TD Interblocages / problème des philosophes

Problème des philosophes

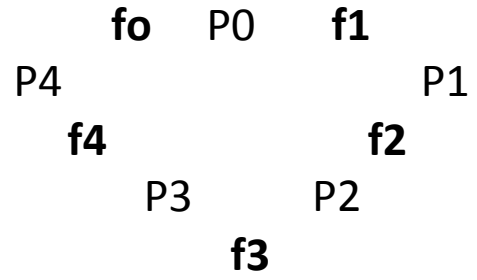


Tableau gardes/actions

Opération	Garde	Action
beginEat(int i)	f[i] && f[i+1%N]	f[i]=FALSE f[i+1%N]=FALSE
endEat(int i)		f[i]=TRUE f[i+1%N]=TRUE

Opération	Garde	Action
beginEat(int i)	!eat[i-1 %N] && ! eat[i+1%N]	eat[i]=TRUE
endEat(int i)		eat[i]=FALSE

Solution directe

```
Class Table {
    boolean f[];
    public Table(int N) {
       f = new boolean[N];
       for (int i = 0; i < N; i++)
          f[i] = TRUE;
    }

    public void synchronized beginEat(int i) {
          while !(f[i] && f[i+1%N])
              try{wait() } catch (InterruptedException e) {}
          f[i] = FALSE;
          f[i+1 %N] = FALSE;
    }

    public void synchronized endEat(int i) {
          f[i] = TRUE;
          f[i+1%N] = TRUE;
          notifyAll();
    } }
```

→Fonctionne

→Risque de famine

Solution sémaphores

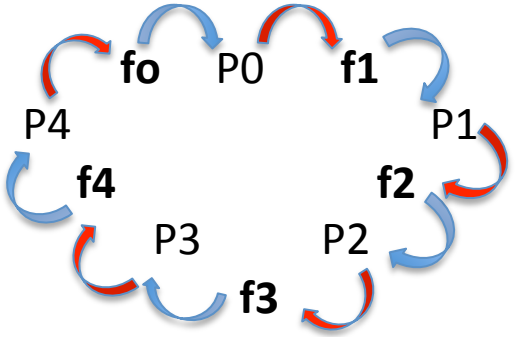
```
Class Table {
    Semaphore f[];

    public Table(int N) {
       f = new semaphore[N];
       for (int i = 0; i < N; i++)
          f[i] = new Semaphore[1);
    }

    public void beginEat(int i) {
         f[i].P();
         f[i+1%N].P() ;
    }

    public void endEat(int i) {
          f[i].V() ;
          f[i+1%N].V() ;
    } }
```

→Fonctionne

→Risque d'interblocage :

Philosophe 0 : f[0].P()
...
Philosophe4 : f[4].P()
Philosophe0 : f[1].P() bloqué
...
Philosophe4 : f[0].P() bloqué

Scenario d’interblocage



Solution sémaphores
Allocation globale

```
Class Table {
    Semaphore f[];
    Semaphore global;

    public Table(int N) {
       global = new Semaphore(1);
       f = new semaphore[N];
       for (int i = 0; i < N; i++)
         f[i] = new Semaphore[1);
    }

    public void beginEat(int i) {
        global.P();
        f[i].P();
        f[i+1%N].P() ;
       global.V();
    }

    public void endEat(int i) {
         f[i].V() ;
         f[i+1%N].V() ;
    } }
```

→Fonctionne

→Limite le parallélisme

- → P0 : f[0].P(), f[1]. P()
- → P1 : f[1].P() bloquant
- → P2: bloqué sur global
- → P3: bloqué sur global
- → P4: bloqué sur global

Limitation nb de philosophes

```
Class Table {
    Semaphore f[];
    Semaphore max;
    public Table(int N) {
       f = new semaphore[N];
       for (int i = 0; i < N; i++)
          f= new Semaphore[1];
       max = new Semaphore(N-1) ;
    }

    public void beginEat(int i) {
        max.P();
        f[i].P();
        f[i+1%N].P();
        max.V();
    }

    public void endEat(int i) {
         f[i].V()
         f[i+1%N].V()
    } }
```

→Fonctionne

→Pas de risque d'interblocage

→Limite un peu le parallélisme

Allocation ordonnée

```
Class Table {
    Semaphore f[];
    Semaphore max;
    public Table(int N) {
       f = new semaphore[N];
       for (int i = 0; i < N; i++)
          f= new Semaphore[1];
    }

    public void beginEat(int i) {
        if (i == N-1 ) {
          f[i+1%N].P();
          f[i].P() ;
        } else {
         f[i].P();
         f[i+1%N].P();
        }
    }

    public void endEat(int i) {
          f[i].V()
           f[i+1%N].V()
    } }
```

→Fonctionne

→Pas de risque d'interblocage