Elasticizing the Linux Operating System for the Cloud

NSF XPS 2015 Workshop

Prof. Rick Han, Prof. Eric Keller
University of Colorado Boulder
Department of Computer Science

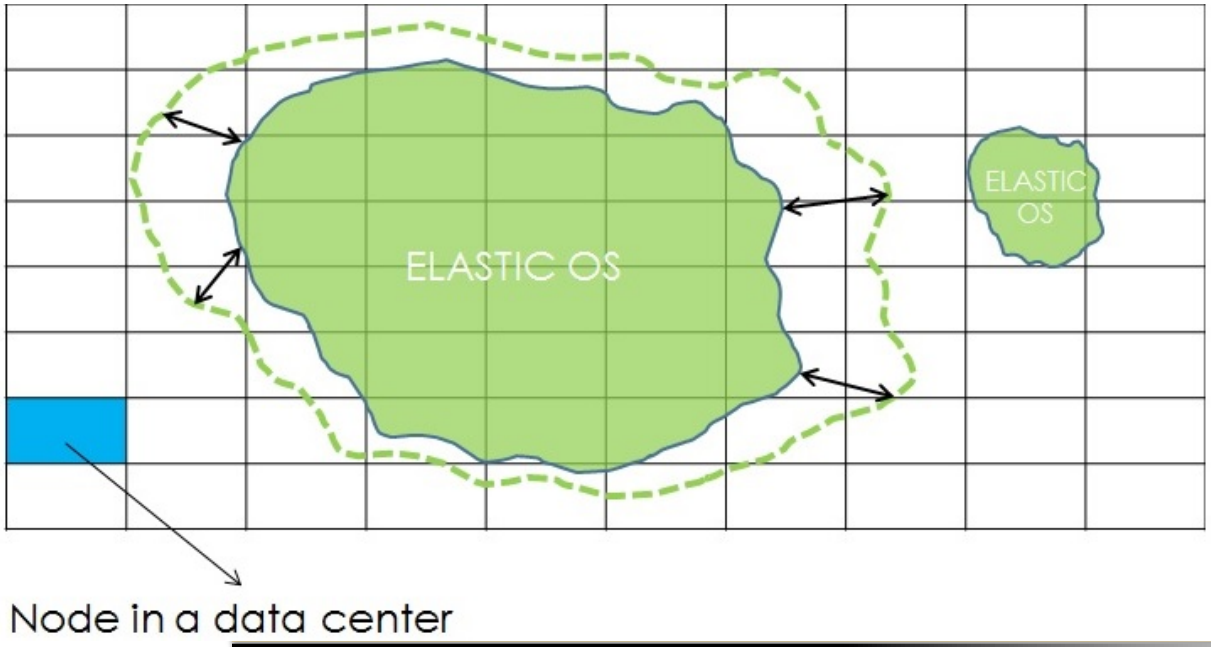
Why aren't Operating Systems more Elastic in a Cloud Era?

- *E* = *Elastic* in EC^2, i.e. *Elastic* Compute Cloud
- OSs typically compiled for static set of hardware
  - i.e. a single multi-core computer

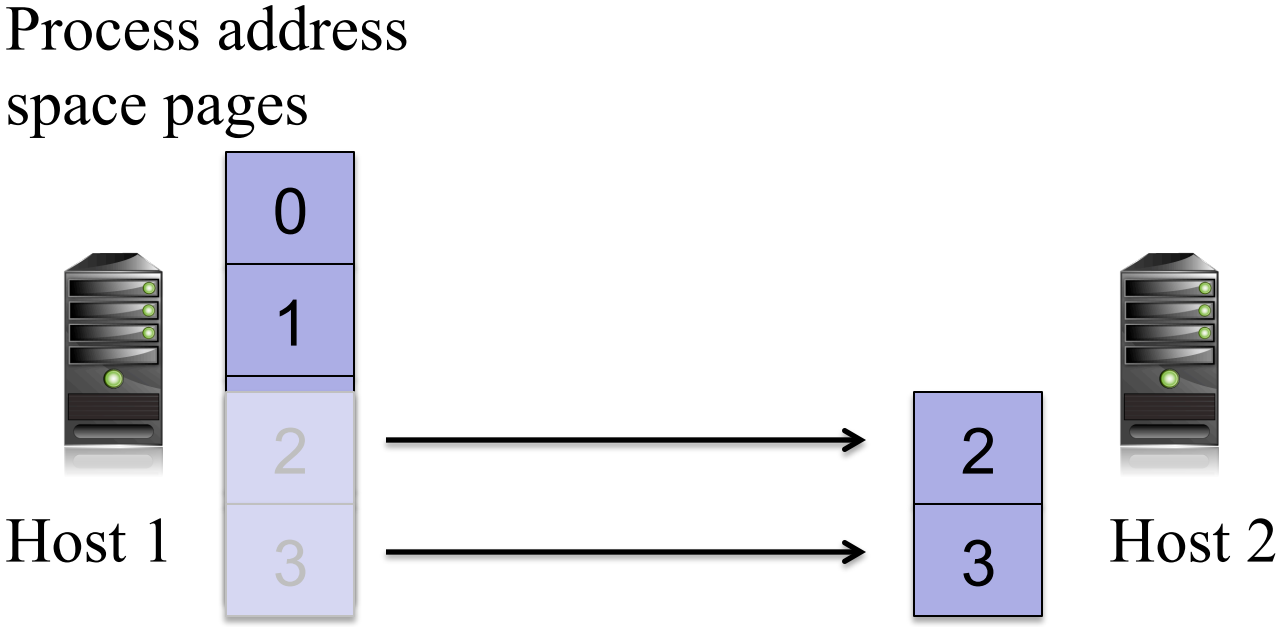
- How do applications scale today?
  - MPI, Map-reduce, … - need resources to do this
  - Fork/replicate multiple VMs/containers, …
    - refactor to synchronize
    - Build scripts to trigger scaling

