3º Exercício Prático
Desenvolvido no Laboratório

Objetivo

Criar um perfil de execução de um programa e otimizar um programa utilizando *lookup table* para substituir uma expressão complicada por um acesso a uma tabela.

Materiais

1. Compilador GCC
2. Ferramenta de análise de desempenho gprof
3. Arquivo Realce.c
4. Imagem utilizada no experimento da aula passada: Lapis.ppm

Desenvolvimento

O experimento de hoje utiliza alguns conceitos de representação de imagens em computadores. Para entender melhor o programa que será usado hoje, estude o conteúdo da página https://www.had2know.com/technology/hsi-rgb-color-converter-equations.html.
O programa deste experimento realça os pixels da imagem que apresentam matiz (*hue*) próxima a uma cor específica, transformando os outros pixels em níveis de cinza. Para entender como transformar pixels RGB em tons de cinza, leia https://pt.wikipedia.org/wiki/Nível_de_cinza .

Executando o programa original

1. Compile o programa Realce.c
    - gcc -pg Realce.c -o Realce -lm
    - O “-pg” faz com que o compilador instrumente o código gerado para fazer medidas e criar um perfil de execução do programa, O “-lm” liga a biblioteca com as rotinas de “math.h” ao seu programa executável.
2. Execute o programa
    - ./Realce
3. Visualize a imagem LapisSai.ppm, abra o arquivo Realce.c e analise o que a função processa() e as chamadas por ela.

   Qual foi o tempo observado na execução?

   Tempo 1: _____ s

4. Visualize o perfil de execução do programa usando o gprof
    - gprof ./Realce

Responda:

Observando a saída do gprof, qual é a função que mais gasta tempo para executar?

Considerando a função que mais gasta tempo executando, anote quanto tempo em porcentagem esta função gasta em relação ao tempo total de execução do programa.

Tempo 2: _____ %

Otimizando o código

Linhas como "**`int hue = round(acos(numerador / denominador) * 180 / M_PI);`**", que usam funções trigonométricas e muitas operações aritméticas, podem levar muito tempo para executar. Substituir linhas como esta por buscas em vetores pode impactar no tempo de execução.

1. Abra o arquivo Realce.c em um editor de texto
2. Inclua o vetor e a função no programa Realce.c. A função "**`criaMatHue()`**" inicia o vetor para que ele possa substitui a linha do calculo do matiz (Hue).

```
int matHue[201];

void criaMatHue() {
  for (int i = 0; i <= 200; i++) {
    matHue[i] = round(acos((i - 100) / 100.0) * 180 / M_PI);
  }
}
```

3. Chame a função "**`criaMatHue()`**" no início da função "**`main()`**".
4. Substitua a linha "**`int hue = round(acos(numerador / denominador) * 180 / M_PI);`**" por

```
int hue;
if (denominador != 0) {
  int indice = numerador * 100 / denominador + 100;
  hue = matHue[indice];
} else {
  hue = 0;
}
```

Note que a seção de código acima simplifica a linha substituída. O valor obtido do vetor **`matHue`** substitui a chamada da função "**`acos()`**", o arredondamento feito pela "**`round()`**" e a transformação de radianos para graus e de float para int.

5. Compile novamente com “-pg” e execute.

   A tabela de *lookup* armazena uma aproximação da linha substituída, portanto a imagem não é exatamente igual, mas visualmente ela deve ser muito parecida. Você consegue ver a diferença?

   Qual foi o tempo observado na execução?

   Tempo 3: _____ s

6. Visualize o perfil de execução do programa usando o gprof

   Responda:
   Observando a saída do gprof, qual é a função que mais gasta tempo para executar?

   Quanto tempo em porcentagem, a função que mais gasta tempo executando gasta em relação ao tempo total de execução do programa?

   Tempo 4: _____ %

   Explique com as suas palavras o que programa faz? Como alterar a cor realçada para vermelho?

Avaliando os resultados

Envie a avaliação dos resultados como descrito no arquivo “Avaliacao Dos Resultados.pdf” na atividade da semana passada.

Conclusão

Algumas expressões complexas podem ser substituídas por acessos a tabelas, resultando em grande economia de tempo de execução