IPADS, SJTU, China http://ipads.se.sjtu.edu.cn/projects/wukong

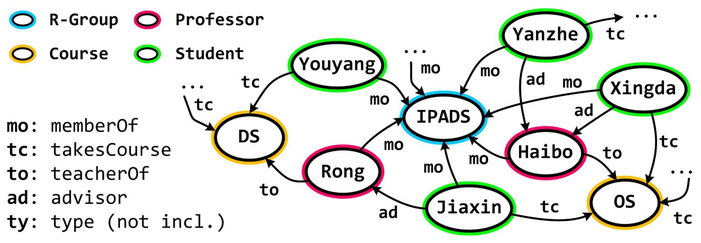
TALK @ Thu, Nov 3
9:00 SESSION 1
Graph Processing & Machine Learning

Fast and Concurrent RDF Queries with RDMA-based Distributed Graph Exploration

Jiaxin Shi, Youyang Yao, Rong Chen, Haibo Chen and Feifei Li

Background

Social Networks, IoT and Business Intelligence apps model data as RDF Graphs and query with SPARQL query language



Existing Solutions:

Triple join
- costly join ops
- redundant im. results

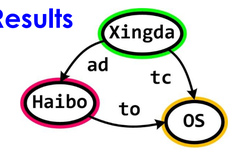
Graph exploration
- one-step pruning
- final centralized join

Sample RDF & SPARQL

```
SELECT ?X ?Y ?Z WHERE {
  ?X teacherof ?Y .
  ?Z takecourse ?Y .
  ?Z advisor ?X .
}
```

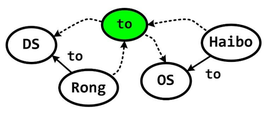
SPARQL Query

Results

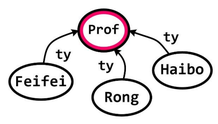


Data Model

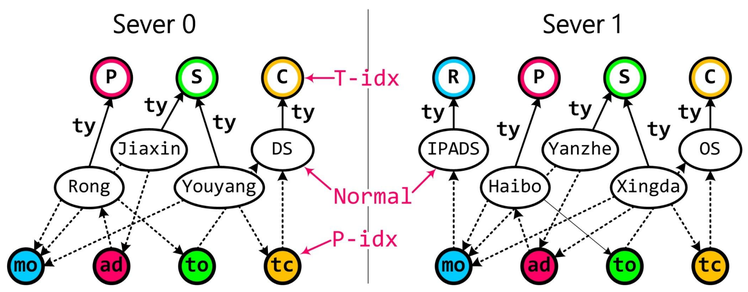
Graph Model and Indexes & Differentiated Graph Partitioning
- Normal Vertex: randomly assign vertex with all edges (edge-cut)
- (Type & Predicate) Index Vertex: split and replicated (vertex-cut)



(a) Predicate Index

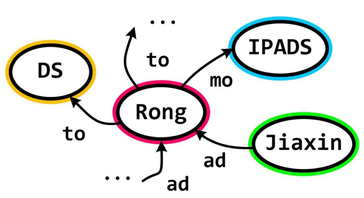


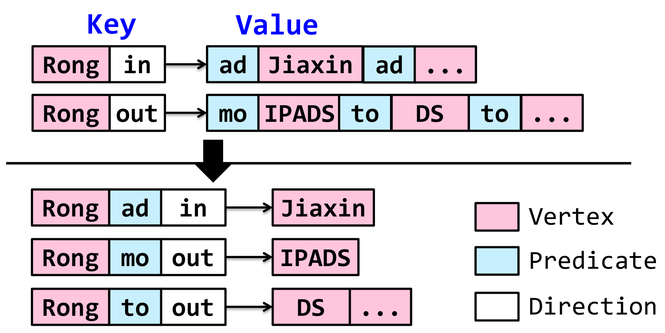
(b) Type Index



RDMA-friendly Predicate-based Store
- Derived from DrTM-KV
- Separate a vertex into multiple KV pairs according to predicate

```
SELECT ?X WHERE {
 Rong teacherof ?X .
}
```





