsets to be an input for the next iteration.

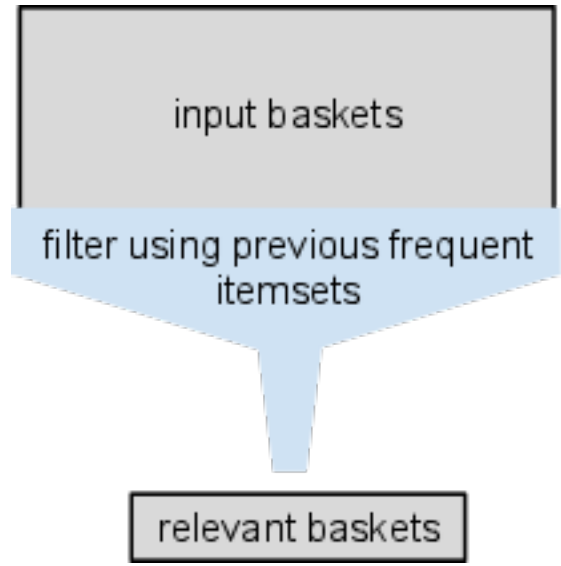
Example: Filtering irrelevant baskets before the pair-iteration.

Given: Frequent items (size=1)

Frequent Items: ({name}, {dateofBirth}, {placeOfBirth}, …)

Check: Baskets for relevance

Basket = Aristotle: (***name, dateofBirth,*** *placeOfBirth,* …)
→ count = 4, count > 1
→ Frequent Basket, keep it
Basket = A: (lc, uc, character, braille, morse, nato)
→ count = 0, count < 1
→ Infrequent Basket, filter out



Evaluation

Results

- frequent pairs in DBpedia
- frequent pair statistics

Benchmarking

- input data size
- number of reduce tasks

Results

DBpedia Ontology Infobox **Properties** (17.5 mil triples)

Locations

{country, name}	Frequency
{point, latitude}	484,461
{point, longitude}	375,864
	375,864

Animals and plant life

{kingdom, order}	155,791
{class, order}	155,504
{family, order}	150,912

Antachara	
Scientific classification	
Kingdom:	Animalia
Phylum:	Arthropoda
Class:	Insecta
Order:	Lepidoptera
Family:	Noctuidae
Subfamily:	Acronictinae
Genus:	***Antachara*** Walker, 1858

