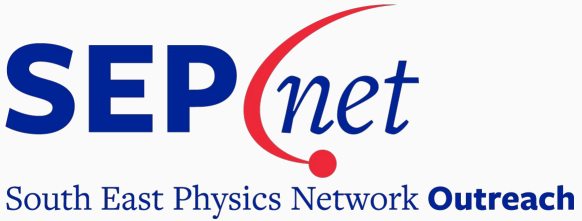
Memory allocators and LRT efficiency

Rosie Hasan
Supervisor: Dr Stewart Martin-Haugh

Science and Technology Facilities Council

Rutherford Appleton Laboratory



Introduction

1. Memory allocator performance tests
    - Triggering
    - Forks and threads
    - Memory allocators
2. Efficiency of the large radius tracker
    - Tracks
    - LRT
    - Efficiency

Memory allocator performance tests

Triggering

Selecting data that is more relevant as there is not enough storage for everything produced at ATLAS

L1: hardware trigger, keeps 1 in 400 events. E.g. total energy in calorimeter or threshold momentums

HLT: softwear, many algorithms working to find event signatures, keeps about 1 in 60 events. Fast algorithms first then almost full reconstruction. 40,000 CPUs available

The events selected by the 2 triggers are then moved into mass storage

Focus on speed to prevent data loss

Threads and forks

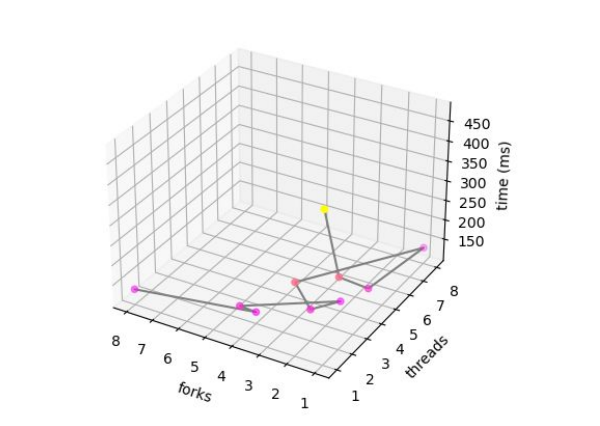
Splitting up the tasks to be completed

Reduce overall time and memory

Threads vs forks:

- Forks: new process, child process identical to mother process
- Threads: subtasks, can share memory
- Forks run independently, run their own threads

1 event per thread, so number of events processed at same time is forks*threads

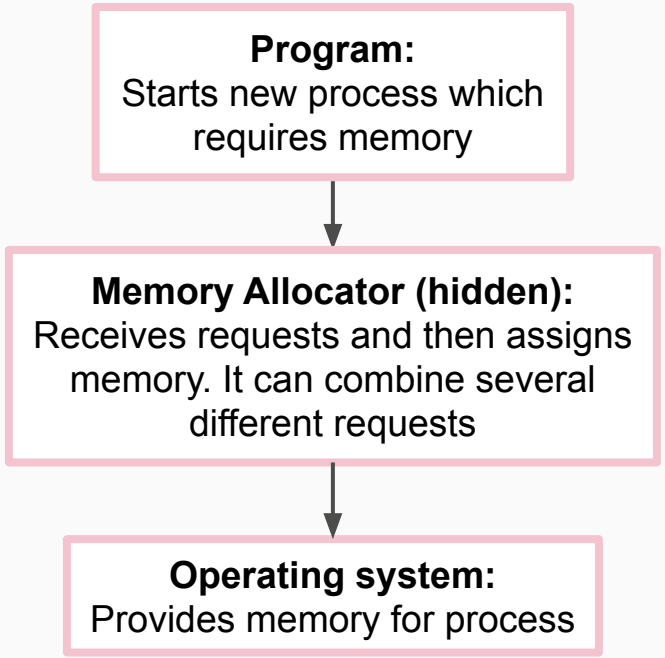


Memory allocators

Memory allocators optimize the assigning and deleting of memory required by processes.

