

1. (a) **选择 PlanA**。理由是，一般情况下优先做选择 (σ) 操作可以减少用于连接元组的数量，进一步降低连接的代价。
(b) **是**。理由是，如果已知 R 和 S 在连接属性值 k 上的分布，并且二者没有交集。
2. 可以画出原有调度的优先图如图 1，从图中可以看到在 0 和 2 中存在环路，所以**这不是一个冲突可串行化调度**。

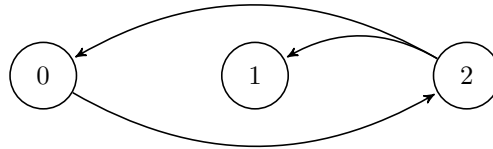


图 1: 第 2 题调度的优先图

3. 用两阶段锁保证题目 2 的调度冲突可并行化，**结果如表 1所示**。
4. **结果如表 2所示**，其中在 T2 执行完 Write(A) 时，由于 $TS(T2) < R-ts(A)$ ，所以 T2 需要回滚。
5. (a) **结果如表 3所示**。
(b) **T1 和 T2 需要 Redo, T3 和 T0 需要 Undo**。根据日志文件，在故障恢复时 T1 和 T2 已经提交，需要 Redo；而 T0 和 T3 未提交故 Undo，需要消除其影响。
(c) **需要增加 $\langle T0, abort \rangle$ 和 $\langle T3, abort \rangle$** ，表明 T0 和 T3 在故障中终止未提交。

| T0 | T1 | T2 |
|------------------|------------------|------------------|
| LOCK-X(A) | | |
| r0(A) | | |
| w0(A) | | |
| LOCK-X(B) | | |
| r0(B) | | |
| w0(B) | | |
| UNLOCK(A) | | |
| | | LOCK-X(A) |
| UNLOCK(B) | | |
| | | w2(A) |
| | | r2(A) |
| | | LOCK-X(B) |
| | | r2(B) |
| | | W2(B) |
| | | UNLCOK(A) |
| | LOCK-S(A) | |
| | | UNLOCK(B) |
| | r1(A) | |
| | LOCK-S(B) | |
| | r1(B) | |
| | UNLOCK(A) | |
| | UNLOCK(B) | |

表 1: 使用两阶段锁保证调度的冲突可并行化

| T1 | T2 | W-ts(A) | R-ts(A) | W-ts(B) | R-ts(B) |
|---------------|---------------|---------|---------------|---------------|---------------|
| | Read(B); | | | | TS(T2) |
| | B := B - 50; | | | | TS(T2) |
| | Write(B); | | | TS(T2) | TS(T2) |
| READ(B); | | | | TS(T2) | TS(T1) |
| | READ(A); | | TS(T2) | TS(T2) | TS(T1) |
| | A := A = 50; | | TS(T2) | TS(T2) | TS(T1) |
| Read(A); | | | TS(T1) | TS(T2) | TS(T1) |
| | Write(A); | 出现错误 | TS(T1) | TS(T2) | TS(T1) |
| Display(A+B); | | | TS(T1) | TS(T2) | TS(T1) |
| | Display(A+B); | | TS(T1) | TS(T2) | TS(T1) |

表 2: 时间戳

| | |
|----|-----------------------------|
| 1 | <T0,start> |
| 2 | <T0, A, 50, 70> |
| 3 | <T2,start> |
| 4 | <start checkpoint (T0, T2)> |
| 5 | <end checkpoint> |
| 6 | <T1, commit> |
| 7 | <T1, B, 30, 20> |
| 8 | <T1, commit> |
| 9 | <T2, C, 35, 70> |
| 10 | <T3, start> |
| 11 | <T3, D, 15, 30> |
| 12 | <T2, commit> |

表 3: 日志文件