

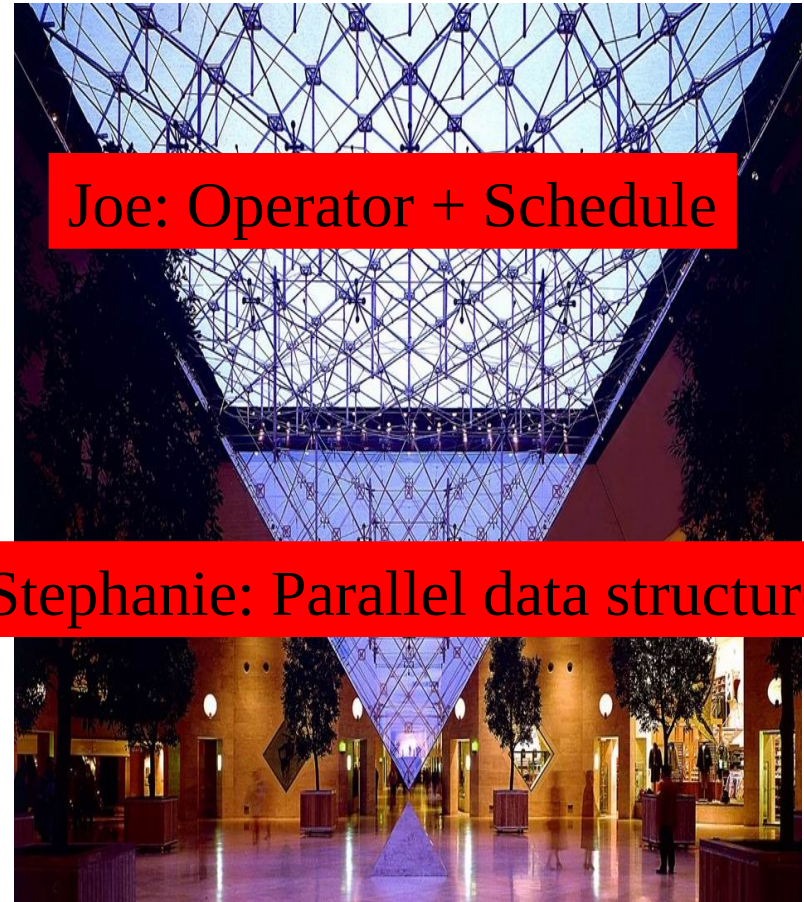
The Galois system  
for  
parallel graph analytics

Keshav Pingali  
The University of Texas at Austin

# Galois system

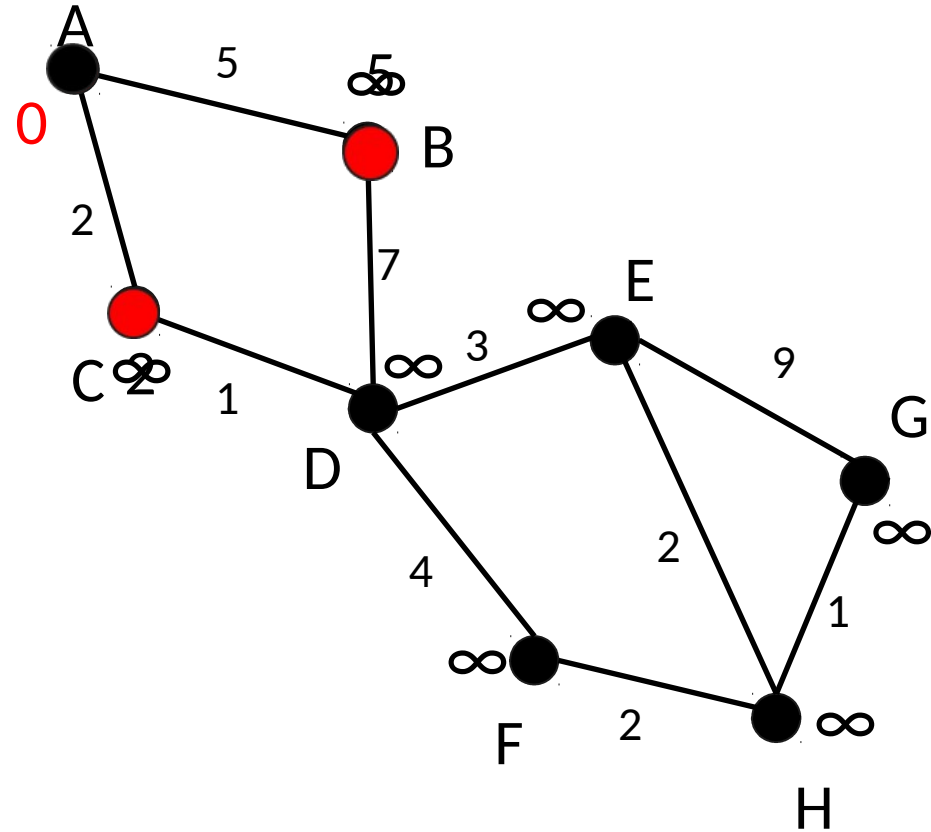
Parallel program = Operator + Schedule + Parallel data structures

