Paths to Fast Barrier Synchronization on the Node

PARAGON
PARALLEL ARCHITECTURE GROUP @NORTHWESTERN

Northwestern University

PRINCETON UNIVERSITY

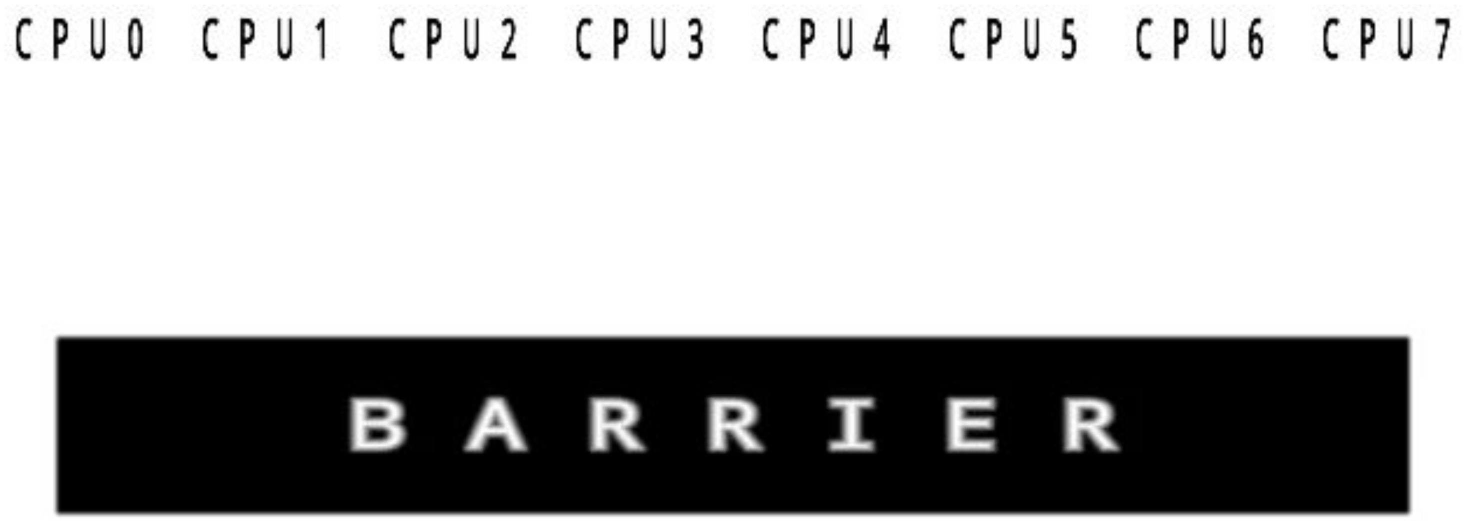
Plab

Conor Hetland Georgios Tziantzioulis Brian Suchy
Michael Leonard Jin Han John Albers
Nikos Hardavellas Peter Dinda

Summary

- Today’s software barriers are slow
- Barrier latency matters
- Intel HARP hardware barrier implementation
- Proposal for minimally invasive hardware barriers on x86
- Speculation on fast barriers on silicon photonic hardware

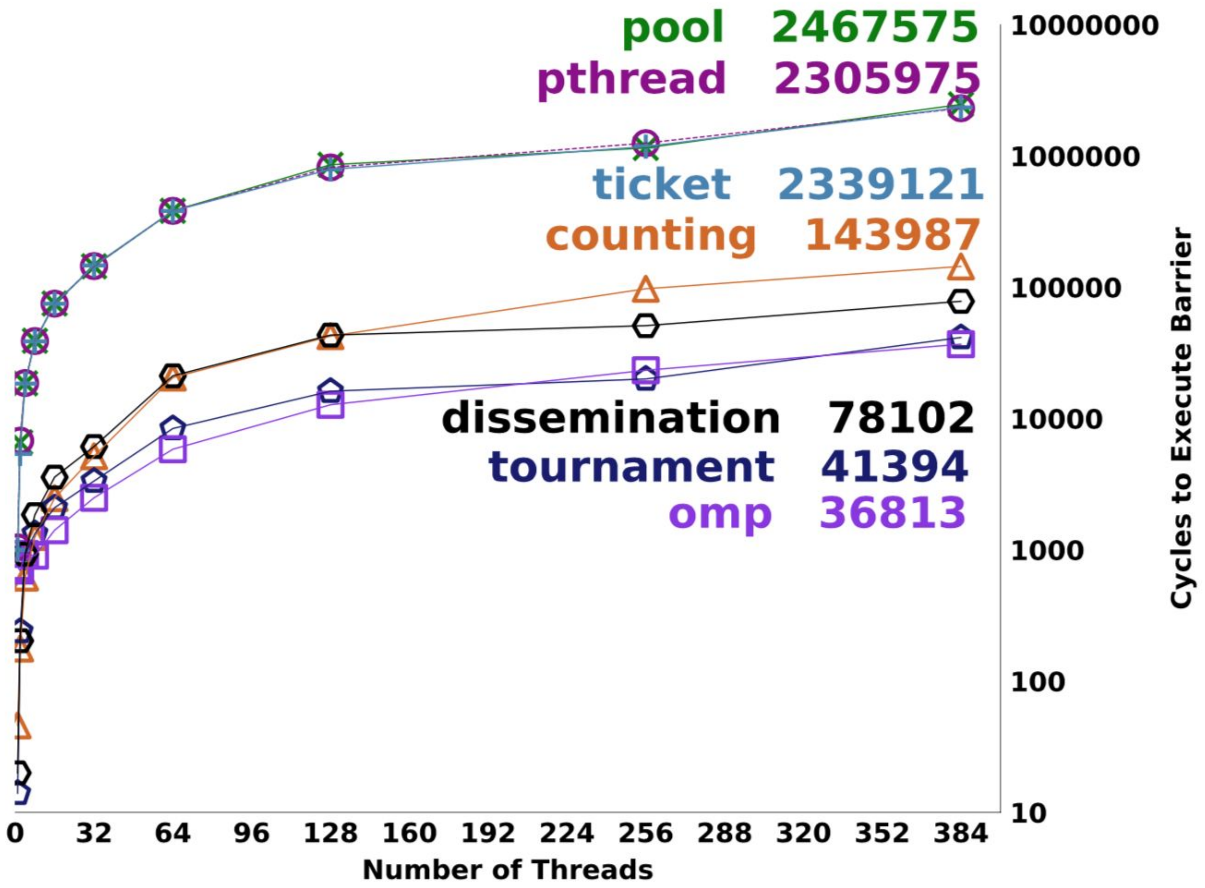
What Kinds of Barriers do We Care About?



Current Barriers Are Slow: Benchmarks

Huge, 8-socket machine:

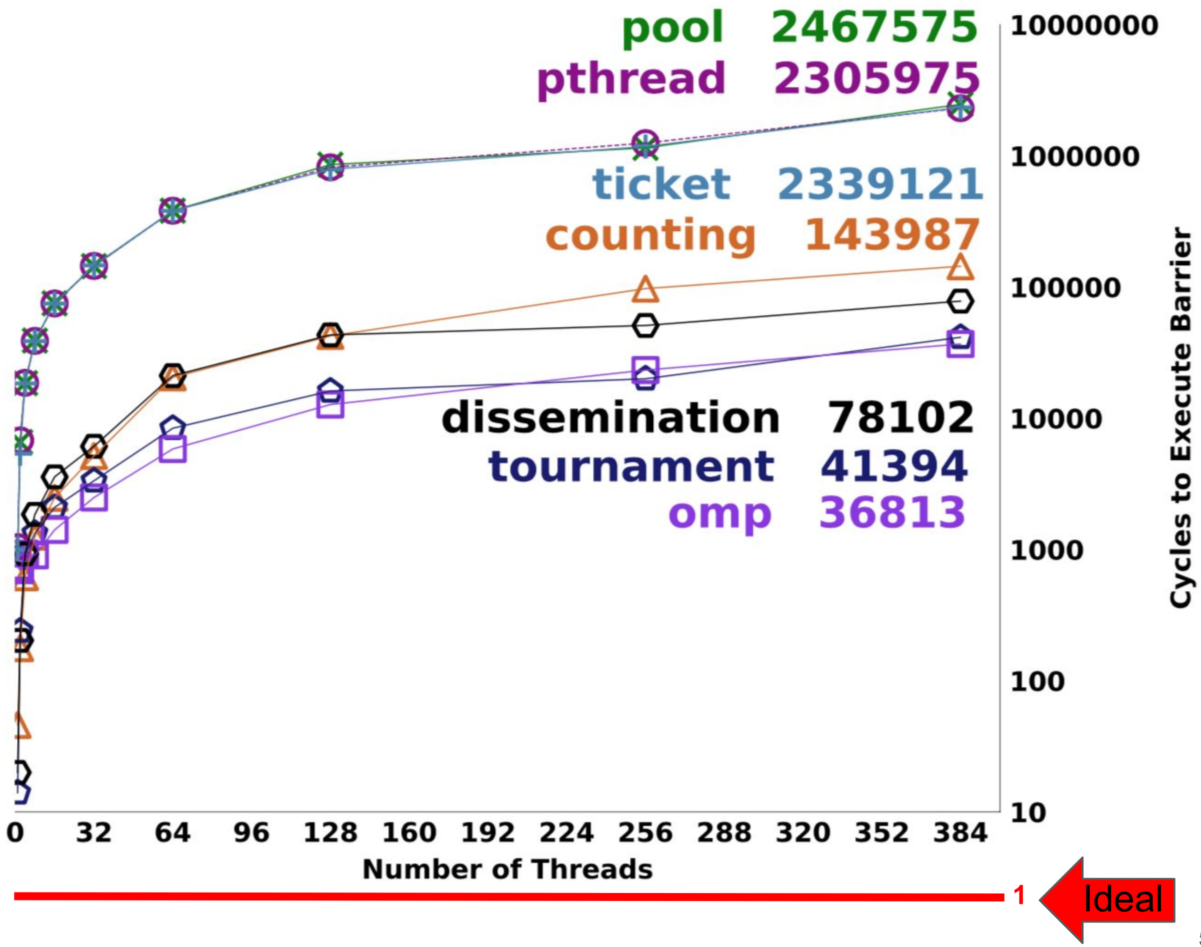
- Supermicro 7089P-TR4T
- eight 24 core, hyperthreaded 2.1 GHz Intel Xeon Platinum 8160 processors (384 hardware threads total)
- 768 GB of RAM split among 8 NUMA zones.



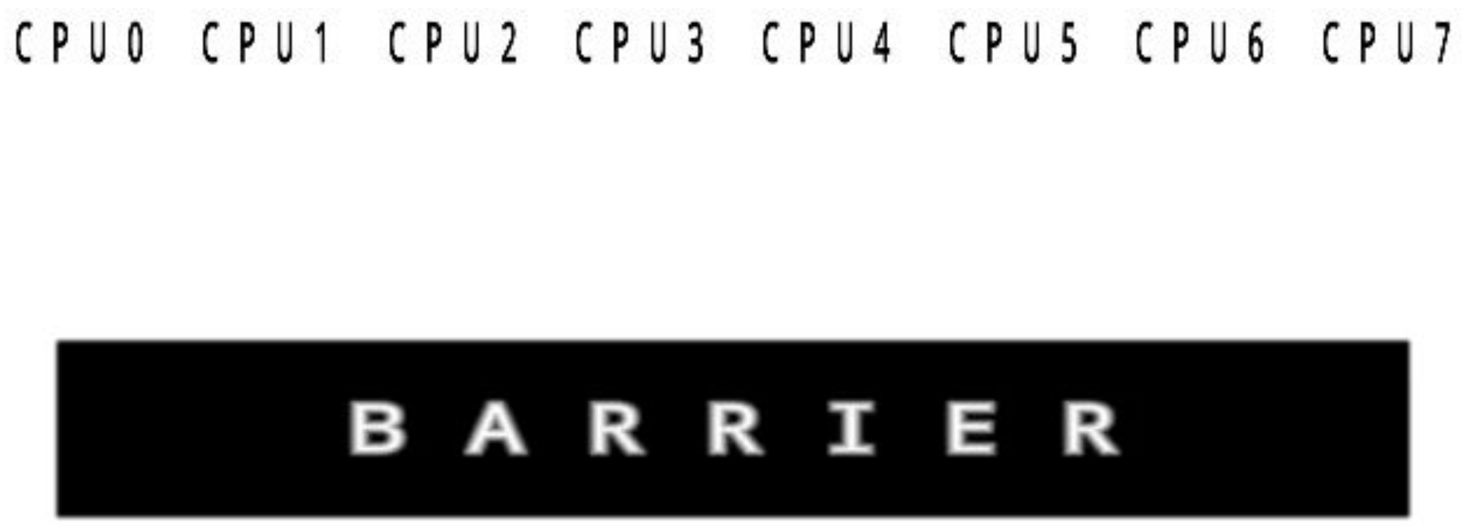
Current Barriers Are Slow: Benchmarks

Huge, 8-socket machine:

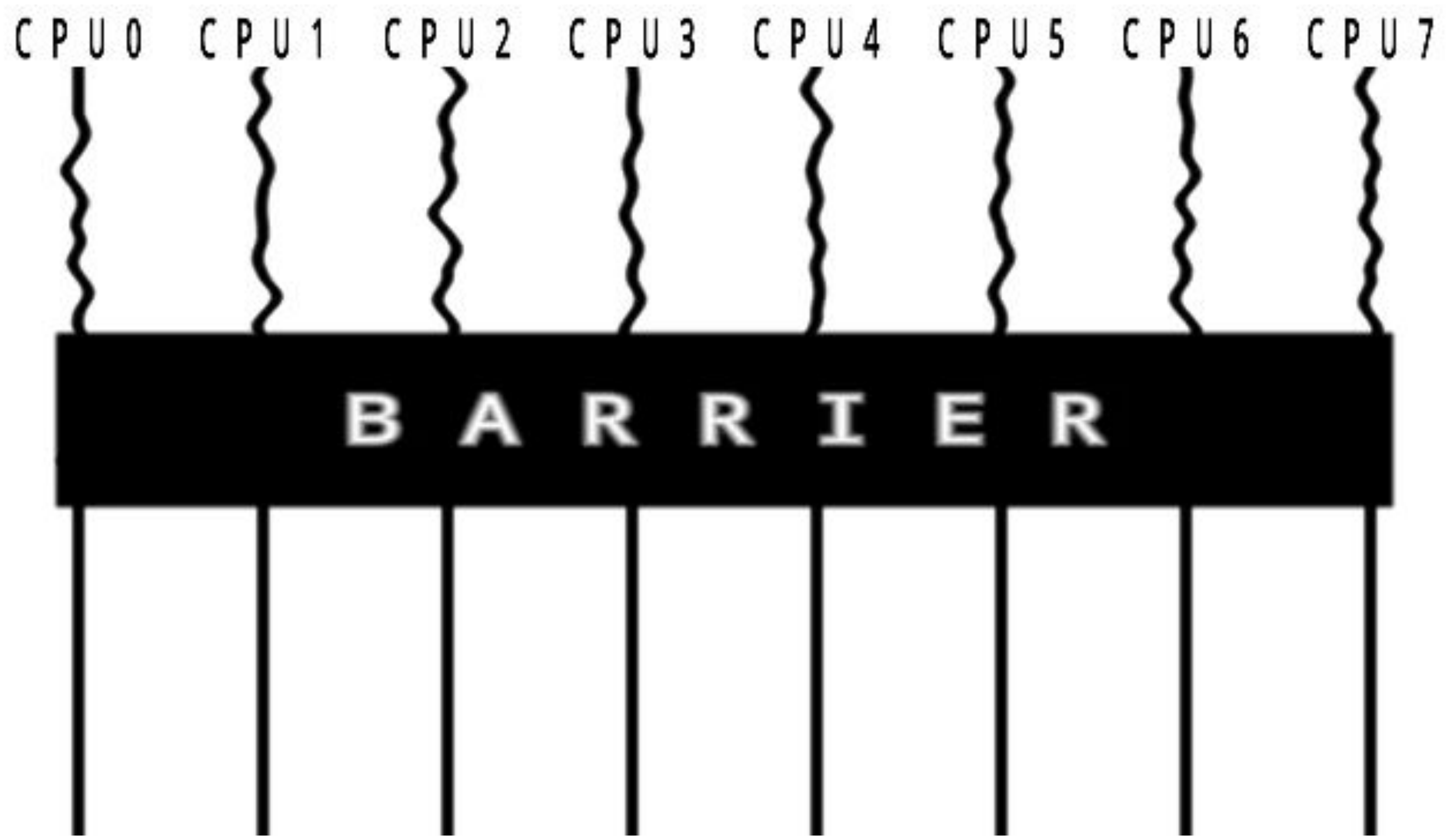
- Supermicro 7089P-TR4T
- eight 24 core, hyperthreaded 2.1 GHz Intel Xeon Platinum 8160 processors (384 hardware threads total)
- 768 GB of RAM split among 8 NUMA zones.



