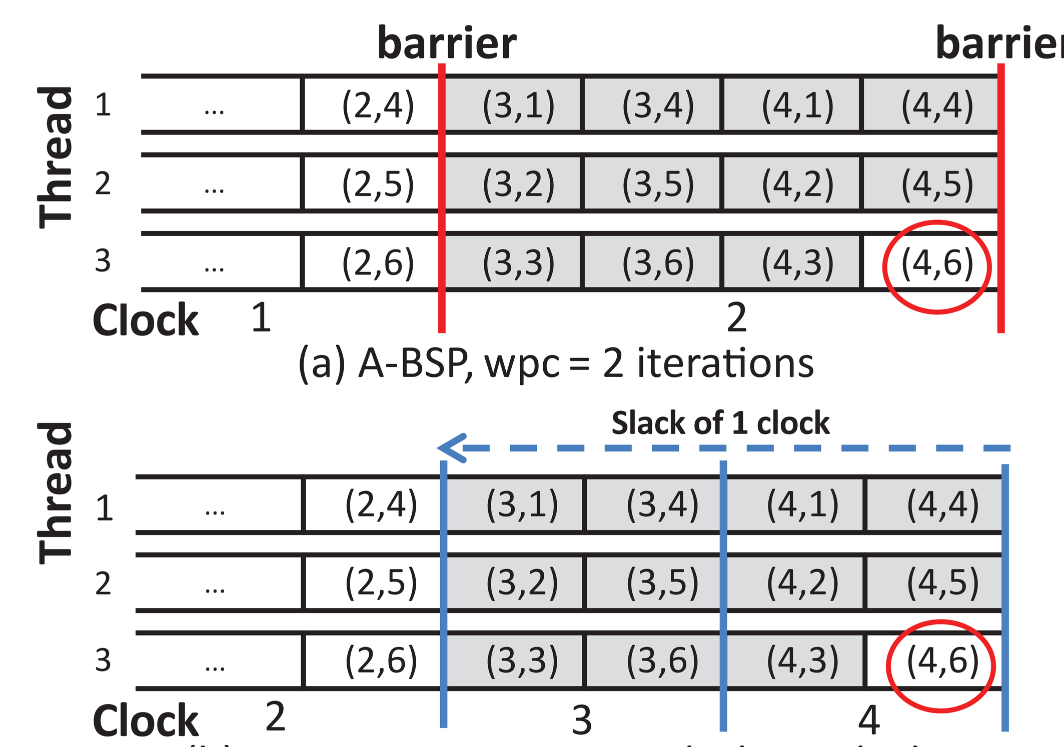
Exploiting Bounded Staleness To Speed Up Big Data Analytics

Henggang Cui, James Cipar, Qirong Ho, Jin Kyu Kim, Abhimanu Kumar, Seunghak Lee, Wei Dai, Jinliang Wei,
Greg Ganger, Phil Gibbons*, Garth Gibson, Eric Xing (CMU, *Intel)

BOUNDED STALENESS IN PARALLEL ML

- Can tune staleness of shared data
- Arbitrarily-sized Bulk Synchronous Parallel (A-BSP)
  - A barrier every some amount of work (a clock)
  - BSP with tunable "work per clock" (WPC)
- Stale Synchronous Parallel (SSP)
  - Fastest worker ≤ "slack" clocks ahead of slowest
  - Tunable "slack" (see LazyTables poster)
- Data staleness bound for "{wpc, slack}"
  - wpc x (slack + 1)
  - SSP {wpc, slack} == A-BSP {wpc x (slack + 1), 0}



