#### P1 进程的前驱后继问题

```
semaphore a,b,c,d,e,f,g,h;
a=b=c=d=e=f=g=h=0;
void S1(){S1 execute; signal(a); signal(b);}
void S2(){wait(a); S2 execute; signal(c); signal(d);}
void S3(){wait(b); S3 execute; signal(e);}
void S4(){wait(c); S4 execute; signal(f);}
void S5(){wait(d); S5 execute; signal(g);}
void S6(){wait(e); S5 execute; signal(h);}
void S7(){wait(f); wait(g); wait(h); S7 execute;}
```

## P2 银行家算法

(1) 系统处于安全状态, 理由如下:

当前状态,进程的需求矩阵如下:

$$Need = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 1 & 7 & 5 & 0 \\ 2 & 3 & 5 & 6 \\ 0 & 6 & 5 & 2 \\ 0 & 6 & 5 & 6 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 \\ 1 & 3 & 5 & 4 \\ 0 & 6 & 3 & 2 \\ 0 & 0 & 1 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 7 & 5 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 2 & 0 \\ 0 & 6 & 4 & 2 \end{bmatrix}$$
(1)

银行家算法计算如下:

		_	_	_			_	_	_					
	Α	В	С	D		Α	В	C	D					
p1	0	0	0	0	p1	0	0	1	2			D	_	ь.
p2	0	7	5	0	p2	1	0	0	0		A	В	C	D
р3	1	0	0	2	p3	1	3	5	4		1	5	2	0
p4	0	0	2	0	p4	0	6	3	2			Avai	iable	;
p5	0	6	4	2	p5	0	0	1	4					
	Г	Need	J.			Alle	ocat	ion						
							et p	1						
						9	er þ							
	Α	В	С	D		Α	В	С	D					
р1	0	0	0	0	р1	0	0	0	0					
p2	0	7	5	0	p2	1	0	0	0		Α	В	С	D
p3	1	0	0	2	р2 р3	1	3	5	4		1	5	3	2
p4	0	0	2	0	р3 р4	0	6	3	2			Avai		
p5	0	6	4	2	р <del>т</del>	0	0	1	4					
23		Need	-	_	25		ocat	_	7					
						g	et p	3						
	Α	В	С	D		Α	В	С	D					
р1	0	0	0	0	р1	0	0	0	0					
p2	0	7	5	0	p2	1	0	0	0		Α	В	С	D
рЗ	0	0	0	0	р3	0	0	0	0		2	8	8	6
p4	0	0	2	0	p4	0	6	3	2			Avai	lable	9
p5	0	6	4	2	p5	0	0	1	4					
	1	Need	1			Alle	ocat	ion						
						g	et p	4						
	Α	В	С	D		Α	В	С	D					
p1	0	0	0	0	р1	0	0	0	0					
p2	0	7	5	0	p2	1	0	0	0		Α	В	С	D
рЗ	0	0	0	0	рЗ	0	0	0	0		2	14	11	8
p4	0	0	0	0	р4	0	0	0	0			Avai	lable	
p5	0	6	4	2	p5	0	0	1	4					
	1	Need	t			Alle	ocat	ion						
						g	et p	2						
		_		_			_	_	_					
	Α	В	С	D		Α	В	С	D					
p1	0	0	0	0	p1	0	0	0	0			_	_	_
p2	0	0	0	0	p2	0	0	0	0		A	В	C	D
р3	0	0	0	0	р3	0	0	0	0		3	14		8
p4	0	0	0	0	p4	0	0	0	0			Avai	iable	,
p5	0	6	4	2	p5	0	0	1	4					
	Г	Need	1			Alle	ocat	ion						
							<u> </u>							
						g	et p							
	Α	В	С	D		Α	В	С	D					
	0	0	0	0	p.1	0	0	0	0					
p. 1	-	0	0	0	p1	0	0	0	0		٨	В	С	D
p1	0	0	0	0	p2		0	0	0		A 3	_	_	_
p2	0	U		_	р3 р4	0	0	0	0		3	14 Avai		
p2 p3	0		Λ.				U.	U	U	i l		_vdl	navit	
p2 p3 p4	0	0	0	0					_					
p2 p3	0		0	0	р <del>4</del>	0	0 ocat	0	0					