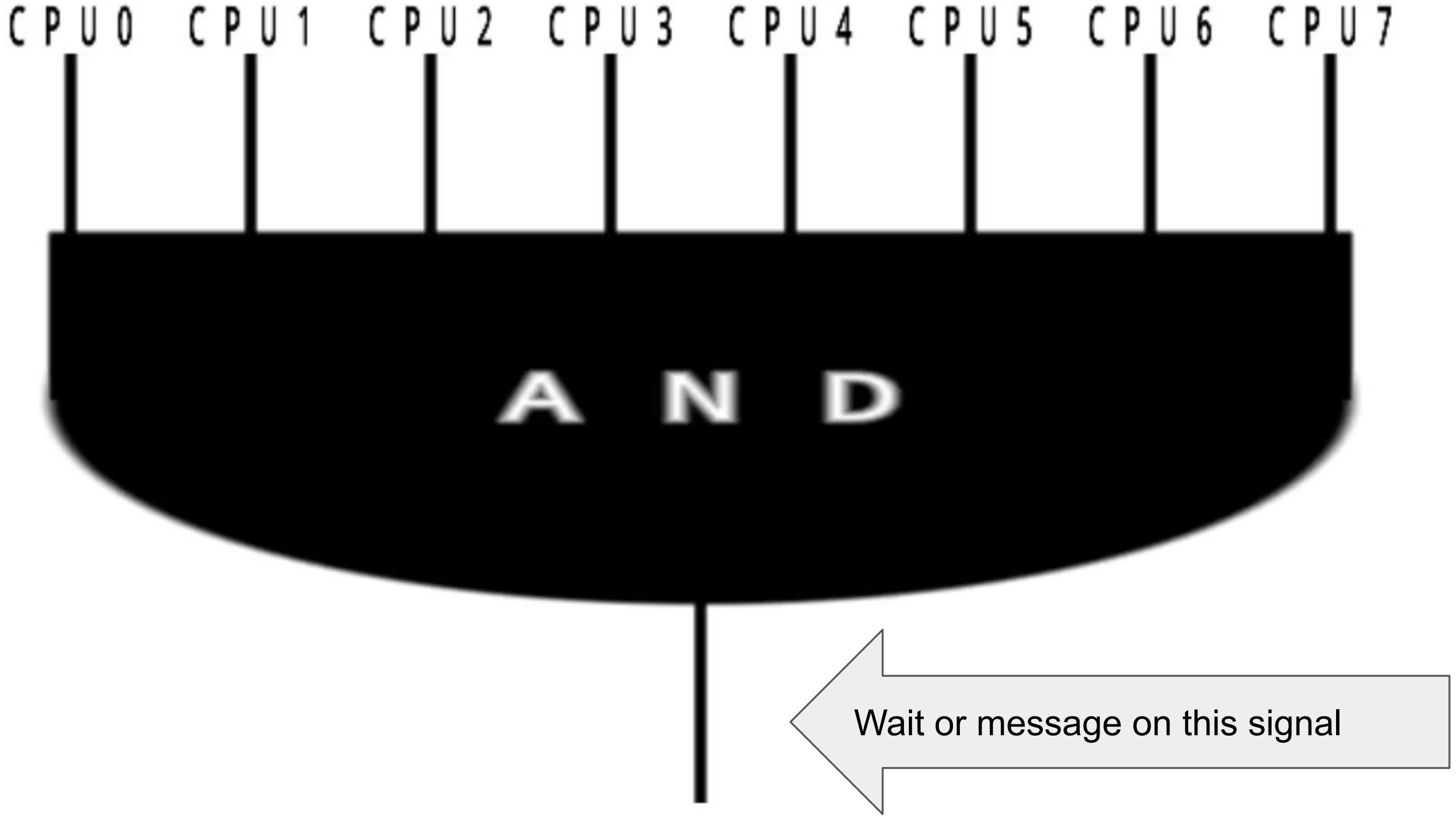
Reminder



The AND Gate: Reduced Barrier



The AND Gate: Reduced Barrier



Summary

- Today’s software barriers are slow
- Barrier latency matters
- Intel HARP hardware barrier implementation
- Proposal for minimally invasive hardware barriers on x86
- Speculation on fast barriers on silicon photonic hardware

NESL

- Nested data parallel runtime
- Operates on collections using abstract vector instructions
- Requires multiple barriers per abstract vector instruction

Blelloch, G.E., Chatterjee, S., Hardwick, J., Sipelstein, J., and Zagha, M. Implementation of a portable nested data-parallel language. JPDC ‘94

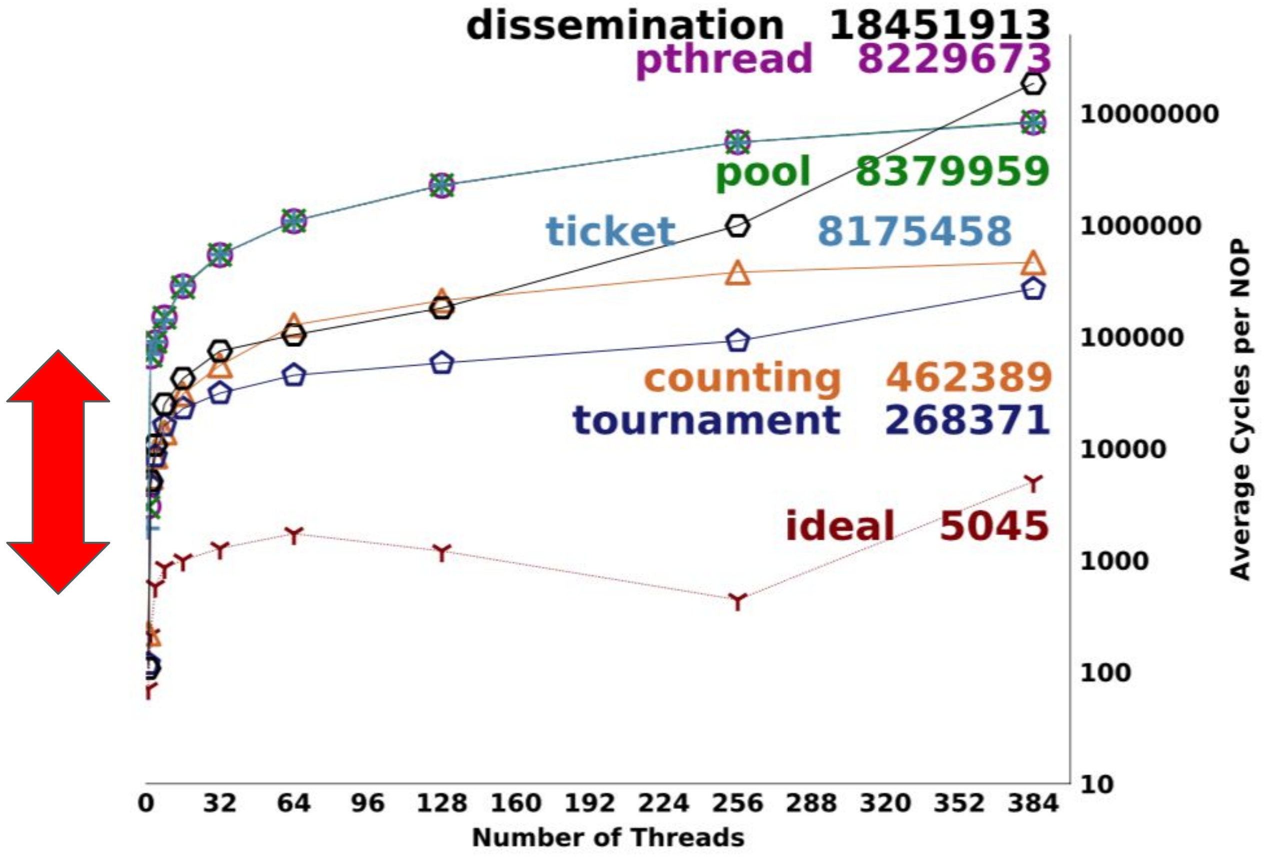
Bergstrom, L., Fluet, M., Rainey, M., Reppy, J., Rosen, S., and Shaw, A. Data-only flattening for nested data parallelism. PPoPP ‘13

