O Primeiro Código
Atividade 1.3

Luiz Fellipe Machi Pereira

I. Introdução

A computação paralela é um tipo de computação em que muitos cálculos ou a execução de processos são realizados simultaneamente. Muitas vezes, grandes problemas podem ser divididos em outros menores, que podem ser resolvidos ao mesmo tempo. Neste relatório apresentamos e analisamos o comportamento de um programa executado sequencialmente e paralelamente com variações na entrada e ma quantidade de threads.

II. Metodologia

Para comparação de desempenho entre uma execução sequencial e um paralela, bem como o impacto causado pela variação da quantidade de threads utilizadas, foi utilizado a implementação para resolução do *Conta X*. O problema *Conta X* consiste em contar quantas vezes um determinado elemento *X* aparece em uma sequencia de caracteres. Em sua versão sequencial, o vetor é percorrido em um laco, elemento por elemento, e quando o elemento é encontrado um contador é incrementado. Na versão paralela, o vetor é dividido em partes, em suma de mesmo tamanho, de acordo com a quantidade de threads solicitadas, no qual cada thread fica responsável por contar quantas vezes o elemento apareceu naquela fatia do vetor, as contagens são então somadas para obter o resultado final. Para performar os testes uma maquina com as seguintes configurações foi utilizada: processador Intel(R) Core(TM) i5-9300H CPU @ 4,1 GHz GHz com 8 CPUs (4 cores por socket e duas threads por core) e 128 KiB de memoria cache L1; 8 Gb de memoria RAM distribuídas em blocos de 128 Mb; sistema operacional Arch Linux, com kernel Linux 5.8.5-arch1-1 e compilador GCC na versão 10.2.0.

III. Resultados

Devido a restrições de memoria, a quantidade de elementos no vetor precisou ser reduzida para evitar falhas referentes a alocação de memoria, dada esta restrição, o tempo de execução de cada programa foi reduzido, bem como a expressividade da diferença de tempo entre a execução sequencial e paralela. Para computar os valores referencia, todas as variações foram testadas três vezes e a média entre elas foi tomada e um gráfico de Speedup[1] foi confeccionado (Figura 1). Os tamanhos dos vetores utilizados, representados por valor 1, 2 e 3, foram, respectivamente 183000000, 183300000 e 183700000; preenchidos com o valor 3 em 10% de cada vetor e os demais com valores aleatórios diferentes de 3.

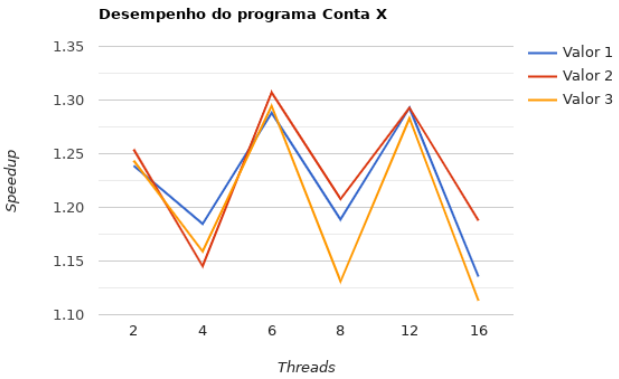

Fig. 1. Gráfico de Speedup para diferentes tamanhos de vetores.

Em geral, gostaríamos que sempre que aumentássemos a quantidade de threads utilizadas em um programa, o seu desempenho aumentasse da mesma forma, porém em casos práticos não é isto que ocorre. Diversos fatores podem interferir na execução de um programa e fazer com que o desempenho de uma thread seja superior a outra, tais como escalonamento realizado pelo SO, paginação de cache, quantidade de memoria disponível por cache, quantidade de chamadas para realizar operações atômicas,etc. A Figura 1 exemplifica a proposição anterior. Apesar do aumento na quantidade de Threads utilizadas o valor de Speedup não teve um crescimento linear, pelo contrario, apresentou variações positivas e negativas.

Referências

[1] LIN, Calvin et al. Principles of parallel programming. Pearson Education India, 2008.

Universidade Estadual de Maringá, Departamento de Informática

[1]Speedup é uma métrica que mede o desempenho relativo de dois sistemas processando o mesmo problema. Representa a melhoria na velocidade de execução de uma tarefa executada em duas arquiteturas semelhantes com recursos diferentes.