#### (2) 如果进程P2提出需要(0,4,2,0)个资源的请求,系统可以去满足它的请求。理由如下:

A B C D	A B C D		A B C D	A B C D	
p1 0 0 0 0	p1 0 0 1 2		p1 0 0 0 0	p1 0 0 1 2	
p2 0 7 5 0	p2 1 0 0 0	A B C D B get (0, 4, 2, 0)	p2 0 3 3 0	p2 1 4 2 0	A B C D
p3 1 0 0 2	p3 1 3 5 4	1 5 2 0	p3 1 0 0 2	p3 1 3 5 4	1 1 0 0
p4 0 0 2 0	p4 0 6 3 2	Available	p4 0 0 2 0	p4 0 6 3 2	Available
p5 0 6 4 2	p5 0 0 1 4		p5 0 6 4 2	p5 0 0 1 4	
Need	Allocation		Need	Allocation	

	Α	В	С	D			Α	В	С	D				
p1	0	0	0	0		р1	0	0	1	2				
р1 p2	0	3	3	0		p2	1	4	2	0	Α	В	С	D
р2 p3	1	0	0	2		р2 p3	1	3	5	4	1	1	0	0
•						•						_	ilable	
p4	0	0	2	0		p4	0	6	3	2		۸۷al	IIdDIE	8
p5	0	6	4	2	-	p5	0	0	1	4	-			
	ı	Nee	u				Allo	ocat	ion					
							а	et p	1					
							9	-						
	Α	В	С	D			Α	В	С	D				
р1	0	0	0	0		р1	0	0	0	0				
p2	0	3	3	0		p2	1	4	2	0	Α	В	С	D
р3	1	0	0	2		р3	1	3	5	4	1	1	1	2
p4	0	0	2	0		p4	0	6	3	2			ilable	
p5	0	6	4	2		p5	0	0	1	4				
		Nee	-	_		25		ocat						
							g	et p	3					
		_	_	_				_	_	_				
_	Α	В	С	D			Α	В	С	D				
p1	0	0	0	0		p1	0	0	0	0				L_
р2	0	3	3	0		p2	1	4	2	0	Α	В	С	D
р3	0	0	0	0		рЗ	0	0	0	0	2	4	6	6
р4	0	0	2	0		p4	0	6	3	2		Avai	ilable	9
р5	0	6	4	2		p5	0	0	1	4				
	- 1	Nee	d				Allo	ocat	ion					
							g	et p	94					
							_	•						
	Λ	D	_	_						_				
_4	Α	В	С	D			Α	В	С	D				
•	0	0	0	0		p1	A 0	B 0	C 0	0				-
p1 p2	0	0	0	0		p2	A 0	B 0 4	C 0	0	A	В	С	D
p2 p3	0 0	0 3 0	0 3 0	0 0		p2 p3	A 0 1	B 0 4 0	C 0 2	0 0	2	10	9	8
p2 p3 p4	0 0 0	0 3 0	0 3 0	0 0 0		p2 p3 p4	A 0 1 0	B 0 4 0	C 0 2 0	0 0 0	2	10		8
p2 p3 p4	0 0 0 0	0 3 0 0	0 3 0 0 4	0 0		p2 p3	A 0 1 0 0 0	B 0 4 0 0	C 0 2 0 0	0 0	2	10	9	8
p2 p3 p4	0 0 0 0	0 3 0	0 3 0 0 4	0 0 0		p2 p3 p4	A 0 1 0 0 0	B 0 4 0	C 0 2 0 0	0 0 0	2	10	9	8
o2 o3 o4	0 0 0 0	0 3 0 0	0 3 0 0 4	0 0 0		p2 p3 p4	A 0 1 0 0 0 Allo	B 0 4 0 0 0	C 0 2 0 0 1	0 0 0	2	10	9	8
o2 o3 o4	0 0 0 0	0 3 0 0	0 3 0 0 4	0 0 0		p2 p3 p4	A 0 1 0 0 0 Allo	B 0 4 0 0	C 0 2 0 0 1	0 0 0	2	10	9	8
p2 p3 p4	0 0 0 0	0 3 0 0	0 3 0 0 4	0 0 0		p2 p3 p4	A 0 1 0 0 0 Allo	B 0 4 0 0 0	C 0 2 0 0 1	0 0 0	2	10	9	8
p2 p3 p4 p5	0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4	0 0 0 0 2		p2 p3 p4 p5	A 0 1 1 0 0 0 Allo	B 0 4 0 0 0 ocat	C 0 2 0 0 1 1 ion	0 0 0 0 4	2	10	9	8
o2 o3 o4 o5	0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 d	0 0 0 0 2 D		p2 p3 p4 p5	A 0 1 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 2 0 0 1 1 iion C C 0	0 0 0 0 4	2	10 Avai	9 ilable	8
p2 p3 p4 p5 p1 p2	0 0 0 0 0	0 3 0 6 Need	0 3 0 0 4 d	0 0 0 0 2 2 D 0		p2 p3 p4 p5 p1 p2	A 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 2 0 0 1 1 ion C C 0 0 0	0 0 0 4 4	2 A	10 Avai	9 ilable	8 P
