4º Exercício Prático
Desenvolvido no Laboratório

Objetivo

Comparar duas ferramentas de levantamento de perfis de execução de programas.

Materiais

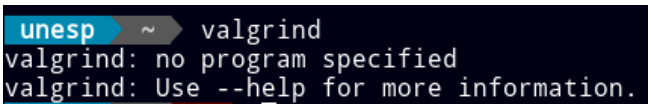
1. Compilador GCC
2. Ferramentas de análise de desempenho Gprof e Callgrind do Valgrind
3. Arquivo Acessos.cpp
4. Arquivo com textos: biblia-em-txt.txt.

Desenvolvimento

O experimento de hoje usa um programa que faz a contagem de ocorrências de uma letra em um arquivo de texto. Duas ferramentas de levantamento de perfis serão empregadas ao programa com o objetivo de compará-las. No desenvolvimento serão analisadas as rotinas que mais precisam de tempo para executar e como elas podem ser substituídas para melhorar o desempenho.
Para identificar os pontos onde a otimização pode gerar melhores resultados (*hotspots*), será usada a ferramenta Callgrind do Valgrind.

Preparanto o ambiente

1. Para usar o Valgrind é necessário instalá-lo, para isso abra um terminal e use os seguintes comandos
    - Os dois primeiros comandos são apenas para corrigir alguma instalação feita anteriormente que não foi concluída.
        - sudo dpkg --configure -a
        - sudo apt --fix-broken install
            - Quando questionado se “Você quer continuar? [S/n]”, pressione <Enter>
    - O próximo comando instala o Valgrind e o kcachegrind. Este último apresenta uma visualização gráfica do perfil de execução.
        - sudo apt install valgrind kcachegrind
            - Quando questionado se “Você quer continuar? [S/n]”, pressione <Enter>
    - Após a instalação, mas ainda no terminal, tente executar o comando:
        - valgrind
    - Se funcionar, será apresentado algo como:



Executando o programa com o Gprof

1. Compile o programa Acessos.cpp

    - gcc -g -pg Acessos.cpp -o Acessos
- O “-g” faz com que o compilador prepare o código para que possa ser depurado, mantendo informações das variáveis e das linhas do código fonte e disponibilizando informações para auxiliar aos programadores detectarem erros no programa. O Valgrind usa essas informações de depuração para apresentar informações mais claras aos programadores.

2. Execute o programa. Esta execução gera também o arquivo “gmon.out” com os dados do perfil. Use o comando:
    - ./Acessos

3. Observe a saída do programa Acessos e responda:

   Quantas vezes a letra ‘a’ aparece no texto?

   Ocorrências da letra ‘a’: _____

   Qual foi o tempo observado na execução?

   Tempo 1: _____ s

4. Executando o gprof para visualizar o perfil de execução do programa. O gprof usa o arquivo “gmon.out” para representar os dados do perfil da última execução. Use o comando:
    - gprof ./Acessos

5. Observe a saída do gprof e responda:

   Excluindo a função main(), qual a rotina mais executada no programa?

   Ocorrências da letra ‘a’: _____

   Tente identificar onde podemos focar os esforços para melhorar o tempo de execução do programa. Note que as rotinas fopen, fgetc, feof, fclose, não aparecem no perfil de execução.

Executando o programa com o Valgrind

1. Para que o Valgrind levante o perfil de execução, ele deve ser chamado com a indicação do programa que será executado, pois o Valgind chamará o programa que se deseja obter o perfil. Portanto, execute o comando
    - valgrind --tool=callgrind ./Acessos
- O Valgrind executa o programa em uma máquina virtual para levantar o perfil de execução. Por isso, o programa demora muito mais tempo para executar. Durante a execução um arquivo chamado “callgrind.out.<id>” é criado com o perfil da execução. O <id> é o número de identificação do processo de execução do programa.

2. Após a execução, para visualizar o perfil, use comando:
    - callgrind_annotate --auto=yes callgrind.out.<id>
    - para identificar o id, veja na pasta atual qual é a data e hora que o arquivo callgrind.out.<id> foi criado.
3. Procure na saída do callgrid_annotate a linha "-- Auto-annotated source: Acessos.cpp". Abaixo desta linha, encontra-se o código fonte com as indicações do número de instruções lidas (Ir) e executadas em cada linha e suas porcentagens. Observe que algumas linhas extras, contendo o texto " => ???:0x", aparecem seguindo a linha do fonte onde funções de bibliotecas são chamadas. Cada uma destas linhas representam os dados da execução de uma rotina da biblioteca.

   Observando a saída do callgrid_annotate, responda qual é a função das bibliotecas padrões da linguagem C que tem o maior impacto no tempo de execução do programa?

   Nome da função: _____

   Considerando esta função, anote quanto tempo em porcentagem esta função gasta em relação à execução total do programa.

   Porcentagem: _____ %

Otimizando o programa Acessos.cpp

Sabendo qual a rotina que mais tem impacto na execução do programa, pense numa forma de otimizar o programa, teste a sua ideia e tente explicar o motivo da sua otimização funcionar. A entrega desta otimização não é obrigatória e não contará nas avaliações.

Avaliando os resultados

Envie a avaliação dos resultados como descrito no arquivo "Avaliacao Dos Resultados.pdf".

Conclusão

Existem muitas ferramentas de levantamento de perfil de execução de programa, cada uma com suas características e formas diferentes de apresentação das informações de perfil obtidos.