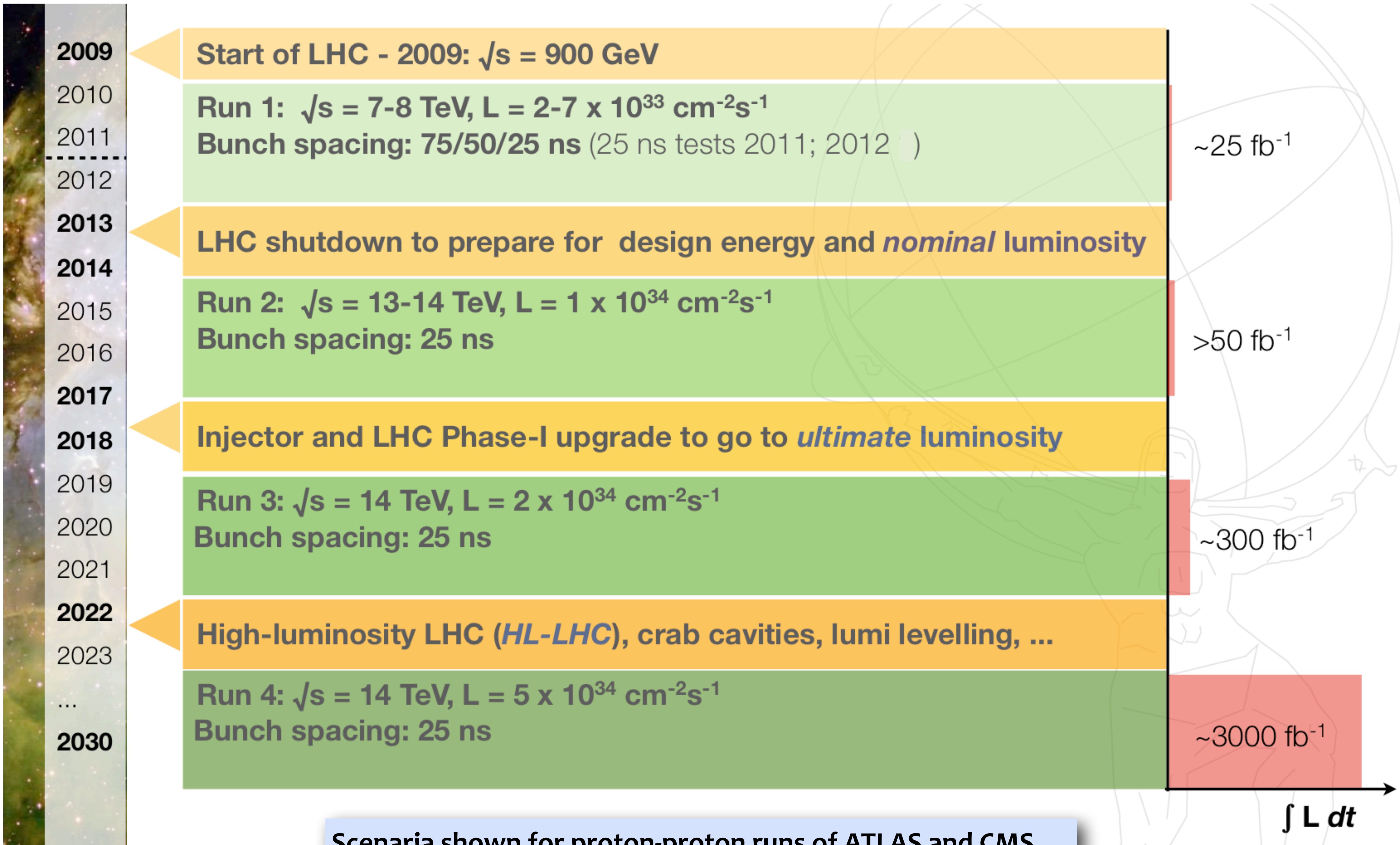
The Geant Vector Prototype an update

Geant 4 Workshop – Sevilla, Spain
September 25, 2013
Federico Carminati

Ack

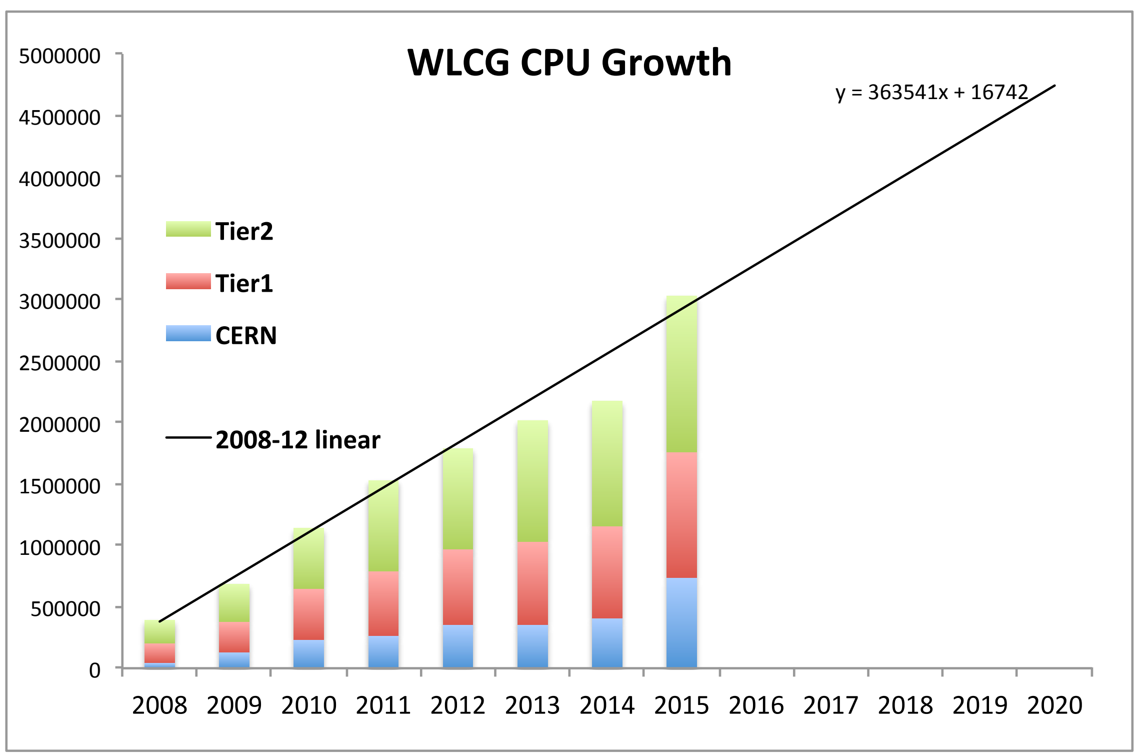
- Thanks to Geant4 for having invited me here

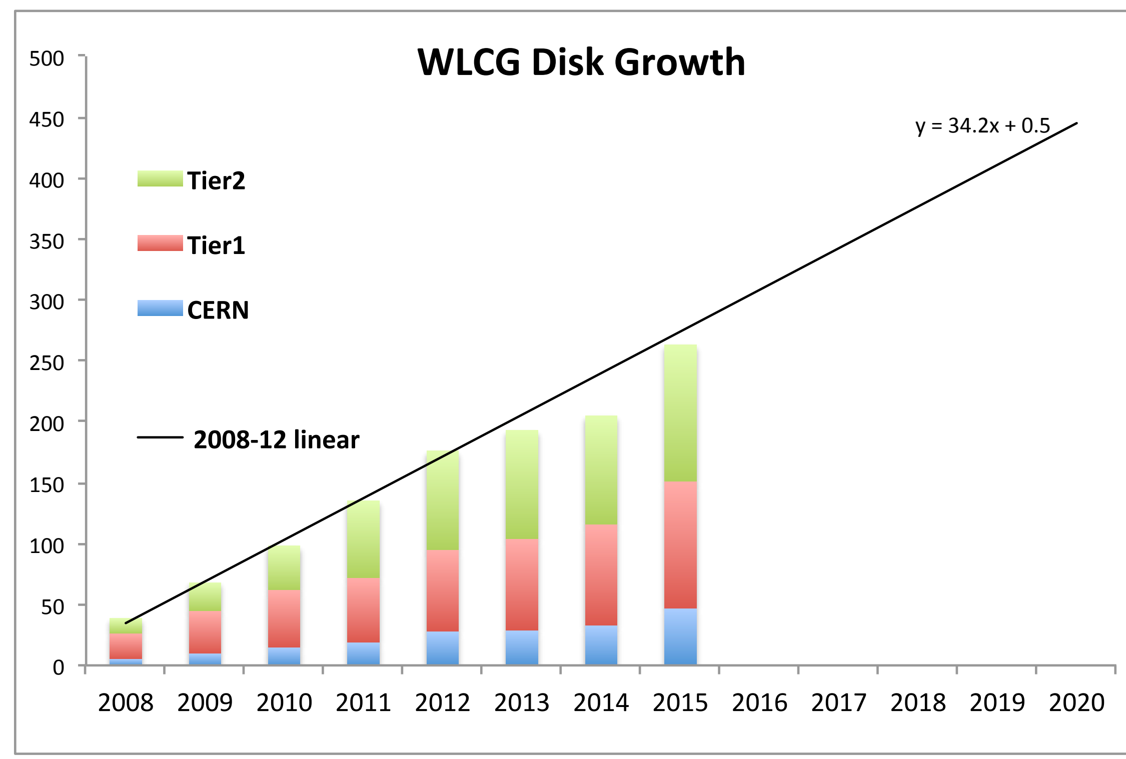
A luminous future for HEP...



Scenaria shown for proton-proton runs of ATLAS and CMS, LHCb and Alice follow different strategies.

Requirements vs resources





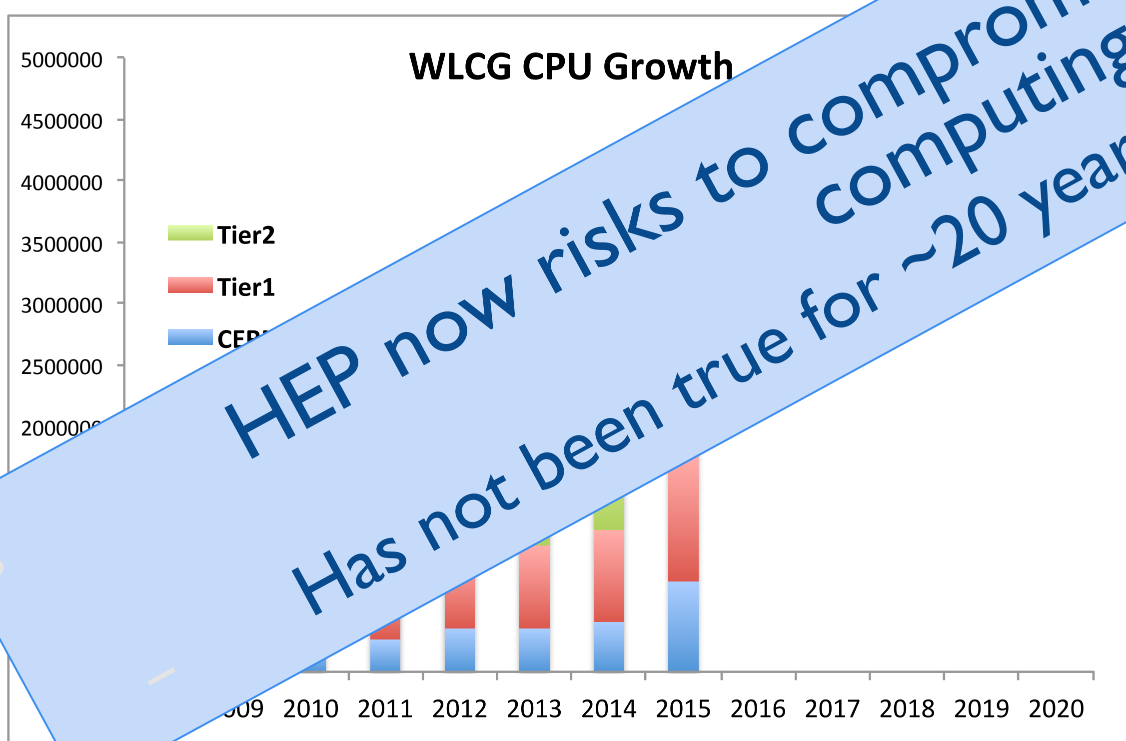
2008-2012 was essentially a linear increase – with ~flat budgets

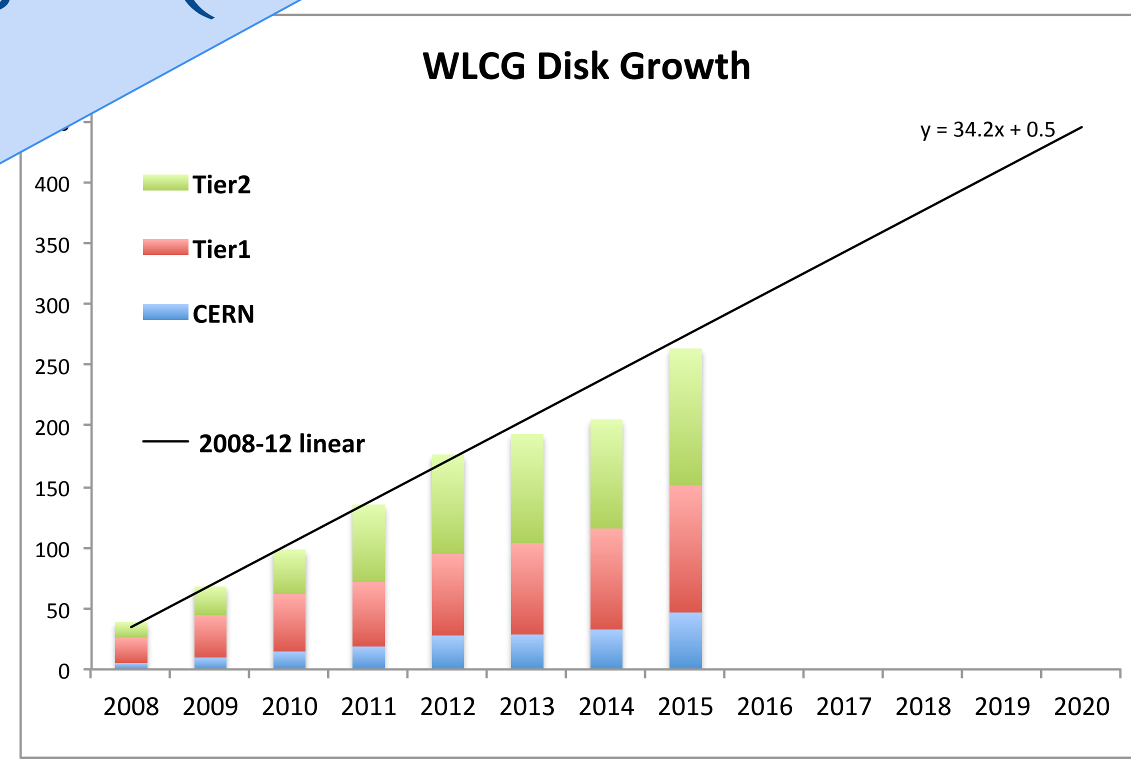
Requirements vs resources

HEP now risks to compromise physics because of lack of computing resources

Has not been true for ~20 years

(from I.Bird's presentation)





2008-2012 was essentially a linear increase – with ~flat budgets

From the 2013 update to the European Strategy for Particle Physics

g. Theory is a strong driver of particle physics and provides essential input to experiments, witness the major role played by theory in the recent discovery of the Higgs boson, from the foundations of the Standard Model to detailed calculations guiding the experimental searches. *Europe should support a diverse, vibrant theoretical physics programme, ranging from abstract to applied topics, in close collaboration with experiments and extending to neighbouring fields such as astroparticle physics and cosmology.* ***Such support should extend also to high- performance computing and software development.***

i. The success of particle physics experiments, such as those required for the high-luminosity LHC, relies on innovative instrumentation, state-of-the-art infrastructures and large-scale data-intensive computing. *Detector R&D programmes should be supported strongly at CERN, national institutes, laboratories and universities. Infrastructure and engineering capabilities for the R&D programme and construction of large detectors, as well as* ***infrastructures for data analysis, data preservation and distributed data-intensive computing should be maintained and further developed.***

