Synchronization (Practice)

ICS332

Operating Systems

Henri Casanova (henric@hawaii.edu)

(q1) Race Conditions

```
int x = 0;

void f() { x++; }
void g() { x--; }
```

- Two threads:
  - A calls f() 10 times
  - B calls g() 20 times
- What are the possible values of x at the end of the program?

(q1) Answer

```
int x = 0;

void f() { x++; }
void g() { x--; }
```

- Two threads:
  - A calls f() 10 times
  - B calls g() 20 times
- What are the possible values of x at the end of the program?

- Extreme #1: ALL updates of A are lost: x = -20
- Extreme #2: ALL updates of B are lost: x = +10
- Any anything in between
- Final answer: -20, -19, …., +9, +10

(q2) How many Locks?

```
int x = 0;

void f() { x++; }
void g() { x--; }
```

- How many locks should I use to remove the race condition we saw in the previous question?

(q2) Answer

- How many locks should I use to remove the race condition we saw in the previous question?

- Answer: One lock
- Because both the x++ and x-- instruction uypdate the same variable

```
int x = 0;
lock_t lock;

void f() {
  lock.lock();
  x++;
  lock.unlock();
}
void g() {
  lock.lock();
  x--;
  lock.unlock();
}
```

(q3) How many Locks?

- Many threads call `f()` and `g()` willy-nilly
- Could I prevent all race conditions using a single lock?
- How many locks should I use for best concurrency?

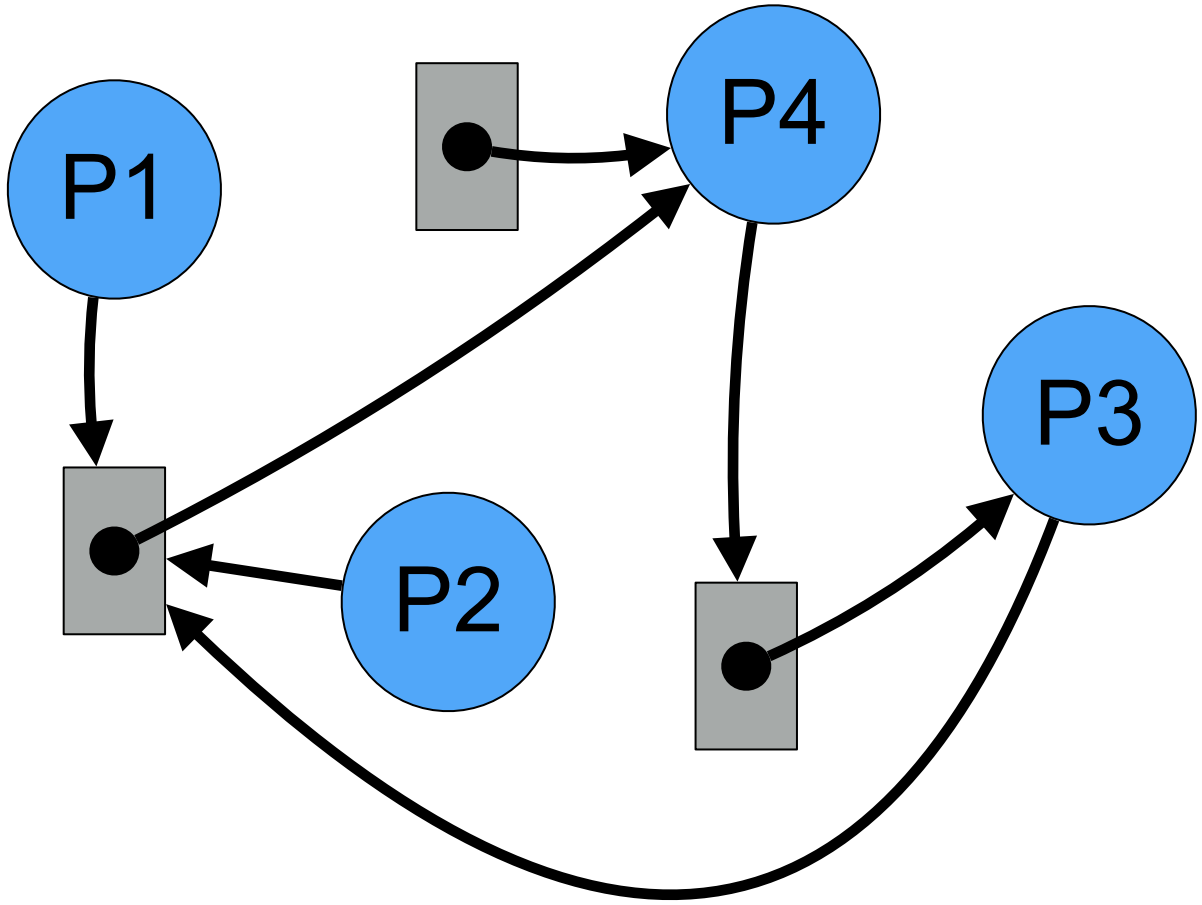
```
int a = 2;
int c = 10;
int x = 0;

void f(int x) {
  x = x * 4;
  a++;
}

void g() {
  a += c;
  c = 2;
}
```

