LHC Event Generation with GPUs

Joshua Isaacson

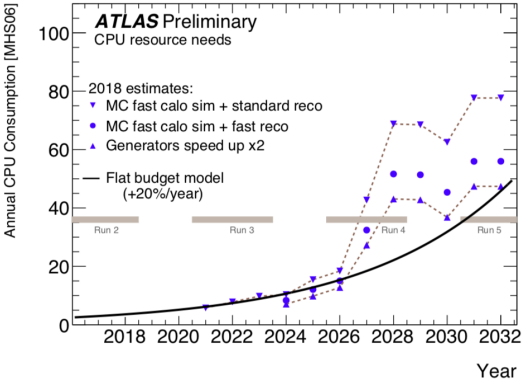
New Perspectives 2020

20 July 2020

FERMILAB-SLIDES-20-061-T

Motivation

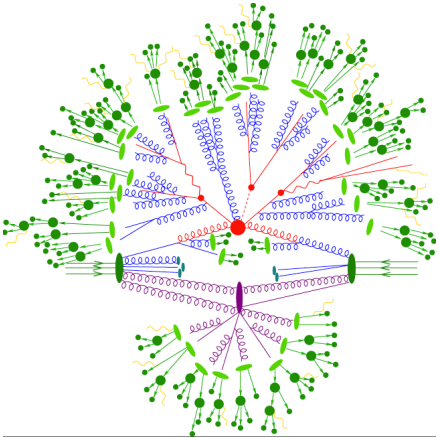
- LHC requires large number of Monte Carlo events
- Due to CPU costs, MC statistics will become significant uncertainty



[ATLAS]

Event Schematics

Factorization of an event:

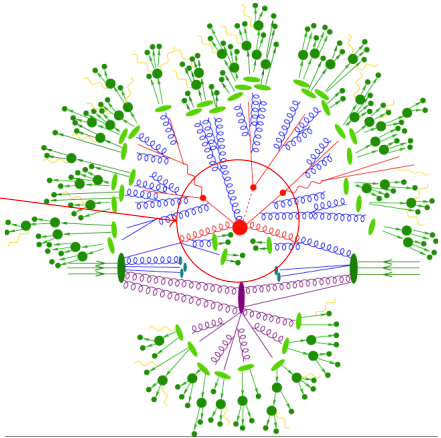


[Image from the Sherpa Authors]

Event Schematics

Factorization of an event:

- Hard Process

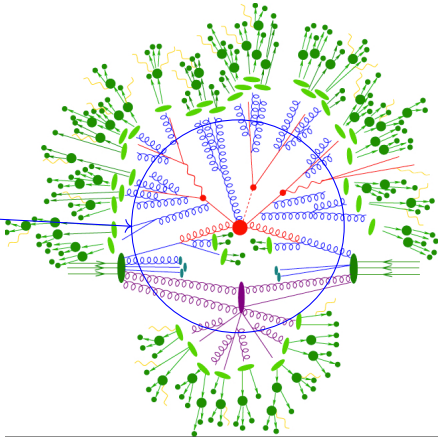


[Image from the Sherpa Authors]

Event Schematics

Factorization of an event:

- Hard Process
- Parton Shower

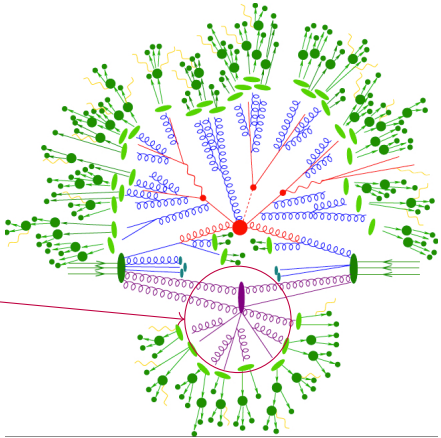


[Image from the Sherpa Authors]

Event Schematics

Factorization of an event:

- Hard Process
- Parton Shower
- Multiple Parton Interations

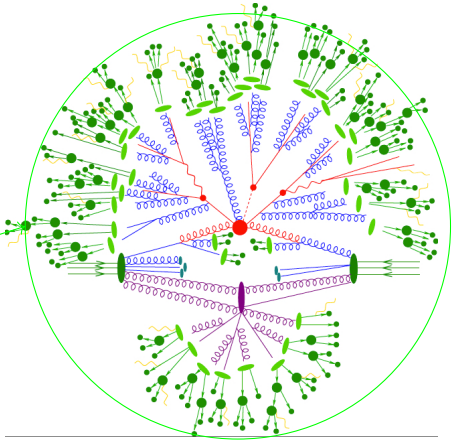


[Image from the Sherpa Authors]

