SAND2011-3429C

ESWC 2011: High-performance Computing Applied to Semantic Databases

Eric Goodman (Sandia National Laboratories)
Edward Jimenez (Sandia National Laboratories)
David Mizell (Cray Inc.)
Sinan al-Saffar (Pacific Northwest National Laboratories)
Bob Adolf (Pacific Northwest National Laboratories)
David Haglin (Pacific Northwest National Laboratories)

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Overview

One Usage Scenario

- Data scattered on the web
- Interoperability is essential
- Performance not vital

Our Usage Scenario

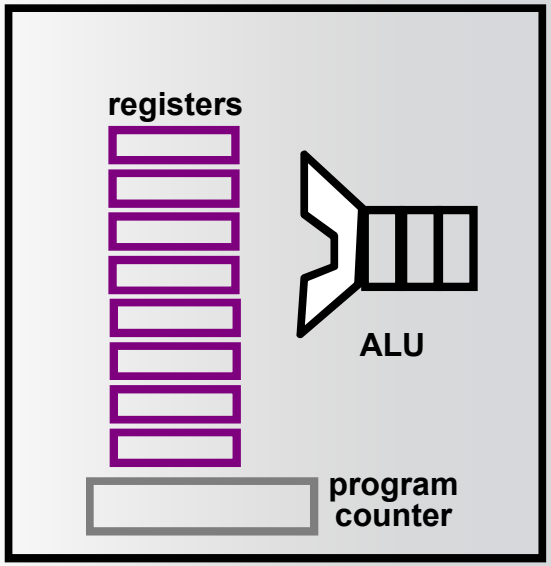
- Put data all on one system
- Data is graph-like, doesn't fit relational model
- Complex Queries
- Performance is imperative

Approaches

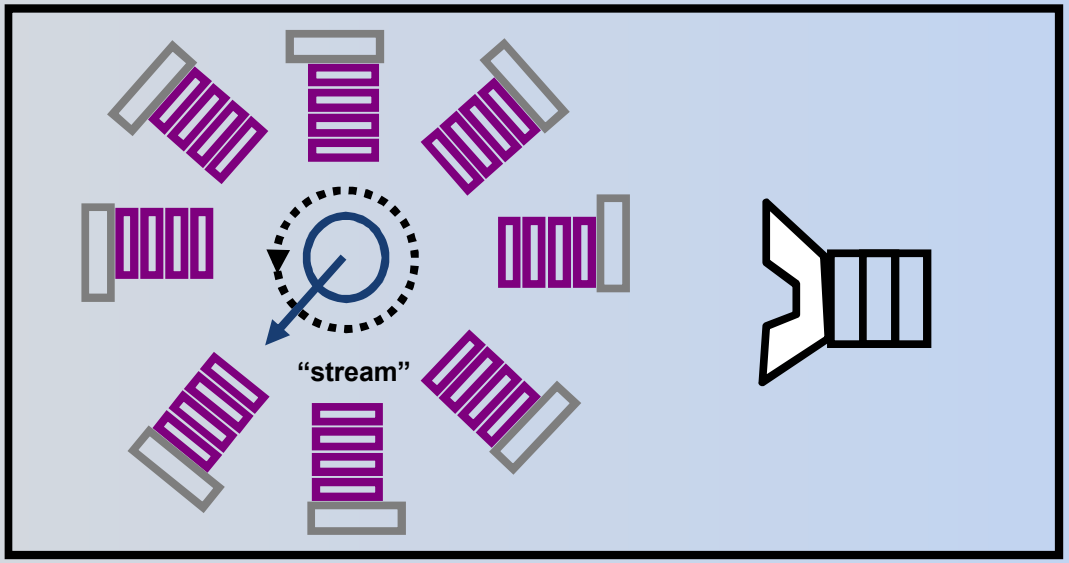
- **Common Thread to Previous Approaches**
  - Commodity Hardware
  - Distributed Memory
  - MapReduce
- **Our Claim**
  - For performance on Semantic Web applications, more specialized hardware/software is needed
    - Shared memory
      - 1-32 TB range sufficient for many data sets
    - Latency-tolerant processor or algorithm design

Multithreading

- Many threads per processor core; small thread state
- Thread-level context switch at every instruction cycle