(q3) Answer

- Many threads call `f()` and `g()` willy-nilly
- Could I prevent all race conditions using a single lock? YES, but concurrency would be low
- How many locks should I use for best concurrency? TWO locks
  - Protect updates to `x`
  - Protect updates to `a`
  - (`c` is never updated)

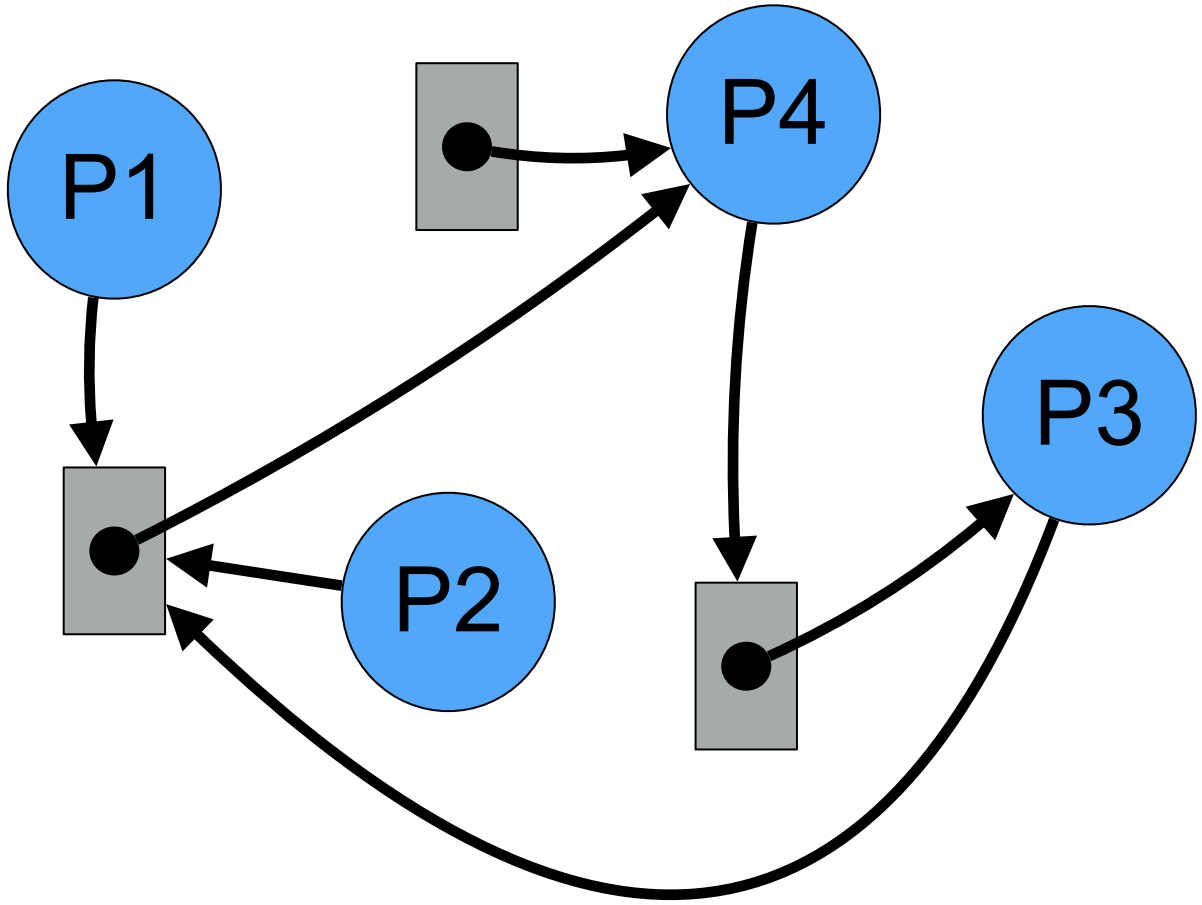
```
int a = 2;
int c = 10;
int x = 0;
lock_t lock_a;
lock_t lock_x;

void f(int x) {
  lock_x.lock();
  x = x * 4;
  lock_x.unlock();
  lock_a.lock();
  a++;
  lock_a.unlock();
}

void g() {
  lock_a.lock();
  a += c;
  lock_a.unlock();
  c = 2;
}
```

