Nswap: a network swapping module for Linux clusters

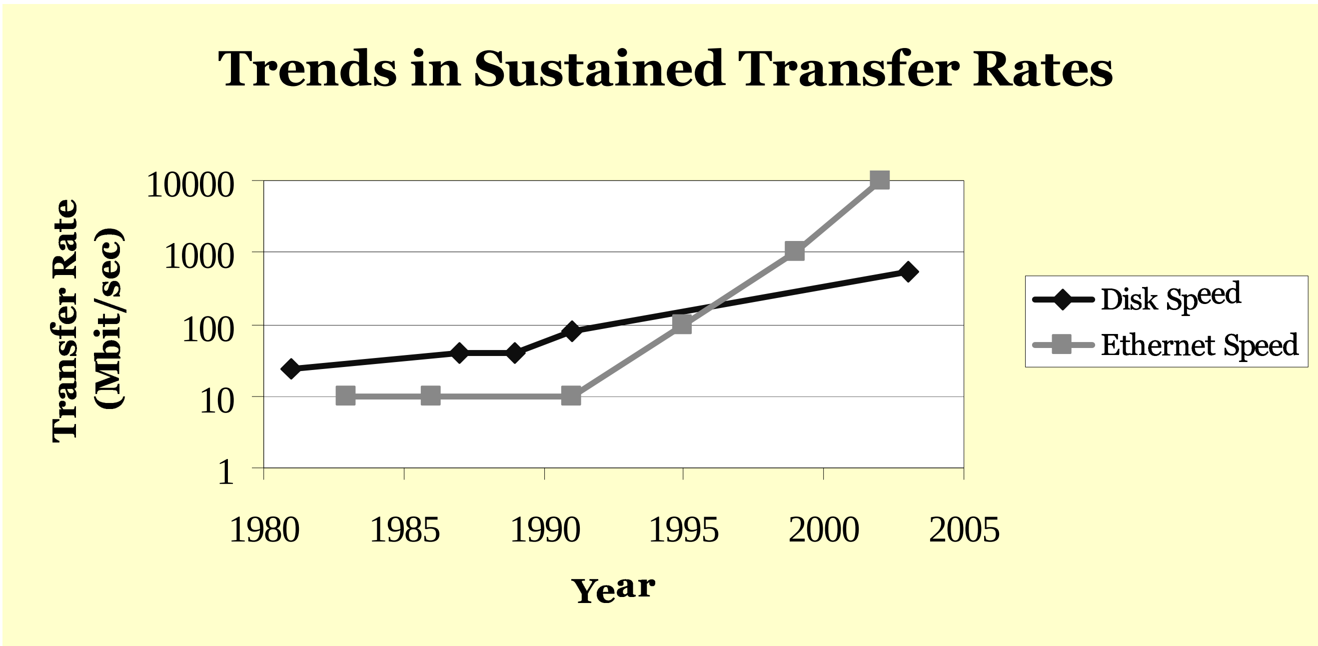
Tia Newhall, Sean Finney,
Kuzman Ganchev, Michael Spiegel
Computer Science Department
Swarthmore College
Swarthmore, PA USA
newhall@cs.swarthmore.edu

Network Swapping

- Let cluster nodes transparently share each other's RAM as swap space
  - when one node's memory is overcommitted it swaps to the idle memory of other nodes in the cluster
- Part of SSI support for cluster systems
  - Cluster as single, large parallel machine
  - Idle cluster RAM as a single, large, shared swap partition
- Also applicable to any NW of PCs/WS

Why Network Swapping?