In our impl every thread does this:

```
while(!done) {

    pc = update_pc();

    pc_agreement_barrier();

    decode(pc);

    decode_agreement_barrier();

    execute_and_writeback(pc);

    writeback_agreement_barrier();

}
```

Barrier Speed Matters: NESL (VCODE) Interpreter

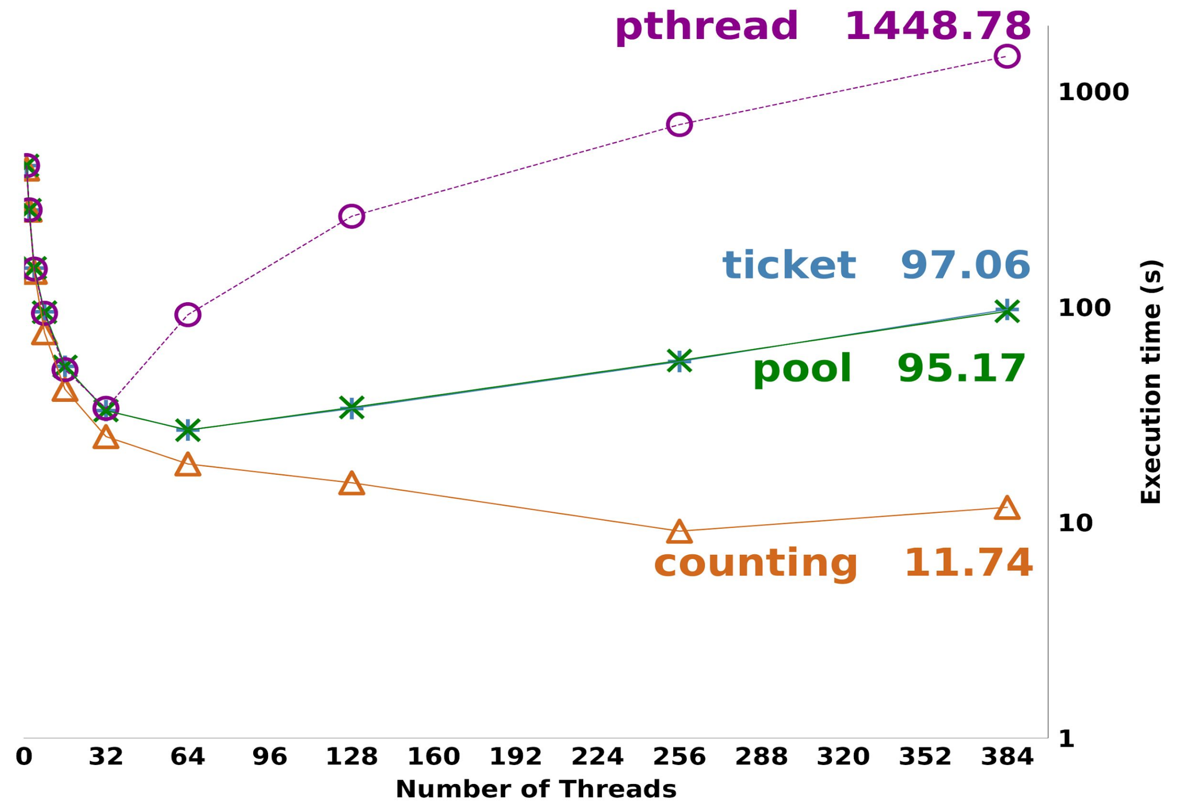
Performance with ideal barrier would be 2 orders of magnitude better



Barrier Speed Matters: PARSEC Streamcluster

Better barriers enable finer grain, better scaling

Plenty of room for improvement



Similar Results Across All Benchmarked Machines

NUMA-8 (previous graphs):

- Supermicro 7089P-TR4T
- eight 24 core, hyperthreaded 2.1 GHz Intel Xeon Platinum 8160 processors (384 hardware threads total)
- 768 GB of RAM split among 8 NUMA zones.

NUMA-4:

- Dell R815
- four 16 core 2.1 GHz AMD Opteron 6272 processors
- 128 GB of RAM split among 4 NUMA zones.

Similar Results Across All Benchmarked Machines

Xeon Phi:

- Essentially a Supermicro 5038ki, and includes a Intel Xeon Phi 7210 processor running at 1.3 GHz.
- 64 cores, each of which has 4 hardware threads
- 16 GB of MCDRAM, and more loosely to 96 GB of conventional DRAM.

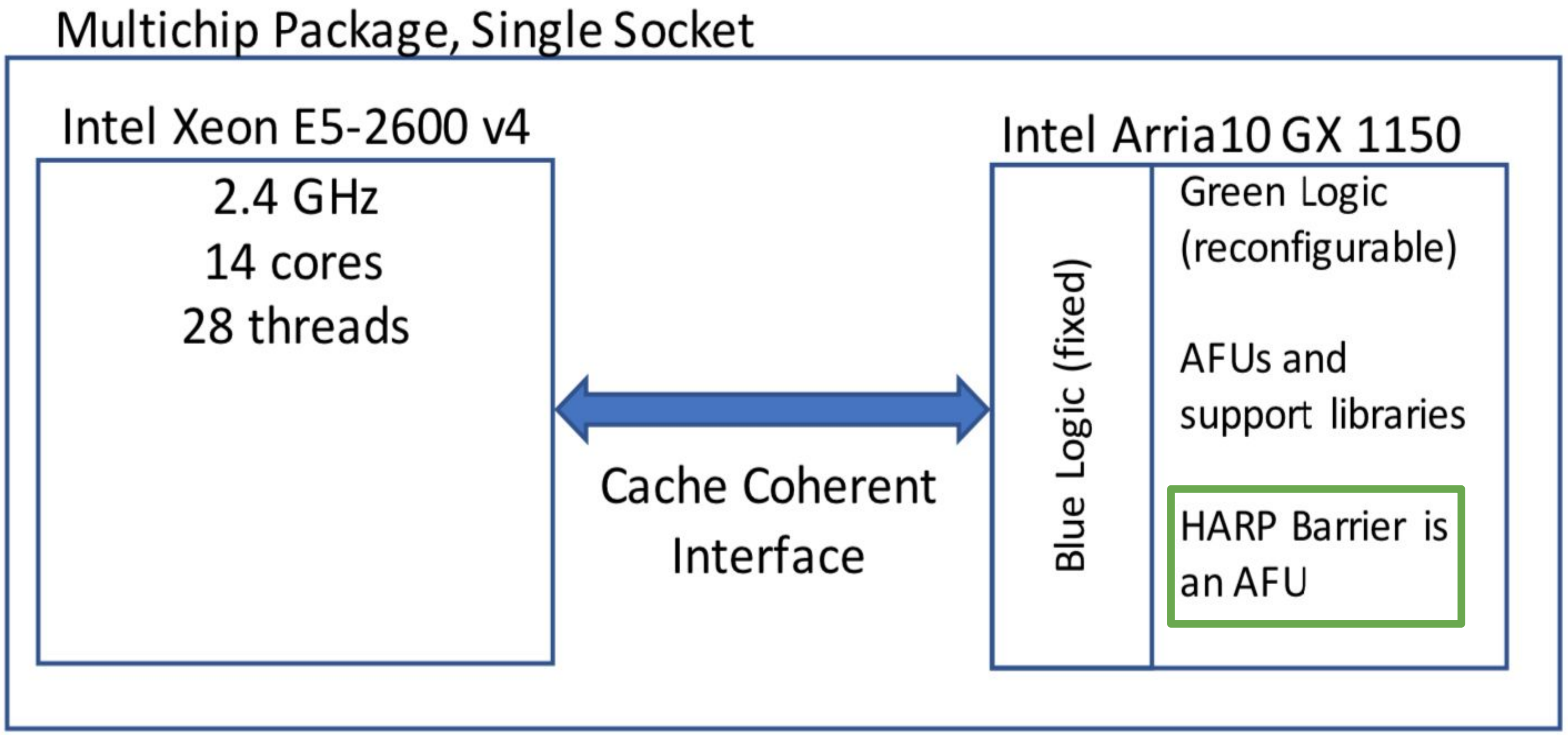
HARP:

- **prototype Intel platform integrates a Broadwell Xeon processor and a large FPGA**
- **14 cores, each of which has 2 hardware threads**

