Trinity

A Distributed Graph Engine on a Memory Cloud

UNIVERSITY OF CAMBRIDGE

Motivation

Polarized view on previous approaches:

	Graph Database	Query Processing	Graph Analytics	Scalability
Neo4J	Yes	Yes	No	No
HyperGraph DB	Yes	Yes	No	No
MapReduce	No	No	Yes	Yes
Pregel	No	No	Yes	Yes

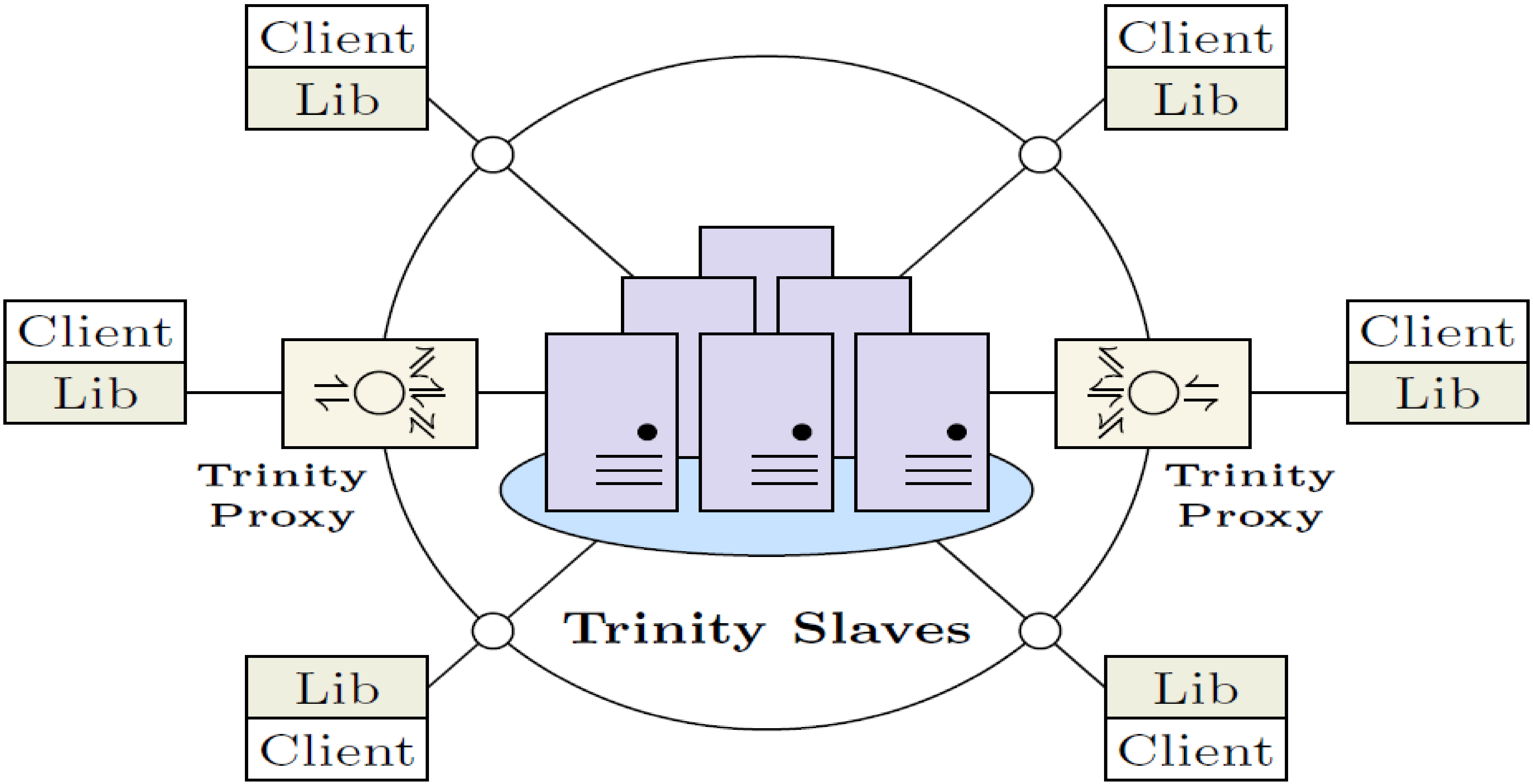
