

a. （1）串行执行：先T13，后T14

|  |  |
| --- | --- |
| 初始值 | A=0，B=0 |
| T13 | A=0，B=1 |
| T14 | A=0，B=1 |

此时满足一致性需求，A=0∪B=0

（2）串行执行：先T14，后T13

|  |  |
| --- | --- |
| 初始值 | A=0，B=0 |
| T13 | A=1，B=0 |
| T14 | A=1，B=0 |

此时满足一致性需求，A=0∪B=0

b.

|  |  |
| --- | --- |
| T13 | T14 |
| read(A) |  |
| read(B) |  |
|  | read(B) |
|  | read(A) |
| if A=0 then B:=B+1 |  |
| write(B) |  |
|  | if B=0 then A:=A+1 |
|  | write(A) |

由上表可知，最终A=1，B=1，不满足一致性需求A=0∪B=0。

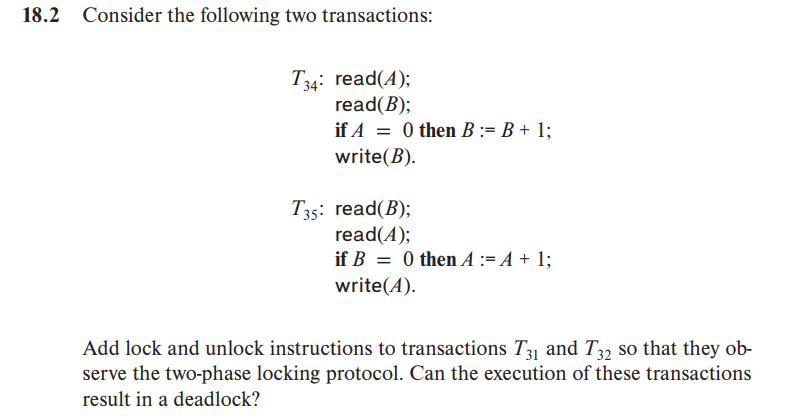
但串行化调度都满足，所以其执行产生不可串行化调度。

c. 不存在，因为串行化调度满足A=0∪B=0。而在并发执行过程中，若T13先执行read(A)，则必有B=1，而read(B)是T14的第1条语句，所以为实现并发，read(B)会出现在write(B)之前，所以读到的B必定为0，所以T14会使A=1。

若T14先执行read(B)，则同理见上。

所以，无论以什么顺序并发执行，都不能满足一致性条件

A=0∪B=0。



对于T34：

lock-S(A)

read(A)

lock-X(B)

read(B)

if A = 0

then B := B + 1

write(B)

unlock(A)

unlock(B)

对于T35：

lock-S(B)

read(B)

lock-X(A)

read(A)

if B = 0

then A := A + 1

write(A)

unlock(B)

unlock(A)

所以可列出下表：

|  |  |
| --- | --- |
| T34 | T35 |
| lock-S(A) |  |
|  | lock-S(B) |
| read(A) |  |
|  | read(B) |
| lock-X(B) |  |
|  | lock-X(A) |

由于X锁和S锁不兼容, 所以T34等待T35释放S(B)，T35则等待T34释放S(A)，导致死锁。