- “Scalable” parallel programming:
  - small number of expert programmers (Stephanies) must support large number of application programmers (Joes)
  - cf. SQL
- Galois system:
  - Stephanie: library of concurrent data structures and runtime system
  - Joe: application code in sequential C++
    - Galois set iterator for highlighting opportunities for exploiting parallelism



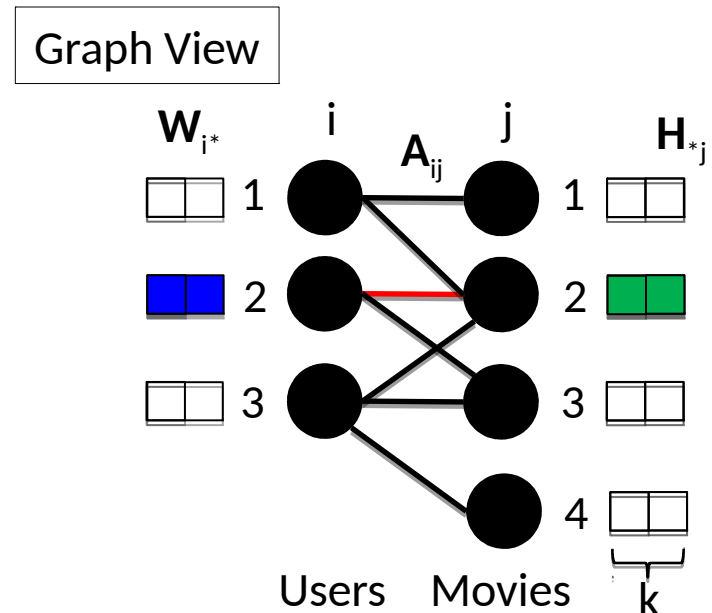
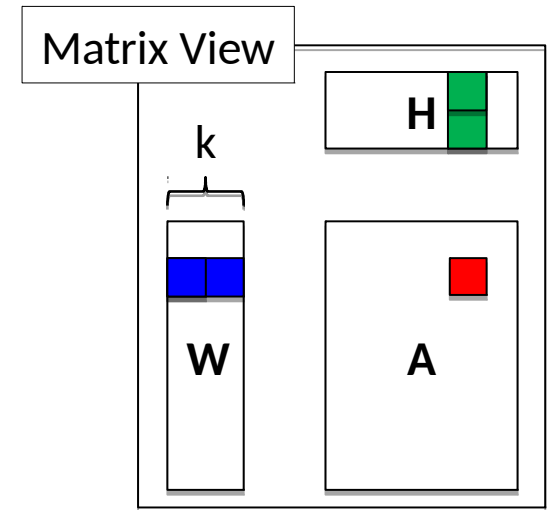
# Example: Graph analytics