The 4 memory allocators I am looking at:

- glibcmalloc/stdcmalloc- default on linux
- tcmalloc- athena default, made by google
- jemalloc- used by firefox
- mimalloc - used by Bing, made by microsoft



Method

Running the Atlas high level triggering system on a sample of 100 events using ./test_trigP1_v1Dev_build.py, working exclusively on pc-tbed-pub-22

Altering this to include input on the number of threads, forks, concurrent events and memory allocators. Combinations of forks and threads up to 8 events always keeping threads = slots

Using Rafal's script ./extract-times.sh to find time spent processing events

Studying the pss and throughput from the output of each run:

- Pss : shows how much memory is being used
- Throughput: events per second, the amount of time processing an event can vary, some longer than others and initialisation time

Throughput- allocators

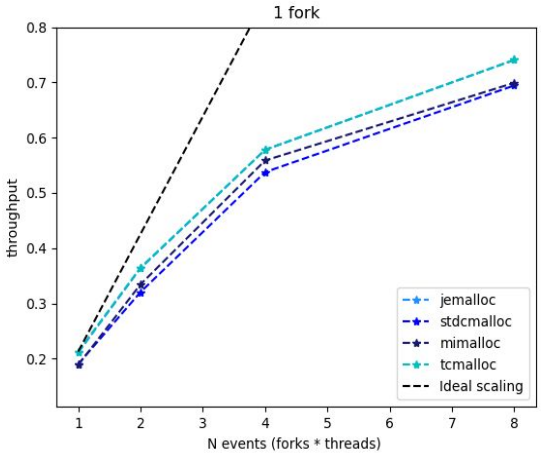
Throughput increases with the number of events as expected

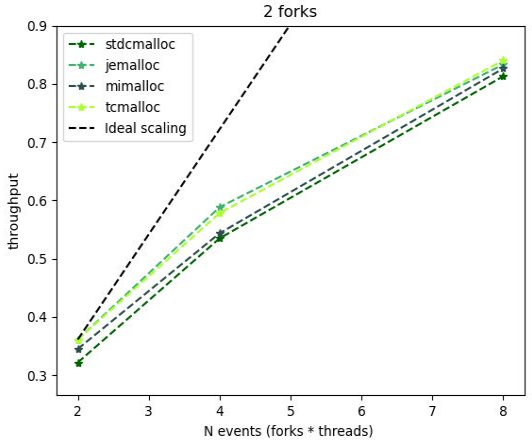
2 forks and 4 thread combination best for all allocators

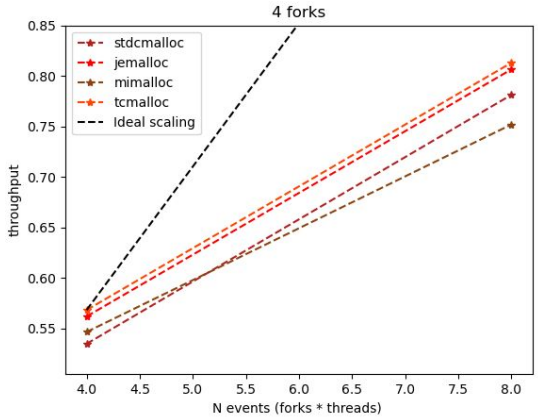
Jemalloc and tcmalloc have best throughput

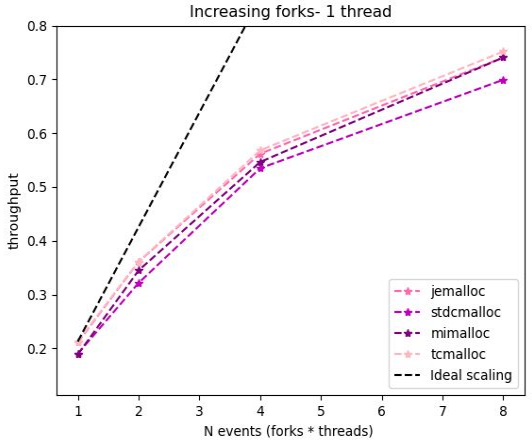
Look at other factors to see if one allocator is better than another

