Ligra: A lightweight Graph Processing Framework for Shared Memory

Authored by Julian Shun and Guy E. Blelloch

Presented by: Ruquaiya Shuaaibu (rs2377)

Motivation/Background

Massive graphs in social, web, and biological networks

Most frameworks (Pregel, GraphLab, Pegasus, etc.) target distributed clusters

But modern multicore servers can hold terabytes of RAM → billions of edges fit in memory

Can shared-memory systems rival distributed frameworks for large-scale graph processing?

Research Questions

How can graph algorithms be parallelized efficiently on shared memory?

Can a lightweight abstraction simplify graph programming?

Can such a framework achieve performance comparable to large distributed systems?

Shared vs Distributed

Aspect	Shared Memory (Ligra)	Distributed (Pregel, Grap, etc)
Communication	In-memory (fast)	Network (slow)
Complexity	Simple, no partitioning	Complex message passing
Performance per core	Higher	Lower
Scalability	Limited by RAM	Scales across nodes
Sync Method	Atomic (CAS)	Bulk sync/message passing

Ligra: Core Idea

Two key primitives:

VERTEXMAP: Apply a function over active vertices

EDGEMAP: Apply a function over edges from active vertices

Framework automatically switches between:

Sparse mode (few active vertices)

Dense mode (many active vertices)

Inspired by hybrid BFS optimisation

Simplified example

Start with root vertex

Use EDGEMAP to expand to neighbors

Uses atomic compare-and-swap (CAS) to mark visited vertices

Framework handles parallelism automatically

Algorithms Supported

Breadth-First Search (BFS)

PageRank

Betweenness Centrality

Graph Connectivity

Radii Estimation

Bellman-Ford Shortest Paths

Results

Setup: 40-core Intel Xeon (256 GB RAM)

Graphs up to 12.9 billion edges

Highlights:

20–35× speedup vs single-thread baseline

Per-core performance > distributed frameworks (Pregel, GraphLab, etc.)

Often faster overall despite fewer cores

Very concise code (BFS in ~15 lines)

Main Contributions

Lightweight framework with two simple primitives (VERTEXMAP, EDGEMAP)

Adaptive hybrid traversal (sparse ↔ dense)

Efficient, simple implementations of core graph algorithms

Demonstrates that shared-memory systems can rival or outperform distributed systems per core

Opinions/critiques

Data Scale Has Outgrown Single Machines

Lacks Fault Tolerance & Elasticity

Static Graph Assumption

Still Influential & Efficient