The Event Buffer Management for MT-SNiPER

Jiaheng Zou, Tao Lin, Weidong Li, Xingtao Huang
Ziyan Deng, Guofu Cao, Zhengyun You

CHEP 2018, Sofia Jul. 12, 2018

Outline

- SNiPER Software Framework
- Global Stream and Buffer
- Performance Test
- Summary

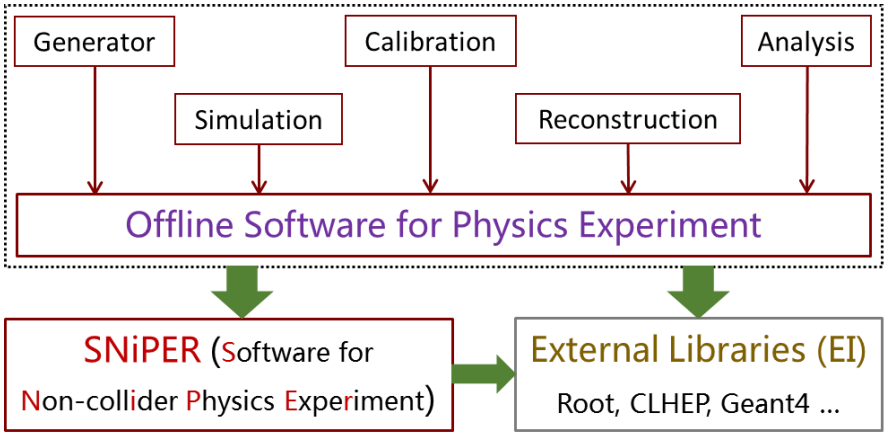
The SNiPER Software Framework

SNiPER: a general purpose offline software framework

- Specific to non-collider HEP experiments
- Lightweight and simple to use
- Open source (LGPL 3.0)

Implementation

- Hybrid of C++ and Python
- Thread safe



Current Status

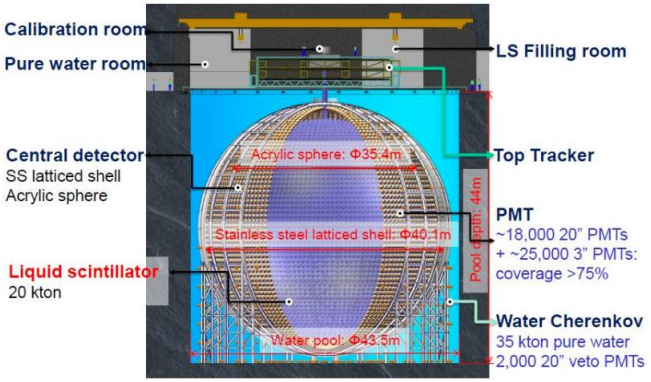
- Used by many experiments (JUNO, LHAASO, CSNS, nEXO)
- More potential users

Event Memory for JUNO

Anti-neutrino Inverse Beta Decay

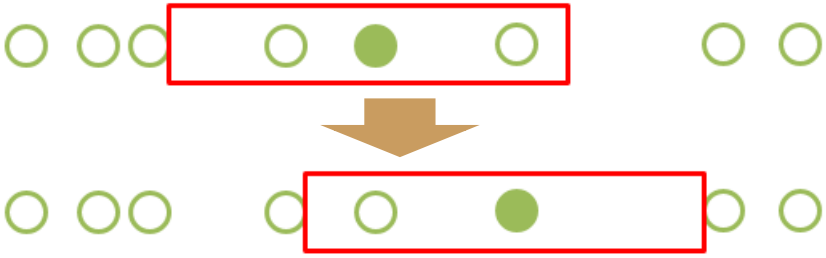
- In Central Detector, e^+ and n are triggered separately

$$\bar{\nu}_e + p \rightarrow e^+ + n$$
$$e^+ + e^- \rightarrow 2\gamma$$
$$n + Gd \rightarrow Gd^* \rightarrow Gd + \gamma_s \ (8\mathrm{MeV})$$