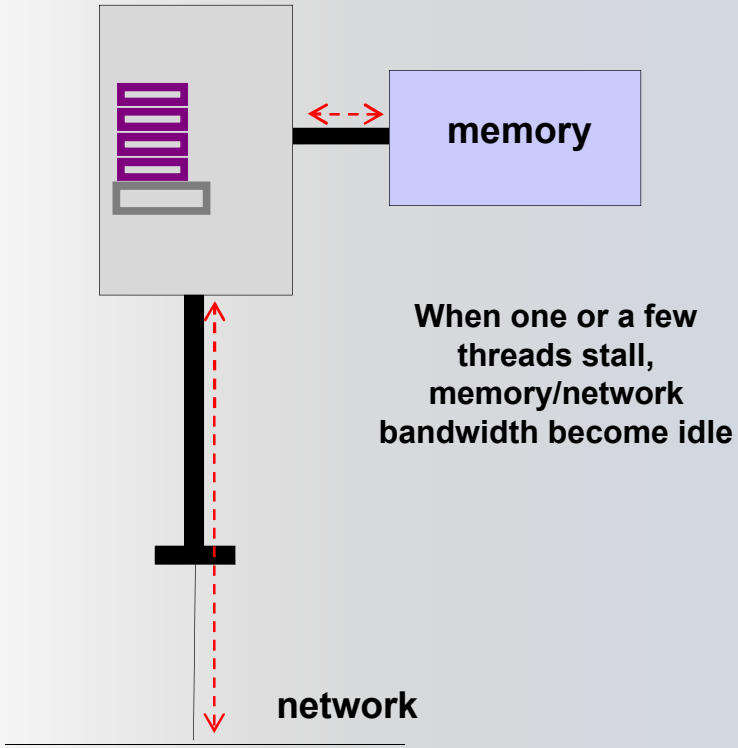
conventional processor



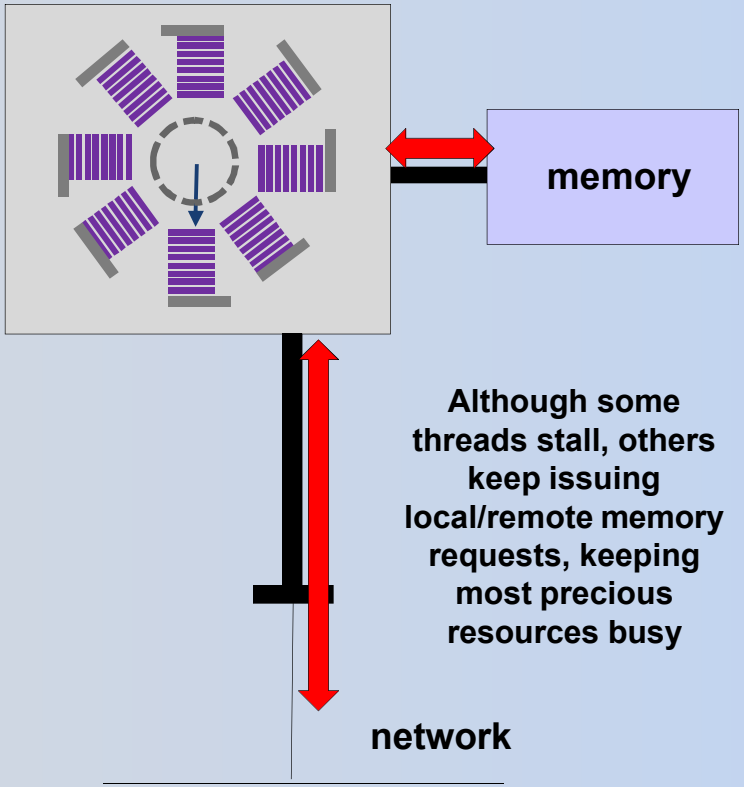
multithreaded processor

Keeping the Bottlenecks Saturated

- Conventional processor

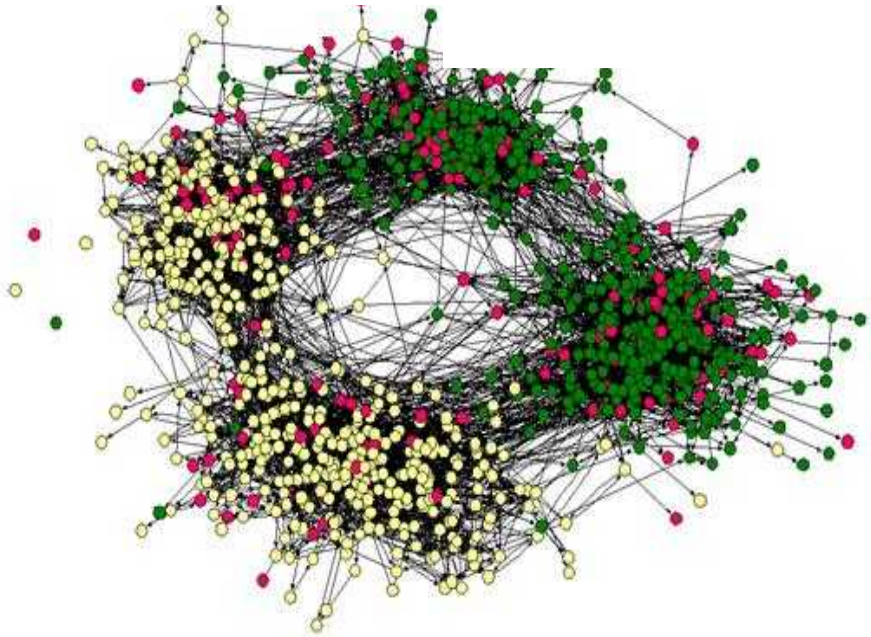


- Multithreaded processor



XMT's Ideal Application Characteristics

- Huge data structures
  - Too large for one node of conventional system
- No locality of reference
  - No way to partition data structure so that most references are local
- But lots of parallelism
- i.e., great big ugly graphs



