Subject: Re: [PATCH] Hookup group-scheduler with task container infrastructure
Posted by Dmitry Adamushko on Tue, 11 Sep 2007 09:53:51 GMT
View Forum Message <> Reply to Message

On 11/09/2007, Srivatsa Vaddagiri <vatsa@linux.vnet.ibm.com> wrote:
>
> [...]

I guess, 'rq->curr == tsk' implies a task was on the 'rq' (before dequeueing) in this particular case. What's about a minor optimization like below (plus, let's make use of task_running()):

[btw., real-time task can be also added to a container, right? I guess, it can be used at least for group-aware load-balancing of rt_tasks... otherwise, I guess, cpu-resource controlling is not that applicable to, say, SCHED_FIFO :-)
Anyway, to this goal rt_task should become aware of group-related bits of the 'struct sched_entity'... and at the moment, the code below is effectively just a 'nop' for them.. right?]

```
/* change task's runqueue when it moves between groups */
static void sched_move_task(struct container_subsys *ss, struct container *cont,
                    struct container *old_cont, struct task_struct *tsk)
{
	int on_rq, running;
	unsigned long flags;
	struct rq *rq;

	rq = task_rq_lock(tsk, &flags);
	update_rq_clock(rq);

	running = task_running(rq, tsk);
	on_rq = tsk->se.on_rq;
	if (on_rq) {
		dequeue_task(rq, tsk, 0);

		if (unlikely(running) && tsk->sched_class == &fair_sched_class)
			tsk->sched_class->put_prev_task(rq, tsk);
	}

	set_task_cfs_rq(tsk);

	if (on_rq) {
		enqueue_task(rq, tsk, 0);

		if (unlikely(running) && tsk->sched_class == &fair_sched_class)
			tsk->sched_class->set_curr_task(rq);
	}
```

```
	task_rq_unlock(rq, &flags);
}
```

> --
> Regards,
> vatsa
>

--
Best regards,
Dmitry Adamushko

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers