HĐH - Ôn tập cuô´i kỳ - Part 3 (Bài tập)

LƯU BIÊU NGHỊ · THỨ TƯ, 19 THÁNG 6, 2019 ·

#StudyWithMe #HĐH

Chào các bạn !

Sau phâ`n lý thuyê´t, hôm nay chúng mình sẽ cùng ôn lại một sô´ bài tập nhé !

Bài tập chương 5.

Câu 1 : Xét giải pháp phâ`n mê`m do Dekker đê` nghị để tổ chức truy xuâ´t độc quyê`n cho 2 tiê´n trình. Hai tiê´n trình P0 và P1 chia sẻ các biê´n sau :

- Biê´n flag : Là một Array [0...1] với kiểu dữ liệu boolean (khởi tạo tâ´t cả giá trị đê`u là false).
- Biê´n turn : Nhận 2 giá trị 0 hoặc 1 (thể hiện lượt chạy của process 1 hoặc 2).

Xét 2 tiê´n trình kí hiệu là i và j. Câ´u trúc một tiê´n trình Pi (hiển nhiên i có thể nhận giá trị 0 hoặc 1, và j nhận giá trị còn lại) như sau :

```
var flag: array [0..1] of boolean;
turn: 0..1;
repeat

        flag[i] := true;
        while flag[j] do
                if turn = j then
                begin
                        flag[i] := false;
                        while turn = j do no-op;
                        flag[i] := true;
                end;

                critical section

        turn := j;
        flag[i] := false;

                remainder section

until false;
```

Cấu trúc chương trình.

Giải pháp có thoả 3 yêu cầu trong việc giải quyết tranh chấp không ?

Câu 2 : Xét giải pháp đồng bộ hoá sau :

```
while (TRUE) {
int j = 1-i;
flag[i]= TRUE;
turn = i;
while (turn == j && flag[j]==TRUE);
critical-section ();
flag[i] = FALSE;
   Noncritical-section ();
}
```

Giải pháp đồng bộ hoá.

Giải pháp trên có thoả yêu cầu độc quyền truy xuất (Mutual Exclusion) không ?

Câu 3 : Xét hai tiến trình sau :

```
process A  {while (TRUE)   na = na +1;      }
process B  {while (TRUE)   nb = nb +1;      }
```

Tiến trình A và B.

a. Đồ`ng bộ hoá xử lý của 2 tiế´n trình trên, sử dụng 2 semaphore tổng quát, sao cho tại bấ´t kỳ thời điểm nào cũng có nb <= na <= nb + 10, ban đầ`u giá trị của na và nb đề`u bằ`ng 0.

b. Nế´u giảm điề`u kiện chỉ có là na <= nb + 10, giải pháp của bạn sẽ được sửa chữa như thế´ nào ?

c. Giải pháp của bạn có còn đúng nế´u có nhiề`u tiế´n trình loại A và B cùng thực hiện ?

Câu 4 : Xét 2 tiế´n trình xử lý đoạn chương trình sau :

process P1 { A1 ; A2 }
process P2 { B1 ; B2 }

Tiế´n trình P1 và P2.

Đồ`ng bộ hoá hoạt động của 2 tiế´n trình này sao cho cả A1 và B1 đề`u hoàn tấ´t trước khi A2 và B2 bắ´t đầ`u.

Câu 5 : Cho tiế´n trình có đoạn chương trình sau :

process P1 { for (i = 1; i <= 100; i ++) A_i }
process P2 { for (j = 1; j <= 100; j ++) B_j }

Tiế´n trình P1 và P2.

Đồ`ng bộ hoá hoạt động của 2 tiế´n trình này sao cho với k bấ´t kỳ (2<=k<=100), Ak chỉ có thể bắ´t đầ`u khi B(k-1) đã kế´t thúc và Bk chỉ có thể bắ´t đầ`u khi A(k-1) đã kế´t thúc.

Câu 6 :

Một biế´n X được chia sẻ bởi hai tiế´n trình cùng thực hiện đoạn code sau:

```
do{

X = X +1;

if ( X == 20) X = 0;

}while ( TRUE );
```

Bắ´t đầ`u với giá trị X = 0, chứng tỏ rằ`ng giá trị X có thể vượt quá 20. Sửa lại đoạn code trên để giá trị của X không vượt quá 20.

Câu 7:

Một hãng sản xuất xe đạp có các bộ phận sản xuất hoạt động song song:

- Bộ phận sản xuất khung xe

void SXKhung(){

printf("San xuat khung");

}

- Bộ phận sản xuất bánh xe

void SXBanhXe(){

printf("San xuat banh xe");

}

- Bộ phận lắp ráp: Sau khi có đủ 1 khung và 2 bánh thì tiến hành lắp ráp.

void LapRapXe(){

printf("Lap rap xe");

}

Hãy đồng bộ hoạt động của các bộ phận trên theo nguyên tắc: tại mỗi thời điểm chỉ cho phép sản xuất 1 khung xe, cần chờ đủ 2 bánh xe để gắn vào khung xe hiện tại này trước khi sản xuất một khung xe khác.

Bài tập chương 6.

Bài 1 : Cho 1 hệ thống có 4 tiến trình P1 đến P4 và 3 loại tài nguyên R1 (3), R2(2), R3(2). P1 giữ 1 R1 và yêu cầu 1 R2, P2 giữ 2 R2 và yêu cầu 1 R1 và 1 R3, P3 giữ 1 R1 và yêu cầu 1 R2, P4 giữ 2 R3 và yêu cầu 1 R1.

- Vẽ đồ thị tài nguyên cho hệ thống này ?
- Deadlock ?
- Tìm chuỗi an toàn (nếu có) ?

Bài 2 :

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

Yêu cầu tài nguyên của các process.

Sử dụng thuật toán Banker :

- Tìm Need ?
- Hệ thống có an toàn không ?
- Nếu P1 yêu cầu (0,4,2,0) thì có thể cấp phát cho nó ngay không ?

Bài 3 : Sử dụng thuật toán Banker xem các trạng thái sau có an toàn hay không ? Nếu có thì đưa ra chuỗi thực thi an toàn, nếu không thì nêu rõ lý do không an toàn ?

	Allocation	*Max*
	A B C D	*A B C D*
P_0	3 0 1 4	5 1 1 7
P_1	2 2 1 0	3 2 1 1
P_2	3 1 2 1	3 3 2 1
P_3	0 5 1 0	4 6 1 2
P_4	4 2 1 2	6 3 2 5

Bài 3.

a. Available = (0,3,0,1).

b. Available = (1,0,0,2).

Bài 4 : Trả lời các câu hỏi sau sử dụng giải thuật Banker.

	Allocation	*Max*
	A B C D	*A B C D*
P_0	3 0 1 4	5 1 1 7
P_1	2 2 1 0	3 2 1 1
P_2	3 1 2 1	3 3 2 1
P_3	0 5 1 0	4 6 1 2
P_4	4 2 1 2	6 3 2 5

Bài 4.

a. Hệ thố´ng có an toàn không ? Đưa ra chuỗi an toàn nê´u có ?

b. Nê´u P1 yêu cầ`u (1,1,0,0) thì có thể cấ´p phát cho nó ngay không ?

Bài tập chương 7.

Bài 1 : Xét một không gian địa chỉ có 12 trang, mỗi trang có kích thước 2K, ánh xạ vào bộ nhớ vật lý có 32 khung trang.

a. Địa chỉ logic gồ`m bao nhiêu bit ?

b. Địa chỉ physic gồ`m bao nhiêu bit ?

c. Bảng trang có bao nhiêu mục ? Mỗi mục trong bảng trang cầ`n bao nhiêu bit ?

Bài 2 : Xét một hệ thố´ng sử dụng kỹ thuật phân trang, với bảng trang được lưu trữ trong bộ nhớ chính.

a. Nê´u thời gian cho một lầ`n truy xuấ´t bộ nhớ bình thường là 200ns thì mấ´t bao nhiêu thời gian cho một thao tác truy xuấ´t bộ nhớ trong hệ thố´ng này ?

b. Nê´u sử dụng TLBs với hit-ratio là 75%, thời gian để tìm tròn TLBs xem như bằ`ng 0, tính thời gian truy xuấ´t bộ nhớ trong hệ thố´ng.

Bài 3 : Xét bảng phân đoạn trong hình :

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Bảng phân đoạn bài 3.

Tính địa chỉ vật lý tương ứng với các địa chỉ logic sau đây :

a. 0,430.

b. 1,10.

c. 2,500.

d. 3,400.

e. 4,112.

