--5. synchronization

Semaphore mutex = 1;

void philosopher (int i) { while(TRUE) { think(); take_forks(i); eat(); put_forks(i); } }

void take_forks(int i) { down(&mutex); state[i] = HUNGRY; test(i);// try to get 2 forks up(&mutex); down(&s[i]); }

void test(i) { if (state[i] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING) { state[i] = EATING; up(&s[i]); } }

--6. Mem-management strategies

Fragmentation:

External Fragmentation: Tổng không gian bộ nhớ tồn tại để đáp ứng yêu cầu, nhưng không liên tục.

Internal Fragmentation: Bộ nhớ được phân bổ có thể lớn hơn một chút so với yêu cầu.

1. For each of the following decimal virtual addresses, compute the virtual page number and offset for a 4-KB page: 20000, 32768, 60000.

   4KB → 2^12 bytes → offset = last 12 bits, virtual page number = address >> 12

   20000 → 0x4E20. p = 0x4, d = 0xE20

2. A machine has 48-bit virtual addresses and 32-bit physical addresses. Pages are 8 KB. How many entries are needed for the page table?

   Virtual address space = physical address space * virtual page (= page size = 2^offset = 8KB)

   m = (m-n) + n

   m – n = 48 – 13 = 35 (bits) → 2^35 entries

3. A computer with a 32-bit address uses a two-level page table. Virtual addresses are split into a 9-bit top-level page table field, an 11-bit second-level page table field, and an offset. How large are the pages and how many are there in the address space?

   32-bit address, 2-level page table → 9-11-12 (bits) → 12 offset → page size = 2^12

   Virtual address space = 9 + 11 = 20 (bits) → 2^20 pages in the virtual address space

4. If FIFO page replacement is used with four page frames and eight pages, how many page faults will occur with the reference string 0172327103 if the four frames are initially empty? Now repeat this problem for LRU.

String: 0172327103, 4 frames

FIFO	0	1	7	2	3	2	7	1	0	3
Fr1	0				3					x
Fr2		1						x	0	
Fr3			7				x			
Fr4				2		x				

Hit = 4, miss = 6 → page fault = 6

LRU	0	1	7	2	3	2	7	1	0	3
Fr1	0				3				0	
Fr2		1						x		
Fr3			7				x			
Fr4				2		x				3

LRU: hit = 3, miss = 7

FIFO (First-In-First-Out): Thay thế trang được đưa vào bộ nhớ đầu tiên.

Optimal Algorithm: Thay thế trang sẽ không được sử dụng trong thời gian dài nhất.

LRU (Least Recently Used): Thay thế trang ít được sử dụng gần đây nhất.

Second-Chance (Clock): Sử dụng bit tham chiếu để quyết định trang nào cần thay thế.

2 dàn khỉ:

```
Semaphore mutex = 1;
Semaphore okToCross = 1;
int countEast = 0;
int countWest = 0;
int currentDirection = NONE;

void monkeyWantsToCross(direction D) {
    wait(mutex);
    while (currentDirection != NONE && currentDirection != D) {
        signal(mutex);
        wait(okToCross); // chờ khi không đúng chiều
        wait(mutex);
    }

    if (currentDirection == NONE) {
        currentDirection = D;
    }

    if (D == EAST) countEast++;
    else countWest++;

    signal(mutex);
}

void monkeyCrosses(direction D) {
    // đi qua dây
}

void monkeyLeaves(direction D) {
    wait(mutex);
    if (D == EAST) countEast--;
    else countWest--;

    if ((D == EAST && countEast == 0) || (D == WEST && countWest == 0)) {
        currentDirection = NONE;
        signal(okToCross); // cho chiều còn lại đi
    }

    signal(mutex);
}
```

--6+. Virtual memory management

Prepaging: Giảm số lượng lỗi trang xảy ra khi khởi động quá trình bằng cách tải trước các trang cần thiết.

Page Size: Lựa chọn kích thước trang phải cân nhắc đến phân mảnh, kích thước bảng, và độ cục bộ.

TLB Reach: Khả năng truy cập bộ nhớ từ TLB, bao gồm việc tăng kích thước trang hoặc cung cấp nhiều kích thước trang.

Program Structure: Cấu trúc chương trình có thể ảnh hưởng đến số lượng lỗi trang, ví dụ như cách truy cập mảng hai chiều.

