Subject: [PATCH][for -mm] Fix and Enhancements for memory cgroup [4/6] avoid handling !LRU page in mem_cgrou
Posted by KAMEZAWA Hiroyuki on Tue, 09 Oct 2007 09:53:10 GMT
View Forum Message <> Reply to Message

This patch makes mem_cgroup_isolate_pages() to be

 - ignore !PageLRU pages.
 - fixes the bug that it makes no progress if page_zone(page) != zone
   page once find. (just increment scan in this case.)

kswapd and memory migraion removes a page from list when it handles
a page for reclaiming/migration.

__isolate_lru_page() doesn't moves page !PageLRU pages, then, it will
be safe to avoid touching the page and its page_cgroup.

Signed-off-by: KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>

 mm/memcontrol.c | 13 ++++++++---
 1 file changed, 10 insertions(+), 3 deletions(-)

Index: devel-2.6.23-rc8-mm2/mm/memcontrol.c
===
--- devel-2.6.23-rc8-mm2.orig/mm/memcontrol.c
+++ devel-2.6.23-rc8-mm2/mm/memcontrol.c
@@ -227,7 +227,7 @@ unsigned long mem_cgroup_isolate_pages(u
  unsigned long scan;
  LIST_HEAD(pc_list);
  struct list_head *src;
- struct page_cgroup *pc;
+ struct page_cgroup *pc, *tmp;

  if (active)
   src = &mem_cont->active_list;
@@ -235,11 +235,18 @@ unsigned long mem_cgroup_isolate_pages(u
   src = &mem_cont->inactive_list;

  spin_lock(&mem_cont->lru_lock);
- for (scan = 0; scan < nr_to_scan && !list_empty(src); scan++) {
- pc = list_entry(src->prev, struct page_cgroup, lru);
+ scan = 0;
+ list_for_each_entry_safe_reverse(pc, tmp, src, lru) {
+ if (scan++ > nr_taken)
+   break;
   page = pc->page;
   VM_BUG_ON(!pc);

```
+  if (unlikely(!PageLRU(page))) {
+   scan--;
+   continue;
+  }
+
   if (PageActive(page) && !active) {
    __mem_cgroup_move_lists(pc, true);
    scan--;
```