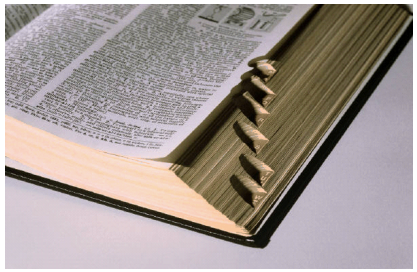
Summary of Results

- **Dictionary Encoding**
  - 2.4-3.3 times faster
- **RDFS Closure**
  - 6.0-9.0 times faster
- **Query**
  - 2.1- 28 times faster

Dictionary Encoding

<http://www.Department12.University0.edu/GraduateStudent9>
<http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#advisor>
<http://www.Department12.University0.edu/FullProfessor6> .

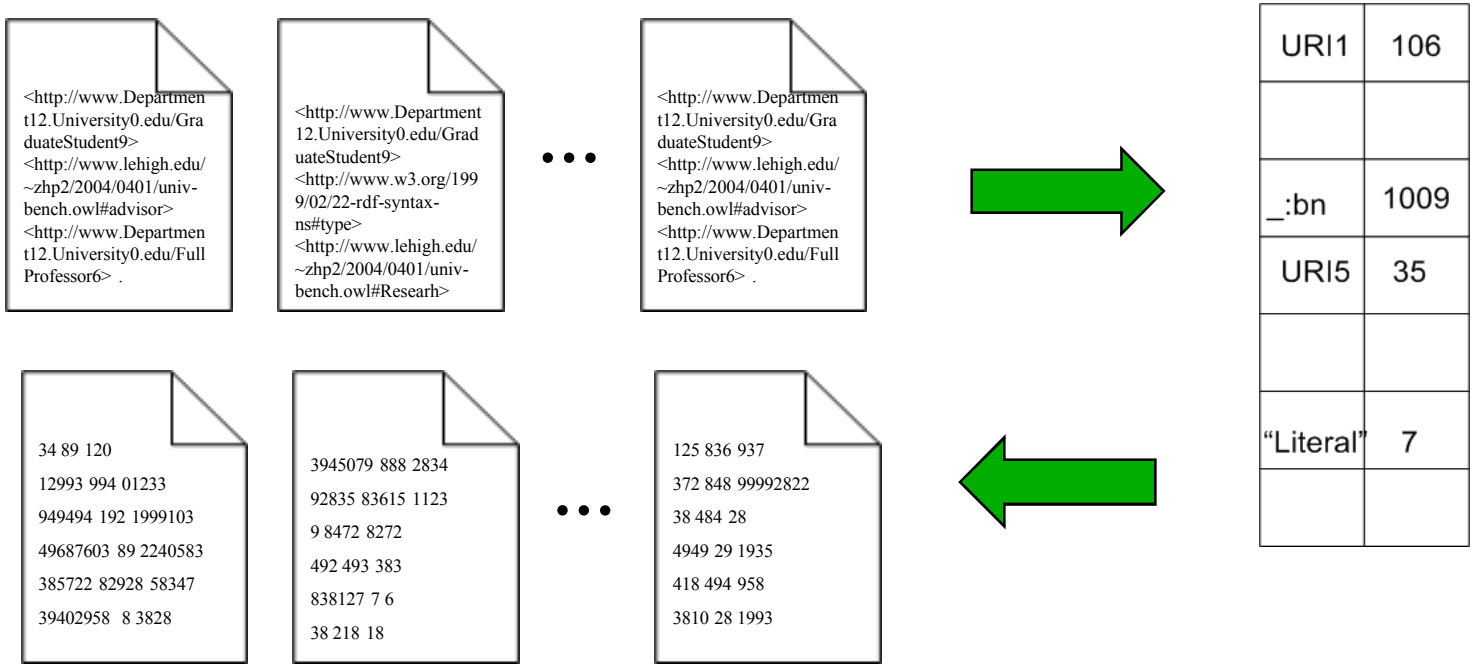
<http://www.Department12.University0.edu/GraduateStudent9>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#ResearchAssistant> .



1 2 3

1 4 5

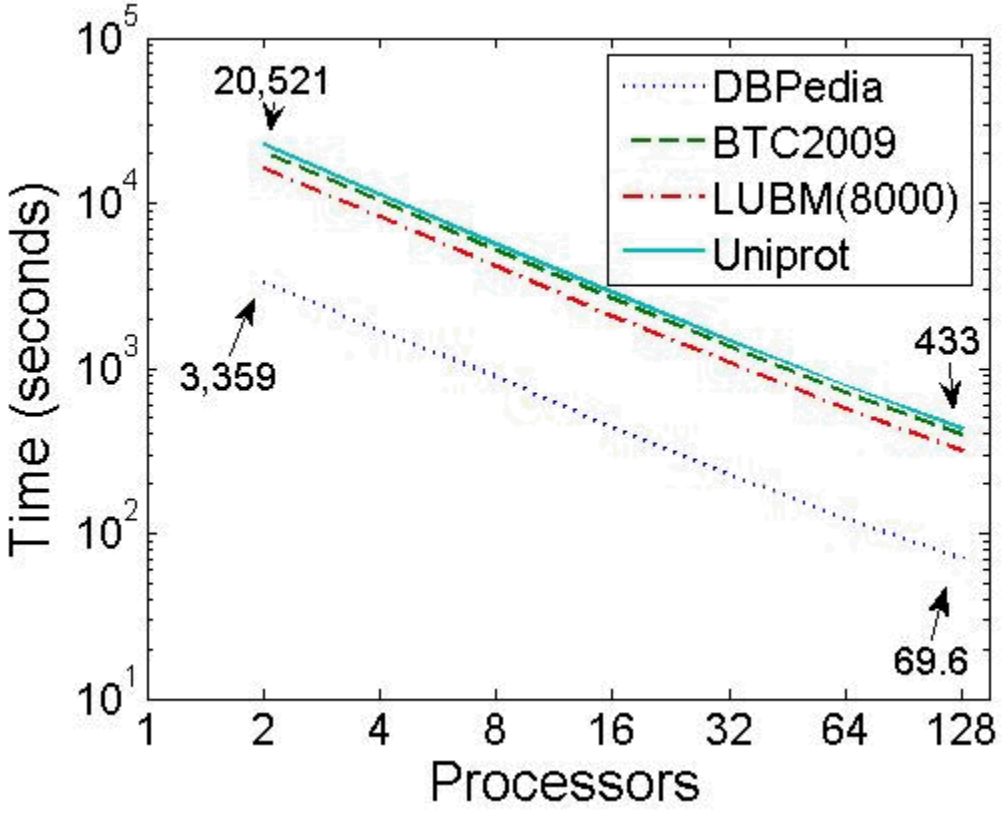
Dictionary Encoding on the XMT



Dictionary Encoding: Algorithm

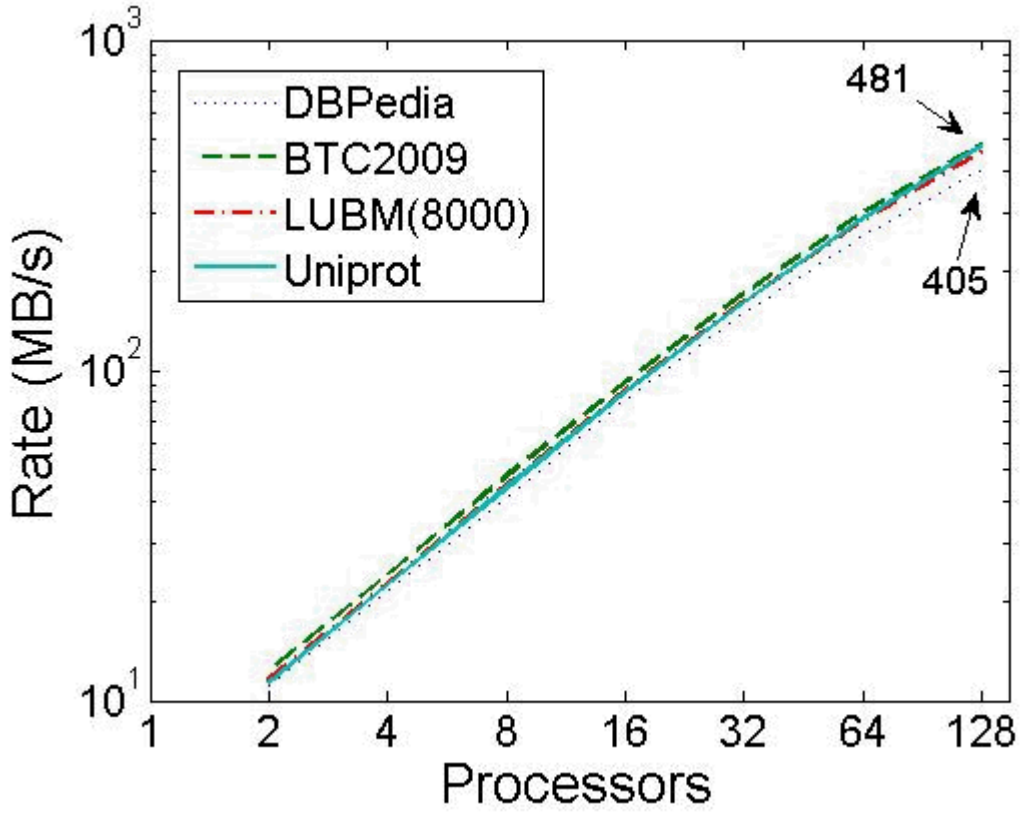
1. Tokenize ntriple/nquad formatted files
2. Create list of new elements by referencing against existing hash table
3. Insert list of new elements into temporary hash table with a key of 1
4. Assign contiguous set of integer ids to new keys in range *[current_max + 1, current_max + num_new_keys]*
5. Add new keys to end of consolidated character array
6. Resize forward and reverse maps if necessary based on current capacity and *num_new_keys + num_current_keys*
7. Go through tokenized file, and assign integer ids and add new integer ids to forward and reverse maps