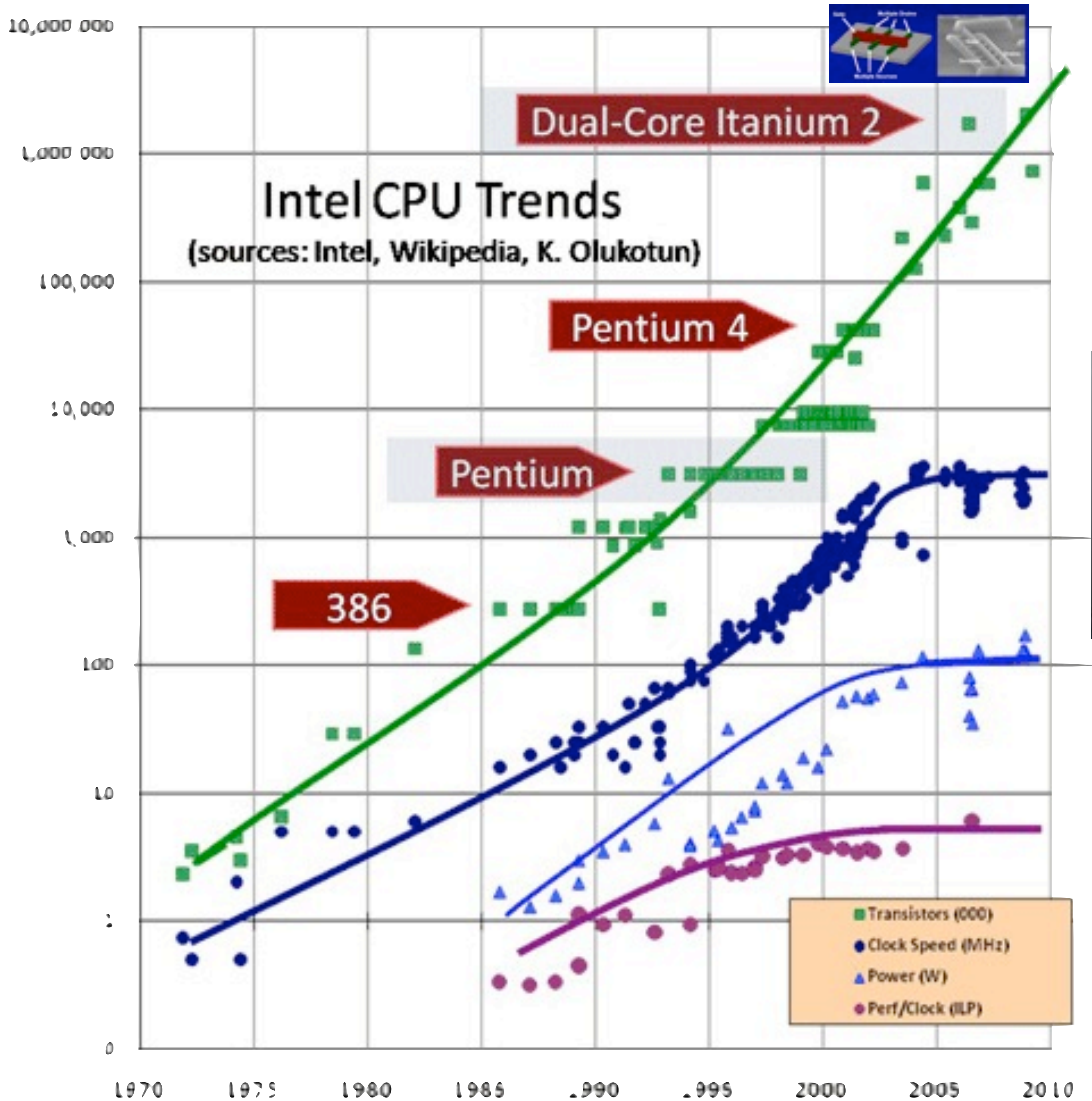
From the 2013 update to the European Strategy for Particle Physics

g. Theory is a strong driver of particle physics and provides essential input to experiments, witness the major role played by theory in the recent discovery of the Higgs boson, from the foundations of the Standard Model to detailed calculations guiding the experimental searches. *Europe should support a diverse, vibrant theoretical physics programme, ranging from abstract to applied topics, in close collaboration with experiments and extending to neighbouring fields such as astroparticle physics and cosmology.* ***Such support should extend also to high- performance computing and software development.***

i. The success of particle physics experiments, such as those required for the high-luminosity LHC, relies on innovative instrumentation, state-of-the-art infrastructures and large-scale data-intensive computing. *Detector R&D programmes should be supported strongly at CERN, national institutes, laboratories and universities. Infrastructure and engineering capabilities for the R&D programme and* construction of large detectors, as well as ***infrastructures for data analysis, data preservation and distributed data-intensive computing should be maintained and further developed.***

High Performance Computing

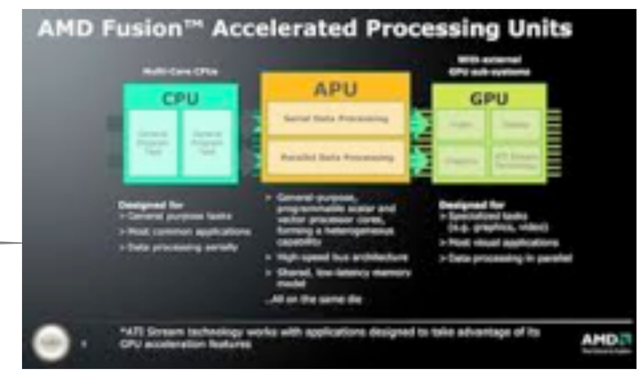
Trends…



While transistor density has been following Moore's law, frequency and power consumption did not…

Tesla k10 GPU (NVIDIA)

Intel Many Integrated Core Architecture



AMD "on board" GPU for fine grain, low latency GPU applications

Texas Instruments DSPs

ARM CPUs

ATOM CPUs

The Eight dimensions

- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

The Eight dimensions

- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Possibly running different jobs as we do now is the best solution

The Eight dimensions

- The “dimensions of performance”
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Multi-core, Multi-socket → Gain in memory footprint and time-to-solution but not in throughput

Multi-node → Possibly running different jobs as we do now is the best solution

The Eight dimensions

- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Very little gain to be expected and no action to be taken

Gain in memory footprint and time-to-solution but not in throughput

Possibly running different jobs as we do now is the best solution

The Eight dimensions

- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Micro-parallelism: gain in throughput and in time-to-solution

Very little gain to be expected and no action to be taken

Gain in memory footprint and time-to-solution but not in throughput

Possibly running different jobs as we do now is the best solution

The Eight dimensions

- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Micro-parallelism: gain in throughput and in time-to-solution

Very little gain to be expected and no action to be taken

Gain in memory footprint and time-to-solution but not in throughput

Possibly running different jobs as we do now is the best solution

Expected limits on performance scaling			
	SIMD	ILP	HW THREADS
MAX	8	4	1.35
INDUSTRY	6	1.57	1.25
HEP	1	0.8	1.25

Expected limits on performance scaling (multiplied)			
	SIMD	ILP	HW THREADS
MAX	8	32	43.2
INDUSTRY	6	9.43	11.79
HEP	1	0.8	1

The Eight dimensions

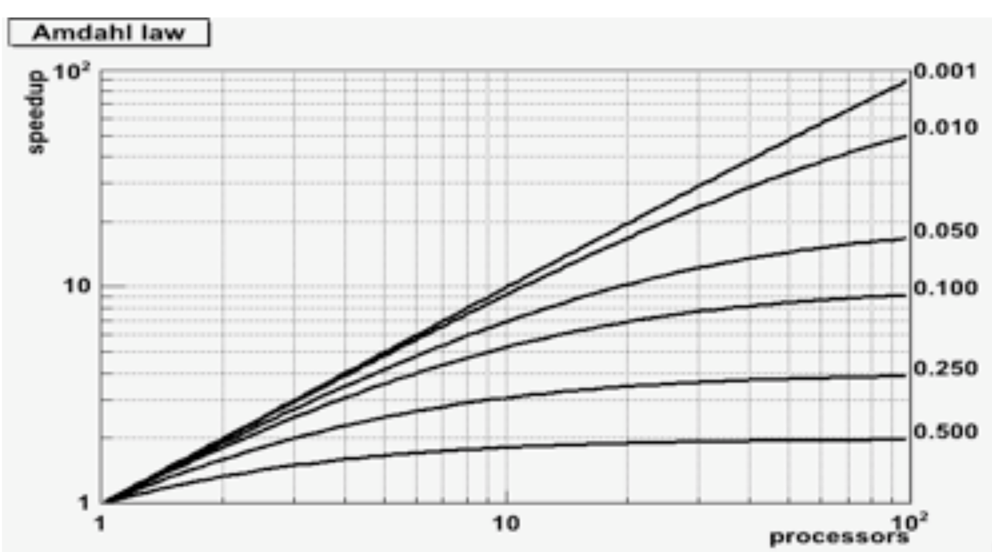
- The "dimensions of performance"
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node

Micro-parallelism: gain in throughput and in time-to-solution

Very little gain to be expected and no action to be taken

Gain in memory footprint and time-to-solution but not in throughput

Possibly running different jobs as we do now is the best solution



OpenLab@CHEP12

The concurrency forum

- A Concurrency Forum has been established in 2011 to
  - Share knowledge amongst the whole community
  - Form a consensus on the best concurrent programming models and on technology choices
  - Develop and adopt common solutions
- Bi-weekly meeting with an active and growing participation of laboratories and experiment
- An R&D programme of work on a number of demonstrators to explore technology
  - 16 projects have been launched with deliverables and goals

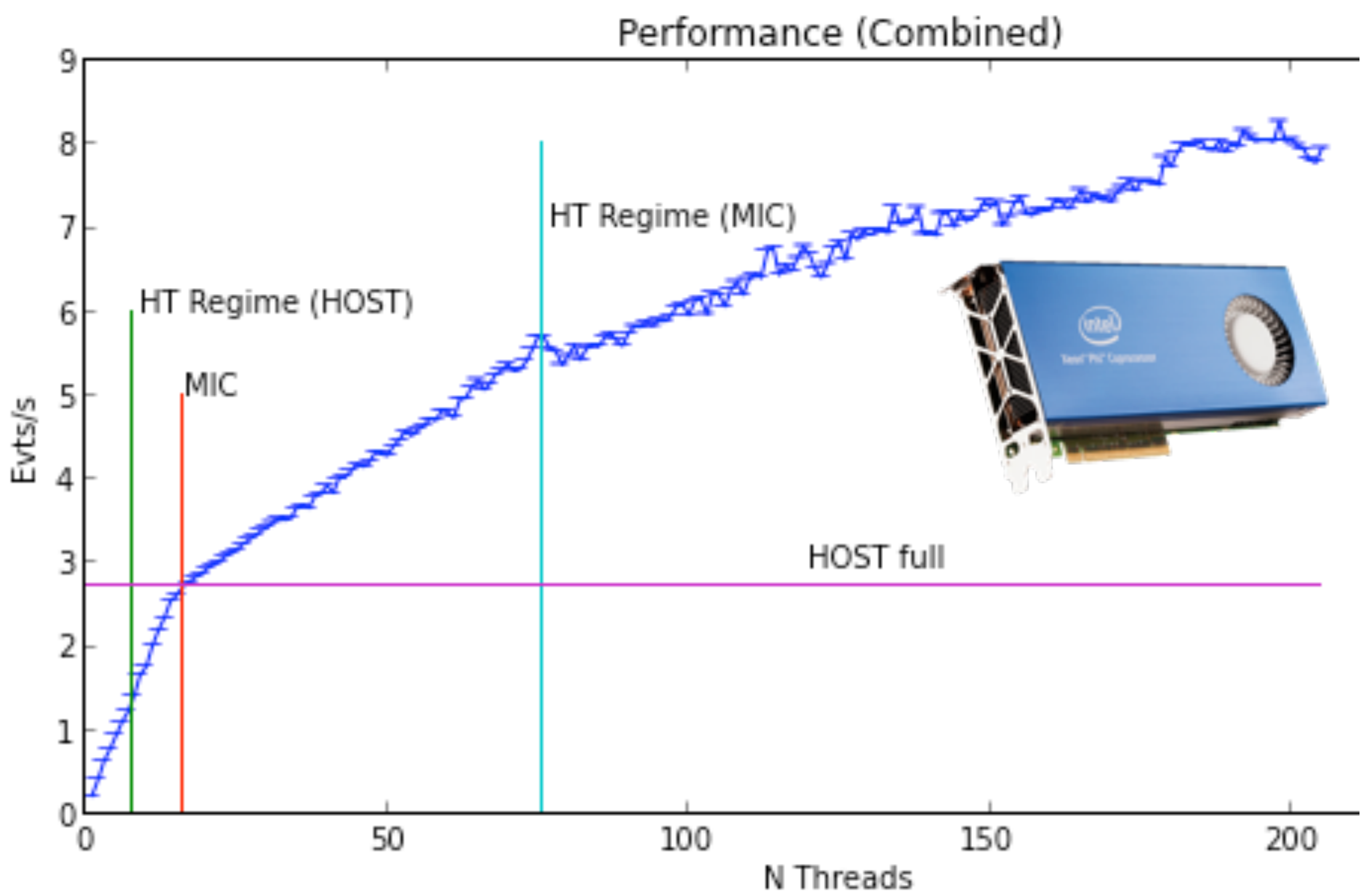
- http://concurrency.web.cern.ch

A TechPark

- An important element of this is to have (diverse and advanced!) hardware & software to test and the right connection to the companies' engineers
  - GPUs, ARMs, compilers, debuggers, profilers
- The model pioneered by CERN openlab is a good one, and we are building on it
- It is open to the community working with us and complementary to similar facilities elsewhere
- It should NOT be demand driven but technology driven, to generate and motivate demands from the user community
  - Planning & procurement for this is under way

Geant4 Multi-threading

- Parallelism at level of event - for simple migration of experiments' "user" code
  - Part of next Geant4 10.0 production release (Dec 2013)



Preliminary, Courtesy of A.Dotti, SLAC

- Demonstrates
  - Linear scaling of throughput with number of threads (up to 40 CPU threads or 200 on Xeon Phi),
  - Large savings in memory: 40MB extra memory per thread (working to reduce it further.)
- Extension of parallelism level possible with deeper changes in "user" code:
  - Tests underway for primary track parallelism by ATLAS (trial integration with ISF)
  - Plan to investigate *track-level parallelism*, to evaluate potential for efficiency improvements.

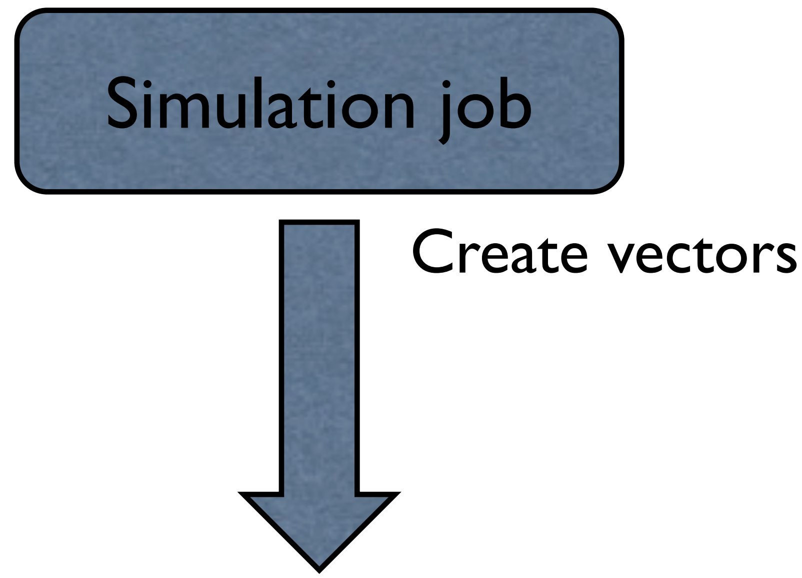
A fresh look at the Simulation

- The most CPU-bound and time-consuming application in HEP with large room for speed-up
  - Largely experiment independent
  - Precision depends on (the inverse of the sqrt of) the number of events
- Grand strategy
  - Explore from a performance perspective, no constraints from existing code
  - Expose the parallelism at all levels, from coarse granularity to micro-parallelism at the algorithm level
  - Integrate from the beginning slow and fast simulation in order to optimise both in the same framework
  - Explore if-and-how existing physics code (GEANT4) can be optimised in this framework Improvements (in geometry for instance) and techniques are expected to feed back into reconstruction

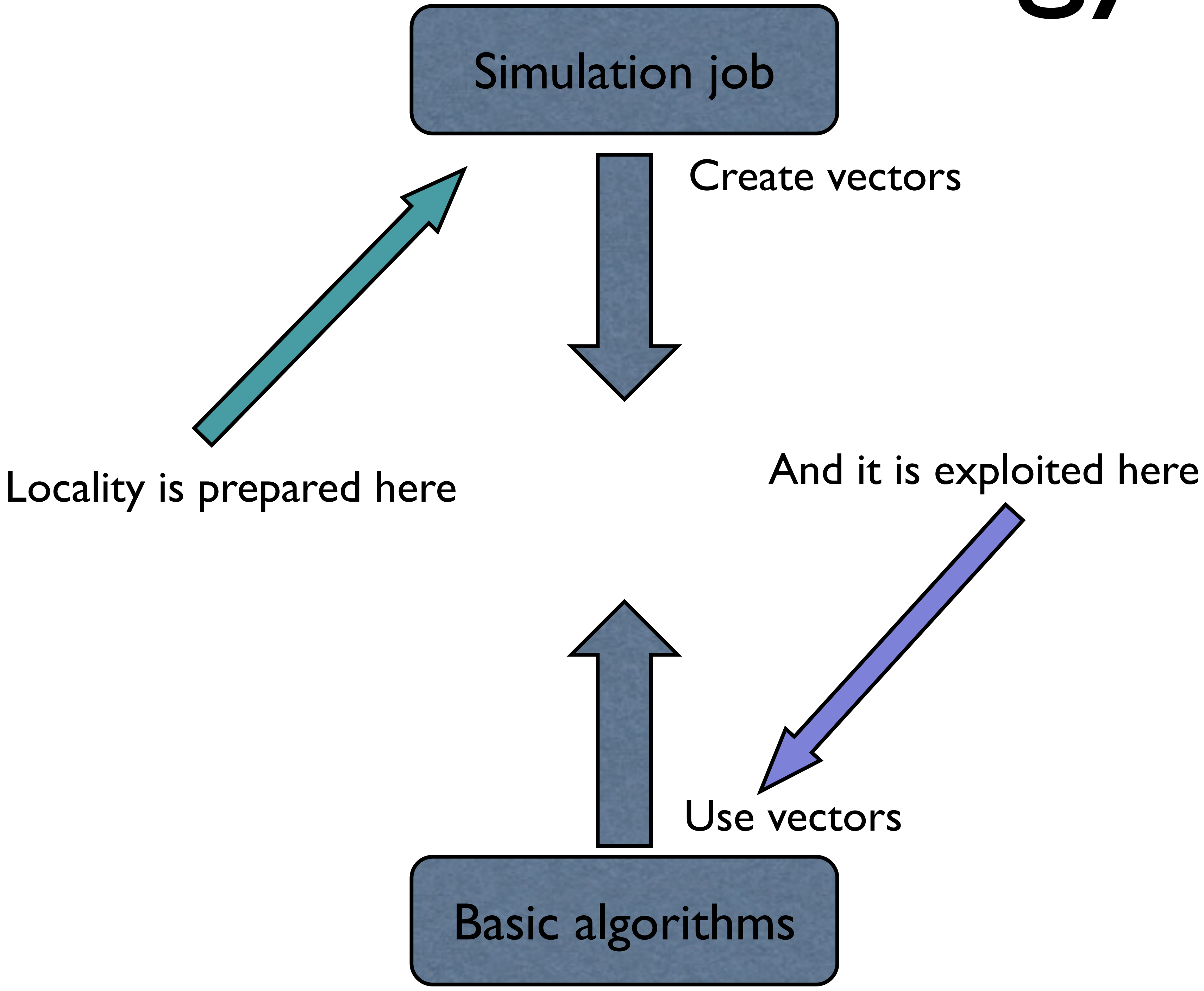
FNAL Geant GPU Prototype

- CERN-FNAL collaboration to
  - Develop and study the performance of various strategies and algorithms that will enable Geant4 to use multiple computational threads
- See Soon's presentation for latest status
- Kernel scheduling and CPU/GPU communication
  - Need to run the GPU Prototype as part of a full vectorized prototype to enable a end-to-end testing.
  - Implemented a broker than can schedule the processing of tracks on the GPU with maximum flexibility
- Focus has been on NVidia hardware, increasing collaboration with Geant Vector Prototype