(a) A-BSP, wpc = 2 iterations

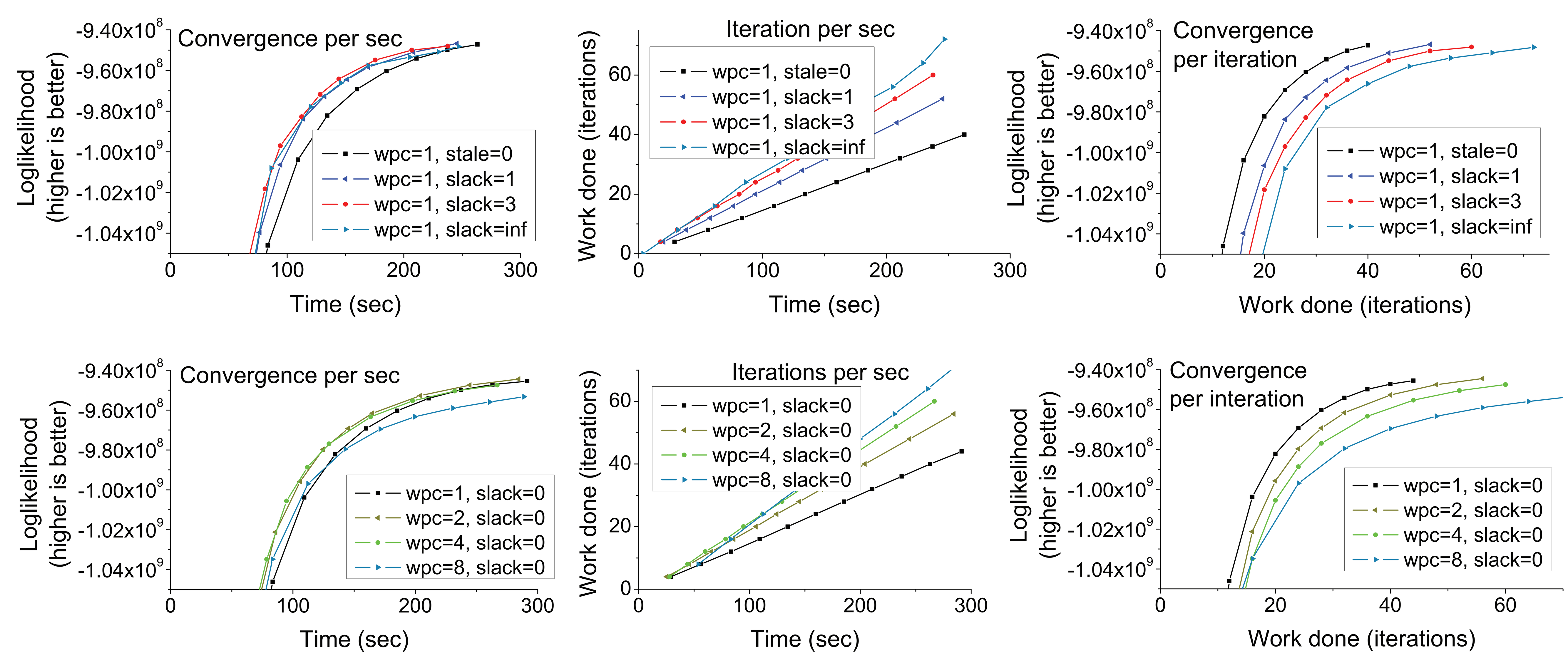
(b) SSP, wpc = 1 iteration, slack = 1 clock

EXPERIMENTAL SETUP

- App: Topic Modeling (LDA with Gibbs sampling)
  - Nytimes dataset (300k documents, 100m words)
  - Similar results for other ML apps
- Hardware (2 clusters)
  - Default: 8 64-core machines with 128GB RAM
  - vCloud: 32 8-core machines with 15GB RAM