Dictionary Encoding: Total Time



Sandia National Laboratories

Dictionary Encoding: Rates



Dictionary Encoding: Comparison

Data Set	Raw Size(GB)	Compression Ratio	Size Dictionary on Disk (GB)	Size Dictionary in Memory (GB)
BTC2009	247	4.34	31.1	44.8
DBPedia	36.5	3.2	5.65	9.15
LUBM	185	4.37	17.7	31.7
Uniprot	250	3.94	19.6	33.2

Data Set	MapReduce Rate (MB/s)	XMT Rate (MB/s)	Improvement
DBPedia	36.4	120	3.29
LUBM	67.1	162	2.41
Uniprot	48.8	161	3.3

Comparison to Urbani et al. “Massive Semantic Web data compression with MapReduce.” 2010

RDF Schema (RDFS) Closure

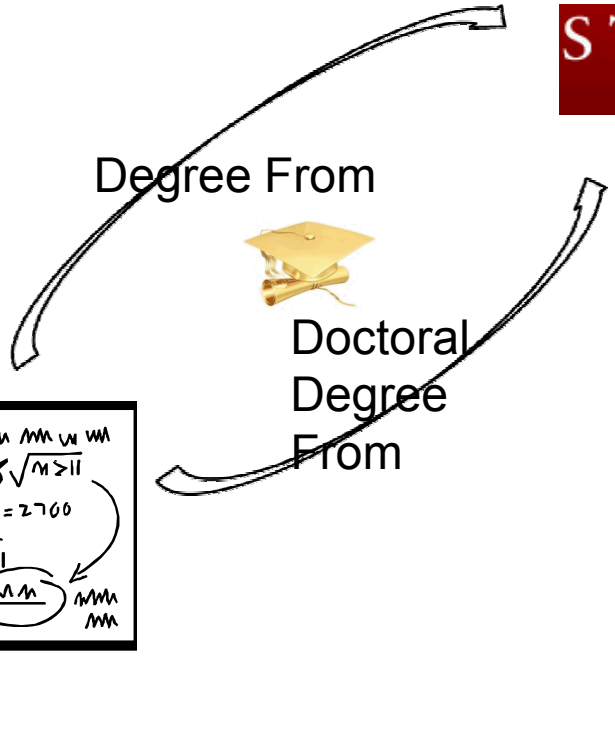
- **RDFS allows definition of basic ontological elements**
- **Inference is the application of ontological rules to data**
- **Inference can be done at query execution time, or as a preprocessing step**
  - We materialize the inferred triples in memory
- **We implement closure on only the "interesting subset," i.e. rules requiring two antecedents:**
  - Subclass inheritance and transitivity
  - Subproperty inheritance and transitivity
  - Domain
  - Range

