内存管理项目文档 1552681 陈淇格

1.开发环境

语言：java

IDE:Intellij IDEA

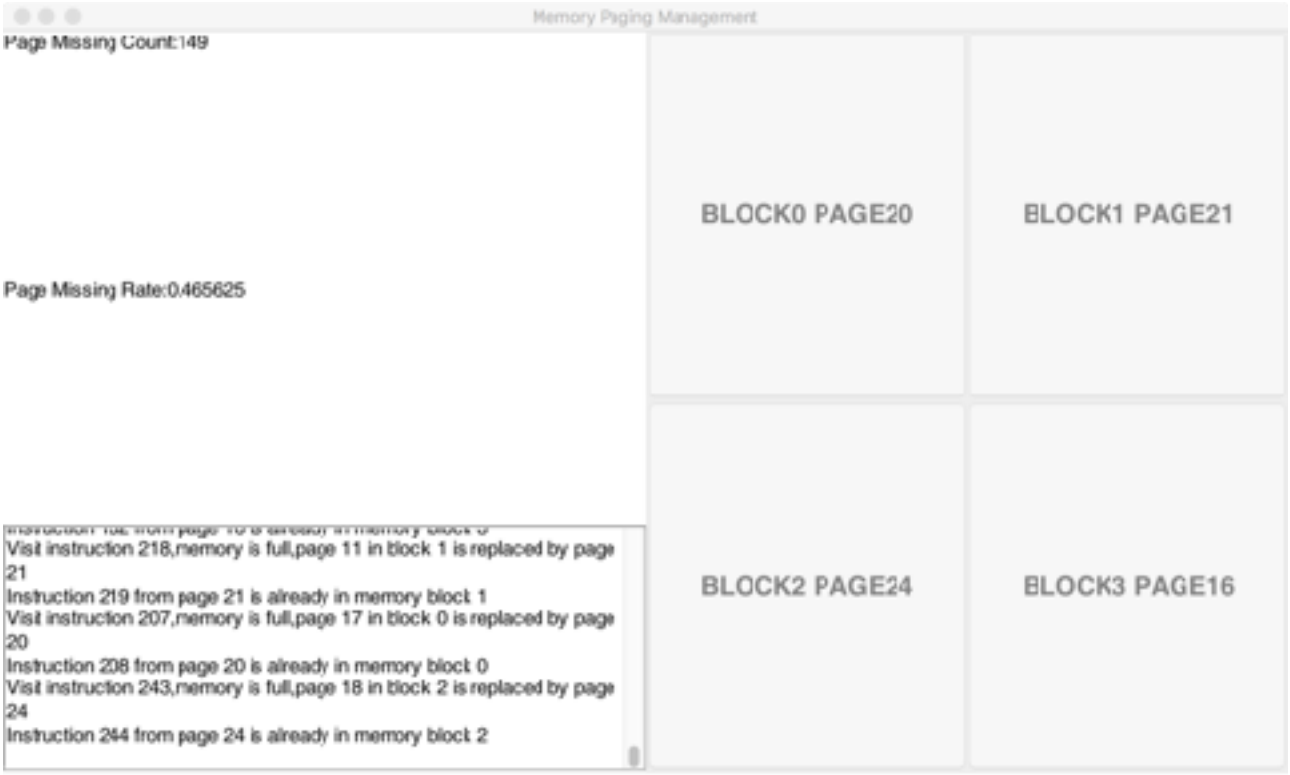
环境要求：安装java

2.使用说明

打开文件夹中的MemoryPagingManagement.jar文件，程序开始执行。

左边的信息栏显示了当前的缺页数、缺页率和执行的指令信息。

右边的内存块表示了当前在内存块中的页面号。



3.项目简介

这是一个模拟请求调页存储管理方式的项目。

一个作业共有320条指令，每个页面可存放10条指令，它的地址空间为32页。分配给一个作业的内存块为4。初始状态下所有页还没有调入内存。在模拟过

程中，如果所访问指令在内存中，则显示其物理地址，并转到下一条指令。如果没有在内存中，则发生缺页，此时需要记录缺页次数，并将其调入内存。

如果4个内存块中已装入作业，为了在内存满的情况下实现页面置换，需要将新的页面与内存中某个页面进行置换。在前几条指令中频繁使用的页面很可能在后面几条指令中被频繁使用，很久没被使用的页面可能在未来的较长一段时间内不会被用到，因此选择LRU算法，在每次调换时找到最久未使用的页面调出内存。

4.算法实现

（1）数据结构

这里用存放页面的数组来模拟物理内存块，用双向链表模拟页面队列。

```
private page []memoryBlocks;//物理内存
private page head;//链表头
private page tail;//链表尾
```

链表中存放的节点为页面的节点。从链表头到链表尾，存放着从旧到新的页面。新的页面从链表尾插入，旧的页面从链表头移出。

```
class page
{
    public int pageID;//页号
    public page prev;//前一页节点
    public page next;//后一页节点
    public page(int id,page prev,page next)
    {
        this.pageID=id;
        this.prev=prev;
        this.next=next;
    }
}
```

（2）访问指令步骤

作业中的指令访问顺序为50%的指令顺序执行，25%均匀分布在前地址部分，25%均匀分布在后地址部分。指令访问次数满320后，停止执行。

由于一个页面存放十条指令，可以用指令号除以10向下取整来表示指令所在的页面号。

①如果访问的指令所在页面不在内存中

发生缺页。

当内存不满时，指令所在页面插入链表的尾部，成为最新被访问的页面。同时，将页面放入一个空的内存块中。

内存为满时，链表的头部，即最久未被访问的页面移出链表。访问的指令所在的页面插入链表的尾部。同时，被移出链表的页面所在的内存块装入访问的指令所在的页面。

②如果访问的指令在内存中

因为访问了指令所在页面，需要更新页面在链表中的位置，以维持链表从旧到新的顺序规则。将此页面从链表中删除，再将页面插入链表尾部，表示这个页面最近被访问过。

```
public void addTail(int instructionID)//插入链表尾部
public void removeHead()//移除链表头部
```

下面的execute为核心算法，使用以上描述的访问指令步骤，根据指令在或不在内存中，以及内存的的空闲情况，来安排指令在链表中的位置及指令在内存块中的位置。

```
public void execute(int instructionID)//执行某条指令
```

下面的lru实现了作业中指令访问顺序的安排，并在main函数中被调用。

```
public void lru()
```

5.可视化

使用了java swing进行内存块及缺页率、访问信息的可视化。

页面左侧的信息栏为infoPanel的一个实例，右侧的内存块为memoryPanel的一个实例。

```
class infoPanel extends JPanel//信息面板
class memoryPanel extends JPanel//内存面板
```