8º Exercício Prático
Desenvolvido no Laboratório

Objetivo

Observar o impacto da memória cache no tempo de execução de diferentes tamanhos de matrizes.

Materiais

1. Compilador GCC
2. Arquivo Cache.cpp
3. Imagem Lapis.ppm

Desenvolvimento

O exercício de hoje consiste em transformar para preto e branco regiões da imagem de tamanhos diferentes processando cada pixel linha por linha e comparando os tempos com o processamento feitos coluna por coluna das mesmas regiões.

Executando o programa sem otimização

1. Compile o programa Acessos.cpp
    - gcc -g -O2 Cache.cpp -o Cache
2. Execute o programa.
    - ./Cache

Qual foi o tempo observado na execução?

Tempo 1: _____ s

Processando partes da imagem

1. Abra o arquivo Cache.cpp em um editor de texto
2. Observe que existe uma constante K definida a linha 11 do Cache.cpp, que aparece também na função processa() e na função main()
3. Procure a função processa()

```
void processa(struct Pixel img[ALTU_IMG][LARG_IMG],
              struct Pixel imgSai[ALTU_IMG][LARG_IMG]) {
  int i, j, tomDeCinza;
  for (i = 0; i < ALTU_IMG / K; i++) {
    for (j = 0; j < LARG_IMG / K; j++) {
      struct Pixel pixel = img[i][j];
      // Calcula o tom de cinza correspondente ao RGB do pixel:
      tomDeCinza = 0.299 * pixel.r + 0.587 * pixel.g + 0.114 * pixel.b;
      pixel.r = tomDeCinza;
      pixel.g = tomDeCinza;
      pixel.b = tomDeCinza;
```

```
      imgSai[i][j] = pixel;
    }
  }
}
```

4. Altere a constante K na linha 11 para o valor 16

   ```
   #define K 16
   ```

   Qual foi o tempo observado na execução?

   Tempo 2: _____ s

5. Altere agora a constante K na linha 11 para o valor 128

   ```
   #define K 128
   ```

   Qual foi o tempo observado na execução?

   Tempo 3: _____ s

   Observe as linhas 99, 100 e 118. Explique por que os tempos 1, 2 e 3 são praticamente iguais. Para explicar pense: quantos pixels iguais ou diferentes são processados no total pelos códigos nas linhas de 118 a 120? Com K = 1, são processados 2560 * 1600 * 30 * 1 * 1 = 122880000 pixels, mas quantos são processados com K = 16 e com K = 128? O que acontecerá com a imagem gerada com K = 16? **Coloque a explicação no documento de análise de desempenho.**

Influência da Cache na execução

1. Troque de posição dos dois for's das linhas 99 e 100, deixando a função processa como apresentada abaixo:

```
void processa(struct Pixel img[ALTU_IMG][LARG_IMG],
              struct Pixel imgSai[ALTU_IMG][LARG_IMG]) {
  int i, j, tomDeCinza;
  for (j = 0; j < LARG_IMG / K; j++) {    // linha 99
    for (i = 0; i < ALTU_IMG / K; i++) {  // linha 100
      struct Pixel pixel = img[i][j];
      tomDeCinza = 0.299 * pixel.r + 0.587 * pixel.g + 0.114 * pixel.b;
      pixel.r = tomDeCinza;
      pixel.g = tomDeCinza;
      pixel.b = tomDeCinza;
      imgSai[i][j] = pixel;
    }
  }
}
```

2. Faça agora três medidas de tempo, para K = 1, K = 16 e K = 128

Qual foi o tempo observado na execução com K = 1?

Tempo 4: _____ s

Qual foi o tempo observado na execução com K = 16?

Tempo 5: _____ s

Qual foi o tempo observado na execução com K = 128?

Tempo 6: _____ s

Analisando os resultados

Explique por que alterar o K não altera o número de pixels processados no programa.

Descreva a imagem obtida e explique o motivo de ter ficado como observada.

Calcule agora as relações $\frac{tempo\,4}{tempo\,1}$, $\frac{tempo\,5}{tempo\,2}$, $\frac{tempo\,6}{tempo\,3}$. Apresente o valor das relações e **tente** explicar por que as relações não são próximas, mesmo que o número de dados processados nas imagens tenham sido iguais. Observe que áreas menores de memórias possibilitam que todas essas áreas sejam armazenadas na cache ao mesmo tempo.

Envie a avaliação dos resultados como descrito no arquivo “Avaliacao Dos Resultados.pdf”, mas inclua uma comparação entre os pares de tempos e as explicações que conseguir fazer.

Conclusão

Quantias menores de dados processados possibilitam que todos estes dados permaneçam na memória cache o que ajuda no desempenho, por outro lado, em processamentos que envolvem mais dados do que podem ser mantidos na cache ao mesmo precisam que esses dados recebam acessos preferencialmente na mesma localidade pois acessos aos dados próximos aumentam a possibilidade de serem encontrados na cache.