Performance Monitoring

ATLAS contribution

Andrea Dotti (adotti@slac.stanford.edu) ; SD/EPP/Computing

Zachary Marshall; LBL

Foreword

This presentations contains thoughts on the subject from the two authors, based on the experience we had in performing profiling and benchmarking for ATLAS

ATLAS Experience

Obviously CPU profiles are a starting point, but they are not as interesting as one might hope for a typical G4 simulation job:

- The biggest gains we've gotten have come from physics changes, not algorithm changes
- With each compiler jump we seem to get another improvement, which is great.

1: Memory churn

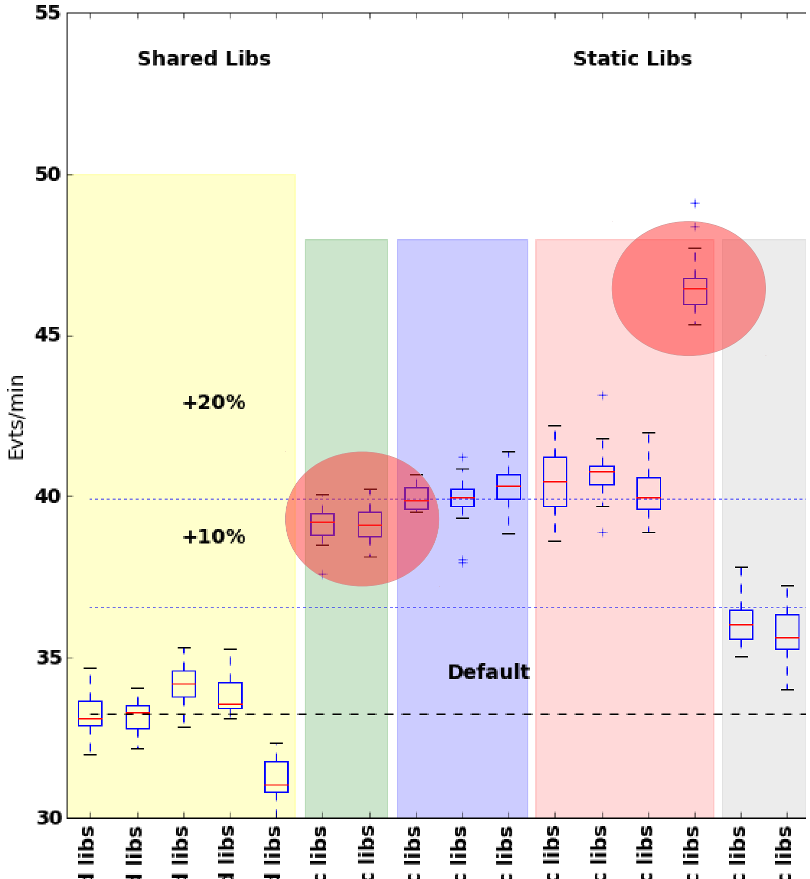
ATLAS experience:

- triggered by need to reduce memory usage (and not CPU speed)
- looked at memory churn (new/delete cycles)
- algorithms and code was reviewed as part of the task
- as a side effect: important speedup obtained for CPU, both as direct result of reduced memory churn and as improving some altorithms

Similar experience by the Geant4 collaboration:

- rewrite of Bertini code, reducing churn, substantially making it faster

2: Compiler options



New compiler versions bring important speedups (several %) **for free**

Static linkage: ~20%
PGO: ~20%

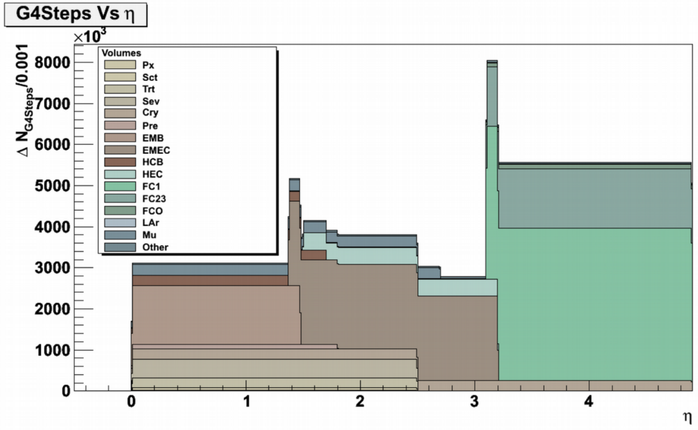
Gains from static linkage and PGO are more important in MT applications (the more threads the more gain, static becomes 30% on KNL)