NB: 1 fork graph the jemalloc and tcmalloc line overlap





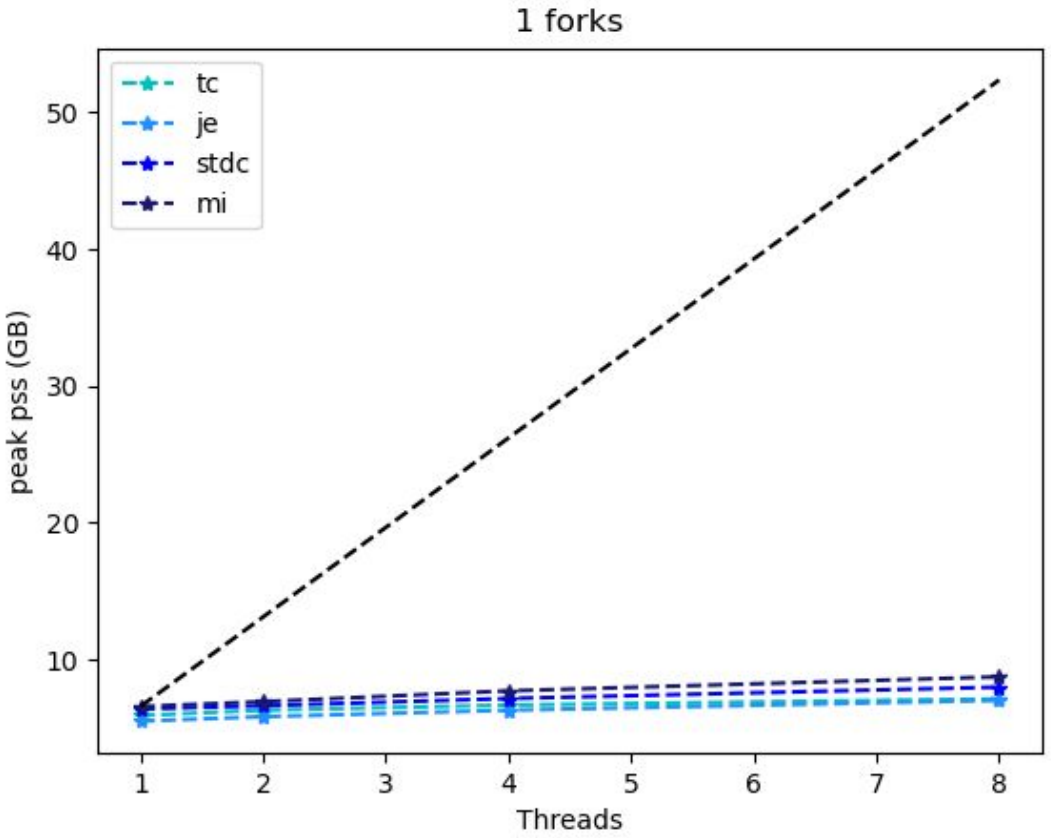




Peak pss

Comparing to scaling line, equivalent to the amount of memory required if using N computers

