Bruno multi-thread

A. Di Simone

INFN CNAF

Introduction

- Full simulation software is not exploiting at all the many parallelization possibilities offered by modern computing
  - Not only SuperB, but the vast majority of existing HEP simulations run sequential, single-thread programs
- Main limitation in the past has been that the main simulation toolkit (Geant4) was not designed to be run in any non-sequential mode
- Recently, things have changed, with the release of a prototype of G4 suitable for multi-thread applications
  - Result of several years of development, now ready for usage by “experts”, even though most likely not yet for official deployment
- SuperB is in the very interesting situation of being a new project, with a brand new simulation software
  - Fully featured
  - Relatively small code base
  - Very small community of developers
- The decision was taken to experiment with the new G4 prototype and adapt Bruno to use it

Geant4-MT

- It is a variant of the stock Geant4 distribution, whose general architecture has been modified to allow event-by-event parallelism
  - A master thread initializes the geometry and the “shareable” parts of the physics
  - Then it launches *n* worker threads
    - Within the same physical computing element
  - Generated events are dispatched to the worker threads for processing
    - Every event is fully simulated by one thread
    - Every thread simulates only one event at a time
  - Equivalent to splitting the generated events into subsamples and processing them through *n* independent processes
    - Multi-threading saves most of the initialization time, and vastly reduces the memory footprint, as threads can share memory

Bruno and multi-threading

- Migrating a simple application to Geant4-MT is just a matter of compiling against the new G4 and modifying the client code in a few well identified places
  - Procedure is well documented
- Unfortunately, Bruno is not a simple application, and no existing migration how-to could be effectively used
- Moreover, Bruno was not thought to be run in parallel, and some parts of its code had to be adapted/rewritten
  - Work presented here is the result of patient inspection of the G4-MT code, rewriting of some Bruno code, interaction with friendly G4 developers, educated guessing and sheer luck
- Results are encouraging, but far from being ready for deployment and production

Migrating Bruno (1)

- Migration has been planned and divided into steps
  - Compile against the new G4
    - Goal: adapt existing code to the new interface
  - Launch a job to read the full SuperB geometry and process in parallel some geantinos. **No sensitive detectors, no customization**, **no writing to file**
    - Goal: test basic tracking functionality, geometry sharing between threads.
    - Note: no physics yet, no magnetic field, no hits, no MC truth
  - Add the magnetic field
  - Migrate one by one the user actions (i.e. our customizations), solve any problems
    - Goal: at this stage MC truth is created (though not written to root file)
    - Some simulation output becomes accessible, like the number or recorded true particles

Migrating Bruno (2)

- Activate one by one the Sensitive detectors, solve any problems
  - Goal: detector hits are now created and stored in memory (not written to root file)
  - Number of hits per event is accessible
- First full scale performance test, using single electrons
  - Goal: turn on some ***physics***, at last
- Migrate the RadBhabha generator
  - Goal: being able to simulate realistic events
  - Requires careful handling of (lots of) static data

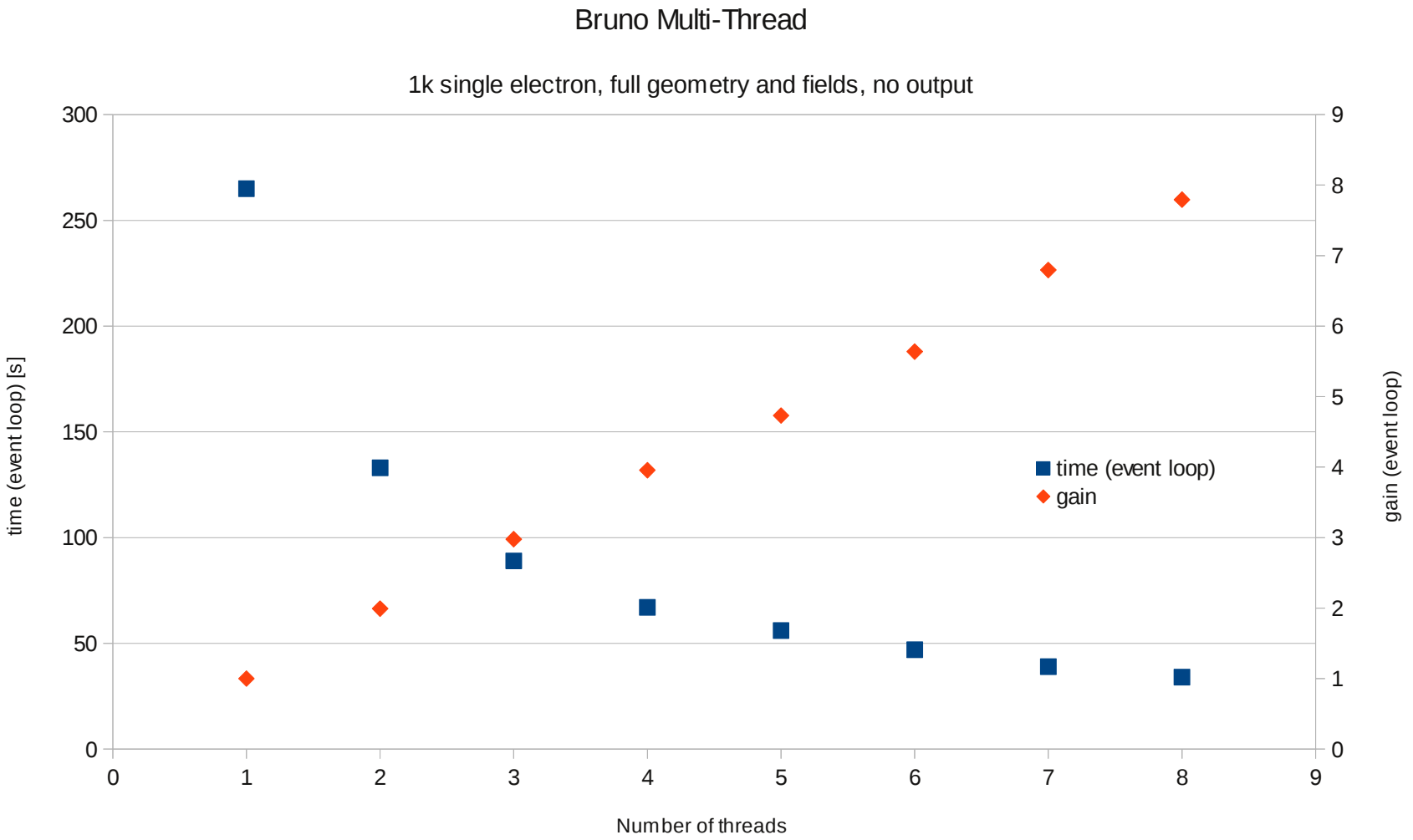
- Now, we have a running, fully featured simulation, with the same functionalities as the existing Bruno release
- This means that, as far as G4 is concerned, the migration is completed
- What we still miss is the persistency, i.e. the ability to write hits and MC truth to file
  - This requires dealing with ROOT, not with G4

Bruno migration - status

- All steps described above have been completed, with the exception of the output to ROOT, which is proving to be more problematic than expected
  - In a nutshell, some parts of ROOT's I/O are not (meant to be) thread safe
  - Ongoing interactions with the ROOT team to find some workarounds
- Will show in the following some performance plots
- Not a sophisticated analysis yet
  - Run jobs interactively on a CNAF UI
  - Monitor memory and time to complete the event loop as a function of the number of working threads
    - Initialization time subtracted: only event loop considered

Bruno-MT: performance (1)

