Nswap2L: Transparently Managing Heterogeneous Cluster Storage Resources for Fast Swapping

Tia Newhall
E. Ryerson Lehman-Borer
Benjamin Marks

Swarthmore College Computer Science Dept.
Swarthmore, PA USA
Data Intensive Computing

“The next 10 years will be shaped primarily by new algorithms that make sense of massive and diverse datasets and discover hidden value”  Genevieve Bell, Intel Fellow, from SC’13 Keynote

• Require significant resources for good performance (or even feasibility)

• Stress the memory hierarchy
  • WS too large to fit in RAM  →  swapping
  • Out-of-core algorithms  →  temporary files
General Purpose Clusters

• Variable WL lead to resource usage imbalances
  • Some nodes swapping, others have idle RAM

• Network RAM Storage is an option:
  • Use idle RAM of remote nodes for swap space
Cluster Storage Devices

Heterogeneous:

- HDD, Flash SSD, PCM(?), Network RAM, local and remote

Different Strengths:

- Network RAM: fast, volatile, variable capacity
- PCM: fast, expensive, capacity issues
- Flash SSD: faster R than W, erasure blocks, wear-out
- HDD: slow, cheap, still widely used

⇒ Cluster Storage likely to remain heterogeneous
Storage and Performance

• Making best use of heterogeneous storage will have significant impact on application performance

• Problem: Tuning node OS’s swapping subsystem for every possible combination of underlying storage is not possible
  • Still mostly optimized for disk
  • Some SSD support (TRIM/Discard)
Our Solution: Nswap2L

2 Level Device Driver Design:

- Transparent: to OS a single, fast, random access device
- Manages underlying heterogeneous storage devices
  Adaptable/Tunable Policies for Data Placement & Prefetching

![Diagram of Nswap2L architecture]

- OS Swapping Subsystem
  - Read, Write, Discard

- Nswap2L Top-level driver
- Network RAM driver
- Flash driver
- Disk driver
- Other drivers

- Network
  - remote node
  - remote node
  - remote node

- Flash SSD
- Disk
- local or remote cluster storage
System Architecture

- Linux 4.0 driver, added as swap partition on nodes
  - Underlying devices at load & added later (via /sys)
- OS sends Read/Write/Discard to swap in/out/free

Nswap2L: Top-level Device Driver

- Multi-threaded
- Slotmap
- Placement Policies on W
- dmio to most bottom-level devices
- Nework RAM part of Nswap2L
**Nswap2L Network RAM**

**Scalable:** P2P design, local estimate available Network RAM

**Adaptable:** Amount of RAM available for Network RAM grows & shrinks w/cluster workload RAM usage

---

**Nswap2L Network RAM (P2P design)**
- **Share Slotmap & Threads**
  - Encode server in Slotmap entry
  - Policies pick remote server use IPTable estimates of space
- **Stores remotely swapped pages**
  - Maintain & resize RAM Cache

**IP Table**

<table>
<thead>
<tr>
<th>IP</th>
<th>space amts</th>
<th>open socks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Other Device I/O**
- dmio Interface
  - Disk driver
  - Flash driver
  - Other drivers
  - … others

---

**Network**
Prefetching

• Moving page data between underlying devices
  • Transparent to OS: I/O internal to Nswap2L
• Take advantage of strengths of different devices
  (ex) Prefetching from Network RAM to Flash:
    • free up NW RAM for future fast Writes
    • increase Read parallelism over Flash and NW RAM

Tunable Policies
via /sys interface

  How much?
  How often?
  Which pages?
  To/From?
Nswap2L Performance

Nswap2L loaded as Linux 4.0.4 device driver, and added as swap partition on 16 node cluster, 10 Gbit Ethernet. Underlying Network RAM, Flash SSD & HDD.
Benefit of Prefetching

• Swapping out to faster Network RAM with prefetching some pages from Network RAM to slower Flash

• Increase degree of Read parallelism over Flash and Network RAM

• Get slightly faster runtime than swapping to faster Network RAM alone
Benefit of Adaptable Policies

• Flash available for entire run, Network RAM becomes available after about 5 minutes of runtime

• Adaptable Data Placement Policies allow Nswap2L to discover faster underlying storage and make use of it as it becomes available

• Significantly faster runtime than statically swapping to Flash alone
Nswap2L:

• Fast Backing Store for Swap in Clusters

• Transparently Manages Heterogeneous Cluster Storage Devices, including its own Network RAM
  • Presents as single, fast swap device to node OSs

• Adaptable polices result in performance improvements over fastest single swap device
Current and Future Directions

• Expand Nswap2L to be used for other types of backing storage
  • File data, particularly targeting temporary files

• Further investigating adaptive policies
  • Data placement
  • Prefetching
  • Network RAM growing/shrinking
  • Implementing a more Extensible Policy interface
    • adding in new policy on the fly

• Evaluate on Larger Systems, Scalability
Thank You

Questions?

www.cs.swarthmore.edu/~newhall/nswap2L.html