UNIVERSITY OF CAMBRIDGE

Concept

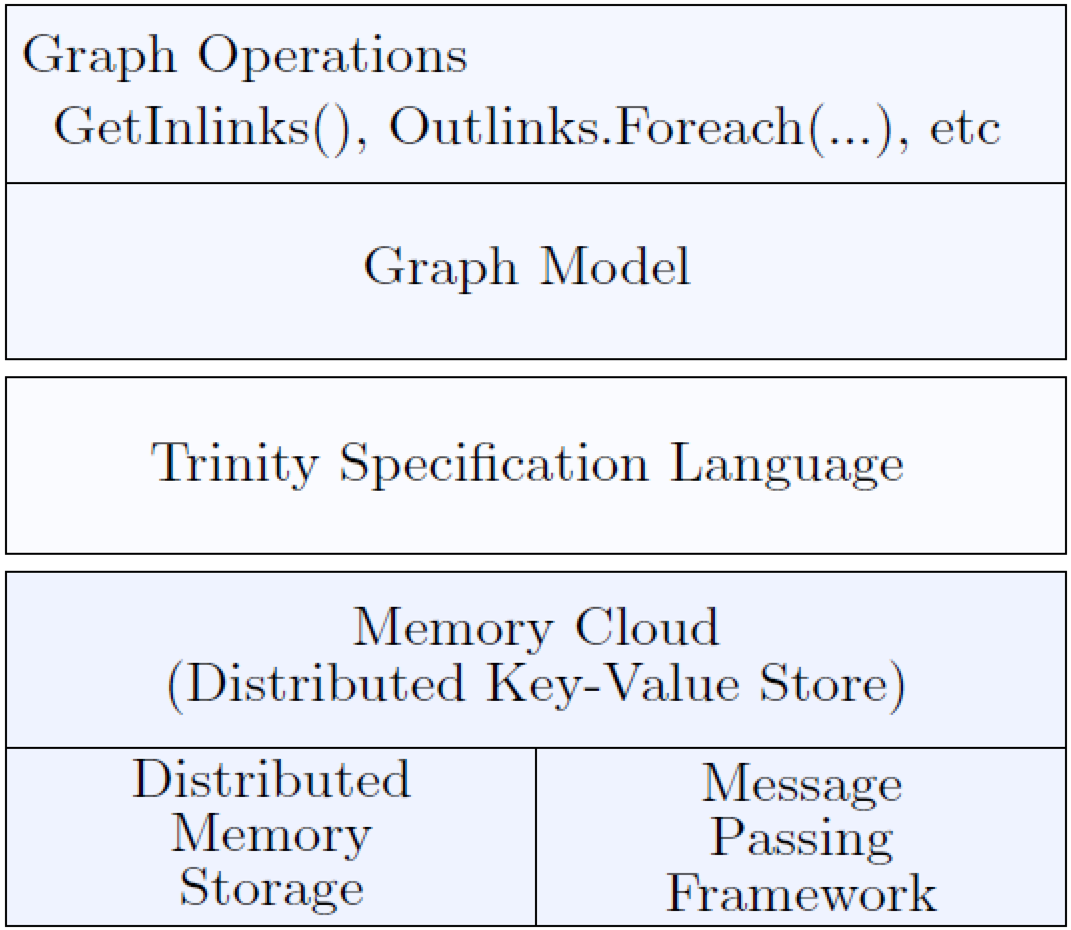
- Globally shared memory
- Distributed key/value store
- Random access abstraction for online queries (BFS, graph-matching)
- Scalabililty through partitioning
- Restrictions on message passing for improved performance