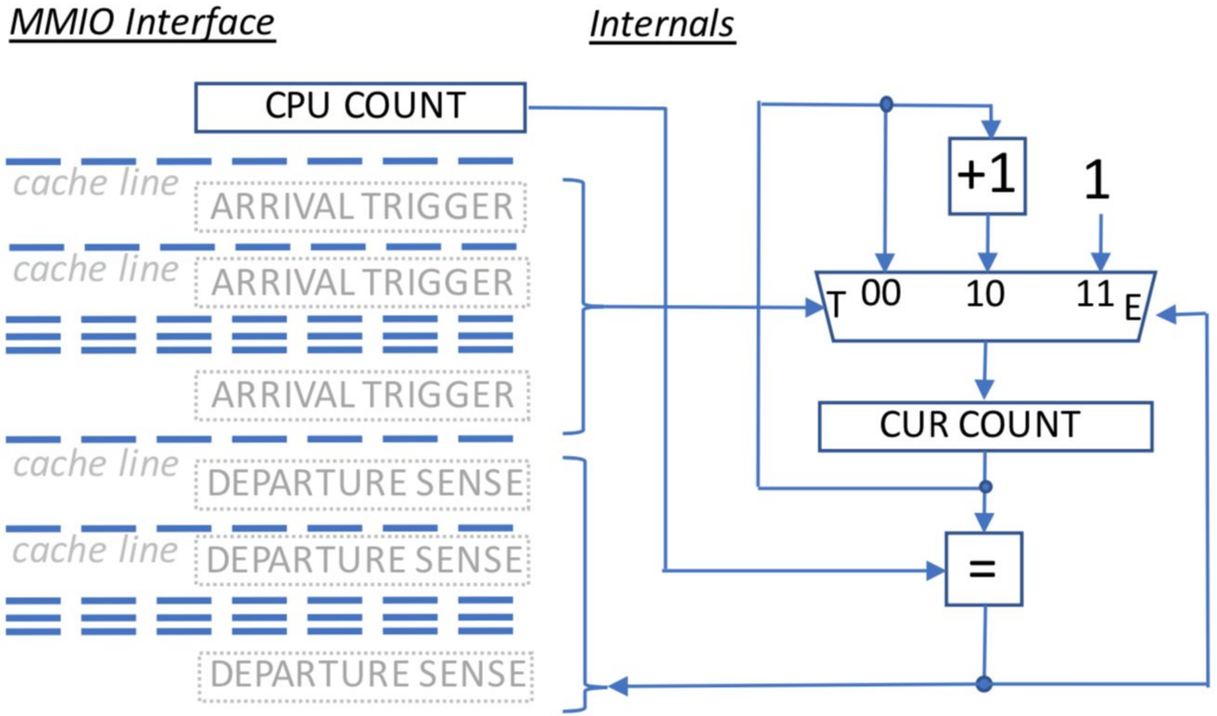
Summary

- Today’s software barriers are slow
- Barrier latency matters
- Intel HARP hardware barrier implementation
- Proposal for minimally invasive hardware barriers on x86
- Speculation on fast barriers on silicon photonic hardware

The HARP


Multichip Package, Single Socket
Intel Xeon E5-2600 v4
2.4 GHz
14 cores
28 threads
Cache Coherent Interface
Intel Arria10 GX 1150
Blue Logic (fixed)
Green Logic (reconfigurable)
AFUs and support libraries
HARP Barrier is an AFU

Our Own Hardware Barrier



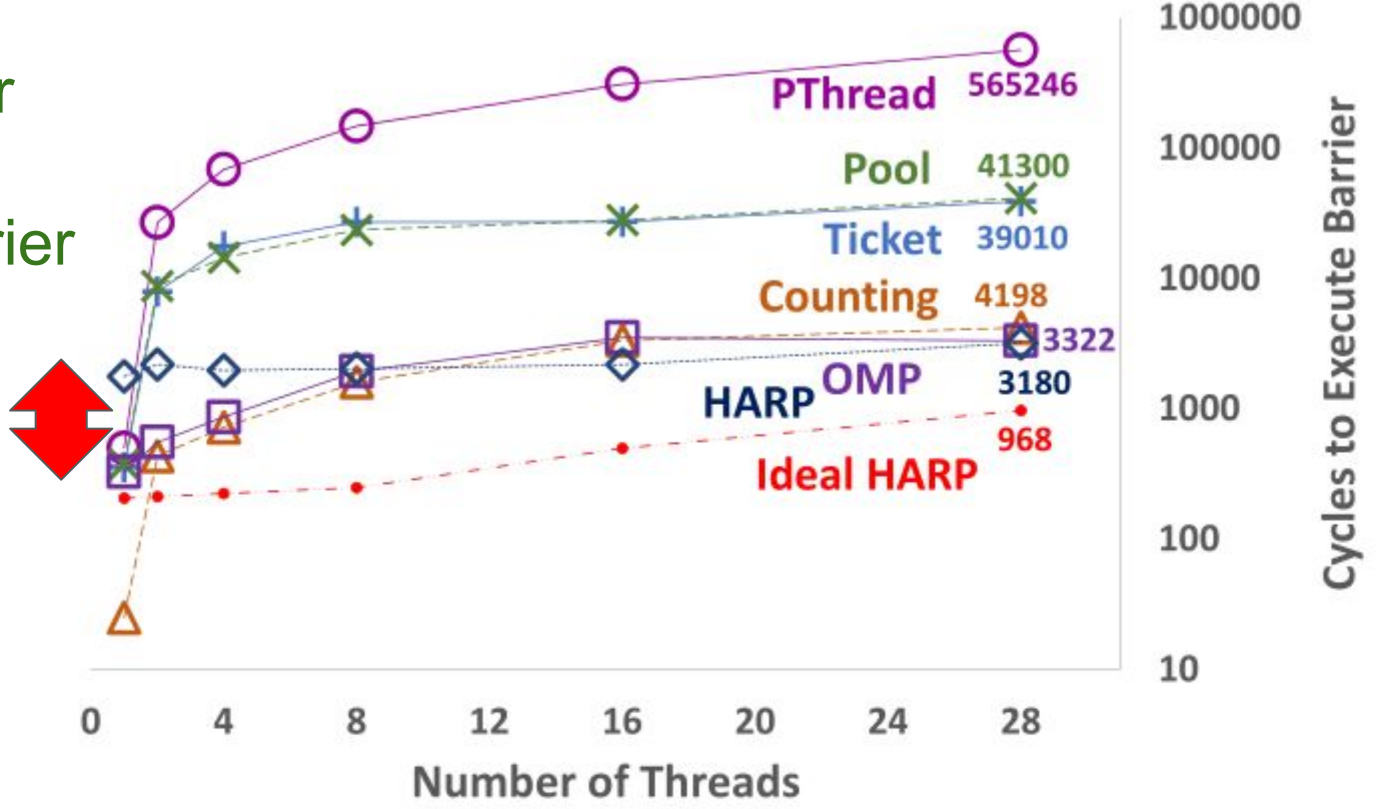
Each hardware thread has a private, cacheline-separated arrival and departure interface to allow for maximum read/write parallelism in the CPU/FPGA interface

TE = 00 Idle
TE = 10 Arrival Trigger, Current Round
TE = 11 Arrival Trigger, Next Round
Reset on write to CPU COUNT (CUR COUNT set to 0)