(q4) Deadlock Graph



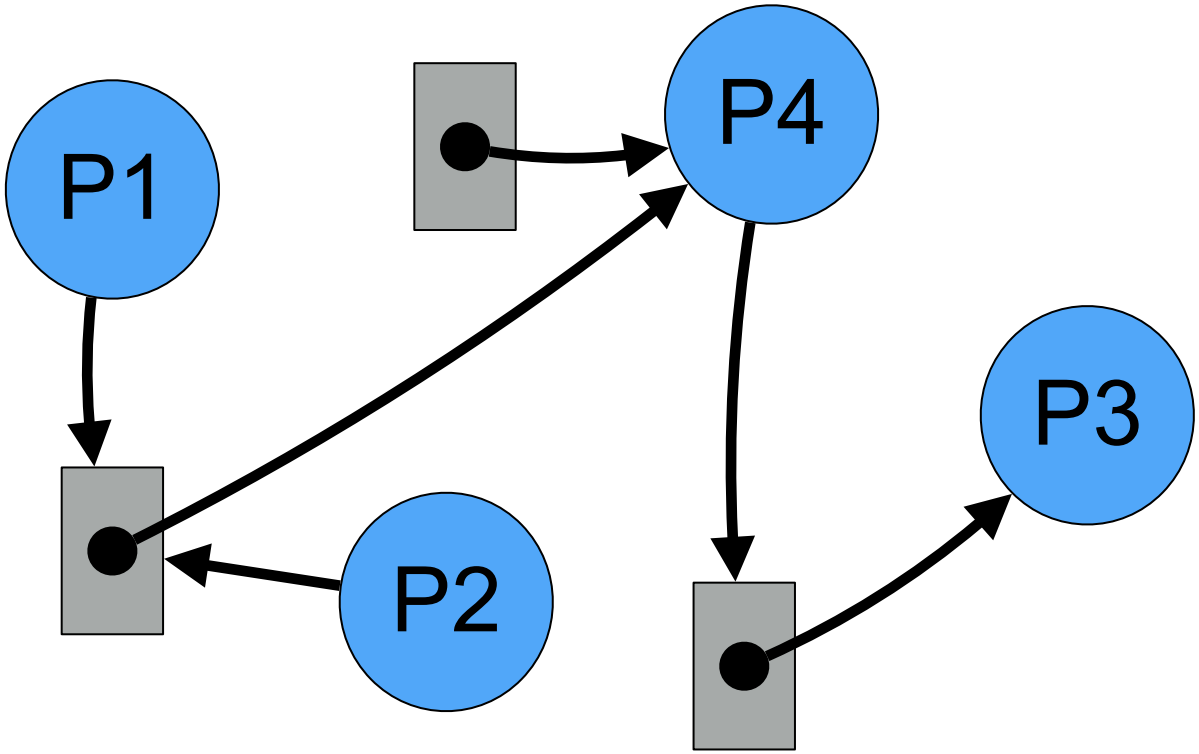
- Is there a deadlock?

