作业 10

毕定钧 2021K8009906014

本次作业包含:

10.1 假设一台计算机上运行的一个进程其地址空间有 8 个虚页（每个虚页大小为 $4KB$，页号为 1 至 8），操作系统给该进程分配了 4 个物理页框（每个页框大小为 $4KB$），该进程对地址空间中虚页的访问顺序为 1 3 4 6 2 3 5 4 7 8。假设分配给进程的 4 个物理页框初始为空，请计算:

（1）如果操作系统采用 $CLOCK$ 算法管理内存，那么该进程访存时会发生多少次 $page\ fault$？当进程访问完上述虚页后，物理页框中保存的是哪些虚页？

（2）如果操作系统采用 LRU 算法管理内存，请再次回答（1）中的两个问题。请回答虚页保存情况时，写出 LRU 链的组成，标明 LRU 端和 MRU 端。

10.2 假设一台计算机给每个进程都分配 4 个物理页框，每个页框大小为 $512B$。现有一个程序对一个二维整数数组（$uint32\ X[32][32]$）进行赋值操作，该程序的代码段占用一个固定的页框，并一直存储在内存中。程序使用剩余 3 个物理页框存储数据。该程序操作的数组 X 以列存储形式保存在磁盘上，即 $X[0][0]$ 后保存的是 $X[1][0]$、$X[2][0]$...$X[31][0]$，然后再保存 $X[0][1]$，以此类推。当程序要赋值时，如果所赋值的数组元素不在内存中，则会触发 $page\ fault$，操作系统将相应元素以页框粒度交换至内存。如果该进程的物理页框已经用满，则会进行页换出。该程序有如下两种写法。

写法 1:

$$
\begin{aligned}
&for(int\ i = 0;\ i < 32;\ i++)\\
&\quad for(int\ j = 0;\ j < 32;\ j++)\\
&\quad\quad X[i][j]\ =\ 0
\end{aligned}
$$

写法 2:

$$
\begin{aligned}
&for(int\ j = 0;\ j < 32;\ i++)\\
&\quad for(int\ i = 0;\ i < 32;\ j++)\\
&\quad\quad X[i][j]\ =\ 0
\end{aligned}
$$

请分析使用这两种写法时，各自会产生多少次 $page\ fault$？(注：请写出分析或计算过程)

10.3 假设一个程序有两个段，其中段 0 保存代码指令，段 1 保存读写的数据。段 0 的权限是可读可执行，段 1 的权限是可读可写，如下所示。该程序运行的内存系统提供的虚址空间为 $14-bit$ 空间，其中低 $10-bit$ 为页内偏移，高 $4-bit$ 为页号。

当有如下的访存操作时，请给出每个操作的实际访存物理地址或是产生的异常类型（例如缺页异常、权限异常等）

（1）读取段 1 中 $page$ 1 的 $offset$ 为 3 的地址

（2）向段 0 中 $page$ 0 的 $offset$ 为 16 的地址写入

（3）读取段 1 中 $page$ 4 的 $offset$ 为 28 的地址

（4）跳转至段 1 中 $page$ 3 的 $offset$ 为 32 的地址

Segment 0		Segment 1	
Read/Execute		Read/Write	
Virtual Page #	Page frame #	Virtual Page #	Page frame #
0	2	0	On Disk
1	On Disk	1	14
2	11	2	9
3	5	3	6
4	On Disk	4	On Disk
5	On Disk	5	13
6	4	6	8
7	3	7	12

10.4 假设一个程序对其地址空间中虚页的访问序列为 $0, 1, 2, \ldots, 511, 422, 0, 1, 2, \ldots, 511\ 333, 0, 1, 2, \ldots$，即访问一串连续地址（页 0 到页 511）后会随机访问一个页（页 422 或页 333），且这个访问模式会一直重复。请分析说明：

（1）假设操作系统分配给该程序的物理页框为 500 个，那么，*LRU*，*Second Chance* 和 *FIFO* 这三种算法中哪一个会表现较好（即提供较高的缓存命中率），或是这三种算法都表现不佳？为什么？

10.1

(1)

访存时共发生 9 次缺页，进程结束后保存的是 5、4、7、8。根据 *CLOCK* 算法，按如下方式标注，数据均为操作后状态，页框标号仅作区分，顺序为从 *Oldest Page* 开始读取的环路：

操作	页框 1	R1	页框 2	R2	页框 3	R3	页框 4	R4	缺页
初始状态	NULL	-	NULL	-	NULL	-	NULL	-	-
访问页 1	1	0	NULL	-	NULL	-	NULL	-	是
访问页 3	1	0	3	0	NULL	-	NULL	-	是
访问页 4	1	0	3	0	4	0	NULL	-	是
访问页 6	1	0	3	0	4	0	6	0	是
访问页 2	3	0	4	0	6	0	2	0	是
访问页 3	3	1	4	0	6	0	2	0	否
访问页 5	6	0	2	0	3	0	5	0	是
访问页 4	2	0	3	0	5	0	4	0	是
访问页 7	3	0	5	0	4	0	7	0	是
访问页 8	5	0	4	0	7	0	8	0	是

