Comparação/Confirmação dos Resultados Obtidos com *IOZone Benchmark*, com a *Framework Dtrace*

Sérgio Caldas
Universidade do Minho
Escola de Engenharia
Departamento de Informática
Email: a57779@alunos.uminho.pt

***Resumo*—Neste trabalho, desenvolvido no âmbito da disciplina de Engenharia de Sistemas de Computação (ESC), inserida no perfil de Computação Paralela e Distribuída (CPD) do curso de Engenharia Informática, tem como objetivo estudar uma variada gama de testes, constituintes do *IOZone Benchmark*. Este *Benchmark*, é usado para fazer testes de performance de *Filesystems*. Depois de definidos os testes que vou usar, posteriormente, tenho de confirmar esses resultados com a ferramenta *Dtrace*, criando diversas scripts que irão fazer traçados dinâmicos de forma a obter os mesmos resultados que foram obtidos na execução do *IOZone Benchmark*.**

1. Introdução

Com a resolução deste trabalho, é suposto testar o *Filesystem* (no meu caso) da máquina *Solaris 11*, recorrendo a diversos testes do *IOZone Benchmark*.

Os testes que irei efectuar baseiam-se sobretudo na análise de operações, tais como, *Read, write, re-read, re-write*.

Depois de efetuados os testes em cima referidos, utilizarei a *Framework DTrace*, com a criação de diversas scripts em *D* capazes de calcular os mesmos resultados que foram obtidos com o *IOZone Benchmark*, por forma a poder comparar/comprovar os dois resultados.

2. Ferramentas Utilizadas

Na realização deste trabalho, para além do *IOZone Benchmark*, irão ser utilizadas mais duas ferramentas, a ferramenta *DTrace* e a ferramenta *truss* (ferramenta equivalente ao *strace* contudo é usada em *Solaris*).

O *DTrace* é uma *Framework* que permite fazer traçados dinâmicos, esta é usada para solucionar problemas no *Kernel* e aplicações em produção, em tempo real. O *Dtrace* pode ser utilizado para se obter uma visão geral da execução do sistema, como a quantidade de memória, o tempo de CPU, os recursos usados por os processos activos. Esta *Framework* permite fazer traçados muito mais rebuscados e detalhados, tais como, por exemplo a lista de processos que tenta aceder a um ficheiro.

No âmbito deste trabalho a *Framework DTrace*, é utilizada (através do uso *Scripts* em *D*) com o intuito de se obter os mesmos (ou aproximadamente os mesmos) resultados que se obteve com o *IOZone Benchmark*, por forma a confirmar/comparar esses resultados.

A ferramenta/comando *truss*, é uma ferramenta que executa um determinado comando e produz um traçado de todas as chamadas ao sistema que esse comando produz, os sinais que o comando recebe, e as falhas da máquina que ocorrem. Cada linha do *output* retorna a falha ou o nome do sinal ou o nome da chamada ao sistema com os argumentos e valores de retorno.

No âmbito deste trabalho, a ferramenta *truss* é usada de forma a nos ajudar a compreender o comportamento de toda a aplicação que estamos a executar (neste caso o *IOZone Benchmark*), ver quais as chamadas ao sistema utilizadas pela a aplicação, bem como os sinais e falhas que esta produz. Basicamente esta ferramenta é usada apenas como forma de estudar o comportamento do *Benchmark*.

3. *IOZone Benchmark*

O *IOZone* é uma ferramenta de *Benchmark* para *Filesystem's*, com este *benchmark* é possível gerar e medir uma grande variedade de operações, esta ferramenta foi adaptada para diversos sistemas, correndo em diversos sistemas operativos.

O *IOZone* é util para determinar uma basta gama de análises sobre um *Filesystem* de um determinada plataforma de um determinado vendedor. Este *Benchmark* testa o desempenho de ficheiros I/O para as seguintes operações:

- read;
- write;
- re-read;
- re-write;
- read backwards;
- read strided;
- fread;
- fwrite;
- random read/write;
- pread/pwrite variants;
- aio_read;
- aio_write;
- mmap.

Com o uso deste *Benchmark*, obtemos uma grande gama de fatores de desempenho relativos ao *Filesystem*, com isto o cliente consegue ver os pontos fortes e fracos de uma plataforma e de um sistema operativo e tomar uma decisão mais equilibrada quanto a sua escolha.

4. Testes/*Flags* a Realizar/Usar com *IOZone*

Para correr o *IOZone Benchmark*, executa-se o seguinte comando, juntamente, com as *flags* que desejarmos.

```
/opt/csw/bin/iozone
```

Nesta secção, irei apresentar todos os testes que pretendo realizar com o *Benchmark IOZone*, bem como as *Flags* que pretendo usar na execução do *Benchmark*. Todos os resultados do *IOZone Benchmark* são em Kbytes/sec.