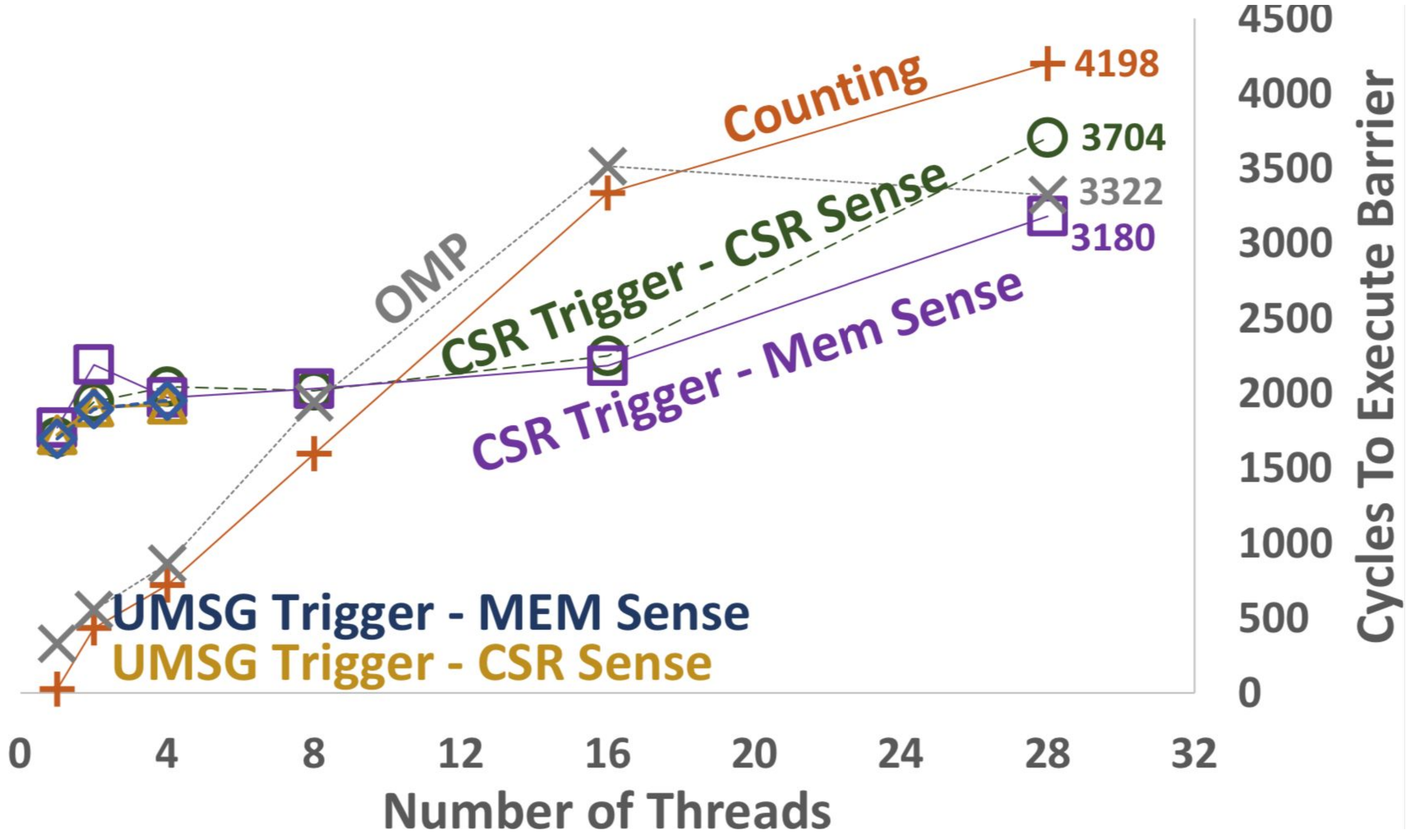
Performance

Slightly faster than best software barrier

Performance limited by interconnect latency



Extensive Exploration of Communication



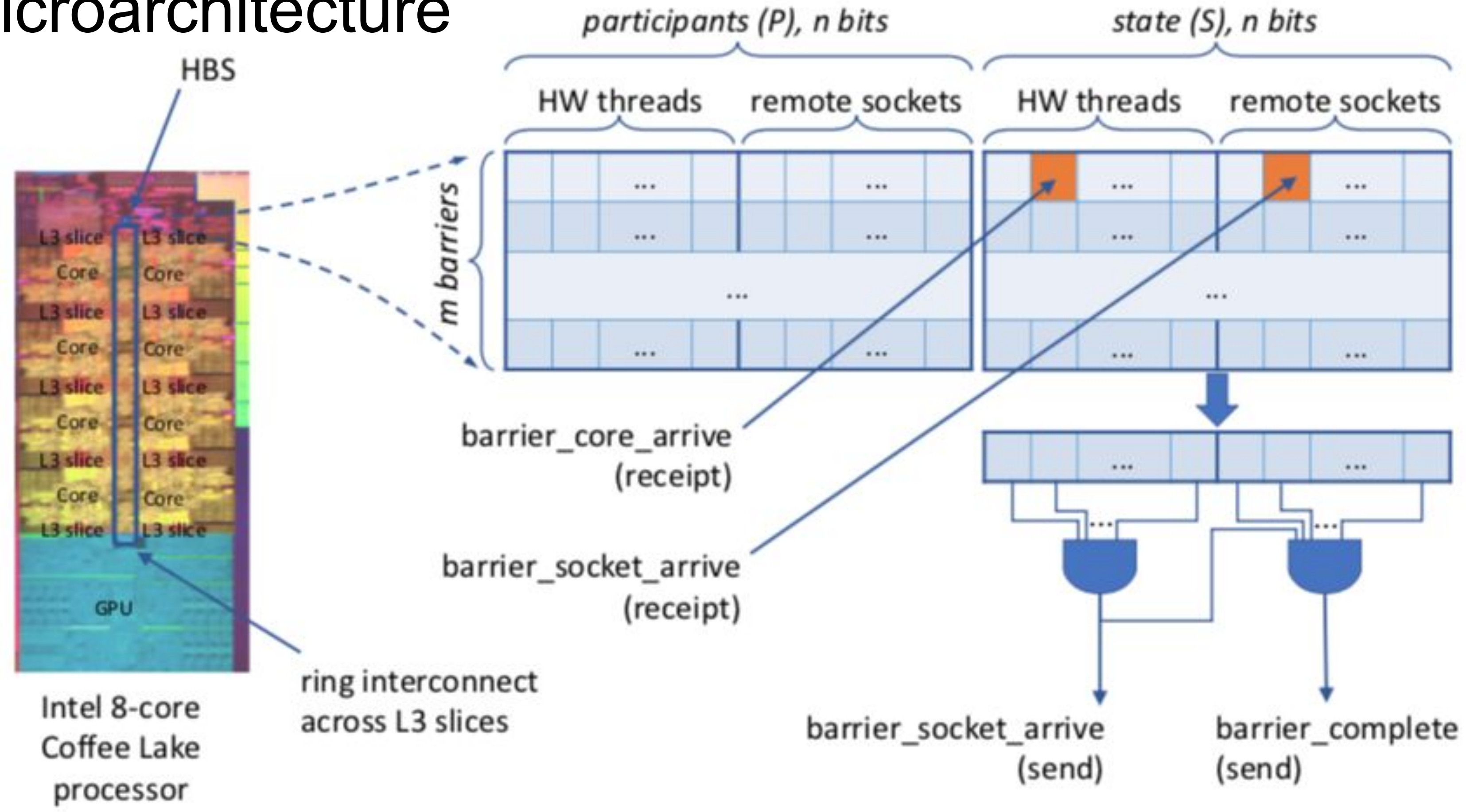
Summary

- Today’s software barriers are slow
- Barrier latency matters
- Intel HARP hardware barrier implementation
- Proposal for minimally invasive hardware barriers on x86
- Speculation on fast barriers on silicon photonic hardware

Proposed ISA Extension

- Only two proposed instructions:
  - barinit %rax (privileged)
  - barwait %rax (unprivileged)
- MSRs
  - Multiple subset barriers
  - Timeout for protection
- barinit would be wrapped in a syscall with a timeout
  - **`int bar = create_barrier(thread_list, timeout);`**
- Minimally invasive and secure

Microarchitecture


HBS
L3 slice
Core
GPU
Intel 8-core Coffee Lake processor
ring interconnect across L3 slices
participants (P), n bits
state (S), n bits
HW threads
remote sockets
HW threads
remote sockets
m barriers
barrier_core_arrive (receipt)
barrier_socket_arrive (receipt)
barrier_socket_arrive (send)
barrier_complete (send)

Performance Analysis

- A 4-socket system with 28 cores per socket with could implement support for 128 simultaneous subset barriers using only 944 bytes of storage
- Latency is cost of an L3 access (20-40 cycles) plus a round trip around the socket interconnect (370 cycles on hyper transport or 440 cycles with QPI on a 4 socket machine)

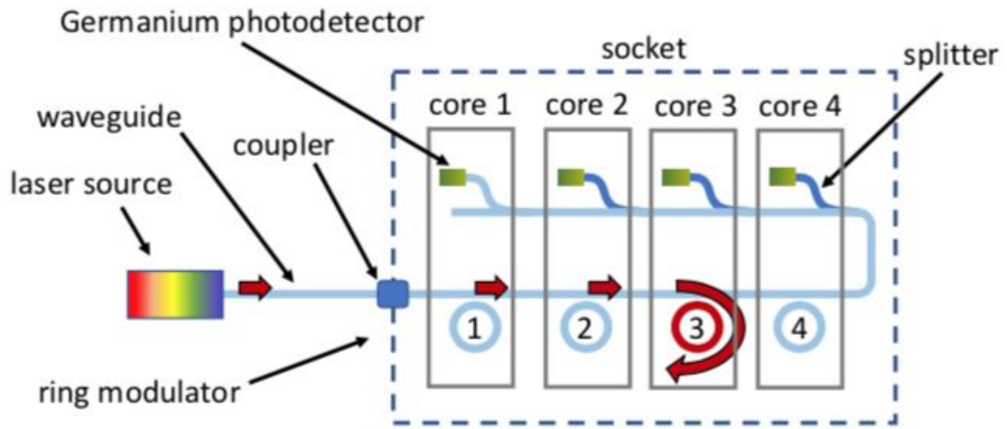
Summary

- Today’s software barriers are slow
- Barrier latency matters
- Intel HARP hardware barrier implementation
- Proposal for minimally invasive hardware barriers on x86
- Speculation on fast barriers on silicon photonic hardware