I/O Interlock: Các trang được sử dụng cho I/O phải được khóa để tránh bị thay thế bởi thuật toán thay thế trang.

Bạn có muốn tìm hiểu chi tiết hơn về bất kỳ

--7. disk scheduling
Common Algorithms:

FCFS: Processes requests in the order they arrive.

Elevator (SCAN): Moves the disk arm in one direction, servicing requests, then reverses direction.

C-SCAN (Circular SCAN): Similar to SCAN but only services requests in one direction, then returns to the start.

SSTF (Shortest Seek Time First): Selects the request closest to the current disk arm position.

C-LOOK: Similar to C-SCAN but only goes as far as the last request in each direction

RAID (Redundant Array of Independent Disks):

RAID 0: Striping: Phân chia dữ liệu giữa các ổ đĩa. Ưu điểm: Hiệu suất cao. Nhược điểm: Không có dự phòng dữ liệu. RAID 1: Mirroring: Sao chép dữ liệu giữa các ổ đĩa. Ưu điểm: Độ tin cậy cao. Nhược điểm: Dung lượng giảm (chỉ sử dụng được một nửa). RAID 2: Bit-level striping và mã sửa lỗi Hamming (ECC). Ưu điểm: Tính toàn vẹn dữ liệu cao. Nhược điểm: Chi phí cao, phức tạp. RAID 3: Byte-level striping và một ổ đĩa chuyên dụng lưu trữ thông tin parity. Ưu điểm: Hiệu suất tốt cho đọc/ghi tuần tự. Nhược điểm: Hiệu suất kém cho đọc/ghi ngẫu nhiên. RAID 4: Block-level striping và một ổ đĩa chuyên dụng lưu trữ thông tin parity. Ưu điểm: Hiệu suất đọc ngẫu nhiên tốt. Nhược điểm: Hiệu suất ghi ngẫu nhiên thấp. RAID 5: Block-level striping và phân phối thông tin parity giữa các ổ đĩa. Ưu điểm: Hiệu suất cao, dự phòng tốt. Nhược điểm: Phức tạp hơn RAID 0 và RAID 1.

--10. File system

-Single-Level Directory

Structure: A single directory for all users.

Problem: Difficulties with naming and grouping files.

-Two-Level Directory

Structure: Each user has a separate directory.

Advantages: More efficient searching, allows the same file name for different users.

Disadvantages: No ability to group files.

-Tree-Structured Directory

Structure: A tree-shaped directory that allows file grouping.

Advantages: Efficient searching, supports absolute and relative paths.

Disadvantages: More complex management.

-Acyclic-Graph Directory

Structure: Allows directories and files to be shared.

Advantages: Supports aliasing (alternative names) and solves the dangling pointer problem.

Disadvantages: Requires management of reverse pointers and solutions for the dangling pointer problem.

-General Graph Directory

Structure: Allows links to files and directories.

Advantages: More flexible in managing links.

Disadvantages: Requires loop detection algorithms to ensure no cycles.

-Comparison

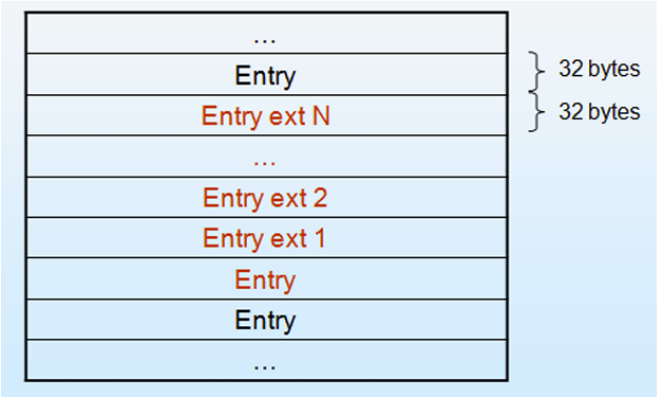
Single-Level vs Two-Level: Single-Level is simpler but faces naming issues, while Two-Level solves this problem but lacks file grouping capabilities.

Tree-Structured vs Acyclic-Graph: Tree-Structured is easier to manage, but Acyclic-Graph allows for file and directory sharing.

