

实验结果

1. 总体设计

当出现缺页异常，需调入新页面而内存已满时，通过页面置换算法选择能够选择被置换的物理页面，从而达到解决上述问题。

页面置换算法的设计目标是尽可能减少页面的调入调出次数，把未来不再访问或短期内不访问的页面调出。

本实验主要功能包括生成页面访问随机序列和页面置换，并输出各个算法的缺页率、算法开销。页面置换算法。其中页面置换可选择最佳置换算法、先进先出置换算法、最近最久未使用置换算法、页面缓冲置换算法、改进的 clock 算法。

2. 实验结果

根据实验功能，我们将程序分为 5 个模块，主要包括选择菜单设计、虚拟页面结构结构设计、页面访问随机序列设计、5 种页面置换算法设计、性能分析比较设计。

2.1. 页面访问随机序列生成设计

```
typedef struct MemSchedule
{
    WorkItem* Workspace = NULL;
    int *VisitSeq = NULL;
    int N = 64;
    int p = 0;
    //工作集大小
    int e = 5;
    int m = 1;
    //定义访问序列长度o
    int length;
```

```

//当前已使用物理块数
int work_len = 0;
//发生替换的物理块号
int change = 0;
float t, r;
// 队列长度
int queue_free_len = 0;
int queue_modified_len = 0;
WorkItem free[2];
WorkItem Modified[2];

}MemSchedule;

```

2.2. 最佳置换算法

1. 算法简介

1. 基本思想：选择永不使用或是在最长时间内不再被访问（即距现在最长时间才会被访问）的页面淘汰出内存
2. 评价：理想化算法，具有最好性能（对于固定分配页面方式，本法可保证获得最低的缺页率），但实际上却难于实现，故主要用于算法评价参照。

2.实验结果

页面访问序列									
	4	9	8	9	7	6	5	3	44 41
访问	4	:	内存	<	4		>	==>缺页	缺页率0.0
访问	9	:	内存	<	4	9		>==>缺页	缺页率0.0
访问	8	:	内存	<	4	9	8	>==>缺页	缺页率0.0
访问	9	:	内存	<	4	9	8	>	
访问	7	:	内存	<	7	9	8	>==>缺页	缺页率25.0
访问	6	:	内存	<	6	9	8	>==>缺页	缺页率40.0
访问	5	:	内存	<	5	9	8	>==>缺页	缺页率50.0
访问	3	:	内存	<	3	9	8	>==>缺页	缺页率57.1
访问	44	:	内存	<	44	9	8	>==>缺页	缺页率62.5
访问	41	:	内存	<	41	9	8	>==>缺页	缺页率66.7

2.3. 先进先出算法

1. 算法简介

基本思想：选择最先进入内存即在内存驻留时间最久的页面换出到外存，进程已调入内存的页面按进入先后次序链接成一个队列，并设置替换指针以指向最老页面

评价：简单直观，但不符合进程实际运行规律，性能较差，故实际应用极少

2. 实验结果

```
是否设定调度算法参数(Y/N):Y
请输入虚拟内存尺寸N:32
请输入工作集起始位置p:12
请输入工作集包含页数e:5
请输入工作集移动效率m:3
*****参数*****
虚拟内存:32      起始位置p:12
包含页数e:5      移动效率m:3
*****
生成随机内存访问序列:
14 16 12 16 15 15 16 12 12 14 13 13 14 14 13 16 14 15 14 13 13 12 14 13 15 16 14 16 12 16 13
14 15 16 13 13 15 14 16 14 14 16 13 16 15 12 12 14 13 12 14 15 13 12 16 12 12 14 16 16 1
7 17 13 14 14 16 17 17 16 15 16 15 17 14 13 17 16 13 15 14 14 14 13 15 13 17 16 15 15 13 14
14 15 13 17
*****先入先出算法*****
seq      1      2      3      4      5
14:      14@    *      *      *      *
16:      14      16@    *      *      *
12:      14      16      12@    *      *
16:      14      16      12      *      *
15:      14      16      12      15@    *
15:      14      16      12      15      *
16:      14      16      12      15      *
12:      14      16      12      15      *
12:      14      16      12      15      *
14:      14      16      12      15      *
```

