

MC504 - Sistemas Operacionais

Semáforos e os Filósofos Famintos

Islene Calciolari Garcia

Primeiro Semestre de 2017

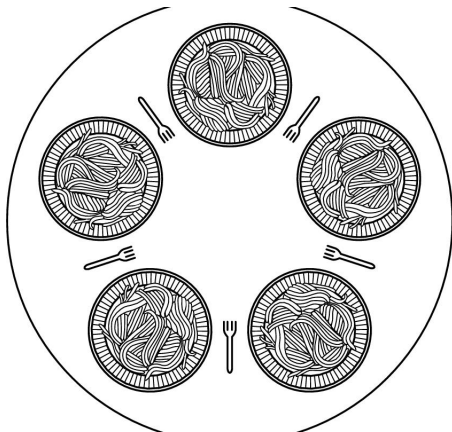
Sumário

Introdução

Semáforos

Controle no garfo

Controle no filósofo

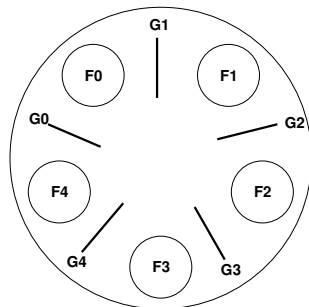


Tanenbaum: Figura 1.20

Boas soluções

- ▶ ausência de *deadlock*
- ▶ ausência de *starvation*
- ▶ alto grau de paralelismo

Representação da mesa



-	T	-	T	-	T	-	T	-	T	-
-	T	-	T	-	H	-	T	-	T	-
-	T	-	T	-	H	-	T	-	T	-
-	T	-	T	-	E	-	T	-	T	-

- ▶ Semáforos são *contadores especiais* para recursos compartilhados.
- ▶ Proposto por Dijkstra (1965)
- ▶ Operações básicas (atômicas):
 - ▶ decremento (down, wait ou P)
bloqueia se o contador for nulo
 - ▶ incremento (up, signal (post) ou V)
nunca bloqueia

Semáforos

Comportamento básico

- ▶ `sem_init(s, 5)`

- ▶ `wait(s)`

```
if (s == 0)
    bloqueia_processo();
else s--;
```

- ▶ `signal(s)`

```
if (s == 0 && existe_processo_bloqueado)
    acorda_processo();
else s++;
```

Implementação com Semáforos

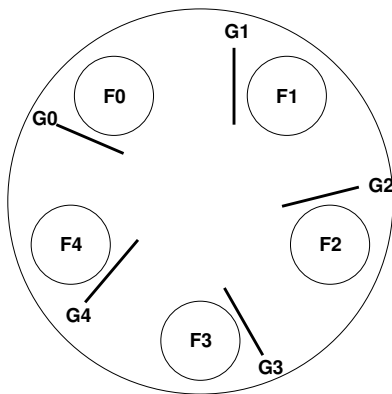
Um semáforo por garfo

- ▶ `sem_init(&garfo, 1)`
- ▶ `wait(&garfo)`
- ▶ `signal(&garfo)`

Filósofo i:

```
while (true)
    pensa();
    wait(garfo[i]);
    wait(garfo[(i+1) % N]);
    come();
    signal(garfo[i]);
    signal(garfo[(i+1) % N]);
```

Deadlock



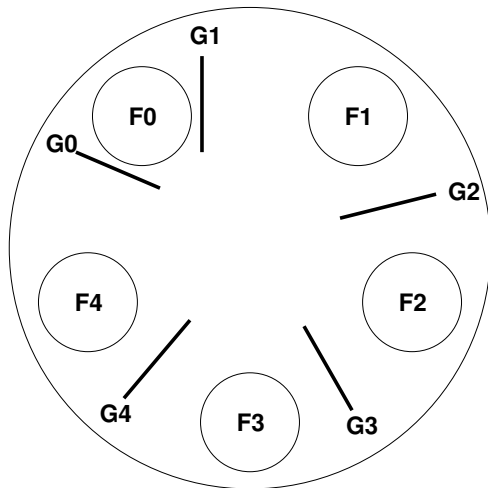
Veja códigos: `deadlock.c` e `deadlock-bug-exibicao.c`

```
semaforo lock = 1;
```

Filósofo i:

```
while (true)
    pensa();
    wait(lock);
    wait(garfo[i]);
    wait(garfo[(i+1) % N]);
    come();
    signal(garfo[(i+1) % N]);
    signal(garfo[i]);
    signal(lock);
```

