MadGraph on GPU

Junichi Kanzaki (KEK)

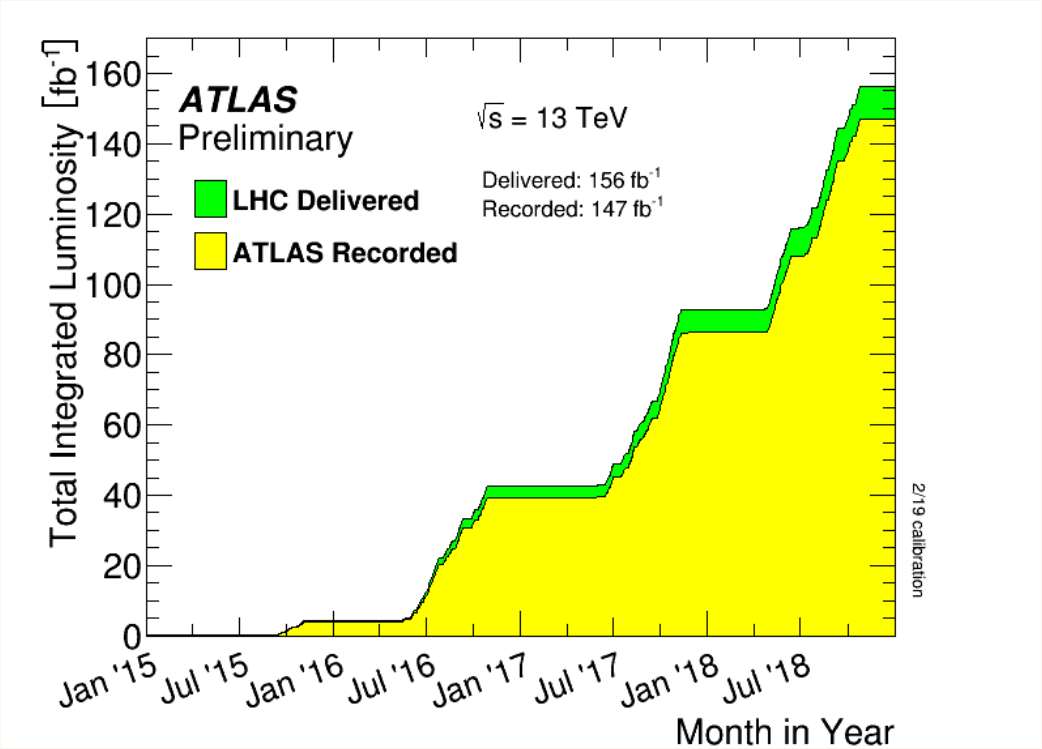
HOW2019

@ THOMAS JEFFERSON NATIONAL ACCELERATOR FACILITY

March 19, 2019

We Need More Speed ...

