ZFS for NGS data analysis

saving space from the galactic expansion

Davide Cittaro - Cogentech (Milan, Italy)
Galaxy DevCon 2010 - CHSL NY
Motivation
Motivation

• Deploy Galaxy to serve a small NGS facility
  • Data delivery
  • End-users can perform their own analysis
Motivation

- Deploy Galaxy to serve a small NGS facility
- Data delivery
- End-users can perform their own analysis
- Reliable and efficient data storage
Motivation

• Deploy Galaxy to serve a small NGS facility
  • Data delivery
  • End-users can perform their own analysis
• Reliable and efficient data storage
• Overcome some Galaxy limitations
  • Loose control on files fate
  • Overcrowding of disks
NGS = IT nightmare?
NGS = IT nightmare?

- Data size
  - Disk occupancy
  - Data transfer
- Backup
NGS = IT nightmare?

• Data size
  • Disk occupancy
  • Data transfer
  • Backup

• Algorithm performance
  • Resource consumption (RAM, threads)
  • Time
Resource consumption

Rough time to align 1M reads

* 4 threads enabled
‡ search on 10 indexes
Resource consumption

Rough time to align 1M reads

- blat: 1,974 minutes
- eland: 77 minutes
- bwa*: 4 minutes
- bfast*: 114 minutes

* 4 threads enabled
‡ search on 10 indexes
Resource consumption

RAM usage to align 1M reads

```
20
15
10
5
```

- blat
- eland
- bwa
- bfast

RAM (Gb)
Resource consumption

RAM usage to align 1M reads

- blat: 4 GB
- eland: 1 GB
- bwa: 2 GB
- bfast: 16 GB
Data size

Disk occupancy by data type

<table>
<thead>
<tr>
<th>File Size (Mb)</th>
<th>fastq</th>
<th>BAM</th>
<th>bedgraph</th>
<th>bigWig</th>
<th>bed/gff</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,125</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>750</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>375</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data size

Disk occupancy by data type

- fastq: 1,000 Mb
- BAM: 500 Mb
- bedgraph: 1,200 Mb
- bigWig: 400 Mb
- bed/gff: 2 Mb
ZFS
ZFS

• “The last word in File Systems”
ZFS

- “The last word in File Systems”
- Many features for data storage and handling
  - Block-level deduplication
  - Block-level compression
  - Self-healing capabilities
ZFS

• “The last word in File Systems”
• Many features for data storage and handling
  • Block-level deduplication
  • Block-level compression
  • Self-healing capabilities
• Not (yet) network/parallel filesystem
ZFS

- “The last word in File Systems”
- Many features for data storage and handling
  - Block-level deduplication
  - Block-level compression
  - Self-healing capabilities
- Not (yet) network/parallel filesystem
- It’s free
# ZFS block level compression

<table>
<thead>
<tr>
<th>File type</th>
<th>Compress ratio</th>
<th>Useful</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>edland genome index</td>
<td>1x</td>
<td>✘</td>
<td>These are little anyway</td>
</tr>
<tr>
<td>BWT genome index</td>
<td>1x</td>
<td>✘</td>
<td>bowtie and bwa</td>
</tr>
<tr>
<td>bfast genome index</td>
<td>1x</td>
<td>✘</td>
<td></td>
</tr>
<tr>
<td>BAM</td>
<td>1x</td>
<td>✘</td>
<td>Already compressed</td>
</tr>
<tr>
<td>Illumina runs</td>
<td>1.3x</td>
<td>✔</td>
<td>w/out images and GERALD results</td>
</tr>
<tr>
<td>Text files</td>
<td>1.8x - 3x</td>
<td>✔</td>
<td>Most of the data</td>
</tr>
<tr>
<td>bigWig/bigBed</td>
<td>1.5x</td>
<td>✔</td>
<td>Best way to deal with UCSC</td>
</tr>
</tbody>
</table>
## ZFS block level compression

