Semáforos

Walter Fetter Lages

w.fetter@ieee.org

Universidade Federal do Rio Grande do Sul
Escola de Engenharia
Departamento de Engenharia Elétrica

ENG04008 Sistemas de Tempo Real

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Introdução

- Conceito introduzido por Dijkstra (1968)
- Simplificam os protocolos para sincronização
- Eliminam a necessidade de espera-ocupada
- Um semáforo é uma variável inteira, não negativa na qual se pode fazer apenas duas operações (além da inicialização):
  - `P(s)=wait(s)`
  - `V(s)=signal(s)`

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Operações nos Semáforos

- `P(s)=wait(s)`
  - Se `s` > 0 decrementa `s`, senão suspende o processo
- `V(s)=signal(s)`
  - Se existe algum processo suspenso em `s`, libera um deles, senão incrementa `s`
    - Não é especificado qual proceso é liberado
- As operações nos semáforos são atômicas
- Semáforos binários
- Semáforos contadores

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Exemplo de Sincronização

```
void p1(void)
{
     ...
    wait(s1);
     ...
}
```

```
void p2(void)
{
     ...
    signal(s1);
     ...
}
```

Exemplo de Exclusão Mútua

```
void p1(void)
{
    for(;;)
    {
        wait(s1);
     /*seção crítica*/
        signal(s1);
      ...
    }
    ...
}
```

```
void p2(void)
{
    for(;;)
    {
        wait(s1);
      /*seção crítica*/
        signal(s1);
      ...
    }
    ...
}
```

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Semáforos POSIX

- POSIX 1003.1b

```
#include <semaphore.h>
int sem_init(sem_t *sem,int pshared,
      unsigned int value)
int sem_wait(sem_t *sem)
int sem_trywait(sem_t *sem)
int sem_post(sem_t *sem)
int sem_getvalue(sem_t *sem,int *sval)
int sem_destroy(sem_t *sem)
```

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Semáforos System V

- Semáforos são um conjunto de valores de semáforos
- A criação é independente da inicialização
- Semáforos não são destruidos com os processos
  - Semáforos existem no escopo sistema e não no escopo dos processos que os utilizam
- Grande parte das operações realizadas por uma única função

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Criação

```
int semget(key_t key,int nsems,
           int semflg)
```

- Retorna o identificador dos semáforos associado a `key`
- `key` é uma chave única
  - `IPC_PRIVATE` garante a unicidade
- `nsems` é o número de semáforo no conjunto
- `semflg` *bitwise or* entre
  - `IPC_CREAT` cria um novo semáforo
  - modo de acesso: `rwxrwxrwx`
    - `SEM_R, SEM_R >> 3, SEM_R >> 6`
    - `SEM_W, SEM_W >> 3, SEM_W >> 6`

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Controle

```
int semctl(int semid,int semnun,int cm
        union semun arg)

union semun
{
        int val;
        struct semid_ds *buf;
        ushort *array;
};
```

- Executa as operações de controle nos semáforos

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Controle

- `IPC_STAT` copia informação de controle
- `IPC_SET` altera as permissões na estrutura de controle
- `IPC_RMID` remove o semáforo
- `GETVAL` retorna o valor de `semnum`
- `SETVAL` altera o valor de `semnum`
- `GETPID` retorna o `PID` da última operação em `semnum`

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Controle

- `GETNCNT` retorna o número de processos bloqueados esperando um incremento em `semnum`
- `GETZCNT` retorna o número de processos bloqueados esperando que o valor de `semnum` seja 0
- `GETALL` obtém os valores dos semáfors no conjunto
- `SETALL` altera os valors dos semáfors no conjunto

Operações

```
int semop(int semid,struct sembuf sops[],
          size_t nsops)

struct sembuf
{
          short sem_num;
          short sem_op;
          short sem_flg;
};
```

- Executa operações atômicas no conjunto de semáforos

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Operações

- `sem_flg` *bitwise or* entre `IPC_NOWAIT` e `SEM_UNDO`
- `sem_op`
  - > 0
    - Corresponde à operação `signal`
    - `sem_op` é somado ao valor do semáforo
    - Se `SEM_UNDO` for especificado, `sem_op` é subraído do valor de ajuste do semáforo para o processo
  - $= 0$ o processo é bloqueado até que
    - O valor do semáforo seja 0, ou
    - O semáforo seja removido do sistema, ou
    - O processo receba um sinal

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Operações

- `sem_op` < 0
  - Corresponde à operação `wait`
  - Se o valor do semáforo $\geq$ |`sem_op`|
    - O valor absoluto de `sem_op` é subtraído do valor do semáforo
    - Se `SEM_UNDO` for especificado, o |`sem_op`| é somado ao valor de ajuste do semáforo para o processo
  - Se o valor do semáforo é $<$ |`sem_op`|
    - O processo é bloqueado até que
      - O valor do semáforo $\geq$ |`sem_op`|, ou
      - O semáforo seja removido do sistema, ou
      - O processo receba um sinal

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Ajuste de semáforos

- Quando um processo termina as operações de `wait` em semáforos (alocações de recursos) realizadas por els não são desfeitas
- Quando as operações incluem a `flag` `SEM_UNDO`, o *kernel* rastreia as operações de `wait` realizadas e realiza os ajustes quando o processo termina.

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Problemas com Semáforos

- Primitivas de baixo nível
- Basta a omissão de uma ocorrência para que o programa não funcione corretamente
- Semáforos não implementam a exclusão mútua diretamente, apenas um mecanismo que permite a implementação de exclusão mútua
- Semáforos System V são desnecessariamente complicados
- Semáforos POSIX exigem memória compartilhada entre os processos