Evaluation

Wukong outperforms state-of-the-art systems for both latency and throughput, usually at the scale of orders of magnitude

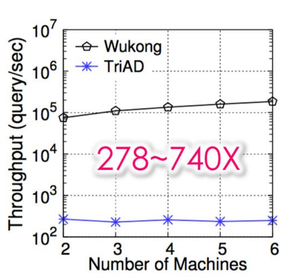
Setting
- A 6-node cluster (each: 20 cores, 64GB DRAM, 2 x IB)
- Benchmark: LUBM, WSDTS, DBPSB, YAGO2
- Baseline: TriAD, Trinity.RDF, RDF-3X, BitMat, etc.

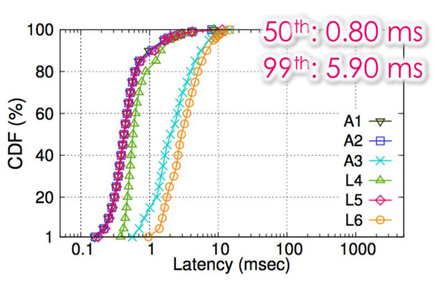
LUBM 10240	Wukong	TriAD	TriAD-SG (200K)	Trinity .RDF	SHARD
L1	516	2,110	1,422	12,648	19.7E6
L2	78	512	695	6,081	4.4E6
L3	203	1,252	1,225	8,735	12.9E6
L4	0.41	3.4	3.9	5	10.6E6
L5	0.17	3.1	4.5	4	4.2E6
L6	0.89	63	4.6	9	8.7E6
L7	464	10,055	11,572	31,214	12.0E6
Geo. M	16	190	141	450	9.1E6

Factor Analysis
- BASE: Trinity.RDF-like
- +RDMA: RDMA comm.
- +FHP: enable Full-history
- +IDX: Index-vertex
- +PBS: predicate-based KV
- +DYN: mode switch

LUBM 10240	BASE	+RDMA	+FHP	+IDX	+PBS	+DYN
L1	9,766	9,705	888	853	814	516
L2	2,272	2,161	1,559	84	79	78
L3	421	404	404	205	203	203
L4	1.49	0.79	0.78	0.78	0.56	0.41
L5	1.00	0.39	0.39	0.39	0.31	0.17
L6	3.84	1.40	1.37	1.37	1.17	0.89
L7	2,176	2,041	657	494	466	464
Geo. M	102.3	69.1	39.6	22.6	19.9	15.7

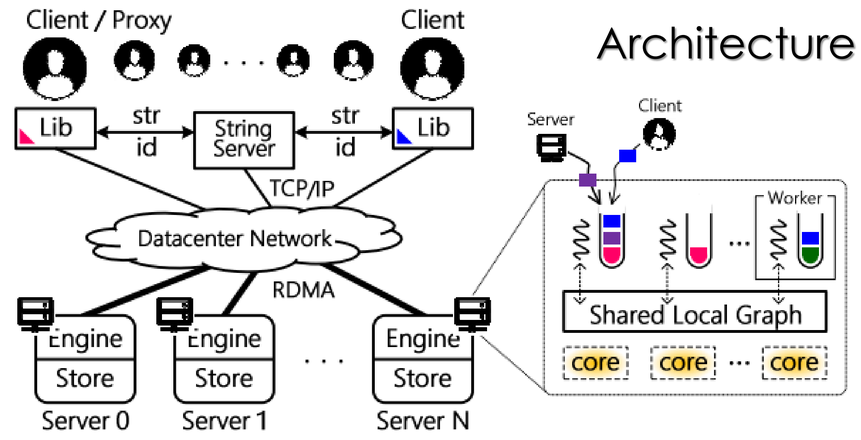
A study on concurrent queries processing
- Mixture workload
- 269K queries/sec
- 99th latency: 5.9ms





