DGPAPP Tutorial

Daniel Centeno Einloft
Vinicius Meirelles Pereira

20 May 2014

Introdução

Na programação paralela, o principal objetivo é diminuir o tempo de execução de um algoritmo através da sua divisão em diferentes blocos de execução, que serão executados por diversos processadores ou núcleos. Existem várias maneiras de testar se um algoritmo paralelo é melhor que o sequencial, e todas dependem dos seus tempos de execução. Como cada teste deve ser executado pelo menos 30 vezes, para se obter um intervalo de confiança aceitável, existe uma grande quantidade de dados gerados.

O Data Generator for Perfomance Analysis of Parallel Programs (DGPAPP) é um software livre projetado para coletar dados de testes de performance, organizá-los e realizar uma série de cálculos. Depois, o programa salva e organiza os resultados em arquivos diferentes.

Cálculos

Na sua versão atual, o DGPAPP é capaz de calcular o speedup e a eficiência, além de coletar e organizar os dados. Pretendemos adicionar novas funções no futuro, como variância, desvio padrão, entre outras informações estatísticas.

Speedup

Speedup mede o quão rápido um algoritmo paralelo é, quando comparado com sua versão sequencial. Ele é definido por:

$$S_t = \frac{T_1}{T_t}, \tag{1}$$

onde:
$_t$ é o número de threads ou processos;
S_t é o speedup;
T_1 é o tempo de execução do algoritmo sequencial;
T_t é o tempo de execução do algoritmo paralelo.

O speedup ideal é obtido quando $S_t = t$.

Assim, o speedup só depende dos tempos médios do algoritmo sequencial e paralelo.

Eficiência

A eficiência de um algoritmo paralelo mostra o quanto dos processadores ou núcleos são utilizados durante a sua execução. Este valor é útil pois o speedup só mede a velocidade do algoritmo, sem considerar o número de processadores ou núcleos utilizados. Portando, se um algoritmo rodar duas vezes mais rápido que o sequencial, mas for dividido entre oito threads, seu speedup é alto, mas sua eficiência não, e assim, provavelmente não é uma boa opção.

A eficiência pode ser obtida através da equação:

$$E_t = \frac{S_t}{t} = \frac{T_1}{tT_t}, \tag{2}$$

sendo:
$_t$ o número de threads ou processos;
E_t o valor da eficiência;
S_t o valor do speedup;

T_1 o tempo de execução do algoritmo sequencial;
T_t o tempo de execução do algoritmo paralelo.

Instalação

Como Instalar

Para instalar o DGPAPP, abra o diretório em que ele está e execute o seguinte comando no terminal do linux:

$ sudo make

Então, execute:

$ sudo make install

Depois disso, DGPAPP vai ter sido instalado em /usr/local/DGPAPP/.
Caso deseje compilar o programa de forma local, execute:

$ sudo make local

Com esse comando, DGPAPP vai ter sido compilado, e seu executável estará na pasta *bin*.

Como Desinstalar

Se você deseja desinstalar o programa, execute o seguinte comando no diretório do DGPAPP:

$ sudo make uninstall

Como Usar

Lista de Comando

- **-h or –help** : mostra a lista de comandos.
- **-v or –version** : mostra a versão e os créditos do programa.

- **-f** : especifica o filtro.
- **-l** : especifica o nome do arquivo de log.
- **-t** : especifica o número de threads do arquivo de log atual.
- **-c** : especifica o comentário que vai ser mostrado na base de cada coluna de dados dos arquivos de saída.
- **-o** : especifica o nome do arquivo de saída.
- **-p** : especifica o número de casas decimais dos dados nos arquivos de saída.

Usando a Linha de Comando

Os comandos podem ser inseridos em qualquer ordem, desde que o número de threads seja indicado depois do nome do arquivo de log, pois o número de threads é associado com este arquivo.

Um exemplo da sintaxe é:

$ dgpapp -f filtro -l log -t numero_de_threads -l log -t numero_de_threads -l ... -c comentario -o `nome_do_arquivo_de_saída` -p casas_decimais

O filtro é a palavra que o DGPAPP vai procurar nos arquivos de log. Ele vai procurar por todas as ocorrências desse filtro, e salvar os valores numéricos que as seguem. Log é o nome do arquivo de log, que deve ser sucedido pelo seu número de threads. O comentário é o nome que vai ser adicionado no final de cada coluna de dados gerada para identificar o teste. Como as colunas vão ser reorganizadas de forma que seus tamanhos sejam decrescentes, da esquerda para a direita, é muito importante, mas não obrigatório, adicionar um comentário. O nome do arquivo de saída vai ser o nome genérico dos arquivos que serão gerados. Seus nomes vão conter o nome escolhido, seguido pela variável calculada naquele arquivo. O número de casas decimais, como o comentário, não é obrigatório, e se não for informado, o programa irá usar um valor padrão (duas casas decimais). A ordem dos comandos é irrelevante.

