Introduction

Graphs are growing fast and are becoming increasingly popular. Many problems can be modeled and solved using graphs.

We compare some state-of-the-art non-uniform memory access (NUMA) aware systems and Giraph in terms of performance, both on real world and synthetic graphs. Therefore, we evaluate the three algorithms Single-Source Shortest-Paths (SSSP), Breadth-First Search (BFS) and PageRank (PR). We compare not only the time the frameworks indicate, they need to compute, but also the time they actually run and consume resources, as well as the resulting overhead time.

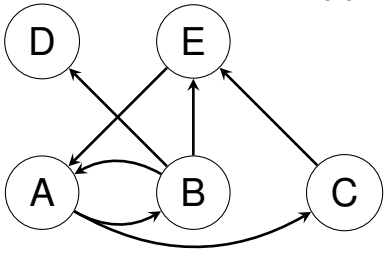
Overview

Preliminaries

A *weighted, directed graph* is the pair $G = (V, E, w)$ where the *vertex set* is $V \subseteq \mathbb{N}$ and the E is the *edge set* with

$$E \subseteq \{(x, y) \mid x, y \in V, x \neq y\}$$

and $w : E \to \mathbb{R}$ is a mapping of edge to a weight.



Algorithms:

- Single-Source Shortest-Paths (SSSP) is the problem of finding the shortest path from a starting vertex to every other vertex in the input graph.
- Breadth-first search (BFS) is a search problem on a graph.
- PageRank (PR) is a link analysis algorithm that weighs the vertices of a graph, measuring the vertices relative importance within the graph

Push & Pull

Hugepages

SSSP, BFS, PR, Push/Pull, Hugepages

Frameworks

Evaluation

5 Machines, with

- 96 cores, of which 48 virtual
- 256 GB of RAM each, one machine only 128 GB
- Ubuntu 18.04.2 LTS

Measurements:

- **execution time**: time from start to finish of the console command
- **calculation time**: time the framework actually executed the algorithm
- **overhead**: time difference between execution time and calculation time

(time to read the input graph, initialization, etc.)

Graph	# Vertices (M)	# Edges (M)
flickr	0.1	2
orkut	3	117
wikipedia	12	378
twitter	52	1963
rMat27	63	2147
friendster	68	2586
rMat28	121	4294

each test case (graph, framework, algorithm) was run 10 times

Production Case

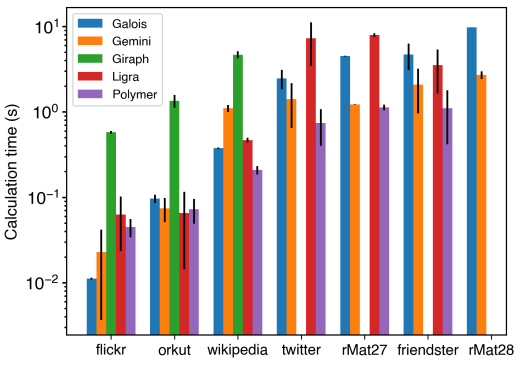
- running system, that performs multiple calculations on a single graph
- without the need of reloading graph data with every calculation
- short calculation times should be preferred because the overhead time is only spent once on startup and amortizes quickly

Messfehlern bei Galois Calc was sagen

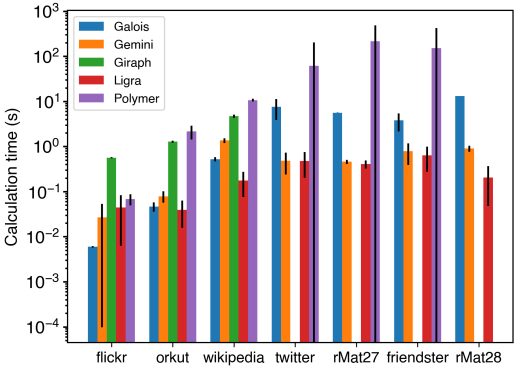
Research Case

- individual calculations on a graph, i.e. for each calculation, the graph has to be loaded
- the algorithm can change frequently
- requiring the framework to be relatively fast on different algorithms
- overall small execution times and small overhead are preferred