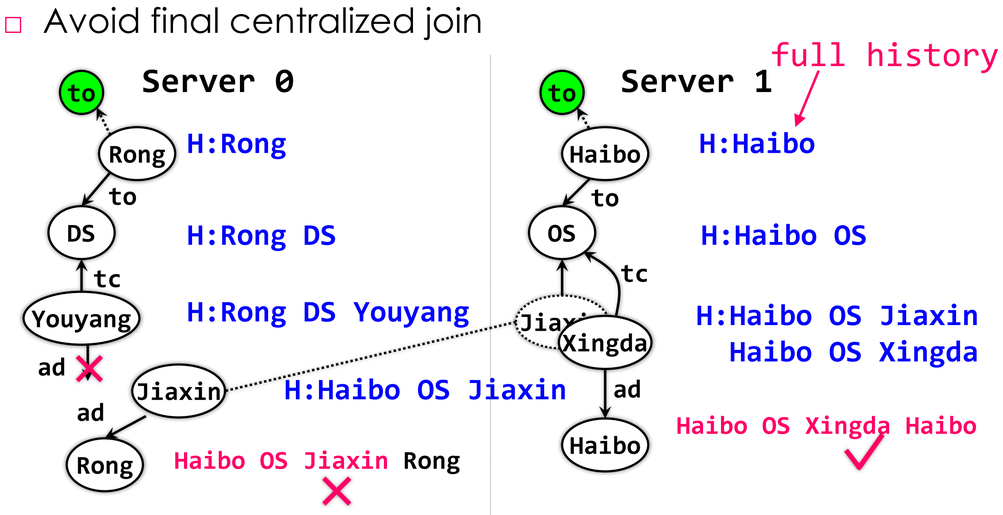
Overview

Wukong: a distributed in-memory RDF store that leverages RDMA-based graph exploration to support fast and concurrent SPARQL queries
- Graph Store: index vertex, differentiate partitioning, predicate-based KV store
- Query Engine: full-history pruning, in-place/ fork-join execution, work-oblige scheduling
- Communication: one-sided RDMA ops

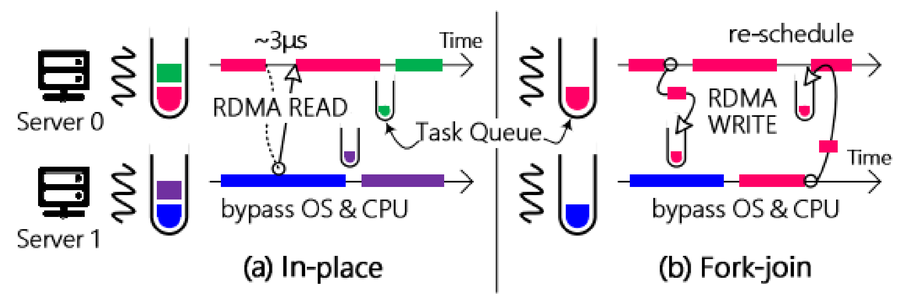


Query Processing

Full-history Pruning
- Observation: the latency of RDMA is relatively insensitive to payload sizes *(e.g., 8bytes: 1.56us vs. 2KB: 2.25us)*
- Prune non-matching results early
- Avoid final centralized join



Dynamic Execution Mode Switch
- In-place mode: migrate data (ultra-low latency)
- Fork-join mode: migrate execution (parallelism)
- Make decisions in runtime according to #RDMA-ops



Worker-obliger Algorithm
- Latency of a query may vary significantly (e.g., 3000X)
- Provide a latency-centric work stealing
- Oblige queries in straggling workers

```
1 int next = 1

OBLIGER()
2  s = state[(tid+next)%N]
3  q = NULL
4  s.lock()
5  if (s.cur == tid //reentry
6   || s.end < now)
7    s.cur = tid;
8    s.end = now + T
9    next++
10   q = s.dequeue()
11 s.unlock()
12 return q
```

```
SELF()
13  s = state[tid]
14  s.lock()
15  s.cur = tid
16  s.end = now + T
17  next = 1
18  q = s.dequeue()
19  s.unlock()
20  return q

NEXT_QUERY()
21  if (q = OBLIGER())
22    return q
23  return SELF()
```