p2 p3 p4 p5 p1 p2 p3	0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0		p2 p3 p4 p5 p1 p2 p3	A 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion C 0 0 0 0 0	0 0 0 0 4	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4	0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0		p2 p3 p4 p5 p1 p2 p3 p4	A 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 2 0 0 1 1 ion C 0 0 0 0 0 0	0 0 0 0 4 4 D 0 0 0	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4	0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0		p2 p3 p4 p5 p1 p2 p3	A 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 0 1 1 0 0 0 0 1 1	0 0 0 0 4	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4	0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0		p2 p3 p4 p5 p1 p2 p3 p4	A 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 0 1 1 0 0 0 0 1 1	0 0 0 0 4 4 D 0 0 0	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4	0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0		p2 p3 p4 p5 p1 p2 p3 p4	A 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 0 0 0 1 1 ion	0 0 0 0 4 4 D 0 0 0	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4	0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2		p2 p3 p4 p5 p1 p2 p3 p4	A 0 0 0 Allo 9 9	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion 0 1 1 ion 0 5 5	0 0 0 0 4	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4 p5	0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0 2		p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 Allo g	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion 0 0 1 1 ion C C C C C C C C C C C C C C C C C C C	0 0 0 0 4	A 3	B 14	9 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4 p5	0 0 0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 D 0 0 0 2		p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 0 Allo	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion 0 0 1 1 ion 0 5 C 0 0	0 0 0 0 4	A 3	B 14 Avai	G C 11 ilable	D 8
p2 p3 p4 p5 p1 p2 p3 p4 p5	0 0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0 2 2		p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 Allo 9 A 0 0 0 0 O O O O O O O O O O O O O O O	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion 0 5 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 4 4 D 0 0 0 4	A 3	B 14 Avai	C C 111	D 8 8
p2 p3 p4 p5 p1 p2 p3 p4 p5	0 0 0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 D 0 0 0 2		p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 0 Allo	B 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion C 0 0 1 1 ion C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 4	A 3	B 14 Avai	C 11 C C 12	D 8 8 9 P P P P P P P P P P P P P P P P P
p2 p3 p4 p5 p1 p2 p3 p4 p5	0 0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 2 D 0 0 0 2 2		p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 Allo 9 A 0 0 0 0 O O O O O O O O O O O O O O O	B 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion 0 5 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 4 4 D 0 0 0 4	A 3	B 14 Avai	C C 111	D 8 8 9 P P P P P P P P P P P P P P P P P
•	0 0 0 0 0 0 0 0 0 0 0 0	0 3 0 0 6 Need	0 3 0 0 4 dd	0 0 0 0 2 D 0 0 0 2		p2 p3 p4 p5 p1 p2 p3 p4 p5 p1 p2 p3 p4 p5	A 0 0 0 Allo g	B 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	C 0 0 1 1 ion C 0 0 1 1 ion C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 4	A 3	B 14 Avai	C 11 C C 12	D 8 8 9 P P P P P P P P P P P P P P P P P

