MAC0438 – Programação concorrente – 1s2012
EP1
Data de Entrega: 09/04/2012

Prof. Daniel Macêdo Batista

1 Problema

O Tour de France é uma das principais competições de ciclismo do mundo. Nela diversos ciclistas competem em busca da conquista de diversos prêmios. Três dos principais prêmios são:

- Camiseta amarela: atribuída ao ciclista com o menor tempo individual;
- Camiseta verde: atribuída ao ciclista com a melhor pontuação em etapas planas;
- Camiseta branca com bolas vermelhas: atribuída ao ciclista com a melhor pontuação em etapas de montanha.

Ao término de cada etapa, o tempo de cada ciclista vai sendo somado e o ciclista com o menor tempo no final do Tour é o vencedor da camiseta amarela.

A camiseta verde e a camiseta branca com bolas vermelhas são definidas a partir da ordem de chegada em diversos checkpoints dentro de cada etapa. Os checkpoints que definem a pontuação para a camiseta verde ficam localizados em trechos planos e os checkpoints que definem a pontuação para a camiseta branca com bolas vermelhas ficam localizados nos topos de montanhas.

A sua tarefa neste EP é simular uma etapa do Tour de France, fornecendo ao término da simulação as três listas que definem a classificação para as camisetas amarela, verde e branca com bolas vermelhas. A simulação deve considerar que m ciclistas estão competindo e que em qualquer momento na estrada apenas n ciclistas podem estar lado a lado, já que o tamanho da estrada não é infinito. Dessa forma, se n ciclistas conseguirem pedalar na mesma velocidade alguns serão obrigados a ficar atrás para que não ocorram acidentes. Note que ultrapassagens são permitidas desde que todos os ciclistas envolvidos na ultrapassagem consigam ficar lado a lado na estrada (o valor de n nunca será menor que 2).

2 Requisitos

2.1 Linguagem

O simulador deve ser escrito em C e toda a gerência de threads deve ser feita utilizando POSIX threads (pthreads). Programas escritos em outra linguagem ou utilizando alguma biblioteca extra para gerenciar as threads terão nota zero.

Informações sobre como programar utilizando pthreads podem ser encontradas na seção 4.6 do livro do Andrews (basta ler até a subseção 4.6.1 inteira), na página da wikipedia em `http://en.`

wikipedia.org/wiki/POSIX_Threads ou no tutorial da IBM disponível em http://www.ibm.com/developerworks/library/l-posix1/index.html.

2.2 Detalhes do código

Seu simulador deve criar m threads "ciclista". Seu código terá duas opções de execução. Na primeira opção todos os ciclistas conseguem pedalar na mesma velocidade (50Km/h) durante toda a etapa do Tour, independente da quantidade de trechos planos e de montanha. Na segunda opção cada ciclista deve ter três velocidades definidas de forma aleatória. Uma para trechos planos, uma para trechos de subida e outra para trechos de descida. Em ambos os casos, a velocidade deve ser sorteada entre 20 e 80Km/h. Você pode sortear os valores seguindo a distribuição de probabilidade de sua preferência, ou ainda pode definir classes de ciclistas (melhores em trechos planos, melhores em subidas, etc...).

Seu código deve possuir um vetor compartilhado "estrada" que tem uma quantidade de posições igual à distância d da etapa em quilômetros. Em um dado instante de tempo, a posição i de estrada deve ter armazenado os identificadores de todos os ciclistas que estão dentro do intervalo $[i, i + 1)$Km. Cada thread ciclista tem a obrigação de escrever seu identificador na posição correta do vetor estrada a cada momento em que ele entra em um novo quilômetro, e de remover seu identificador da posição referente ao quilômetro que ele acabou de sair.