- Single-source shortest-path
- Many algorithms
  - Dijkstra (1959)
  - Bellman-Ford (1957)
  - Chaotic relaxation (1969)
  - Delta-stepping (1998)
- Common structure:
  - Each node has distance label  $d$
  - Operator:  
relax-edge( $u,v$ ):
    - if  $d[v] > d[u] + \text{length}(u,v)$
    - then  $d[v] \leftarrow d[u] + \text{length}(u,v)$
  - Active node: unprocessed node whose distance field has been lowered
  - Different algorithms use different schedules
  - Schedules differ in parallelism, locality, work efficiency

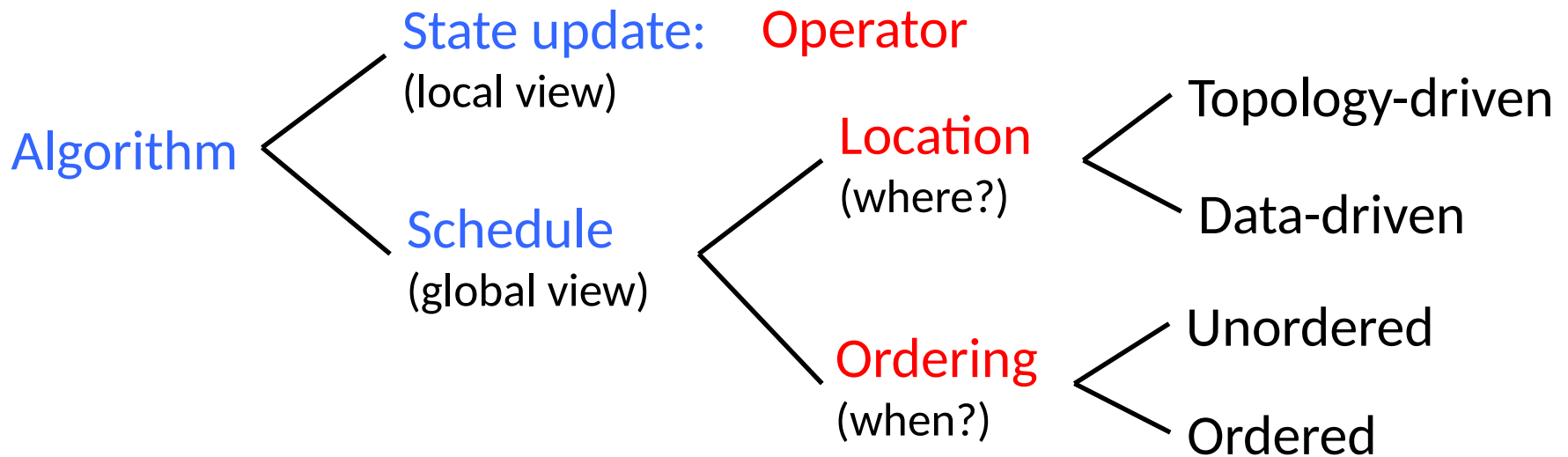
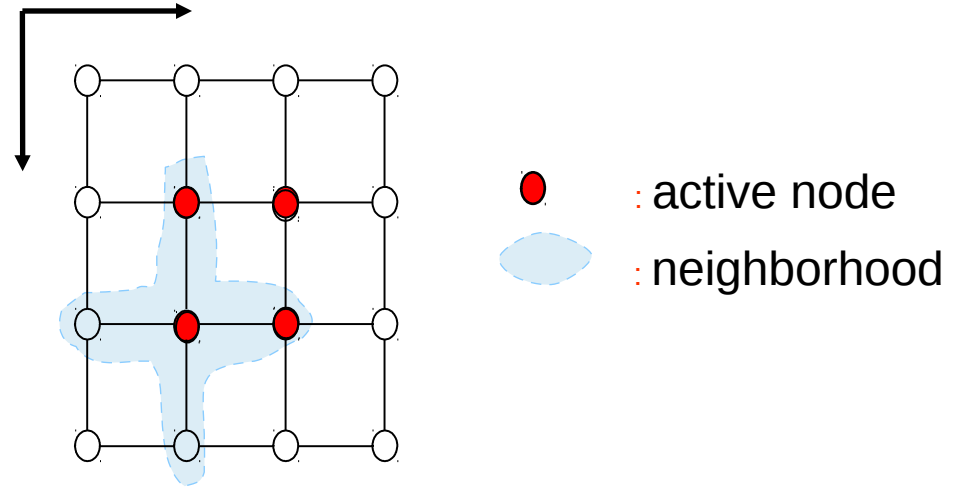
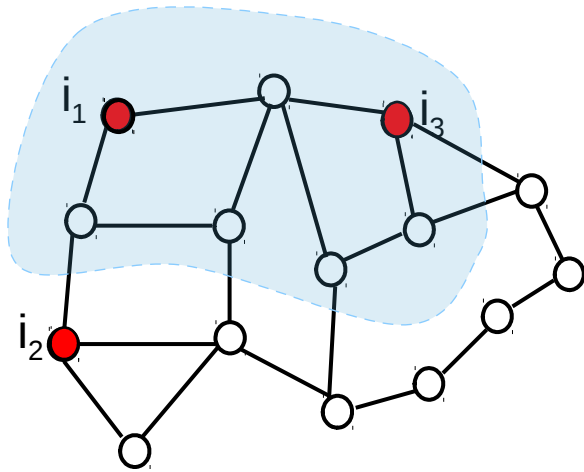


