

Coordenação por espera ocupada

Este projeto visa expor os problemas advindos do acesso concorrente de tarefas a estruturas de dados compartilhadas e explorar algumas das soluções possíveis para a coordenação dos acessos, usando mecanismos de espera ocupada.

Problema

Você deve criar um programa com duas threads POSIX, que manipulam uma fila de inteiros de forma concorrente. Cada thread retira o primeiro elemento da fila e coloca um novo elemento no fim da fila:

```
while (1)
{
    velho = retira_primeiro_elemento_da_fila()
    novo = random() % 100
    poe_elemento_no_fim_da_fila (novo)
    imprime operação efetuada e estado da fila
}
```

Como fila, deve ser usada a implementação de fila circular construída anteriormente.

Observações

A saída do programa deve seguir o formato abaixo (obviamente com valores aleatórios):

```
thread 0: tira 34,   põe 81,   fila: 47  2 19 66 32 60  9 11 38 81
thread 1: tira 47,   põe 55,   fila:  2 19 66 32 60  9 11 38 81 55
thread 0: tira  2,   põe 31,   fila: 19 66 32 60  9 11 38 81 55 31
thread 1: tira 19,   põe 17,   fila: 66 32 60  9 11 38 81 55 31 17
...
```

A fila tem capacidade para 10 inteiros, está inicialmente cheia (valores aleatórios) e tem comportamento FIFO.

Observe que a ordem de ativação das threads depende do mecanismo de sincronização utilizado. Por exemplo, caso não seja usado nenhum algoritmo, a ordem é qualquer; caso seja usado o algoritmo de alternância escrita, a ordem deve ser "t0 t1 t0 t1 ...".

Roteiro

1. Implemente o programa sem nenhum mecanismo de sincronização e observe se a fila se comporta corretamente.
2. Implemente o algoritmo de coordenação por alternância estrita (uma thread de cada vez) e observe o comportamento da fila.
3. Implemente o algoritmo de coordenação de Peterson¹ e observe o comportamento da fila.
4. Proponha e implemente uma forma de medir o número de inserções na fila por segundo.
5. Meça o desempenho das três implementações (sem sincronização, com alternância e com o algoritmo de Peterson) e justifique os resultados observados.

A entregar

1. As implementações solicitadas
2. Um relatório sucinto (e no [formato correto](#)) descrevendo os resultados observados nas experiências.

¹⁾

O algoritmo de Peterson pode ter problemas de funcionamento em máquinas *dual-core* ou multi-processadas; consulte o livro de Sistemas Operacionais para maiores informações.

From:

<https://wiki.inf.ufpr.br/maziero/> - **Prof. Carlos Maziero**

Permanent link:

https://wiki.inf.ufpr.br/maziero/doku.php?id=so:espera_ocupada

Last update: **2011/09/26 22:32**