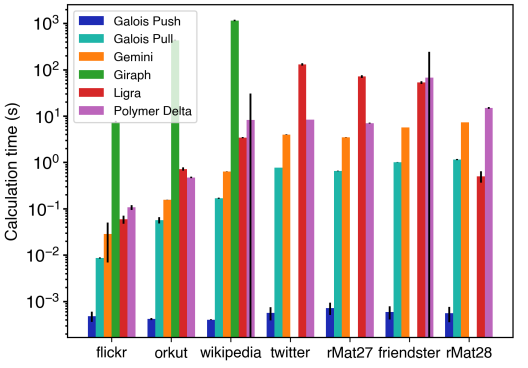
Production Case Single Node



(a) SSSP



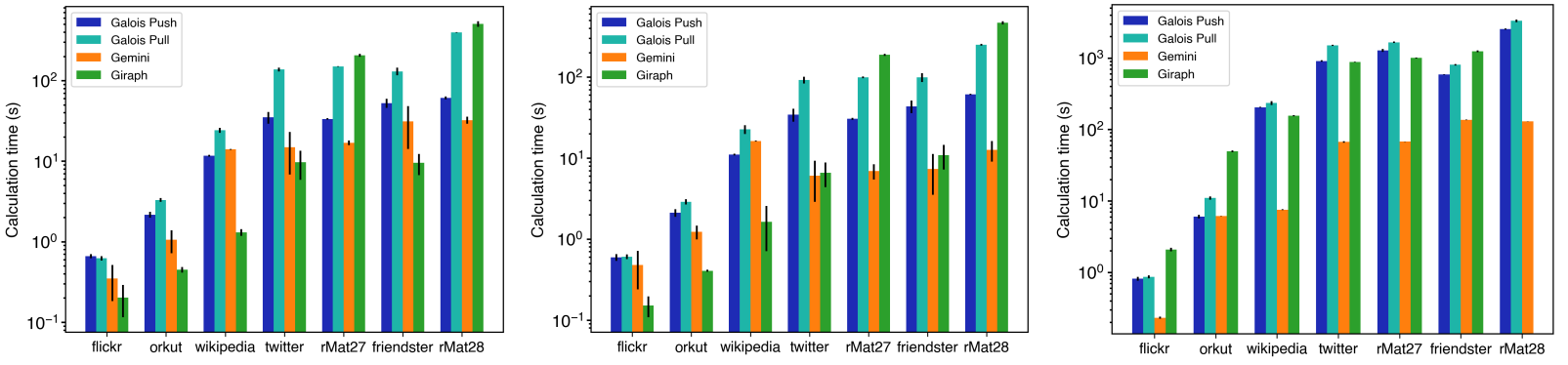
(b) BFS



(c) PR

- Giraph is either very slow or requires too much RAM (>256 GB)
- On SSSP, Polymer is fastest, followed by Gemini on second place
- On BFS, Gemini and Ligra are comparable and fastest on the larger graphs
- On PR, Galois is fastest. But we exclude Galois Push because of possible measuring errors.
- Message-based approach can compete with shared-memory