# Matrix Completion using SGD

- **Matrix completion problem**
  - Given incomplete database  $A$  of (user,movie) ratings, predict missing ratings
- **An optimization problem**
  - Find  $m \times k$  matrix  $\mathbf{W}$  and  $k \times n$  matrix  $\mathbf{H}$  ( $k \ll \min(m,n)$ ) such that  $\mathbf{A} \approx \mathbf{WH}$
  - Low-rank approximation
- **SGD algorithm**
  - Round-based execution
  - In each round, iterate over edges and heuristically update node labels



# Operator formulation



Parallel program = Operator + Schedule + Parallel data structure

# Galois programs: parallel execution

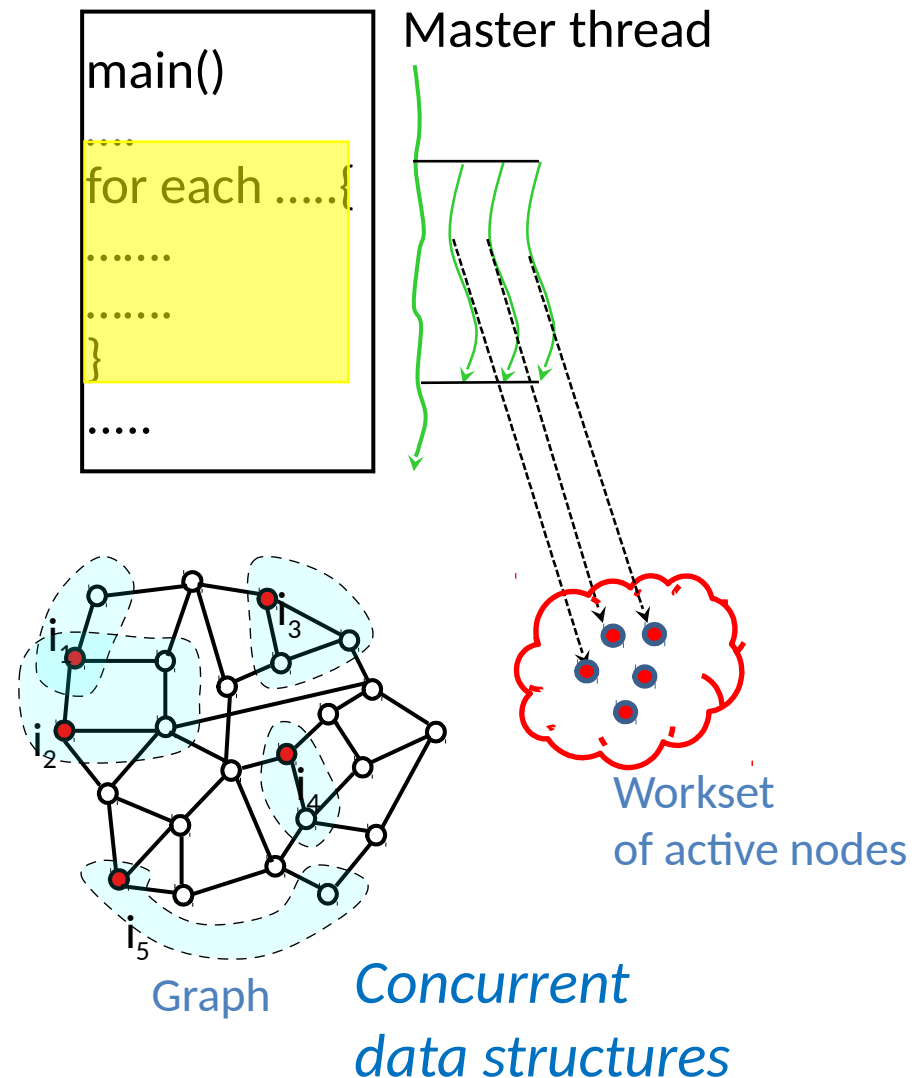
## • Joe program

- Sequential C++
- Galois set iterator: for each
  - New elements can be added to set during iteration
  - Body is operator; data structure accesses through API
  - Optional scheduling specification (cf. OpenMP)
  - Highlights opportunities in program for exploiting amorphous data-parallelism

## • Runtime system

- Ensures serializability of iterations
- Execution strategies
  - Optimistic
  - Interference graphs

## Application Program



# Scheduling in Galois

- Users write high-level scheduling policies
  - DSL for specifying policies
- Separation of concerns
  - Users understand their operator but have vague understanding of scheduling
  - Schedulers are difficult to implement efficiently
- Galois system synthesizes scheduler by composing elements from a scheduler library

# Hello graph Galois Program

```
#include "Gabis/Gabis.h"
#include "Gabis/Graphs/LC_Graph.h"

struct Data { int value; float f; };

typedef Gabis::Graph::LC_CSR_Graph<Data, void> Graph;
typedef Gabis::Graph::GraphNode Node;

Graph graph;

struct P {
    void operator()(Node n, Gabis::UserContext<Node> & ctx){
        graph.getData(n).value += 1;
    }
};

int main(int argc, char** argv) {
    graph.structureFromGraph(argv[1]);
    Gabis::for_each(graph.begin(), graph.end(), P());
    return 0;
}
```