<table>
<thead>
<tr>
<th>File type</th>
<th>Compress ratio</th>
<th>Useful</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>eland genome index</td>
<td>1x</td>
<td>✗</td>
<td>These are little anyway</td>
</tr>
<tr>
<td>BWT genome index</td>
<td>1x</td>
<td>✗</td>
<td>bowtie and bwa</td>
</tr>
<tr>
<td>bfast genome index</td>
<td>1x</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>BAM</td>
<td>1x</td>
<td>✗</td>
<td>Already compressed</td>
</tr>
</tbody>
</table>
# ZFS block level compression

<table>
<thead>
<tr>
<th>File type</th>
<th>Compress ratio</th>
<th>Useful</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>eland genome index</td>
<td>1x</td>
<td>✗</td>
<td>These are little anyway</td>
</tr>
<tr>
<td>BWT genome index</td>
<td>1x</td>
<td>✗</td>
<td>bowtie and bwa</td>
</tr>
<tr>
<td>bfast genome index</td>
<td>1x</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>BAM</td>
<td>1x</td>
<td>✗</td>
<td>Already compressed</td>
</tr>
<tr>
<td>Illumina runs</td>
<td>1.3x</td>
<td>✓</td>
<td>w/out images and GERALD results</td>
</tr>
<tr>
<td>Text files</td>
<td>1.8x - 3x</td>
<td>✓</td>
<td>Most of the data</td>
</tr>
<tr>
<td>bigWig/bigBed</td>
<td>1.5x</td>
<td>✓</td>
<td>Best way to deal with UCSC</td>
</tr>
</tbody>
</table>
Why should I care?
Why should I care?

CLI

fastq.gz → bwa → samtools → BAM
Why should I care?

**CLI**

```
fastq.gz → bwa → samtools → BAM
```

**Galaxy (default)**

```
fastq → bwa → SAM → samtools → BAM
```
Why should I care?

CLI

fastq.gz → bwa → samtools → BAM

925 Mb

Galaxy (default)

fastq → bwa → SAM → samtools → BAM
Why should I care?

CLI

fastq.gz → bwa → samtools → BAM

925 Mb

Galaxy (default)

fastq → bwa → SAM → samtools → BAM

4.1 Gb

Sunday, May 16, 2010
Why should I care?

CLI

- fastq.gz → bwa → samtools → BAM
- 925 Mb

Galaxy (default)

- fastq → bwa → SAM → samtools → BAM
- 2.3 Gb
ZFS file deduplication
ZFS file deduplication

- Introduced with ZFS v21 (OpenSolaris b128)
  - block-level deduplication (general purpose)
  - synchronous (requires high threaded OS)
  - SHA256 hashing algorithm
ZFS file deduplication

- Introduced with ZFS v21 (OpenSolaris b128)
- block-level deduplication (general purpose)
- synchronous (requires high threaded OS)
- SHA256 hashing algorithm
- Supported by OpenSolaris only (and derivatives such as Nexenta Core 3)
Why should I care?
Why should I care?

fastq
Why should I care?

fastq → A → BAM → bedgraph
Why should I care?

fastq

BAM

B

A

bedgraph

bedgraph
Why should I care?

fastq

BAM

bedgraph

Sunday, May 16, 2010
Our galactic experience
Our galactic experience

- Installed to provide CARPET (Collection of Automated Routine Programs for Easy Tiling)
Our galactic experience

• Installed to provide CARPET (Collection of Automated Routine Programs for Easy Tiling)

• Not high performance server
Our galactic experience

- Installed to provide CARPET (Collection of Automated Routine Programs for Easy Tiling)
- Not high performance server
- Few users worldwide (< 20)
Galaxy @ IFOM-IEO Campus

Galaxy is for Biologists
Use this site to access popular sources of data like the UCSC Table Browser. Run analyses right on the spot using a variety of integrated tools. Your results are always available and can be easily shared with others. Just watch how.

Custom features for this Galaxy installation

Galaxy team is a part of BX at Penn State.

This project is supported in part by NSF and the Huck Institutes of the Life Sciences.