Baixíssimo paralelismo



Veja código: `sem_central.c`

O que acontece se `lock == 2`?

```
semaforo lock = 2;
```

Filósofo i:

```
while (true)
    pensa();
    wait(lock);
    wait(garfo[i]);
    wait(garfo[(i+1) % N]);
    come();
    signal(garfo[(i+1) % N]);
    signal(garfo[i]);
    signal(lock);
```

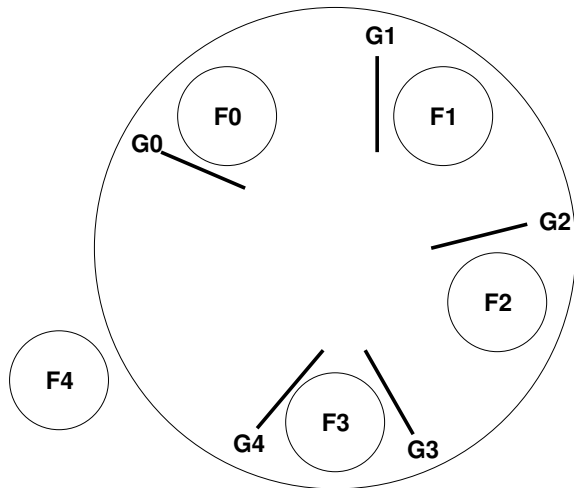
Menos lugares à mesa

```
semaforo lugar_mesa = N-1;
```

Filósofo i:

```
while (true)
    pensa();
    wait(lugar_mesa);
    wait(garfo[i]);
    wait(garfo[(i+1) % N]);
    come();
    signal(garfo[(i+1) % N]);
    signal(garfo[i]);
    signal(lugar_mesa);
```

Menos lugares à mesa

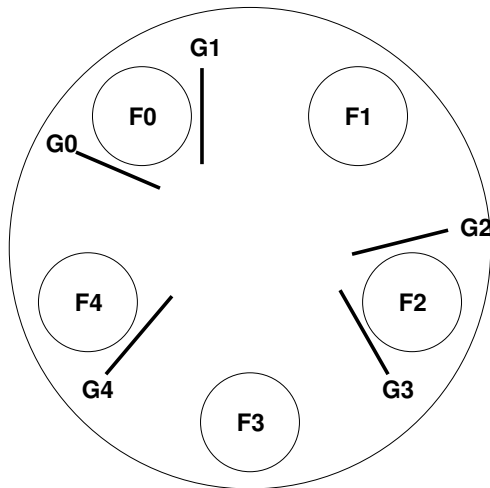


Veja código: `limite_lugares.c`

Solução assimétrica

```
while (true)
    pensa();
    if (i % 2 == 0)
        wait(garfo[i]);
        wait(garfo[(i+1) % N]);
    else
        wait(garfo[(i+1) % N]);
        wait(garfo[i]);
    come();
    signal(garfo[(i+1) % N]);
    signal(garfo[i]);
```

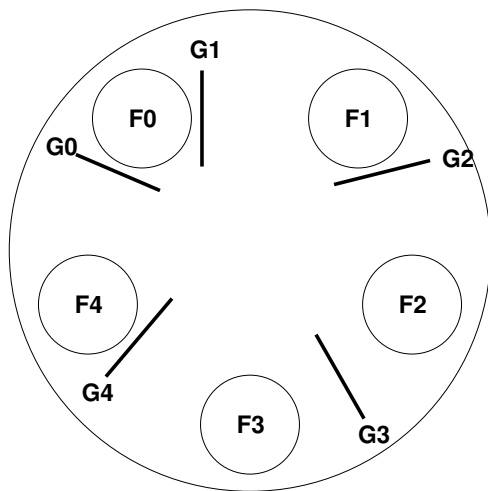

Solução assimétrica



Veja código: `assimetrica.c`

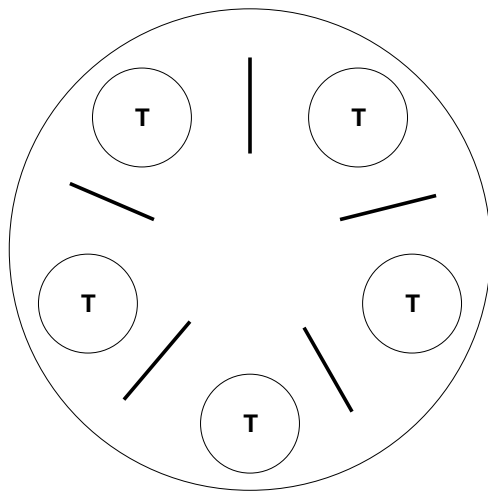
Solução assimétrica

Baixo paralelismo?!