All found to be a lot less than scaling

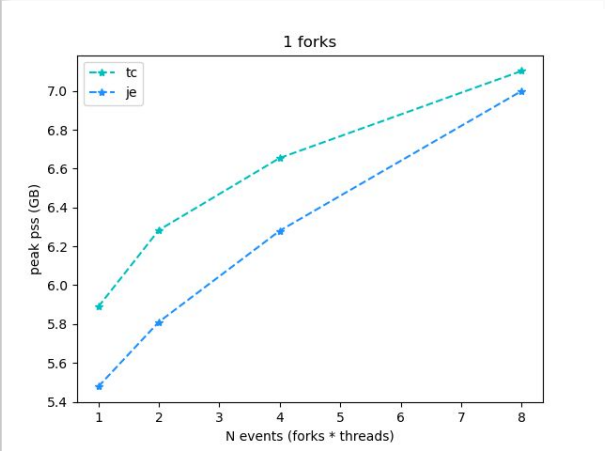


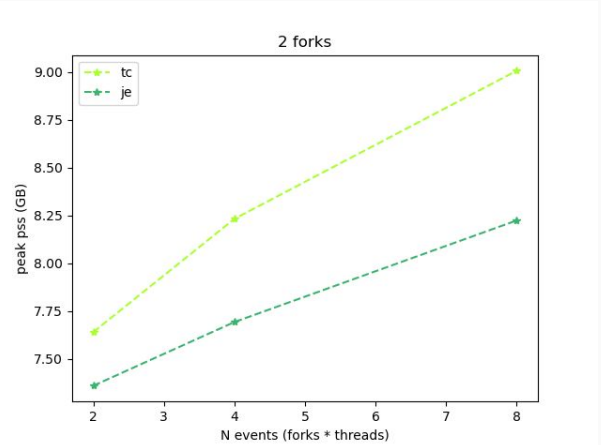
Peak pss

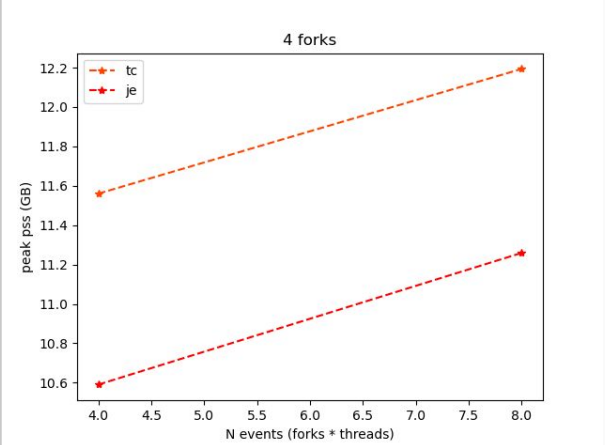
Jemalloc uses the least memory

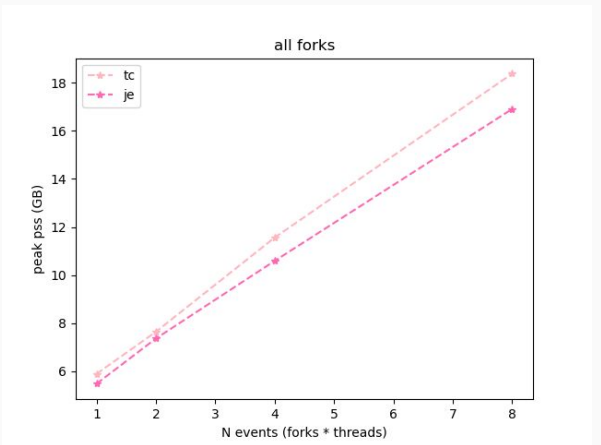
May be a reason to use jemalloc over tcmalloc

Mi and stdc not competitive for throughput or memory









Summary: memory allocators

Use of different memory allocators, forks and threads on memory usage and throughput of the HLT

jemalloc better throughput and memory usage overall

Being tested in the next technical run

Efficiency of the LRT

LRT

Large radius tracking

High level trigger does track reconstruction in different ROI’s identified by the L1 trigger

Originally designed for tracks that start at or near the initial collision, LRT new algorithms optimized to find tracks starting at a larger radius

Track parameters

Defining tracks by perigee parameters

- d0: radial distance from beam line
- z0: distance along beamline from collision
- ϕ(phi): curve of track
- η(eta): from θ (theta)
- pT: transverse momentum

