Subject: Re: [PATCH v2 2/5] account guest time per-cgroup as well.
Posted by Glauber Costa on Mon, 28 May 2012 13:26:25 GMT
View Forum Message <> Reply to Message

On 05/26/2012 08:44 AM, Paul Turner wrote:
> On 04/09/2012 03:25 PM, Glauber Costa wrote:
>> In the interest of providing a per-cgroup figure of common statistics,
>> this patch adds a nr_switches counter to each group runqueue (both cfs
>> and rt).
>>
>> To avoid impact on schedule(), we don't walk the tree at stat gather
>> time. This is because schedule() is called much more frequently than
>> the tick functions, in which we do walk the tree.
>>
>> When this figure needs to be read (different patch), we will
>> aggregate them at read time.
>>
>>

Paul,

How about the following patch instead?

It is still using the cfs_rq and rt_rq's structures, (this code actually only touches fair.c as a PoC, rt would be similar).

Tasks in the root cgroup (without an se->parent), will do a branch and exit. For the others, we accumulate here, and simplify the reader.

My reasoning for this, is based on the fact that all the se->parent relations should be cached by our recent call to put_prev_task (well, unless of course we have a really big chain)

This would incur a slightly higher context switch time for tasks inside a cgroup.

The reader (in a different patch) would then be the same as the others:

```
+static u64 tg_nr_switches(struct task_group *tg, int cpu)
+{
+ if (tg != &root_task_group)
+  return rt_rq(rt_nr_switches, tg, cpu)
                    +fair_rq(nr_switches, tg, cpu);
+
+ return cpu_rq(cpu)->nr_switches;
+}
```

I plan to measure this today, but an extra branch cost for the common

case of a task in the root cgroup + O(depth) for tasks inside cgroups
may be acceptable, given the simplification it brings.

Let me know what you think.

File Attachments

1) alternative.patch, downloaded 458 times