Filósofos famintos

Um semáforo por filósofo



```
semaforo lock;  
semaforo filosofo[N] = {0, 0, 0, ..., 0}  
int estado[N] = {T, T, T, ...,T}
```

Filósofo i:

```
while (true)  
    pensa();  
    pega_garfos();  
    come();  
    solta_garfos();
```

Solução do livro Tanenbaum

Codificação simplificada

```
pega_garfos()
    wait(lock);
    if (estado[fil_esq(i)] != E && estado[fil_dir(i)] !=
        estado[i] = E;
        signal(lock);
    else
        estado[i] = H;
        signal(lock);
        wait(filosofo[i]);
```

Solução do livro Tanenbaum

Codificação simplificada

```
testa_garfos(int k)
    if (estado[k] == H && estado[fil_esq(k)] != E &&
        estado[fil_dir(k)] != E)
        estado[k] = E;
        signal(filosofo[k]);

solta_garfos()
    wait(lock);
    estado[i] = T;
    testa_garfos(fil_esq(i));
    testa_garfos(fil_dir(i));
    signal(lock);
```

Solução do livro Tanenbaum

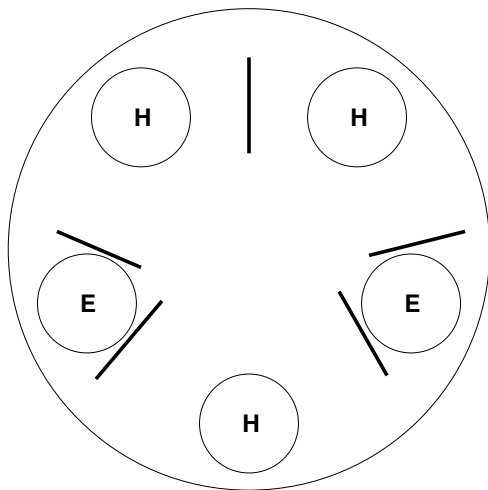
Codificação original

```
testa_garfos(int k)
    if (estado[k] == H && estado[fil_esq(k)] != E &&
        estado[fil_dir(k)] != E)
        estado[k] = E;
        signal(filosofo[k]);

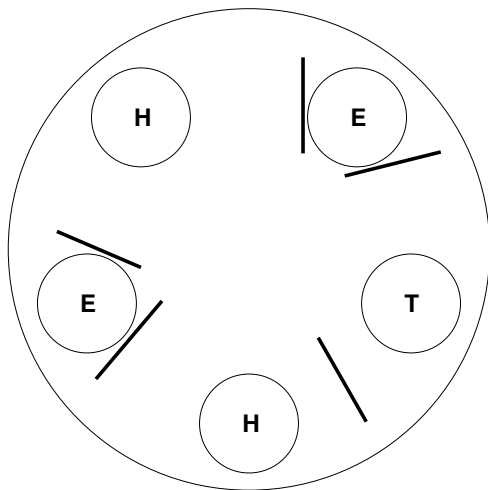
pega_garfos()                solta_garfos()
    wait(lock);                wait(lock);
    estado[i] = H;              estado[i] = T;
    testa_garfos(i);            testa_garfos(fil_esq(i));
    signal(lock);               testa_garfos(fil_dir(i));
    wait(filosofo[i]);          signal(lock);
```