Coordinates: 52.91°N 0.12°W

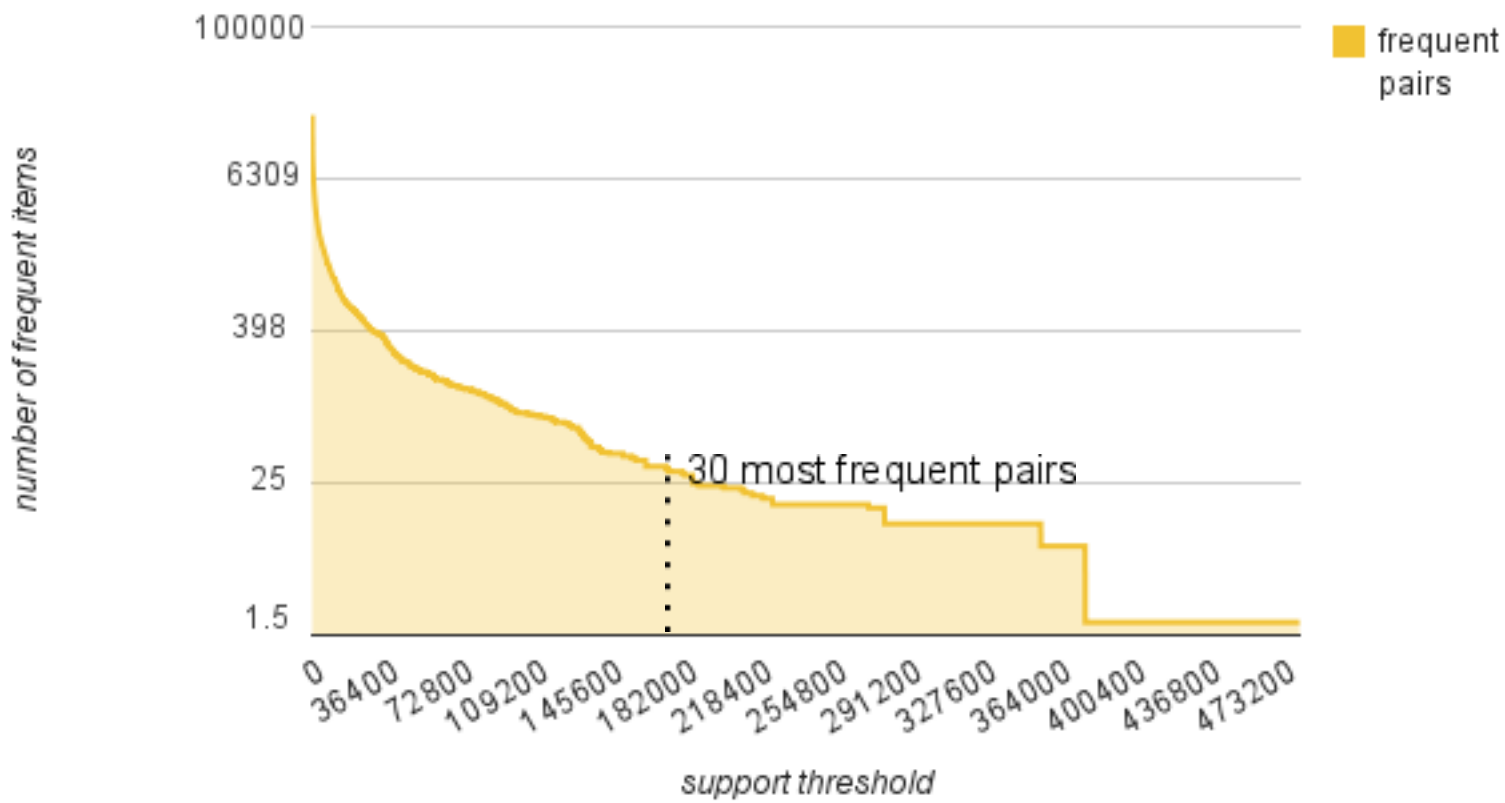
Asperton

Asperton shown within Lincolnshire

OS grid reference	TF2637
Shire county	Lincolnshire
Region	East Midlands
Country	England
Sovereign state	United Kingdom