Data structure  
Declarations

Operator

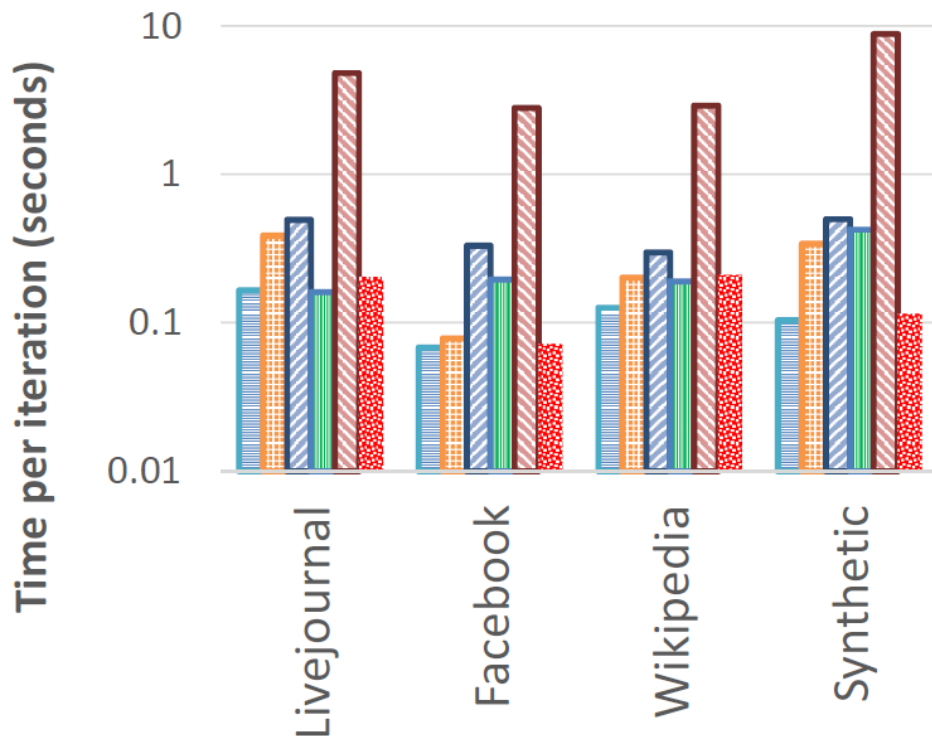
Galois Iterator



# Performance studies

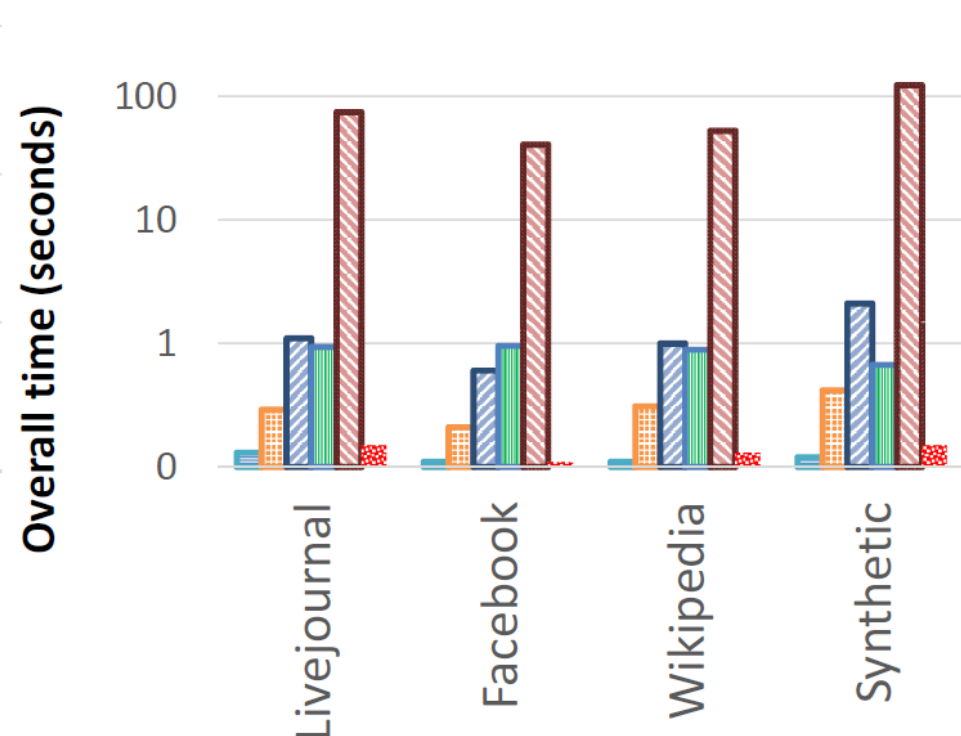
# Intel Study: Galois vs. Graph Frameworks