(q4) Answer



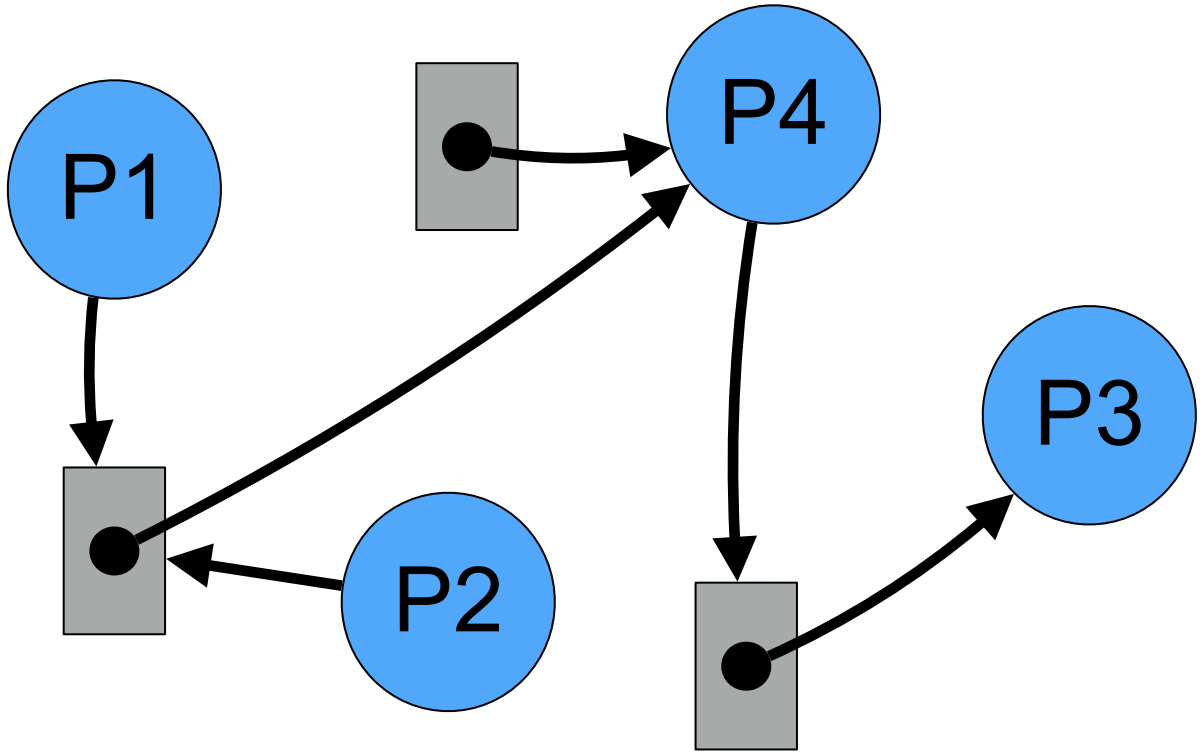
- Is there a deadlock? YES

(q4) Deadlock Graph



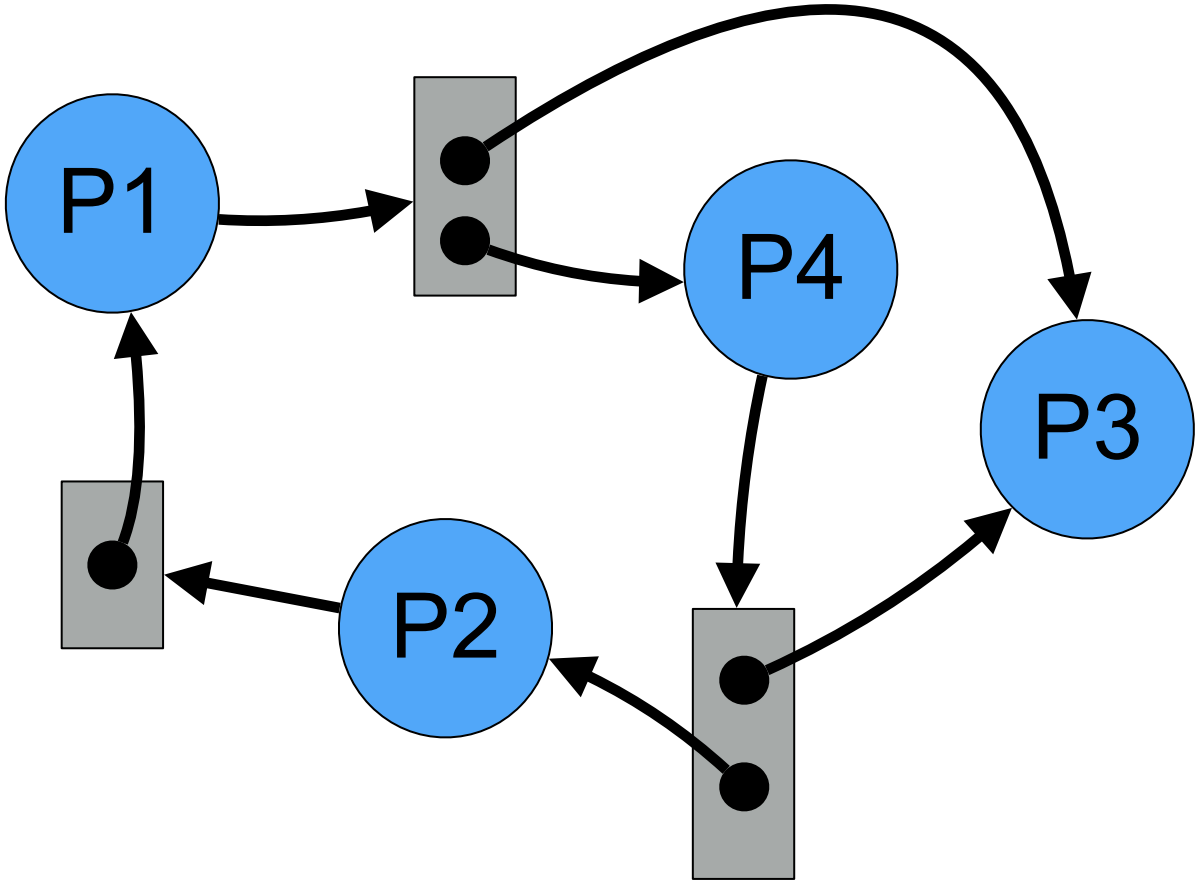
- Is there a deadlock?

