Beware

By

J Crowcroft from work by Malte Schwarzkopf & Frank McSherry

Computer Laboratory & Unafilliated

University of Cambridge

UNIVERSITY OF CAMBRIDGE

tl;dr #1

- Network speed may not matter with a Spark based stack, but it does matter to higher performance analytics stacks, and for graph processing especially.
- By moving from a 1G to a 10G network, we see a 2x-3x improvement in performance for timely dataflow.

tl;dr #2

- A well balanced distributed system offers performance improvements even for graph processing problems that fit into a single machine;
- running things locally isn't always the best strategy

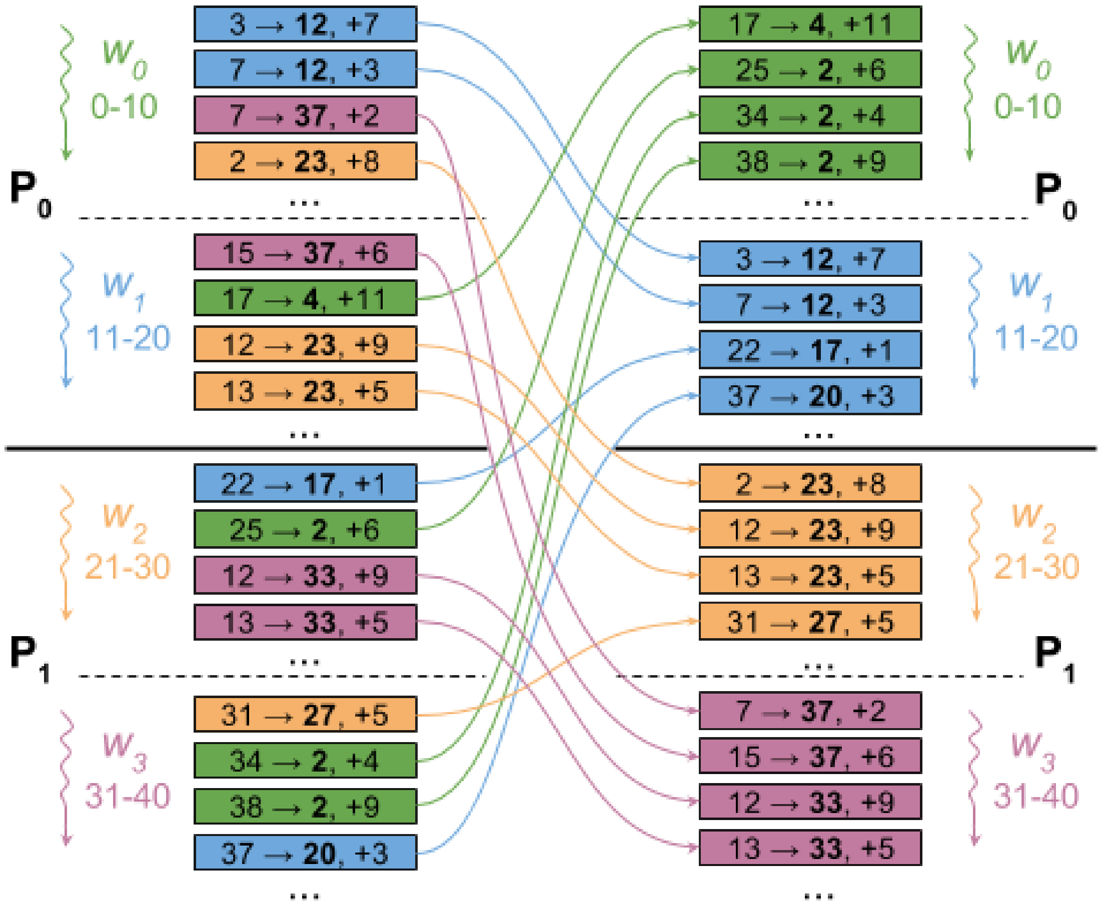
tl;dr #3

- PageRank performance on GraphX is primarily system bound. We see a 4x-16x performance increase when using timely dataflow on the same hardware, which suggests that GraphX (and other graph processing systems) leave an alarming amount of performance on the table

