Subject: [PATCH 2/5] store number of iowait events in a task_group
Posted by Glauber Costa on Thu, 02 Feb 2012 14:19:29 GMT
View Forum Message <> Reply to Message

Instead of just having the rq to hold them, this patch stores
the nr_iowait figures for each task_group, except for the root
task group. That one is kept using the numbers originating from
the rq.

Signed-off-by: Glauber Costa <glommer@parallels.com>
---
 include/linux/sched.h | 1 +
 kernel/sched/core.c | 42 ++++++++++++++++++++++++++++++++++++++----
 2 files changed, 39 insertions(+), 4 deletions(-)

diff --git a/include/linux/sched.h b/include/linux/sched.h
index 5b8ff53..b629c1e 100644
--- a/include/linux/sched.h
+++ b/include/linux/sched.h
@@ -1207,6 +1207,7 @@ struct sched_entity {

  u64   nr_migrations;

+ atomic_t   nr_iowait;
 #ifdef CONFIG_SCHEDSTATS
  struct sched_statistics statistics;
 #endif
diff --git a/kernel/sched/core.c b/kernel/sched/core.c
index 455810f..fe35316 100644
--- a/kernel/sched/core.c
+++ b/kernel/sched/core.c
@@ -2665,7 +2665,41 @@ static inline void task_group_account_field(struct task_struct *p, int
index,
 #endif
 }

+static void task_group_inc_nr_iowait(struct task_struct *p, int cpu)
+{
+ struct task_group *tg;
+ struct rq *rq = cpu_rq(cpu);
+
+ rcu_read_lock();
+ tg = task_group(p);
+
+ atomic_inc(&rq->nr_iowait);
+
+ while (tg && tg != &root_task_group) {
+ atomic_inc(&tg->se[cpu]->nr_iowait);

```
+  tg = tg->parent;
+ }
+ rcu_read_unlock();
+
+}
+
+static void task_group_dec_nr_iowait(struct task_struct *p, int cpu)
+{
+ struct task_group *tg;
+ struct rq *rq = cpu_rq(cpu);
+
+ rcu_read_lock();
+ tg = task_group(p);
+
+ atomic_dec(&rq->nr_iowait);
+
+ while (tg && tg != &root_task_group) {
+  atomic_dec(&tg->se[cpu]->nr_iowait);
+  tg = tg->parent;
+ }
+ rcu_read_unlock();

+}
 /*
  * Account user cpu time to a process.
  * @p: the process that the cpu time gets accounted to
@@ -4677,12 +4711,12 @@ void __sched io_schedule(void)
  struct rq *rq = raw_rq();

  delayacct_blkio_start();
- atomic_inc(&rq->nr_iowait);
+ task_group_inc_nr_iowait(current, cpu_of(rq));
  blk_flush_plug(current);
  current->in_iowait = 1;
  schedule();
  current->in_iowait = 0;
- atomic_dec(&rq->nr_iowait);
+ task_group_dec_nr_iowait(current, cpu_of(rq));
  delayacct_blkio_end();
 }
 EXPORT_SYMBOL(io_schedule);
@@ -4693,12 +4727,12 @@ long __sched io_schedule_timeout(long timeout)
  long ret;

  delayacct_blkio_start();
- atomic_inc(&rq->nr_iowait);
+ task_group_inc_nr_iowait(current, cpu_of(rq));
  blk_flush_plug(current);
```

```
  current->in_iowait = 1;
  ret = schedule_timeout(timeout);
  current->in_iowait = 0;
- atomic_dec(&rq->nr_iowait);
+ task_group_dec_nr_iowait(current, cpu_of(rq));
  delayacct_blkio_end();
  return ret;
 }
--
1.7.7.4
```