Galaxy build: 1349
maintained by Davide Gittaro, Cogentech c/o IFOM-IEO Campus
Our galactic experience
Our galactic experience

- 20% files are duplicated
Our galactic experience

• 20% files are duplicated
• 27% overhead in disk space usage
Our galactic experience

- 20% files are duplicated
- 27% overhead in disk space usage
- Not really high throughput data (i.e. smaller sizes)
ZFS Summary

- Provides efficient way to store terabytes
ZFS Summary

• Provides efficient way to store terabytes
• NGS pipelines can take advantage of ZFS properties
ZFS Summary

• Provides efficient way to store terabytes
• NGS pipelines can take advantage of ZFS properties
• As galaxy gives loose control on files, ZFS will handle most of the issues, transparently
ZFS Summary

• Provides efficient way to store terabytes

• NGS pipelines can take advantage of ZFS properties

• As galaxy gives loose control on files, ZFS will handle most of the issues, transparently

• Native support only on FreeBSD and [Open] Solaris (and derivatives)
Porting to UNIX
Porting to UNIX

- Not going to benchmarks, ZFS alone is worth it
Porting to UNIX

- Not going to benchmarks, ZFS alone is worth it
- Porting may be a nightmare
  - GNU/Linux is the most used OS
  - Some libraries/functions are taken for granted
Porting to UNIX

• Not going to benchmarks, ZFS alone is worth it
• Porting may be a nightmare
  • GNU/Linux is the most used OS
  • Some libraries/functions are taken for granted
• You’ll need gdb and vim
Porting to UNIX

- Not going to benchmarks, ZFS alone is worth it
- Porting may be a nightmare
  - GNU/Linux is the most used OS
  - Some libraries/functions are taken for granted
- You’ll need gdb and vim
- Get familiar with BSD userland
Porting to UNIX

- Not going to benchmarks, ZFS alone is worth it
- Porting may be a nightmare
  - GNU/Linux is the most used OS
  - Some libraries/functions are taken for granted
- You’ll need `gdb` and `vim`
- Get familiar with BSD userland
- Galaxy works
Things are there...
Things are there...

- Many includes may be taken for granted
  - type definitions
  - low level definitions (sys)
  - ...

Sunday, May 16, 2010
Things are there...

```bash
~/bwa-0.5.7 gmake
...
gcc -c -g -Wall -O2 -m64 -DHAVE_PTHREAD  bwtgap.c -o bwtgap.o
In file included from bwtgap.c:4:
bwtgap.h:8: error: expected specifier-qualifier-list before 'u_int32_t'
bwtgap.c: In function 'gap_push':
bwtgap.c:58: error: 'gap_entry_t' has no member named 'info'
bwtgap.c:58: error: 'u_int32_t' undeclared (first use in this function)
bwtgap.c:58: error: (Each undeclared identifier is reported only once)
bwtgap.c:58: error: for each function it appears in.)
bwtgap.c:58: error: expected ';' before 'score'
...
```
Things are there...

diff -Naur bwa-0.5.7/bwt.h bwa-0.5.7.fbsd/bwt.h
--- bwa-0.5.7/bwt.h 2010-03-01 16:36:39.000000000 +0100
+++ bwa-0.5.7.fbsd/bwt.h 2010-04-26 16:32:45.452492363 +0200
@@ -29,6 +29,7 @@
#define BWA_BWT_H

#include <stdint.h>
+#include <unistd.h>

// requirement: (OCC_INTERVAL%16 == 0)
#define OCC_INTERVAL 0x80
diff -Naur bwa-0.5.7/bwt_lite.h bwa-0.5.7.fbsd/bwt_lite.h
--- bwa-0.5.7/bwt Lite.h 2010-03-01 16:36:39.000000000 +0100
+++ bwa-0.5.7.fbsd/bwt Lite.h 2010-04-26 16:33:17.42072963 +0200
@@ -2,6 +2,7 @@
#define BWT_LITE_H_

#include <stdint.h>
+#include <unistd.h>

typedef struct {
    uint32_t seq_len, bwt_size, n_occ;
}
… but may be broken…
... but may be broken...

- Tried to recycle Illumina IPAR for galaxy
... but may be broken...