- Network speeds are getting faster, disk speeds are not keeping up



- There is almost always some idle memory in the cluster even when some nodes are overloaded
  - Usually 2/3 idle, about 1/3 idle under heavy loads

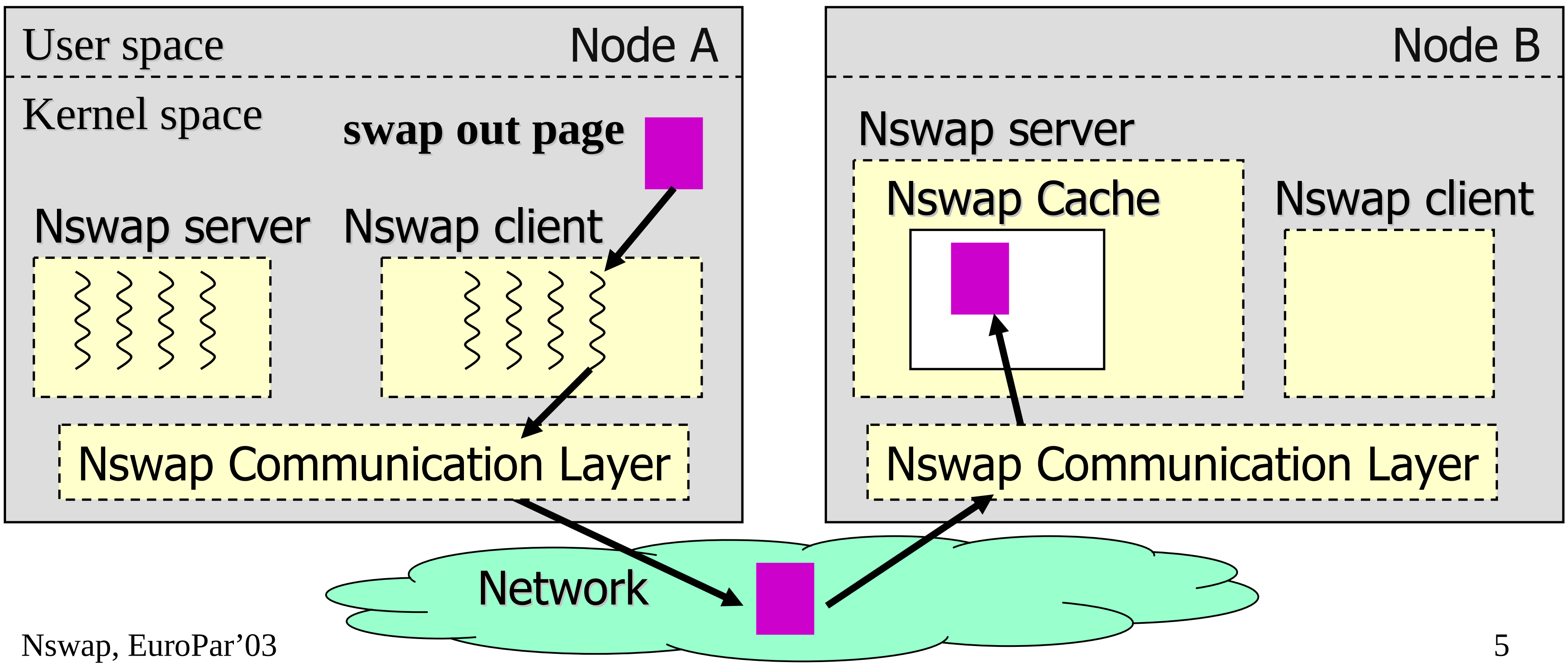
Nswap

- **Network swapping module for Linux clusters**
  - **Transparency**
    - Processes don't need to do anything special to use Nswap
  - **Efficiency and Portability**
    - Make swapping in and out fast to node doing the swapping
    - Kernel level implementation as Linux lkm
  - **Scalability**
    - Point-to-Point model
    - Don't require complete, nor accurate, global state info.