Event Schematics

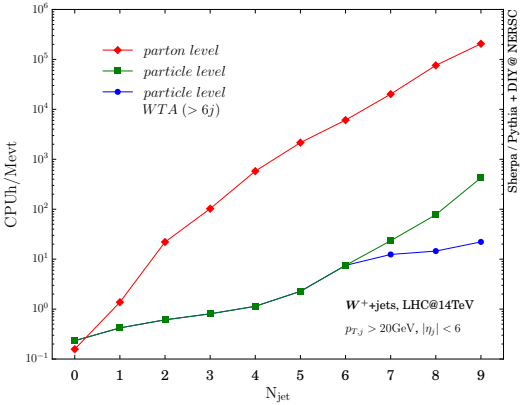
Factorization of an event:

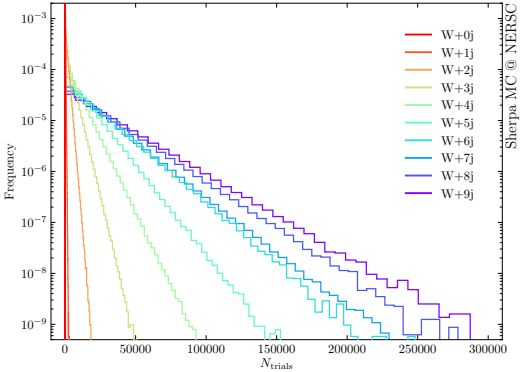
- Hard Process
- Parton Shower
- Multiple Parton Interations
- Hadronization



[Image from the Sherpa Authors]

Motivation





[S. Höche, S. Prestel, H. Schulz, 1905.05120]

- Time to generate an event dominated by hard process not shower
- Large computational cost for unweighting at high multiplicity

Recursive Matrix Element Generation

Brends-Giele Recursion

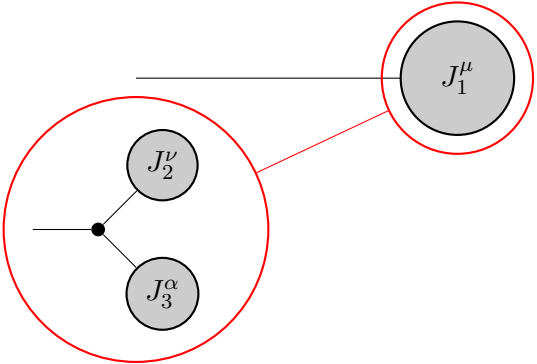
- Reuse parts of calculation
- Most efficient for high multiplicity
- Reduces computation from $\mathcal{O}(n!)$ to $\mathcal{O}(e^n)$

[Nucl. Phys. B306(1988), 759]

Recursive Matrix Element Generation

Brends-Giele Recursion

- Reuse parts of calculation
- Most efficient for high multiplicity
- Reduces computation from $\mathcal{O}(n!)$ to $\mathcal{O}(e^n)$



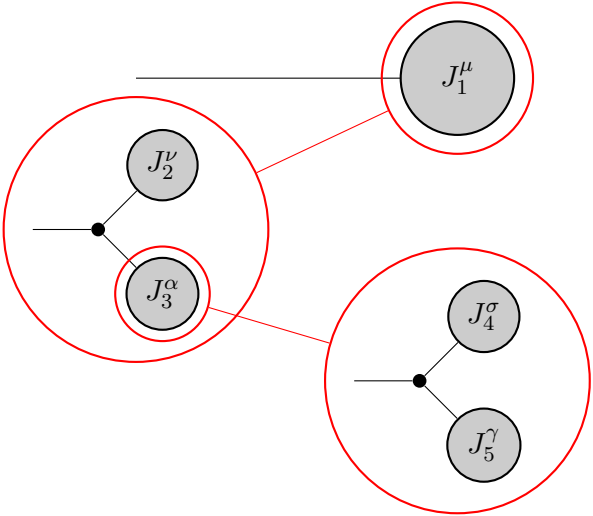
[Nucl. Phys. B306(1988), 759]

Recursive Matrix Element Generation

Brends-Giele Recursion

- Reuse parts of calculation
- Most efficient for high multiplicity
- Reduces computation from $\mathcal{O}(n!)$ to $\mathcal{O}(e^n)$

[Nucl. Phys. B306(1988), 759]

