Homework 01

PB20000296 郑滕飞

**1.**

用户：操作系统满足不同的用户存在不同的需求——PC用户希望便捷、**易于使用**、**可视化良好**的体验，并不太在乎资源利用；共享计算机用户则需要尽量大的**资源利用率**；专用系统用户有专用资源，但有时也需要使用共享资源，因此**权衡使用**十分重要；移动电脑的资源少，更关心**适用性**与**电池寿命**；有些电脑则几乎不需要用户界面。

系统：操作系统是一个用于控制程序正确执行，避免错误与不恰当使用的**控制程序**，也是一个管理所有资源，以有效、公平的方式处理冲突请求的**资源分配器**。

**2.**

多道程序设计：组织作业之间的切换（**作业调度**）以保证CPU持续工作（**并发**），从而充分使用系统资源。

多任务系统：CPU切换速度很快，从而可以在程序运行时与其**交互**，**响应时间**较短，允许多个用户共享一台计算机，每个用户至少有一个程序在内存中。

**3.**

层次：[寄存器-高速缓存-内存](一级储存)-[固态磁盘-硬盘-光盘-磁盘](二级储存)

**缓存**思想：将信息复制到速度更快的储存系统中以方便调用（如内存就可以看作二级储存结构的缓存）

**4.**

系统调用：系统调用与函数类似，但是其在操作系统内（**内核**）执行，提供了进程与内核的**程序接口**。

与API：开发人员一般根据API（**应用编程接口**）设计程序，这些函数往往会为程序员调用的系统调用。比起直接使用系统调用，API具有更好的可移植性、少一些细节且更容易应用。尽管如此，API函数和相关的系统调用间还是常存在紧密联系的。实际使用时，很多语言提供了**系统调用接口**，截取API函数并调用对应的系统调用。通过这样的设计，操作系统接口的大多数细节被隐藏起来，可由运行时库来管理，调用者也只需遵循API即可。

**5.**

工作机制：计算机通过一个**模式位**来区分**用户模式**与**内核模式**，以此区分操作系统执行的任务与用户执行的任务。用户通过系统调用请求操作系统服务时，系统切换到内核模式以满足请求，返回前再切换至用户模式。

采用原因：通过将可能引起损害的机器指令设定为**特权指令**，只允许在内核模式执行，使操作系统可以保护自身与其他系统组件。

**6.**

用户功能服务：用户界面（命令行界面/批处理界面/图形用户界面）、程序执行、I/O操作、文件系统操作、通信、错误检测

操作系统服务：资源分配、记账、保护与安全

**7.**

单片结构：将所有功能集成到一层中，除系统程序外**一切**都在内核中。优点：系统调用与内核通信开销非常小；缺点：难以实现与设计。

层次化结构：将操作系统分层，每一层只与上一层有接口。优点：容易构造与调试；缺点：难以合理定义各层、效率问题。

微内核结构：从内核中删去所有不必要的部件，靠**消息传递**为各种服务提供通信。优点：便于扩展、安全性、可靠性；缺点：增加系统功能开销导致性能受损。

模块化结构：采用一系列**可加载的内核模块**组成，通过模块链入额外服务。优点：类似分层但**更加灵活**，类似微内核但无需消息传递而**更高效**。

**8.**

机制与策略分离：机制决定**如何做**，策略决定**做什么**，机制具有一定的通用性，而策略会随时间地点改变，分离可以使策略的改变只需要重新定义一些系统参数，由此拥有了很大的**灵活性**。

