2020 年春季学期操作系统第二次作业

2018 级计算机科学与技术（中美合作）

包亦航
2018011890

2020 年 5 月

❖ 有三类资源A(17)、B(5)、C(20)。有5个进程P_1—P_5。T_0时刻系统状态如下：

	最大需求	已分配
P_1	5 5 9	2 1 2
P_2	5 3 6	4 0 2
P_3	4 0 11	4 0 5
P_4	4 2 5	2 0 4
P_5	4 2 4	3 1 4

？？？？？：

(1)T_0时刻是否为安全状态，给出安全系列。

(2)T_0时刻，P_2: Request(0,3,4)，能否分配，为什么？

(3)在(2)的基础上P_4：Request(2,0,1)，能否分配，为什么？

(4)在(3)的基础上P_1：Request(0,2,0)，能否分配，为什么？

解答：

(1)

进程 \ 资源情况	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	1	2	3	4	7
P2	5	3	6	4	0	2	1	3	4
P3	4	0	11	4	0	5	0	0	6
P4	4	2	5	2	0	4	2	2	1
P5	4	2	4	3	1	4	1	1	0

我们首先列出如下最大需求，分配，以及需求矩阵

由于三类资源的个数分别为 17，5，20，从而目前还剩下的资源，也即 available 向量为 2，3，3

我们在这里开始模拟安全性算法，初始的时候 work 等于 available 向量为 2，3，3

① 找到 P4 可分配，于是 work 变为 4，3，7

② 找到 P2 可分配，于是 work 变为 8，3，9

③ 找到 P3 可分配，于是 work 变为 12，3，14

④ 找到 P5 可分配，于是 work 变为 15，4，18

⑤ 找到 P1 可分配，于是 work 变为 17，5，20

至此所有进程都被加入到了队列，从而这是个安全状态，其中一种安全序列为 P4→P2→P3→P5→P1

(2)

根据银行家算法，我们首先检查[0,3,4]<[1,3,4]，也即 request<need，检查通过。

之后发现[0,3,4]并不完全小于[2,3,3]，也就表示尚无足够的资源，请求需要等待，无法分配资源

(3)

根据银行家算法，我们首先检查[2,0,1]<[2,2,1]，也即 request<need，检查通过。

资源情况 / 进程	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	1	2	3	4	7
P2	5	3	6	4	0	2	1	3	4
P3	4	0	11	4	0	5	0	0	6
P4	4	2	5	4	0	5	0	2	0
P5	4	2	4	3	1	4	1	1	0

第二步检查[2,0,1]<[2,3,3]，也即 request<available，表示有足够的资源
第三步我们就尝试着把资源分配给 P4，更新刚才的表格如下
并更新 available 向量为[0,3,2]
我们在这里开始模拟安全性算法，初始的时候 work 等于 available 向量为 0，3，2

① 找到 P4 可分配，于是 work 变为 4，3，7
② 找到 P2 可分配，于是 work 变为 8，3，9
③ 找到 P3 可分配，于是 work 变为 12，3，14
④ 找到 P5 可分配，于是 work 变为 15，4，18
⑤ 找到 P1 可分配，于是 work 变为 17，5，20

至此所有进程都被加入到了队列，从而这是个安全状态，其中一种安全序列为
P4→P2→P3→P5→P1
从而这种分配是可行的，正式将资源分配给 P4

(4)

在(3)的基础上，我们知道 available 是 0,3,2

资源情况 / 进程	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	3	2	3	2	7
P2	5	3	6	4	0	2	1	3	4
P3	4	0	11	4	0	5	0	0	6
P4	4	2	5	4	0	5	0	2	0
P5	4	2	4	3	1	4	1	1	0

根据银行家算法，我们首先检查[0,2,0]<[3,4,7]，也即 request<need，检查通过。
第二步检查[0,2,0]<[0,3,2]，也即 request<available，表示有足够的资源
第三步我们就尝试着把资源分配给 P1，更新刚才的表格如下
并更新 available 向量为[0,1,2]
这个时候我们就发现，在 0，1，2 情况下找不到进程加入安全序列，于是我们就说如果我们在这里进行分配的话系统就会进入不安全状态，也即这样的分配方法是不合法的，我们撤销刚才的预分配，要求这个请求命令进行等待