      => Each node independently w/o complete info. chooses the remote server to which to swap
  - **Adaptability**
    - Grow/Shrink each node's remote swap cache size based on its local memory needs
    - Remote page migration from server to server
      - avoid writing to disk when a server is full

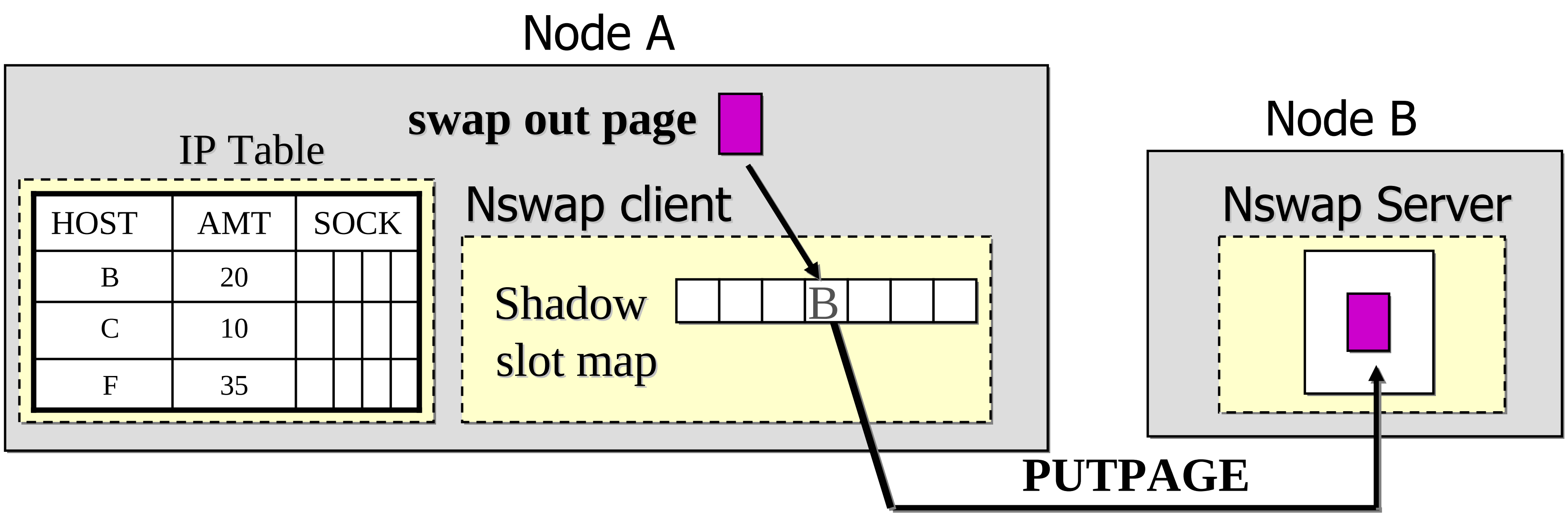
Nswap Architecture

- Each node runs multi-threaded client & server
- **Nswap client** device driver for network swap "device"
  - Kernel makes swap-in & swap-out requests to it
- **Nswap server** manages part of RAM for caching remotely swapped pages (Nswap Cache)

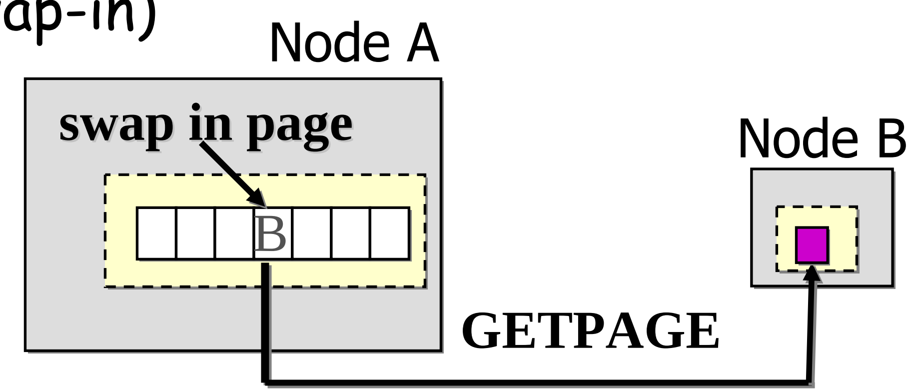


Nswap Communication Protocol

- PUTPAGE (swap-out)