ElasticOS: Let the OS Scale (underneath) the Application

- Improve programmability by letting (Linux) OS dynamically expand and contract across many rack machines as needed [HotOS 2013]
- Apps automatically scale without refactoring

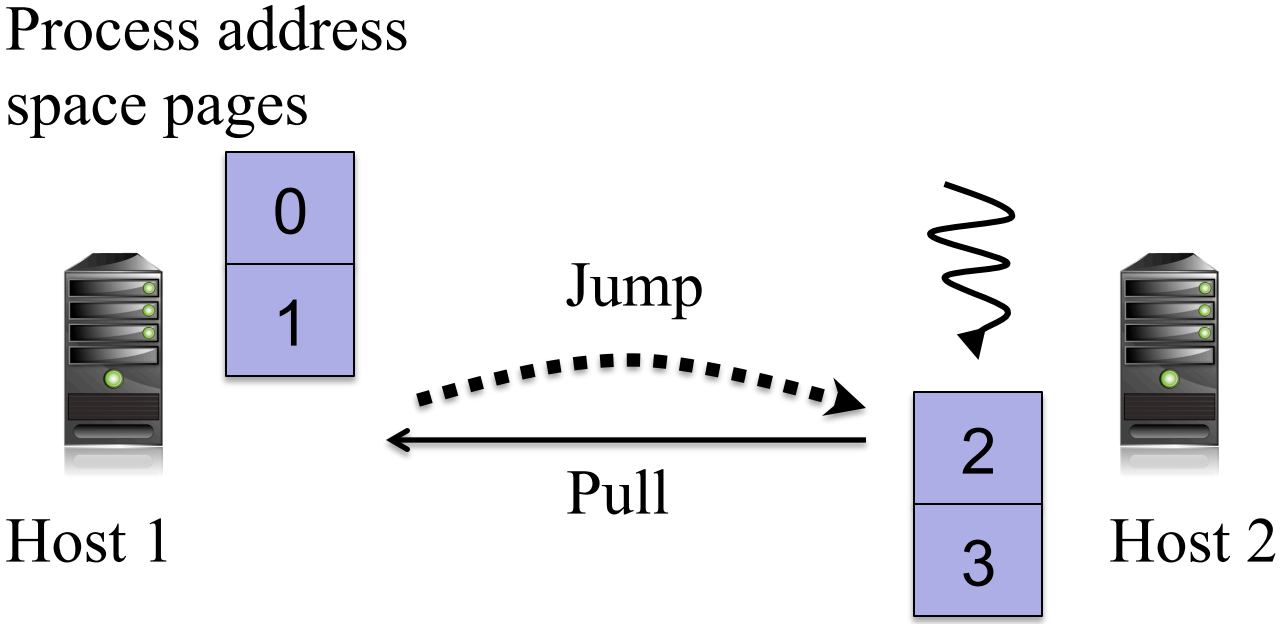


ElasticOS: Shard/Stretch Virtual Memory Across Many Machines



- *Stretch/shard* a single process' address space across many rack machines [HotOS 2013]
  - Need *elastic page table* to locate page
- Stretching triggered by local thrashing