(q4) Answer



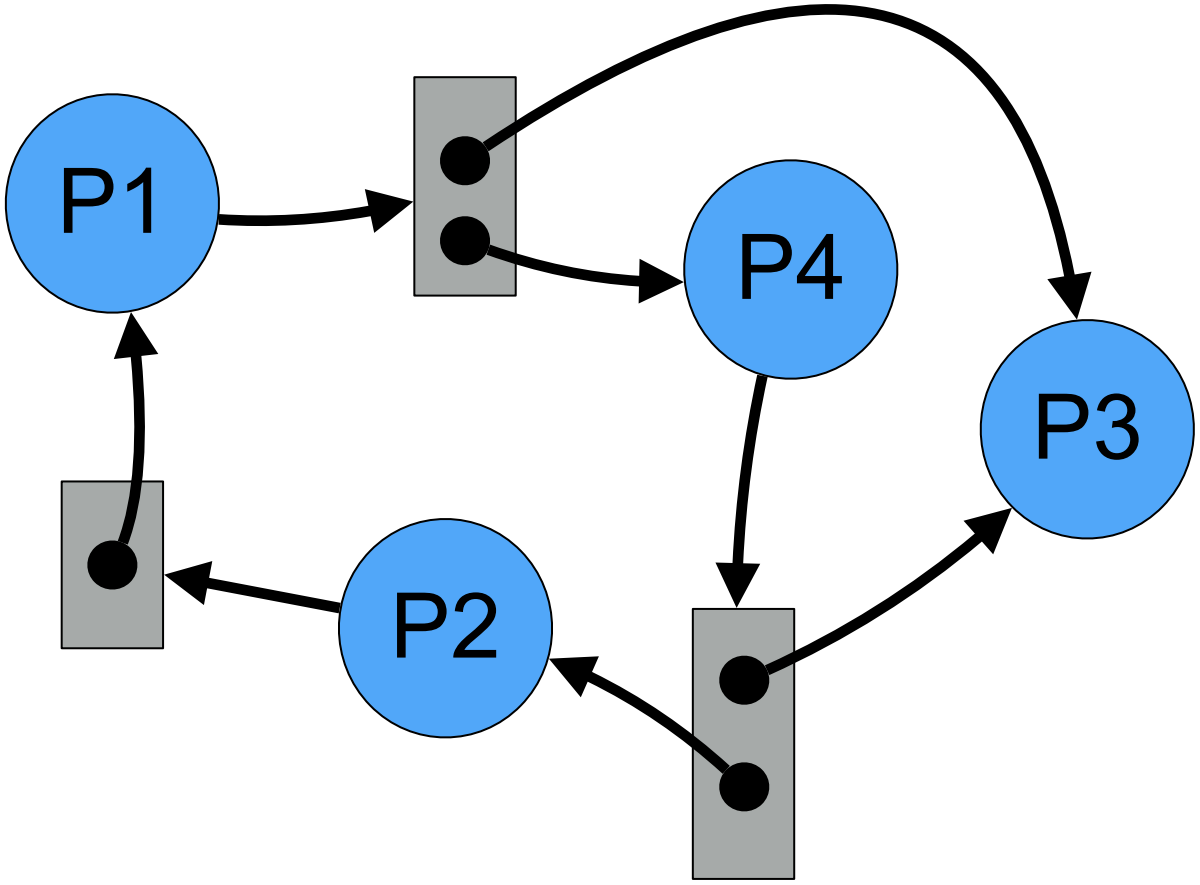
- Is there a deadlock? NO