Grand strategy

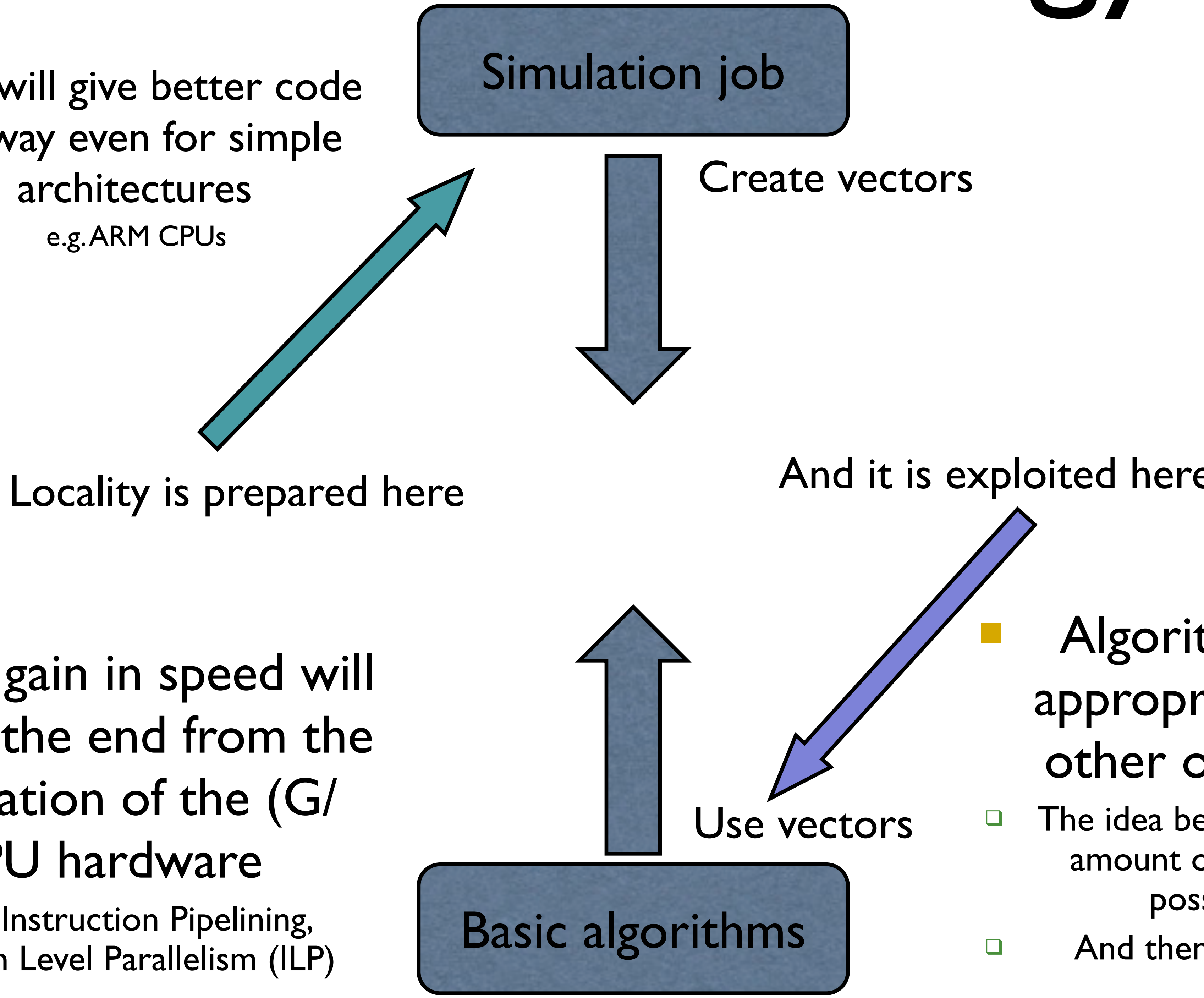


Grand strategy



Grand strategy

- This will give better code anyway even for simple architectures
  - e.g. ARM CPUs

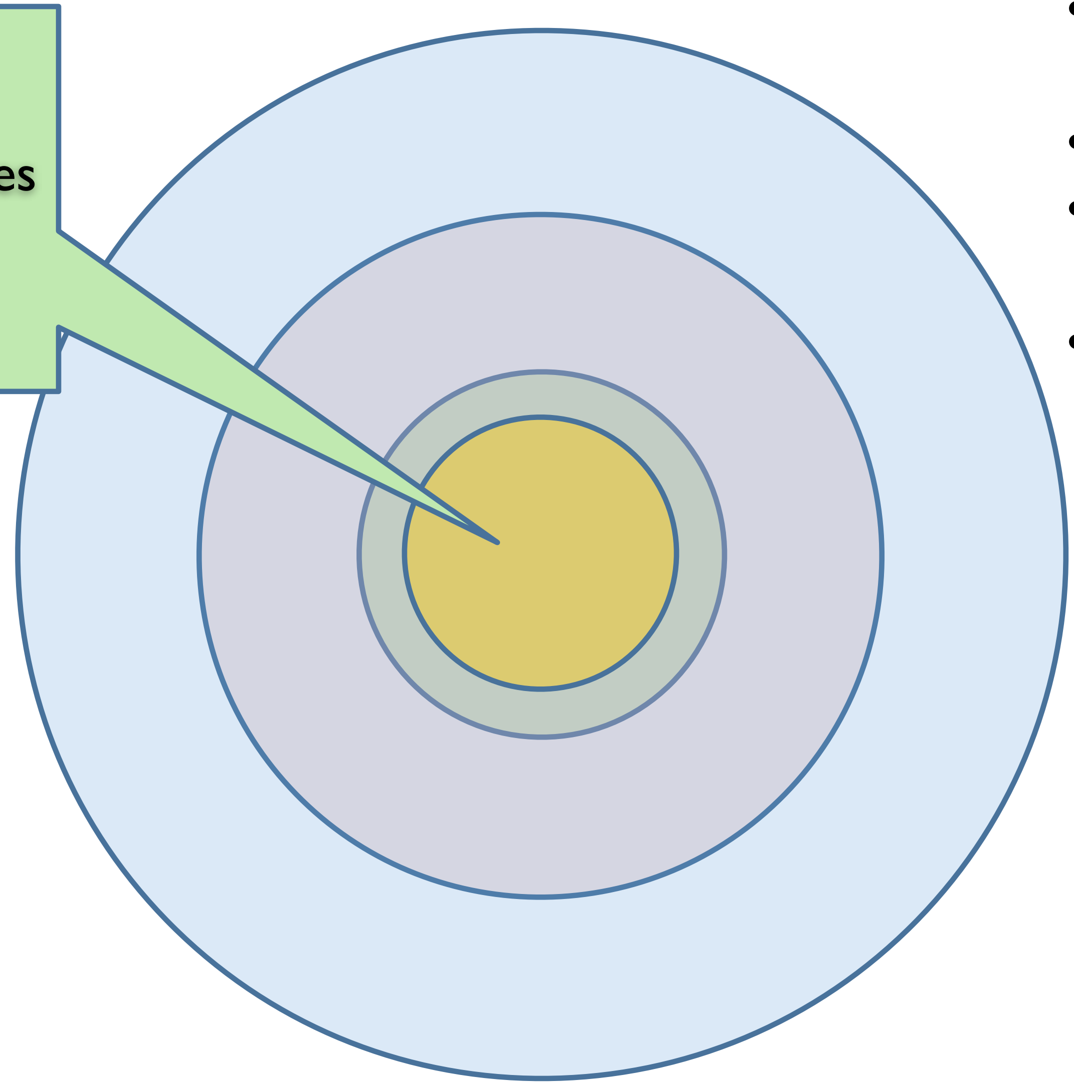


- The real gain in speed will come at the end from the exploitation of the (G/C)PU hardware
  - Vectors, Instruction Pipelining, Instruction Level Parallelism (ILP)

- Algorithms will be more appropriate for one or the other of these techniques
  - The idea being to expose the maximum amount of parallelism at the lowest possible granularity level
  - And then explore the optimisation opportunities

Classical particle transport

- Geometry navigation (local)
- Material – X-section tables
- Particle type - physics processes

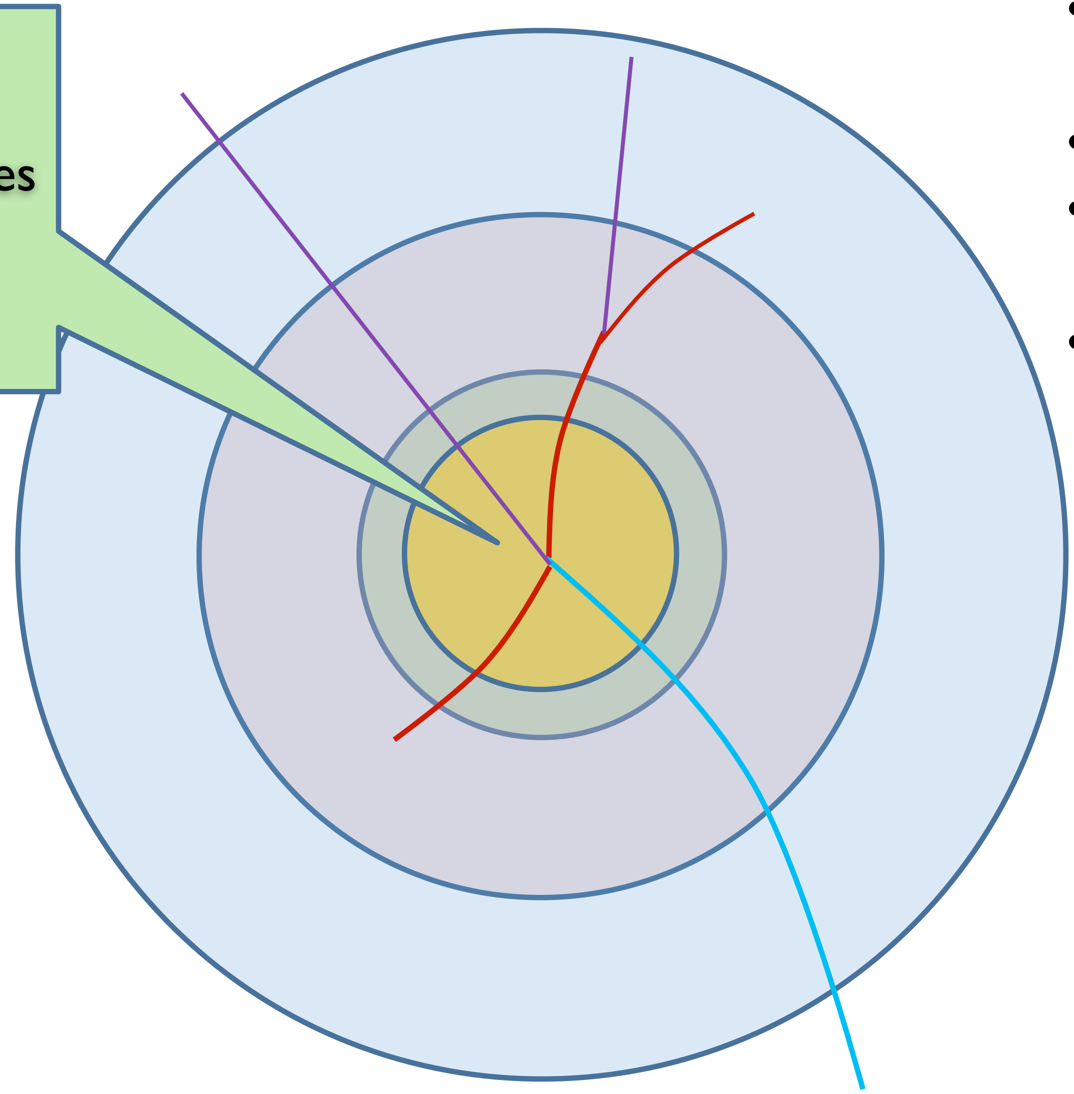


- Navigating very large data structures
- No locality
- OO abused: very deep instruction stack
- Cache misses

- Event- or event track-level parallelism will better use resources but won't improve these points

Classical particle transport

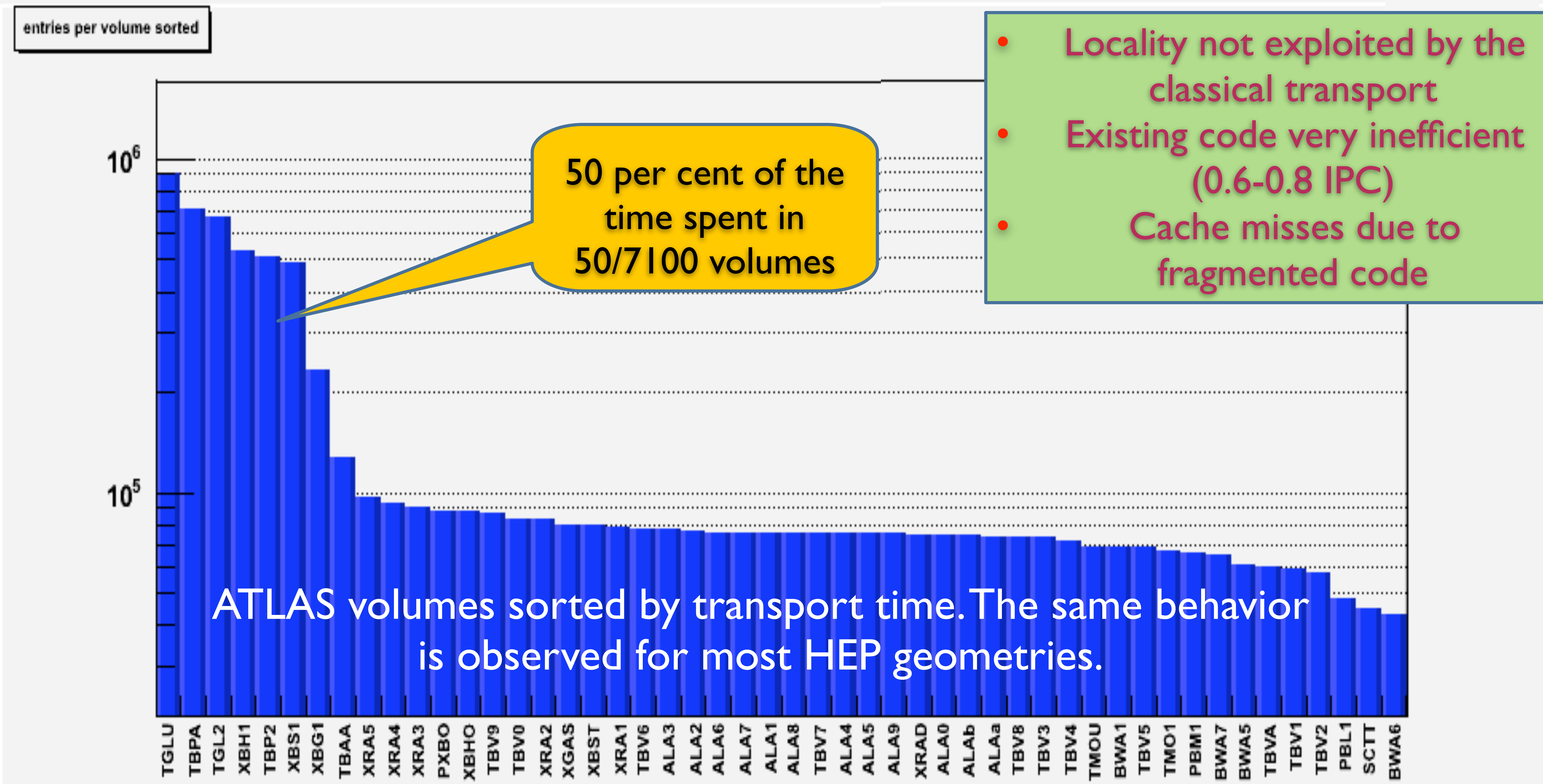
- Geometry navigation (local)
- Material – X-section tables
- Particle type - physics processes



- Navigating very large data structures
- No locality
- OO abused: very deep instruction stack
- Cache misses

- Event- or event track-level parallelism will better use resources but won't improve these points

