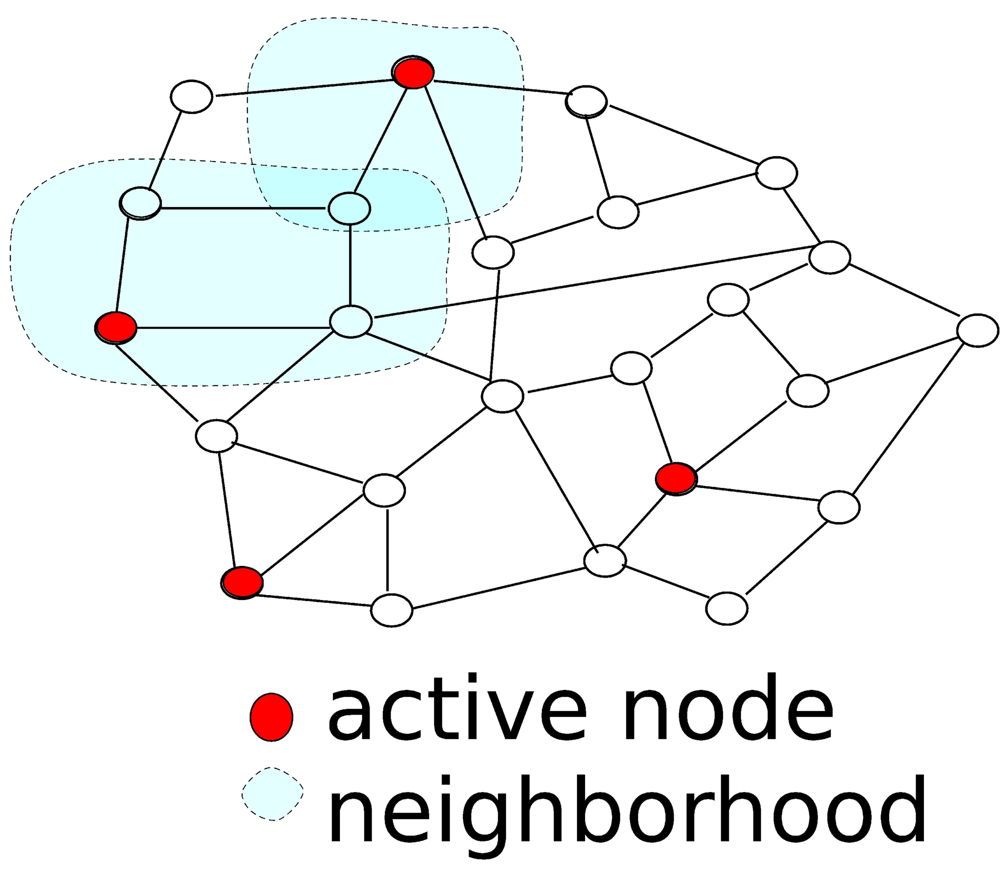
Galois

Scalable Abstractions for Scalable Systems

Andrew Lenharth (lenharth@ices.utexas.edu)
Keshav Pingali (pingali@cs.utexas.edu)
Marc Snir (snir@cs.uiuc.edu)

Parallel program = Operator + Schedule + Parallel data structure

Operator Formulation



Active element
— Site where computation is needed

Operator
— Computation at active element

Activity
— application of operator to active element

Neighborhood
— Set of nodes/edges read/written by activity
— Distinct usually from neighbors in graph

Ordering
— Scheduling constraints on execution order of activities
— Unordered algorithms: no semantic constraints but performance may depend on schedule
— Ordered algorithms: problem-dependent order