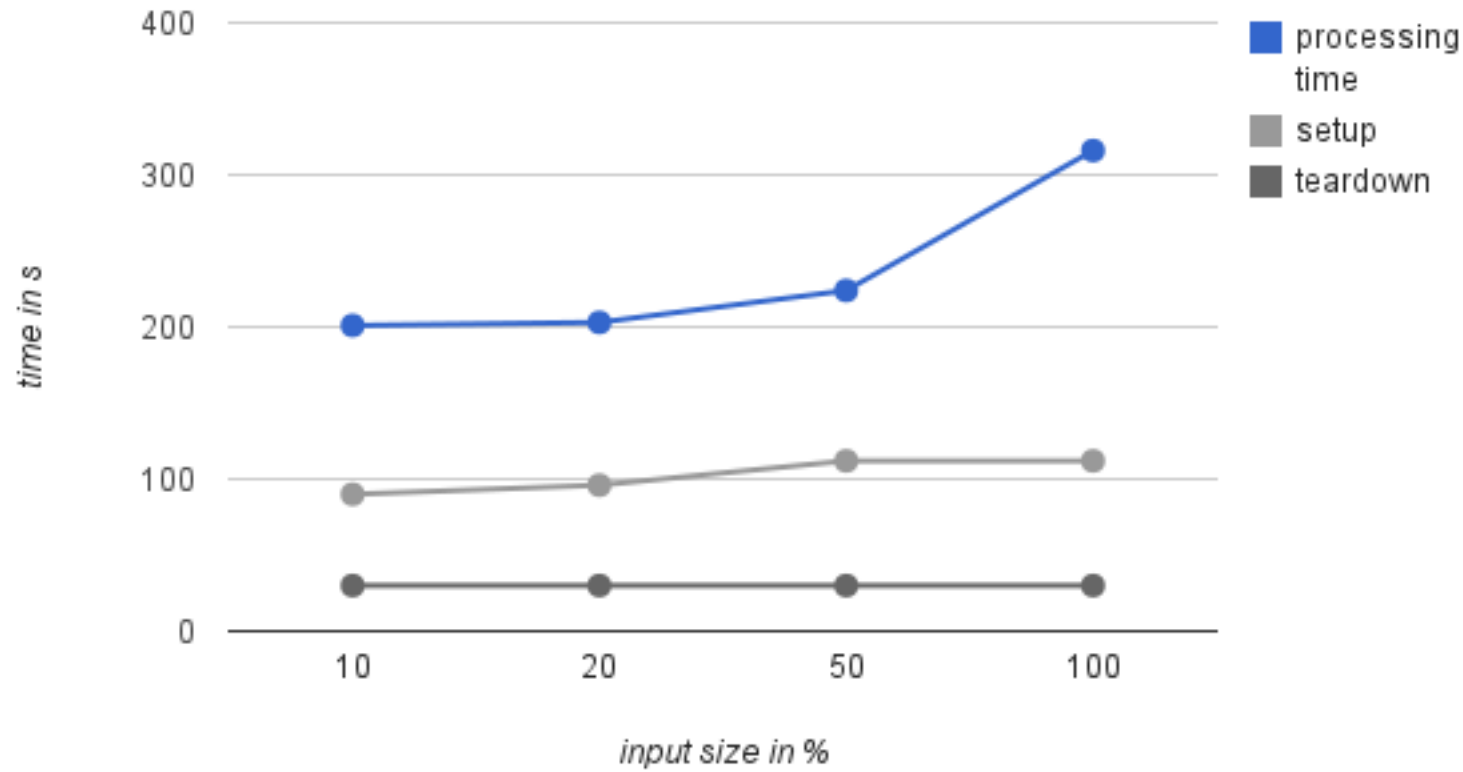
Results

Frequent Pairs statistics in DBpedia



Benchmarks

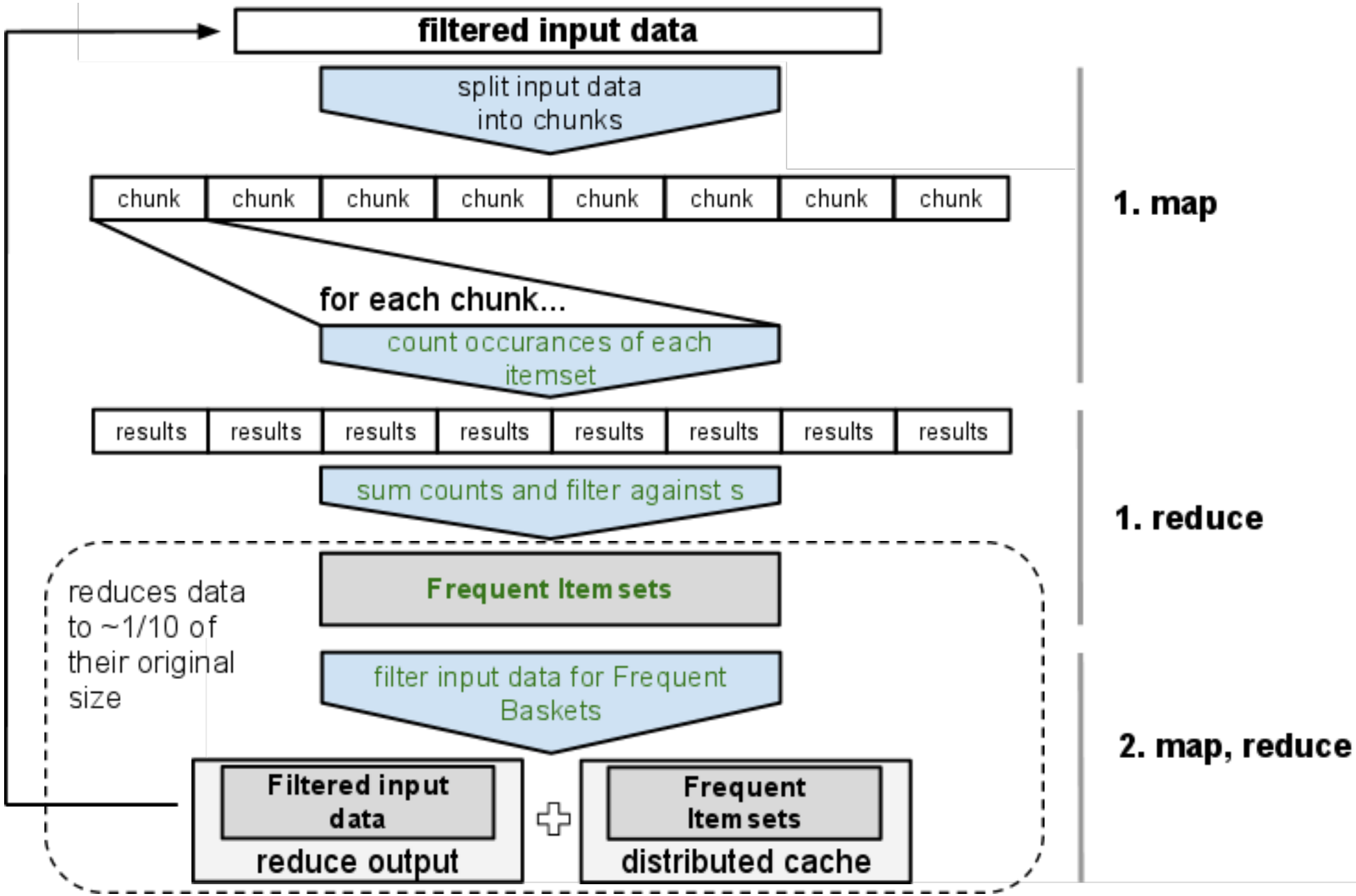
Input size scaling



"Task setup takes a while, so it is best if