MAC 438 - Programação Concorrente - Primeiro Semestre de 2001

Primeiro Exercício-Programa

Data de Entrega: 16 de abril de 2001

Este exercício deve ser desenvolvido em equipes de duas pessoas, a fim de suscitar discussão. Ele é pequeno, mas envolve detalhes que devem ser conhecidos por todos vocês. Não deixe seu colega de equipe fazer tudo sozinho!

Dúvidas sobre o enunciado devem ser enviadas para `reverbel-mac438@ime.usp.br`.

1 O problema

Vocês devem implementar uma solução para um problema clássico de comunicação entre processos: o *Problema do Produtor-Consumidor* (também conhecido como o problema do *bounded buffer*). Trataremos aqui da versão mais simples desse problema, em que apenas dois processos compartilham um área de memória (buffer) comum, de tamanho fixo. Essa área de memória é empregada para se implementar uma fila com capacidade limitada (buffer circular).

Um dos dois processos, o produtor, coloca dados no fim da fila, enquanto que o outro — o consumidor – retira dados do início da fila. Não é possível tecer hipóteses sobre a velocidade do trabalho desses processos, ou seja, é possível que o produtor tanto demore eternidades para produzir novos elementos quanto os produza instantaneamente. Idem para o consumidor, que pode passar períodos arbitrários entre retiradas de elementos da fila.

Problemas aparecem quando o produtor deseja adicionar um elemento à fila, e esta já se encontra lotada. Nesta situação desejamos que o produtor fique algum tempo sem processar, esperando que apareça algum lugar livre no buffer circular. Similarmente, se o consumidor quer retirar um item da fila e a encontra vazia, ele deve aguardar até que o produtor enfileire algum elemento.

O pseudo-código a seguir mostra o comportamento dos dois processos.

```
#define TAM_BUFFER 100

/*---- memória compartilhada entre os processos produtor e consumidor -------*/
int n = 0;              /* comprimento atual da fila */
...                     /* outras variáveis da implementação da fila */

/*------------------- processo produtor -----------------------------------*/
int item;

for ( ; ; ) {
    produz_item(&item);
    if (n == TAM_BUFFER) {
        /* buffer lotado */
        espera_sinal_de_nao_lotado();
    }
    enfileira_item(item);
    /* atualiza variavel compartilhada */
    n++;
    if (n == 1) {
        /* estava vazio o buffer ... */
        manda_sinal_de_nao_vazio();
    }
}
```

```
/*------------------- processo consumidor ----------------------------------*/
int item;

for ( ; ; ) {
    if (n == 0) {
        /* buffer vazio */
        espera_sinal_de_nao_vazio();
    }
    desenfileira_item(&item);
    n--;
    if (n == TAM_BUFFER - 1) {
        /* estava lotado até agora */
        manda_sinal_de_nao_lotado();
    }
    consome_item(item);
}
```

2 Seu arsenal

Você deve fazer este EP em C.

A sua implementação deve envolver dois processos, sem relação parent/child entre eles. A comunicação entre processos para indicar mudanças (de fila cheio para não-cheio, de vazio para não vazio) deve ser feita através de sinais Unix. O material sobre sinais está disponível na pasta 48 do xerox do CAMAT. A área de memória compartilhada deve ser implementada através das primitivas de *shared memory* disponíveis no Unix. O material sobre shared memory também está no xerox do CAMAT, e na página da disciplina há um exemplo de uso (dois processos que compartilham um buffer — um escreve e outro lê).

Certos trechos do programa que acessarem dados compartilhados são regiões críticas, que precisam ser executadas com garantia de exclusão mútua. Identifique esses trechos (atenção para os acessos ao contador do número de elementos no buffer circular) e proteja-os usando um dos protocolos de entrada e saída de região crítica estudados em aula. Documente sua escolha.

Além de entregar os seus programas muito bem comentados, você deve entregar algumas estimativas sobre a disputa pela região crítica (por exemplo, os tempos médio, máximo e mínimo que o processo leva para obter o acesso).

Entregue também uma pequena explicação sobre que complicações apareceriam se tentássemos modificar sua implementação para resolver o problema mais geral, com M produtores e N consumidores concorrentes.

3 Sobre a entrega

Você deverá entregar tres coisas:

- um arquivo tar.gz contendo sua solução (arquivos-fonte, makefile, README, ...);
- uma listagem impressa dos seus arquivos-fonte;
- um relatório impresso sobre sua solução, com respostas para as questões acima.

Por favor, entregue o relatório e a listagem **na secretaria do MAC**, em um saco plástico devidamente fechado. Em breve serão divulgadas instruções para entrega do arquivo tar.gz (ainda não sei se poderemos usar um novo sistema de submissão eletrônica de trabalhos ou se o arquivo será entregue em disquete mesmo).

EPs atrasados não serão aceitos!

Bom trabalho!