Towards Parallel Nonmonotonic Reasoning with Billions of Facts

Ilias Tachmazidis [1,2], Grigoris Antoniou [2,3], Giorgos Flouris [2], Spyros Kotoulas [4]

{tahmazid,antoniou,fgeo}@ics.forth.gr , Spyros.Kotoulas@ie.ibm.com

[1] University of Crete
Department of Computer Science
http://www.csd.uoc.gr

[2] Foundation for Research and Technology - Hellas,
Institute of Computer Science
http://www.ics.forth.gr

[3] University of Huddersfield, UK
http://www. www.hud.ac.uk

[4] IBM Research, Ireland
http:// www.ibm.com

Motivation for this Paper

Situation: In the last two of years, there has been significant progress in parallel reasoning, scaling reasoning up to 100 billion triples. Nevertheless, current approaches have been restricted to monotonic reasoning, namely RDFS and OWL-horst, or have not been evaluated for scalability

Problems: Poor quality data (e.g., involving inconsistency or incompleteness), could easily lead to reasoning triviality when considering rules based on monotonic formalisms

Purpose of this paper:
❶ Analyze how parallelization, using the MapReduce framework, can be used to reason with defeasible rules over huge data sets

❷ Evaluate our approach in terms of scalability

Defeasible Logic

Syntax

Facts
e.g. ***bird(eagle)***

Strict Rules
e.g. ***bird(X) → animal(X)***

Defeasible Rules
e.g. ***bird(X) ⇒ flies(X)***

Defeaters
e.g. ***brokenWing(X) ⇝ ¬ flies(X)***

Priority Relation (acyclic relation on the set of rules)
e.g. ***r: bird(X) ⇒ flies(X)***
r': brokenWing(X) ⇒ ¬ flies(X)
r' > r

Advantages

- Suitable for encoding commonsense knowledge and reasoning
- Avoid triviality of inference due to low-quality data
- Low complexity
  - The consequences of a defeasible theory D can be computed in O(N) time, where N is the number of symbols in D

MapReduce Framework

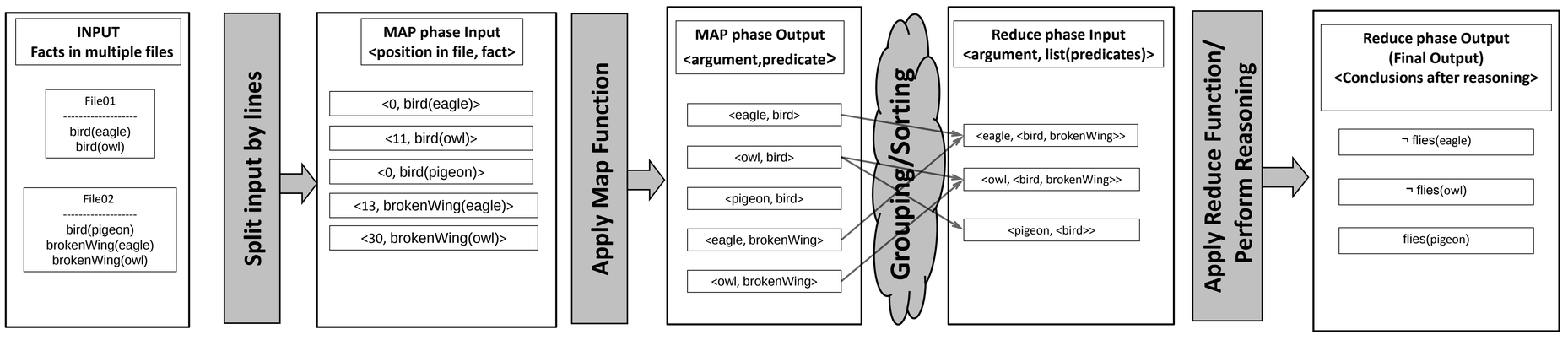
MapReduce Framework

- MapReduce is a software framework introduced by Google in 2004
- Deals with:
  - very large amounts of data (many terabytes)
  - process data fairly quickly
  - use very large numbers of commodity machines (thousands)
- Operates exclusively on <key, value> pairs
- Want an infrastructure that takes care of management tasks
  - distribution of data
  - management of fault tolerance
  - collecting results
- For a specific problem
  - developer writes a few routines
  - routines plug into the general interface

