Benchmarking dos Resultados Obtidos com *IOZone*, e 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

Benchmark [2] é o ato de executar um conjunto de testes padrão de forma a avaliar o desempenho quer de *Hardware* quer de *Software*. O *Benchmark* é muito usado na área de ciências de computação, sendo que o *Benchmark* se divide em ”Passivo”e ”Activo” [1]. O *Benchmark* ”Passivo”, diz respeito à execução de um conjunto de testes padrão, sendo que estes são ignorados até que terminem. O principal objetivo deste tipo de *Benchmark* é obter dados de referência de forma a poder comprovar/comparar com outros resultados. Com o *Benchmark* ”Activo”, testa-se a performance do *Benchmark* enquanto este corre, com isto podemos ver o que testa o *Benchmark* bem como perceber a maneira como o *Benchmark* se comporta, aqui os dados são informação.

Com a resolução deste trabalho, é suposto fazer *Benchmark* ”Activo”ao *Filesystem* (no meu caso) da máquina *Solaris 11*, recorrendo a diversos testes do *IOZone Benchmark* e à ferramenta *Dtrace*.

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. Análise dos Resultados

Nesta secção irei apresentar os resultados que obtive com a execução do *Benchmark IOZone* para diferentes testes, bem como o *output* produzido pelas *scripts* que desenvolvi em *D*. Os testes foram realizados por etapas, a primeira etapa testei as *Syscall's write* e *Rewrite*, na segunda etapa testei *Syscall's read* e *reread* **CONTINUAR**

5.1. *Write* e *Rewrite* - Testado na *Home*

Para testar a *Syscall write (rewrite)*, executei o seguinte comando:

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

sendo que o seu output foi o seguinte:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -i0 -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       4    179488     391934
```

A *script* em *D* por mim desenvolvida e que posteriormente foi executada com *Framework Dtrace* pode ser consultada em baixo:

```
#!/usr/sbin/dtrace -s

#pragma D option quiet

syscall::open*:entry
/execname=="iozone" & uid==29231/
{
        self->path = copyinstr(arg1);
}

syscall::open*:return
/self->path=="iozone.tmp"/
{
        total_time=0;
        size = 0;
        flag = 1;
}

syscall::write:entry
/flag == 1/
{
        self->start_time = timestamp;
        self->w_size = (arg2/1024);
        size = size + self->w_size;
}

syscall::write:return
/self->start_time > 0 & self->w_size > 0/
{
        self->stop_time = timestamp;
        self->elapsed = self->stop_time - self->start_time;
        total_time = total_time + self->elapsed;
}


syscall::close*:entry
/self->path=="iozone.tmp"/
{
        printf("%-12s %s\n","SIZE","ELAPSED TIME");
        printf("%-12d %d\n",size,total_time);
        flag = 0;
}
```

Depois de executada a *script* em cima apresentada, esta apresentou o seguinte *output*:

```
SIZE         ELAPSED TIME
0            0
SIZE         ELAPSED TIME
0            0
SIZE         ELAPSED TIME
10240        50946670
SIZE         ELAPSED TIME
10240        50946670
SIZE         ELAPSED TIME
10240        22982712
```

Ao analisarmos estes resultados podemos constatar que os valores se encontram num intervalo aceitável relativamente aos resultados do *Benchmark IOZone*. Ora vejamos, o tamanho (*SIZE*) encontra-se já em *Kbytes* e o tempo (*ELAPSED TIME*) encontra-se em nanosegundos, por isso necessito converter para segundos, para isso fiz $50946670 * 10^{-9} = 0.05094667$, por fim para se obter o resultado pretendido (*Kb/sec*) fiz: $10240/0.05094667 = 200994.49091 Kb/sec$. Como podemos verificar este valor encontra-se entre um intervalo aceitável ($179488 \leq 200994.49091 \leq 391934$).

Com isto podemos concluir que o *Benchmark IOZone* inicia a contagem do tempo sempre que executa a *syscall open* e termina sempre que executa a *syscall close*. A contagem dos *Bytes* escritos é feita sempre à entrada da *syscall write*.

De referir que este valor obtido através da *script* em *D* contém um erro relativo ao tamanho, uma vez que a medida que se aumenta o tamanho dos ficheiros, os valores resultantes apresentam valores com uma margem de erro maior.

5.2. *Write* e *Rewrite* - Testado na diretoria *diskHitachi*

Neste teste corri a mesma *script* apresentada em cima na directoria *diskHitachi*. Os resultados obtidos depois de ter corrido também o mesmo comando em cima (comando *iozone*) foram os seguintes:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -i0 -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       4   190320   430651
```

E os resultados da *script* em *D* foram os seguintes:

```
SIZE            ELAPSED TIME
0               0
SIZE            ELAPSED TIME
0               0
SIZE            ELAPSED TIME
10240           47833799
SIZE            ELAPSED TIME
10240           47833799
SIZE            ELAPSED TIME
10240           20875625
```

Ao analisarmos os dados da mesma maneira que analisei em cima, isto é fazendo os mesmos cálculos que efectuei em cima, obtive os seguintes resultados:

- Tempo: $47833799 * 10^{-9} = 0.047833799sec$
- Kb/sec: $10240/0.047833799 = 214074.571Kb/sec$

Como podemos verificar o valor resultante é um valor que se encontra dentro de um intervalo aceitável, comparado com os resultados do *Benchmark IOZone*, isto é $190320 \leq 214074.571 \leq 430651$.