Tantos os arquivos de log como o arquivo de saída podem ter seus caminhos indicados (por exemplo, -l /home/log.txt, ou -o /home/testname). O filtro pode ser composto por mais de uma palavra, e, neste caso, deve ser colocado entre aspas.

Erros Comuns

- Número de threads não foi separado de "-t" por um espaço: retorna uma mensagem de erro e o programa é terminado.
- Arquivo de log sequencial não foi inserido na linha de comando: retorna uma mensagem de erro e o programa é terminado.
- Comentário não foi utilizado: não é um erro, porém pode resultar em confusão durante a identificação dos testes.

Arquivos de Saída

O DGPAPP vai gerar três arquivos de saída: efficiency.dat, speedup.dat e averagetime.dat, correspondendo aos arquivos que possuem as informações sobre a eficiência, speedup e tempo médio, respectivamente. A estrutura desses arquivos, porém, precisa ser explicada com mais detalhes.

A primeira vez que o usuário executar o programa através da linha de comando do terminal do linux, serão gerados os arquivos de saída, com os dados organizados em duas colunas: uma para os valores dos números de threads utilizadas e outra para os seus respectivos valores. Essa estrutura só não está presente no arquivo de tempos, onde cada coluna terá os valores de tempo para cada thread, que vai ser indicada por um # na ultima linha.

Quando o usuário executar o programa de novo, serão geradas mais duas colunas. Porém, elas não necessariamente vão estar localizadas à direita das colunas anteriores. Devido ao fato que o programa foi feito para ser utilizado em conjunto com o Gnuplot (uma ferramenta para criação de gráficos) as colunas devem ser organizadas de forma que as novas tenham o mesmo ou menor tamanho que as duas à sua esquerda. Portanto, as colunas vão ser sempre organizadas das maiores às menores, da esquerda para direita.

Nas Figuras 1, 2 e 4, podem ser visualizados exemplos dos arquivos de saída.

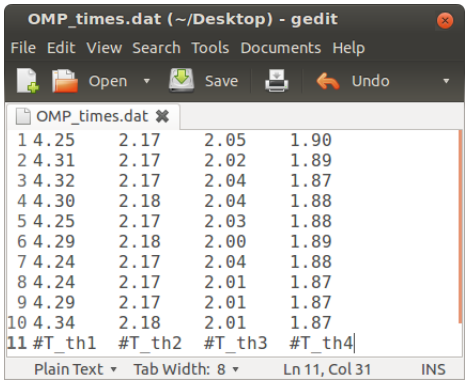

Figure 1: Exemplo do arquivo times.dat

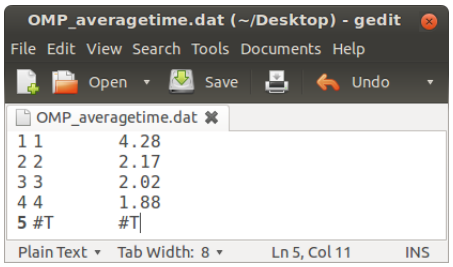

Figure 2: Exemplo do arquivo averagetime.dat

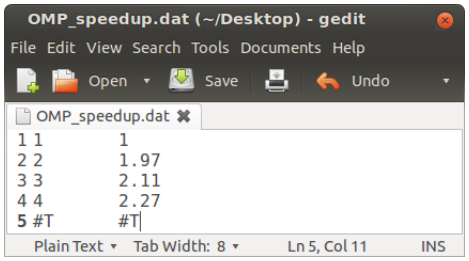

Figure 3: Exemplo do arquivo speedup.dat

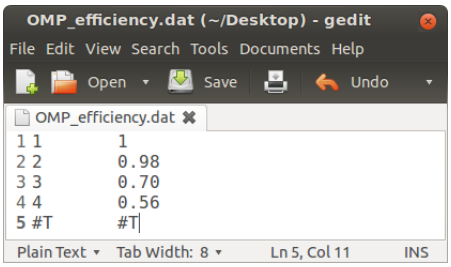

Figure 4: Exemplo do arquivo efficiency.dat

Note que nos exemplos, o nome padrão para os arquivos foi "OMP" e o comentário para o teste foi "T".