First Results of Storing Genomics Data in ROOT

Jose Javier Gonzalez Ortiz
Supervisor: Fons Rademakers

August 11, 2017

CERN Openlab Summer Student Program 2017
Understanding genomic data not trivial at all. Ever tried reading binary code?

Credit: Randall Munroe https://xkcd.com/1605/
Sequence Alignment/Map

Text-based format for storing biological sequences aligned to a reference one.

- SAM - Plaintext
- BAM - Compressed

Challenge: Data size is extremely large (100GBs-TBs for a single genome)

Genomic data size will overtake LHC data in the coming years
ROOT Framework

Don’t reinvent the wheel

Most of the challenges with massive data processing are common to **HEP**. **ROOT** has decades of experience in software design and optimization.
### Conversion - SAM to ROOT

We convert from TAB-delimited format to a ROOT `TTree` with 14 columns.

<table>
<thead>
<tr>
<th>chr</th>
<th>start</th>
<th>length</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
<th>N</th>
<th>Q</th>
<th>R</th>
<th>S</th>
<th>V</th>
<th>Y</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

ROOT handles compression, memory buffers, datatypes, etc.

The columns are defined using a C++ class.
Performance - SAM to ROOT

There is a trade-off between compression and read/write speed for a 100GB file.

- **Speed comparison**
  - **BAM**
  - **ROOT:LZMA**
  - **ROOT:ZLIB**

- **Filesize comparison**
  - **BAM**
  - **ROOT:LZMA**
  - **ROOT:ZLIB**

With ZLIB compression → 4 times faster
With LZMA compression conversion → 15% smaller file
• We also need to be able to view compressed information as fast as possible.
• ROOT columnar structure allows us to just look at chromosome and position columns to optimize speed.
Indexing

Improve read speed by using an Index

- For random access BAM format needs a BAI index.
- ROOT has its own TTreeIndex.
- Only works on int and chromosome are char[].
- We hash the chromosome and store it separately.
- With compression, size increase is \( \approx \frac{1\text{MB}}{10\text{GB}} = 0.01\% \).
ROOT offers many parameters we need to check and profile:

- **Index**: Enabled vs Disabled
- **Compression Algorithm**: ZLIB vs LZMA
- **Split**: Column vs Row
- **Cache**: Enabled vs Disabled
- **Storage**: Local vs Remote

For 20 views, the number of tests per file gives a **Combinatorial Explosion**

\[2 \times 2 \times 2 \times 2 \times 2 \times 20 = 640 \text{ views per file}\]
Launch process automation and parsing is done in Python.

- User and System Time
- I/O operations
- Output logging
- Performance parsing
Multivariate ANOVA was used to determine which parameters are statistically significant to the read speed and by how much.

Over 15 times faster after changing parameters
Future Work

• Debug indexing problem for 1 billion entries file
• Improve read speed by analyzing I/O operations
• Implement & profile other cmds like sort or merge
• Provide API for use in third party scripts
Thank you!