LRU 交换管理器设计文档

一.、概述

本文档描述了一个基于最近最少使用（LRU）算法的页面置换管理器的设计和实现。该管理器用于操作系统中虚拟内存管理，特别是处理页面置换的场景。通过跟踪页面的访问历史，LRU算法确保最久未被访问的页面被置换出去。

二.、设计目标

不考虑实现开销和效率的LRU页替换算法

三、设计思路与代码分析

核心函数如下所示，此函数被调用时，会遍历所有可交换页，若该页近期被访问过，则清除visited，否则加一

```
static int _lru_accessed_check(struct mm_struct *mm)
{
    cprintf("\nbegin accessed check----------------------------------\n");
    list_entry_t *head = (list_entry_t *)mm->sm_priv;   //头指针
    assert(head != NULL);
    list_entry_t *entry = head;
    while ((entry = list_prev(entry)) != head)
    {
        struct Page *entry_page = le2page(entry, pra_page_link);
        pte_t *tmp_pte = get_pte(mm->pgdir, entry_page->pra_vaddr, 0);
        cprintf("the ppn value of the pte of the vaddress is: 0x%x  \n", (*tmp_pte) >> 10);
        if (*tmp_pte & PTE_A)  //如果近期被访问过，visited清零(visited越大表示越长时间没被访问)
        {
            entry_page->visited = 0;
            *tmp_pte = *tmp_pte ^ PTE_A;//清除访问位
        }
        else
        {
            //未被访问就加一
            entry_page->visited++;
        }
        cprintf("the visited goes to %d\n", entry_page->visited);
    }
    cprintf("end accessed check-----------------------------------\n\n");
}
```

下面函数用于选出被换出的页，同样遍历可交换页，维护一个最大的visited值和对应的list_entry，遍历结束后即可得到拥有最大visited值的page用于换出

```
static int _lru_swap_out_victim(struct mm_struct *mm, struct Page **ptr_page, int in_tick)
{
    _lru_accessed_check(mm);
```

```
    list_entry_t *head = (list_entry_t *)mm->sm_priv;
    assert(head != NULL);
    assert(in_tick == 0);

    list_entry_t *entry = list_prev(head);
    list_entry_t *pTobeDel = entry;
    uint_t largest_visted = le2page(entry, pra_page_link)->visited;     //最长时间未被访问的page，比较的是visited
    while (1)
    {
        //entry转一圈，遍历结束
        // 遍历找到最大的visited，表示最早被访问的
        if (entry == head)
        {
            break;
        }
        if (le2page(entry, pra_page_link)->visited > largest_visted)
        {
            largest_visted = le2page(entry, pra_page_link)->visited;
            // le2page(entry, pra_page_link)->visited = 0;
            pTobeDel = entry;
            // curr_ptr = entry;
        }
        entry = list_prev(entry);
    }
    list_del(pTobeDel);
    *ptr_page = le2page(pTobeDel, pra_page_link);
    cprintf("curr_ptr %p\n", pTobeDel);
    return 0;
}
```

四、测试结果

```
check_vmm() succeeded.
SWAP: manager = lru swap manager
BEGIN check_swap: count 2, total 31661
 mm->sm_priv c02110e8 in lru_init_mm
setup Page Table for vaddr 0X1000, so alloc a page
setup Page Table vaddr 0~4MB OVER!
set up init env for check_swap begin!
Store/AMO page fault
page fault at 0x00001000: K/W
Store/AMO page fault
page fault at 0x00002000: K/W
Store/AMO page fault
page fault at 0x00003000: K/W
Store/AMO page fault
page fault at 0x00004000: K/W
set up init env for check_swap over!
write Virt Page c in lru_check_swap
write Virt Page a in lru_check_swap
write Virt Page d in lru_check_swap
write Virt Page b in lru_check_swap
write Virt Page e in lru_check_swap
Store/AMO page fault
page fault at 0x00005000: K/W
curr_ptr 0xffffffffc02258a8
```

```
swap_out: i 0, store page in vaddr 0x1000 to disk swap entry 2
write Virt Page b in lru_check_swap
write Virt Page a in lru_check_swap
Store/AMO page fault
page fault at 0x00001000: K/W
curr_ptr 0xffffffffc02258f0
swap_out: i 0, store page in vaddr 0x2000 to disk swap entry 3
swap_in: load disk swap entry 2 with swap_page in vadr 0x1000
write Virt Page b in lru_check_swap
Store/AMO page fault
page fault at 0x00002000: K/W
curr_ptr 0xffffffffc0225938
swap_out: i 0, store page in vaddr 0x3000 to disk swap entry 4
swap_in: load disk swap entry 3 with swap_page in vadr 0x2000
write Virt Page c in lru_check_swap
Store/AMO page fault
page fault at 0x00003000: K/W
curr_ptr 0xffffffffc0225980
swap_out: i 0, store page in vaddr 0x4000 to disk swap entry 5
swap_in: load disk swap entry 4 with swap_page in vadr 0x3000
write Virt Page d in lru_check_swap
Store/AMO page fault
page fault at 0x00004000: K/W
curr_ptr 0xffffffffc02258a8
```

```
swap_out: i 0, store page in vaddr 0x4000 to disk swap entry 5
swap_in: load disk swap entry 4 with swap_page in vadr 0x3000
write Virt Page d in lru_check_swap
Store/AMO page fault
page fault at 0x00004000: K/W
curr_ptr 0xffffffffc02258a8
swap_out: i 0, store page in vaddr 0x5000 to disk swap entry 6
swap_in: load disk swap entry 5 with swap_page in vadr 0x4000
write Virt Page e in lru_check_swap
Store/AMO page fault
page fault at 0x00005000: K/W
curr_ptr 0xffffffffc02258f0
swap_out: i 0, store page in vaddr 0x1000 to disk swap entry 2
swap_in: load disk swap entry 6 with swap_page in vadr 0x5000
write Virt Page a in lru_check_swap
Load page fault
page fault at 0x00001000: K/R
curr_ptr 0xffffffffc0225938
swap_out: i 0, store page in vaddr 0x2000 to disk swap entry 3
swap_in: load disk swap entry 2 with swap_page in vadr 0x1000
count is 1, total is 8
check_swap() succeeded!
++ setup timer interrupts
100 ticks
```