Native    Combbblas    Graphlab  
Socialite    Giraph    Galois



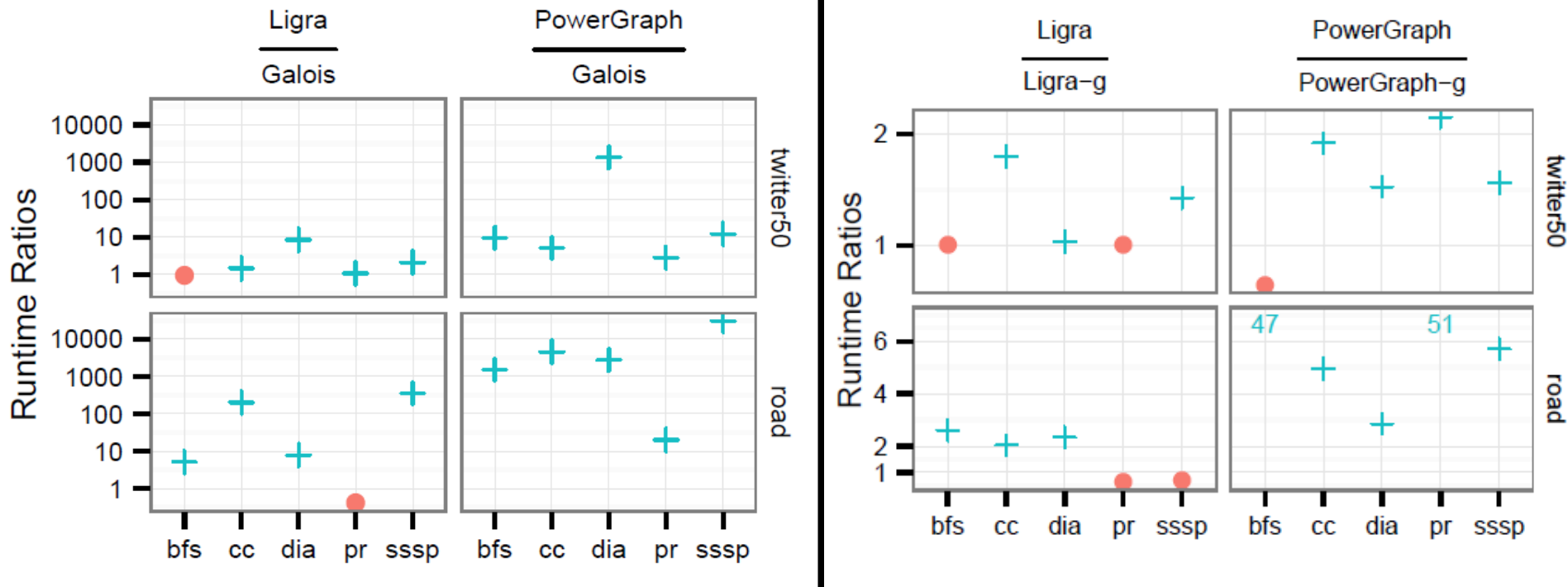
(a) PageRank

Native    Combbblas    Graphlab  
Socialite    Giraph    Galois



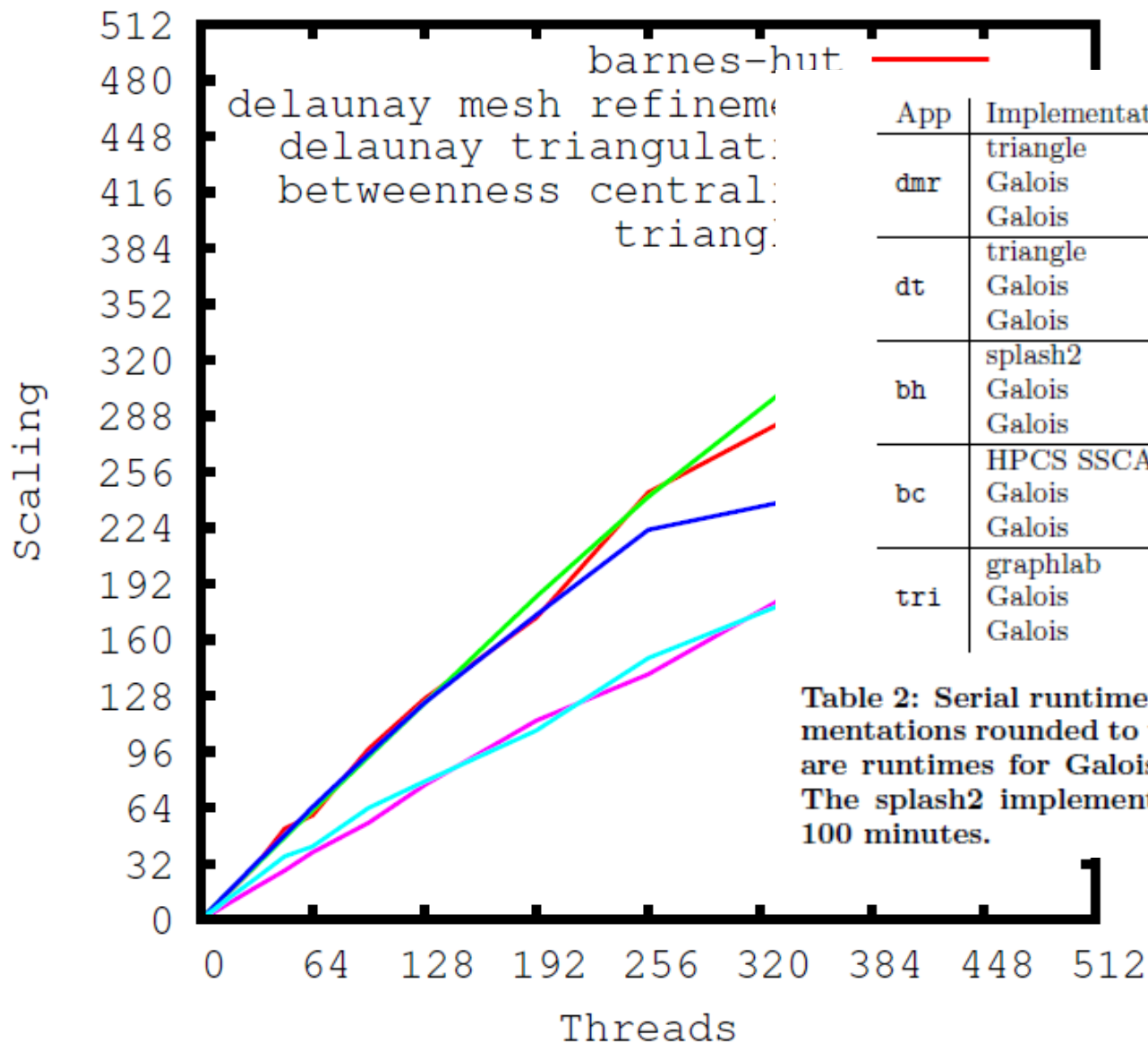
(b) Breadth-First Search

# Galois: Graph analytics



- Galois lets you code more effective algorithms for graph analytics than DSLs like PowerGraph (left figure)
- Easy to implement APIs for graph DSLs on top on Galois and exploit better infrastructure (few hundred lines of code for PowerGraph and Ligra) (right figure)

# Galois: Performance on SGI Ultraviolet



App	Implementation	Threads	Time (s)
dmr	triangle	1	96
	Galois	1	155.7
	Galois	512	0.37
dt	triangle	1	1185
	Galois	1	56.6
	Galois	512	0.18
bh	splash2	1	>6000
	Galois	1	1386
	Galois	512	3.55
bc	HPCS SSCA	1	6720
	Galois	1	5394
	Galois	512	21.6
tri	graphlab	2	531
	Galois	1	7.03
	Galois	512	0.028

**Table 2: Serial runtime comparisons to other implementations rounded to the nearest second. Included are runtimes for Galois algorithms at 512 threads. The splash2 implementation of bh timed out after 100 minutes.**

# Summary

- Operator formulation of algorithms:
  - Data-centric description of algorithms
  - Algorithm = Operator + Schedule
- Parallel program = Operator + Schedule + Parallel data structures
- Complexity of parallel programming arises mainly from parallel data structures
- Galois approach to scalable parallel programming:
  - Stephanie:
    - library of parallel data structures and schedulers
    - runtime system that provides services like load-balancing
  - Joe: specifies operator and schedule

# More information

- Website
  - <http://iss.ices.utexas.edu>
- Download
  - Galois system for multicores
  - Lonestar benchmarks
  - All our papers
- Coming soon
  - Distributed-memory
  - CUDA and OpenCL backends