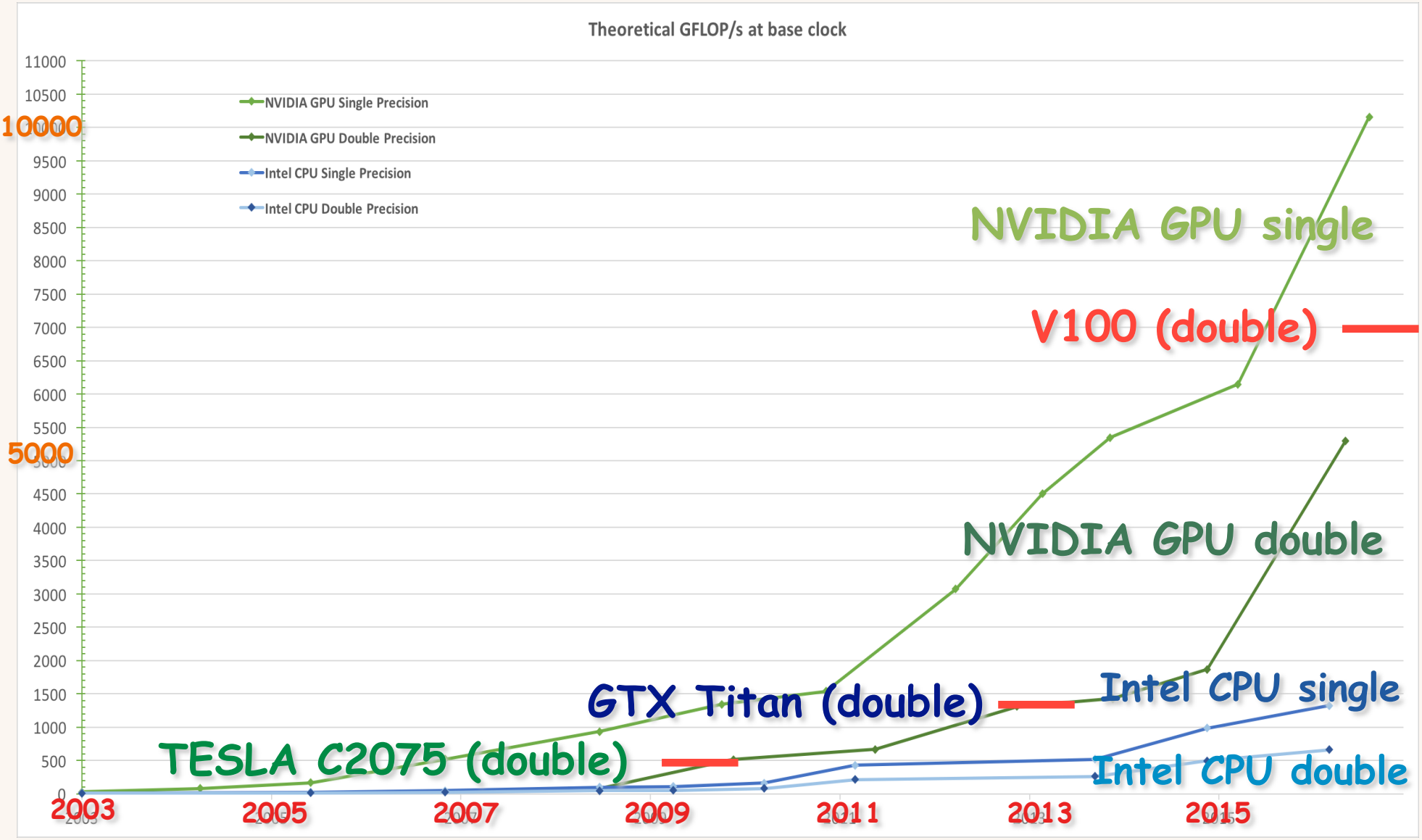
- Increase of amount of LHC data
  - Run1+Run2: ~150fb^{-1} up to 2018
  - And more: 300fb^{-1} until 2022, 3000fb^{-1} until 2035
  - Huge amount of simulation data is necessary for physics analysis to keep reliability of physics analysis results.



MadGraph5_aMC@NLO

- MadGraph is a general framework for computations of cross sections and generation of events of various physics processes at NLO, including new physics, and is widely used in HEP physics analysis.
- Based on the system many useful tools are developed.
- Improvement of its performance by HPC should be essential to keep the reliability of physics results with "big data" analysis.

GFLOP/s at Base Clock



Presented by J. Kanzaki at HOW2019 in March 19, 2019

History

- Since the beginning of 2008 with close collaboration with the MadGraph group we have been working on the development of parallel codes on GPU to improve performance of event generation softwares.
- First we started testing helicity amplitude codes (HELAS in FORTRAN) on GPU (corresponding codes: HEGET in CUDA) and check results and performance.

History

- Basic tests of HEGET functions were done with the QED (n-photon), QCD (n-jet) and more general SM processes with massive particles.
- Test with MC integration is used in the paper for SM processes. VEGAS/BASES* (MC integration) and SPRING* (event generation based on the integration) packages for GPU are developed.

S.Kawabata,Comput.Phys.Commun.41(1986)127.

Bibliography

- QED: K. Hagiwara, J. Kanzaki, N. Okamura, D. Rainwater and T. Stelzer, Eur. Phys. J. C66 (2010) 477, e-print arXiv:0908.4403.
- QCD: K. Hagiwara, J. Kanzaki, N. Okamura, D. Rainwater and T. Stelzer, Eur. Phys. J. C70 (2010) 513, e-print arXiv:0909.5257.
- MC integration (VEGAS & BASES): J. Kanzaki, Eur. Phys. J. C71 (2011) 1559, e-print arXiv: 1010.2107.
- SM: K. Hagiwara, J. Kanzaki, Q. Li, N. Okamura, T. Stelzer, Eur.Phys.J. C73 (2013) 2608 (2013), e-print arXiv:1305.0708v2.