General Graph: The most flexible but complex in managing loops

A

Long file name



Entry ext

Offset	# byte	mean
0	1	Entry order (start at 1)
1	A (10d)	5 Unicode characters –UTF16
B (11d)	1	If entry ext (= **0Fh**)
E (14d)	C (12d)	6 unicode characters
1C (28d)	4	2 unicode characters

Phương pháp	Phân mảnh nội	Phân mảnh ngoại	Ghi chú ngắn gọn
Liên tục	✅ Có	✅ Có	Cần 1 đoạn liên tục trên đĩa.
Danh sách liên kết	❌ Không	✅ Có	Dữ liệu phân mảnh vật lý.
Chỉ mục (Indexed)	❌ Không	✅ Có	Dễ truy xuất ngẫu nhiên, nhưng vẫn phân mảnh ngoại.

Queue: 3, 5, 7
Starting head position: 4

FCFS movements: 1 + 2 + 2 = **5 cylinders**

SSTF movements: 1 + 2 + 4 = **7 cylinders**

C-SCAN movements: 1 + 2 + 3 + 3 = **9 cylinders**

/EXTSUP`1.TPL, FILE, CLUSTER: 2, 3, 4

/DIRENTRY.TPL, FILE, CLUSTER: 5

/TMC1, DIR, CLUSTER: 6

  /., DIR, CLUSTER: 6

  /.., DIR, CLUSTER: 0

  /TIMEZONE.DAT, FILE, CLUSTER: 9, 10, 11

  / FILETY`1.TXT, FILE, CLUSTER: 12, 13, 14, 15

/TMC2, DIR, CLUSTER: 8

  /., DIR, CLUSTER: 8

  /.., DIR, CLUSTER: 0

Boot Sector Interpretation

- 00-02: eb 3c 90 Instructions to jump to boot code
- 03-0a: 4d 53 44 4f 53 35 2e 30
- Name string (MSDOS5.0)
- 0b-0c: 00 02 Bytes/sector (0x0200 = 512)
- 0d : 01 Sectors/cluster (1)
- 0e-0f: 01 00 Size of reserved area (1 sector)
- 10 : 02 Number of FATs (2)
- 11-12: e0 00 Max. number of root directory entries (0x00e0 = 2
- 13-14: 40 0b Total number of sectors (0x0b40 = 2,880)
- 15 : f0 Media type (removable)
- 16-17: 09 00 FAT size (0x0009 = 9 sectors)
- 18-19: 12 00 Sectors/track (0x0012 = 18)
- 1a-1b: 02 00 Number of heads (0x0002 = 2)
- 1c-1f: 00 00 00 00 Number of sector before partition (0)
- 20-23: 00 00 00 00 Total number of sectors (0 because 2B value not equal 0)
- 24 : 00 Drive number (0)
- 25 : 00 Unused
- 26 : 29 Extended boot signature
- 27-2a: cf cd b1 c4 Volume serial number (C4B1-CDCF)
- 2b-35: 4e 4f 20 4e 41 4d 45 20 20 20 20
- Volume label ("NO NAME ")
- 36-3d: 46 41 54 31 32 20 20 20
- File system type label ("FAT12 ")
- 3e-1fd : [snip] Not used
- 1fe-1ff: 55 aa Signature value (0xaa55)

Root Directory SFN Entry Data Structure

Bytes	Purpose
0	First character of file name (ASCII) *or* allocation status (0x00=unallocated, 0xe5=deleted)
1-10	Characters 2-11 of the file name (ASCII); the "." is implied between bytes 7 and 8
11	File attributes (see File Attributes table)
12	Reserved
13	File creation time (in tenths of seconds)*
14-15	Creation time (hours, minutes, seconds)*
16-17	Creation date*
18-19	Access date*
20-21	High-order 2 bytes of address of first cluster (0 for FAT12/16)*
22-23	Modified time (hours, minutes, seconds)
24-25	Modified date
26-27	Low-order 2 bytes of address of first cluster
28-31	File size (0 for directories)

File Attributes

Flag Value	Description
0000 0001 (0x01)	Read-only
0000 0010 (0x02)	Hidden file
0000 0100 (0x04)	System file
0000 1000 (0x08)	Volume label
0000 1111 (0x0f)	Long file name
0001 0000 (0x10)	Directory
0010 0000 (0x20)	Archive

** Bytes 13-22 are unused by DOS*