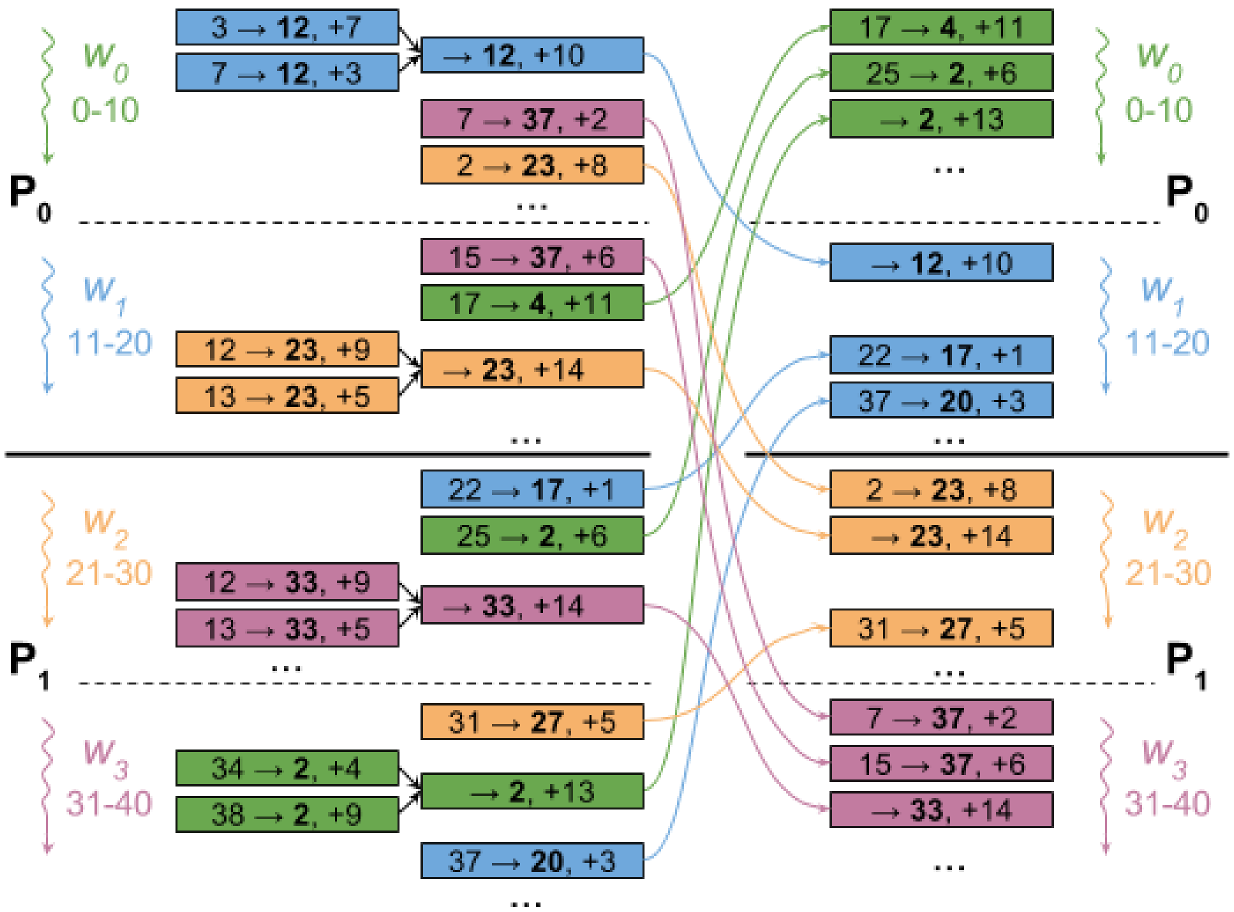
PageRank in Rust

```
fn pagerank(graph: &G: Graph, vertices: usize, alpha: f32)
{
    // mutable per-vertex state
    let mut src = vec![0f32; vertices];
    let mut dst = vec![0f32; vertices];
    let mut deg = vec![0f32; vertices];


    // determine vertex degrees
    for (x,_) in graph.edges() { deg[x] += 1f32; }

    // perform 20 iterations
    for _iteration in (0 .. 20) {

        // prepare src ranks
        for vertex in (0 .. vertices) {
            src[vertex] = alpha * dst[vertex] / deg[vertex];
            dst[vertex] = 1f32 - alpha;
        }

        // do the expensive part
        for (x,y) in graph.edges() { dst[y] += src[x]; }
    }
}
```