Events Time Correlation Analysis

- Event buffer with a time window

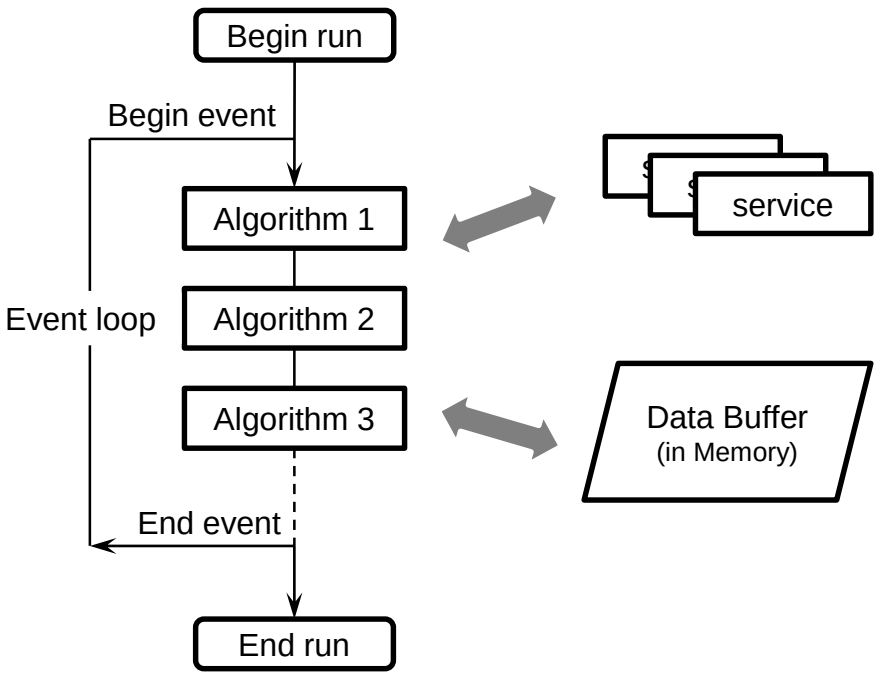


time window moves on the event stream

MT-SNiPER Enhancement

- Event Level Multithreaded Parallel Processing
  - Based on Intel TBB (Threading Building Blocks)
  - Events are dispatched to threads and processed concurrently
- Key Features
  - Non-invasive to the SNiPER kernel modules
    - Parallel features are implemented as a wrapper of the kernel
    - We can switch between serial and parallel mode smoothly
  - Quite straightforward and understandable with multiple instances of SNiPER Task
  - (Almost) transparent to users and algorithm developers
    - Minimize the migration costs from serial to parallel computing

SNiPER Task



A Task manages

- a sequence of Algorithms
- a group of Services
- a Data Buffer for events in memory

An Algorithm represents

- a unit for event data calculation

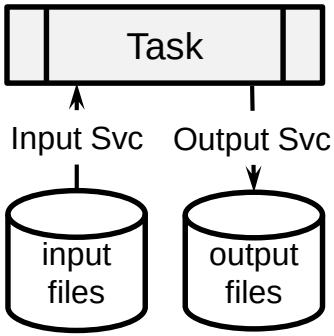
A Service represents

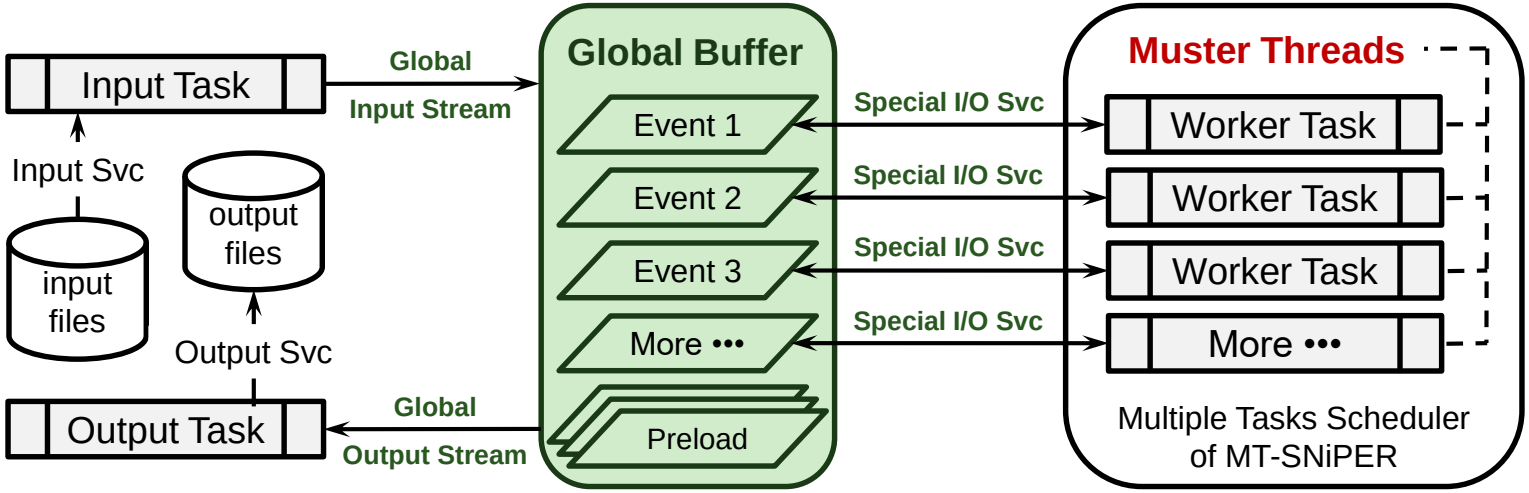
- a unit for common functionality

- Task controls the event loop procedure of the algorithms sequence
- There can be multiple Task instances in a single job
  - Each Task instance can be configured with different contents