Bài 4 : Xét một không gian có bộ nhớ luận lý có 15 trang, mỗi trang có 1024 từ, mỗi từ là 2 byte được ánh xạ vào bộ nhớ vật lý có 32 trang, trả lời các câu hỏi sau :

a. Địa chỉ bộ nhớ vật lý có bao nhiêu bit ?

b. Địa chỉ bộ nhớ luận lý có bao nhiêu bit ?

c. Có bao nhiêu mục trong bảng phân trang ? Mỗi mục chứa bao nhiêu bit ?

Bài tập chương 8.

Bài 1 : Xét chuỗi truy xuất bộ nhớ sau :

1, 2, 3, 4, 3, 5, 1, 6, 2, 1, 2, 3, 7, 5, 3, 2, 1, 2, 3, 6.

Có bao nhiêu lỗi trang xảy ra khi sử dụng các thuật toán thay thế sau đây, biết có 4 khung trang.

a. LRU.

b. FIFO.

c. Chiến lược tối ưu (OPT).

Bài 2 : Một máy tính 32-bit địa chỉ, sử dụng một bảng trang 2 cấp. Địa chỉ ảo được phân bổ như sau : 9 bit dành cho bảng trang cấp 1, 11 bit cho bảng trang cấp 2, và còn lại cho offset. Cho biết kích thước một trang trong hệ thống và địa chỉ ảo có bao nhiêu trang ?

Bài 3 : Giả sử địa chỉ ảo 32-bit được phân tách thành 4 trường a,b,c,d. 3 trường đầu tiên được dùng cho bảng trang 3 cấp, trường thứ 4 dành cho offset. Số lượng trang có phụ thuộc vào kích thước của cả 4 trường này không ? Nếu không, những trường nào ảnh hưởng đến số lượng trang, những trường nào không ảnh hưởng ?

==========

Bài giải:

Chương 5:

Câu 1: *Cơ sở lý thuyết: 3 tính chất của một giải thuật giải quyết tranh chấp:

1) Loại trừ tương hỗ (Mutual exclusion): Khi một tiến đang thực thi trong vùng tranh chấp của nó thì không có process Q nào khác đang thực thi trong CS của Q.

2) Một tiế n trình tạm dừng bên ngoài miề n găng không được ngăn cản các tiế n trình khác vào miề n găng

3) Bounded waiting: Mỗi process chỉ phải chờ để được vào vùng tranh chấ p trong một khoảng thời gian có hạn định nào đó. Không xảy ra tình trạng đói tài nguyên.

Giải:

- Quan sát trên giải thuật:

```
var flag: array [0..1] of boolean;
turn: 0..1;
repeat

        flag[i] := true;
        while flag[j] do
                if turn = j then
                begin
                        flag[i] := false;
                        while turn = j do no-op;
                        flag[i] := true;
                end;

                critical section

        turn := j;
        flag[i] := false;

                remainder section

until false;
```

Giải thuật.

- Nhận thấ y: nế u một tiế n trình Pj muố n vào vùng tranh chấ p, trong khi tiế n trình Pi cũng đang muôn vào vùng tranh chấ p: =>dựa vài biế n turn để xác nhận Pi hay Pj được tiế n hành, do turn chỉ có một giá trị cùng một lúc nên chỉ có một trong 2 được vào vùng tranh chấ p của nó => thoả mãn tính chấ t 1.

- Không có bấ t kỳ động thái ngăn cản tiế n trình Pj vào miề n găng khi Pi đang không ở trong miề n găng (như đổi turn từ j thành i, đổi flag[j] thành false) => thoả mãn tính chấ t 2.

- Khi Pi hoàn thành thì sẽ nhường lượt cho Pj, nế u Pi bị tắ c thì Pj sẽ tiế n và nhường lượt cho Pi => thoả mãn điề u kiện 3.

Bài 2: Ta thấ y code này có đoạn j = 1 – i => đề chỉ đang nói tới xét 2 tiê trình P1 và P0 vì khi i = 0 thì j = 1 và ngược lại.

- Giải thuật này không thoả mãn:

Xét tình huống khi flag[0] =1; turn =0; lúc này P0 vào CS, Nếu lúc đó flag[1] = 1, P1 có thể gán turn = 1 và vào luôn CS (2 tiến trình cùng vào CS một lúc).