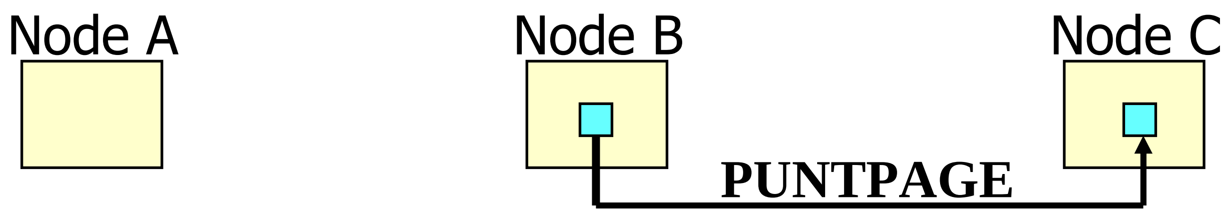
- GETPAGE (swap-in)



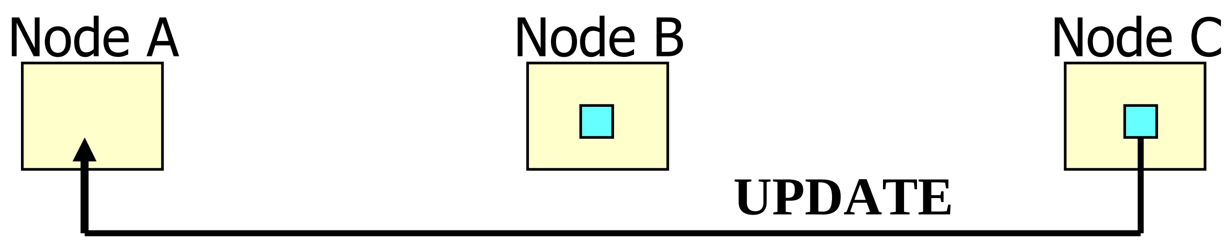
Nswap Communication Protocol

- Page Migration (PUNTPAGE)

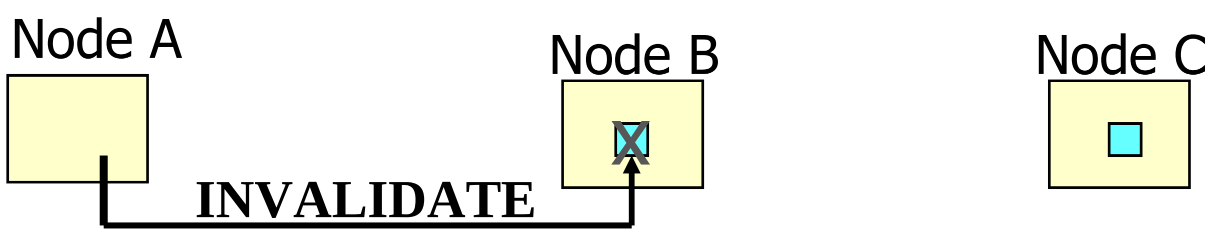
When Server B is full, it migrates A's page to server C



Server C tells A that it now has A's page



Client A tells Server B that it can drop its copy of A's page



Some Complications

- Kernel doesn't inform swap device driver when a slot is no longer being used
  - For disk swap devices this is fine
  - For NW swap devices this results in "dead" pages remaining cached on remote nodes

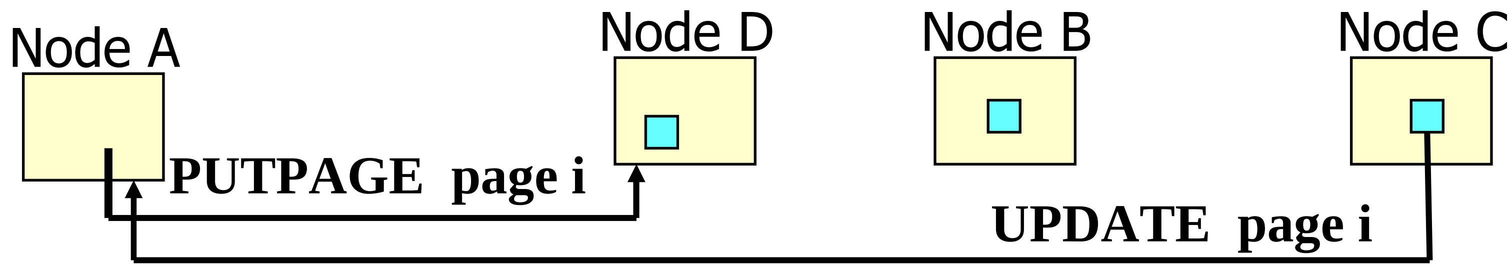
Nswap removes dead pages in 2 ways:

1) A re-use of a slot map results in an INVALIDATE message being set to the server caching the old, dead page
2) A garbage collector thread runs on each node, detecting dead slots and sending INVALIDATES

More Complications

❑ Simultaneous Conflicting Operations:

EX: PUNTPAGE for slot i and a new PUTPAGE for slot i:



- **A** *better not overwrite slot i with "C" losing its new location of "D"*
- *old page i at* **B** *and* **C** *should be dropped*

To detect & handle cases like these:

- extra state kept in shadow slot map
- extra state sent with protocol msgs

Our project so far…

- Implemented as lkm for Linux 2.4.18
- Running on cluster of 8 nodes connected with switched 100 BaseT
  - All nodes have faster disk than Network
    - PII's disk is up to 176 Mb/sec
    - PIII's disk is up to 494 Mb/sec
    - -> We expect to be slower than swapping to disk
- On 100BaseT Ethernet, Nswap is comparable in speed to swapping to faster disk
  - For several workloads Nswap on slower network is faster than swapping to fast disk

Experiments

- Workload 1: sequential R & W to large chunk of memory
  - Best case for swapping to disk
- Workload 2: random R & W to mem
  - Disk arm seeks w/in swap partition
- Workload 3: 1 large file I/O, 1 W1
  - Disk arm seeks between swap and file partitions
- Workload 4: 1 large file I/O, 1 W2

Results

Workload	PIII Disk (494 Mb/s)	Nswap (TCP 100BaseT)	Nswap (UDP 100BaseT)
(1) 1 proc	13.1	154.3	61.3
(1) 4 proc	577.0	1507.9	614.4
(2) 1 proc	266.8	1071.8	**155.5**
(2) 4 proc	68.6	189.3	**50.3**
(3) 1 proc	770.2	1111.0	811.0
(3) 4 proc	727.1	1430.5	**619.5**
(4) 1 proc	923.9	1529.3	**821.7**
(4) 4 proc	502.5	**498.7**	**429.2**

- Nswap faster than swapping to much faster disk for several workloads
- TCP latency hurting Nswap performance

Nswap on Faster Networks

Workload	Disk	10BaseT	100BaseT	1Gb	10Gb
(1) PIII TCP	580.10	5719.00	158.3 speed up 3.8	1075.0 (5.3)	1034.2 (5.5)
(1) PIII UDP	12.27	306.69	56.8 (5.4)	28.9 (10.6)	26.3 (11.6)
(2) PIII UDP	266.79	847.74	153.5 (5.5)	77.3 (10.9)	70.3 (12.1)
(4) PIII UDP	6265.39	9605.91	1733.9 (5.54)	866.2 (11.1)	786.7 (12.2)

Measured on Disk, 10 BaseT and 100 BaseT

Calculated speed-up values for 1 Gbit and 10 Gbit

Speedup = 1 / (1 – FracBandwidth + FracBandwidth/SpeedupBandwidth)

Conclusions

- Space efficient and time efficient implementation of Network Swapping
  - Designed to scale to large clusters
  - Adapts to local memory use on cluster nodes
- Nswap better than swapping to faster disk in several cases
- Nswap on faster NW will out perform disk in most cases
  - Based on NW vs. Disk speed trends, Nswap will be even better in the future

Future Work

- Develop better growing/shrinking policy
- Add reliability scheme to Nswap
- Test on larger, faster, heterogeneous clusters
- Implement faster reliable NW protocol
- Develop a swapping scheme that changes based on workload
  - For some workloads NW swapping may not be best choice