

Processos e Threads

Leitores e escritores

Leitores e escritores

```
semaforo sem_db = 1;
```

Leitor:

```
while(true)
    wait(sem_db);
    le_db();
    signal(sem_db);
```

Escritor:

```
while(true)
    wait(sem_db);
    escreve_db();
    signal(sem_db);
```

Leitores e escritores

- Problema: apenas um leitor pode fazer acesso ao banco de dados por vez
- Veja o código: `l-e-sem-concorrencia.c`
- Possível solução: permitir o acesso simultâneo a vários leitores

Vários leitores simultâneos

```
semaforo sem_db = 1, sem_nl = 1;  
int nl; /* Leitores ativos num dado instante */
```

Leitor:

```
while(true)  
    wait(sem_nl);  
    if (nl == 0) wait(sem_db);  
    nl++;  
    signal(sem_nl);  
    le_db();  
    wait(sem_nl);  
    if (nl == 0) signal(sem_db);  
    signal(sem_nl);
```

Vários leitores simultâneos

- Problema: os escritores podem morrer de fome
- Veja o código: `l-e-starvation.c`
- Como escrever este código usando locks e variáveis de condição?

Leitores simultâneos

Locks e variáveis de condição

Primeira tentativa

```
mutex_t lock_bd; /* Lock para o banco de dados */  
mutex_t lock_nl; /* Lock para o contador nl */  
int nl = 0;      /* Número de leitores ativos */
```

Leitor

```
mutex_lock(&lock_nl);  
if (nl == 0) mutex_lock(&lock_db);  
nl++;  
mutex_unlock(&lock_nl);  
le_db();  
mutex_lock(&lock_nl);  
nl--;  
if (nl == 0)  
    mutex_unlock(&lock_bd);  
mutex_unlock(&lock_nl);
```

Leitores simultâneos

- Problema: Uma thread leitora faz o lock e outra faz o unlock
- Veja o código: l-e-lock.c
- Tipos de lock:
 - FAST
 - RECURSIVE
 - ERROR CHECKING

Leitores simultâneos

Locks e variáveis de condição

Segunda tentativa

```
mutex_t lock; /* Espera pelo banco de dados */  
boolean bloq_leitura = false;
```

```
mutex_t lock_nl; /* Lock para o contador */  
int nl = 0;      /* Número de leitores ativos */
```

Leitor

```
mutex_lock(&lock_nl);  
if (nl == 0)  
    mutex_lock(&lock_bd);  
    bloq_leitura = true;  
    mutex_unlock(&lock_bd);  
nl++;  
mutex_unlock(&lock_nl);  
le_bd();  
/* ... */
```

Leitor

```
/* ...*/  
le_bd();  
mutex_lock(&lock_nl);  
nl--;  
if (nl == 0)  
    mutex_lock(&lock_bd);  
    bloq_leitura = false;  
    cond_signal(&cond_bd);  
    mutex_unlock(&lock_bd);  
mutex_unlock(&lock_nl);
```

Escritor

```
mutex_lock(&lock_bd);  
while (bloq_leitura)  
    cond_wait(&cond_bd, &lock_bd);  
escreve_bd();  
cond_signal(&cond_bd);  
mutex_unlock(&lock_bd);
```

Leitores simultâneos

Locks e variáveis de condição

Terceira abordagem

```
cond_t  cond_bd; /* Espera pelo banco de dados */  
  
mutex_t lock_cont; /* Lock para os contadores */  
int nl = 0;        /* Número de leitores ativos */  
int ne = 0;        /* Número de escritores ativos */
```

Veja o código: l-e-broadcast.c

Leitor

```
mutex_lock(&lock_cont);
while (ne > 0)
    cond_wait(&cond_bd, &lock_cont);
nl++;
mutex_unlock(&lock_cont);
le_bd();
mutex_lock(&lock_cont);
nl--;
if (nl == 0)
    cond_signal(&cond_bd);
mutex_unlock(&lock_cont);
```

Escritor

```
mutex_lock(&lock_cont);  
while (nl > 0 || ne > 0)  
    cond_wait(&cond_bd, &lock_cont);  
ne++;  
mutex_unlock(&lock_cont);  
escreve_db();  
mutex_lock(&lock_cont);  
ne--;  
cond_broadcast(&cond_bd);  
mutex_unlock(&lock_cont);
```

Leitores e escritores

Prioridade para os escritores

```
int nl = 0; /* Número de leitores */
int ne = 0; /* Número de escritores */
int nw = 0; /* Número de escritores esperando */
mutex_t lock_cont;
cond_t cond_esc, cond_leit;
```

Veja o código: l-e-broadcast2.c

Leitor

```
mutex_lock(&lock_cont);
while (ne > 0 || nw > 0)
    cond_wait(&cond_leit, &lock_cont);
nl++;
mutex_unlock(&lock_cont);
/* Leitura */
mutex_lock(&lock_cont);
nl--;
if (nl == 0 && nw > 0)
    cond_signal(&cond_esc);
mutex_unlock(&lock_cont);
```

Escritor

```
mutex_lock(&lock_cont);
nw++;
while (nl > 0 || ne > 0)
    cond_wait(&cond_esc, &lock_cont);
nw--; ne++;
mutex_unlock(&lock_cont);
/* Escrita */
mutex_lock(&lock_cont);
ne--;
if (nw > 0)
    cond_signal(&cond_esc);
else
    cond_broadcast(&cond_leit);
mutex_unlock(&lock_cont);
```

Leitores e escritores

Qual seria um bom compromisso?

- Ausência de starvation
- Leitores simultâneos