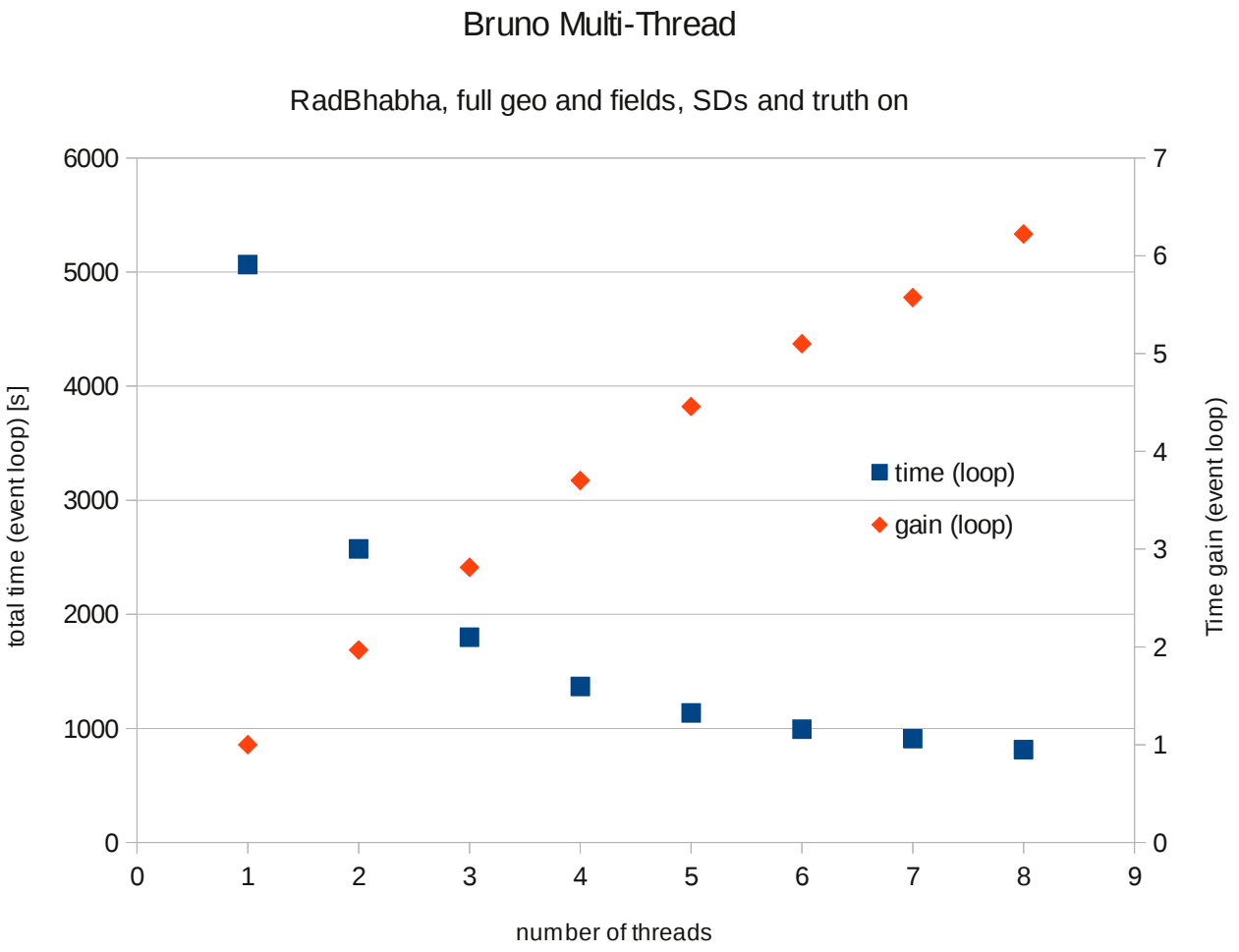
- Single electrons
- Scaling with number of threads is perfectly linear



Bruno-MT: performance (2)