For improvement of performance

- Do in parallel as much as possible:
  the gain of performance depends on the fraction of parallelizable part.
- Fill multiprocessors of GPU as many as possible:
  try to keep the fraction of processors not in use as small as possible.
- Efforts to develop new algorithms optimized to GPU is important to achieve further gain in performance.

MadGraph and GPU

Simplified view of event generation with MadGraph:

1. Generation of amplitude program of various physics processes: helicity amplitude code and amplitude program.
2. Multi-channel integration of multi-physics processes
3. Event generation based on the integrated results with multi physics processes.

Amplitude Program

- Helicity amplitude code generated by "aloha": the functionality to generate CUDA code is already included (not in release version) thanks to the MadGraph team. Need finalization.
- Amplitude function (matrix.f) can be easily converted to CUDA or directly generated in CUDA format.

Monte Carlo integration on GPU

Multi-channel integration of multi-physics subprocesses

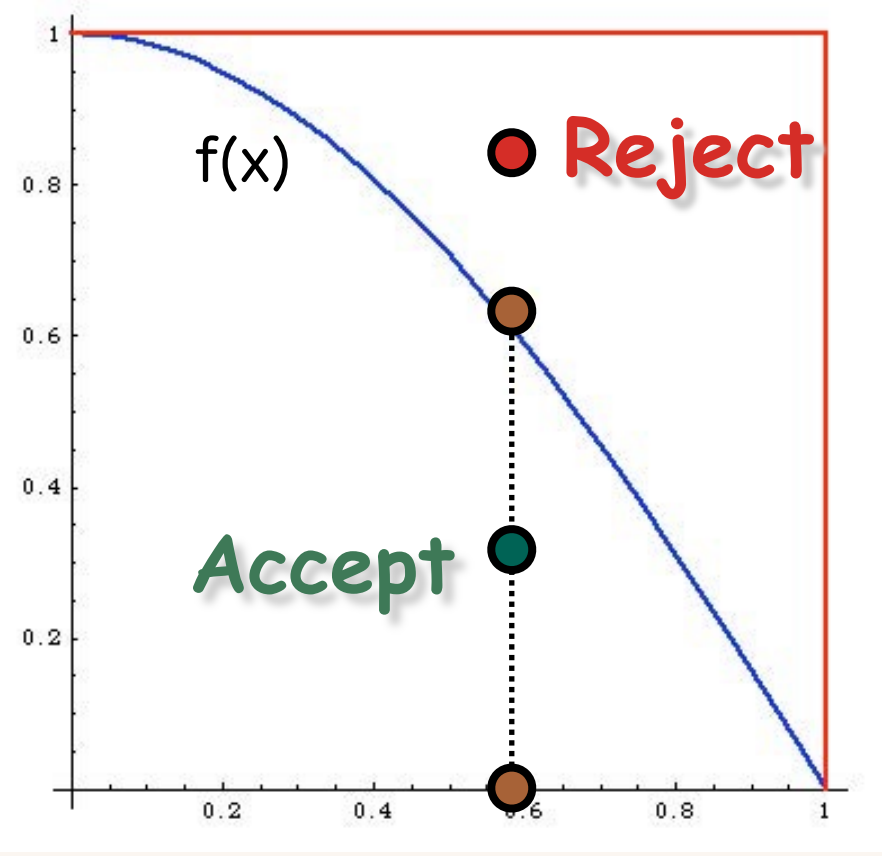
- most time-consuming part of integration is already available on GPU: VEGAS/BASES.
- multi-channel integration can be fitted to GPU more efficiently:

$$I = \int d\vec{\Phi} f(\vec{\Phi}) = \sum_{i=1}^{n} \int d\vec{\Phi}\, g_i(\vec{\Phi}) \frac{f_i(\vec{\Phi})}{g_i(\vec{\Phi})} = \sum_{i=1}^{n} I_i \,,$$