the maps take at least a minute to execute" [1] We only have ~30 seconds per map task.

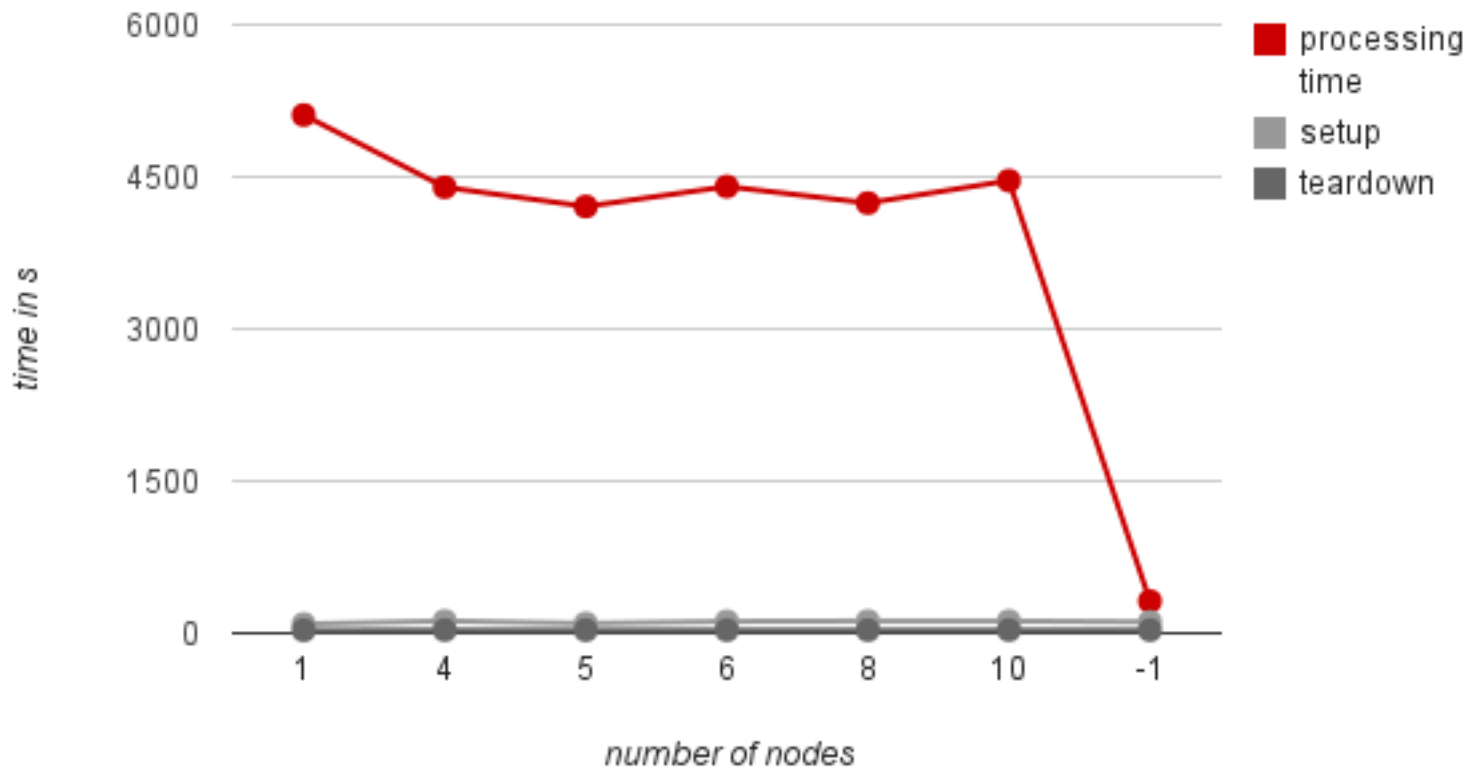
1 http://hadoop.apache.org/common/docs/current/mapred_tutorial.html, Hadoop, 2009