- Tried to recycle Illumina IPAR for galaxy
- Configured the MSA70 to present 25 disks
  - ZFS performance (load balance)
  - RAID-Z data consistency vs. HP P800
... but may be broken...

- Tried to recycle Illumina IPAR for galaxy
- Configured the MSA70 to present 25 disks
  - ZFS performance (load balance)
  - RAID-Z data consistency vs. HP P800
- CISS driver for FreeBSD allows up to 16 disks
… but may be broken…

- Tried to recycle Illumina IPAR for galaxy
- Configured the MSA70 to present 25 disks
  - ZFS performance (load balance)
  - RAID-Z data consistency vs. HP P800
- CISS driver for FreeBSD allows up to 16 disks
- Infinite kernel panic at startup
... but may be broken...

diff -u cissvar.h*
--- cissvar.h   2010-04-26 17:30:29.698264382 +0200
+++ cissvar.h.new       2010-04-26 17:29:59.408384802 +0200
@@ -46,7 +46,7 @@
 /*
  * Maximum number of logical drives we support.
  */
-#define CISS_MAX_LOGICAL       16
+#define CISS_MAX_LOGICAL       32

/*
 * Maximum number of physical devices we support.

... or even missing!
... or even missing!

- samtools’ SNP caller uses `logl()` and `expl()` functions
- `long double` type is architecture dependent
- 64, 80 or 128 bit precision
- not implemented in FreeBSD
... or even missing!

• samtools’ SNP caller uses `logl()` and `expl()` functions

• long double type is architecture dependent

• 64, 80 or 128 bit precision

• not implemented in FreeBSD

• `libstdc++` wrapper?
... or even missing!

• samtools’ SNP caller uses logl() and expl() functions

• long double type is architecture dependent

• 64, 80 or 128 bit precision

• not implemented in FreeBSD

• libstdc++ wrapper?

• MPFR variant?
And also…
And also...

- Illumina Pipeline
  - need to patch for gsed and gmake
And also...

- Illumina Pipeline
  - need to patch for gsed and gmake
- rpy
  - issues with the linker (SEGFAULT!)
And also...

- Illumina Pipeline
  - need to patch for gsed and gmake
- rpy
  - issues with the linker (SEGFAULT!)
- UCSC Genome Browser
  - once found hgTracks issue with qsort()
And also...

- Illumina Pipeline
  - need to patch for gsed and gmake
- rpy
  - issues with the linker (SEGFAULT!)
- UCSC Genome Browser
  - once found hgTracks issue with qsort()
- …
Conclusions
Conclusions

- ZFS is a cost-effective and efficient solution to handle NGS data
Conclusions

- ZFS is a cost-effective and efficient solution to handle NGS data
- It transparently solves many issues which may come with Galaxy
Conclusions

• ZFS is a cost-effective and efficient solution to handle NGS data

• It transparently solves many issues which may come with Galaxy

• ZFS requires alternative OS (although you may try ZFS+FUSE+GNU/Linux)
Conclusions

• ZFS is a cost-effective and efficient solution to handle NGS data

• It transparently solves many issues which may come with Galaxy

• ZFS requires alternative OS (although you may try ZFS+FUSE+GNU/Linux)

• Porting bioinformatics to UNIX may be tricky
Conclusions

• ZFS is a cost-effective and efficient solution to handle NGS data

• It transparently solves many issues which may come with Galaxy

• ZFS requires alternative OS (although you may try ZFS+FUSE+GNU/Linux)

• Porting bioinformatics to UNIX may be tricky

• If one can’t port all applications to UNIX, at least can deploy a ZFS-based file server and export via NFS or iSCSI (or pNFS)
Acknowledgements

<table>
<thead>
<tr>
<th>NGS Facility @ Cogentech</th>
<th>Bioinfo people @ IFOM IEO Campus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Myriam Alcalay</td>
<td>Matteo Cesaroni</td>
</tr>
<tr>
<td>Simone Minardi</td>
<td>Lucilla Luzi</td>
</tr>
<tr>
<td>Gabriele Bucci</td>
<td></td>
</tr>
</tbody>
</table>

Sunday, May 16, 2010