Os discos SSD em termos de escrita têm um desempenho idêntico ou pior que um disco HDD, dai as operações de escrita do disco HDD (*diskHitachi*) serem maiores que no disco SSD (teste *Home*).

5.3. *Read* e *Reread* - Testado na *Home*

Para testar a *Syscall read (reread)*, executei o seguinte comando:

```
/opt/csw/bin/iozone -i0 -i1 -s10m
```

sendo que o seu output foi o seguinte:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -i0 -i1 -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       4   428200   547270  1223574  1247709
```

Quanto a *script* em *D* por mim desenvolvida, para o traçado da *syscall read*, que posteriormente foi executada com *Framework Dtrace* pode ser consultada em baixo:

```
#!/usr/sbin/dtrace -s

#pragma D option quiet

syscall::open*:entry
/execname=="iozone" & uid==29231/
{
        self->path = copyinstr(arg1);
        self->flag = arg2;
}

syscall::open*:return
/self->path=="iozone.tmp" /
{
        self->start_time = timestamp;
        total_time=0;
        size = 0;
        flag = 1;
        total_time_io = 0;
}

syscall::read:return
/self->start_time> 0/
{
        self->r_size = (arg0/1024);
        size = size + self->r_size;
}

syscall::close*:entry
/self->path=="iozone.tmp" /
{
        self->stop_time = timestamp;
        self->elapsed = self->stop_time - self->start_time;
        total_time = total_time + self->elapsed;
        printf("%-12s %s\n","SIZE","ELAPSED TIME");
        printf("%-12d %d\n",size,total_time);
        flag = 0;
}
```

Sendo que o *output* por ela calculado foi o seguinte:

```
SIZE            ELAPSED TIME
0               33698
SIZE            ELAPSED TIME
0               2023
SIZE            ELAPSED TIME
0               67259606
SIZE            ELAPSED TIME
0               134527857
SIZE            ELAPSED TIME
0               53422605
SIZE            ELAPSED TIME
10244           8397888
SIZE            ELAPSED TIME
10244           8219413
```

Ao analisarmos estes resultados podemos constatar que os valores aproximam dos resultados do *Benchmark IOZone*. Ora vejamos, o tamanho (*SIZE*) encontra-se já em *Kbytes* e o tempo (*ELAPSED TIME*) encontra-se em nanosegundos, por isso necessito converter para segundos, para isso fiz $8397888 * 10^{-9} = 0.008397888sec$, por fim para se obter o resultado pretendido (*Kb/sec*) fiz: $10244/0.008397888 = 1219830.5098Kb/sec$. Como podemos verificar este valor encontra-se muito proximo do valor obtido com o *Benchmark IOZone* ($1219830.5098 \simeq 1223574$).

Com isto podemos concluir que o *Benchmark IOZone* inicia a contagem do tempo sempre que executa a *syscall open* e termina sempre que executa a *syscall close*. A contagem dos *Bytes* lidos é feita sempre à saída (*return*) da *syscall read*.

5.4. *Read* e *Reread* - Testado na diretoria *diskHitachi*

Neste teste corri a mesma *script* apresentada em cima na directoria *diskHitachi*. Os resultados obtidos depois de ter corrido também o mesmo comando em cima (comando *iozone*) foram os seguintes:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -i0 -i1 -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       4   158479   250641   937995  1030491
```

E os resultados da *script* em *D* foram os seguintes:

```
SIZE          ELAPSED TIME
0             52608
SIZE          ELAPSED TIME
0             4387
SIZE          ELAPSED TIME
0             170568505
SIZE          ELAPSED TIME
0             341150014
SIZE          ELAPSED TIME
0             152117012
SIZE          ELAPSED TIME
10244         11102378
SIZE          ELAPSED TIME
10244         10081823
```

Ao analisarmos os dados da mesma maneira que analisei em cima, isto é fazendo os mesmos cálculos que efectuei em cima, obtive os seguintes resultados:

- Tempo: $11102378 * 10^{-9} = 0.011102378sec$
- Kb/sec: $10240/0.011102378 = 922685.21212Kb/sec$

Como podemos verificar o valor calculado encontra-se muito proximo do valor obtido com o *Benchmark IOZone* ($922685.21212 \simeq 937995$).

No que toca a operações de leitura, os discos SSD apresentam um melhor desempenho, relativamente aos discos HDD, isto explica o porque dos resultados de leitura no disco SSD (teste *Home*) serem maiores que os do disco HDD (*diskHitachi*). Ora vejamos $1223574 > 937995$.

6. 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.

Na segunda parte deste trabalho desenvolvi *scripts* em *D*, que depois de usadas com a *Framework Dtrace*, produziram resultados semelhantes aos resultados obtidos com o *Benchmark IOZone*. Depois de obtidos esses resultados comparei os dois e pude concluir que, estes são semelhantes. Nesta fase a maior dificuldade com que me deparei foi em perceber como iria fazer as provas na *script D* de forma a obter os resultados que desejava, para ultrapassar essa dificuldade tive de analisar melhor o *output* do comando *truss*, por forma a perceber ainda melhor o comportamento do *Benchmark*.

Como trabalho futuro, gostava de melhorar as *scripts* que desenvolvi de maneira a conseguir obter resultados mais precisos. Também gostava de desenvolver as *scripts* que me propus a desenvolver e estão em falta.

Referências

[1] Active benchmarking. http://www.brendangregg.com/activebenchmarking.html.

[2] Benchmark (computação). https://pt.wikipedia.org/wiki/Benchmark.