EP2

Renê Cardozo - 9797315
Verônica Stocco - 6828626
rene.cardozo@usp.br
veronica.stocco@usp.br

Instituto de Matemática e Estatística
Universidade de São Paulo

Corrida por eliminação

O simulador é composto por dois módulos: aleatorio e pista.

- **aleatorio** contém as funções responsáveis por determinar a velocidade e a probabilidade de quebra dos ciclistas.
- **pista** contém o simulador em si, e será discutido em detalhe nos slides a seguir.

Barreiras, turnos e pistas

O simulador utiliza duas barreiras para gerenciar os turnos da simulação. Desta forma, enquanto as n threads ciclistas restantes aguardam em uma barreira, a barreira referente ao turno seguinte é criada. A operaçao das threads em **void ciclista()** é feita considerando esses dois possíveis turnos (par e ímpar).
Também foi utilizada uma barreira adicional para definir a largada das threads.
A pista é composta por 10 linhas (é possivel editar esse valor em pista.h). Cada linha tem **d** metros, e um **mutex_pos** referente a cada metro / faixa da linha. No início da simulação, os ciclistas são distribuídos nas linhas 0, 2, 4, 6 , 8.

Ultrapassagem

A cada rodada, a posição atual do ciclista é atualizada com **atualiza_posicao**. Para evitar conflitos, são usados 4 mutex, referentes à:

- linha na qual o ciclista se encontra no momento;
- linha para a qual o ciclista pode se mudar caso ocorra uma ultrapassagem;
- próxima faixa (posição) que o ciclista pode ocupar na linha atual;
- próxima faixa (posição) que o ciclista pode ocupar caso ocorra uma ultrapassagem;

Eliminação de ciclistas

A eliminação de ciclistas se dá de duas formas:

- **Por quebra:** a chance de quebra é definida aleatóriamente, conforme as especificações do enunciado.
- **Por ser o último:** o último ciclista a completar uma volta de número par recebe uma flag em seu ID no vetor **elimina_id**. Essa marcação occore no main. A eliminação é executada pela thread referente a esse ID, quando ela for executada.

Quando um ciclista é eliminado, trancam-se os mutex ferentes à linha e faixa que ele ocupava para marcar essa posição como estando livre. Sua colocação é adicionada ao ranking, e outro mutex é utilizado para garantir que apenas uma thread altere o ranking de cada vez.

Ranqueamento

Optou-se por implementar 2 rankings distintos.

- **Ranking a cada rodada:** registra, no ID de cada ciclista, a sua colocação atual. Paralelo a esse ranking, um vetor de ints atômicos $\mathbf{pos}_v oltautilizadoparaarmazenaraposiodoltimocorredordecadavolta$.
- **Ranking final:** conforme cada ciclista é eliminado, armazena-se sua colocação final na corrida, assim como tempo total, última volta e se sua bicicleta quebrou ou não.

As atualizações nos valores dos rankings sempre são feitas utilizando mutexes referentes a eles.

Testes

Máquinas Utilizadas

Gráficos de tempo de execução

Figure: M1, 5 threads

Figure: M1, 50 threads

Figure: M1, 500 threads

Figure: M2, 5 threads

Figure: M2, 50 threads

Figure: M2, 500 threads

Gráficos de impacto no uso de memória