performance influences



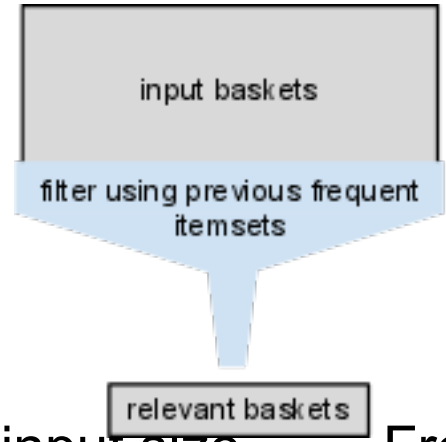
Benchmark Results

Number of reduce tasks

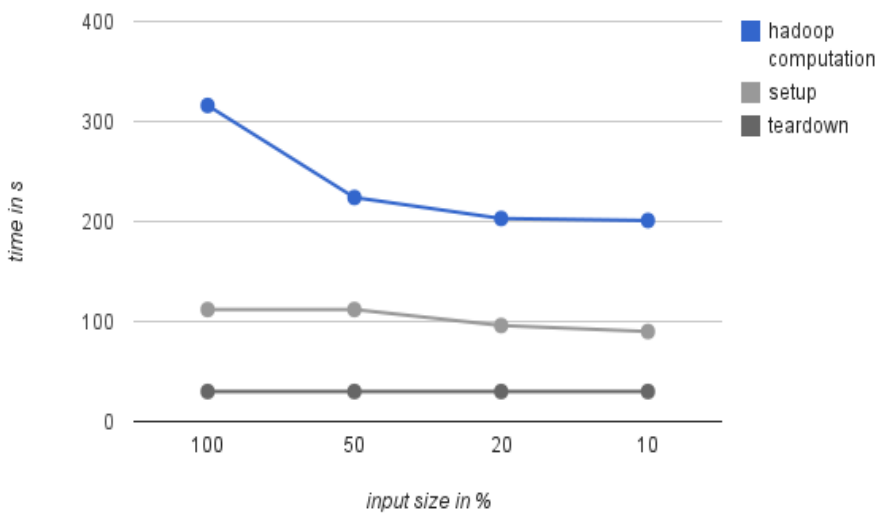


Summary

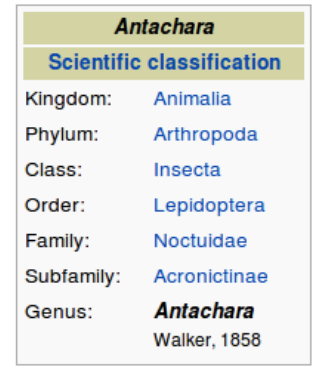
Reducing IO by filtering baskets



Scaling of node number and input size



Frequent Pairs in DBpedia



Next Steps

- implement SON algorithm
  - optional: implement PCY algorithm
- benchmark SON algorithm
  - compare to current results
  -
- find a data set that allows for surprises

PCY algorithm

- during first pass
  - generate & hash pairs: key in hash table
    - increment count of that hash bucket
- during second pass
  - reduce hash table to bitmap "bucket is frequent"
    - 0 if (bucket count < support threshold)
    - 1 else
  - if pair hashes to a non-frequent bucket, it can't be frequent