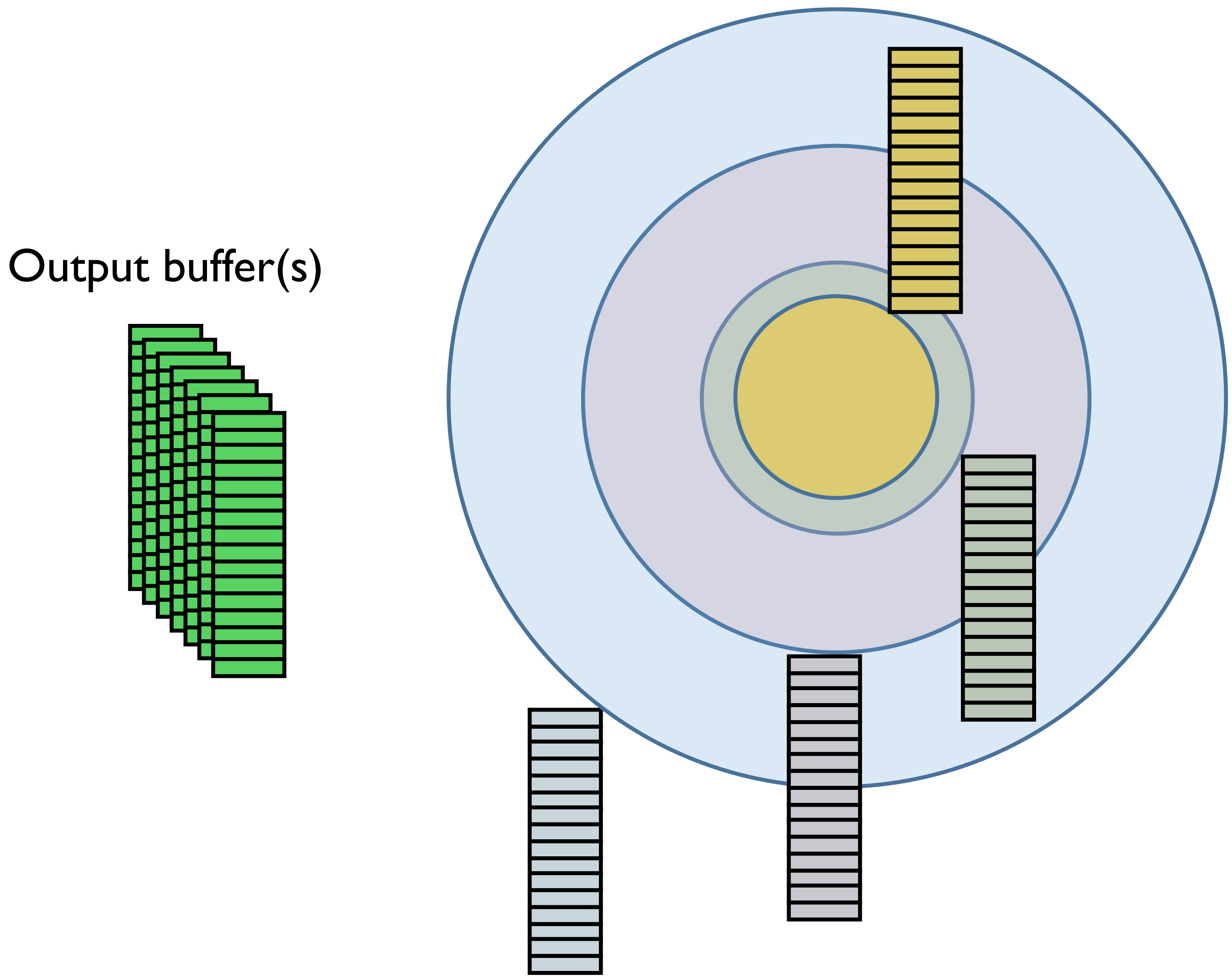
HEP transport is mostly local !



- Locality not exploited by the classical transport
- Existing code very inefficient (0.6-0.8 IPC)
- Cache misses due to fragmented code

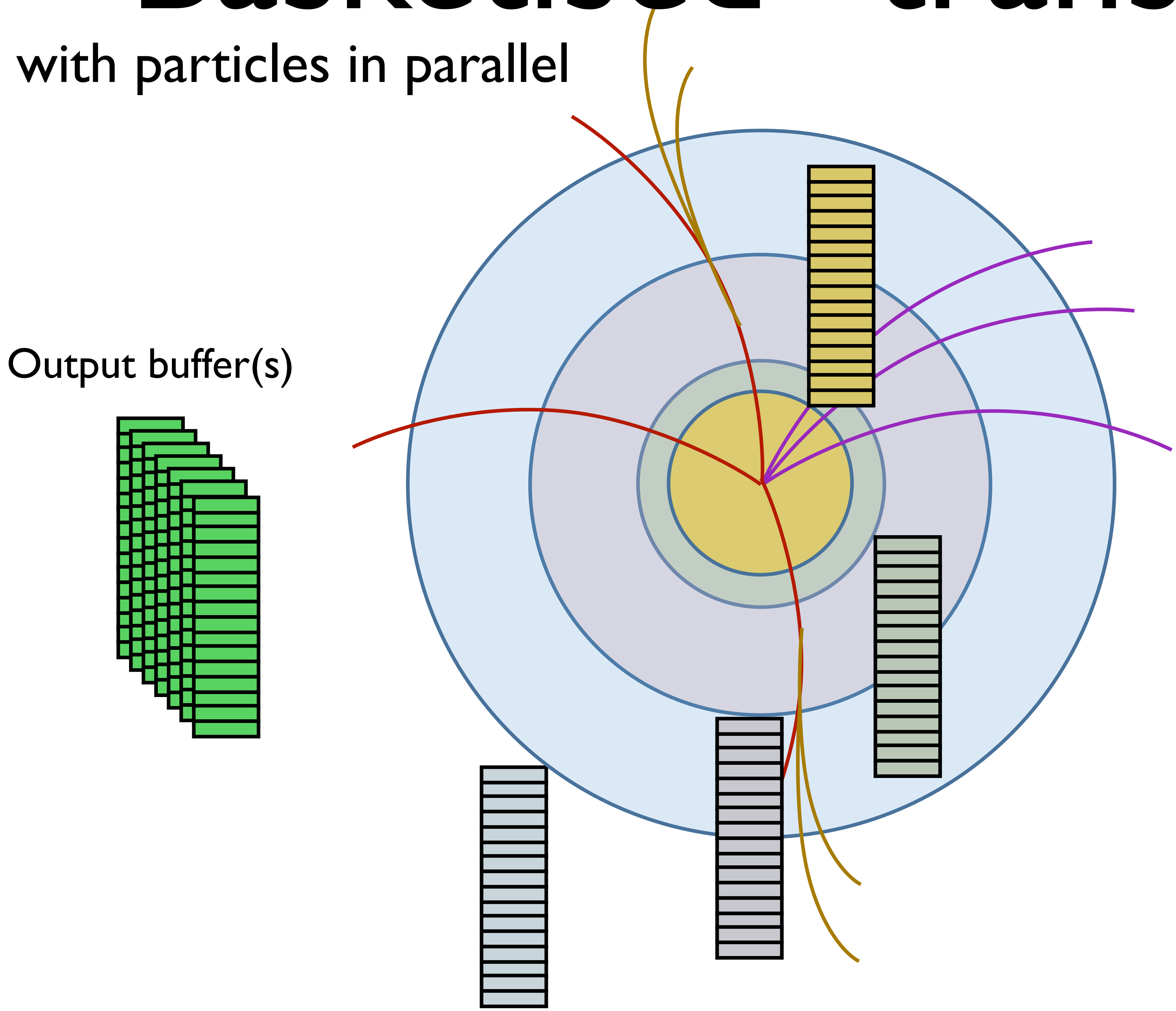
“Basketised” transport

Deal with particles in parallel



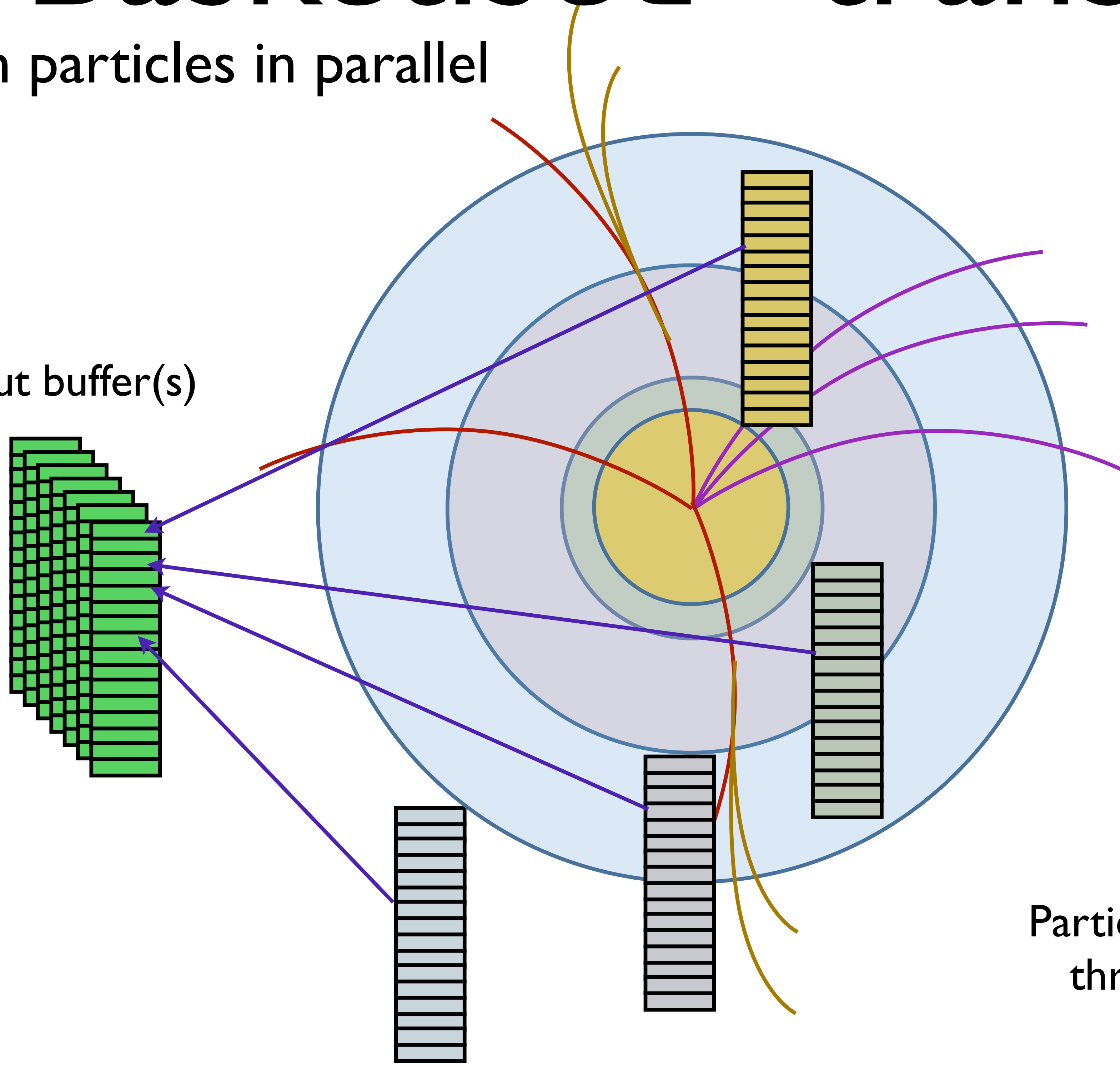
“Basketised” transport

Deal with particles in parallel



“Basketised” transport

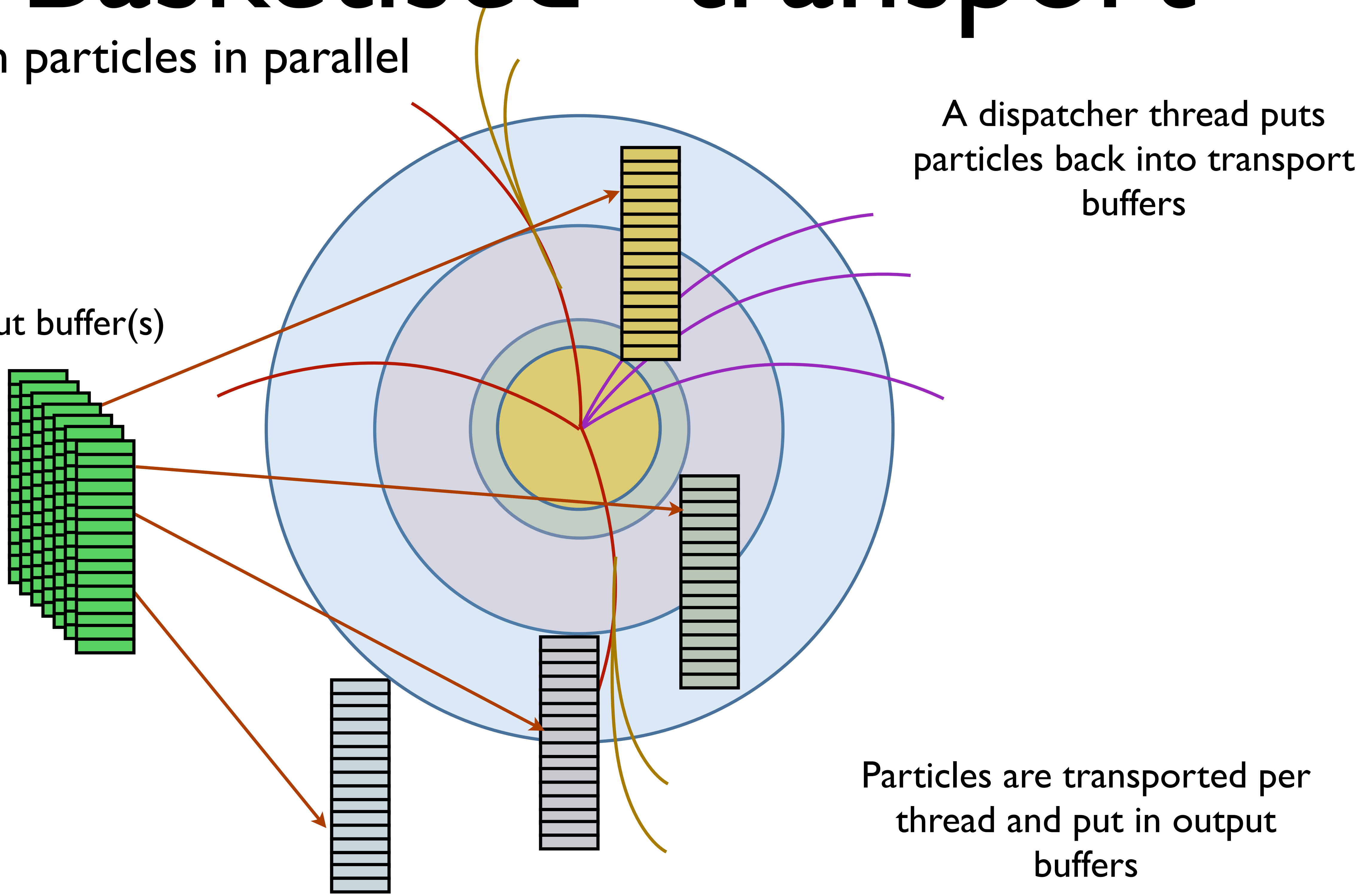
Deal with particles in parallel



Particles are transported per thread and put in output buffers

“Basketised” transport

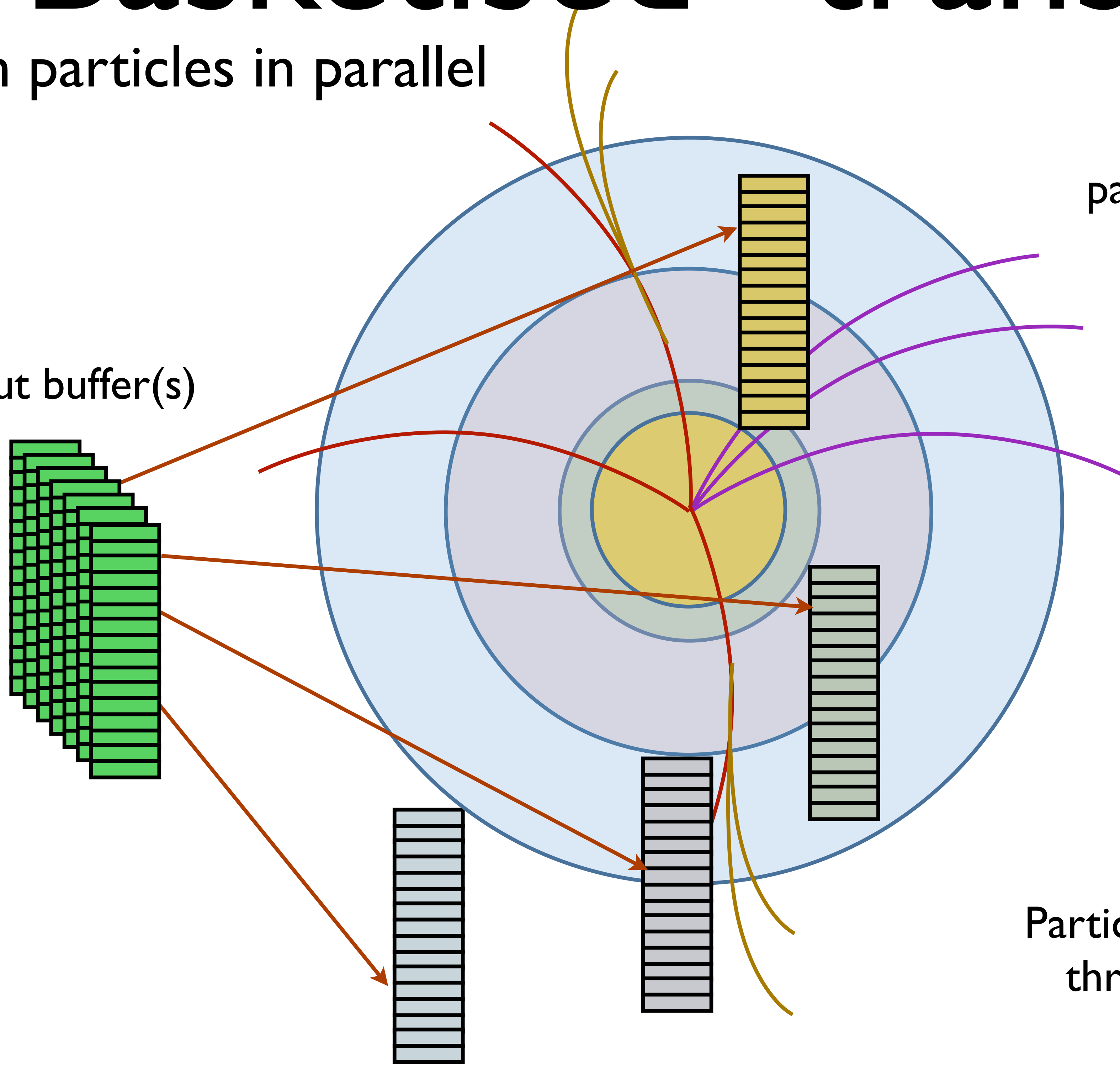
Deal with particles in parallel



“Basketised” transport

Deal with particles in parallel

Output buffer(s)



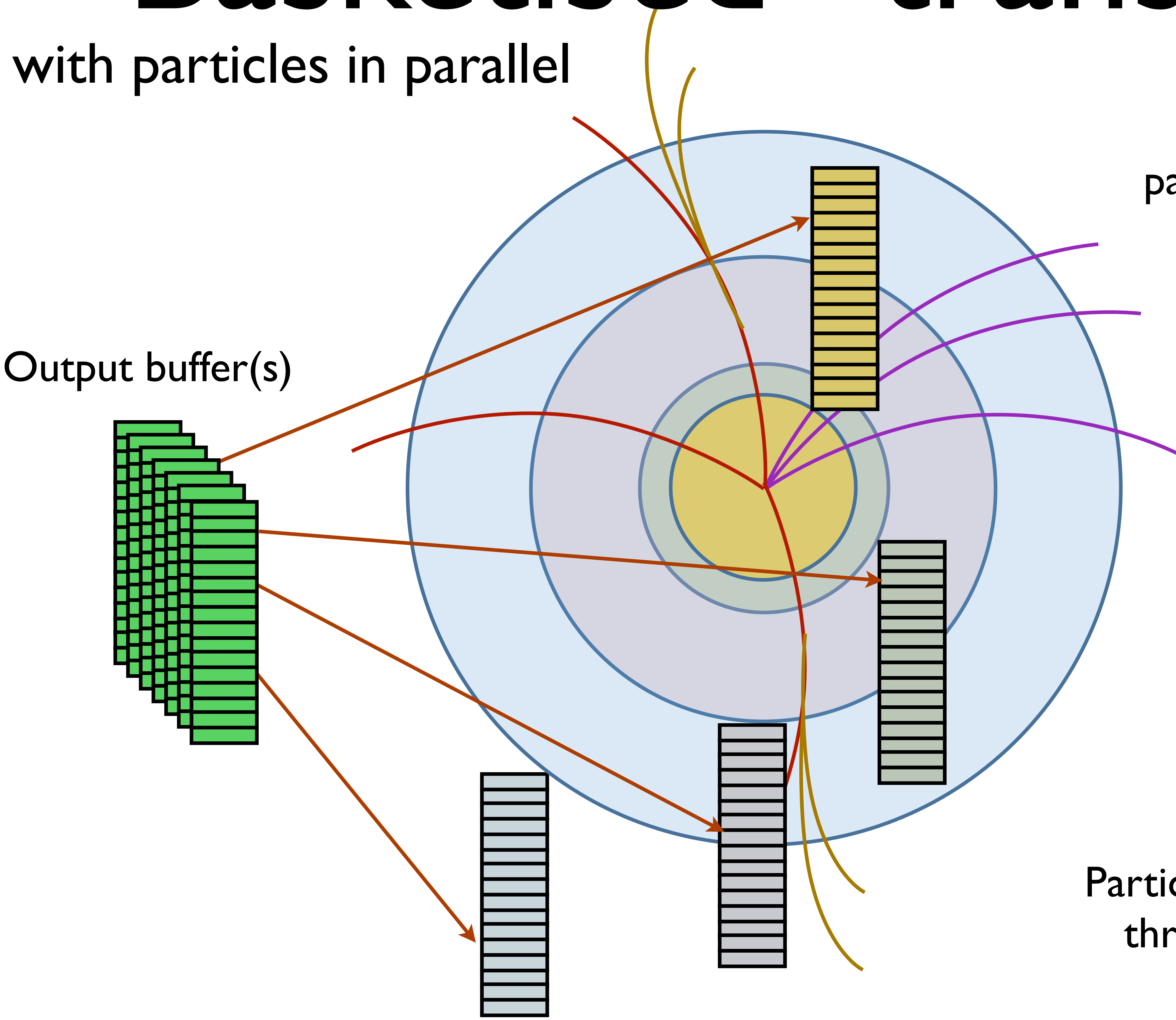
A dispatcher thread puts particles back into transport buffers

Everything happens asynchronously and in parallel

Particles are transported per thread and put in output buffers

“Basketised” transport

Deal with particles in parallel



A dispatcher thread puts particles back into transport buffers

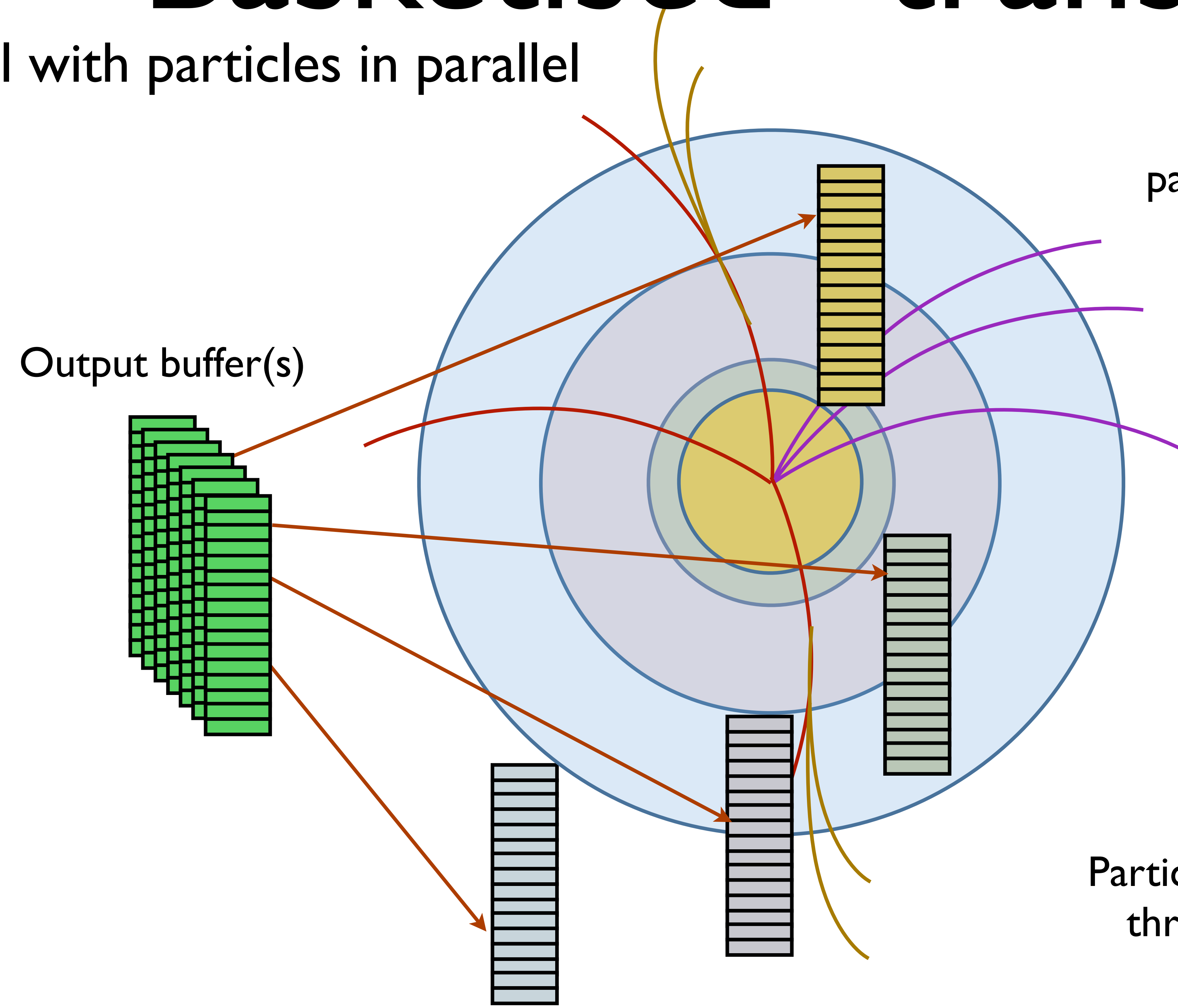
Everything happens asynchronously and in parallel

The challenge is to minimise locks

Particles are transported per thread and put in output buffers

“Basketised” transport

Deal with particles in parallel



A dispatcher thread puts particles back into transport buffers

Everything happens asynchronously and in parallel

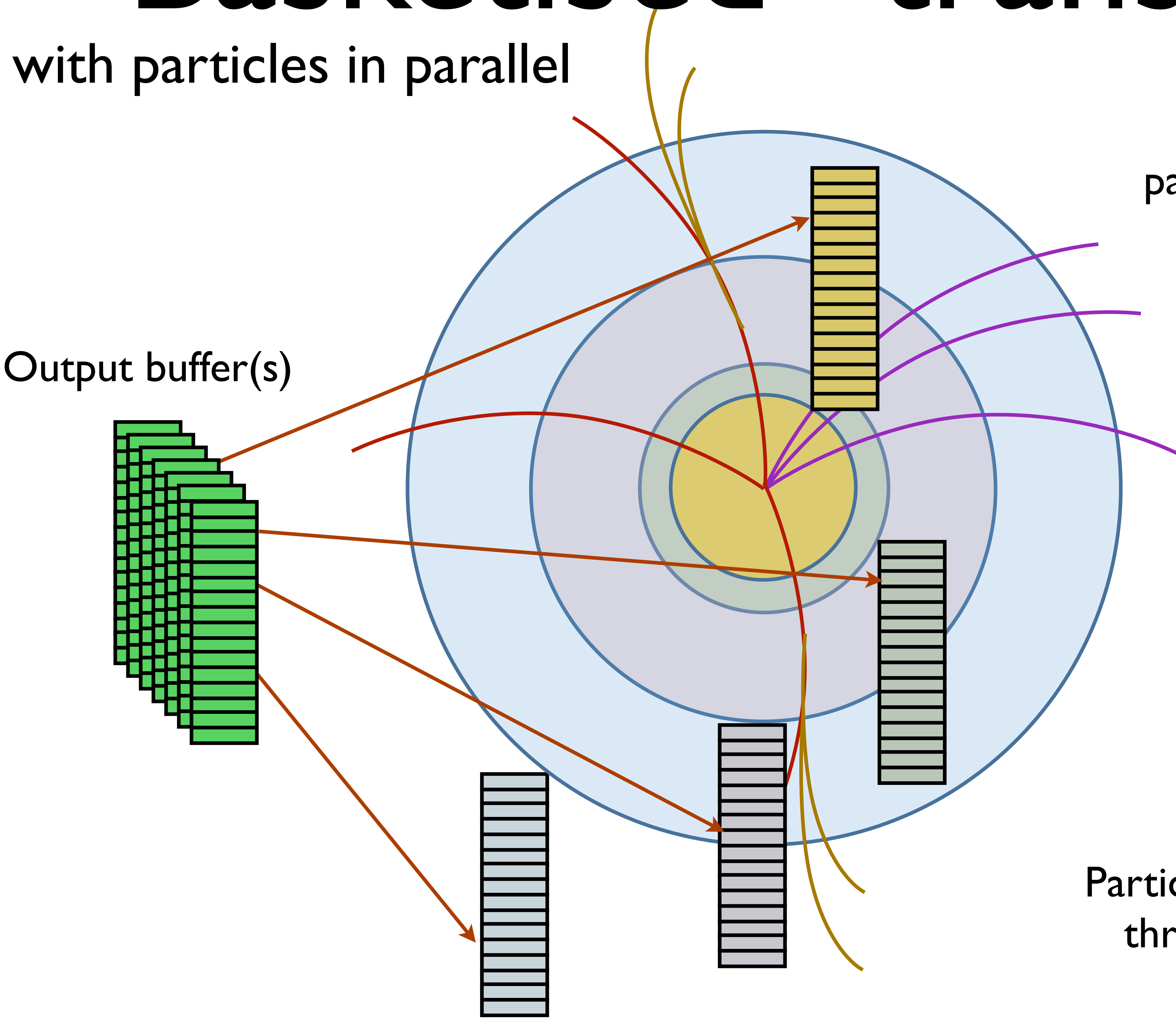
The challenge is to minimise locks

Keep long vectors

Particles are transported per thread and put in output buffers

“Basketised” transport

Deal with particles in parallel



A dispatcher thread puts particles back into transport buffers

Everything happens asynchronously and in parallel

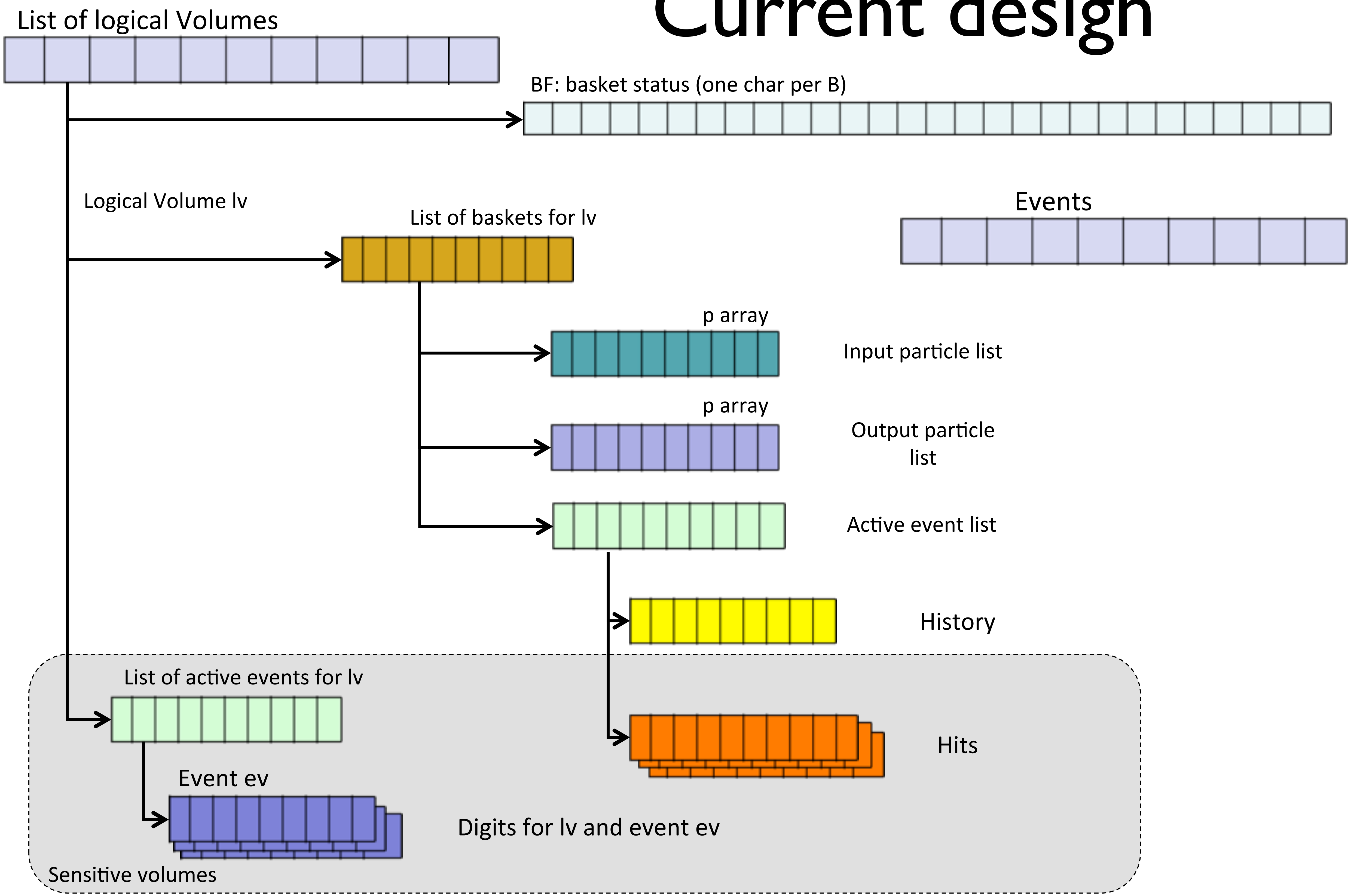
The challenge is to minimise locks

Keep long vectors

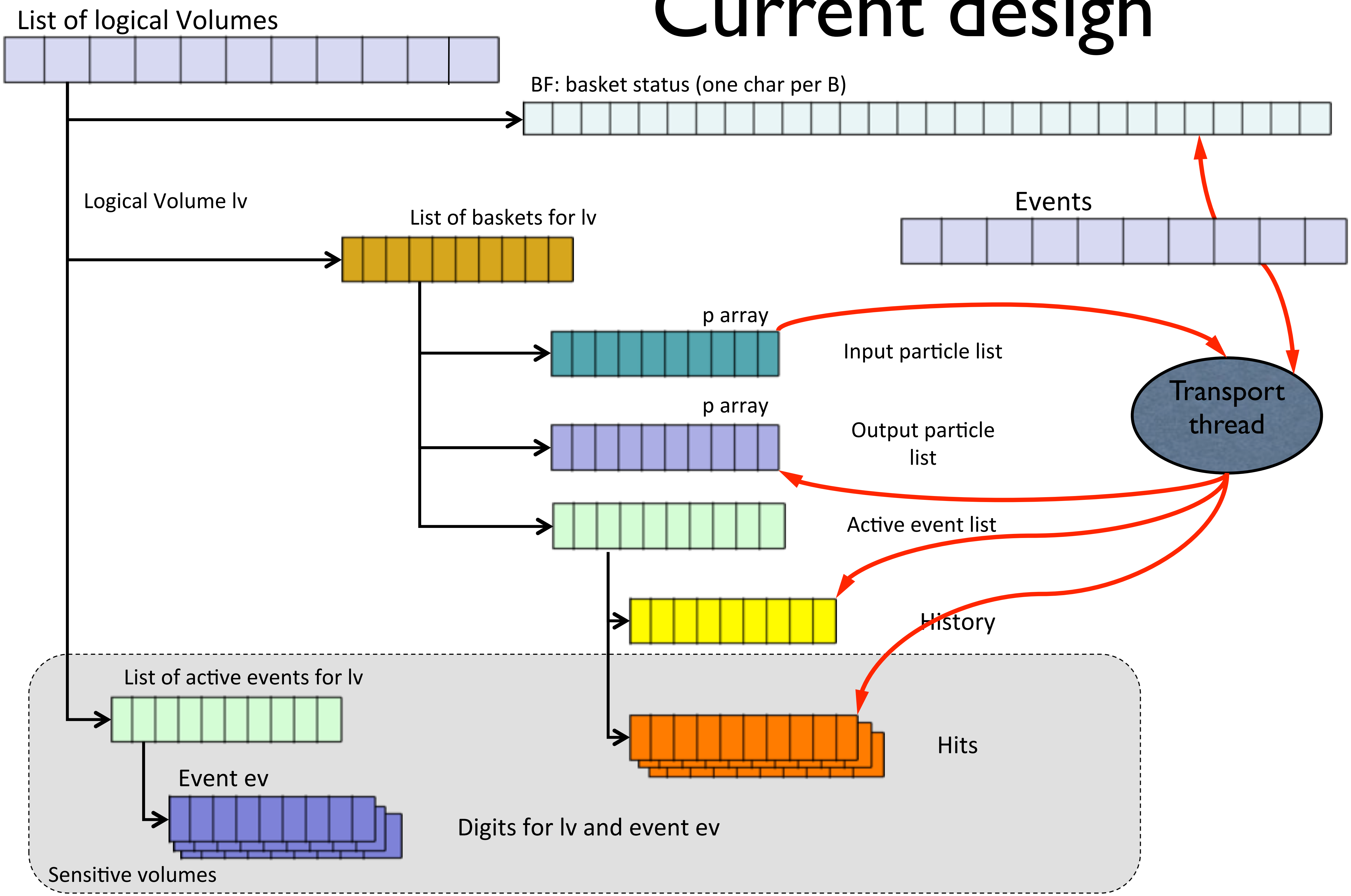
Avoid memory explosion

Particles are transported per thread and put in output buffers

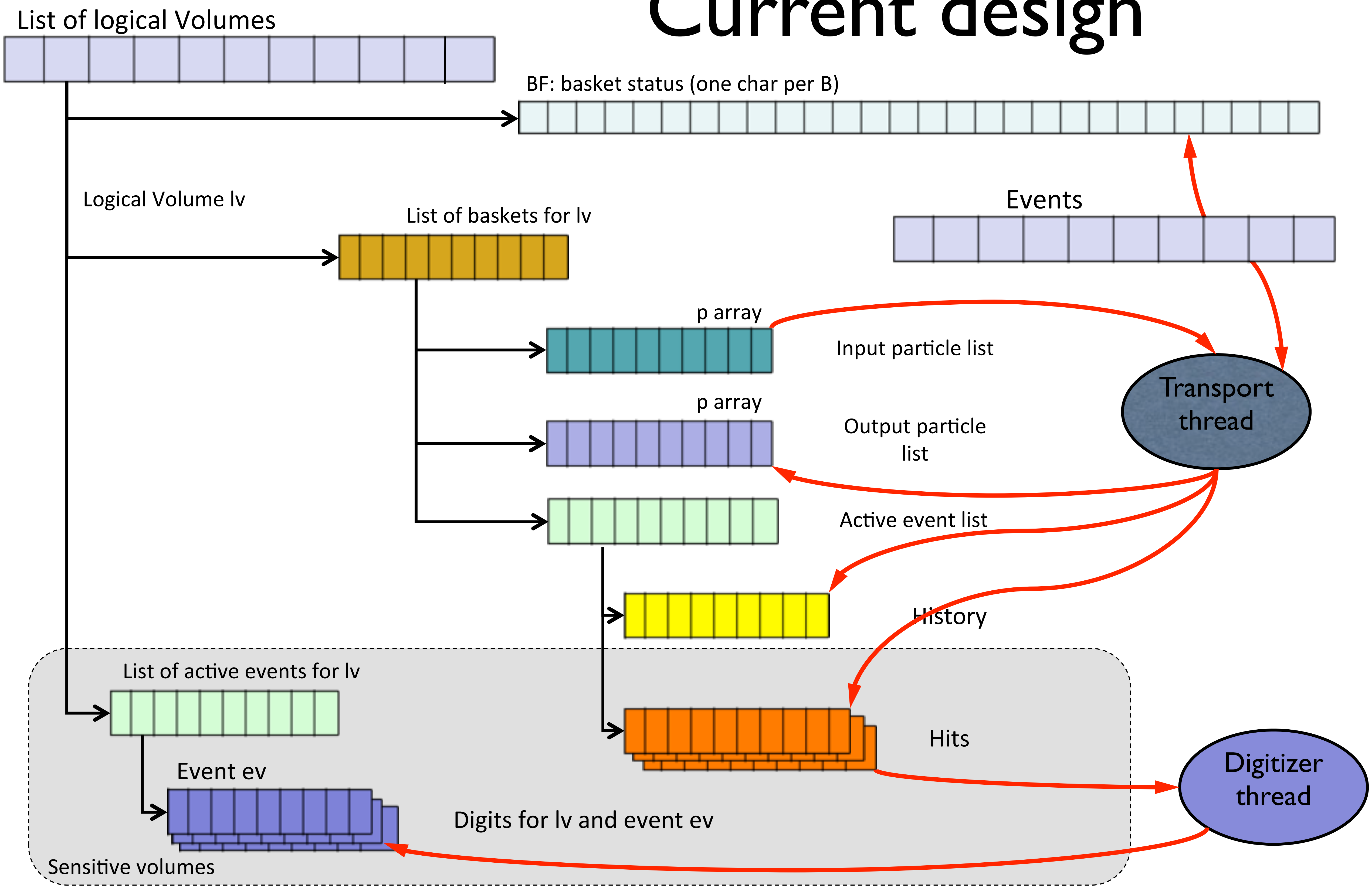
Current design



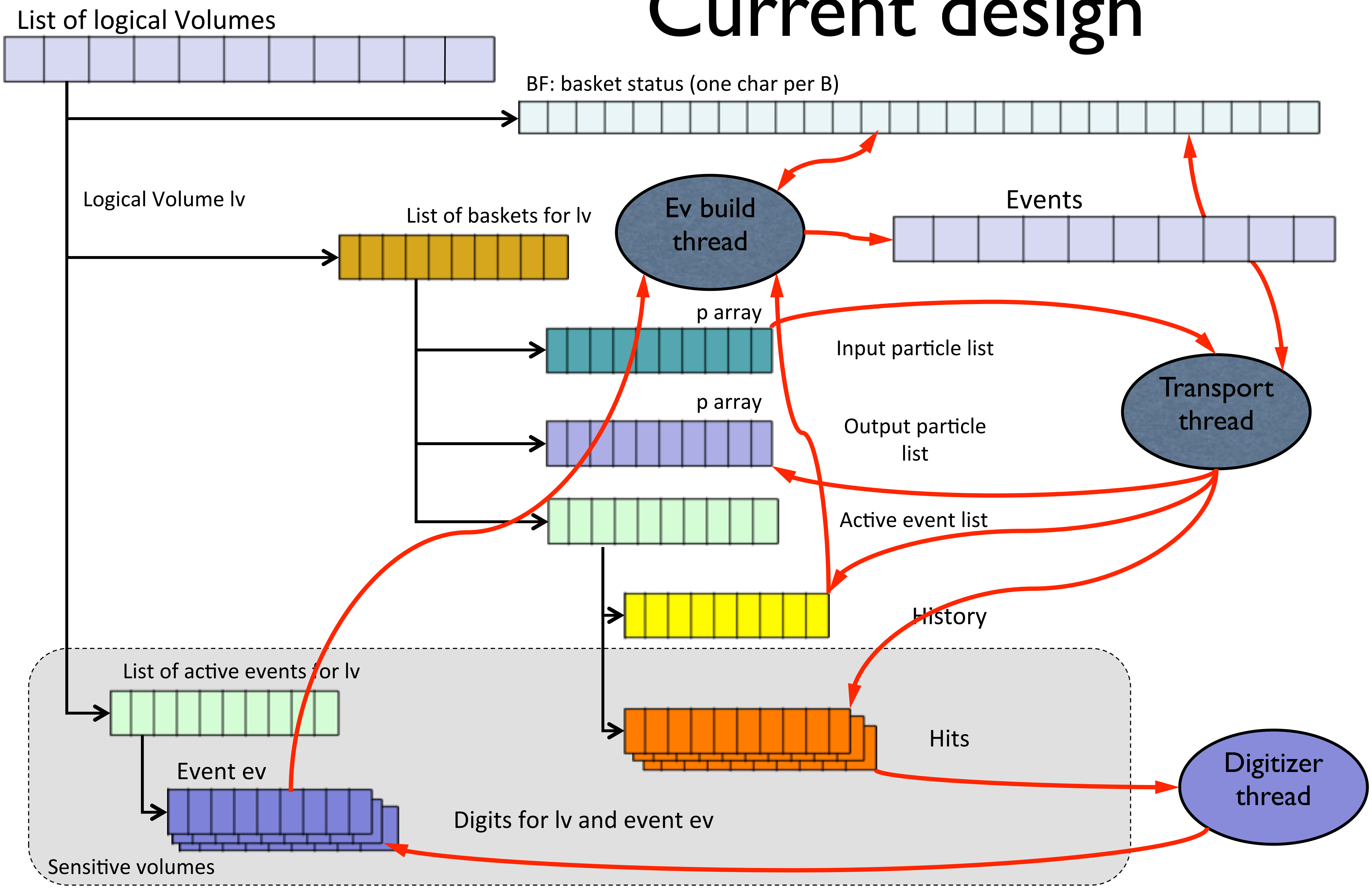
Current design



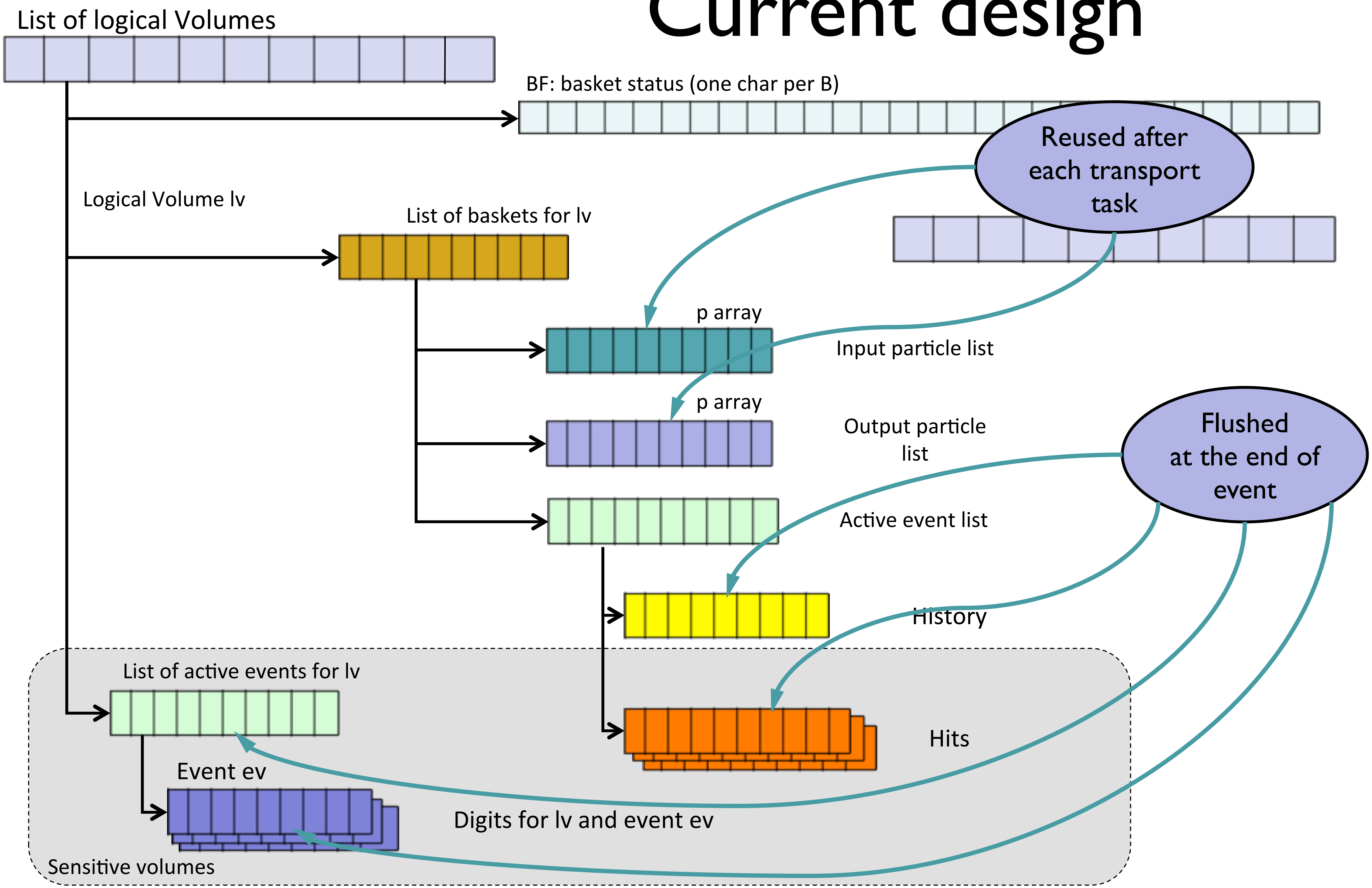
Current design



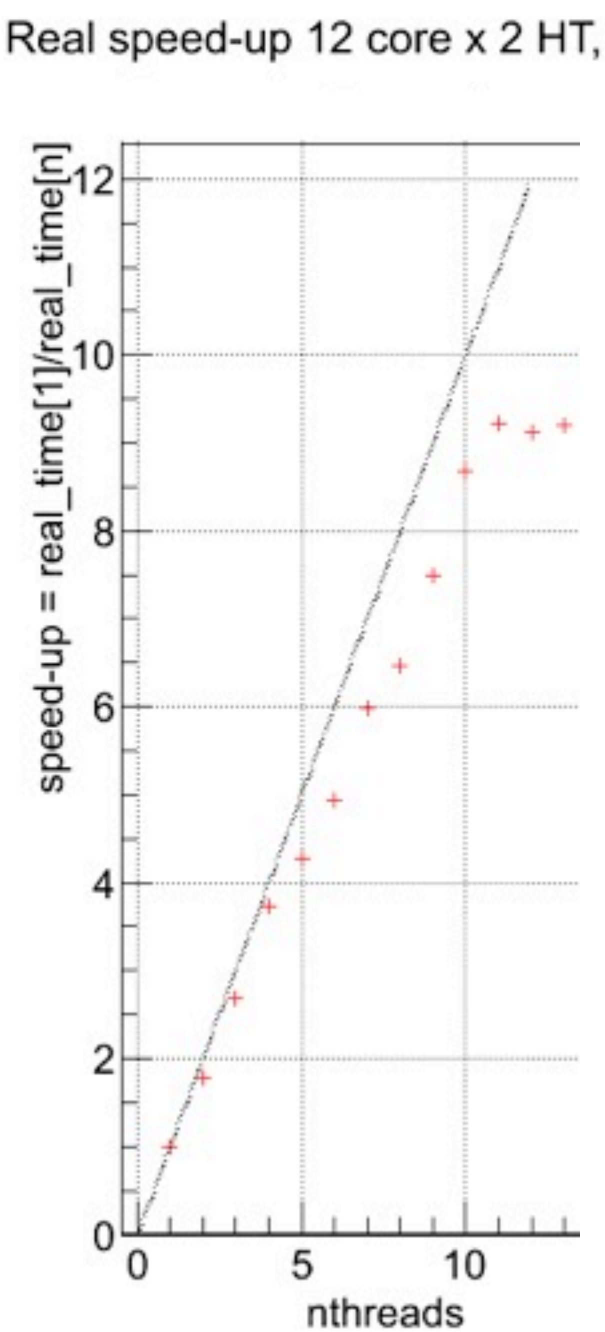
Current design

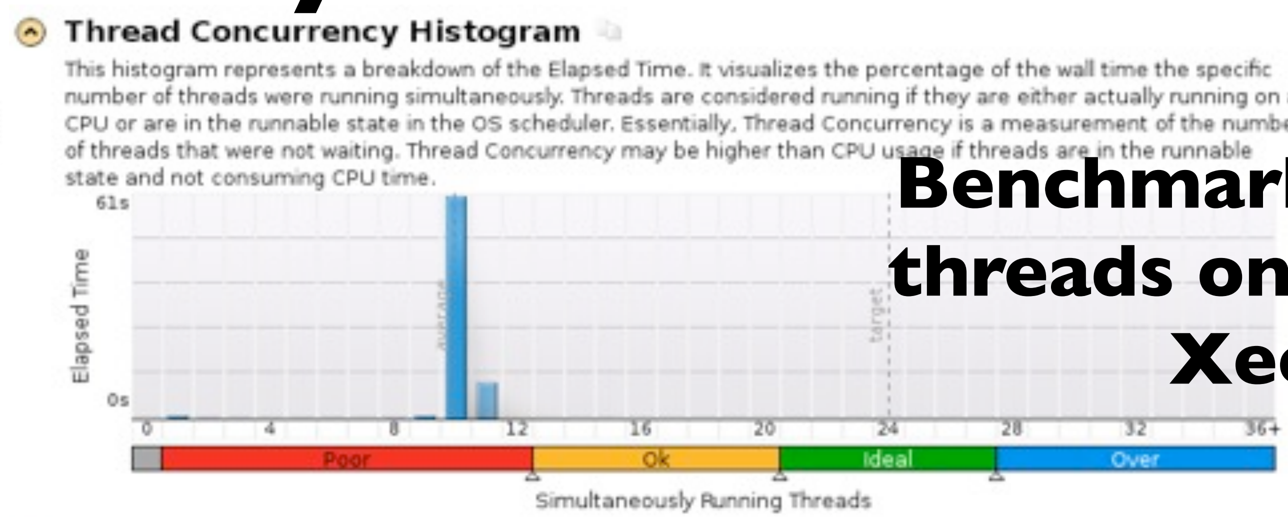


Current design

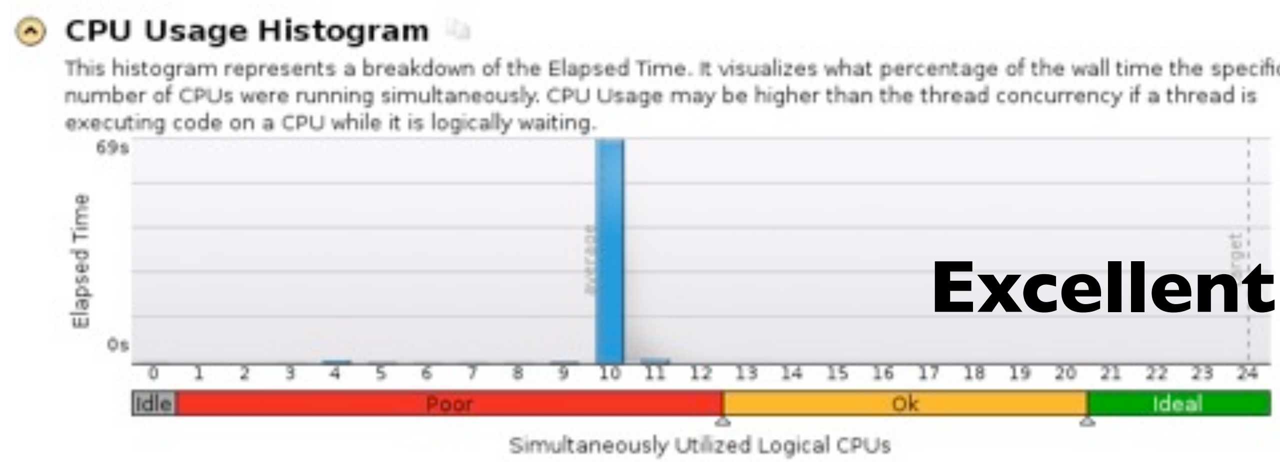


Preliminary benchmarks

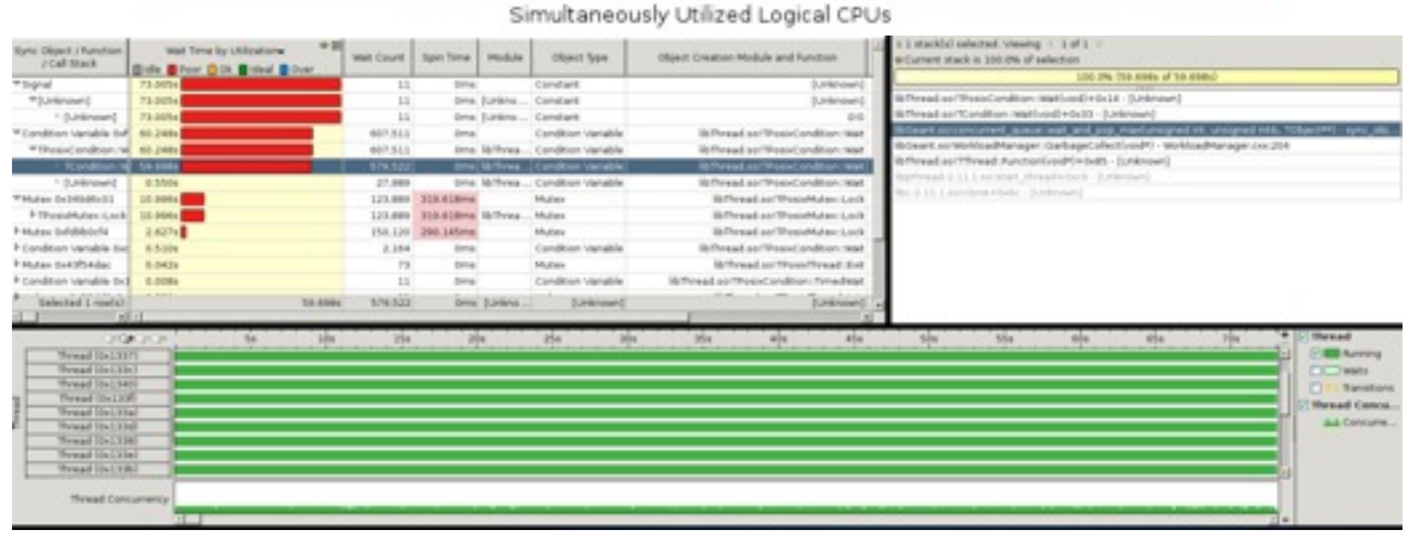




Benchmarking 10+1 threads on a 12 core Xeon



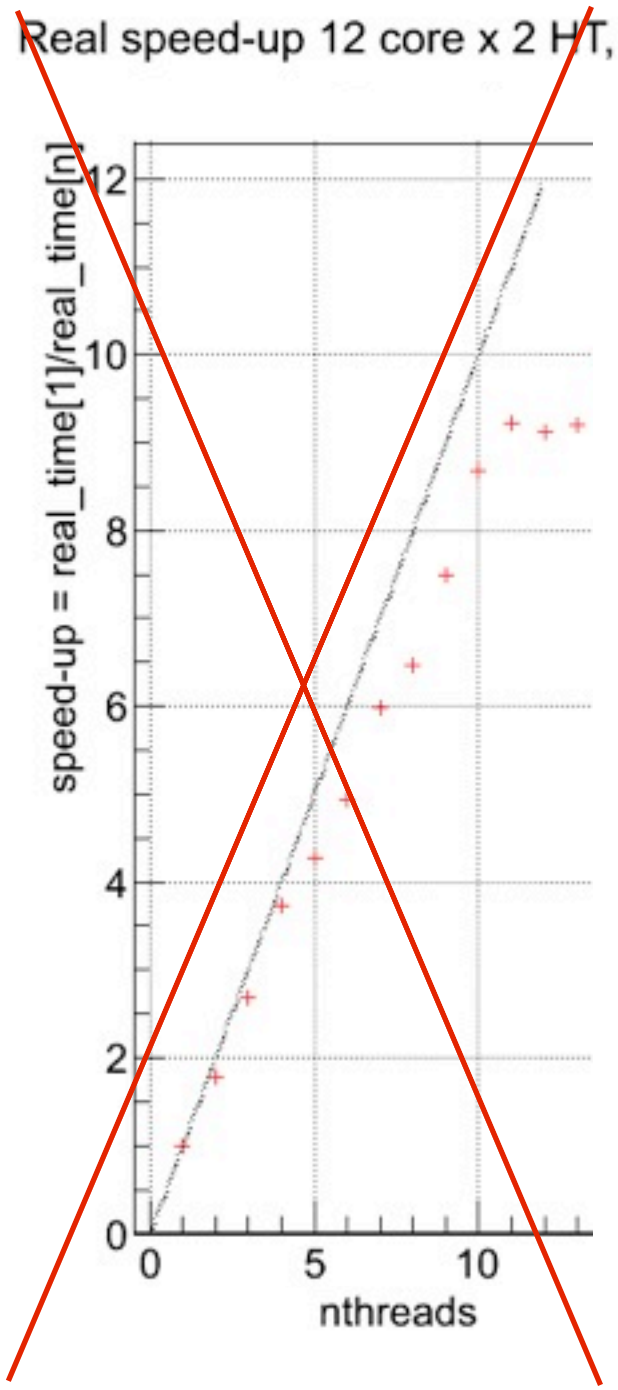
Excellent CPU usage

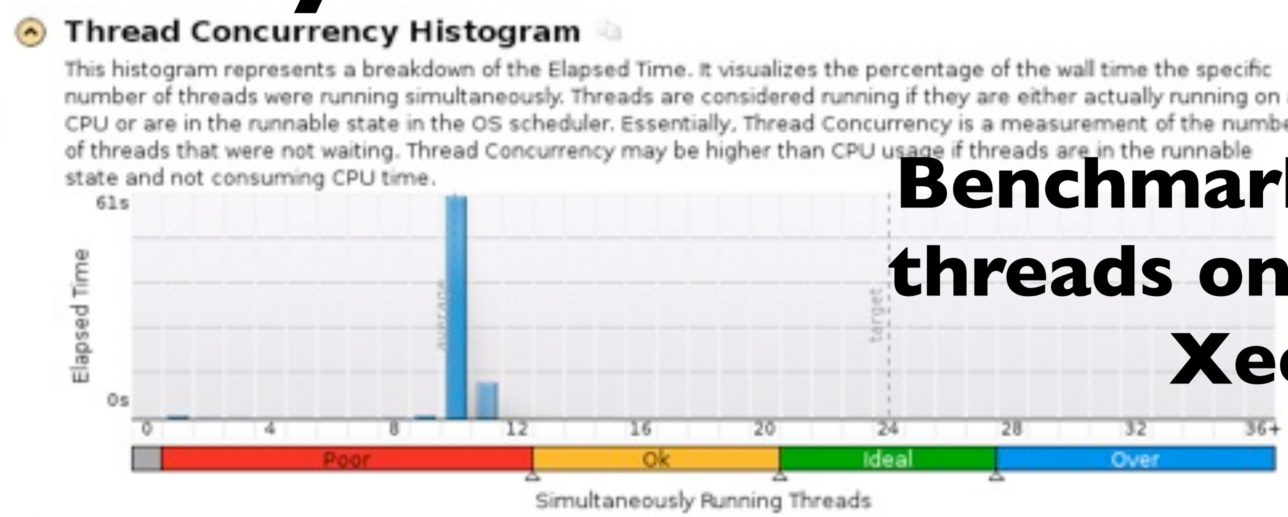


Event re-injection will improve the speed-up

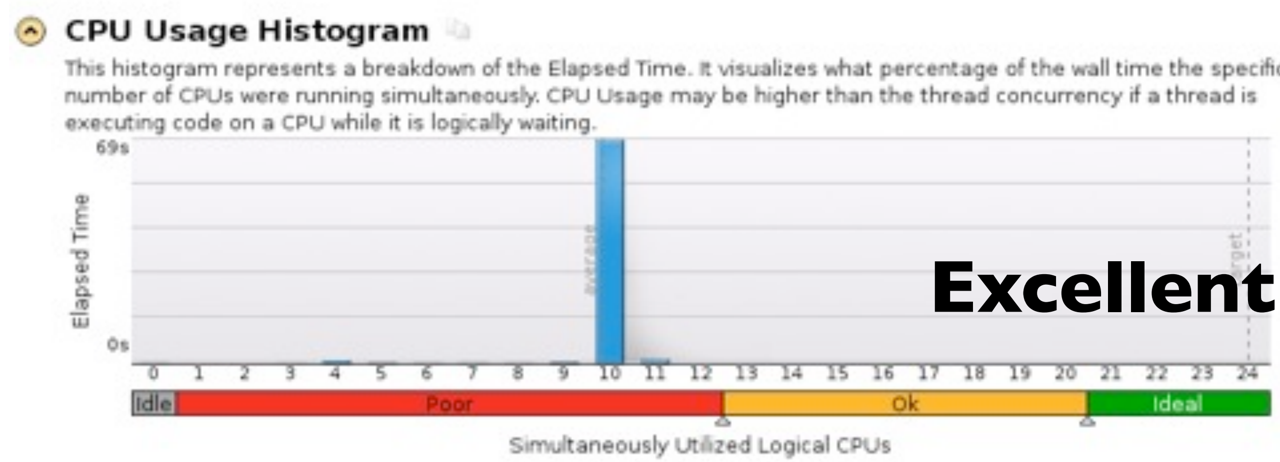
Locks and waits: some overhead due to transitions coming from exchanging baskets via concurrent queues