# P3 处理器调度算法

(1)

FCFS调度算法(非抢占)的平均周转时间为: 1.48 s,调度顺序按照提交时间顺序,为 $1\to 2\to 3\to 4$ ,计算如下:

		I	CFS调度算	法(非抢占	ī )	
1	作业号	提交时间	运行时间	开始时间	结束时间	周转时间
	1	10.00	1	10.00	11.00	1.00
	2	10. 10	0.6	11.00	11.60	1.50
	3	10. 20	0.3	11.60	11. 90	1.70
	4	10.30	0.1	11.90	12.00	1.70
			平均周转时	- 间		1.48

最短作业优先算法(非抢占)的平均周转时间为:  $1.30\,\mathrm{s}$ ,调度顺序按照就绪队列中的运行时间顺序,为 $1\to 4\to 3\to 2$ ,计算如下:

	最	短作业优先	算法(非拍	5占)	
作业号	提交时间	运行时间	开始时间	结束时间	周转时间
1	10.00	1	10.00	11.00	1.00
4	10.30	0.1	11.00	11. 10	0.80
3	10. 20	0.3	11. 10	11.40	1. 20
2	10.10	0.6	11.40	12.00	1. 90
		平均周转时	- 间		1.30

(2) 当CPU使用响应比高者优先算法时,第一个作业完成后,t=11s,后续作业序列按照响应比顺序,为 $4\rightarrow 3\rightarrow 2$ 。计算过程如下:

_					
	响应出	心高者优先的	算法 第一个	个作业完成	后,t=11
	作业号	提交时间	运行时间	等待时间	响应比
	2	10. 10	0.6	0.90	2.50
	3	10. 20	0.3	0.80	3.67
	4	10.30	0. 1	0.70	8.00
			平均周转时	- 间	

## P4 分页存储

由题意得到,页表大小为1024B,分页存储的内存模型示意图如下:

0		0
1		1024
2	页面1	2048
3	页面0	3072
4	页面3	4096

其中页大小为 $1024B=2^{10}B$ ,故二进制虚地址中,0~9位为页内偏移量,10~31位为页面号。

虚地址 1044 ==> 二进制  $1\,0000010100$  ,页面号为1,页内偏移量为 10100,即20。其中页面1对应2号存储块,起始地址为2048。

故实际物理地址为: 2048+20=2068。

## P5 页面置换算法

(1) 最近最少使用 (LRU) 算法:

在2块主存空间中,缺页中断18次

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	3	3	2	2	5	5	2	2	2	2	7	7	က	თ	1	1	თ	3
	2	2	4	4	1	1	6	6	1	1	3	3	6	6	2	2	2	2	6
×	×	×	×	×	×	×	×	×	×		×	×	×	×	×	×		×	×

在4块主存空间中,缺页中断10次

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6	6	6
	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
		3	3	3	3	5	5	5	5	5	3	3	3	3	3	3	3	3	3
			4	4	4	4	6	6	6	6	6	7	7	7	7	1	1	1	1
×	×	×	×			×	×				×	×	×			×			

(3) 先进先出 (FIFO) 调度算法:

在2块主存空间中,缺页中断18次

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	3	3	2	2	5	5	2	2	2	3	3	6	6	2	2	2	3	3
	2	2	4	4	1	1	6					7		3	3	1	1	1	6
×	×	×	×	×	×	×	×	×	×		×	×	×	×	×	×		×	×

在4块主存空间中,缺页中断14次

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1	1	1	5	5	5	5	5	3	3	3	3	3	1	1	1	1
	2	2	2	2	2	2	6	6	6	6	6	7	7	7	7	7	7	3	3
		3	3	3	3	3	თ	2	2	2	2	2	6	6	6	6	6	6	6
			4	4	4	4	4	4	1	1	1	1	1	1	2	2	2	2	2
×	×	×	×			×	×	×	×		×	×	×		×	×		×	