Câu 3:

```
process A  {while (TRUE)  na = na +1;     }
process B  {while (TRUE)  nb = nb +1;     }
```

Đề bài câu 3.

a) nb <= na <= nb + 10.

Cách giải: Chúng ta phải làm 2 việc với 2 semaphore đã cho: đảm bảo na >= nb và na <= nb + 10, như vậy nghĩa là khi na = nb thì tiến trình B bị block cho tới khi A tiến hành được ít nhất một lần, cũng như khi na = nb + 10 thì A bị block cho tới khi B tiến hành được ít nhất một lần.

```
Semaphore_1 = 0;
Semaphore_2 = 10;
Process A:
while (1)
    {
        wait(Semaphore_2);
        na = na + 1;
        signal(Semaphore_1);
    }
Process B:
while (1)
    {
        wait(Semaphore_1);
        nb = nb + 1;
        signal(Semaphore_2);
}
b)
Semaphore_2 = 10;
Process A:
while (1)
    {
        wait(Semaphore_2);
        na = na + 1;
    }
Process B:
while (1)
    {
        nb = nb + 1;
        signal(Semaphore_2);
}
```

Source bài giải.

c) Đúng, vì có thể có nhiều tiến trình loại A hoặc loại B cùng thực hiện nhưng chỉ có 2 biến Semaphore toàn cục mà chúng sẽ thao tác (đối với câu b là 1 Semaphore).

Câu 4:

```
Semaphore_1 = 0;
Semaphore_2 = 0;
process P1 {
A1 ;
wait(Semaphore_1);
signal(Semaphore_2);
A2 ;
}
process P2 {
B1 ;
wait(Semaphore_2);
Signal(Semaphore_1);
B2 ;
}
```

Source bài giải.

Câu 5:

```
Semaphore_1 = 1;
Semaphore_2 = 1;
Process A:
for(int i = 1; i <= 100; i++)
{
wait(Semaphore_1);
Ai;
signal(Semaphore_2)
}
Process B:
for( i = 1; i <= 100; i++)
{
Wait(Semaphore_2);
Bi;
signal(Semaphore_1);
}
```

Source bài giải.

Câu 6:

do{

X = X +1;

if (X == 20) X = 0;

}while (TRUE);

Do X được chia sẻ chung ở cả hai tiế n trình và chỉ bị reset khi X == 20 nên X có thể vượt quá 20 khi:

- Có một lý do trong máy đột ngột khiế n tiế n trình 2 dừng lại. Sau đó tại tiế n trình 1, khi X = 19 thì tiế n trình 2 được release đúng ngay đoạn X = X + 1, và cộng dồ n với X = 20 sau lệnh X = X + 1 của tiế n trình 1 dẫn tới X vượt quá 20 và còn bị cộng tới vô cùng.

*Đây là code demo với cách giả định P2 bị crash và release bằ ng một lệnh sleep nhỏ, kế t quả của X sẽ vượt quá 20 trong tích tắ

```
#include "stdio.h"
#include "semaphore.h"
#include "pthread.h"
#include "signal.h"
#include <stdlib.h>
#include <unistd.h>

pthread_t ThreadA;
pthread_t ThreadB;

sem_t Semaphore_1;
sem_t Semaphore_2;

int isLoop = 1;
int X = 0;

void *FunctionA(void *data)
{
    while (1)
    {
        X++;
        printf("%d\n", X);
        if (X == 20)
        {
            X = 0;
        }
    }
}

void *FunctionB(void *data)
{
    while (1)
    {
        sleep(0.1); // Crash
        X++;
        printf("%d\n", X);
        if (X == 20)
        {
            X = 0;
        }
    }
}
```

Code demo (1).

```
int main()
{
    sem_init(&Semaphore_1, 1, 0);
    sem_init(&Semaphore_2, 1, 10);
    pthread_create(&ThreadB, NULL, &FunctionB, NULL);
    pthread_create(&ThreadA, NULL, &FunctionA, NULL);
    while (isLoop)
    {
    }
    return 0;
}
```

Code demo (2).

- Để giải quyết vấn đề này (làm cho X luôn nhỏ hơn 20) thì ta có thể giải quyết bằng cách đưa 2 tiến trình về làm 1, tức là làm cho cùng một lúc chỉ có một trong hai tiến trình được chạy bằng một biến Semaphore khởi tạo bằng 1.