Shown here: simplified HEP setups with HepExpMT public benchmark (SLAC+LBL activity)

3: Main contribution Looking at physics

Number of G4Steps Units in 10^6	e^-	e^+	Photon	$\pi^\pm$	Proton	Neutron	Other	Total (%)
Px	2.32	0.24	2.37	0.68	0.14	0.30	0.21	0.3%
Sct	2.42	0.37	6.52	0.82	0.20	0.89	0.25	0.6%
Trt	5.37	1.05	40.20	1.82	0.50	5.46	0.53	2.9%
Sev	49.88	7.35	40.63	0.42	0.35	3.82	0.46	5.4%
Cry	48.92	6.48	49.01	0.34	0.40	7.38	0.41	6.0%
Pre	3.99	0.49	10.41	0.09	0.06	1.73	0.03	0.9%
EMB	79.94	11.76	66.26	0.74	0.56	32.59	0.38	10.2%
EMEC	137.13	22.73	114.97	1.34	1.03	63.34	0.65	18.1%
HCB	10.10	0.96	14.88	0.14	0.18	11.46	0.25	2.0%
HEC	17.49	1.67	23.01	0.23	0.30	20.79	0.58	3.4%
FC1	303.56	31.82	204.14	1.79	1.61	64.26	0.80	32.2%
FC23	90.15	13.18	42.67	0.91	0.83	87.99	0.72	12.5%
FCO	10.23	1.46	3.80	0.10	0.12	1.01	0.03	0.9%
LAr	2.50	0.25	2.97	0.06	0.11	0.54	0.05	0.3%
Mu	31.85	4.34	22.09	0.11	0.23	4.25	0.50	3.4%
Other	4.89	0.86	10.13	0.13	0.11	1.05	0.04	0.9%
Total (%)	42.4%	5.6%	34.6%	0.5%	0.4%	16.2%	0.3%	100%

Particles in detectors can teach us a lot about where to look for opportunities



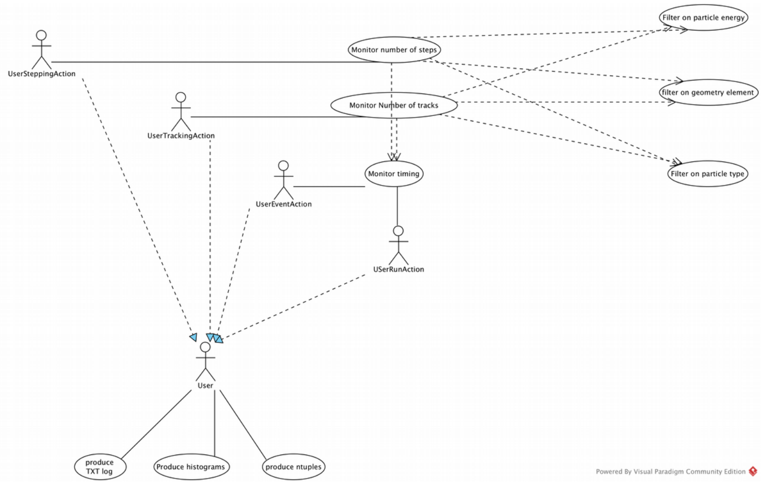
Geant4 plans

Triggered by the discussions with ATLAS

Geant4 is going to provide a general tool to monitor the number of steps/tracks and their time:

1) In a given detector element
2) For tracks of a given energy range
3) For tracks of a given species
4) A combination of the above or any user-defined selection

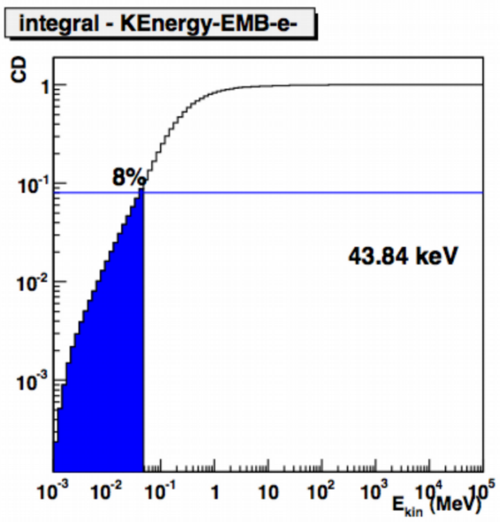
- No user code needed (except for user-defined selections)
- Output: console, histograms, ntuples



Should be available in 10.4.beta (under discussion)