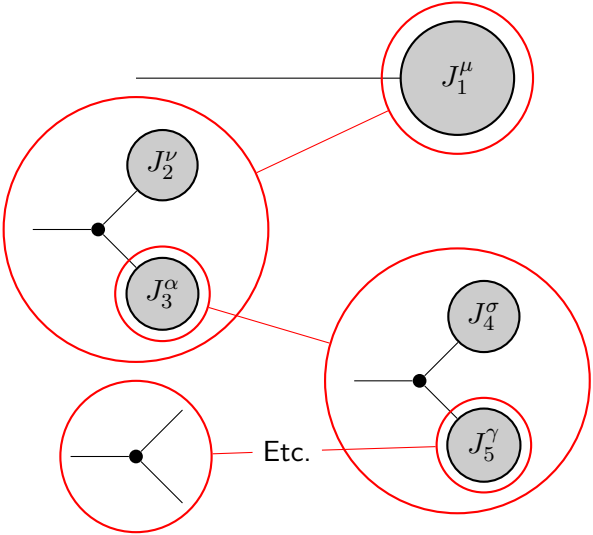


Recursive Matrix Element Generation

Brends-Giele Recursion

- Reuse parts of calculation
- Most efficient for high multiplicity
- Reduces computation from $\mathcal{O}(n!)$ to $\mathcal{O}(e^n)$

[Nucl. Phys. B306(1988), 759]



Why a GPU Implementation?

Next-Gen Supercomputer Aurora:

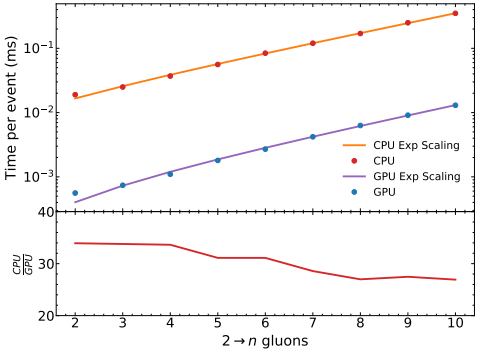
[https://alcf.anl.gov/aurora]

- Event generation is trivially parallelizable
- Aurora Compute Nodes:
  - 2 Intel Xeon processors
  - 6 Xeon arch-based GPUs
  - Unified Memory
- Take advantage of modern supercomputer setups

Preliminary Results

Results using:

- CPU: 2.66 GHz Xeon
- GPU: GTX 1050

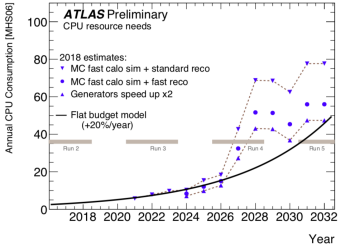


$2 \to n$ gluons	CPU (ms/pt)	GPU (ms/pt)	Speed-Up
2	1.9×10^{-2}	5.6×10^{-4}	33.5
3	2.5×10^{-2}	7.4×10^{-4}	33.2
4	3.7×10^{-2}	1.1×10^{-3}	34.0
5	5.6×10^{-2}	1.8×10^{-3}	30.9
6	8.4×10^{-2}	2.7×10^{-3}	30.9
7	1.2×10^{-1}	4.2×10^{-3}	28.6
8	1.7×10^{-1}	6.3×10^{-3}	27.7
9	2.5×10^{-1}	9.1×10^{-3}	28.0
10	3.5×10^{-1}	1.3×10^{-2}	27.2

Approximately a factor of 30 speedup

Conclusions

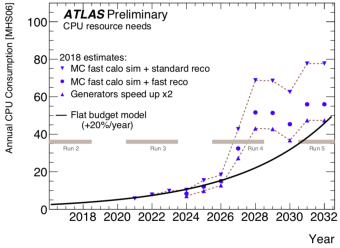
Cost of Event Generation:



- Matrix elements most expensive

Conclusions

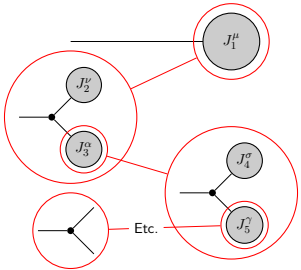
Cost of Event Generation:



- Matrix elements most expensive

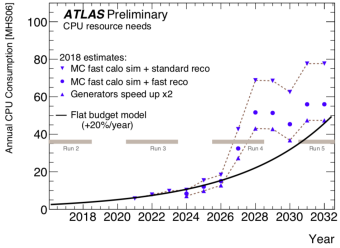
Brends-Giele:

- Optimal generation
- Event generation trivially parallelizable



Conclusions

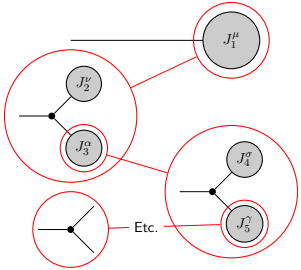
Cost of Event Generation:



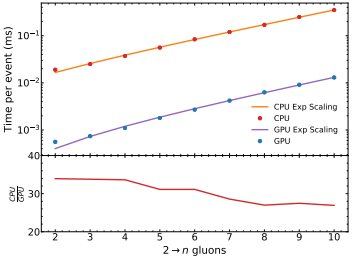
- Matrix elements most expensive

Brends-Giele:

- Optimal generation
- Event generation trivially parallelizable



Preliminary Results:



- Speedup by factor of 30
- Future: Use dedicated high performance GPU