Câu 7:

Semaphore_Khung = 0;

Semaphore_Banh = 0,;

Semaphore_LapRap = 1;

Process Khung:

while(true){

wait(Semaphore_LapRap);

printf(“San xuat khung”);

signal(Semaphore_Khung);

}

}

Process BanhXe:

while(true){

printf(“San xuat banh xe”);

signal(Semaphore_Banh);

}

Process LapRapXe:

while(true){

down(Semaphore_Khung);

down(Semaphore_Banh);

down(Semaphore_Banh);

printf("Lap rap xe");

signal(Semaphore_LapRap);

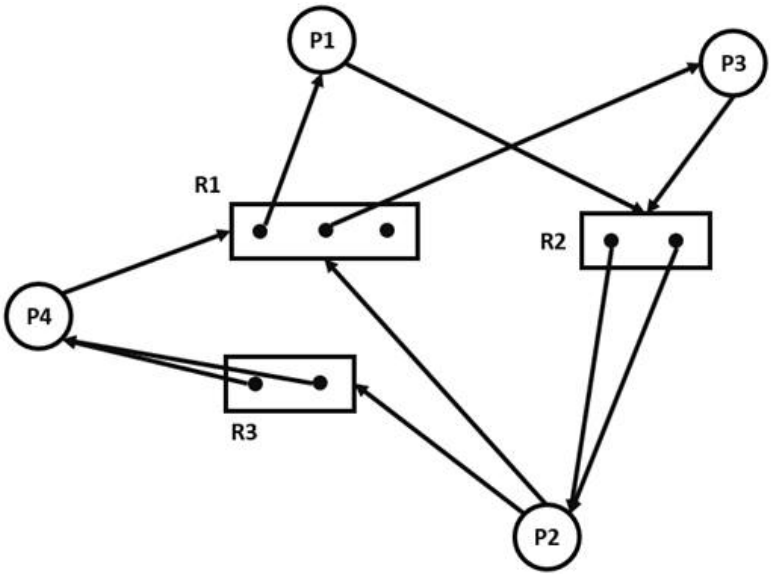
}

Chương 6:

Câu 1 : Cho 1 hệ thô´ng có 4 tiê´n trình P1 đê´n P4 và 3 loại tài nguyên R1 (3), R2(2), R3(2). P1 giữ 1 R1 và yêu cầ`u 1 R2, P2 giữ 2 R2 và yêu cầ`u 1 R1 và 1 R3, P3 giữ 1 R1 và yêu cầ`u 1 R2, P4 giữ 2 R3 và yêu cầ`u 1 R1.

- Vẽ đồ` thị tài nguyên cho hệ thô´ng này ?
- Deadlock ?
- Tìm chuỗi an toàn (nê´u có) ?

a)



Đáp án câu a.

b)

	Allocation			Max			Need			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	1	0	0	1	1	0	0	1	0	1	0	0
P2	0	2	0	1	2	1	1	0	1	1	0	2
P3	1	0	0	1	1	0	0	1	0	1	2	2
P4	0	0	2	1	0	2	1	0	0	2	2	2

Bảng yêu cầu tài nguyên của các tiến trình.

c) Chuỗi an toàn: P4 -> P2 -> P3 -> P1

Câu 2:

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

Đề bài câu 2.

Sử dụng thuật toán Banker :

- Tìm Need ?
- Hệ thống có an toàn không ?
- Nếu P1 yêu cầu (0,4,2,0) thì có thể cấp phát cho nó ngay không ?

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2	0	0	0	0	1	5	2	0
P1	1	0	0	0	1	7	5	0	0	7	5	0	1	5	3	2
P2	1	3	5	4	2	3	5	6	1	0	0	2	2	8	8	6
P3	0	6	3	2	0	6	5	2	0	0	2	0	2	14	11	8
P4	0	0	1	4	0	6	5	6	0	6	4	2	2	14	12	12

Bảng yêu cầu tài nguyên.

=> Chuỗi an toàn: P0 -> P2 -> P3 -> P4 -> P1

=> Hệ thống an toàn.