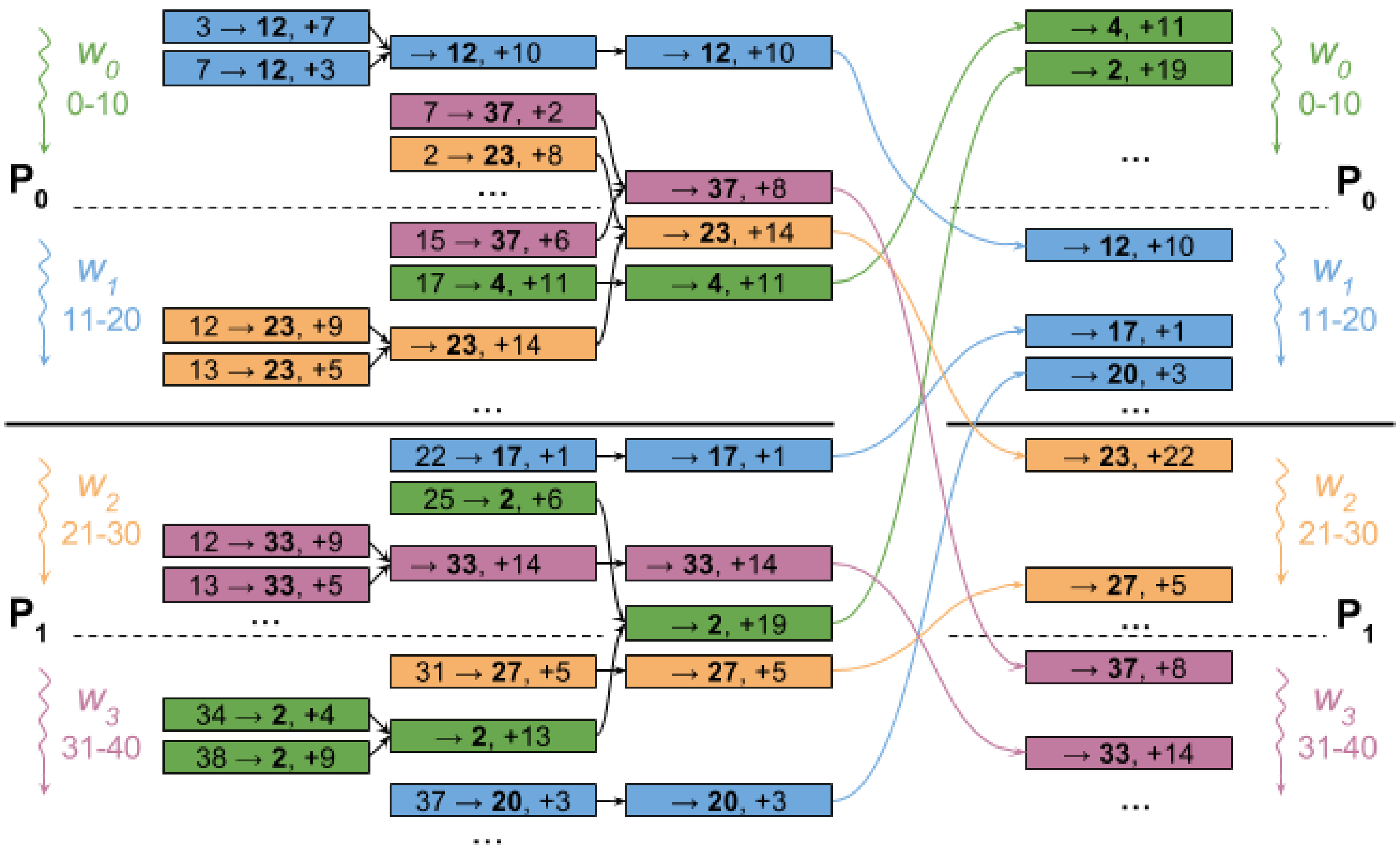
Impl #1: Send everything



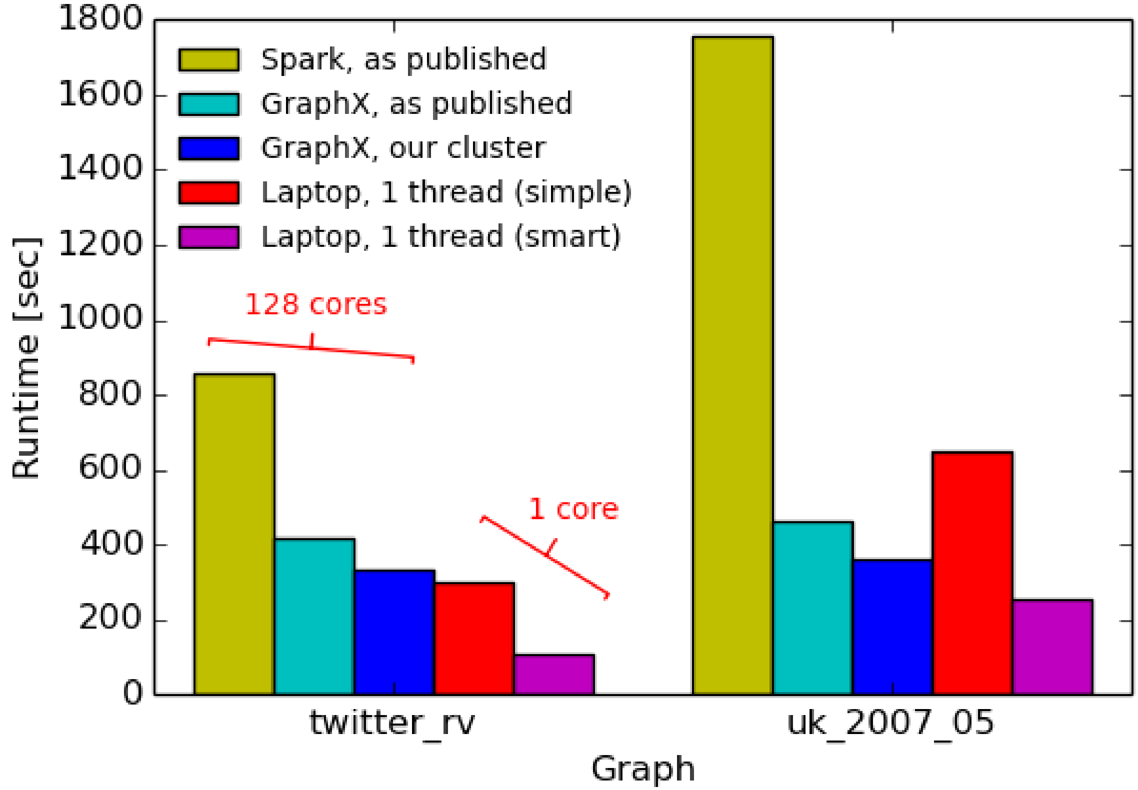
Impl #2: Worker-level aggregation



Impl #3: Process-level aggregation



Some Baseline figures

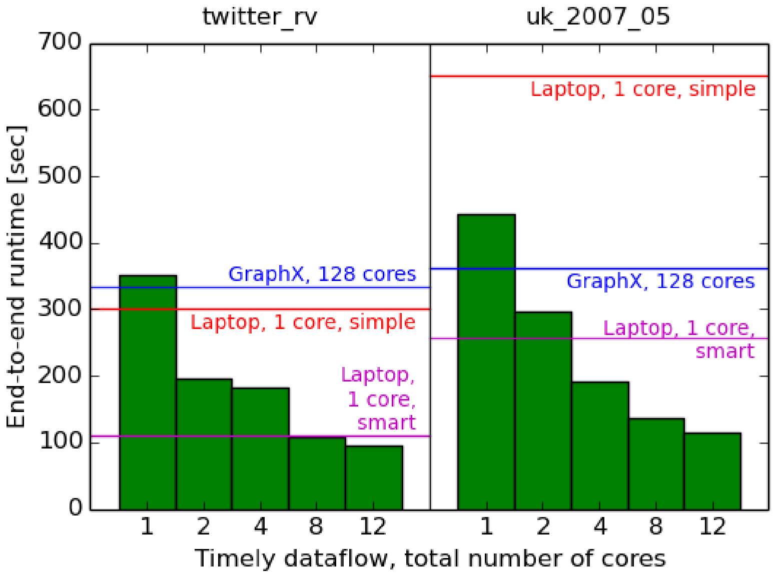


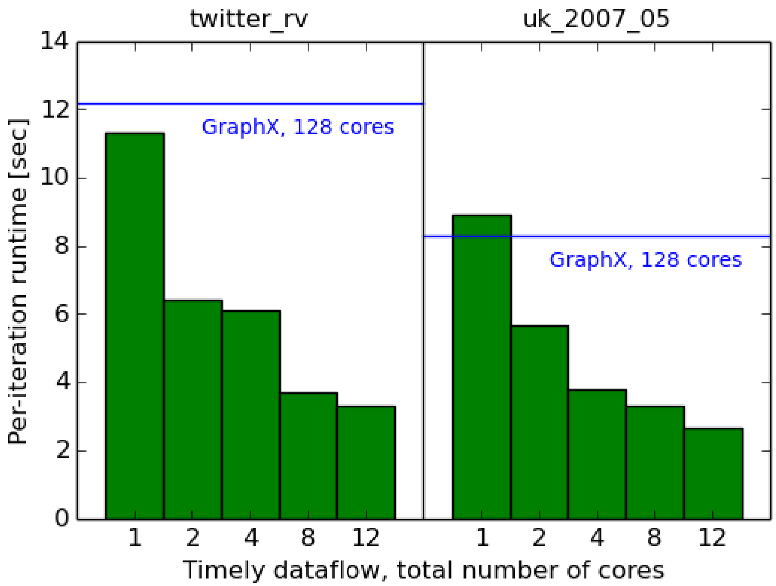
Twenty pagerank iterations, baseline measurements.

System

System	source	cores	twitter_rv	uk_2007_05
Spark	GraphX paper (https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-gonzalez.pdf)	16x8	857s	1759s
GraphX	GraphX paper (https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-gonzalez.pdf)	16x8	419s	462s
GraphX	measured on our cluster	16x8	334s	362s
Single thread (simpler)	COST paper (https://www.usenix.org/conference/hotos15/workshop-program/presentation/mcsherry)	1	300s	651s
Single thread (smarter)	COST paper (https://www.usenix.org/conference/hotos15/workshop-program/presentation/mcsherry)	1	110s	256s