(q5) Deadlock Graph



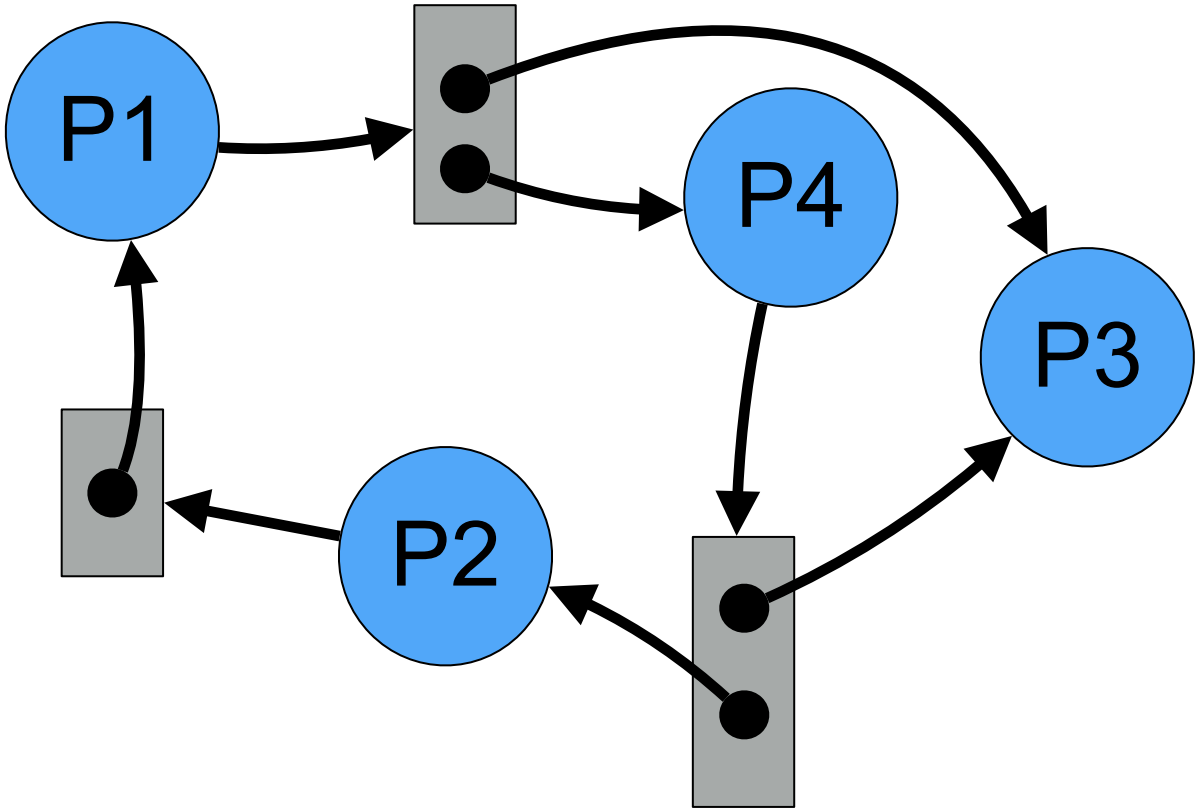
- Is there a deadlock?

