

Re: Active Memory Defragmentation: Our implementation & problems

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-02/0864.html>

From: Richard B. Johnson (root_at_chaos.analogic.com)

Date: 02/04/04

Date: Wed, 4 Feb 2004 14:59:35 -0500 (EST)

To: Timothy Miller <miller@techsource.com>

On Wed, 4 Feb 2004, Timothy Miller wrote:

>
>
> *Richard B. Johnson wrote:*
>
> > *If this is an Intel x86 machine, it is impossible for pages*
> > *to get fragmented in the first place. The hardware allows any*
> > *page, from anywhere in memory, to be concatenated into linear*
> > *virtual address space. Even the kernel address space is virtual.*
> > *The only time you need physically-adjacent pages is if you*
> > *are doing DMA that is more than a page-length at a time. The*
> > *kernel keeps a bunch of those pages around for just that*
> > *purpose.*
> >
> > *So, if you are making a "memory defragmenter", it is a CPU time-sink.*
> > *That's all.*
>
> *Would memory fragmentation have any appreciable impact on L2 cache line*
> *collisions?*
>
> *Would defragmenting it help?*
>
> *In the case of the Opteron, there is a 1M cache that is (I forget) N-way*
> *set associative, and it's physically indexed. If a bunch of pages were*
> *located such that there were a disproportionately large number of lines*
> *which hit the same tag, you could be thrashing the cache.*
>
> *There are two ways to deal with this: (1) intelligently locates pages*
> *in physical memory; (2) hope that natural entropy keeps things random*
> *enough that it doesn't matter.*
>
>

Certainly anybody can figure out some special case where a

Linux–Kernel: Re: Active Memory Defragmentation: Our implementation & problems

cache–line might get flushed or whatever. Eventually you get to the fact that even contiguous physical RAM doesn't have to be contiguous