NUMA information in /proc/vmstat

Asked 4 years, 7 months ago Modified 4 years, 7 months ago

Viewed 1k times

4

I need to get some NUMA related information about my application (I can't use `numatop` tool for example, but I can use `numastat`). Therefore, I have some questions regarding NUMA related fields in `/proc/vmstat`, not being sure that I correctly understand their meaning.

These two are clearly related to allocation of (new) pages.

- `numa_hit` The number of pages that were successfully allocated to this node.
- `numa_miss` The number of pages that were allocated on this node because of low memory on the intended node.

What about access of already allocated pages? I'm especially interested in pages allocated on one node and accessed from another. Are these two the ones I'm looking for?

- `numa_hint_faults`
- `numa_hint_faults_local`

And in the end,

- `numa_pages_migrated` Records how many pages were migrated because they were misplaced.

Is this useful for me if I'm using custom calls from `libnuma`, like `numa_bind` in order to forcefully bind a process to a node? Without auto balancing are there any pages migrated to increment this counter?

linux **performance** **numa**

Share

Improve this question

Follow

asked Apr 26, 2020 at 17:21

jack malkovick

533 ● 2 ● 17

1 Answer

Sorted by: Highest score (default)

2

Those are metrics used to profile the Automatic NUMA Balancing.

The balancer works as follow:

1. When the *process* under inspection is not scheduled a *portion* of its address space is scanned and each page marked as not present.
   This will generate a fault the very next time the process accesses an address in those pages.
   These falts are intended and called *NUMA Hinting Faults* (NHF).

2. When a NHF happens the kernel migrates the page to the memory local to faulting *thread*.

Now, a process may have multiple threads and when the balancer pickups a portion of the address space it cannot know which thread will access which page and so it cannot rule out pages that are already local to a node where one of the thread is executing.

For example, if the process has two threads, A e B, in nodes N1 and N2, the balancer cannot skip page X even if it is already in the local memory of node N1 (or N2). So sometimes, the balancer will find itself in the situation where the NHF page is *already* in the memory closest to the thread, this is called *a local NHF*.

The percentage of *local NHF* over the total *NHF* is the esteem of how much optimal is the topology of the allocated memory.

Share Improve this answer

Follow

answered Apr 26, 2020 at 17:48

Margaret Bloom

43.9k ● 5 ● 87 ● 128

it seems you are right, counters does not seem to be affected by code that was manually bound to a numa node

– jack malkovick Apr 28, 2020 at 18:36