Veja código: tanen-sem.c

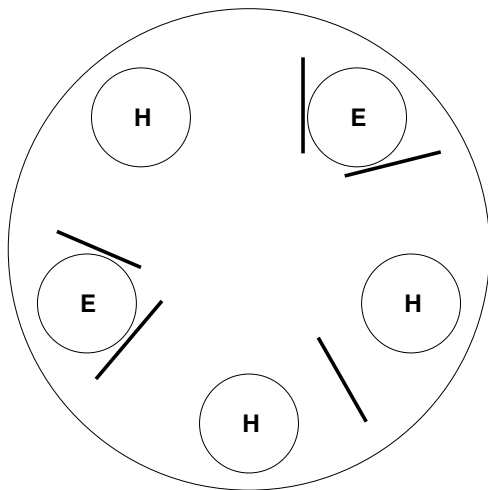
Alto paralelismo



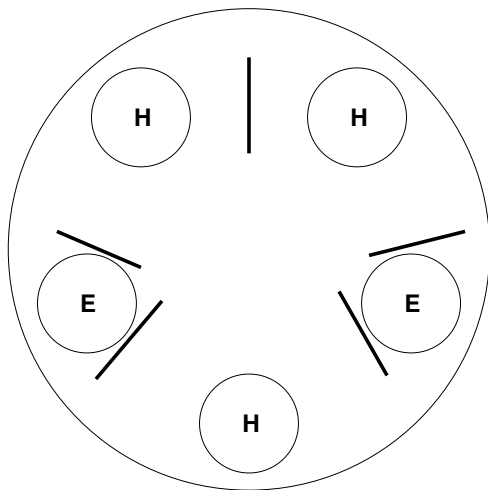
Alto paralelismo



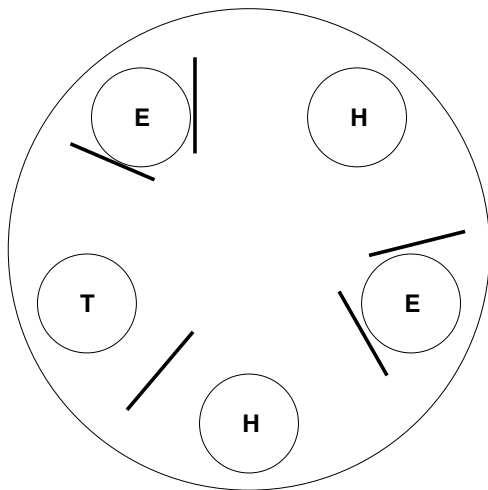
Alto paralelismo



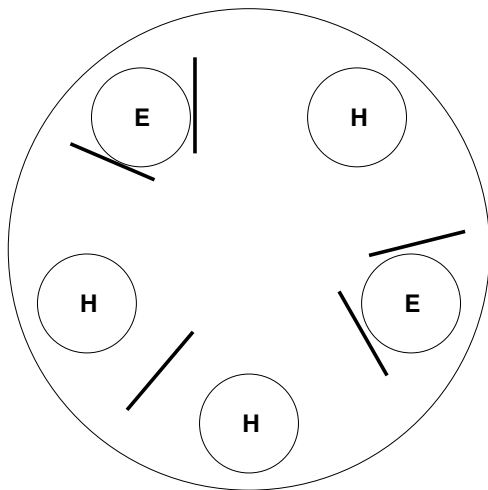
Alto paralelismo



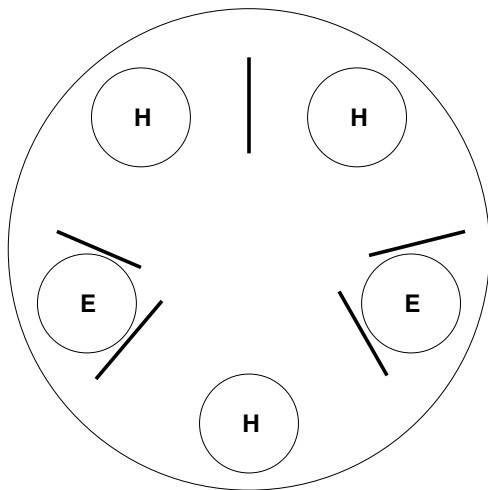
Alto paralelismo



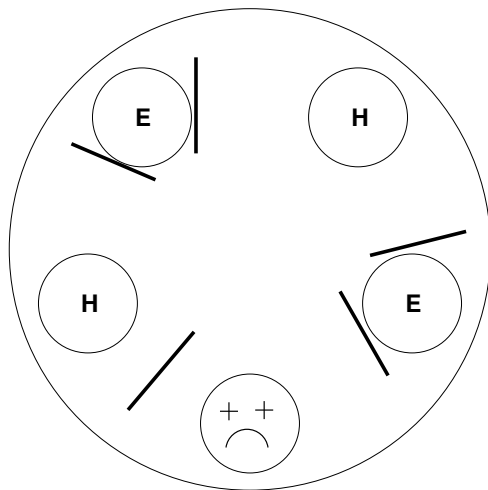
Alto paralelismo



Alto paralelismo



Starvation



Como matar os filósofos de fome?

- ▶ É preciso ajustar os tempos.
- ▶ Veja o código: `tanen-4-2.c` e `tanen-5-1.c`
- ▶ Como implementar `tanen-8-2.c`?
- ▶ É possível garantir alto paralelismo e ausência de starvation ao mesmo tempo?