Code Reusability in MT-SNiPER

- What can be reused from serial version
  - I/O services
  - Algorithms and other services in a Task
- What's new with MT-SNiPER
  - Global I/O Stream and Global Buffer
  - Specialized I/O services to access the Global Buffer in Worker Task(s)
  - Configure multiple Task instances for I/O and Worker of MT-SNiPER



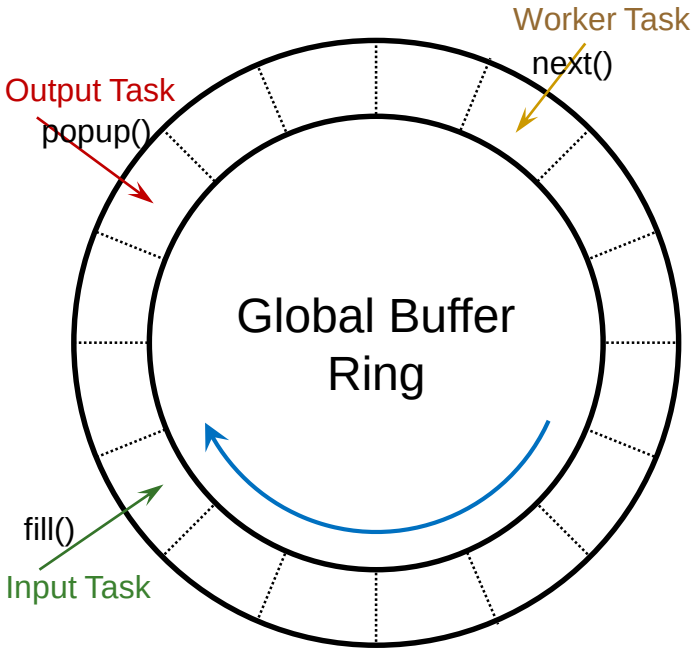


Global I/O Stream and Buffer

- Disk I/O are typical critical resources in MT computing
- Benefits of the Global I/O Stream
  - Global I/O Tasks are decoupled from Worker Tasks
  - Global I/O Tasks is lock free to access the Disk
- Benefits of the Global Buffer
  - Thin lock between Worker Tasks
    - Memory accessing is much faster than disk
    - Only event references are sent to Workers
  - Keep events in right order for writing out (FIFO)

Global Buffer Management

- A ring with configurable capacity and cordon
  - Each element has a status code (valid/being processed/processed)
- Operations (in different threads)
  - Filling by Input Task
    - Fill an event and notify a waiting Worker
    - Wait for a signal when it's full
  - Access by Worker Task
    - Get the next event with a thin lock
    - Notify the Output Task after processing
    - Wait for a signal when empty
  - Popup by Output Task
    - Popup and write the first event if it has been processed
    - Notify the Input Task when the events in buffer is less than cordon
    - Wait for a signal when the first event is not processed