Example: Subproperty Inheritance

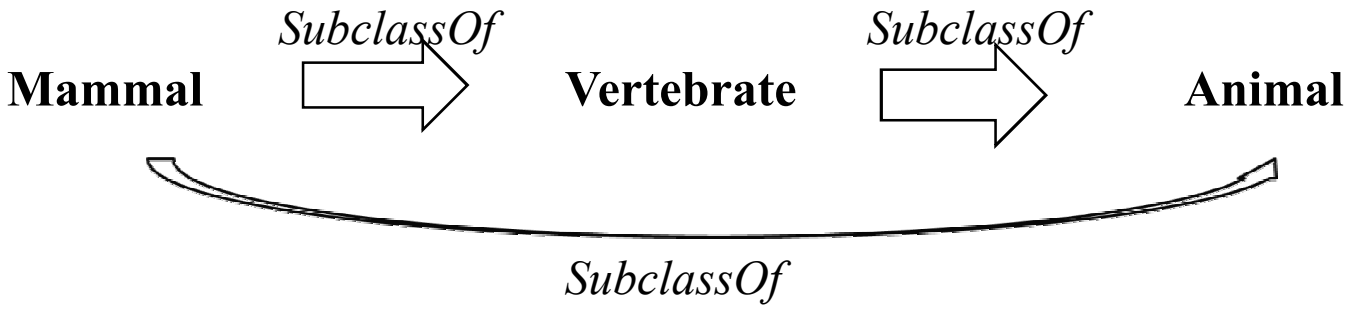
Doctoral Degree From

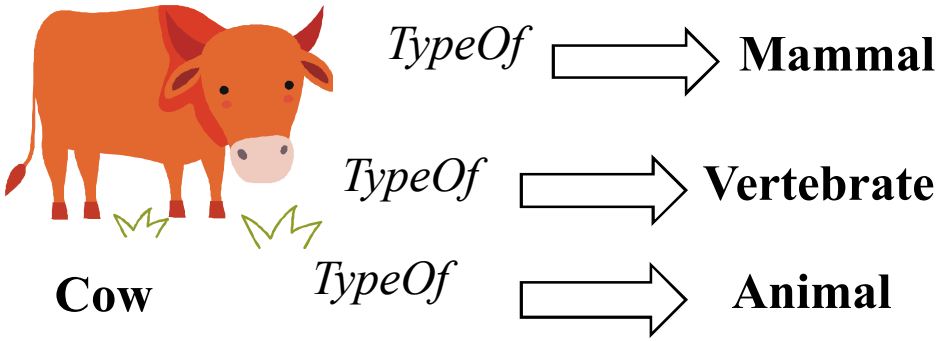
Subproperty Of

Degree From



Example: Subclass Transitivity and Inheritance

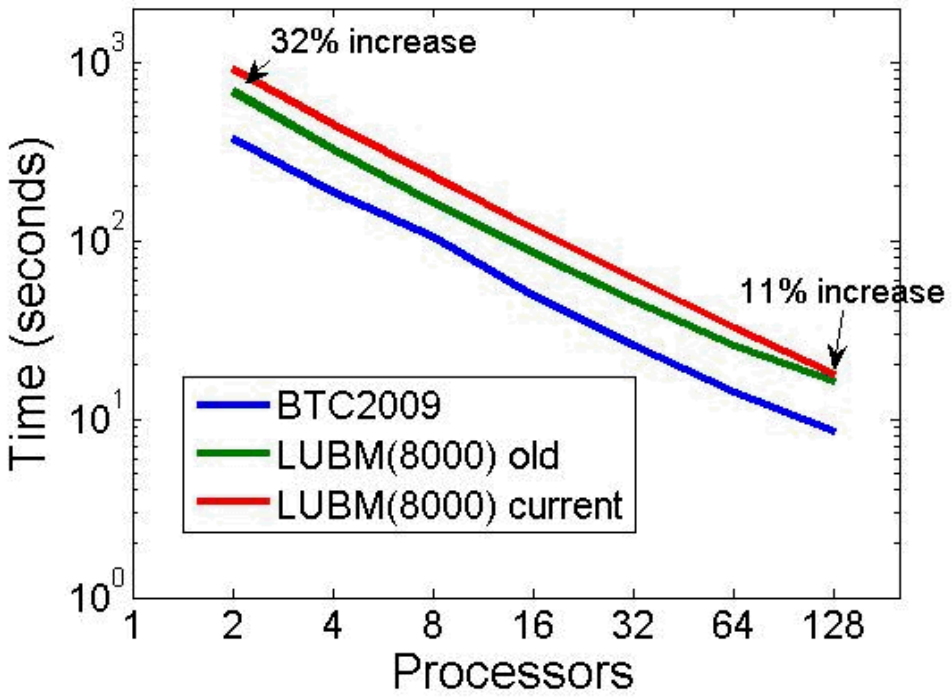




The RDFS Closure Algorithm

- **Read arbitrary triple store in integer format from disk**
- **Create and populate ontology data structures**
  - Create and populate multimaps
  - Apply transitivity rules *rdfs5* (subproperty) and *rdfs11* (subclass)
  - Replicate multimap data structures
- **Insert original triples into hash table**
- **Add matching triples to queues**
- ***rdfs7* Subproperty Inheritance**
  - Add matching triples to domain and range queues
- ***rdfs2* Domain**
  - Add matching triples to subclass queue
- ***rdfs3* Range**
  - Add matching triples to subclass queue
- ***rdfs9* Subclass Inheritance**