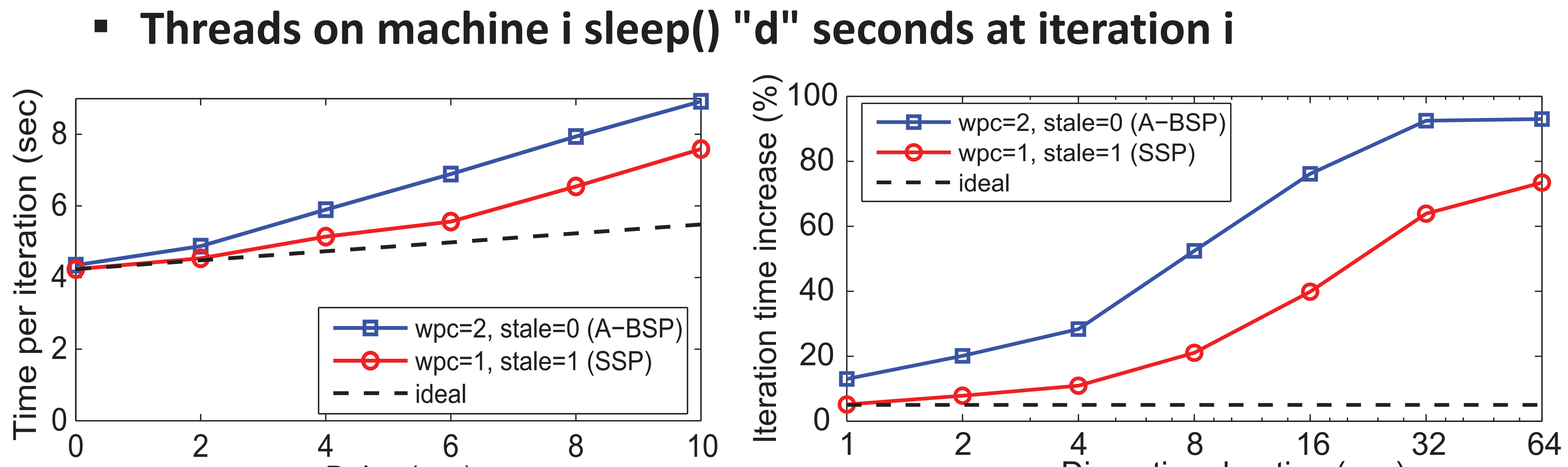
SPEED-EFFECTIVENESS TRADEOFF

- Controlled by staleness bound
- SSP: fixed wpc, change slack
  - More staleness → more iters/sec, less convergence/iter
  - A sweet spot in the middle
- A-BSP: slack is always zero, change wpc
  - Similar tradeoffs



TOLERANCE OF STRAGGLERS

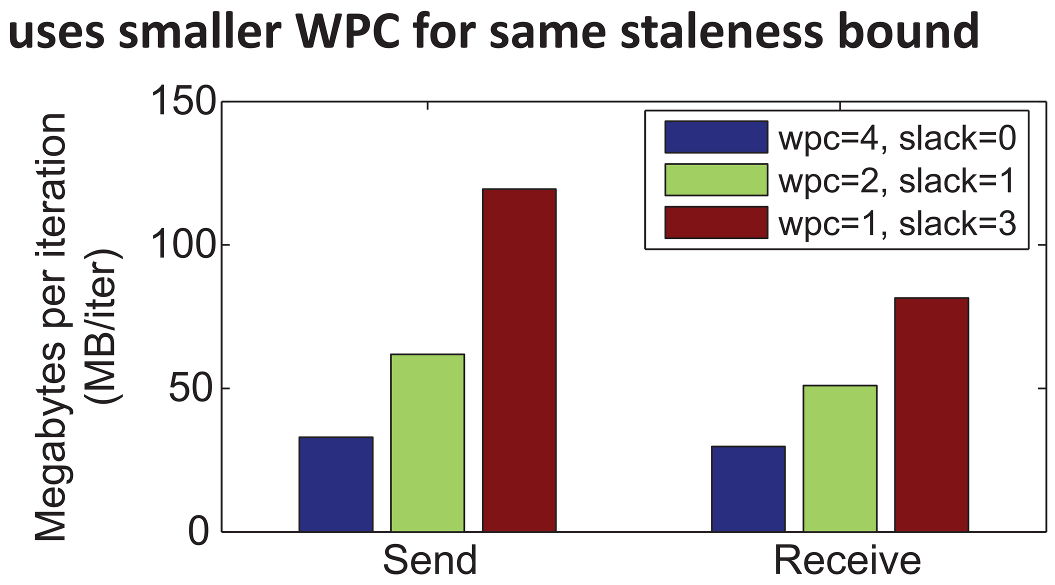
- SSP's slack mitigates small transient delays
- Ex: Delayed threads
  - Threads on machine i sleep() "d" seconds at iteration i



- Ex: Background work (on vCloud)
  - Disrupter process on each machine consumes 50% CPU in each time slot ("t" seconds) with probability 10%

COMMUNICATION OVERHEAD

- Total traffic drops as WPC grows
  - Updates sent every clock, reads on many
  - SSP uses smaller WPC for same staleness bound



Intel Science & Technology Center for Cloud Computing
Carnegie Mellon University
Georgia Tech
intel
PRINCETON UNIVERSITY
UC Berkeley
UNIVERSITY of WASHINGTON