

Tutorat : concurrent systeme

Question

- transaction 中的读写冲突，ACID 属性，事务的【依赖图】
- concurrent control

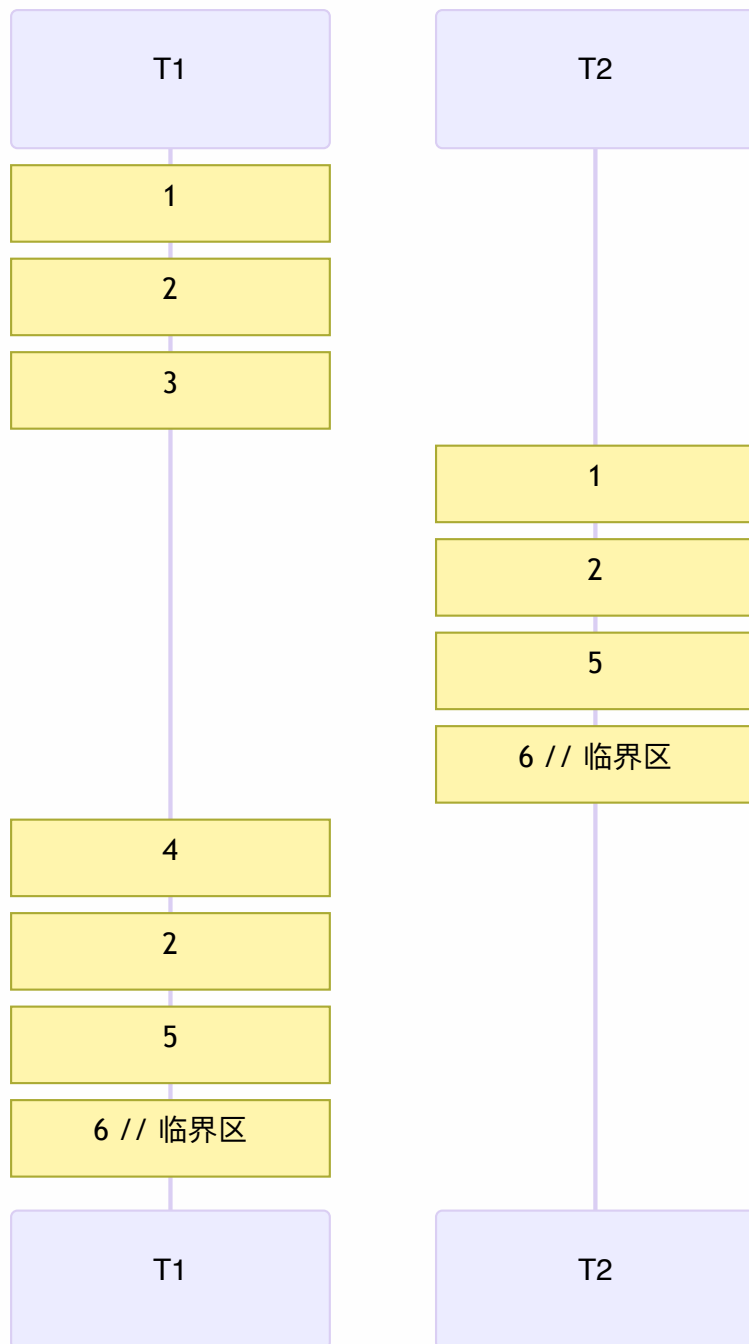
```
n := y + 1;  
t1.write(t1.read(y));
```

