PROGRAMAÇÃO CONCORRENTE NO CÁLCULO DE INCERTEZAS EM MÉTODOS ESTOCÁSTICOS

Douglas de Morais Lima

RELATÓRIO PARCIAL
Programação Concorrente (ICP-361) — 2022/2

1. Descrição do Problema

O projeto dessa solução concorrente tem como objetivo melhor aproveitar o multi-processamento de informações no cálculo do valor esperado de métodos estatísticos reduzindo o tempo de processamento dos dados e obtendo assim um grau de incerteza mais baixo em um tempo menor garantindo a confiabilidade dos dados.

Na teoria da probabilidade temos o que é chamado de processo ou método estocástico, o processo estocástico observa como que uma variável se comporta durante o tempo, de uma maneira onde pelo menos parte é considerada randômica, dado que os eventos probabilísticos são estudos que podem ser quantificados matematicamente.

Seguindo uma lógica tradicional para a solução sequencial desse problema criaríamos um algoritmo que reproduza o comportamento dessa variável aleatória que estamos estudando, após a execução desse algoritmo N vezes, a variável N será a representação do nosso espaço amostral, logo, o valor esperado desse evento será representado pela média dos valores obtidos neste espaço amostral, ou seja a média do obtido pelo algoritmo em N simulações.

Para validar o valor esperado desse algoritmo, incluímos um certo grau de incerteza, isso significa que, observamos o valor obtido nessas N simulações, duas vezes, se a incerteza desejada não for encontrada, repetiremos essas 2 baterias de simulações novamente aumentando gradualmente o espaço amostral observado todas vez que a incerteza não for encontrada.

Para testar a nossa solução usaremos um problema proposto no curso de estatística e probabilidade temos a seguinte situação.

Considere que está inscrito em um círculo um dodecágono regular cujos vértices estão numerados de 0 até 11. Suponha que uma partícula move-se ao longo dos vértices do polígono, e que a cada segundo ela dá um passo no sentido horário ou anti-horário, com igual probabilidade.

O problema propõe a observação da variável em 5000 simulações, logo pela lógica da aplicação executaremos inicialmente 5000 simulações duas vezes e vamos comparar os valores, cada comparação má sucedida, isto é, cada comparação que não satisfaz a precisão executamos 1000 simulações a cada bateria de ensaios.

Um código sequencial foi previamente implementado e testado, a simulação de uma bateria de ensaios com 5000 ciclos teve em média um tempo de execução de 7 segundos. Implementado posteriormente à análise da incerteza teve seu tempo de execução variando entre 14 segundos, encontrando um valor esperado na sua primeira execução.

```
SimulacaoComErro(5000,1)
Começou
5000
66.8894 - 66.8178
Precisão alcançada   0.1  >  0.07159999999998945
```

Alcançando valores de até 20000 ciclos de simulação com tempo de execução de 2 minutos e 21 segundos,

```
SimulacaoComErro(5000,1)
Começou
5000
66.2852 - 67.1292
0.8439999999999941  >  0.1
Começou
10000
67.4484 - 67.1233
0.32510000000000616  >  0.1
Começou
15000
67.867 - 67.0859
0.7811000000000092  >  0.1
Começou
20000
66.7 - 66.6727
Precisão alcançada   0.1  >  0.02729999999999677
```

Valores para duas casas decimais não foram encontrados em menos de 40000 ciclos de simulação

2. Projeto da solução concorrente

A divisão de tarefas do nosso código sequencial se divide na simulação do algoritmo, no cálculo da média dessas simulações e na validação da incerteza dessa média.

A simulação do algoritmo, ainda que possa ser custosa, é uma ação atômica, pois o comportamento da variável é algo que deve ser observado de forma sequencial, então devemos dividir as tarefas de forma concorrente no cálculo da média dessas simulações e na validação da incerteza do valor esperado.

Para o cálculo da média de cada bateria de ensaios podemos dividir a quantidade de ciclos de simulação do somatório dos valores obtidos pela quantidade de threads usadas, de forma que, cada bateria de ensaios possua a mesma quantidade de threads. Se usarmos 2 threads, 1 thread trabalha em cada bateria, 4 threads 2 trabalham em cada bateria e assim por diante.

Para abordar a validação da incerteza dos dados podemos utilizar uma sincronização por barreira, aguardando que todas as simulações estejam concluídas e comparando se a diferença entre as simulações atende a precisão deseja, se não, mais 1000 ciclos são adicionados à simulação e a aplicação reinicia o processamento dos dados

3. Casos de teste e corretude e desempenho

O desempenho do código deve ser provado na obtenção de um valor com uma incerteza desejada em um tempo mais satisfatório que o tempo obtido no teste sequencial, de forma que o speed up seja ao menos linear ao uso dos processadores.

A avaliação de corretude do código, pode ser obtida sem muita complexidade sabendo que a tarefa se trata de uma análise de dados

externos, logo, com o cruzamento de informações de ambos os códigos, sequencial e concorrente, os mesmos deverão apresentar valores similares.

Depois que toda a aplicação for executada ela irá retornar o valor esperado do algoritmo e o tamanho do espaço amostral, que é traduzido pela quantidade de ciclos necessários para a obtenção do valor esperado, então para validar a corretude, apenas executamos o algoritmo de forma sequencial com o número do espaço amostral retornado, o valor obtido deve ser similar nas duas formas.

4. Referências Bibliográficas

- [DF] David Forsyth - Probability and Statistics for Computer Science
- [NR] Nei Rocha - Estatística e Probabilidade