RDFS Closure Results on LUBM



Improvement Factor

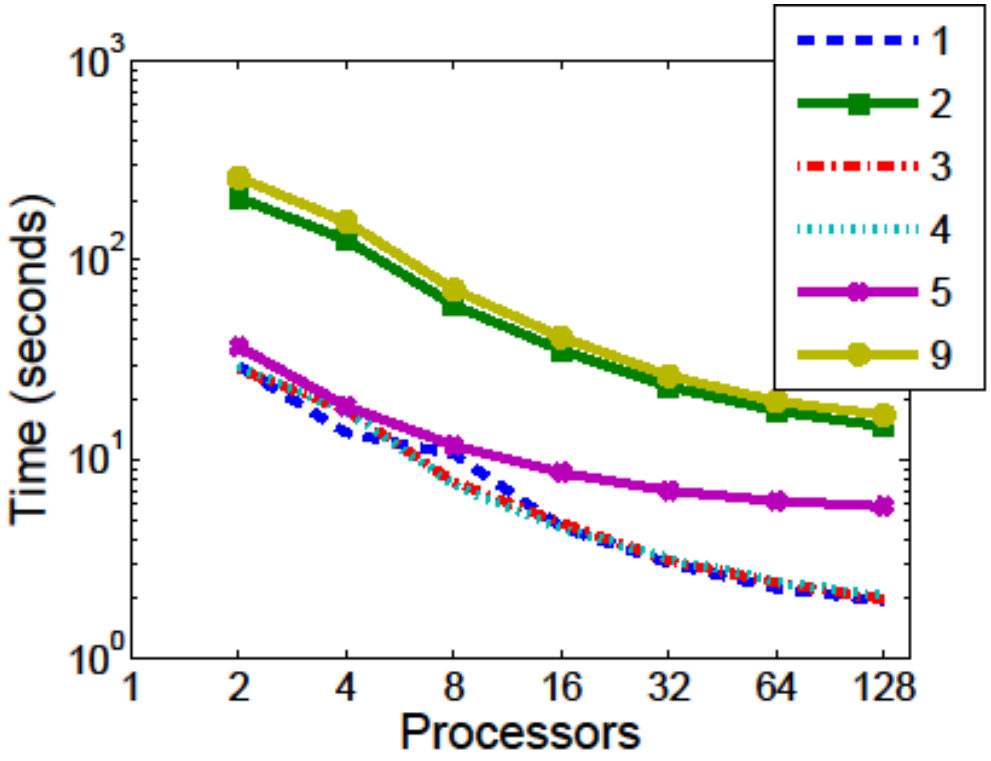
Approach	With I/O	Without I/O
MPI[1]	6.0	6.8
WebPIE[2]	9.0	10.6

[1]Weaver and Hendler. "Parallel Materialization of the Finite RDFS Closure for Hundreds of Millions of Triples." ISWC 2009

[2]Urbani et al. "OWL Reasoning with WebPIE Calculating the Closure of 100 Billion Triples." ESWC 2010

Sprinkle SPARQL

- **Sprinkle SPARQL presented in ESWC paper**
- **Paucity of scalability results in literature**
  - 10 nodes running MapReduce
  - 1 node running BigOWLIM



Query	MapReduce	BigOWLIM
2	13.6	2.12
9	28.0	2.82

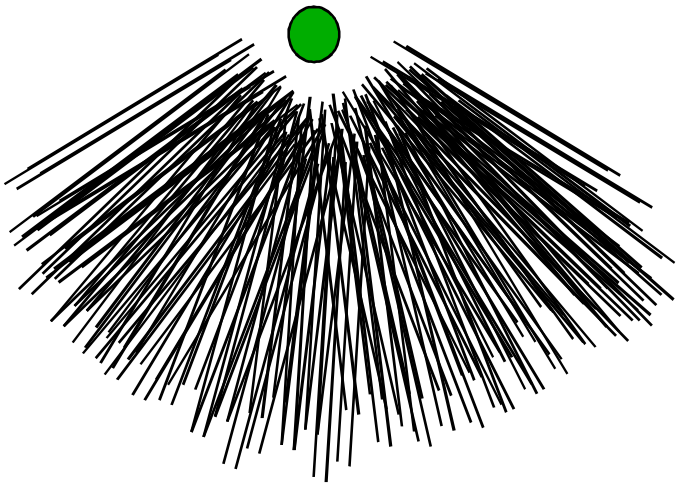
Note: MapReduce method did not operate on inferred set. They hand-encoded expanded queries to catch the possibilities.