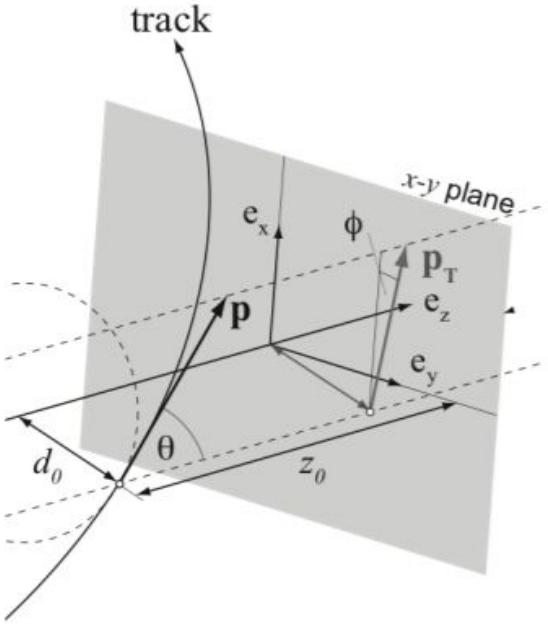


Figure 1.5: Perigee parameters. Figure by A. Salzburger.

https://indico.nbi.ku.dk/event/758/attachments/1684/2344/trackingnote.pdf

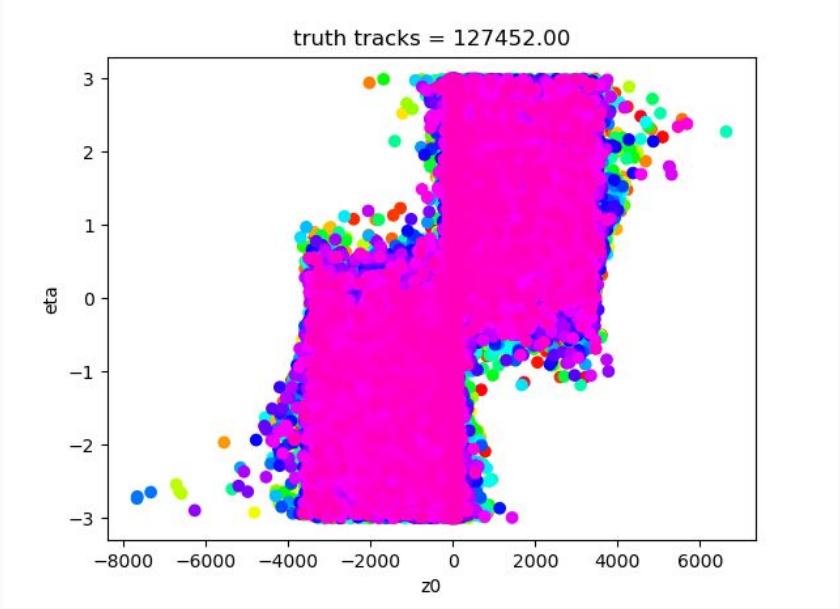
Track types

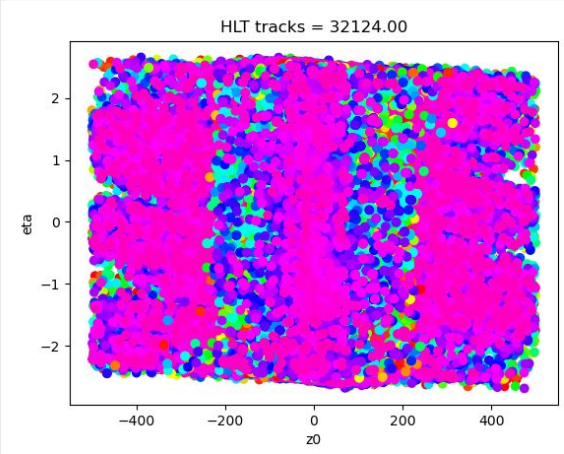
Truth: tracks generated by simulation

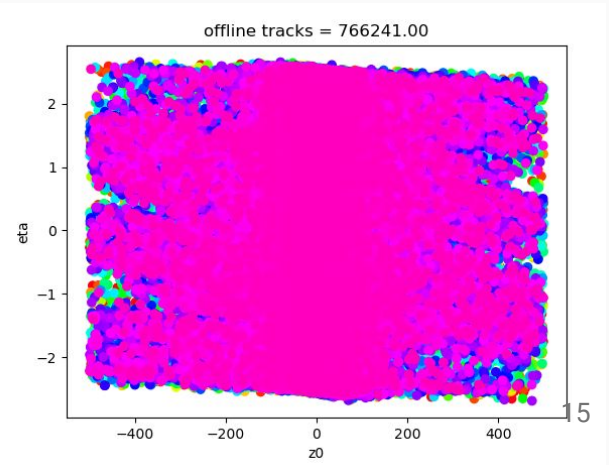
FTF: Region of interest, quick reconstruction

Offline: whole detector, longer time

Plotting all the tracks, 2 parameters







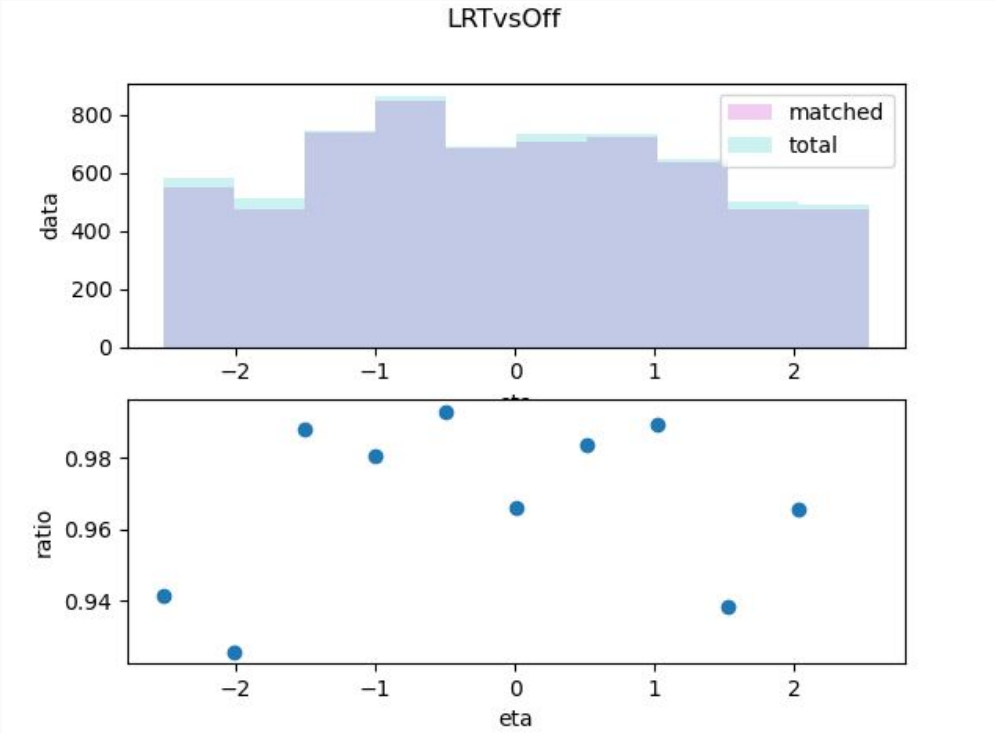
Method

1554 events

Matching tracks $\Delta\phi$<0.1 and $\Delta\eta$<0.1

Compare the histograms and then take the ratio

Overall efficiency and the efficiency in terms of 5 parameters



Efficiency

Overall efficiency

Shows discrepancy between HLT and offline

Planning on investigating the cause of this difference

Comparison	Efficiency
HLT vs Offline	78.44%
LRT vs Offline	63.85 %
HLT vs Truth	42.51 %
LRT vs Truth	25.37 %
Offline vs Truth	77.43%

Summary: LRT efficiency

Looked at 1554 events to find efficiency

Found issue between FTF and offline as found in earlier investigations

Compare matched and unmatched to parameters

Hoping to go on to find reasons for discrepancy