Homework 02

PB20000296 郑滕飞

**1.**

设执行fork层数为n，n=0时结果为1，n=k+1时主程序和fork的子进程均为n=k时情况，故最终结果为，题目中即n=4时，16个。

**2.**

fork成功但子进程中execlp执行失败。

**3.**

子进程fork返回值为0，A=0；

子进程PID为2603，B=2603；

父进程fork返回值为子进程PID，C=2603；

父进程PID为2600，D=2600。

**4.**

子进程输出的数依次为0,-1,-4,-9,16；

父进程输出的数依次为0,1,2,3,4；

调用子进程时，复制了一份参数，拥有不同的地址空间。

**5.**

程序转去执行execlp中的程序，在exit(0)后退出。

当execlp失败时，此行将执行。

**6.**

父进程可能需要获取子进程的**退出状态**等信息，因此必须存在终止状态。

**7.**

僵尸进程：进程已经终止但父进程尚未调用wait，父进程调用wait后即释放。

**8.**

用户：用户代码/常量、全局变量、局部变量、动态分配内存……

内核：系统调用代码、**进程控制块**(PCB)……

**9.**

当exec找到要执行的函数时，执行代码变为**新的程序代码**，局部变量、动态分配内存被清除，全局变量按照新程序重设。

普通函数调用不改变执行代码，保存上述内容。

**10.**

多线程优点：新建进程**消耗资源**较多、很多资源在不同任务中事实上**可以共用**

共享全局变量与堆内存(共享代码、数据与打开的文件等)，各自拥有寄存器与栈内存(各自有PC与线程ID)。

**11.**

**a.**

父进程产生了两个子进程，第一个子进程又产生了两个子进程，其中第一个又产生了子进程，共6个子进程。

**b.**

每个子进程为一个线程，其中两个子进程执行了thread create，共8个线程。

**12.**

子进程输出为5，父进程输出为0。线程共享全局变量，但子进程与父进程变量在不同地址空间。

**13.**

普通管道只在**父子进程间**传递信息(只由创建的进程访问)，具有**单向性**，一端写另一端读，只有进程相互通信时才存在；命名管道可以**双向**且父子关系**并不必须**，建立后可以由多个进程通信，当通信进程完成后仍然存在。

Homework 03

PB20000296 郑滕飞

**1.**

到达/退出实现要求：**互斥**(不允许同时进入关键部分)、不能提前假定关键部分的**执行时间**与CPU数、**进展性**(只有关键部分可能阻塞)、**有限等待**(进程不可能永远无法进入)

严格轮转：满足互斥、不控制执行时间、有限等待，但是不满足进展性(只能严格**交替**进行)

**2.**

原子性：操作中途不会发生上下文切换，因此可以保证。

到达/退出：退出后取消interest，因此可满足进展性，其他三者与严格轮转同理满足。

表现：添加是否需要占用的变量，不限制执行方式，优于严格轮转。

**3.**

出现条件：**互斥**运行、获得部分资源后**等待**其余资源、不允许抢占、等待资源形成**循环**

**4.**

\*括号内g代表执行完后的各资源数量

a.

T4(3,2,2,4) -> T0(4,4,2,6) -> T1(4,5,3,8) -> T3(5,7,3,9) -> T2

b.

T2(5,6,5,1) -> T4(6,6,5,2) -> T3(7,8,5,3) -> T1(7,9,6,5) -> T0

c.

不安全，资源B无法满足任何进程的要求

d.

T3(2,7,2,3) -> T4(3,7,2,4) -> T2(4,9,6,4) -> T1(4,10,7,6) -> T0

**5.**

信号量定义：一种拥有up与down两个基本原子操作的数据类型

作用：作为共享的对象标记可用资源数，控制进程的资源分配，具有互斥性、同步性

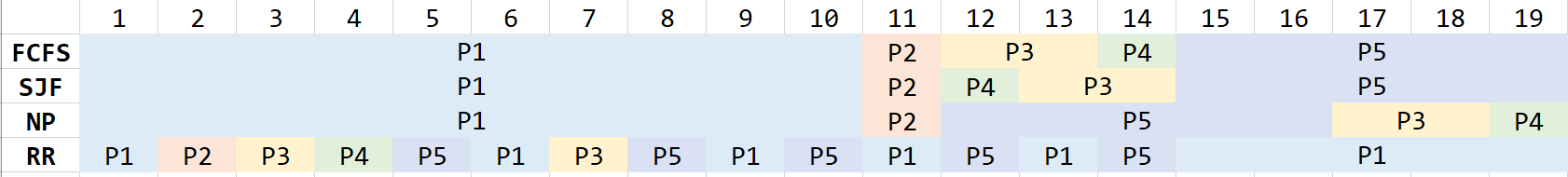
**6.**

用信号量mutex标识互斥锁(用于改变代表每个哲学家是否在吃面全局变量)，每个哲学家对应信号量s[i]表示每个人的需求是否满足。在临界区域前，将状态更新为Hungry并且检查附近的哲学家是否在吃面，若否就吃面并且设置对应的状态，在吃完后还原状态。若附近在吃面则进入等待队列，直到左右的哲学家吃完后唤醒。

**7.**

\*理解为P1到达时其他均未到达，因此SJF与NP均优先处理P1

**a.**



**b.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **P1** | **P2** | **P3** | **P4** | **P5** |
| **FCFS** | 10 | 11 | 13 | 14 | 19 |
| **SJF** | 10 | 11 | 14 | 12 | 19 |
| **NP** | 10 | 11 | 18 | 19 | 16 |
| **RR** | 19 | 2 | 7 | 4 | 14 |

**c.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **P1** | **P2** | **P3** | **P4** | **P5** |
| **FCFS** | 0 | 10 | 11 | 13 | 14 |
| **SJF** | 0 | 10 | 12 | 11 | 14 |
| **NP** | 0 | 10 | 16 | 18 | 11 |
| **RR** | 9 | 1 | 5 | 3 | 9 |