Sandia National Laboratories

LUBM Query 1

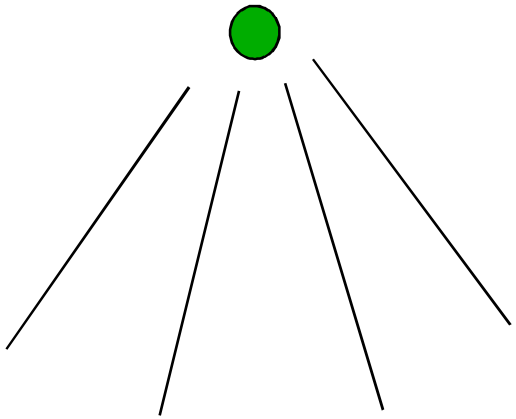
```
SELECT ?X
  WHERE
    {?X rdf:type ub:GraduateStudent}
    {?X ub:takesCourse
     http:www.Department0.University0.edu/GraduateCourse0}
```

All the Graduate Students



20,157,119 matches

All the Students that took a particular course



4 matches

Sandia National Laboratories

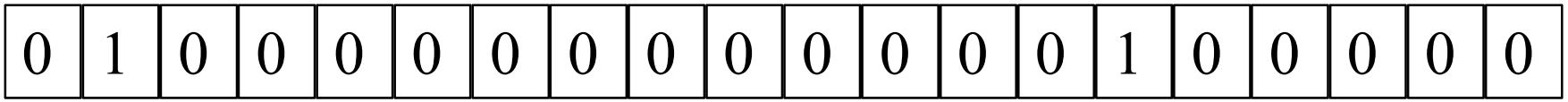
Sprinkle phase

- **Create an array the same size as the order of the graph for each variable in each BGP**
- **Process each BGP**
  - If node fulfills constraint of BGP, increment counter in associated array for the variable
- **The point: Constrain the problem before we start joining**

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sprinkle phase

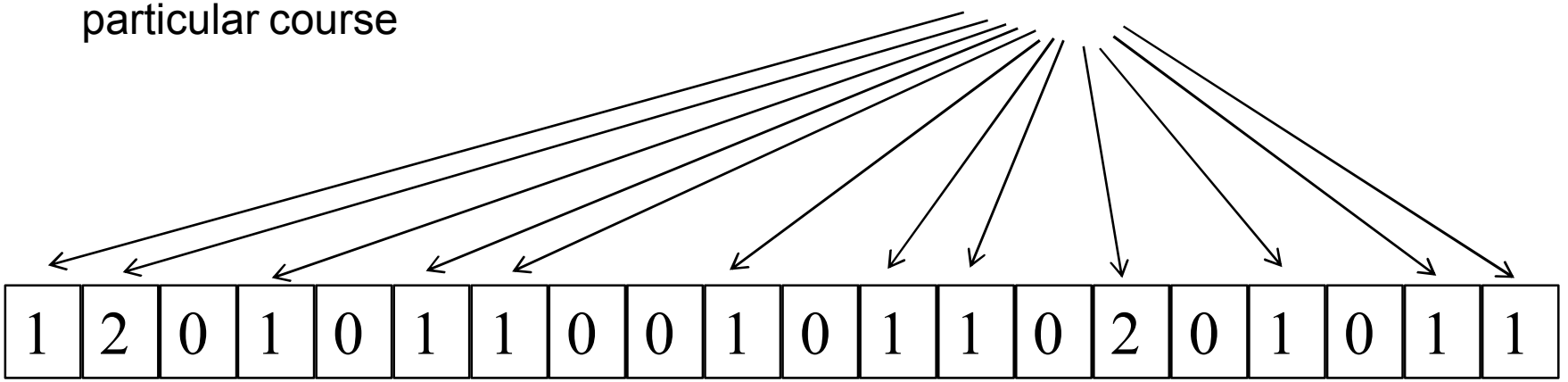
All the Students that took a particular course



Sprinkle phase

All the Students that took a particular course

All the Graduate Students



Future Work / Conclusions

- **XMT shows promise for performance driven Semantic Web applications**
- **Need better benchmarks, especially for querying**
- **Investigate other shared-memory architectures**
  - See if we can obtain software-level latency tolerance
- **Sprinkle SPARQL**
  - Extensions to handle variable in predicate position
  - Perform simple queries more efficiently
  - Formal analysis
- **Expand inference work to OWL Horst**