UNIVERSITY OF CAMBRIDGE

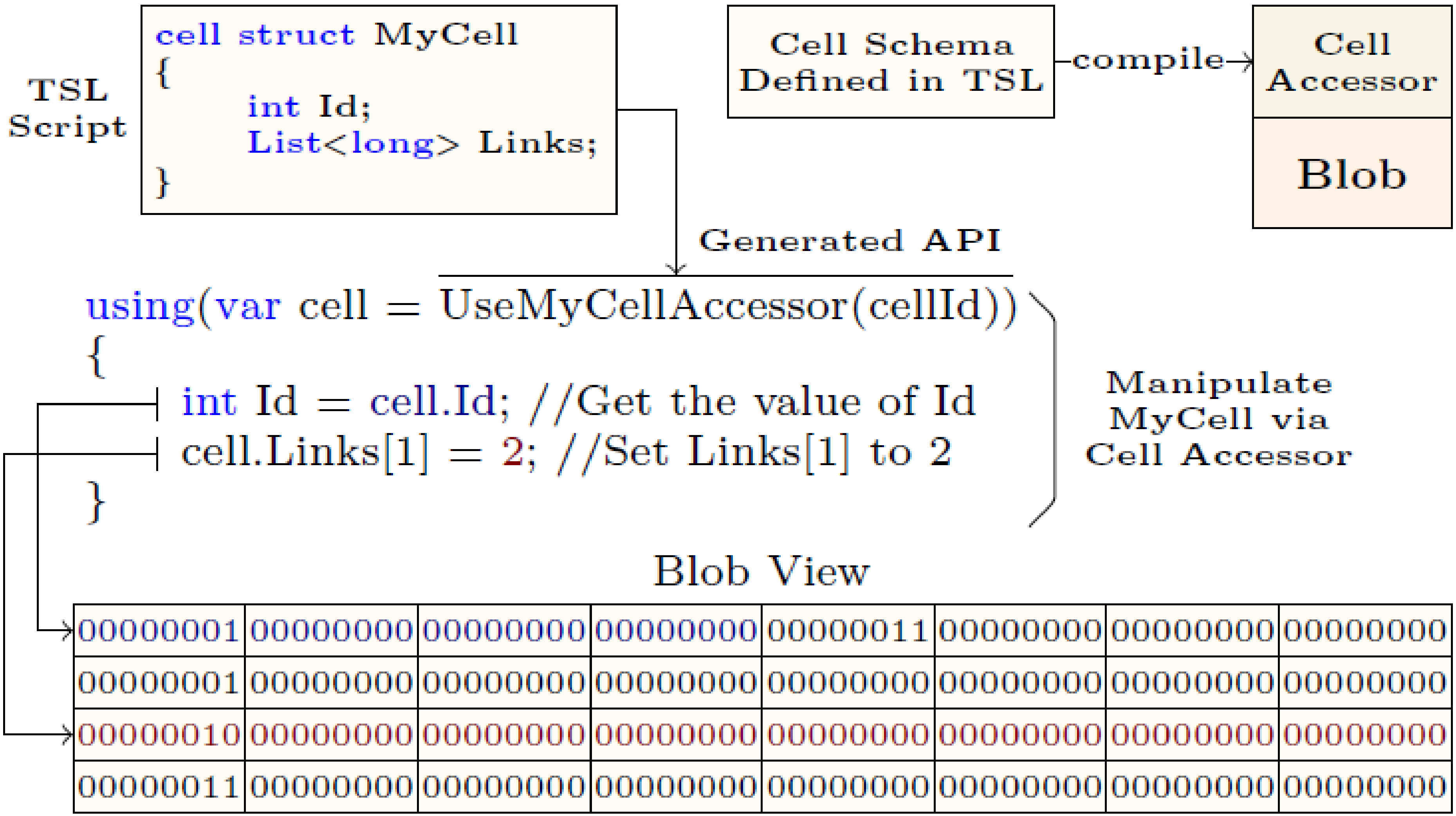
Architecture 1/2



Architecture 2/2



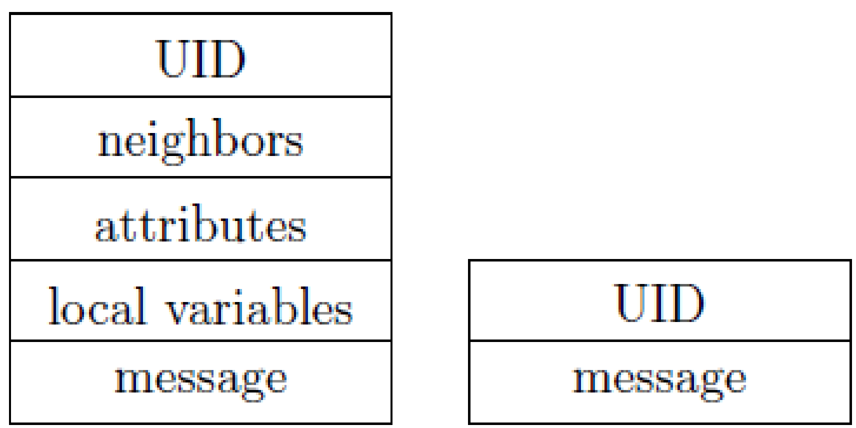
UNIVERSITY OF CAMBRIDGE



UNIVERSITY OF CAMBRIDGE

Message passing model

- Having all messages in memory: infeasible
- Assumption: fixed set of messaging partners
- Bipartite view effective but costly, better:



