Subject: Re: [RFC][only for review] memory controller bacground reclaim [0/5] (Does anyone have an idea abo
Posted by KAMEZAWA Hiroyuki on Thu, 29 Nov 2007 11:53:24 GMT
View Forum Message <> Reply to Message

On Wed, 28 Nov 2007 17:49:23 +0900
KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com> wrote:

> Hi, this set is for memory controller background reclaim.
>
> Merged YAMAMOTO-san's version onto 2.6.23-rc3-mm1 + my NUMA patch.
> And splitted to several sets.
>
> Major changes from his one is
> - use kthread instead of work_queue
> - adjust high/low watermark when limit changes automatically
>   and set default value. (a user can specify his own later.)
>
FYI rather than RFC.

I wrote attached patch and run kernbench on 8CPU/2Node NUMA/ia64.
It does make -j 32.

Memory limitation was 800M. Low/High watermark here was 750M/780M.

== These numbers are stable to some extent.==
2.6.24-rc3-mm2: (Limit: 800M)
Average Optimal -j 32 Load Run:
Elapsed Time 358.933----------(*)
User Time 1069.63
System Time 140.667
Percent CPU 337.333
Context Switches 220821
Sleeps 196912

2.6.24-rc3-mm2 + throttle (Limit:800M)
Average Optimal -j 32 Load Run:
Elapsed Time 266.697----------(*)
User Time 1105.39
System Time 124.423
Percent CPU 471.667
Context Switches 251797
Sleeps 231038

2.6.24-rc3-mm2 + throttle + High/Low watermark.
(low:750M High:780M Limit:800M)
Average Optimal -j 32 Load Run:
Elapsed Time 266.844----------(*)

User Time 1112.9
System Time 112.273
Percent CPU 473.667
Context Switches 251795
Sleeps 220339
==

Seems throttling reclaim has some good effect (for kernbench).
Does anyone have an idea for throttling reclaiming of memory controller ?

Thanks,
-Kame
==
Just and Experimental. add throttling at starting direct reclaim.

```
 mm/memcontrol.c |   22 ++++++++++++++++++----
 1 file changed, 18 insertions(+), 4 deletions(-)

Index: linux-2.6.24-rc3-mm2/mm/memcontrol.c
===================================================================
--- linux-2.6.24-rc3-mm2.orig/mm/memcontrol.c
+++ linux-2.6.24-rc3-mm2/mm/memcontrol.c
@@ -133,6 +133,7 @@ struct mem_cgroup {

  unsigned long control_type; /* control RSS or RSS+Pagecache */
  int prev_priority; /* for recording reclaim priority */
+ atomic_t reclaimers; /* # of processes which calls reclaim */
  /*
   * statistics.
   */
@@ -635,14 +636,27 @@ retry:
   */
  while (res_counter_charge(&mem->res, PAGE_SIZE)) {
   bool is_atomic = gfp_mask & GFP_ATOMIC;
+  int limit;
   /*
    * We cannot reclaim under GFP_ATOMIC, fail the charge
    */
   if (is_atomic)
    goto noreclaim;
-
-  if (try_to_free_mem_cgroup_pages(mem, gfp_mask))
-   continue;
+  /*
+   * Experimental:
+   * Obviously, we need some throttling here.
+   */
```

```
+  limit = num_online_cpus()/4 + 1;
+  if (atomic_add_unless(&mem->reclaimers, 1, limit)) {
+   if (try_to_free_mem_cgroup_pages(mem, gfp_mask)) {
+    atomic_dec(&mem->reclaimers);
+    continue;
+   }
+   atomic_dec(&mem->reclaimers);
+  } else {
+   yield();
+   nr_retries++;
+  }

   /*
    * try_to_free_mem_cgroup_pages() might not give us a full
@@ -1168,7 +1182,7 @@ mem_cgroup_create(struct cgroup_subsys *

  mem->control_type = MEM_CGROUP_TYPE_ALL;
  memset(&mem->info, 0, sizeof(mem->info));
-
+ atomic_set(&mem->reclaimers, 0);
  for_each_node_state(node, N_POSSIBLE)
   if (alloc_mem_cgroup_per_zone_info(mem, node))
    goto free_out;
```

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