$$f_i = \frac{|A_i|^2}{\sum_i |A_i|^2} |A_{\rm tot}|^2 \,,$$

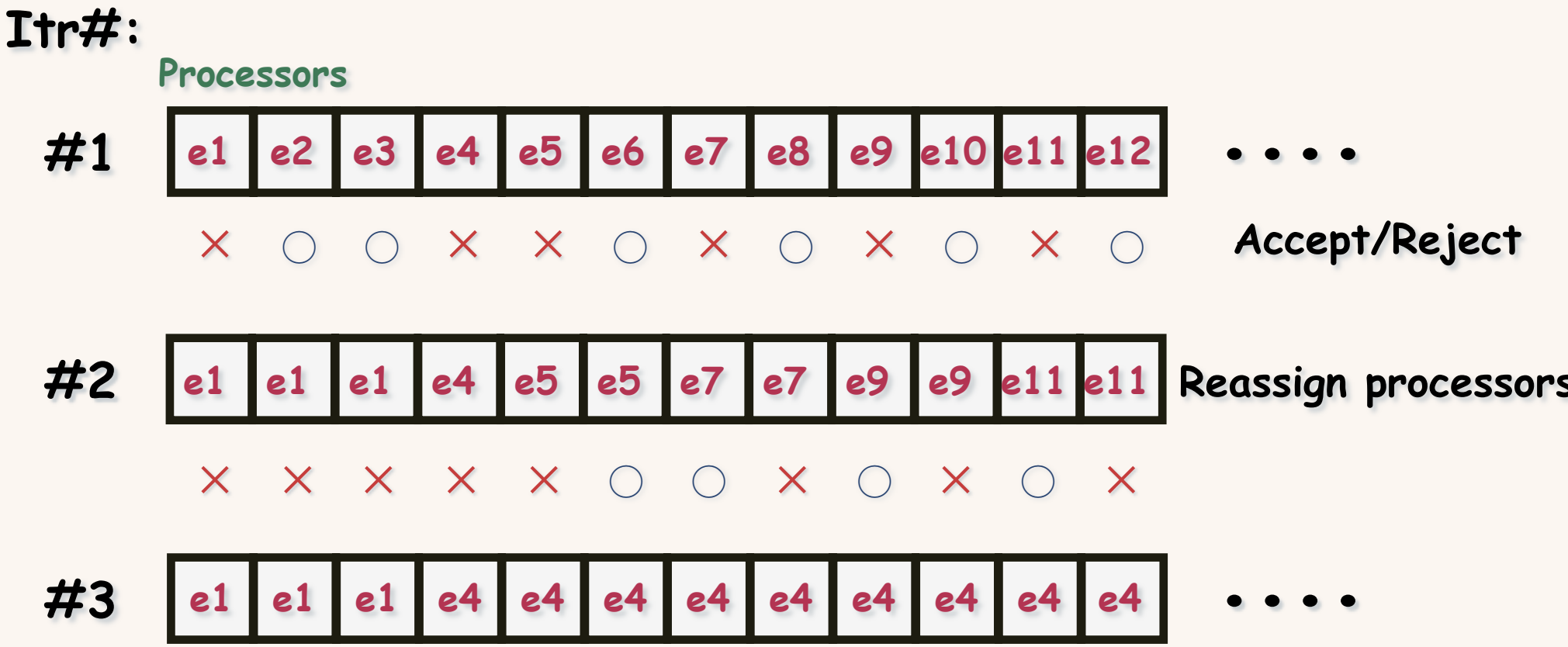
Event Generation on GPU

- Event generation of multi-physics processes
- SPRING generates unweighted events based on BASES integration results with improved and optimized algorithm.
- "Unweighting" requires "acceptance-rejection":
  -> the most inefficient cell determines the total performance.



SPRING on GPU (gSPRING)

- Allocate event generation at phase space points to processors that take care of generation of one event.