Type A vertices   Type B vertices

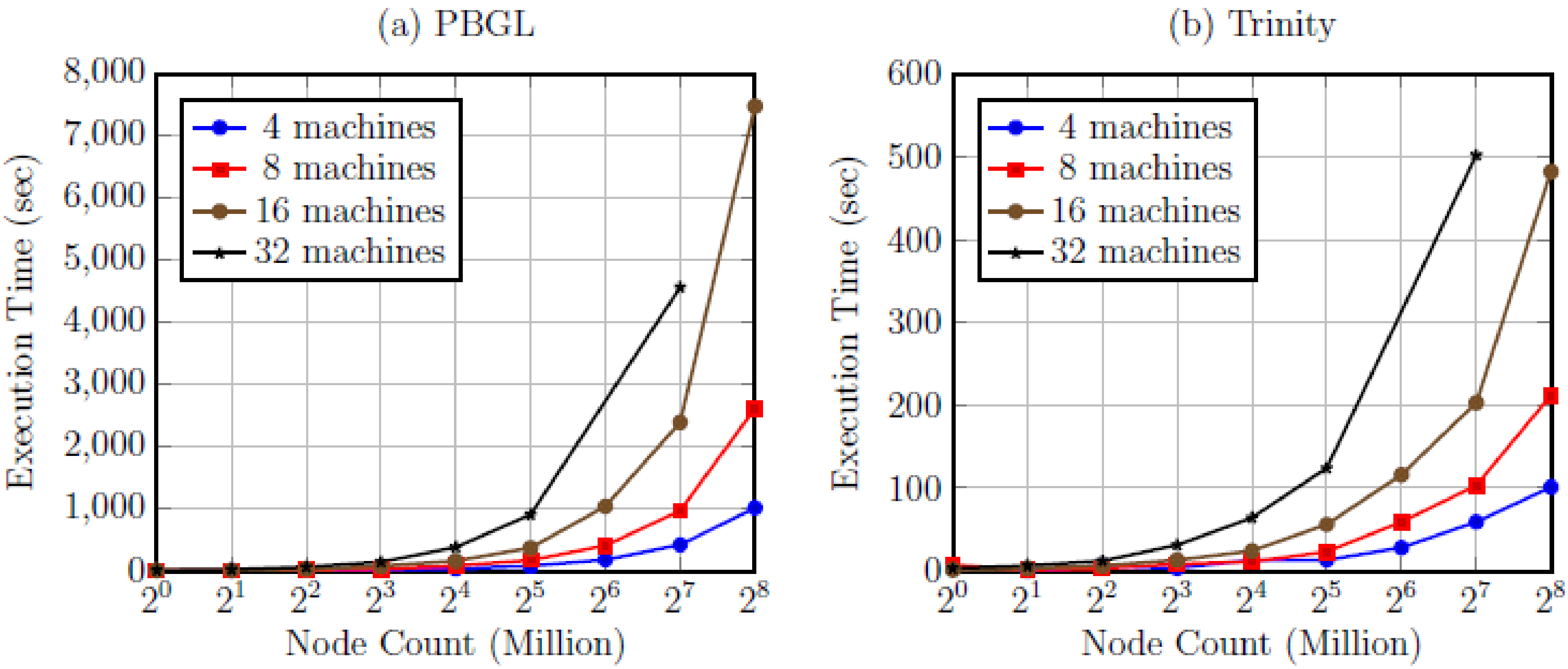
Trinity Specification Language

Object oriented manipulation through cell accessor abstraction

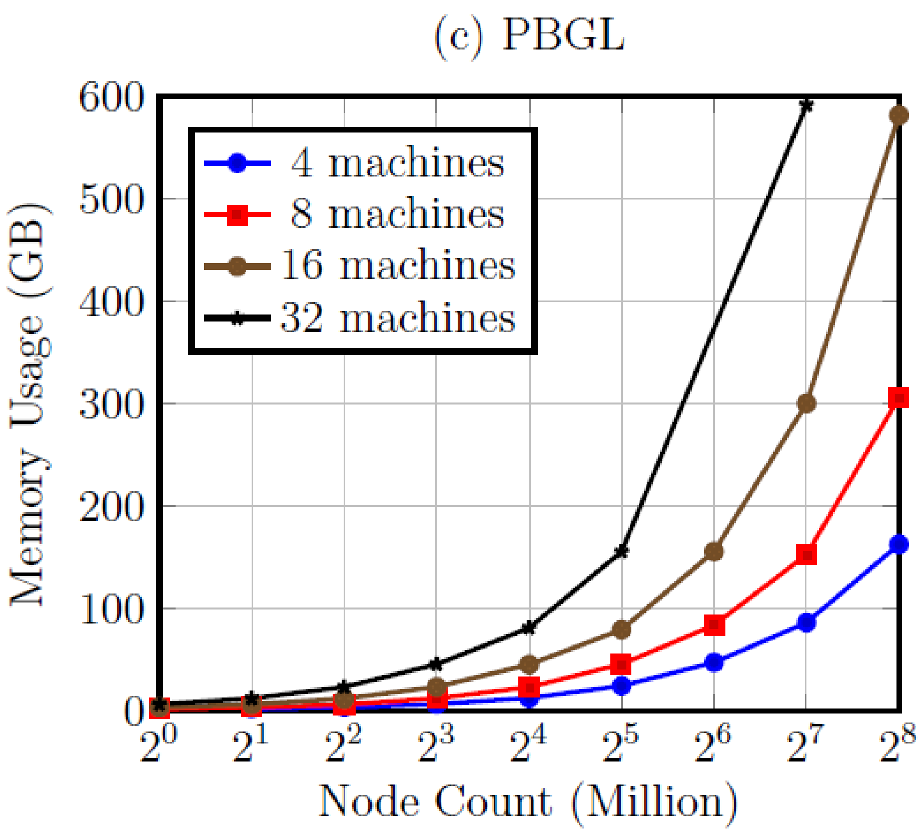
```
[CellType: NodeCell]
cell struct Movie
{
    string Name;
    [EdgeType: SimpleEdge, ReferencedCell: Actor]
    List<long> Actors;
}
```

