Identificação

Osvaldo Henrique Becker - 175963@upf.br

Implementação

Pequenas modificações foram feitas no código, como por exemplo os argumentos recebidos via linha de comando. O código original tem dois argumentos opcionais, um define a quantidade de loops da função principal e o outro define o nome do arquivo de entrada, que se omitido a entrada deve ser digitada ou direcionada. Já o novo código tem dois argumentos obrigatórios, em que um deles define o número de threads em que o programa será executado e o outro define o nome do arquivo de entrada. No novo código a quantidade de loops da função principal é obtida somente através do arquivo de entrada. Se o número de threads informado não for aceitável ou o arquivo de entrada não for encontrado ou tiver erros o programa mostra o erro ocorrido e aborta a execução.

Recursos de programação paralela utilizados

Na parte de paralelização foram utilizados recursos da OpenMP, que é uma API que fornece recursos de programação paralela em C, C++ e Fortran. Apenas cinco linhas referentes a paralelização foram adicionadas no código. Uma delas é a inclusão do cabeçalho omp.h, outra é uma função que define o número de threads em que o programa será executado e as outras três cada uma divide as iterações de um respectivo loop entre as threads.

Análise dos resultados dos testes

Modo de execução	**Tempo de execução**
Sequencial	4m 26s
Paralelo 2 threads	2m 24s
Paralelo 4 threads	1m 23s
Paralelo 8 threads	1m 4s
Paralelo 16 threads	1m 5s

O teste foi realizado em uma máquina com um processador com 4 núcleos físicos, por isso o tempo de execução reduziu significativamente ao ser executado em até 4 threads. Em 8 ainda reduziu um pouco o tempo de execução e por fim, em 16 passou a igualar ou piorar.

Conclusão

Conclui-se que é possível obter bons resultados ao paralelizar um programa, porém dentro de algumas limitações e observações. Limitações como núcleos físicos do processador utilizado e observações de como o código foi paralelizado, já que paralelização desnecessária ou mau utilizada e execução do programa em muitas threads lógicas pode piorar os resultados.