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

Conteúdo

***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* [3], 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* e na segunda etapa testei *Syscall's read* e *reread*.

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

Para testar a *Syscall write (rewrite)* na *home*, usa um sistema de ficheiros *ZFS*, 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*

Para testar o *Benchmark IOZone* no disco HDD (*diskHitachi*) tive de correr o seguinte comando, de referir que tive de usar a *flag -f* para redirecionar o ficheiro temporário para o disco referido.

```
/opt/csw/bin/iozone -f /diskHitachi/a57779/iozone.tmp -i0 -s10m
```

o qual teve o seguinte *output*:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -f /diskHitachi/a57779/iozone.tmp -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   126864   340835
```

Em simultâneo foi executada a seguinte *Script* com o dtrace, de notar também que no predicado em que verifico o *path* (*self->path*) também tive de por a diretoria do disco em causa.

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

#pragma D option quiet

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

syscall::open*:return
/self->path=="/diskHitachi/a57779/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=="/diskHitachi/a57779/iozone.tmp"/
{
        printf("%-12s %s\n","SIZE","ELAPSED TIME");
        printf("%-12d %d\n",size,total_time);
        flag = 0;
}
```

O *Output* da execução da *script* com a *Framework Dtrace* foi o seguinte:

```
SIZE         ELAPSED TIME
0              0
SIZE         ELAPSED TIME
0              0
SIZE         ELAPSED TIME
10240        74204464
SIZE         ELAPSED TIME
10240        74204464
SIZE         ELAPSED TIME
10240        26921551
```

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: $74204464 * 10^{-9} = 0.074204464 sec$
- Kb/sec: $10240/0.047833799 = 137997.08869 Kb/sec$

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 $126864 \leq 137997.08869 \leq 340835$.

Os discos SSD em termos de escrita têm um desempenho idêntico a um disco HDD, dai as operações de escrita do disco HDD (*diskHitachi*) serem semelhantes às do 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*

Para testar o *Benchmark IOZone* no disco HDD (*diskHitachi*) tive de correr o seguinte comando, de referir que tive de usar a *flag -f* para redirecionar o ficheiro temporário para o disco referido.

```
/opt/csw/bin/iozone -f /diskHitachi/a57779/iozone.tmp -i0 -i1 -s10m
```

O *output* da execução deste comando foi o seguinte:

```
        File size set to 10240 kB
        Command line used: /opt/csw/bin/iozone -f /diskHitachi/a57779/iozone.tmp -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   117760   603490  1356985  1521537
```

Em simultâneo foi executado a seguinte *script D*, de maneira a fazer o traçado dinâmico do comando em cima especificado. De referir mais uma vez que no predicado em que verifico o *path* (*self->path*) também tive de por a diretoria do disco em causa.

```
#!/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=="/diskHitachi/a57779/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=="/diskHitachi/a57779/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 seu *output* foi o seguinte:

```
SIZE        ELAPSED TIME
0           59568
SIZE        ELAPSED TIME
0           6010
SIZE        ELAPSED TIME
0           452529670
SIZE        ELAPSED TIME
0           905072189
SIZE        ELAPSED TIME
0           167244259
SIZE        ELAPSED TIME
10244       7771309
SIZE        ELAPSED TIME
10244       6939257
```

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: $7771309 * 10^{-9} = 0.007771309sec$
- Kb/sec: $10240/0.007771309 = 1318182.0463Kb/sec$

Como podemos verificar o valor calculado encontra-se muito proximo do valor obtido com o *Benchmark IOZone* ($1318182.0463 \simeq 1356985$).

5.5. Tabela com todos os Resultados Obtidos

Apresento na tabela 1 todos os resultados obtidos para todos os testes realizados neste trabalho, quer para o *Benchmark IOZone* quer para a *Ferramenta Dtrace*.

		Write	Rewrite	Read	Reread
SSD (Teste Home (ZFS))	IOZone	179488 Kb/sec	391934 Kb/sec	1223574 Kb/sec	1247709 Kb/sec
	Dtrace	200994.49091 Kb/sec		1219830.5098 Kb/sec	
HDD (Teste /diskHitachi (UFS))	IOZone	126864 Kb/sec	340835 Kb/sec	1356985 Kb/sec	1521537 Kb/sec
	Dtrace	137997.08869 Kb/sec		1318182.0463 Kb/sec	

Tabela 1. RESULTADOS OBTIDOS PARA IOZONE, DTRACE PARA DISCOS SSD E HDD PARA AS CHAMADAS AO SISTEMA *WRITE (REWRITE), READ (REREAD)*

Ao analisar a tabela 1 posso concluir que os resultados obtidos para um disco SSD (teste *Home*), para um sistema de ficheiros ZFS, para operações de escrita (*write (rewrite)*) são resultados, ligeiramente mais altos do que aqueles que foram obtidos para um disco HDD (teste */diskHitachi*), com um sistema de ficheiros UFS.

Em termos de operações de leitura (*read (reread)*) tanto para o disco SSD como para o disco HDD, os resultados são semelhantes, sendo que para o disco HDD os resultados são um pouco mais altos, resultados estes que vão um pouco contra o que esperava, uma vez que os discos SSD tem um melhor desempenho do que os discos HDD, como tal tenho algumas dificuldades em explicar este facto, uma vez que isto acontece tanto na execução do *Benchmark IOZone* como com as *scripts Dtrace*. De referir que na operação de re-leitura (*Rewrite*), os resultados apresentados apresentam um melhor desempenho relativamente à operação de leitura, isto deve-se ao facto de o Sistema Operativos manter em cache dados de ficheiros que foram lidos recentemente, como tal o acesso a esses dados é mais rápido.

Nas operações de re-escrita, acontece exatamente o mesmo, isto é, esta apresenta um melhor desempenho comparativamente à operação de escrita, isto porque, a primeira vez que se escreve num ficheiro, para além dos dados também são escritos meta-dados, o que não acontece na operação de re-escrita, uma vez que não é necessário escrever os meta-dados de novo.

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 algumas *scripts*, que na minha opinião, também seriam relevantes para este trabalho.

Referências

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

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

[3] W. D. Norcott. Iozone filesystem benchmark.