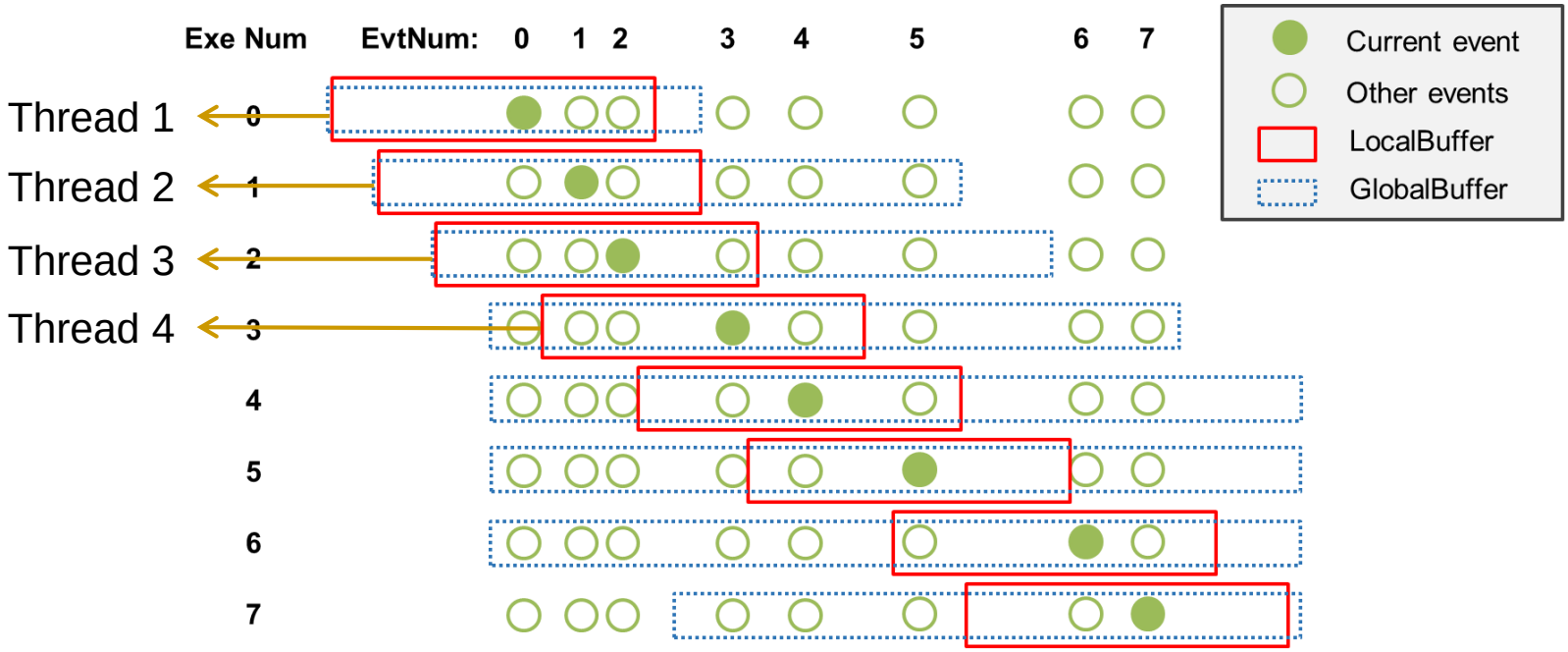
Local Buffer in Worker Task

- Each Worker Task has its own local buffer
  - The event(s) being processed by this Worker
- Keep one event reference of Global Buffer
  - The Global Buffer prototype performs pretty well
  - This is enough in most cases
- Keep more events according to event timestamps
  - Necessary when we need events' time correlation analysis
  - There can be overlaps of Workers' local buffers
  - The events dispatch is a little different, which is being developing
    - A sub-sequence of events are sent to Worker in each accessing

The Global Buffer and Local Buffer

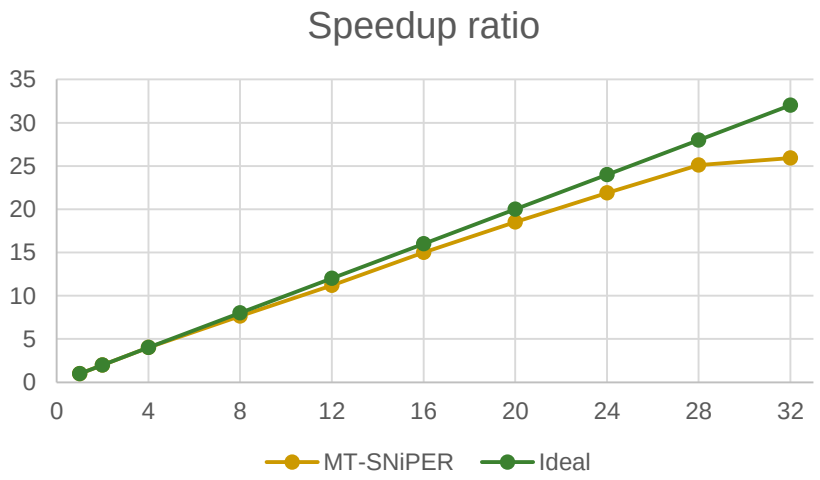
Global Buffer – bind to I/O streams, cache all events in memory, keep events in right order for output

Local Buffer – same as the serial version of DataBuffer



Performance Test

- Test environment and test cases
  - 32 Cores (8 Core Intel(R) Xeon(R) CPU E5 * 4)
  - A dummy algorithm ~ Avg. 100 executions / second



- The speedup ratio is almost linear when < 20 threads
- The resource overhead is very small

Summary

- MT-SNiPER is a non-invasive wrapper for SNiPER, which supports event level multithreaded computing
- Global Stream and Global Buffer is implemented for the data management of MT-SNiPER
  - The concept is simple with SNiPER's multiple Task instances
  - Thin locks and high performance
  - To be improved in case of events' time correlation analyses

Thanks !