
Subject: Re: [PATCH 4/7] chained slab caches: move pages to a different cache when a cache is destroyed.

Posted by [KAMEZAWA Hiroyuki](#) on Wed, 22 Feb 2012 01:25:12 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Tue, 21 Feb 2012 15:34:36 +0400

Glauber Costa <glommer@parallels.com> wrote:

> In the context of tracking kernel memory objects to a cgroup, the
> following problem appears: we may need to destroy a cgroup, but
> this does not guarantee that all objects inside the cache are dead.
> This can't be guaranteed even if we shrink the cache beforehand.
>
> The simple option is to simply leave the cache around. However,
> intensive workloads may have generated a lot of objects and thus
> the dead cache will live in memory for a long while.
>
> Scanning the list of objects in the dead cache takes time, and
> would probably require us to lock the free path of every objects
> to make sure we're not racing against the update.
>
> I decided to give a try to a different idea then - but I'd be
> happy to pursue something else if you believe it would be better.
>
> Upon memcg destruction, all the pages on the partial list
> are moved to the new slab (usually the parent memcg, or root memcg)
> When an object is freed, there are high stakes that no list locks
> are needed - so this case poses no overhead. If list manipulation
> is indeed needed, we can detect this case, and perform it
> in the right slab.
>
> If all pages were residing in the partial list, we can free
> the cache right away. Otherwise, we do it when the last cache
> leaves the full list.
>

How about starting from 'don't handle slabs on dead memcg'
if shrink_slab() can find them....

This "move" complicates all implementation, I think...

Thanks,
-Kame