- 多进程的内存共享，并发，互斥

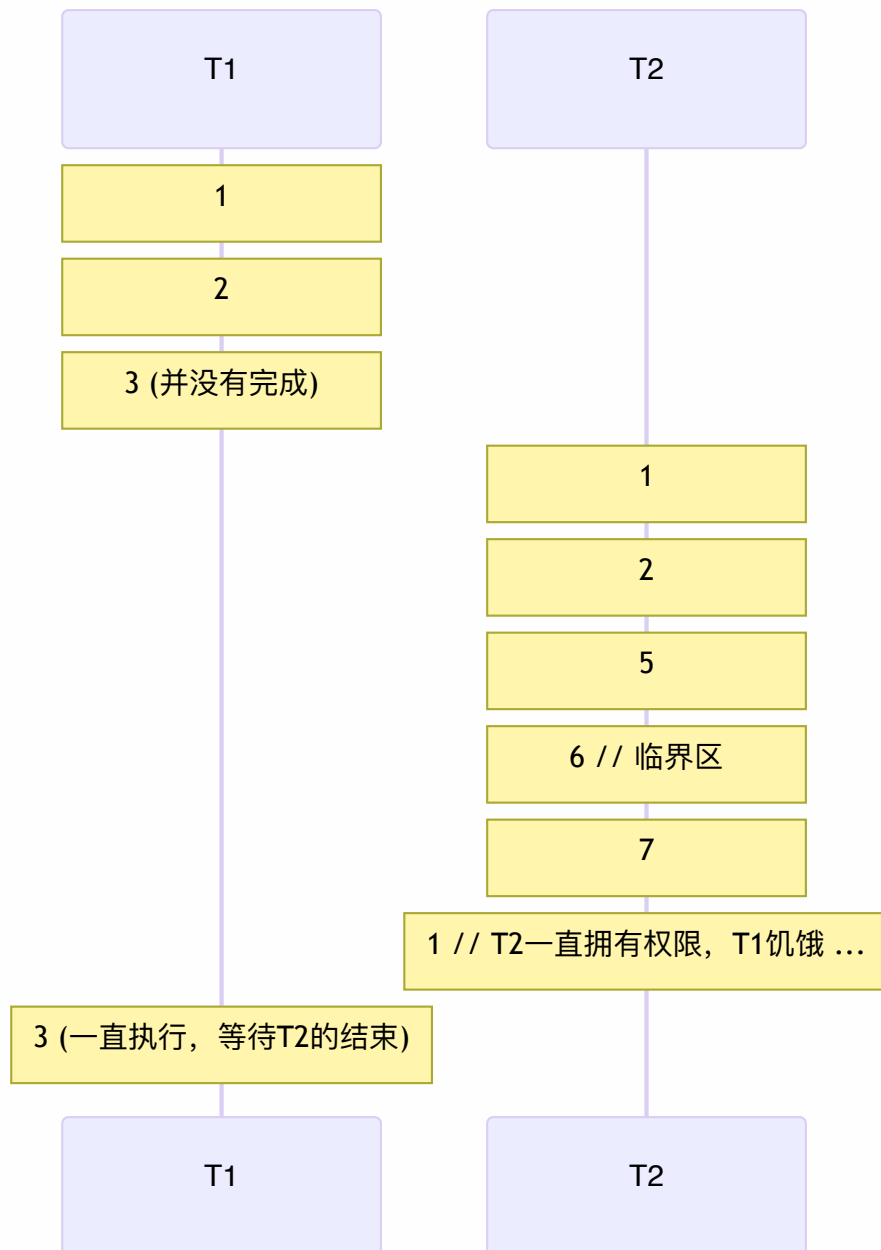
Exercise 1. Concurrent execution

```
demande[i] = true; // 1  
while (tour != i) { // 2  
    while (demand[j]) { // 3  
        // 阻塞;  
    }  
    tour <- i // 4  
} // 5  
// SECTION CRITIQUE // 6  
demande[i] = false; // 7
```

1. 描述：算法并不会确保互斥地访问临界区。



2. 描述：算法会导致饥饿



Exercise 2.

同步问题:

	marins	personnals
N	4	2
V	3	3
disponible	3	3
	0	1

```
Semaphore PE[category] = new Semaphore [0,0];  
mutex := 1 // 互斥信号量
```

```
lever_l_ancre (t){ // reviel de NB[t,M] marin et NB[t,P]  
  for(int i = 1; i < (NB[t,M] + NB[t,P]); i++){  
    monter.down();  
  }  
}
```

```
embarquer (c) {  
  PE(c).down(); // marin可以登船的信号量  
  monter.up(); //  
}
```

```
appeler_equipage (t) {  
  mutex.down();  
  if (dispo[-] >= NB[t] ){  
    for(int i = 1; i < (NB[t,M]; i++){  
      PE.[M].up();  
    }  
  } else {  
    mutex.up(); // why?  
    PLA[t].down(); // Peut_Lancer_lAncre  
  }  
}
```

```
se_presenter(c) {  
  mutex.up();  
  dispo[c]++;  
  if (dispo[-] >= NB[t]){  
    PLA[+].down();  
  }  
}
```

Exercise 3. Transaction