4: Changing Physics

E.m. Option	Cut $e^{\pm}$ (mm)	Cut γ (mm)	CPU perf.	ΔE_{dep}	Notes
Opt0	0.1	0.01	+6%	+0.52%	
Opt0	0.1	0.1	—	—	ATLAS: Reference EM Barrel and End-Cap
Opt0	0.1	0.7	-4%	-0.39%	
Opt0	0.7	0.7	-12%	-0.45%	
Opt2	0.7	0.7	-30%	-1.55%	Apply Cut: EMX (see Section 2.2.2



Some options can bring substantial benefits at the price of different physics output

- They are already available in G4
- Need detailed validation

Fast options in EM: 30% speedup

Production cut threshold studies can bring additional benefits at a smaller physics penalty:

- 8% steps are done by track of <44keV

5: Frozen showers/GFlash

Routinely used in ATLAS:

- Replace particle showering in detectors with pre-generated showers or parametrizations. Additional new ideas being investigated (i.e. use NN to generate hits)
- Full simulations are used to build parametrizations and shower-libs
- Detector specific (strong dependence on geometry): cannot generalize

A comment from Andrea, Geant4 provides GFlash: why is not used?

- In the case of ATLAS accordion calorimeter is so peculiar that requires special studies

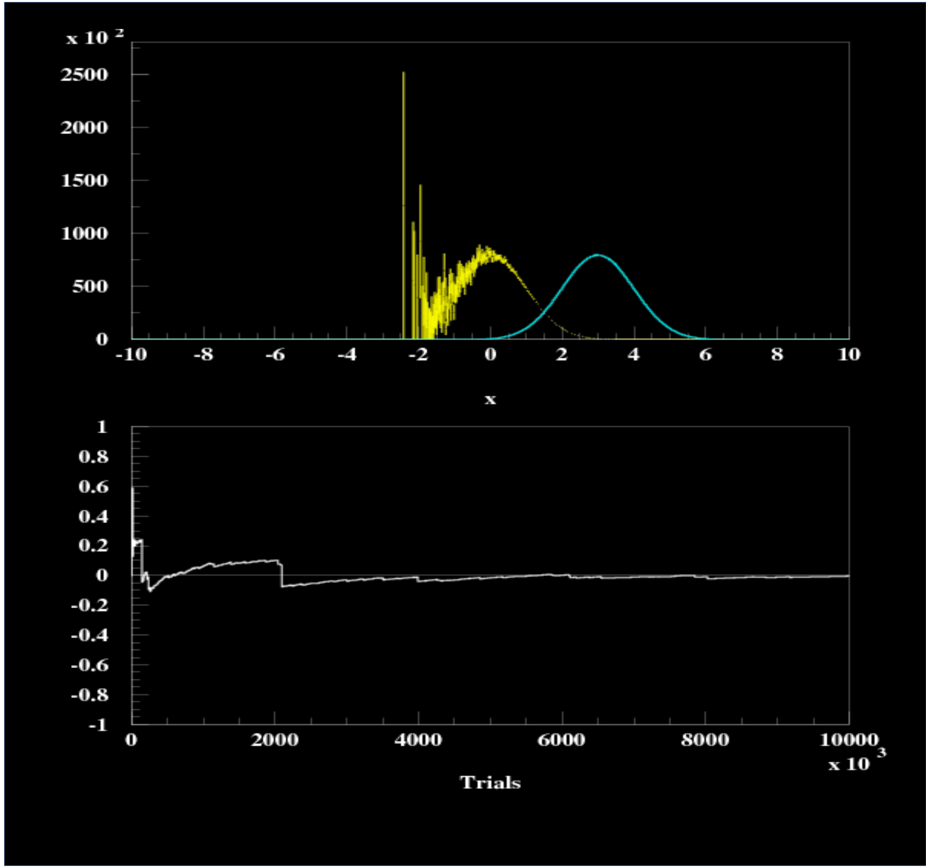
Ideas: Variance Reduction

Standard methods employed in nuclear physics applications and other “rare events” domain

- standard technique with many expert (from non-HEP domain)
- very large gain factors obtained in special applications (x1000 for shielding), not aiming at these factors, but O(10%)

Not routinely used in HEP except the most simple technique (Russian Roulette):

- among N particles of the same “ensemble” (energy, species,geometry), kill N-1, the surviving has a weight of N
- mechanism available in G4 since some time
- requires adapting hits creation to take into account weight
- important: validation and weight monitoring



Conclusion

Reiteration: “The biggest gains we’ve gotten have come from physics changes, not algorithm changes”

Profiling/montioring should be tailored to physics observables

Pure profiling is not very useful if the link to the physics information is not available