Universidade Estadual de Campinas

Faculdade de Engenharia Elétrica e de Computação

Disciplina: EA876 – Introdução a Software de Sistema

Discentes: Guilherme B. Abreu, RA: 173691

André M. Yoshida, RA: 166241 Data:06/06/2018

**Análise de desempenho entre programas Multithread e Multiprocesso**

**Introdução**

O objetivo do projeto realizado foi o de comparar o desempenho de dois programas que realizassem computação paralela para cumprir uma mesma tarefa utilizando o mesmo padrão de design, porém, um deles seria paralelizado com threads e o outro com processos. Além disso, o único fator de desempenho considerado foi o tempo gasto pelo programa ao desempenhar o serviço.

Para fazer a análise, a tarefa escolhida foi aplicar um filtro do tipo blur em uma imagem.

**Método**

O método utilizado para paralelizar os dois programas é baseado no padrão de design Thread Pool. Uma lista de tamanho igual ao número de pixels da imagem é compartilhada entre os processos (ou threads) e nela é guardada o estado de cada pixel, isto é, se o filtro já foi aplicado naquele pixel ou não. No início do programa esta lista é inicializada com nenhuma posição marcada. Cada processo percorre esta lista procurando por posições não marcadas. Quando um processo encontra uma dessas posições, ele aplica o filtro no pixel que está relacionado com aquela posição, e então ela é marcada. Assim, a tarefa é realizada apenas uma vez em cada pixel.

**Resultados**

Para a realização do experimento foi utilizado um computador com 2 núcleos de processamento, cada um com 2 processadores, operando em uma frequência base de 2.9GHz e arquitetura Intel i7-7500U. Além disso, operava sobre ele o sistema Ubuntu 16.04 LTS.

Neste ambiente, para um filtro blur de comprimento 10, e executando cada programa 50 vezes numa imagem com resolução 1920 x 1280, as médias de tempo obtidas foram as seguintes:

* Com 4 processos: 3.4320 ± 0.3112 [s]
* Com 4 threads: 3.2904 ± 0.4249 [s]

**Discussão e Conclusão**

Analisando os resultados, podemos observar que apesar do tempo médio obtido utilizando threads ter sido menor, não podemos inferir que os threads foram mais rápidos que os processos. Pois vemos que o desvio padrão no experimento é consideravelmente grande, ou seja, houveram casos em que os processos aplicaram o filtro com um desempenho maior.

Isso era de fato esperado, pois a comutação entre threads é mais rápida e pelo fato de threads compartilharem a memória o custo para criá-los é bem menor também, o que justifica o tempo médio menor. Outro fato esperado é os altos valores para o desvio padrão, já que os programas foram executados apenas 50 vezes.

Assim, concluímos que threads em média tiveram um desempenho maior em relação a processos.