Timely dataflow impl

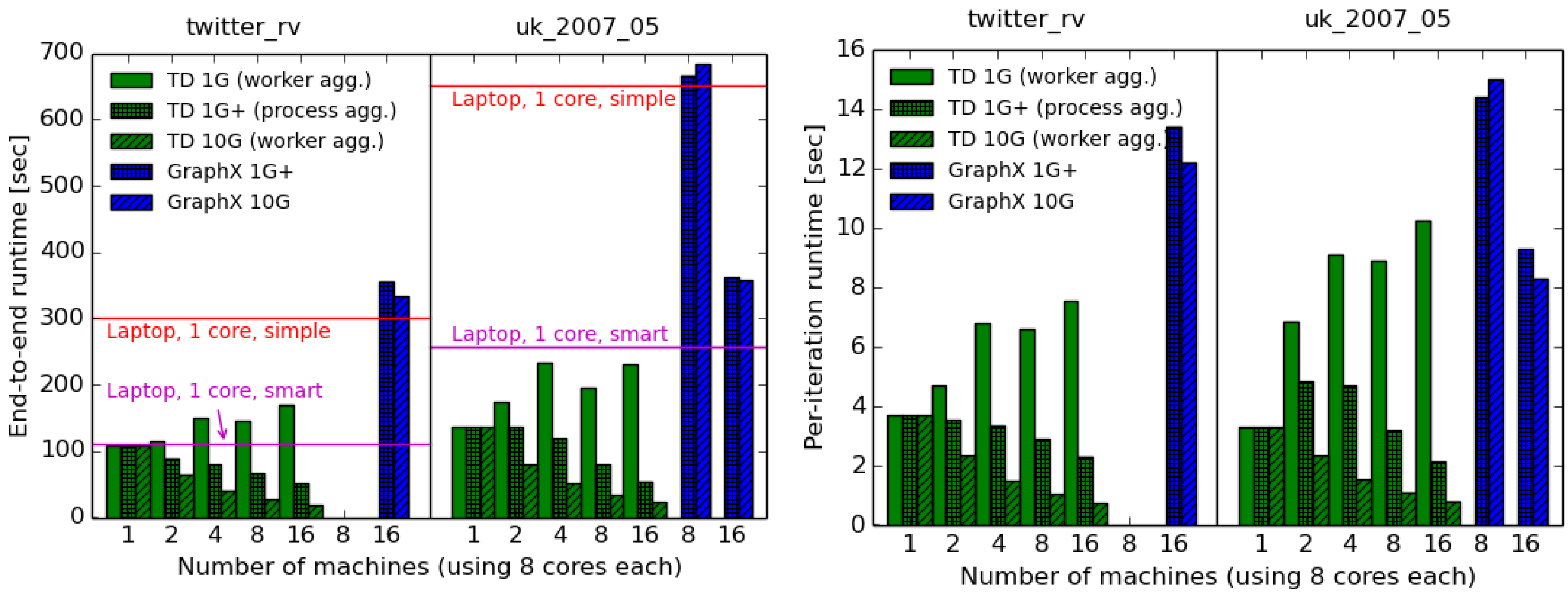




Twenty pagerank iterations on one machine, multiple threads.

System	cores	twitter_rv	uk_2007_05
Timely dataflow	1	350.7s (11.33s)	442.2s (8.90s)
Timely dataflow	2	196.5s (6.39s)	297.3s (5.67s)
Timely dataflow	4	182.4s (6.12s)	192.0s (3.78s)
Timely dataflow	8	107.6s (3.70s)	137.1s (3.29s)
Timely dataflow	12	95.0s (3.32s)	114.5s (2.65s)

Now you can have multiple …



Conclusions 1

- As we have seen, the three implementations (GraphX and the two timely dataflow ones) have *different bottleneck resources*.
- GraphX does more compute and is CPU-bound even on the 1G network, whereas the leaner timely dataflow implementations become CPU-bound only on the 10G network.
- Drawing conclusions about the scalability or limitations of either system based on the performance of the other is likely misguided.

Conclusions 2

- Fast 10G networks *do* help reduce reduce the runtime of parallel computations by significantly more than 2-10%: we've seen **speedups up to 3x going from 1G to 10G**.
- However, the structure of the computation and the implementation of the data processing system must be suited to fast networks, and different strategies are appropriate for 1G and 10G networks.
- For the latter, being less clever and

Conclusions 3

- Distributed data processing makes sense even for graph computations where the graph fits into one machine.
- When computation and communication are overlapped sufficiently, **using multiple machines yields speedups up to 5x** (e.g., on twitter_rv, 1x8 vs. 16x8). Running everything locally isn't necessarily faster.

Conclusions 4

- Can make PageRank run **16x faster per iteration using distributed timely dataflow** than using GraphX (from 12.2s to 0.75s per iteration).
- This tells us something about how much scope for improvement there is even over numbers currently considered state-of-the-art in research!

For more details&followup

- See
  http://www.cl.cam.ac.uk/research/srg/netos/camsas/blog/2015-07-08-timely-pagerank-part1.html

- Ack to Malte Schwarzkopf & Frank McSherry