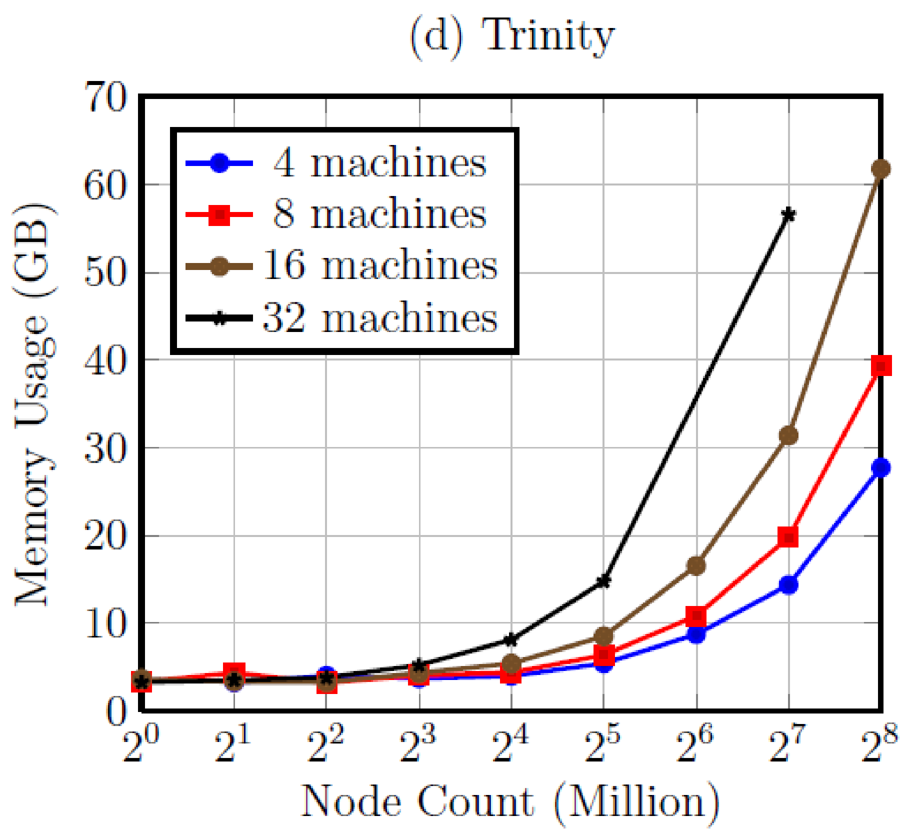
UNIVERSITY OF CAMBRIDGE

Evaluation - BFS



Evaluation – Memory usage





UNIVERSITY OF CAMBRIDGE

Conclusion

- Unified approach for distributed graph processing
- Efficient shared memory abstraction
- Fault recovery through checkpointing (depending on task)
- Library / coordination unclear
- No ACID transactions