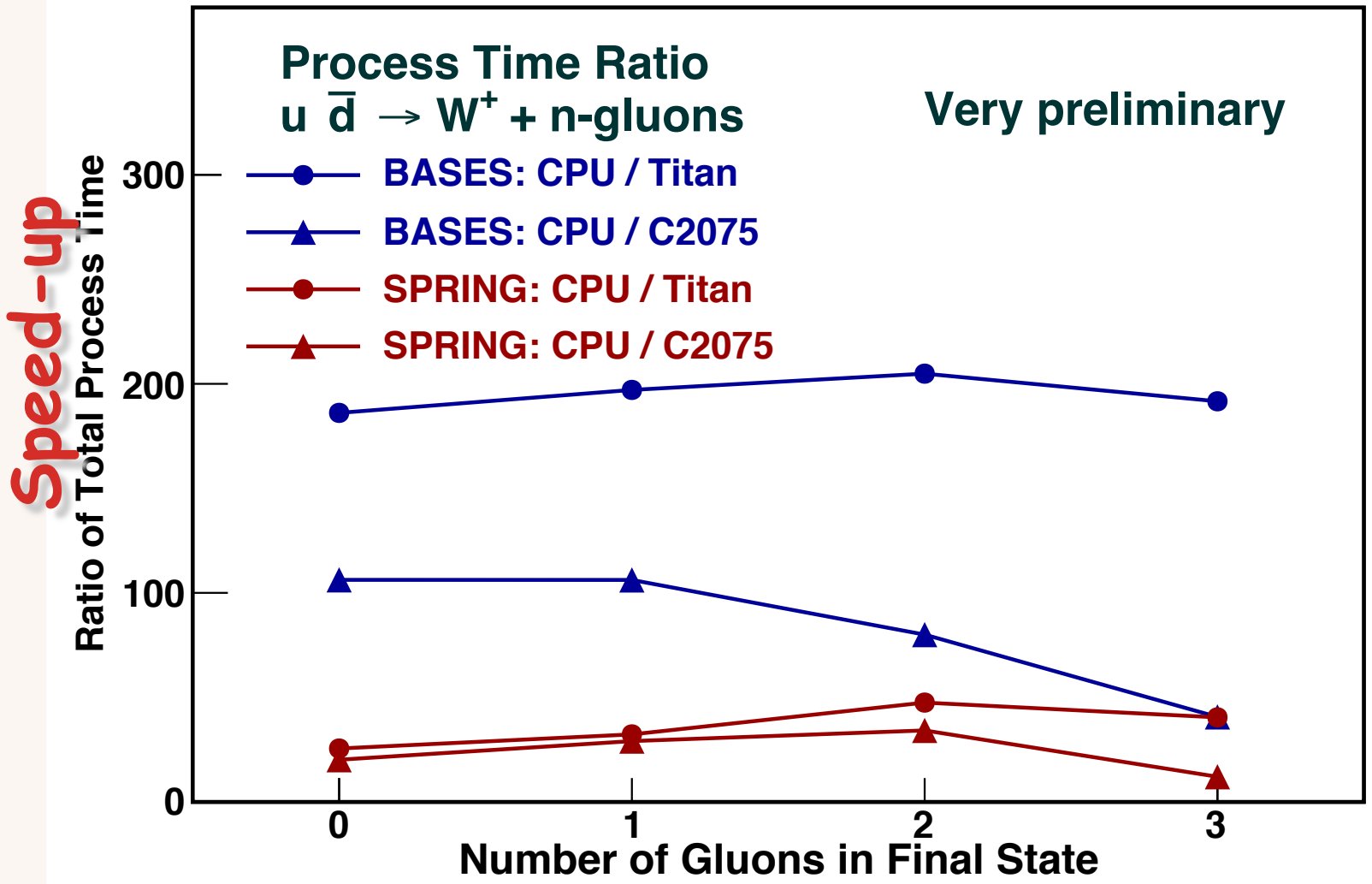
Our GPU Environment

	Titan	C2075	GTX580	GTX285	GTX280	9800GTX
CUDA cores	2688	448	512	240	←	128
Global Memory	6.1GB	5.4GB	1.5GB	2GB	1GB	500MB
Constant Memory	64KB	64KB	64KB	64KB	←	64KB
Shared Memory/block	48KB	48KB	48KB	16KB	←	16KB
Registers/block	65536	32768	32768	16384	←	8192
Warp Size	32	32	32	32	←	32
Clock Rate	0.88GHz	1.15GHz	1.54GHz	1.30GHz	←	1.67GHz
Announced	2013	2011	2010	2009		2008

- Code development: NVDIA GPUs + CUDA
- GTX Titan: Peak floating point performance
  4.5TFLOPS (single), max. 1.3TFLOPS (double)
- cf. TESLA V100: 5120 CUDA cores, 7TFLOPS (double)

Speed-up with GPU

Comparison of GTX Titan and TESLA C2075 with single core CPU (u+dx -> W^+ + gluons).



MadGraph on GPU

- Even though performance related parts of the system, "MC integration and event generation", are available on GPU, still more developments are necessary, especially for the handling multi-physics process generations.
- We will first accomplish efficient multi-channel integration of physics process.
- Multi-GPU and multi-task should be taken into account.

Personal Hope

- Not only for HLT and pattern recognitions performance improvements of software for offline analysis by HPC is necessary for coming huge amount of data.
- Other than event generations detector simulations (full and fast) and user analysis environment (ex.ROOT) are good candidate: we tried to run "PGS", fast simulation program included in MadGraph, on GPU and obtained speed-up of roughly one order.
- Performance of GPU hardware is rapidly evolving!