Using ROOT in the field of Genome Sequencing

GSoC Project Introduction

Presented by Aditya Pandey
21 May 2025

About Me

Education

Birla Institute of Technology, Mesra (2022-Present)

Bachelor of Technology

Experience

Google Summer of Code 2024 contributor

INCF, 2024

- Developed interactive Python tools for neuronal morphology conversion
- Implemented robust error handling and validation mechanisms
- Created simulation-ready models for neuroscience research

Technical Skills

Programming Languages

- Proficient: Python, C, C++
- Familiar: Go, JavaScript, Rust

Framework & Libraries

- Web Development: Django, React.js, Node.js
- Scientific: ROOT Framework, htslib
- Data Analysis: Numpy, Pandas

Current Challenges in Genomic Data

- Genomic sequencing data is growing exponentially
- Current genomic datasets typically range from 10-30GB per sample
- Standard formats (BAM/CRAM) face performance bottlenecks
- Researchers experience long processing times for common queries
- Traditional formats prioritize compression over query performance

Project Overview

Enhancing Genomic Data Management with ROOT

- ROOT: Advanced data analysis framework developed at CERN
- Previous GeneROOT work implemented ROOT Alignment Maps (RAM) using TTree format, showing 4× performance gains
- This project will implement ROOT's next-generation RNTuple format for genomics
- Evolving from TTree-based RAM to RNTuple-based RAM for improved performance

Key RNTuple Features:

- Improved memory mapping for efficient data access
- Type safety through templated interfaces
- Enhanced parallelization capabilities
- Columnar data structure ideal for genomic queries

Project Goals

Advancing Genomic Data Management

Main Objectives:

- Validate previous GeneROOT benchmark results with TTree
- Implement genomic data model using RNTuple's templated fields
- Develop conversion utilities between BAM/CRAM and RNTuple
- Implement intelligent file splitting strategies
- Create efficient query tools for genomic region retrieval
- Compare performance with established formats
- Compare performance between BAM/CRAM and both TTree and RNTuple-based RAM formats

Technical Approach

Why RNTuple for Genomic Data?

Current Format Limitations:

- BAM: Binary format with moderate compression
- CRAM: Reference-based compression (30-60% smaller than BAM)
- Both prioritize storage efficiency over query performance
- Limited support for random access and selective data decoding

Our Solution

  - Implement RNTuple-based RAM format, store genomic data in efficient columnar structure
  - Only load needed fields (position, chromosome) during filtering
  - Optimize for genomic queries, improve search algorithms for faster region access
  - Organize data by chromosome for better performance
  - Enhance efficiency, reduce memory usage through optimized storage
  - Implement batch processing for large datasets
  - Ensure reliability, use type-safe interfaces to prevent errors
  - Create comprehensive conversion tools for existing formats

Project Timeline

1 → 2 → 3 → 4 → 5

COMPLETED FOUNDATION

- ✓ Environment setup with ROOT & htslib
- ✓ Test datasets acquisition (9GB/15GB BAM files)
- ✓ Initial ramview optimization
- ✓ Preliminary compression testing

OPTIMIZATION STRATEGIES

- Comprehensive compression strategy analysis
- Memory efficiency improvements
- Complete TTree-based RAM benchmarking

RNTUPLE MODEL DEVELOPMENT

- Design genomic data model for RNTuple
- Implement type-safe field interfaces
- Develop memory mapping architecture
- Create conversion prototypes

ADVANCED IMPLEMENTATION

- Develop BAM/CRAM to RNTuple converters
- Implement chromosome-based splitting
- Implement region-based organization
- Create read group separation logic

VALIDATION & DELIVERY

- Build efficient query tools for genomic regions
- Comprehensive performance benchmarking
- Comparative analysis documentation
- Final code delivery and reporting

Expected Outcomes

Performance Goals:

- Faster access to genomic regions with RNTuple-based RAM
- Reduced memory requirements for large datasets
- Efficient data compression without sacrificing speed
- Type-safe interfaces preventing common errors

Scientific Impact:

For Researchers

- Accelerate genomic analysis workflows from hours to minutes
- Enable interactive exploration of large-scale genomic datasets
- Facilitate more complex queries across larger cohorts
- Reduce infrastructure costs for genomics research

Thank you!