2.4. 最近最久未使用置换算法

1. 算法简介

基本思想：以“最近的过去”作为“最近的将来”的近似，选择最近一段时间最长时间内未被访问的页面淘汰出内存。

2. 实验结果

```

=====页面访问序列=====
26 28 26 26 31 32 26 30 45 46

=====
访问 26 : 内存< 26      > ==>缺页 缺页率0.0
访问 28 : 内存< 26 28   > ==>缺页 缺页率0.0
访问 26 : 内存< 26 28   >
访问 26 : 内存< 26 28   >
访问 31 : 内存< 26 28 31 > ==>缺页 缺页率0.0
访问 32 : 内存< 26 32 31 > ==>缺页 缺页率20.0
访问 26 : 内存< 26 32 31 >
访问 30 : 内存< 26 32 30 > ==>缺页 缺页率28.6
访问 45 : 内存< 26 45 30 > ==>缺页 缺页率37.5
访问 46 : 内存< 46 45 30 > ==>缺页 缺页率44.4

```

2.5. 改进的 clock 算法

1. 算法简介

基本思想：

1. 从查寻指针当前位置起扫描内存分页循环队列，选择 $A=0$ 且 $M=0$ 的第一个页面淘汰；若未找到，转 2)
2. 开始第二轮扫描，选择 $A=0$ 且 $M=1$ 的第一个页面淘汰，同时将经过的所有页面访问位置 0；若不能找到，转 1)

评价：与简单 Clock 算法相比，可减少磁盘的 I/O 操作次数，但淘汰页的选择可能经历多次扫描，故实现算法自身的开销增大

2.实验结果

```

*****执行CLOCK算法*****
请输入要分配的页框数：3
请输入要随机生成访问序列的长度：10

=====页面访问序列=====
50 48 50 47 53 51 51 51 53 52

=====
访问 50 : 内存< 50      > ==>缺页 缺页率0.0
访问 48 : 内存< 48      > ==>缺页 缺页率0.0
访问 50 : 内存< 50      > ==>缺页 缺页率0.0
访问 47 : 内存< 47      > ==>缺页 缺页率33.3
访问 53 : 内存< 47 53   > ==>缺页 缺页率50.0
访问 51 : 内存< 47 53 51 > ==>缺页 缺页率60.0
访问 51 : 内存< 47 53 51 >
访问 51 : 内存< 47 53 51 >
访问 53 : 内存< 47 53 51 >

```

2.6. 页面缓冲置换算法

1. 算法简介

基本思想：

1. 设立空闲页面链表和已修改页面链表
2. 采用可变分配和基于先进先出的局部置换策略，并规定被淘汰页先不做物理移动，而是依据是否修改分别挂到空闲页面链表或已修改页面链表的末尾
3. 空闲页面链表同时用于物理块分配
4. 当已修改页面链表达达到一定长度如 Z 个页面时，一起将所有已修改页面写回磁盘，故可显著减少磁盘 I/O 操作次数

2.实验结果

```
访问 2 : 内存< 2 >
访问 2 : 内存< 2 >
访问 4 : 内存< 2 4 >
访问 2 : 内存< 2 4 >
访问 4 : 内存< 2 4 >
访问 1 : 内存< 2 4 1 >
访问 2 : 内存< 2 4 1 >
访问 4 : 内存< 2 4 1 >
访问 3 : 内存< 4 1 3 > ==>缺页
访问 4 : 内存< 4 1 3 >
访问 15 : 内存< 1 3 15 > ==>缺页
访问 15 : 内存< 1 3 15 >
访问 14 : 内存< 14 3 15 > ==>缺页
访问 15 : 内存< 14 3 15 >
访问 13 : 内存< 13 3 15 > ==>缺页
访问 15 : 内存< 13 3 15 >
访问 14 : 内存< 14 3 15 > ==>缺页
访问 13 : 内存< 13 3 15 > ==>缺页
访问 15 : 内存< 13 3 15 >
访问 14 : 内存< 14 3 15 > ==>缺页
```