As *Flags* que irei usar são as seguintes:

- **-I** - Esta *flag* usa *DIRECT I/O* para todas as operações de ficheiros. Diz ao *Filesystem*, para todas as operações ignorar o *buffer cache* e ir diretamente ao disco;
- **-a** - Esta *flag* é usada para ativar o modo automático, basicamente, esta flag ativa todos os testes disponíveis do *Benchmark*;
- **-r#** - Esta *flag* é usada para definir o tamanho do registo, podemos usar **-r#k** (tamanho em *Kbytes*), **-r#m** (tamanho em *Mbytes*), **-r#g** (tamanho em *Gbytes*);
- **-s#** - Esta *flag* é usada para definir o tamanho do ficheiro a testar, podemos usar **-s#k** (tamanho em *Kbytes*), **-s#m** (tamanho em *Mbytes*), **-s#g** (tamanho em *Gbytes*);
- **-t#** - Executa o *Benchmark* com *throughput mode*. Esta opção permite ao utilizador especificar quantas *Threads* ou Processos pretende ativar durante os testes;

No que toca aos testes, apresento em baixo todos os que pretendo realizar:

- ***Write* e *Rewrite*** - para este teste é usado a *flag* **-i0**. O comando completo pode ser consultado em baixo para um ficheiro de 10 Mb com um *Record Size* de 64 Kb:

```
/opt/csw/bin/iozone -i0 -r64k -s10m
```

  - ***Write*** - mede a *performance* de escrever um ficheiro novo, quando um ficheiro é escrito, não é só os dados que são escritos mas também "meta-dados". Estes "meta-dados"contém informação que permite controlar onde os dados estão armazenados nas unidades físicas de armazenamento. Estes meta-dados contém informação da diretoria e do espaço alocado entre outros dados;
  - ***Rewrite*** - Este teste mede a *performance* de escrever um ficheiro já existente. Quando um ficheiro existente é escrito, o trabalho necessário é menor uma vez que os "meta-dados"já existem. É normal a *performance* de rescrever um ficheiro já existente ser maior que a *performance* de escrever um ficheiro novo.

O output produzido pelo comando apresentado em cima foi:

```
Record Size 64 kB
File size set to 10240 kB
Command line used: /opt/csw/bin/iozone -i0 -r64k -s10m
Output is in kBytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 kBytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.

kB      reclen write   rewrite
10240      64     119047     119346
```

Como podemos verificar pelo *output* em cima, o *Record Size* é de 64 kB e o tamanho do ficheiro é de 10240 kB ou seja 10 Mb. Quanto aos resultados obtidos vemos que as operações de escrita num novo ficheiro (*write*) foram feitas a uma velocidade de 119047 kBytes/sec e as operações de escrita num ficheiro já existente (*Rewrite*) foram feitas a uma velocidade de 119346 kBytes/sec.

- ***Read* e *Reread*** - para este teste é usado a *flag* **-i1**. O comando completo pode ser consultado em baixo para um ficheiro de 10 Mb com um *Record Size* de 64 Kb:

```
/opt/csw/bin/iozone -i1 -r64k -s10m
```

  - ***Read*** - Este teste mede a *performance* de leitura de um ficheiro já existente;
  - ***Reread*** - Este teste mede a *performance* de leitura de um ficheiro que foi lido recentemente. É normal a *performance* ser maior uma vez que o sistema operativo mantém em cache os dados dos arquivos que foram lidos recentemente. Esta *cache* pode ser usada para satisfazer as leituras e melhorar a *performance*.

O output produzido pelo comando apresentado em cima foi:

```
Record Size 64 kB
File size set to 10240 kB
Command line used: /opt/csw/bin/iozone -i0 -i1 -r64k -s10m
Output is in kBytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 kBytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.

kB    reclen     write   rewrite     read     reread
10240      64     118673    119714    2637083    3101102
```

Como podemos verificar pelo *output* em cima, o *Record Size* é de 64 kB e o tamanho do ficheiro é de 10240 kB ou seja 10 Mb. Quanto aos resultados obtidos vemos que as operações de leitura de um ficheiro

já existente (*read*) foram feitas a uma velocidade de 2637083 kBytes/sec e as operações de escrita de um ficheiro lido recentemente (*Reread*) foram feitas a uma velocidade de 3101102 kBytes/sec. Como podemos verificar a velocidade das operações *Reread* é mais elevada, o que já era de se esperar, visto que o sistema operativo mantém em *cache* os dados de arquivos lidos recentemente o que melhora a performance.

5. Conclusão

Numa primeira abordagem a este trabalho, decidi explorar bem o *IOZone Benchmark*, realizando vários testes, com várias *flags*, por forma a perceber bem o funcionamento deste *Benchmark*. Para além da realização destes testes, ainda usei a ferramenta *truss* de forma a perceber qual o comportamento desta aplicação, isto é, que chamadas ao sistema utiliza, quantas vezes são feitas, quantas operações de escrita/leitura são feitas, entre outras análises.

Nesta fase do trabalho a maior dificuldade com que me deparei, foi essencialmente com a análise dos resultados resultantes do comando *truss*, uma vez que, quando executava o comando *truss*, para o *IOZone*, para um determinado tamanho de ficheiro, com um determinado *Record Size*, estava à espera de um resultado (no meu caso, um determinado numero de operações de escrita) e na verdade o resultado era um que não estava à espera, situação que para já não consigo perceber bem.

Como trabalho futuro e para a próxima fase, espero realizar todos os testes em cima descritos, bem como comprovar/comparar os resultados obtidos com o *IOZone Benchmark*, com *Scripts* em *D* que irão ser usadas com a *Framework Dtrace*.