Paralelização de Algoritmos utilizando OpenMP

Carlos Alberto Franco Maron[1]

[1] Programa de Pós-Graduação em Ciência da Computação
Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil

carlos.maron@acad.pucrs.br

Resumo. *Este trabalho apresenta resultados da paralelização de algoritmos utilizando a biblioteca OpenMP. O trabalho foi dividido em duas etapas. A primeira consiste na implementação de um algoritmo para encontrar ocorrências de palavras em um texto. A segunda etapa é solucionar um deadlock colocado de maneira proposital no algoritmo, e posteriormente paralelizá-lo*

1. Implementação

As etapas do trabalho foram seguidas utilizando a linguagem de programação em C e a biblioteca OpenMP para paralelização dos algoritmos.

A primeira etapa do trabalho propõe a implementação de um algoritmo que faça a busca de palavras em um texto, e encontre a quantidade de ocorrências destas palavras. O código desenvolvido para solucionar esse problema é executado da seguinte maneira, ao passar através por parâmetro o arquivo texto que deve ser lido, o algoritmo faz uma leitura inicial completa para encontrar o tamanho total deste arquivo. Posteriormente, ele é armazenado em memória, realizando a busca da quantidade total das palavras e também encontrando os delimitadores de cada palavra. O ponto aonde o paralelismo ocorre é no momento que o algoritmo está buscando as ocorrências das palavras no texto.

A segunda parte do trabalho do trabalho tem o propósito de solucionar o problema de um algoritmo. Propositalmente o algoritmo foi modificado para que sua execução fosse comprometida. Basicamente o algoritmo realiza o produto de dois vetores utilizando duas constantes. O primeiro problema ocorre durante a inicialização do vetor, que devido ao seu tamanho, a forma como foi implementado não utiliza a alocação dinâmica em memória, e isto reflete no problema conhecido como ”estouro de pilha”. Desta forma, a solução para este problema foi a adequação deste vetor de forma dinâmica. Neste algoritmo, são inicializadas duas variáveis do tipo *lock*, ao qual é destinada para controlar o acesso ao vetor criado anteriormente. Esses *locks* são utilizados dentro de duas seções paralelas que realizam o cálculo dos vetores. O segundo problema encontrado neste algoritmo foi a ocorrência de *deadlock* devido a posição errada no qual foi inserido os *locks* dentro do código.

A etapa adicional do segundo exercício consiste na paralelização deste algoritmo de cálculo do vetor. O detalhe importante desta parte, é que o algoritmo está divido em duas seções, e para que ocorra o paralelismo dentro das seções, é importante utilizar o parâmetro *omp_set_nestead(1)*, que permite a a criação de novas *threads* em seções já paralelas.

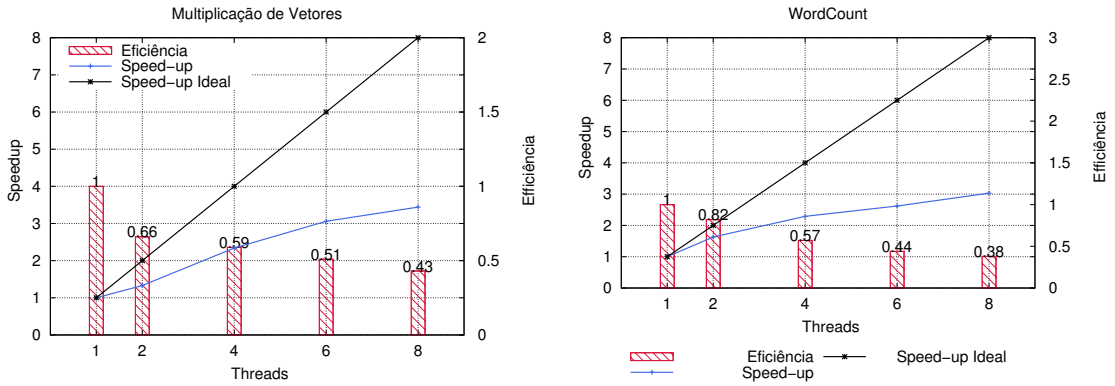

Figure 1. Desempenho dos Métodos de Escalonamento no OpenMP.

2. Resultados

A Figura 1 apresenta os resultados obtidos pelo paralelismo dos algoritmos de contagem de palavras e de multiplicação de vetores.

3. Dificuldades Encontradas

Durante a realização da primeira etapa do trabalho, o código inicial estava sendo desenvolvido em C++. Porém, algumas limitações da linguagem, fizeram que o algoritmo desenvolvido não fosse capaz de ser paralelizado. Há o que indica, que algumas funções da linguagem acabam interferindo no endereçamento de memória dos vetores dentro da seção paralela. Implementando o segundo algoritmo, os métodos usados dentro da seção paralela não favoreceram os resultados, mostrando um *speed-up* pouco expressivo.

Um dos desafios do exercício 2 foi perceber a utilização do parâmetro *nestead* e a ocorrência dos *deadlocks*. A pouca experiência pode ser justificativa pela dificuldade em encontrar o problema. Porém, os resultados obtidos com *speed-up* deixa em dúvida se o tipo de entrada usado no algoritmo tem um ganho eficiênte.

4. Conclusões

Neste trabalho, foram apresentados alguns resultados da paralelização de algoritmos utilizando a biblioteca OpenMP. Todos os testes e implementações realizadas neste trabalho serviram para o estudos de algumas das principais diretivas da biblioteca de programação.

As etapas deste trabalho nos permitem mostrar outros domínios para se obter paralelismo com OpenMP. Foi possível perceber algumas limitação do mesmo, como relatado na seção das dificuldades encontradas. O exercícios 1 é importante ser reavaliado, pois a forma que foi implementado o algoritmo não favoreceu o seu ganho.