Nê´u P1 yêu cầ`u (0,4,2,0):

Ta có: Request1 <= Need1

Request1 <= Available

Giả sử cấ´p phát tài nguyên cho P1 thành công, ta có:

Available = Available – Request1 = (1,1,0,0)

Need1 = Need1 - Request1 = (0,3,3,0)

Allcation1 = Allocation1 + Request1 = (1,4,2,0)

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2	0	0	0	0	1	1	0	0
P1	1	4	2	0	1	7	5	0	0	3	3	0	1	1	1	2
P2	1	3	5	4	2	3	5	6	1	0	0	2	2	4	6	6
P3	0	6	3	2	0	6	5	2	0	0	2	0	3	8	8	6
P4	0	0	1	4	0	6	5	6	0	6	4	2	3	8	9	10

Bảng yêu cầ`u tài nguyên.

Sau khị cấ´p phát ta có chuỗi an toàn là: P0 -> P2 -> P1 -> P4 -> P3 => Có thể cấ´p phát được.

Câu 3 : Kiểm tra xem các trạng thái sau có an toàn hay không ? Nê´u có thì đưa ra chuỗi thực thi an toàn, nê´u không thì nêu rõ lý do không an toàn ?

	Allocation	*Max*
	A B C D	*A B C D*
P_0	3 0 1 4	5 1 1 7
P_1	2 2 1 0	3 2 1 1
P_2	3 1 2 1	3 3 2 1
P_3	0 5 1 0	4 6 1 2
P_4	4 2 1 2	6 3 2 5

Đề` câu 3.

a. Available = (0,3,0,1).

b. Available = (1,0,0,2).

a)

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	3	0	1	4	5	1	1	7	2	1	0	3	0	3	0	1
P1	2	2	1	0	3	2	1	1	1	0	0	1	3	4	2	2
P2	3	1	2	1	3	3	2	1	0	2	0	0	5	6	3	2
P3	0	5	1	0	4	6	1	2	4	1	0	2	5	11	4	2
P4	4	2	1	2	6	3	2	5	2	1	1	3				

Đáp án câu a.

=> Trạng thái không an toàn do không tìm được giá trị Needi nhỏ hơn Available = (5,11,4,2)

b)

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	3	0	1	4	5	1	1	7	2	1	0	3	1	0	0	2
P1	2	2	1	0	3	2	1	1	1	0	0	1	3	2	1	2
P2	3	1	2	1	3	3	2	1	0	2	0	0	6	3	3	3
P3	0	5	1	0	4	6	1	2	4	1	0	2	6	8	4	3
P4	4	2	1	2	6	3	2	5	2	1	1	3	10	10	5	5

Đáp án câu b.

=> Chuỗi an toàn: P1 -> P2 -> P3 -> P4 -> P0

=> Trạng thái an toàn

Bài 4 : Trả lời các câu hỏi sau sử dụng giải thuật Banker.

a. Hệ thống có an toàn không ? Đưa ra chuỗi an toàn nếu có ?

b. Nếu P1 yêu cầu (1,1,0,0) thì có thể cấp phát cho nó ngay không ?

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	2 0 0 1	4 2 1 2	3 3 2 1
P_1	3 1 2 1	5 2 5 2	
P_2	2 1 0 3	2 3 1 6	
P_3	1 3 1 2	1 4 2 4	
P_4	1 4 3 2	3 6 6 5	

Đề bài 4.

a)

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	2	0	0	1	4	2	1	2	2	2	1	1	3	3	2	1
P1	3	1	2	1	5	2	5	2	2	1	3	1	5	3	2	2
P2	2	1	0	3	2	3	1	6	0	2	1	3	6	6	3	4
P3	1	3	1	2	1	4	2	4	0	2	1	2	7	10	6	6
P4	1	4	3	2	3	6	6	5	2	2	3	3	10	11	8	7

Đáp án câu a.

Chuỗi an toàn: P0 -> P3 -> P4 -> P1 -> P2

b)

Ta có: Request1 <= Need1

Request1 <= Available

Giả sử cấp phát tài nguyên cho P1 thành công:

Available = Available – Request1 = (2,2,2,1)

Need1 = Need1 – Request1 = (1,0,3,1)

Allocation1 = Allocation1 + Request1 = (4,2,2,1)

	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	2	0	0	1	4	2	1	2	2	2	1	1	2	2	2	1
P1	4	2	2	1	5	2	5	2	1	0	3	1	4	2	2	2
P2	2	1	0	3	2	3	1	6	0	2	1	3	5	5	3	4
P3	1	3	1	2	1	4	2	4	0	2	1	2	6	9	6	6
P4	1	4	3	2	3	6	6	5	2	2	3	3	10	11	8	7

Đáp án câu b.