A saída do seu programa deve ser um relatório informando de minuto em minuto os nomes dos ciclistas que estão em cada um dos quilômetros da etapa e, a cada checkpoint, deve ser informado quem foram os três primeiros colocados. Ao término da corrida (depois que todos os ciclistas chegarem), as três listas devem ser informadas com os nomes de todos os ciclistas, não apenas dos três primeiros. Em caso de empate, em cada checkpoint e na colocação final, você tem liberdade para definir o mecanismo que defina quem fica na frente de quem (isso deve ser feito para todas as colocações. Não só para a primeira). Antes da simulação começar também deve ser apresentado um relatório com as características de cada ciclista, ou seja, suas três velocidades.

Não há um formato padrão para a saída do seu programa. Basta que ela informe tudo que foi solicitado no parágrafo anterior.

Com relação à entrada, seu simulador deve receber apenas um argumento que é o nome do arquivo com as informações sobre a etapa a ser simulada. O formato do arquivo segue abaixo:

```
m
n
U|A
d
P|S|D
k
P|S|D
k
P|S|D
k
...
```

As letras minúsculas representam valores inteiros:

- m: a quantidade de ciclistas
- n: a largura da pista em número de ciclistas

- d: a distância da etapa em quilômetros
- k: a distância de cada trecho da etapa (o tipo de etapa é definido na linha imediatamente acima)

As letras maiúsculas de fato aparecerão como letras no arquivo:

- U: a velocidade de todo o ciclista é uniforme (50Km/h)
- A: as três velocidades de cada ciclista devem ser definidas de forma aleatória dentro do intervalo [20,80]Km/h
- P: a quilometragem da linha imediatamente abaixo é referente a um trecho plano
- S: a quilometragem da linha imediatamente abaixo é referente a um trecho de subida
- D: a quilometragem da linha imediatamente abaixo é referente a um trecho de descida

Segue abaixo um exemplo de arquivo definindo 50 ciclistas, uma pista com largura que cabe até 10 ciclistas, uma distância de 200Km e quatro trechos distintos: trecho plano de 50Km, subida de 10Km, descida de 20Km e um outro trecho plano de 120Km. As velocidades dos ciclistas nesse exemplo devem ser definidas de forma aleatória.

```
50
10
A
200
P
50
S
10
D
20
P
120
```

Para todas as entradas considere que os checkpoints para a camiseta verde sempre são localizados no meio dos trechos planos. Conforme explicado anteriormente, os checkpoints para a camiseta branca com bolas vermelhas sempre estão localizados nos topos das montanhas. As pontuações para ambas as camisas seguem a lista abaixo:

- Primeiro colocado: 45
- Segundo colocado: 35
- Terceiro colocado: 25
- Quarto colocado: 15
- Quinto colocado: 10
- Sexto colocado: 5

- A partir do sétimo colocado: 0

Não há necessidade de validar a entrada.

Lembre que seu programa é um simulador. Ou seja, a simulação não precisa levar as 4 horas que uma etapa do Tour de France leva. Basta que os cálculos referentes aos tempos estejam de acordo com a velocidade definida para cada ciclista.

3 Relatório

Você deve entregar um relatório apresentando uma saída para a execução do código com cada um dos seguintes arquivos de entrada:

`arquivo1.txt`:

```
5
2
A
200
P
50
S
10
D
10
P
50
S
10
D
10
P
60
```

`arquivo2.txt`:

```
25
5
A
182
P
10
S
10
D
10
P
20
```

```
S
6
D
10
P
20
S
10
D
10
P
30
S
10
D
10
P
26
```

Apesar do relatório apresentar os resultados para os dois arquivos acima, não significa que o EP será avaliado apenas com essas duas entradas.

O relatório também deve explicar como foram implementadas as questões que ficaram em aberto: atribuição das velocidades quando a simulação for com A ao invés de U e definição do desempate.

4 Sobre a entrega

Você deve entregar um arquivo .tar.gz contendo os seguintes itens:

- fonte, Makefile (ou similar), arquivo LEIAME;
- relatório em .pdf.

O desempacotamento do arquivo .tar.gz deve produzir um diretório contendo os itens. O nome do diretório deve ser ep1-membros_da_equipe. Por exemplo: ep1-joao-maria.

A entrega do .tar.gz deve ser feita através do PACA.

O EP pode ser feito individualmente ou em dupla.