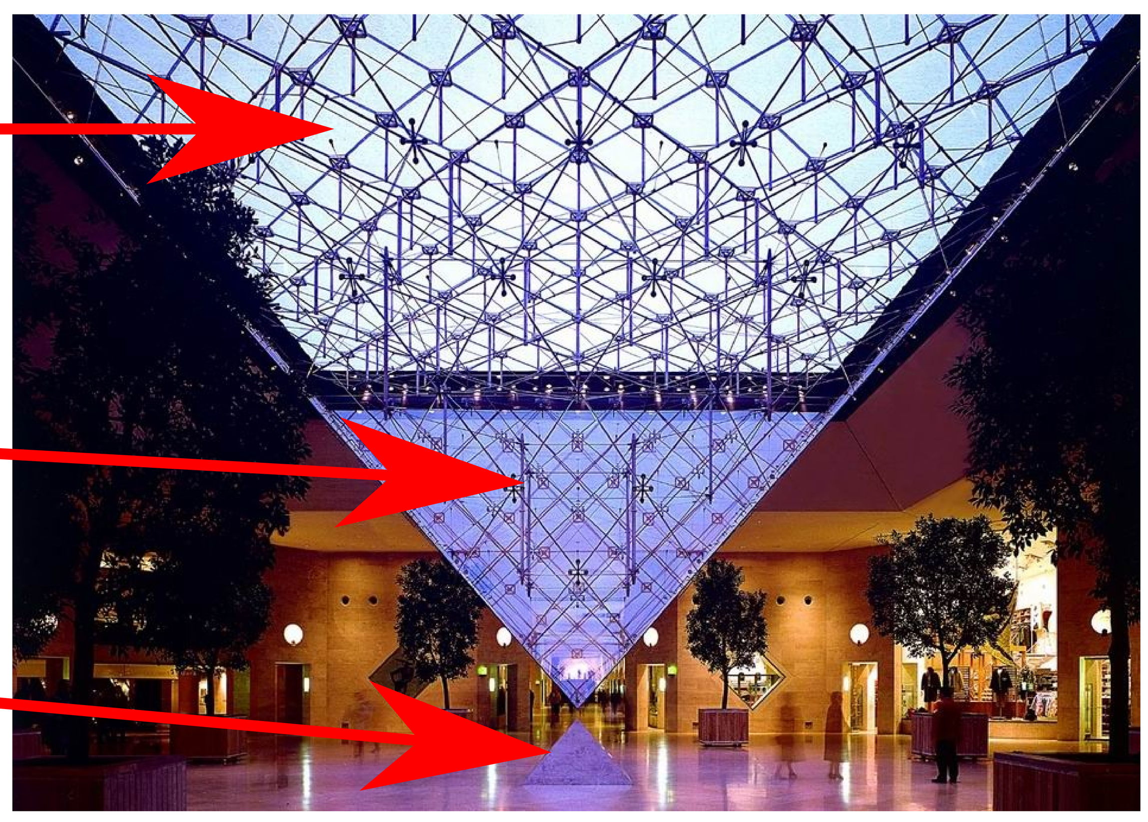
Amorphous data-parallelism
— Multiple active nodes can be processed in parallel subject to neighborhood and ordering constraints

Multi-Level Programming

Joe Programmer: Operator and Schedule Specification

Stephanie Programmer: Parallel Data Structures

Obi-Wan Programmer: Synchronization, NUMA, Scalable Runtime

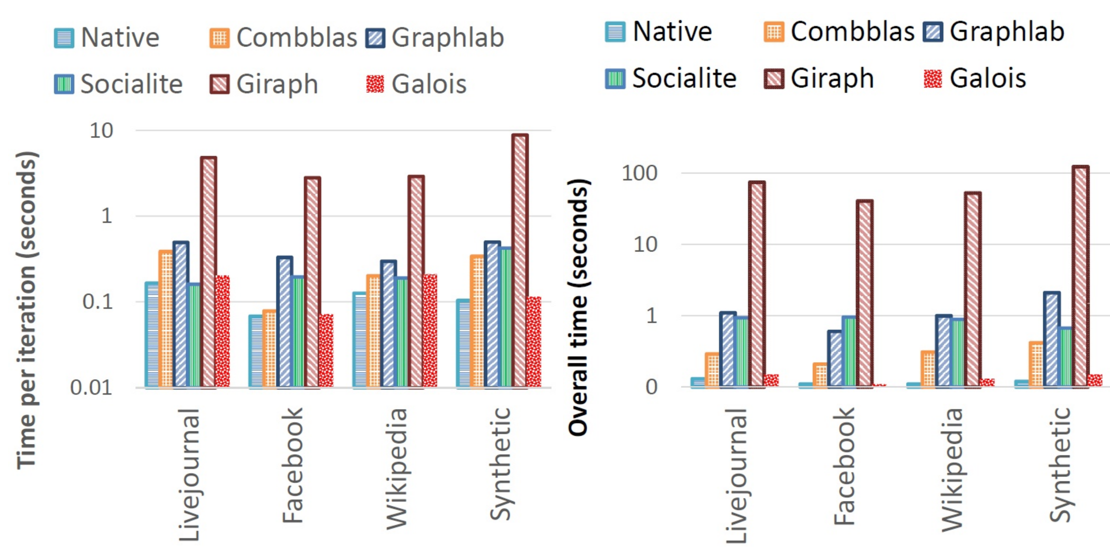


Ubiquitous parallelism:
— small number of expert programmers (Stephanies) must support large number of application programmers (Joes)
— cf. SQL

Stephanie
— Library of concurrent data structures
— Provides serializable, atomic execution of activities

Joe
— Application code in stylized, sequential C++
— Uses Galois set iterator for highlighting opportunities for exploiting ADP and Galois data structures for concurency control