Chuỗi an toàn: P0 -> P3 -> P4 -> P1 -> P2

Chương 7:

Câu 1 : Xét một không gian địa chỉ có 12 trang, mỗi trang có kích thước 2K, ánh xạ vào bộ nhớ vật lý có 32 khung trang.

a. Địa chỉ logic gồm bao nhiêu bit ?

b. Địa chỉ physic gồm bao nhiêu bit ?

c. Bảng trang có bao nhiêu mục ? Mỗi mục trong bảng trang cần bao nhiêu bit ?

Giải:

a)

Ta có: 12 <= 24 => cần 4 bit để biểu diễn số trang.

2K <= 211 => cần 11 bit để biểu diễn độ dời trong 1 trang

ở Cần 15 bit để biểu diễn địa chỉ logic

b)

Ta có: 32 <= 25 => cần 5 bit để biểu diễn số khung trang

ở Cần 16 bit để biểu diễn

c) Bảng trang có 16 mục, mỗi mục 5 bit.

Bài 2 : Xét một hệ thống sử dụng kỹ thuật phân trang, với bảng trang được lưu trữ trong bộ nhớ chính.

a. Nếu thời gian cho một lần truy xuất bộ nhớ bình thường là 200ns thì mất bao nhiêu thời gian cho một thao tác truy xuất bộ nhớ trong hệ thống này ?

b.Nếu sử dụng TLBs với hit-ratio là 75%, thời gian để tìm tròn TLBs xem như bằng 0, tính thời gian truy xuất bộ nhớ trong hệ thống.

Giải:

a) Thời gian truy xuất trong bộ nhớ trong hệ thống đã cho: 2x200 = 400 ns

b) Nếu sử dụng TLBs với hit-ratio là 75%:

(200 + 0) x 0.75 + (200 + 200 + 0) x 0.25 = 250

Bài 3 : Xét bảng phân đoạn trong hình :

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Segment table.

Tính địa chỉ vật lý tương ứng với các địa chỉ logic sau đây :

a) 0,430. => 219 + 430 = 649

b) 1,10. => 2300 + 10 = 2310

c) 2,500. => Không hợp lệ

d) 3,400. => 1327 + 400 = 1727

e) 4,112. => Không hợp lệ

Bài 4 : Xét một không gian có bộ nhớ luận lý có 15 trang, mỗi trang có 1024 từ, mỗi từ là 2 byte được ánh xạ vào bộ nhớ vật lý có 32 khung trang, trả lời các câu hỏi sau :

a) Địa chỉ bộ nhớ vật lý có bao nhiêu bit ?

b) Địa chỉ bộ nhớ luận lý có bao nhiêu bit ?

c) Có bao nhiêu mục trong bảng phân trang ? Mỗi mục chứa bao nhiêu bit ?

Giải:

a) Ta có: 32 <= 25 => cần 5 bit biểu diễn số khung trang,

1024 WORD = 2048 bytes <= 211 => cần 11 bit biểu diễn độ dời

Cần 16 bit để biểu diễn bộ nhớ vật lý.

b) Ta có: 15 <= 24 => cần 4 bit để biểu diễn số trang.

=> cần 15 bit để biểu diễn địa chỉ logic

c)

Có 16 mục trong bảng phân trang, mỗi mục 5 bit.

Chương 8:

Bài 1 : Xét chuỗi truy xuất bộ nhớ sau :

1, 2, 3, 4, 3, 5, 1, 6, 2, 1, 2, 3, 7, 5, 3, 2, 1, 2, 3, 6.

