CART – The Implementation

Rahul Iyer

Agenda

- Linux MM Primer
  - Nodes
  - Zones
  - Pages
- New Stuff
- Changes to Old Stuff
- Where I’m at
- Where I go from here
- Challenges

Linux MM Primer

- There are 3 fundamental structures in Linux MM
  - Nodes
  - Zones
  - Pages
- Nodes are divided into Zones
- Zones are divided into pages

Linux MM (Contd.) - Nodes

- Memory is divided into Nodes
- Each Node represents the memory local to a processor in NUMA architecture
- Each node is represented by a pg_data_t
- Nodes are chained together by a list with head pgdat_list
- On UMA, there is only one Node
- This node is described by contig_page_data

Linux MM (Contd.) - Zones

- Nodes consist of Zones
- Currently there are 3 zones per node
  - ZONE_DMA
  - ZONE_NORMAL
  - ZONE_HIGHMEM
- Each Zone is represented by a struct zone
- Zone descriptors are stored in the array node_zones in a node
- Page Allocation and Freeing are *per Zone*

Linux MM (Contd.) - Pages

- Pages here refers to *Physical pages*
- Each page is described by a struct page
- Each page can be in one of 3 lists:
  - Active List
  - Inactive List
  - Slab Allocator
- Placed on a list via the page->lru field

New Stuff

- 4 new lists – T1, T2, B1 and B2
  - These lists replace the active and inactive lists
  - Blame the authors of CART for the ‘stunning’ nomenclature
- Created kernel threads that scan pages for their accessed bits
  - One kernel thread per node
  - Wake up and run every 10 seconds (currently arbitrary value)
  - Uses kernel timers
- The scans are used to decide when to move a page across lists

Changes to Existing Stuff

- Active_list and inactive_list replaced by T1 and T2 in struct zone
- Additional field accessed in struct page
- Modification in make_page_accessed() to move struct page to list T1
- Modification to shrink_list() to operate on the new lists
- Modification of the page_fault() to handle touches on B1 or B2

Where I’m at

- Implemented T1 and T2 lists
- Kernel threads up and running
- Mark_page_accessed() modified

Where I go from here…

- Current Prototype
  - Modification to shrink_list()
  - Modification of page_fault()
  - Set *un*arbitrary value for the kernel thread wakeup
  - Hope to finish by early/mid next week
- Longer term
  - Design a generic evictor framework
  - Port existing evictor to the above framework
  - Port CART to the above framework

Challenges

- It’s kernel code
  - Kernels have a way of blowing up in your face… frequently!
- LONG… VERY LONG compile times
- Cannot be a module
- Tests and quizzes!!