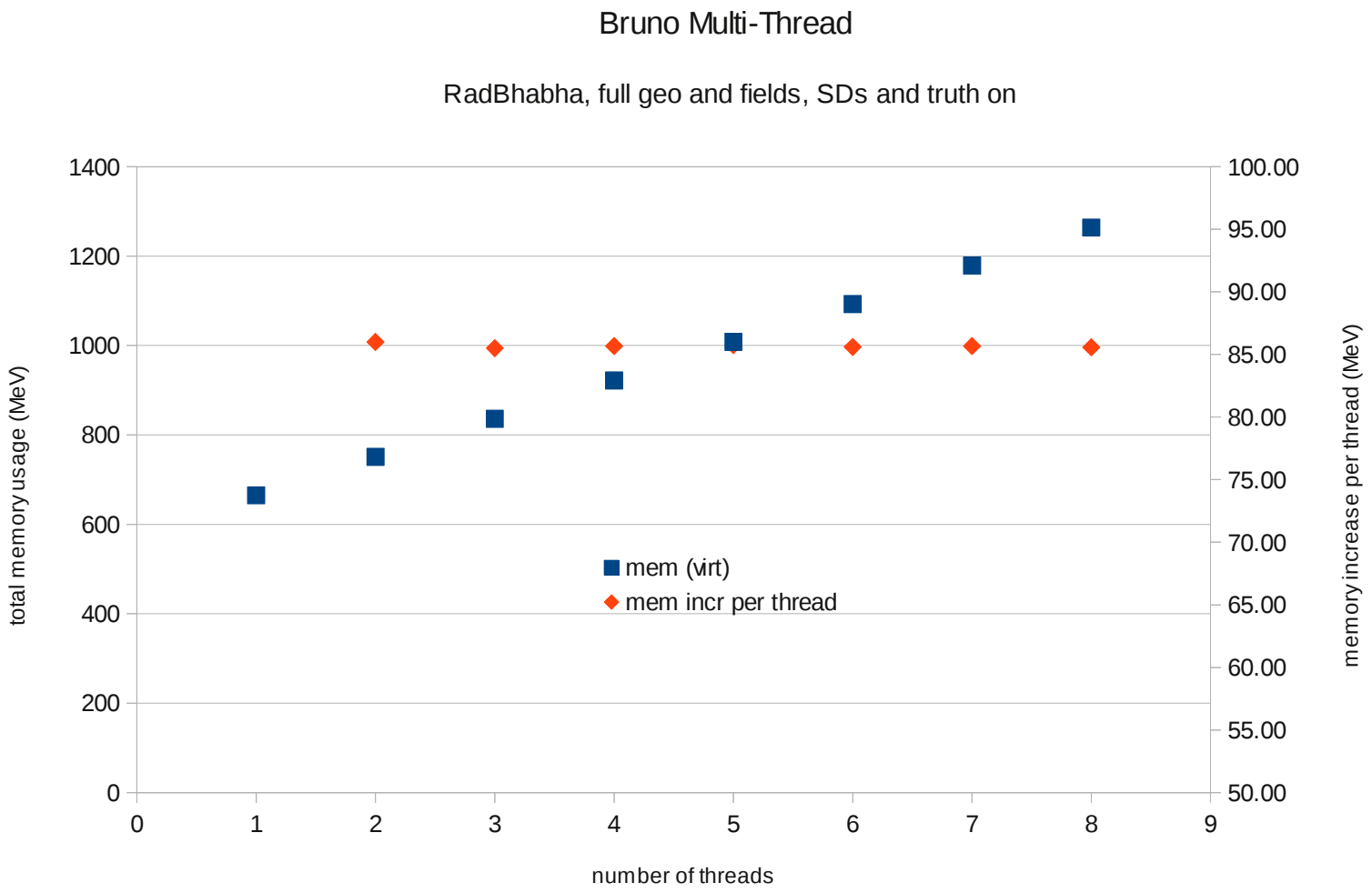
- Radbhabha
  - minDE set to 0.7 to speed up simulation and allow interactive running
    - Not physically accurate, but realistic event topology

- Linearity of the scaling is not as perfect as for single electrons
  - Needs better investigation
  - May come from the generator (several mutexes had to be used)



Bruno-MT: performance (3)

- Memory increase per additional thread in the RadBhabha benchmark



Conclusions

- The migration of Bruno to Geant4-MT was a very interesting exercise
- Result is a fully functional simulation software
  - Performance scales as expected
- Some points still need attention
  - User Interface: some custom commands and configuration hooks were disabled and need to be migrated
  - Random number management and reading input events from file
  - Writing simulation result to file (pending developments/workarounds on the ROOT side)
  - Integration with the default SuperB build system: all development was done in custom (cmake-based) setup.
- Plans
  - Not clear to me how to proceed from here
  - The exercise is interesting per se, and complete enough, as it is
    - In any case, this is unlikely to go into production any time soon, since the G4 release it uses is itself not recommended for “real life” applications