访问页 1，初始为空，发生缺页，将页 1 读取至内存；

访问页 3，剩余页框 3，发生缺页，将页 3 读取至内存；

访问页 4，剩余页框 2，发生缺页，将页 4 读取至内存；

访问页 6，剩余页框 1，发生缺页，将页 6 读取至内存；

访问页 2，发生缺页，当前最老页为页 1，R 位为 0，替换页；

访问页 3，不发生缺页，将页 3 访问位置 1；

访问页 5，发生缺页，当前最老页为页 3，R 位为 1，将其 R 位置 0 并移动指针至页 4，R 位为 0，替换页；

访问页 4，发生缺页，当前最老页为页 6，R 位为 0，替换页；

访问页 7，发生缺页，当前最老页为页 2，R 位为 0，替换页；

访问页 8，当前最老页为页 3，R 位为 0，替换页。

(2)

访存时共发生 9 次缺页，进程结束后保存的是 5、4、7、8。根据 LRU 算法，每次替换最长时间未访问的页，过程如下：

操作	页框 1 (LRU)	未访问时间	页框 2	未访问时间	页框 3	未访问时间	页框 4 (MRU)	未访问时间	缺页
初始状态	NULL	-	NULL	-	NULL	-	NULL	-	-
访问页 1	1	0	NULL	-	NULL	-	NULL	-	是
访问页 3	1	1	3	0	NULL	-	NULL	-	是
访问页 4	1	2	3	1	4	0	NULL	-	是
访问页 6	1	3	3	2	4	1	6	0	是
访问页 2	3	3	4	2	6	1	2	0	是
访问页 3	4	3	6	2	2	1	3	0	否
访问页 5	6	3	2	2	3	1	5	0	是
访问页 4	2	3	3	2	5	1	4	0	是
访问页 7	3	3	5	2	4	1	7	0	是
访问页 8	5	3	4	2	7	1	8	0	是

10.2

根据题目所述，每个页框能够存储 $\frac{512\ B}{4\ B} = 128$ 个 $uint32$ 类型数字，这也就意味着数组 $X[32][32]$ 需要使用 8 页，第 i 页存储 $X[0 \sim 31][(4i-4) \sim (4i-1)]$。

写法 1：

按行访问时，每访问 4 个数即需要进行换页，观察规律可以发现，首先存入 $X[0 \sim 3]$、$X[4 \sim 7]$、$X[8 \sim 11]$，之后读取 $X[12 \sim 15]$，发生缺页，假设采用 $FIFO$ 算法进行页替换，那么每次换页均会发生缺页，不妨将 8 个虚页记为 1、2、3、4、5、6、7、8，那么 3 个物理页框存储的页依次为 $1--$、$12-$、123、234、345、456、567、678、781、812、...，并进行循环，共发生 $8 \times 32 = 256$ 次 $Page\ Fault$。

写法 2：

按列访问时，由于本身就是按列存储的，只需要进行 3 次读取的缺页和 5 次替换的缺页，共发生

8 次 $Page\ Fault$。

10.3

虚页大小为 $1KB$，假定物理页框大小为 $4KB$，起始地址为 $0x0$。

(1)

$addr = 0x0 + 14 \times 4KB + 3 = 0xE043$，不会触发异常。

(2)

$addr = 0x0 + 2 \times 4KB + 16 = 0x2010$，由于权限为读、执行，不可写，会触发权限异常。

(3)

触发缺页异常，由于没有对应物理页框，无法计算地址。

(4)

$addr = 0x0 + 6 \times 4KB + 32 = 0x6020$，由于权限为读、写，不可执行，会触发权限异常。

10.4

LRU 算法

每个周期访问 513 页。根据 LRU 算法，第一个周期前 512 次访问不命中，第 513 次访问命中；之后的每个周期，由于前一周期的第 513 次访问将页 422 或页 333 放入最后，因此它的前 512 次访问会命中 1 次，第 513 次访问命中。故第 n 个周期命中率为 $\frac{1+2\times(n-1)}{513n} = \frac{2n-1}{513n}$。

$Second\ Chance$ 算法

同样的，第一个周期前 512 次访问不命中，第 513 次访问命中；之后的每个周期，同 LRU 算法一样，前 512 次访问会命中 1 次，第 513 次访问命中。故第 n 个周期命中率也为 $\frac{1+2\times(n-1)}{513n} = \frac{2n-1}{513n}$。

$FIFO$ 算法

对 FIFO 算法来说, 每一周期都与第一周期一致, 仅命中 1 次, 故第 n 个周期命中率为 $\frac{n}{513n} = \frac{1}{513}$。

可以看到，三种算法均表现不佳，原因在于每周期访问 513 页恰好大于分配的 500 页，而且每周期访问的顺序几乎一致，因此每次访问一页的时候它都是刚刚被替换出内存，导致了不命中。