(q5) Answer



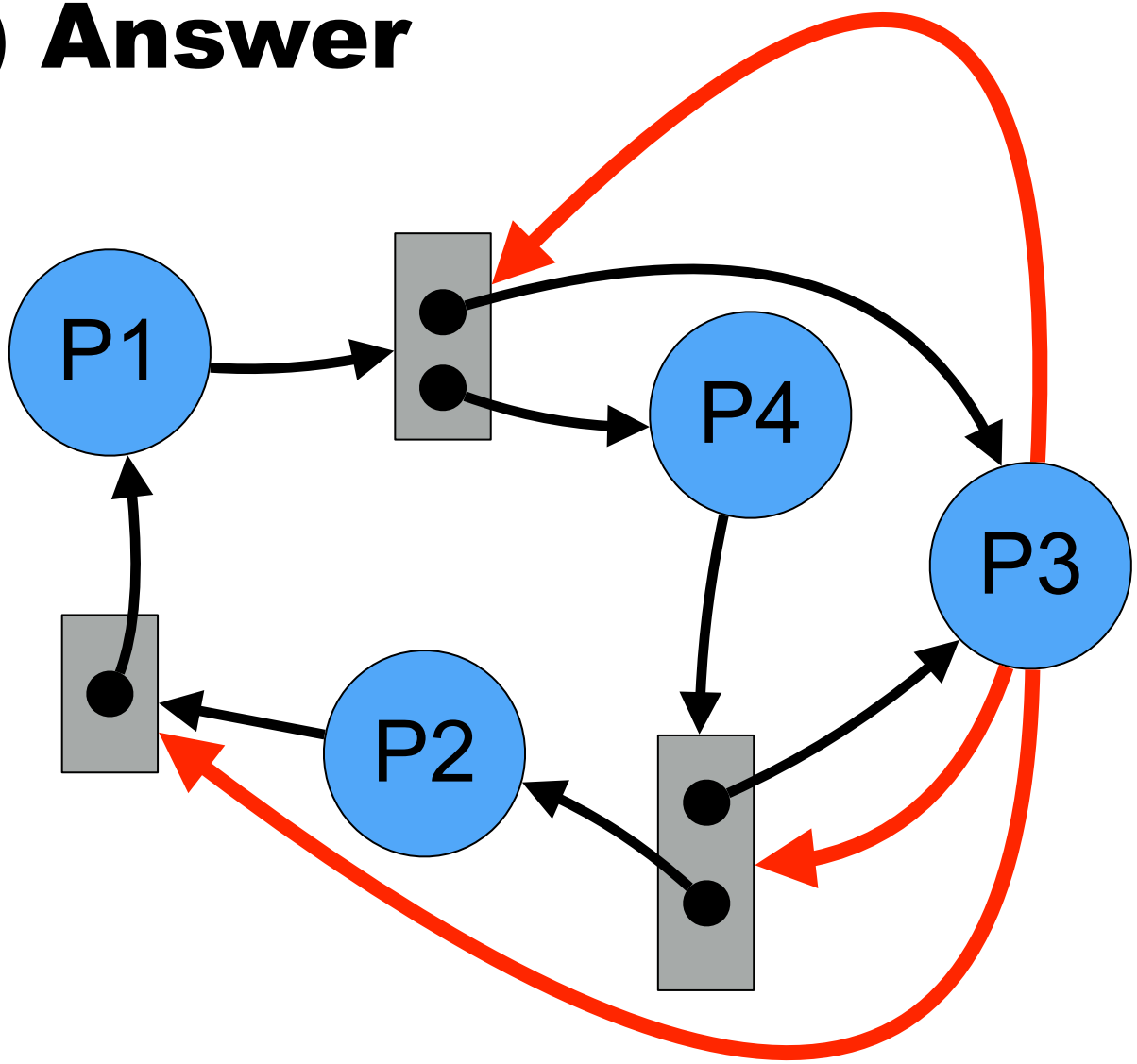
- Is there a deadlock? NO

(q6) Deadlock Graph



- Add an edge so that there IS a deadlock

(q6) Answer



- THREE options