

# Parallel forward reasoning

Spyros Kotoulas  
VU Amsterdam, LarKC

with

Jacopo Urbani, Jason Maassen,  
Niels Drost, Frank Seinstra, Frank van  
Harmelen, Henri Bal



# Outline

- Focus
- Marvin (BTC '08, JWS, WWW '10)
- WebPIE (ISWC '09, ESWC '10)
- The secret!



# Focus

- RDFS/OWL-horst materialisation on clusters
- Highly scalable
  - In terms of processing cores, meaning that algorithms should be highly parallelisable
  - In terms of input size, meaning we can not assume that we have enough memory
- All machines controlled by the same authority



# Machinery

- **Run on a cluster**
  - Up to 64 quad-core nodes
  - No exotic networks (Gbitethernet)
  - No exotic HDDs (One standard HDD per node)
  - Limited memory (4GB/node)

# Marvin

- n Designed to compute the materialization on (almost) unstructured networks
- n P2P network where nodes compute the materialization locally and exchange data
  - n Data is exchanged in a "smart" way
  - n Approximate reasoning    full closure at finite time
- n It can be used not only for reasoning, but also for other tasks!

# Marvin

- **Divide – Conquer – Swap**
  1. Split the input across peers
  2. Calculate the closure
  3. If you want more completeness, Goto 1.
- Can calculate the RDFS closure of 200M triples in 7 minutes.



# WebPIE

- n Uses MapReduce to distribute computation
- n Implements the RDFS and the OWL terminology rule sets
- n Full forward inference
- n SameAs statements are not materialised



## WebPIE - RDFS reasoning

- Schema triples are replicated to all nodes
- Instance triples are “streamed”
- Main challenges were:
  - Reduce duplicate derivations
  - Reduce number of iterations



# WebPIE - OWL Horst Reasoning

## Additional challenge:

Replicating schema triples is not enough. Rules fire on multiple instance triples.

## Solution:

Develop rule-specific optimisations for transitivity, sameAs, and someValuesFrom/allValuesFrom

# WebPIE - Results

Nodes	Runtime (hours)	Speedup
8	44.4	1.00
16	22.3	1.99
32	10.6	4.17
64	5.0	8.78

Dataset	Input (Triples)	Output (Triples)	Runtime (h:m:s)
<i>RDFS</i>			
Falcon	32.5M	863.7M	0:04:19
Swoogle	78.8M	1.50B	0:07:15
BigWeb	864.8M	30.0B	0:56:57
<i>OWL-horst</i>			
LDSR	862M	935M	3:31:00
Uniprot	1.5B	2.0B	6:05:49
LUBM	1.1B	496M	0:36:36

(a)

Input (Triples)	Output (Triples)	Runtime (h:m:s)	Throughput (Kt/sec)
1.07B	0.50B	0:36:36	455.2
2.14B	0.99B	0:59:40	602.6
10.71B	4.97B	4:03:37	684.6
102.50B	47.56B	45:46:12	606.8

(b)



# The secret!

- Do not maintain full indexes, maintain only the indexes required for reasoning
- Partitioning: all joins are performed between one triple belonging to the partition and one or more triples in memory

Even if it means that we need multiple passes  
Allows massive distribution using M/R



## Limitations / Future (or current) work

- Querying
  - Working on HBase
- “Black hole” model, we do not support distributed data
- No quality control
- Adding data is inefficient