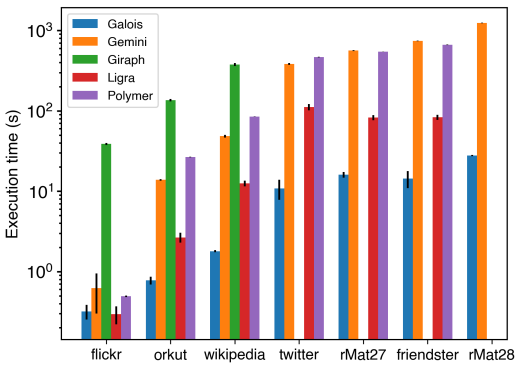
Production Case Distributed



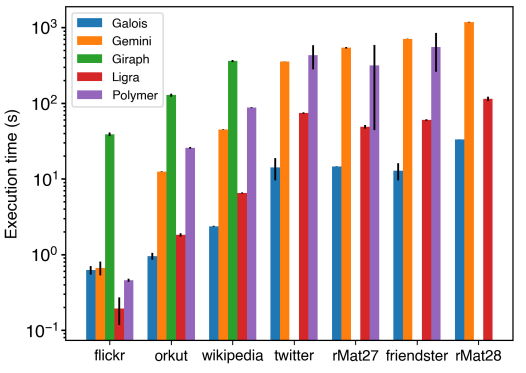
(a) SSSP (b) BFS (c) PR

- Giraph is fastest on SSSP and BFS on the real world graphs
- Giraph has problems with synthetic graphs
- Gemini is fastest on PR, with Giraph on second place

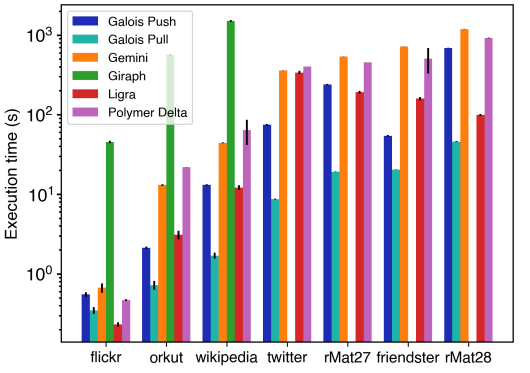
Research Case Single Node



(a) SSSP



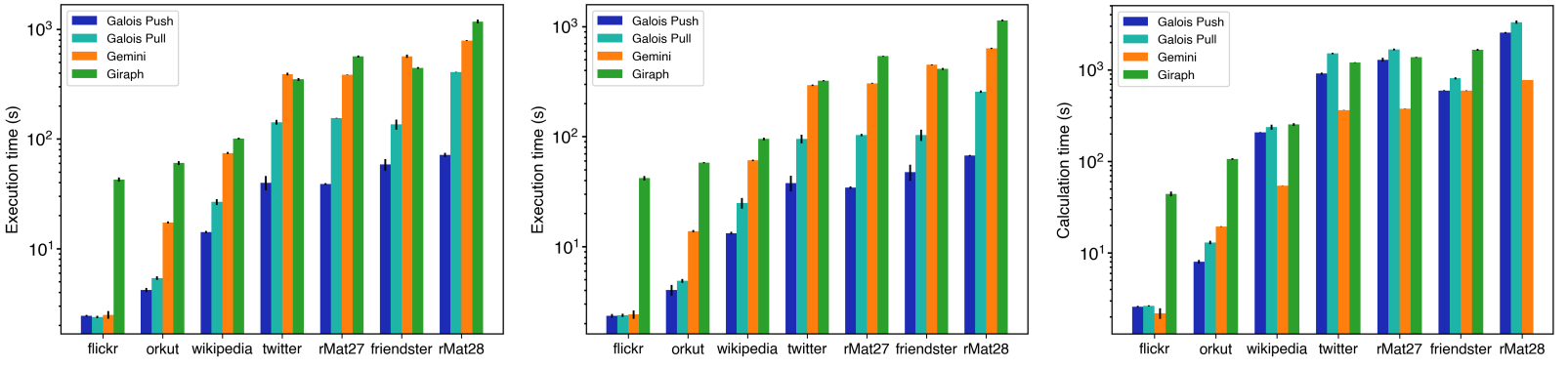
(b) BFS



(c) PR

- Giraph is either slowest or requires too much RAM (>256 GB)
- Galois is fastest in almost all cases, second fastest is Ligra
- Gemini and Polymer are comparably slow

Research Case Distributed



(a) SSSP (b) BFS (c) PR

- Galois Push is faster than Pull in all cases
- Either Galois implementation is faster than any other frameworks on SSSP or BFS
- Gemini is fastest on PR and comparable to Giraph on SSSP and BFS

Conclusion and Outlook

Für jeden Fall ist das beste Framework sehr spezifisch. Blabla.

Outlook: Immer wieder solche Tests machen, viele der Frameworks ändern sich immer wieder weiter und so, nicht wahr.