Có bao nhiêu lỗi trang xảy ra khi sử dụng các thuật toán thay thế sau đây, biết có 4 khung trang.

a. LRU.

b. FIFO.

c. Chiến lược tối ưu (OPT).

a)

a)

	1	2	3	4	3	5	1	6	2	1	2	3	7	5	3	2	1	2	3	6
0	1	1	1	1	1	5	5	5	5	5	5	3	3	3	3	3	3	3	3	3
1		2	2	2	2	2	1	1	1	1	1	1	1	5	5	5	5	5	5	6
2			3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2
3				4	4	4	4	6	6	6	6	6	7	7	7	7	1	1	1	1
	*	*	*	*		*	*	*	*			*	*	*			*			*

⇨ Có 13 lỗi trang

Đáp án câu a.

b)

	1	2	3	4	3	5	1	6	2	1	2	3	7	5	3	2	1	2	3	6
0	1	1	1	1	1	5	5	5	5	5	5	3	3	3	3	3	3	2	2	2
1		2	2	2	2	2	1	1	1	1	1	1	7	7	7	7	7	7	3	3
2			3	3	3	3	3	6	6	6	6	6	6	5	5	5	5	5	5	6
3				4	4	4	4	4	2	2	2	2	2	2	2	2	1	1	1	1
	*	*	*	*		*	*	*	*			*	*	*			*	*	*	*

Đáp án câu b.

=> Có 15 lỗi trang

c)

	1	2	3	4	3	5	1	6	2	1	2	3	7	5	3	2	1	2	3	6
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	6
1		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3				4	4	5	5	6	6	6	6	6	7	5	5	5	5	5	5	5
	*	*	*	*		*		*					*	*						*

Đáp án câu c.

=> Có 9 lỗi trang.

Bài 3 : Giả sử địa chỉ ảo 32-bit được phân tách thành 4 trường a,b,c,d. 3 trường đầ`u tiên được dùng cho bảng trang 3 cấ´p, trường thứ 4 dành cho offset. Sô´ lượng trang có phụ thuộc vào kích thước của cả 4 trường này không ? Nê´u không, những trường nào ảnh hưởng đê´n sô´ lượng trang, những trường nào không ảnh hưởng ?

Giải:

Sô´ lượng trang chỉ phụ thuộc vào kích thước trường d và bă`ng: 2(32 – d)

Các trường a, b, c có thể thay đổi kích thước nhưng vẫn không làm thay đổi sô´ trang nê´u trường d không thay đổi.

Bài 4: Giả sử có một hệ thô´ng sử dụng kỹ thuật phân trang theo yêu cầ`u. Bảng trang được lưu trữ trong các thanh ghi. Để xử lý một lỗi trang tô´n 8 miliseconds nê´u có sẵn một khung trang trô´ng, hoặc trang bị thay thê´ không bị sửa đổi nội dung, và tô´n 20 miliseconds nê´u trang bị thay thê´ bị sửa đổi nội dung. Mỗi truy xuấ´t bộ nhớ tô´n 100 nanoseconds. Giả sử trang bị thay thê´ có xác suấ´t bị sửa đổi là 70%.

Tỷ lệ phát sinh lỗi trang phải là bao nhiêu để có thể duy trì thời gian truy xuấ´t bộ nhớ (effective acess time) không vượt quá 200 nanoseconds ?

Gọi tỉ lệ xảy ra lỗi trang cầ`n tìm là r

Do bảng trang được lưu trữ trong các thanh ghi => thời gian truy xuấ´t vào bàng trang không đáng kể.

=> Thời gian truy xuấ´t bộ nhớ: 100 nanoseconds

– Thời gian trung bình để xử lý lỗi trang: 0,3 * 8 + 0,7 * 20 = 16.4 miliseconds

– Theo đề`: EAT <= 200 nanoseconds

=> EAT = 100 + r*16.4 <= 200

=> r <= (200-100)/16,4 = 0,609%

Bài giải được thực hiện bởi : **Nguyễn Văn Đông.**

Phù, đây là bài viế´t cuố´i cùng nă`m trong chuỗi bài StudyWithMe HĐH rô`i. Bài này tụi mình post chậm hơn lịch dự kiế´n một ngày để có thời gian chuẩn bị kỹ hơn cho các bạn.

Cảm ơn các bạn đã theo dõi chuỗi bài viế´t StudyWithMe. Hy vọng sẽ được tiế´p tục gặp lại các bạn ở các hoạt động tiế´p theo của BHT !

Chúc các bạn thi thật tốt !

Nếu có bất kỳ thắc mắc hoặc sai sót nào, các bạn có thể góp ý tại comment bên dưới nhé !

Xem ôn tập cuối kỳ - Lý thuyết (part 1) : http://bit.ly/2KsBhE4

Xem ôn tập cuối kỳ - Lý thuyết (part 2) : http://bit.ly/2FnJt4C