**d.**

FCFS为9.6，SJF为9.4，优先级算法为11，RR为5.4，RR最短。

**e.**

FCFS: 公平，但平均等待时间往往会较长。

SJF: 平均等待时间更短，但是需要估算完成时间(允许抢占则可能有上下文切换问题)。

RR: 等待时间短(但也可能更长)，但频繁上下文切换。优点为所有进程都在前进。

优先级: 可以根据实际情况调整，但需要相应设计好的优先级算法。

**8.**

单调速率调度：对**固定间隔**到达CPU的进程采用，固定优先级，允许抢占。

最短时限优先：优先处理**时限最近**的进程。

最短时限更优的例子：当进程的间隔相差不规则时(如进程1每9单位到达，需要用时5单位、进程2每7单位到达，需要用时4单位时)，最短时限优先能排下所有进程(刚才的例子中，进程会占据62/63的时间，单调速率无论如何都会导致阻塞，而最短时限可以成功处理)。

Homework 04

PB20000296 郑滕飞

**1.**

段错误：对段的**非法访问**(如试图写入只读段或访问未分配段)。

TLB(转址旁路缓存)：用于**缓存**最近访问的页，从而无需每次在页表中搜索。

页错误：当**使用的虚拟页没有对应物理内存**时CPU发送的中断。

按需调页：先分配虚拟页，实际使用时再给虚拟页分配物理内存。

**2.**

抖动定义：进程没有足够页框而导致频繁发送页错误，导致调页时间过长。

发生情况：如过多进程同时进行导致频繁页错误，CPU利用率反而下降。

**3.**

**a.**

先访问页表再访问内存，时间为100ns。

**b.**

75%的指令只需要52ns，25%指令需要100ns，故平均时间64ns。

**4.**

TLB未找到+无页错误：先访问一次某虚拟地址，时隔很久后重新访问，此时虽然未记录在TLB中但有物理内存。

TLB未找到+页错误：首次访问某虚拟地址时，此时TLB未记录也未分配物理内存。

TLB找到+无页错误：最近访问的虚拟地址再次被访问，此时其在TLB中且已分配物理内存。

TLB找到+页错误：不可能。TLB找到说明之前被访问过，不可能发生页错误。

**5.**

发生页错误时，70%情况需要20ms，30%情况需要8ms，从而平均时间14400000ns。

最高比例满足，即约0.0007%。

**6.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LRU replacement** | | | | | | | | | | | | | | | | | | | |  |
| **7** | **2** | **3** | **1** | **2** | **5** | **3** | **4** | **6** | **7** | **7** | **1** | **0** | **5** | **4** | **6** | **2** | **3** | **0** | **1** |  |
| **7** | **7** | **7** | **1** | **1** | **1** | **3** | **3** | **3** | **7** | **7** | **7** | **7** | **5** | **5** | **5** | **2** | **2** | **2** | **1** |  |
| **N** | **2** | **2** | **2** | **2** | **2** | **2** | **4** | **4** | **4** | **4** | **1** | **1** | **1** | **4** | **4** | **4** | **3** | **3** | **3** |  |
| **N** | **N** | **3** | **3** | **3** | **5** | **5** | **5** | **6** | **6** | **6** | **6** | **0** | **0** | **0** | **6** | **6** | **6** | **0** | **0** |  |
| **F** | **F** | **F** | **F** |  | **F** | **F** | **F** | **F** | **F** |  | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **18 Page Faults** |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **FIFO replacement** | | | | | | | | | | | | | | | | | | | |  |
| **7** | **2** | **3** | **1** | **2** | **5** | **3** | **4** | **6** | **7** | **7** | **1** | **0** | **5** | **4** | **6** | **2** | **3** | **0** | **1** |  |
| **7** | **7** | **7** | **1** | **1** | **1** | **1** | **1** | **6** | **6** | **6** | **6** | **0** | **0** | **0** | **6** | **6** | **6** | **0** | **0** |  |
| **N** | **2** | **2** | **2** | **2** | **5** | **5** | **5** | **5** | **7** | **7** | **7** | **7** | **5** | **5** | **5** | **2** | **2** | **2** | **1** |  |
| **N** | **N** | **3** | **3** | **3** | **3** | **3** | **4** | **4** | **4** | **4** | **1** | **1** | **1** | **4** | **4** | **4** | **3** | **3** | **3** |  |
| **F** | **F** | **F** | **F** |  | **F** |  | **F** | **F** | **F** |  | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **F** | **17 Page Faults** |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Optimical replacement** | | | | | | | | | | | | | | | | | | | |  |
| **7** | **2** | **3** | **1** | **2** | **5** | **3** | **4** | **6** | **7** | **7** | **1** | **0** | **5** | **4** | **6** | **2** | **3** | **0** | **1** |  |
| **7** | **7** | **7** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** |  |
| **N** | **2** | **2** | **2** | **2** | **5** | **5** | **5** | **5** | **5** | **5** | **5** | **5** | **5** | **4** | **6** | **2** | **3** | **3** | **3** |  |
| **N** | **N** | **3** | **3** | **3** | **3** | **3** | **4** | **6** | **7** | **7** | **7** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |  |
| **F** | **F** | **F** | **F** |  | **F** |  | **F** | **F** | **F** |  |  | **F** |  | **F** | **F** | **F** | **F** |  |  | **13 Page Faults** |