Hadoop

- Hadoop is used heavily, among others, by IBM, Yahoo!, Facebook and Twitter
- Hadoop includes
  - a distributed file system, HDFS
  - a MapReduce framework
  - a web monitoring interface

Algorithm Description



MapReduce Algorithm

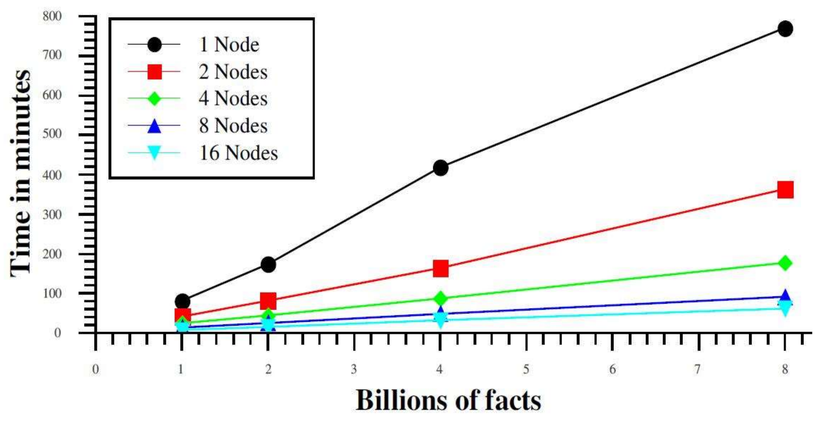
```
map(Long key, String value) :
  // key: position in document (irrelevant)
  // value: document line (a fact)
  argumentValue = extractArgumentValue(value);
  predicate = extractPredicate(value);
```

```
reduce(String key, Iterator values) :
  // key: argument value
  // values : list of predicates (facts)
  List listOfFacts;
  Reasoner reasoner = Reasoner.getCopy();
  for each v in values
     listOfFacts.add(v);
  reasoner.Reason(listOfFacts);
  Emit(key , reasoner.getResults());
```

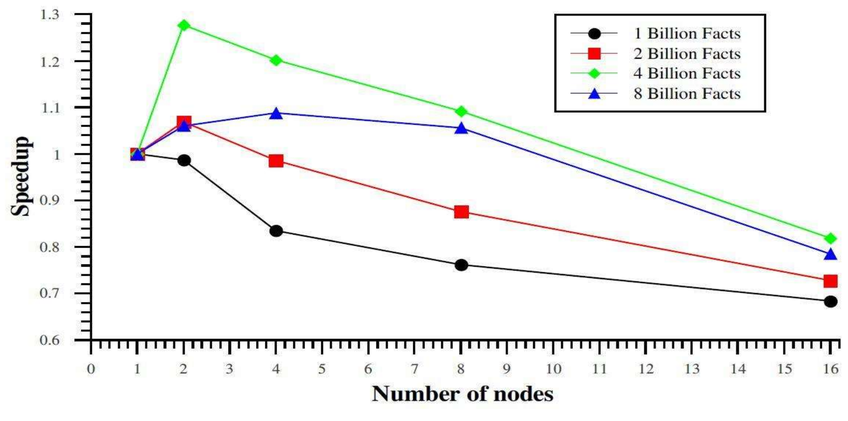
Rule set

r1 : bird(X) → animal(X),
r2 : bird(X) ⇒ flies(X),
r3 : brokenWing(X) ⇒ ¬ flies(X),
r3 > r2

Experimental Results



Runtime in minutes as a function of dataset size, for various numbers of nodes



Scaled speedup for various dataset sizes

13th International Conference on Principles of Knowledge Representation and Reasoning (KR-2012)

This work was partially supported by the PlanetData NoE (FP7:ICT-2009.3.4, #257641)