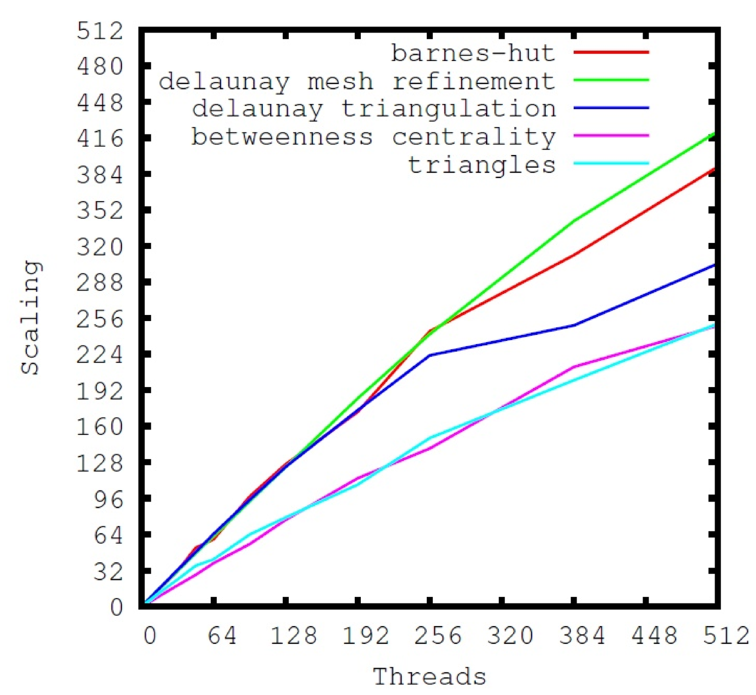
Native Performance



(a) PageRank (b) Breadth-First Search

Intel HPC Study: Galois implementations are comperable to hand-written and optimized code.
"Navigating the maze of graph analytics frameworks" Nadathur et al SIGMOD 2014

Scaling



Numa-aware runtime and data structures for high multi-core scaling. Real performance — single-thread performance is on par with optimized serial code.

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

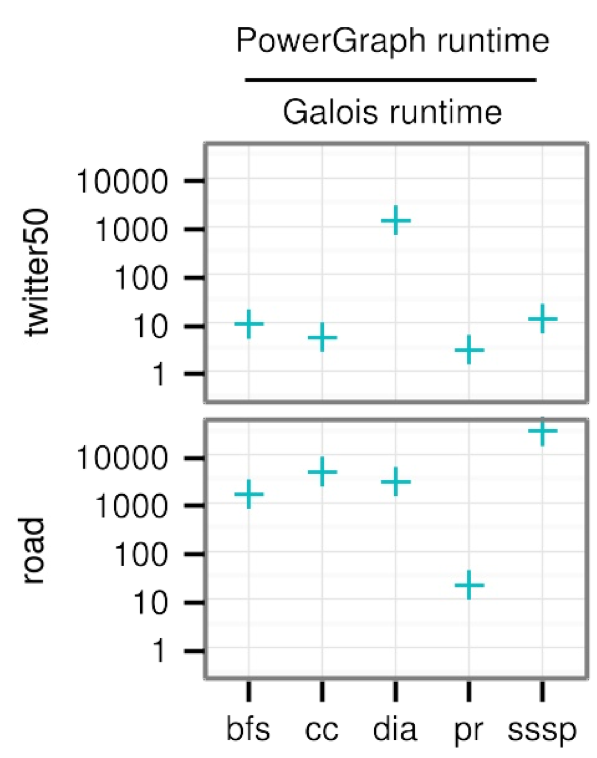
Table 2: Serial runtime comparisons to other implementations rounded to the nearest second. Included are runtimes for Galois algorithms at 512 threads. The splash2 implementation of bh timed out after 100 minutes.

Think Beyond a Vertex

The best algorithm may not be expressible as a vertex program
— Connected components with union-find

The best algorithm may require application-specific scheduling
— Priority scheduling for SSSP

Autonomous scheduling required for high-diameter graphs
— Coordinated scheduling uses many rounds and has too much overhead



One Model, Many Targets

GPUs
— Optimized implementation strategies for coalesing, synchronization, high-thread counts
— Scheduling and load balancing for highly-parallelism
— Multi-gpu support

FPGAs
— Optimized scheduling and conflict detection
— Targets network of FPGAs architectures

Distributed Memory
— Transparent support for distributed memory for arbitrarily complex irregular algorithms

Xeon PHI
— Numa and memory optimizations lead to out-performing simple pthread/openMP/openCL codes

Heterogeneous and emerging
— Mixed GPU/CPU
— In progress: Cluster of multicore/CPU, coherent CPU/FPGA