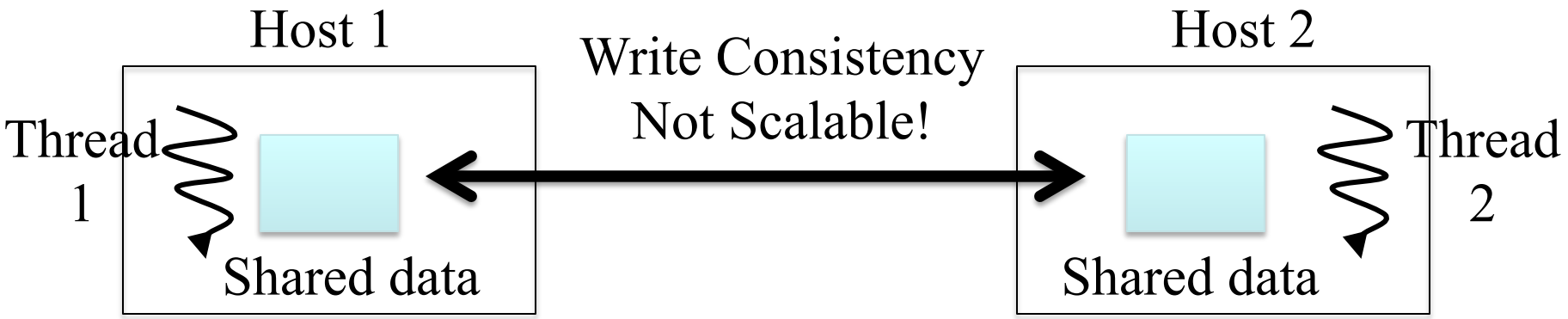
ElasticOS: Pulling, Pushing, Jumping



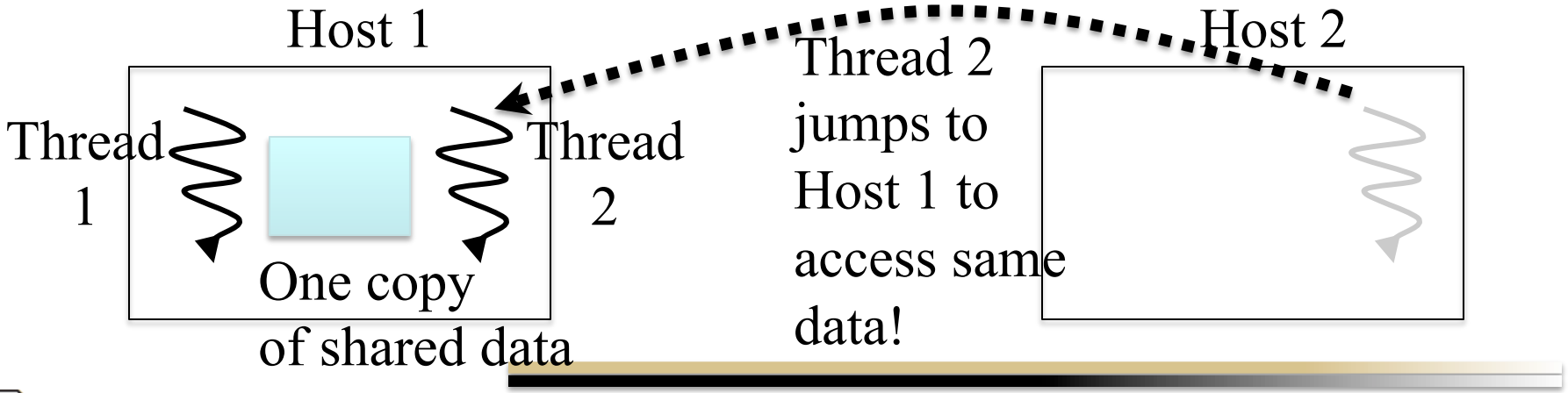
- Thread needing a page can
  - *Pull it*
  - *Jump to it if too many pulls (go towards locality)*
- If out of space, evict/push pages using LRU

How Elastic OS Differs from DSM

- DSM:



- ElasticOS sharding & jumping:



How ElasticOS Differs from SSI

- Traditional Single System Image (SSI) OSs
  - Kerrighed, MOSIX, LinuxPMI, etc.
- Common characteristics of SSI implementations:
  - Process migration
  - Distributed Shared Memory
- In process migration, a process can move but doesn't expand beyond a single machine
  - Couldn't *stretch* a process over many machines, e.g. large in-memory DB

ElasticOS: Status

- High risk high reward research
- Modifying Linux 2.6.38 kernel to implement stretching, jumping, pulling and pushing
  - Stretching is implemented
  - Jumping in one direction is implemented
  - Pulling and pushing is still under construction
- Examining locality of applications in a simpler model
  - Knowing when to jump, pull and push depends on locality of application
  - A single NUMA cache-coherent multi-core domain with different delays to different memory banks

Thanks

Feedback welcome!

Contact Rick Han
rhan@cs.colorado.edu