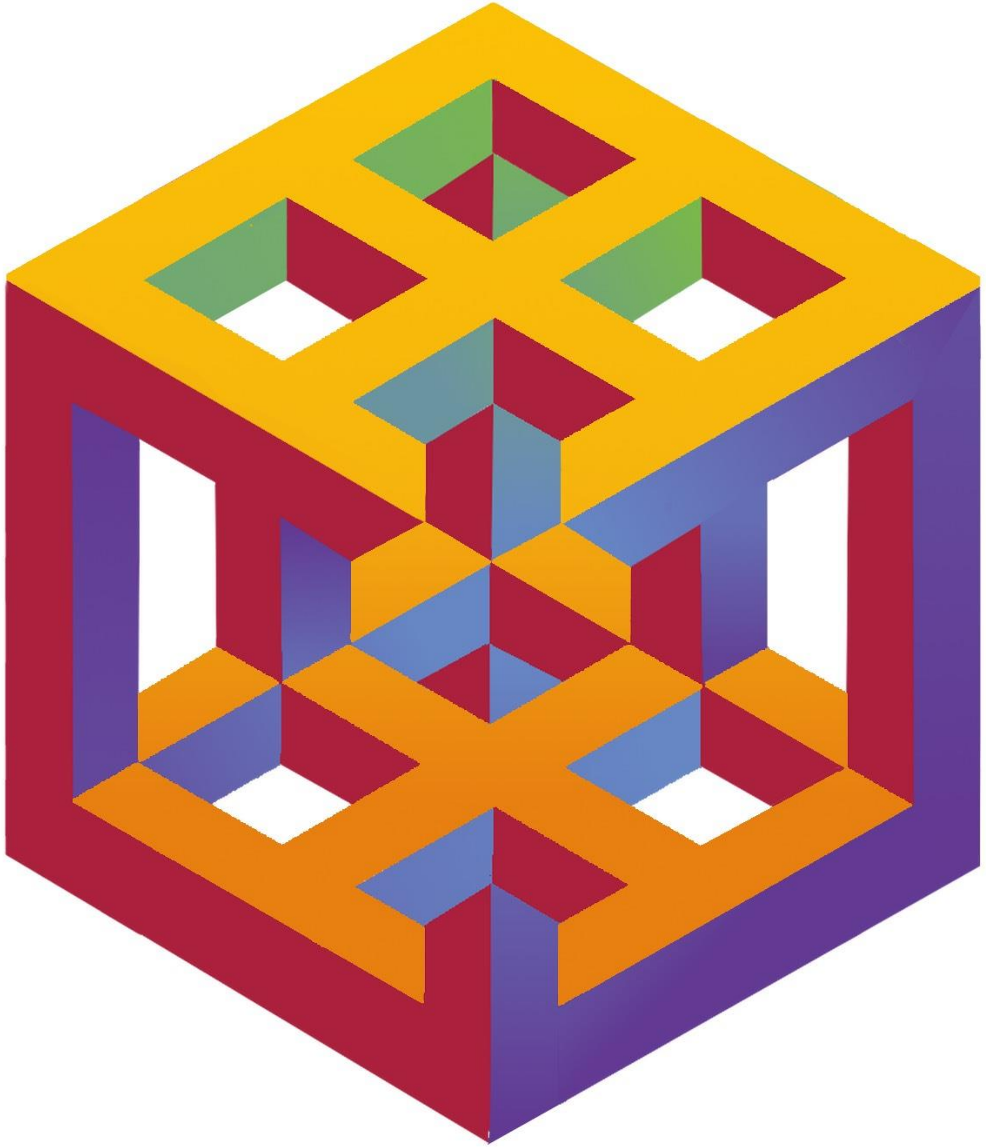
PARTISAN SCALING THE DISTRIBUTED ACTOR RUNTIME

Christopher S. Meiklejohn
Heather C. Miller
Peter Alvaro

Carnegie Mellon University
Baskin Engineering UC SANTA CRUZ
isr institute for SOFTWARE RESEARCH



MOTIVATION

Distributed systems programming is still very hard:

- How to manage state?
- How do we manage concurrency?
- How do we leverage parallelism?

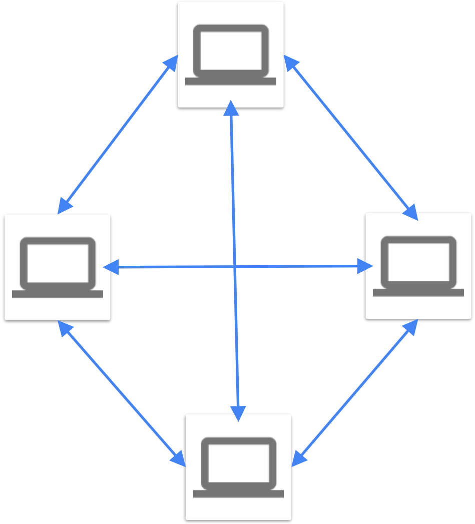
Distributed actors are good! (and, a good match to distributed systems, too!)

- Encapsulation for state
- Pervasive concurrency – thousands of actors working together
- Asynchronous messaging – no shared memory between actors
- Transparent messaging and serialization – easy programming model!

Demonstrated success:

- Erlang: Call of Duty, League of Legends, WhatsApp
- Orleans: Halo, Gears of War

DISTRIBUTED ACTORS: TODAY'S DRAWBACKS



Scalability

- All-to-all communication is expensive and prohibitive
- Nodes need to know about all other nodes

Latency

- Multiplexed TCP connection is a bottleneck
- Many actors reduced to a single connection's speed
- Congestion:
  - network latency, queueing delay
- Contention:
  - competing for shared resources, slow-sender vs. fast-sender

PARTISAN

Design of an alternative runtime system for distributed actor systems

- Design and prototype implementation in Erlang

Runtime selection of communications overlay network

- Specialize overlay selection to communications pattern of application
- No modification to application code

Provides reduced latency and increased scalability

- Enable parallelism on the network
- Schedule messages efficiently on the network

Results:

- Order of magnitude increase in cluster size
- Up to 13.5x reduction in latency and 38.07x increase in throughput

Come to our talk!
11:20 AM, Track 2: Runtimes
July 10th