Interactive Hydra with ROOT: macros and prompt

Status report

A. Augusto Alves Jr

Presented at 37th ROOT Parallelism, Performance and Programming Model,
CERN, April 19, 2018

Hydra and ROOT

Hydra is a header-only, templated C++11 framework designed to perform common tasks found in HEP data analyses on massively parallel platforms.

- It is implemented on top of the C++11 Standard Library and a variadic version of the Thrust library.
- Hydra is designed to run on Linux systems and to deploy parallelism using OpenMP, CUDA and TBB on the suitable devices.

Status of the integration with ROOT:

- Previous report: `https://indico.cern.ch/event/708311/`
- Two issues solved: ROOT-9335 and ROOT-9326
- **Status:** ROOT can now run Hydra interactively: macros and prompt
- In the ROOT environment, Hydra algorithms run using TBB and CPP backends.

Configuration

- Configuration: `export ROOT_INCLUDE_PATH=/path_to_hydra`
- Backend selection:

```
#ifndef HYDRA_HOST_SYSTEM
#define HYDRA_HOST_SYSTEM CPP // or TBB
#endif

#ifndef HYDRA_DEVICE_SYSTEM
#define HYDRA_DEVICE_SYSTEM TBB // or CPP
#endif
```

- Invoking Hydra:

```
//prompt or macro
#include <hydra/device/System.h>
...
auto data = hydra::device::vector(1e6);
...
```

Example: 3-body phase-space generation

```
1   // includes <hydra/...> omitted for brevity
2
3   void three_body_phsp(size_t nentries=100000)  {
4
5     double B0_mass    = 5.27955;   // B0 mass
6     double Jpsi_mass  = 3.0969;    // J/psi mass
7     double K_mass     = 0.493677;  // K+ mass
8     double pi_mass    = 0.13957061;// pi mass
9     // Mother particle
10    hydra::Vector4R B0(B0_mass, 0.0, 0.0, 0.0);
11    // Create PhaseSpace object for B0-> K pi J/psi
12    hydra::PhaseSpace<3> phsp{Jpsi_mass, K_mass, pi_mass};
13    // Timing
14    std::chrono::duration<double, std::milli> elapsed_d;
15    {
16            hydra::Decays<3, hydra::device::sys_t> Events_d(nentries);
17            auto start = std::chrono::high_resolution_clock::now();
18            phsp.Generate(B0, Events_d.begin(), Events_d.end());
19            auto end = std::chrono::high_resolution_clock::now();
20            elapsed_d = end - start;
21    }
22  }
```

- Execution in the usual way: `root three_body_phsp.C` or `root three_body_phsp.C++`

Comparison with `ROOT::TGenPhaseSpace`

	ACLIC / Compiled (ms)	CLING (ms)
Hydra/TBB(standalone)	59.021	–
Hydra/CPP(standalone)	395.617	–
Hydra/TBB	63.107	15990.5 (15.9 s)
Hydra/CPP	392.729	36916.2 (36.9 s)
ROOT::TGenPhaseSpace	384.696	1461.6 (1.4 s)

- Number of events: 1 million
- System: Intel Xeon CPU E5-2620 v3 2.40GHz x 12

Summary

- Hydra is working well in ROOT environment, both ACLIC and CLING.
- A suite of examples/macros has be added to the develop branch in the GitHub repository: `https://github.com/MultithreadCorner/Hydra`
- New version to be released ASAP.