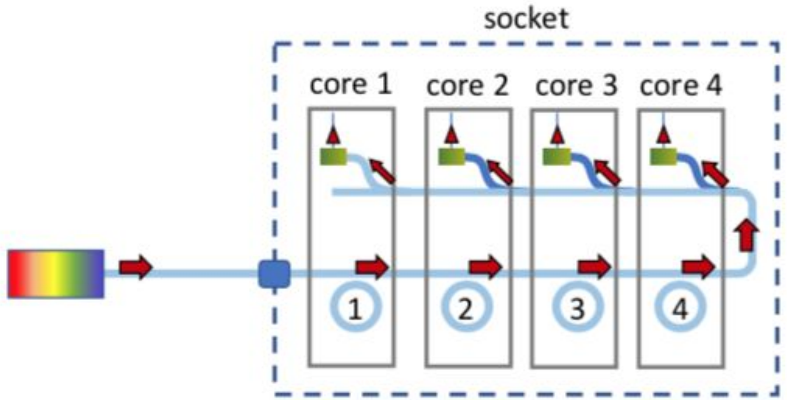
Silicon Photonic Barrier Design

Optical interconnects provide barrier functionality essentially for free

Binkert, N., Davis, A., Lipasti, M., Schreiber, R. S., and Vantrese, D. Nanophotonic barriers. PICA '09

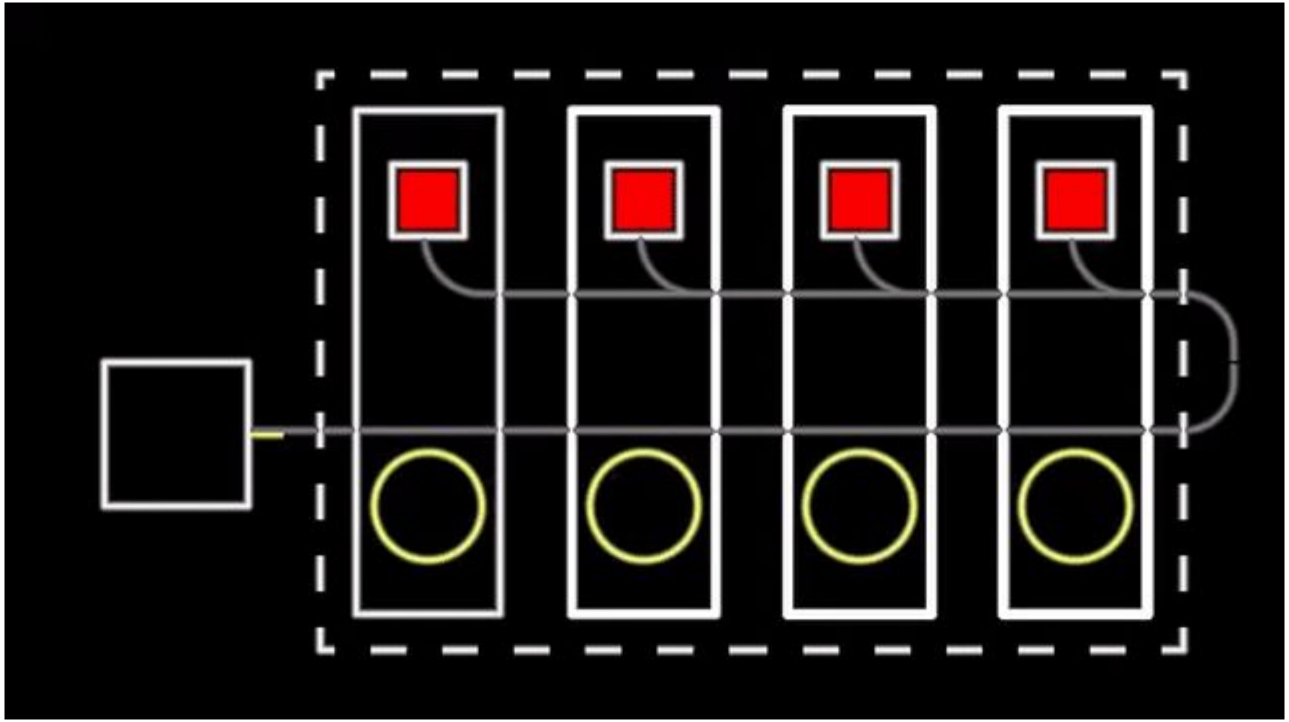


(a) Core 3 has not arrived at the barrier



(b) All cores have arrived at the barrier

Silicon Photonic Barrier Animated



Related Work

- Abellan, J.L., Fernandez, J., and Acacio, M.E. A g-line-based network for fast and efficient barrier synchronization in many-core cmps ICPP 2010
- Classic and modern distributed memory parallel machines such as the Cray T3E, Thinking Machines CM5, Ultracomputer, iWarp, and Blue Gene/L
- Purdue PAPERS

Future Work

- Skylake version of the HARP
- gem5 simulation of x86 proposal
- Other OS and runtime acceleration ideas:
  - ‘Functional’ page tables
  - transparent huge pages
  - kernel same-page merging

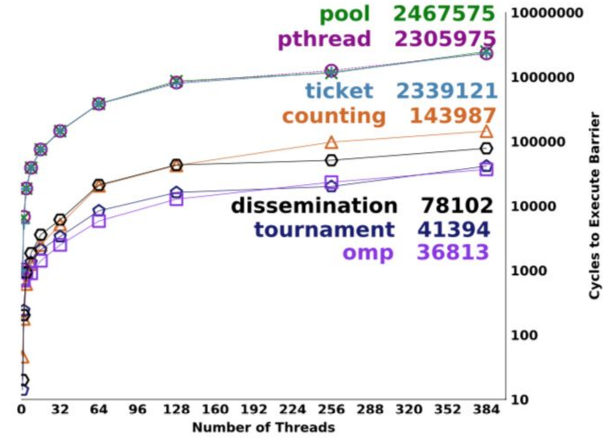
For More Information

- Conor Hetland – conorhetland2015@u.northwestern.edu
- Prescience Lab – http://presciencelab.org
- Acknowledgements – NSF, DOE, Intel

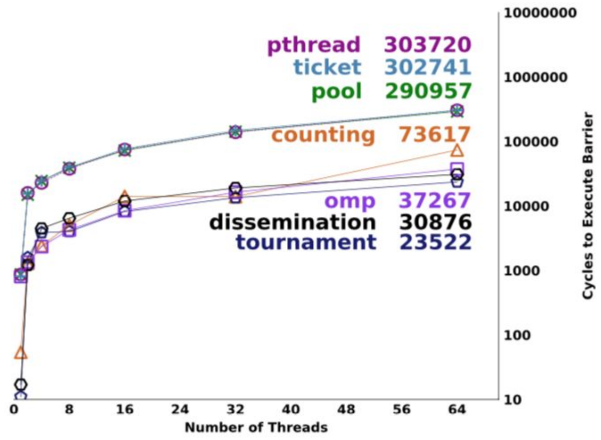
PARAG@N
PARALLEL ARCHITECTURE GROUP @NORTHWESTERN

Plab

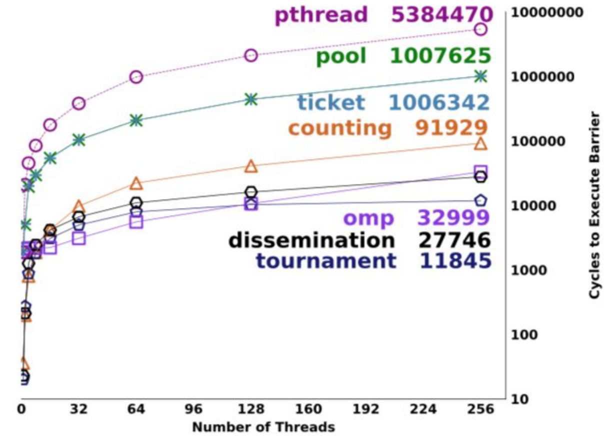
Latency



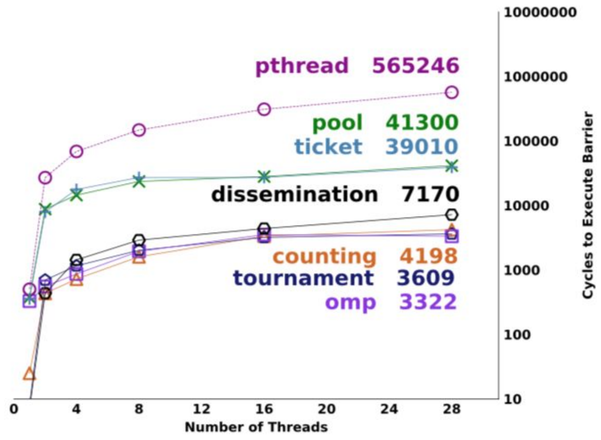
(a) NUMA (8 socket Intel)