**7.**

Belady反常：页框数量增加时页错误反而增加。

栈算法：保证页框少时其中的页是页框多时的情况的子集。

Homework 05

PB20000296 郑滕飞

**1.**

**a.**

RMW：由于RAID-5有一个校验块，RAID-6有两个校验块，因此RAID-5需访问两个，RAID-6需访问3个。

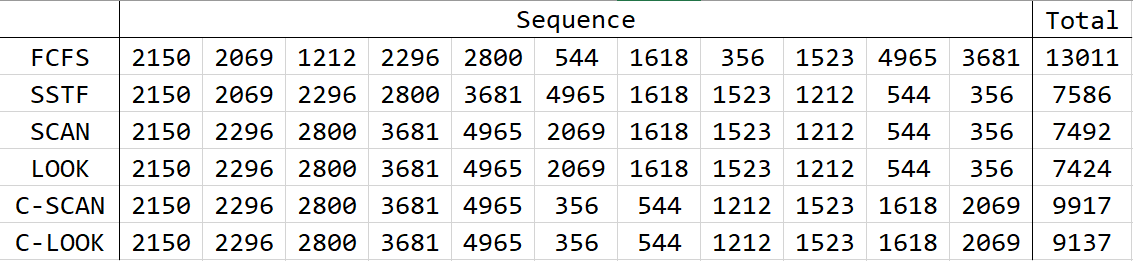
RRW：RAID-5与RAID-6都需要访问一列块，因此均为5个。

**b.**

RMW：RAID-5涉及两个校验块，因此需访问7+2=9个；RAID-6涉及三列块，共6个校验块，因此需访问7+6=13个。

RRW：RAID-5涉及两列，RAID-6涉及三列，因此RAID-5需访问10个，RAID-6需访问15个。

**2.**



**3.**

打开文件列表：任何文件使用前需通过系统调用open()打开，所有打开的文件排列在此列表中，通过close()关闭并移出列表。

作用：在对文件进行操作时**不用每次在目录中搜索**，直接在打开文件列表中查找并操作即可，便于判断文件是否被占用。

**4.**

文件拥有者允许读、写、执行(7)，文件所在组只允许读、执行(5)，其他用户也只允许读、执行(5)。

**5.**

问题：大文件存储需要**移动已有文件**，开销很大；文件长度难以**增长**。

解决方案：分为等大小的块，用类似**链表**的方式管理。

**6.**

优点：**集中**保存了下一个块的信息，降低对文件的随机访问所需要访问的块数。

最大问题：**碎片化**，每个文件的最后一个块很可能用不满，占用内存较多。

**7.**

**a.**

先访问根目录数据，从中找到a的inode，访问inode后访问子目录a的数据，找到b的inode，访问inode后访问子目录b的数据，找到c的inode，访问inode后访问文件c的数据，共7次。

**b.**

此时a.中三次对inode的访问都不需要disk I/O操作，因此共4次。

**8.**

一个块可以储存个指针，直接映射能储存个比特的数据，一级间接映射能储存个比特的数据，同理二级间接映射能储存个比特的数据，三级间接映射能储存个比特的数据，求和可知文件大小最大可以为70403120791552比特。

**9.**

硬链接：对已有文件的另一个目录入口(即**另一个路径**)，删除文件时需要删除所有链接。

符号链接：储存已有文件路径的新数据块，已有路径的**快捷方式**。

**10.**

数据日志：写日志->日志提交->检查点->释放

元数据日志：写数据->写元数据日志(可与前同时)->日志提交->元数据检查点->释放

区别：数据日志中数据块在磁盘上写入了**两次**，元数据日志不在日志中写入数据块，只需要写入**一次**。

**11.**

轮询：对I/O的**每个字节**等待直到标识忙碌的比特位为0

中断：I/O设备可发送可屏蔽或不可屏蔽的**中断**，在每条指令执行后检查

DMA：将数据在I/O设备与内存中**直接通信**

**12.**

I/O子系统功能：I/O调度、缓冲、缓存、处理错误、假脱机、I/O保护、电源管理等