MAC0438 – Programação concorrente – 1s2013
EP3
Datas de Entrega: 03/06/2013 e 17/06/2013

Prof. Daniel Macêdo Batista

1 Problema

Neste EP você deverá resolver o problema das abelhas e dos ursos.

Existem $N > 1$ threads abelha bondosas que alimentam $B > 1$ threads urso presas em gaiolas. As abelhas alimentam os ursos com mel que elas coletam. A vida dos ursos presos resume-se a comer e dormir. A vida das abelhas é mais agitada. Elas coletam mel e carregam para um pote que tem capacidade de armazenar até $H > 100$ porções de mel. Cada abelha só consegue carregar e armazenar 1 porção por vez. O tempo que uma abelha leva para depositar a porção de mel no pote é $t \in N^*$. A cada instante de tempo é permitido que até 100 abelhas estejam enchendo o pote simultaneamente. Quando o pote está cheio, a última abelha que depositou mel no pote acorda um urso para que ele possa comer. As abelhas ficam paradas enquanto o urso come o mel do pote. O urso gasta o tempo $T \in N^*$ para esvaziar o pote. Quando o urso esvazia o pote ele ruge para as abelhas avisando que o mel terminou, volta a dormir e as abelhas voltam à tarefa de encher o pote.

Resolva esse problema utilizando monitor. Além de seguir a descrição acima, a sua implementação deve garantir a justiça no acesso que os ursos terão ao pote (as abelhas não podem o tempo todo acordar sempre o mesmo urso). Note que sua solução não deve inserir restrições a mais no problema. Por exemplo, uma solução que só dê acesso sequencial das abelhas ao pote, quando na verdade mais 99 poderiam estar enchendo o pote em paralelo, estará errada e receberá nota final ZERO.

2 Requisitos

2.1 Monitor

Neste EP você deverá manipular monitores utilizando apenas as funções vistas em sala de aula com exatamente os mesmos nomes abaixo:

`empty(cv), wait(cv), wait(cv,rank), signal(cv), signal_all(cv), minrank(cv)`

Note que seu EP não precisa utilizar todas as funções (obviamente as funções `signal` e `wait` são obrigatórias). EPs que não implementem pelo menos as funções `signal` e `wait` receberão nota ZERO. Você deve indicar no LEIAME do seu EP onde estão as implementações de todas as funções que você implementou para manipular monitores.

2.2 Linguagem

Seu programa pode ser escrito em qualquer linguagem de programação desde que o compilador seja gratuito e desde que exista compilador para GNU/Linux, pois o EP será corrigido neste SO.

Algumas linguagens já possuem algumas estruturas voltadas para a implementação de programas com monitores. Leia a página da wikipedia a respeito disso: `https://en.wikipedia.org/wiki/Monitor_%28synchronization%29#History` . Caso você utilize alguma linguagem que já venha com as funções exigidas na subseção anterior implementadas, você pode utilizá-las, mas note que os nomes exigidos para as funções **devem** ser mantidos. Por exemplo, se em uma dada linguagem a função para acordar um processo é `SignalZ` você precisa criar um *alias*, uma outra função, etc... chamada `signal` que por sua vez chamará a `SignalZ`.

2.3 Entrada e Saída

O seu EP deverá receber na linha de comando os seguintes parâmetros nesta ordem: N, B, H, t e T. Todos são valores inteiros respeitando os limites apresentados na descrição do problema.

O seu EP deverá imprimir na saída padrão as seguintes informações quando eventos especiais acontecerem:

- Estado das abelhas: enchendo o pote, parada porque o urso está comendo mel, voando porque não tem espaço para encher o pote;
- Quantidade de vezes que cada abelha acordou um urso;
- Quantidade de vezes que cada urso foi acordado;

Os eventos especiais são os seguintes:

- Pote cheio;
- Pote na metade quando o urso está comendo;
- Pote vazio;
- Pote na metade quando as abelhas estão enchendo;

3 Entregas

Até o dia 3/6 você deverá enviar os algoritmos de todos os monitores que você vai utilizar no seu programa. Escreva os algoritmos em um arquivo `.pdf` e juntamente com os comentários que forem necessários para o entendimento do algoritmo. Essa primeira entrega vale 3,0.

Até o dia 17/06 você deverá enviar um arquivo .tar.gz contendo os seguintes itens:

- fonte(s), Makefile (ou similar), arquivo LEIAME e um relatório.

O desempacotamento do arquivo .tar.gz **deve** produzir um diretório contendo os itens. O nome do diretório **deve** ser ep3-membros_da_equipe. Por exemplo: ep3-joao-maria.

O relatório deve relatar a execução do seu programa para diversas combinações dos parâmetros de entrada. Para **cada** combinação de parâmetros você **deve** apresentar **2** gráficos. O eixo x desses gráficos

será o tempo de relógio real. O eixo y de um gráfico vai ser a média do número de vezes que cada urso comeu e o eixo y de outro gráfico vai ser a média do número de vezes que cada abelha acordou um urso. É de se esperar que ambos os gráficos comecem com valores pequenos e que esses valores vão convergindo com o passar do tempo. A velocidade de convergência vai depender dos parâmetros.

Deixe bem claro no relatório quanto tempo durou as execuções. Para esse relatório ser coerente, a duração, que vai ser o maior valor de x dos gráficos, de cada execução deve ser igual. Também deixe claro quais foram as combinações de valores utilizados para gerar os gráficos. Não há um limite para a quantidade de gráficos.

A segunda entrega vale 7,0.

O EP pode ser feito individualmente ou em dupla. Todas as duas entregas devem ser feitas pelo paca.