(b) NUMA (4 socket AMD)

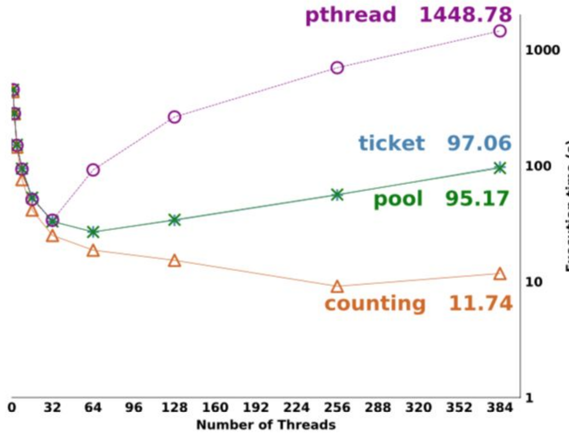


(c) Phi KNL

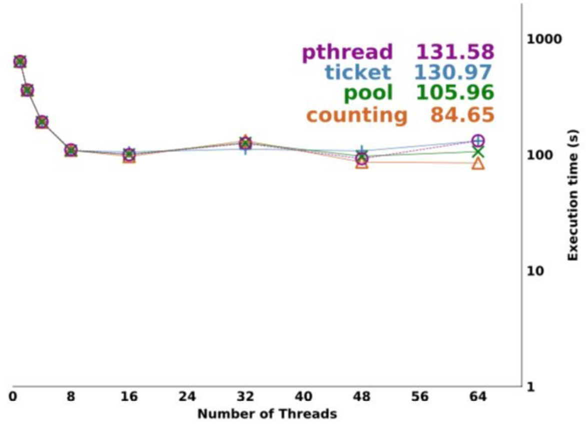


(d) HARP

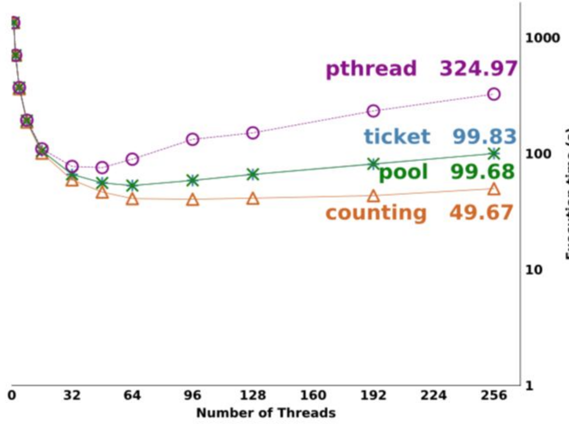
Streamcluster



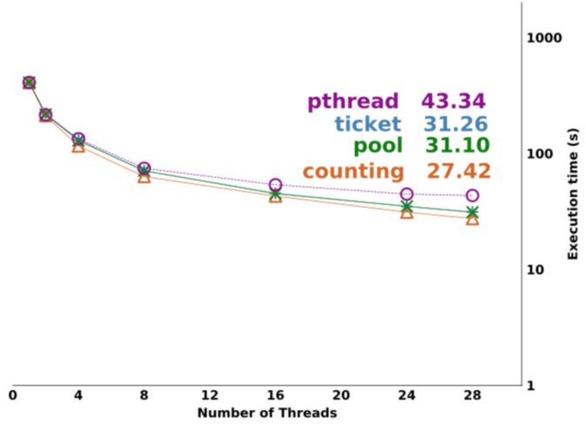
(a) NUMA (8 socket Intel)



(b) NUMA (4 socket AMD)

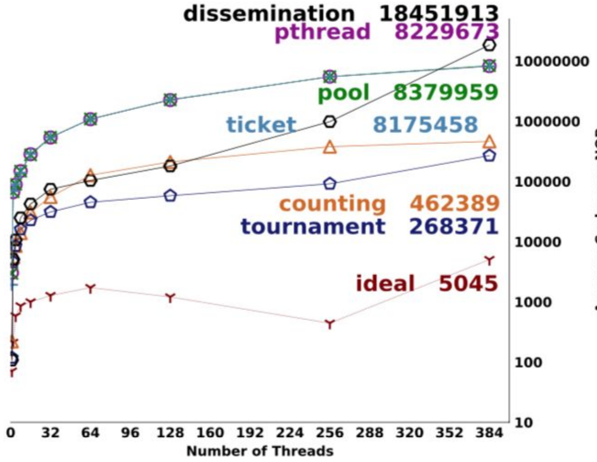


(c) Phi KNL

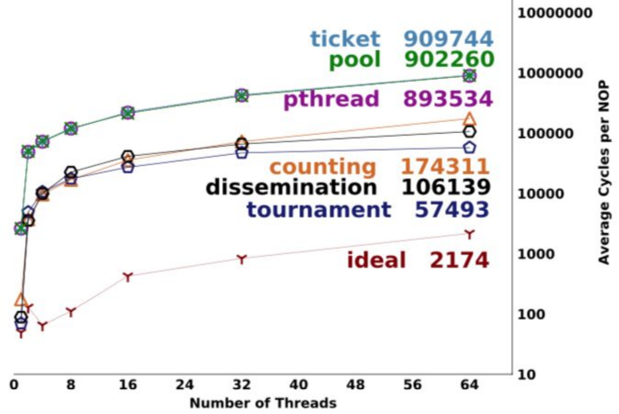


(d) HARP

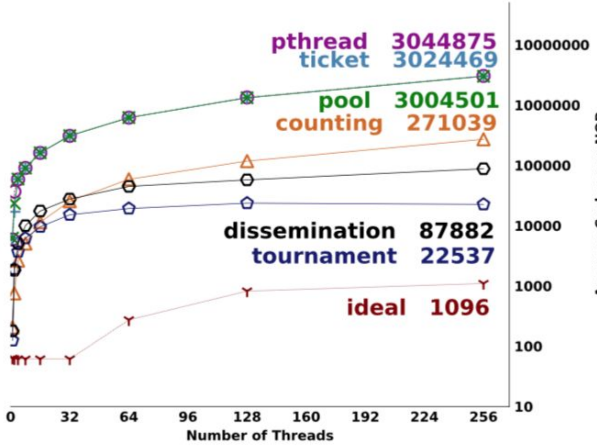
NESL



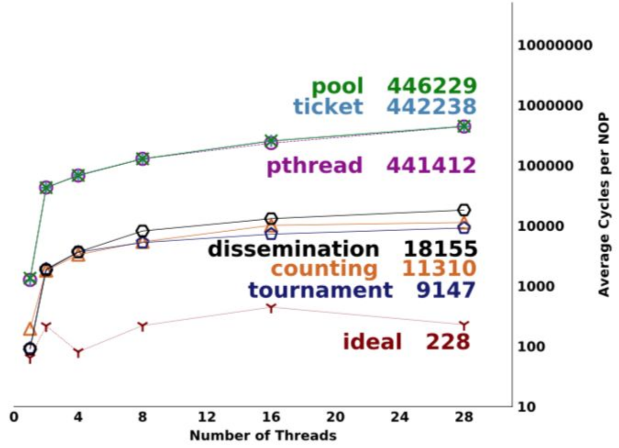
(a) NUMA (8 socket Intel)



(b) NUMA (4 socket AMD)



(c) Phi KNL



(d) HARP