Preliminary benchmarks

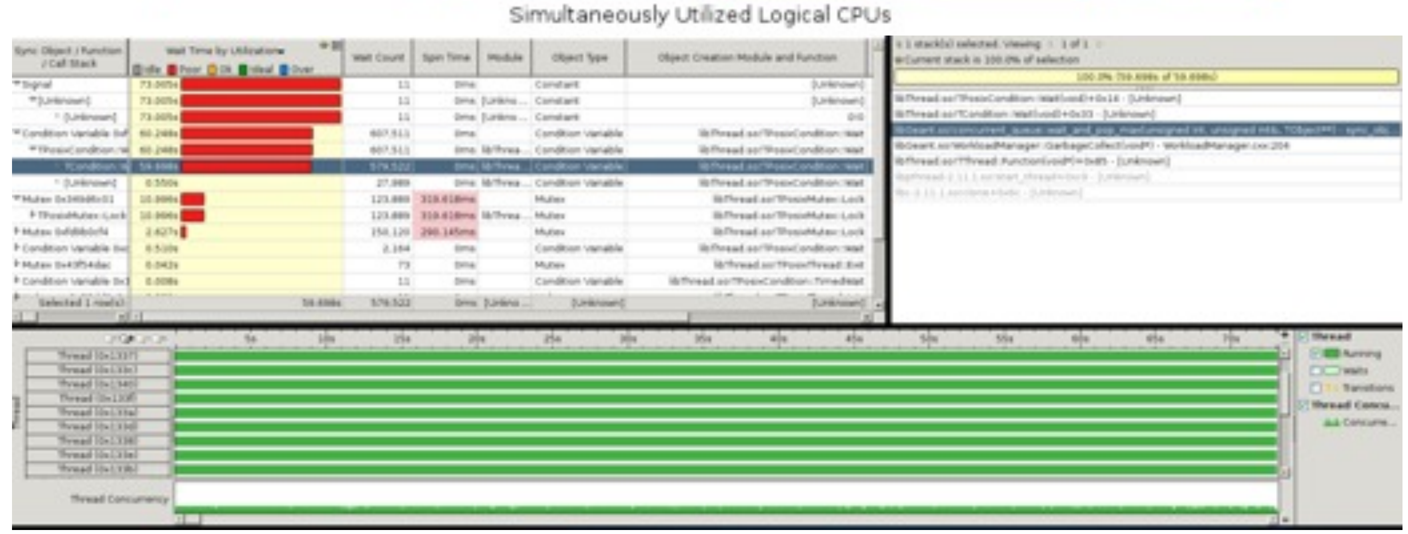




Benchmarking 10+1 threads on a 12 core Xeon



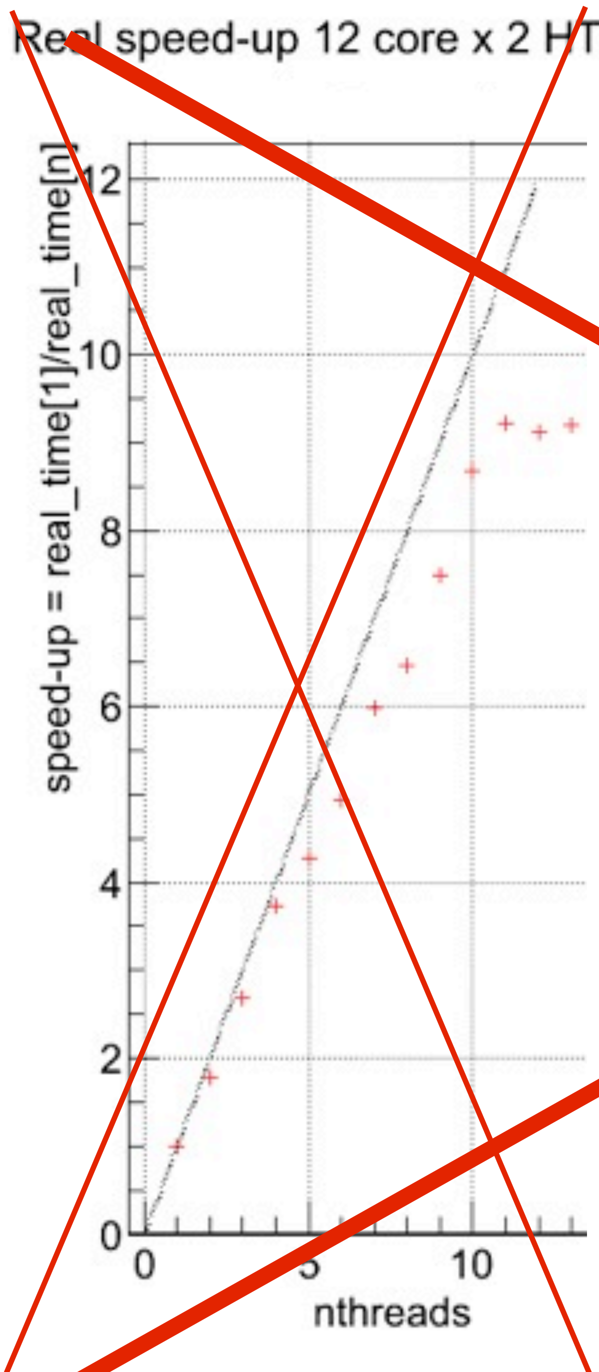
Excellent CPU usage

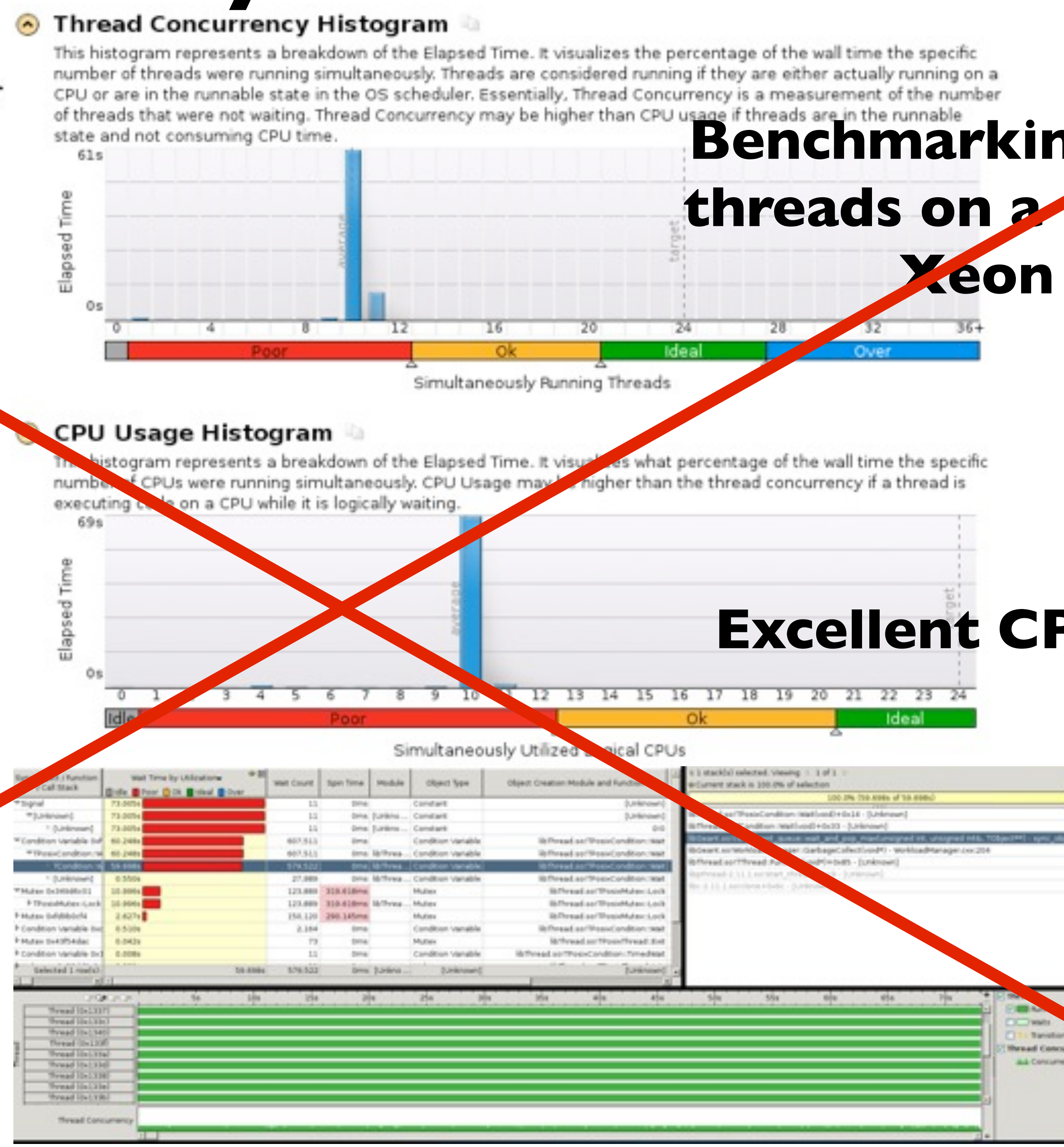


Event re-injection will improve the speed-up

Locks and waits: some overhead due to transitions coming from exchanging baskets via concurrent queues

Preliminary benchmarks





Benchmarking 10+1 threads on a 12 core Xeon

Excellent CPU usage

Event re-injection will improve the speed-up

Locks and waits: some overhead due to transitions coming from exchanging baskets via concurrent queues

Vector processing: Update on Gains for Geometry Calculations

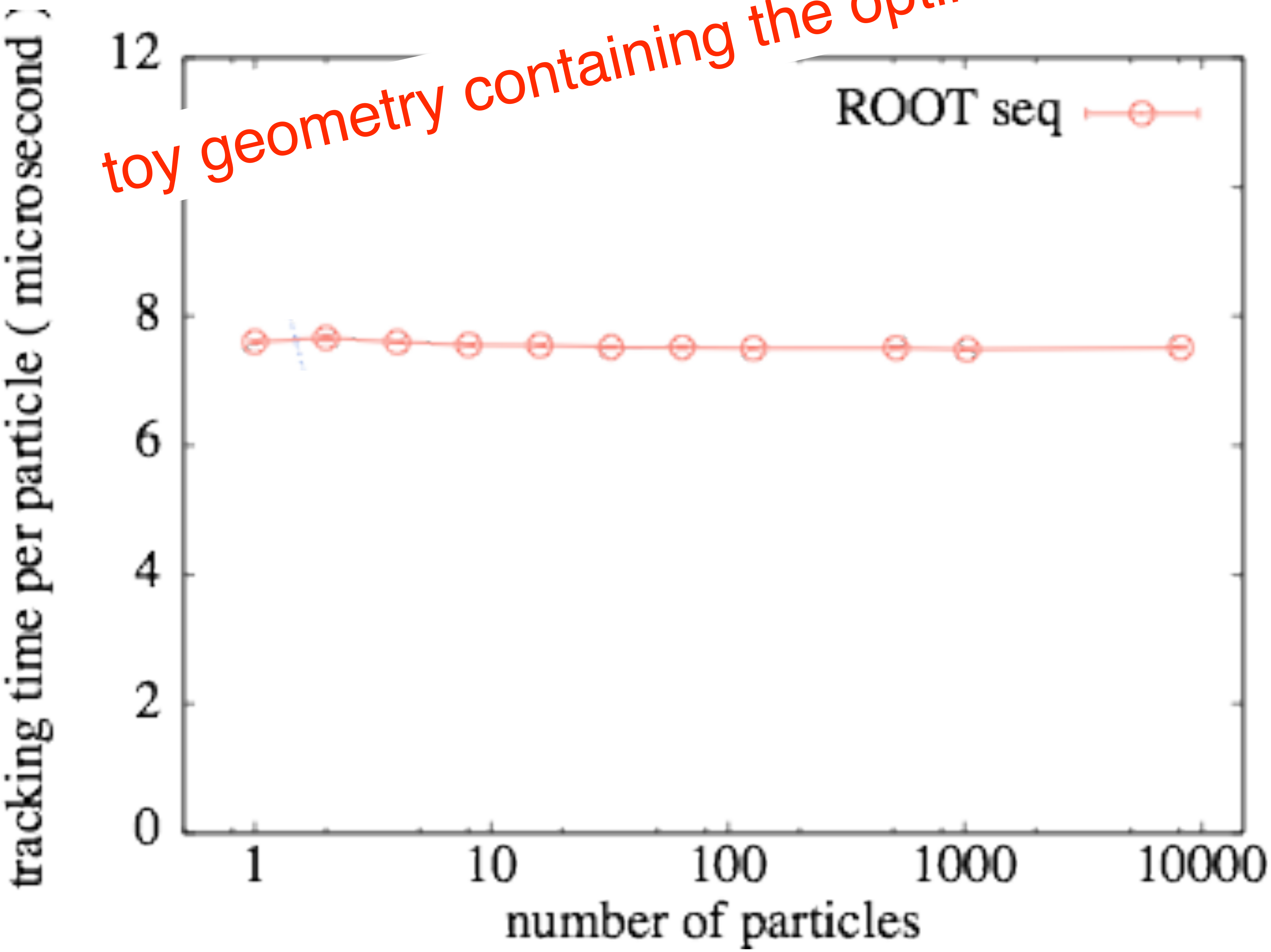
- Motivation: How much can geometry navigation gain from vector processing of particles?
  - Benefit from SIMD instruction sets
  - Benefit from instruction cache reuse
- To address second point, developed a more systematic benchmark scheme to quantify gains from instruction cache reuse (no code changes necessary)
- For any shape/volume, benchmarker creates automatic test cases (tracks) and probes geometry performances for varying number of particles

(slide by S. Wenzel)

Results from Benchmark: Overall Runtime

- time of processing/navigating N particles (P repetitions) using scalar algorithm (ROOT) versus vector version

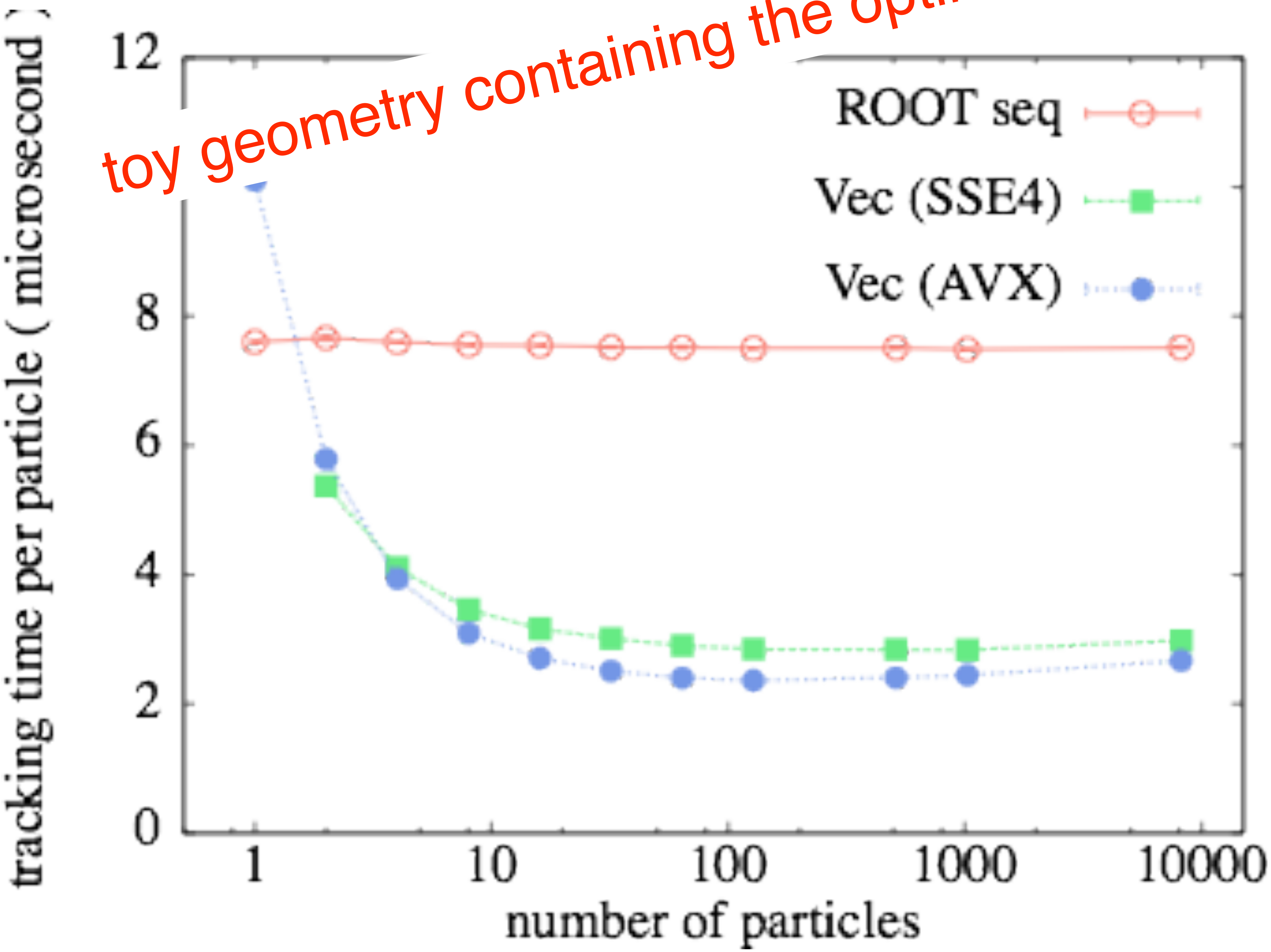
toy geometry containing the optimized solids



Results from Benchmark: Overall Runtime

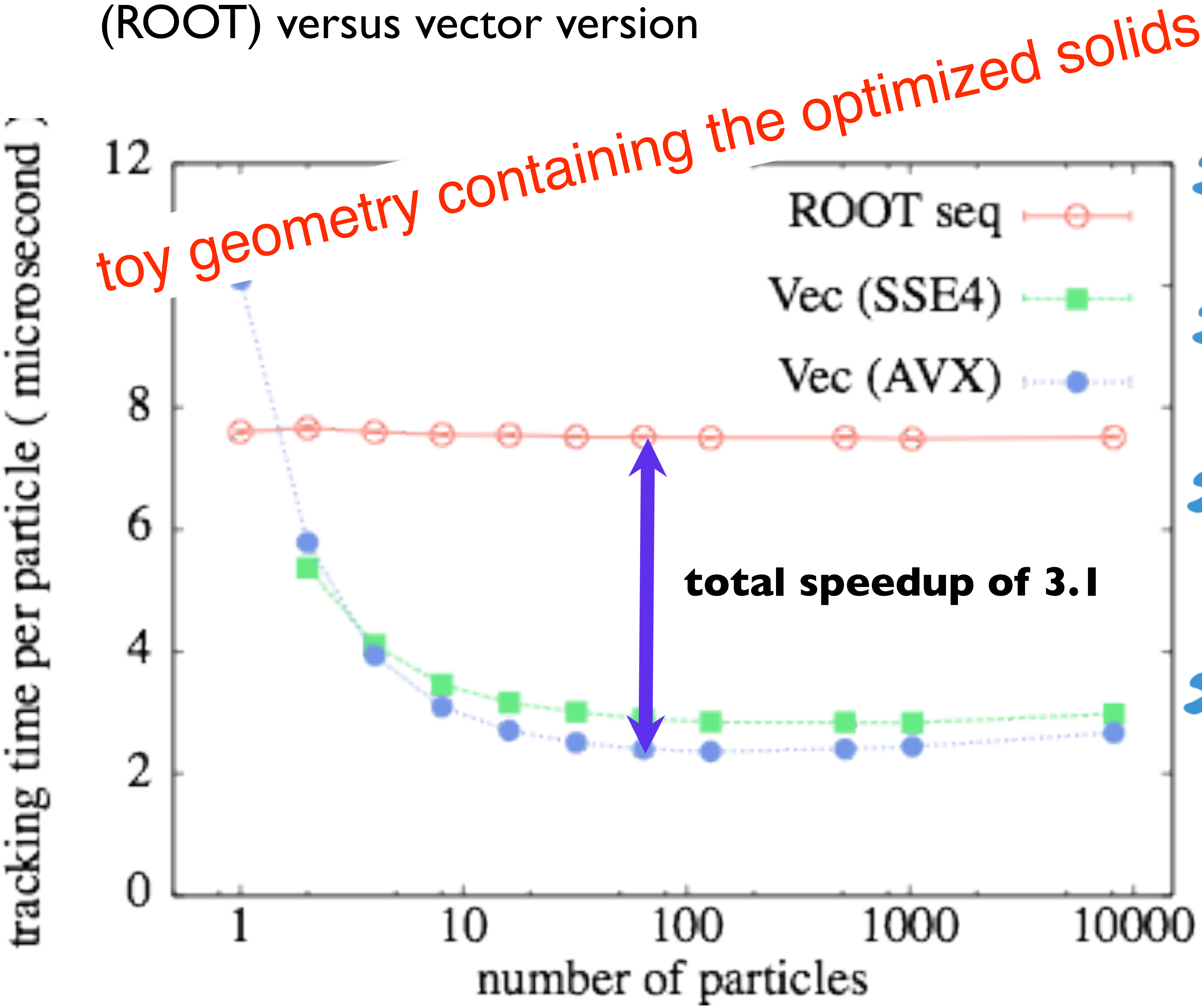
- time of processing/navigating N particles (P repetitions) using scalar algorithm (ROOT) versus vector version

toy geometry containing the optimized solids



Results from Benchmark: Overall Runtime

- time of processing/navigating N particles (P repetitions) using scalar algorithm (ROOT) versus vector version

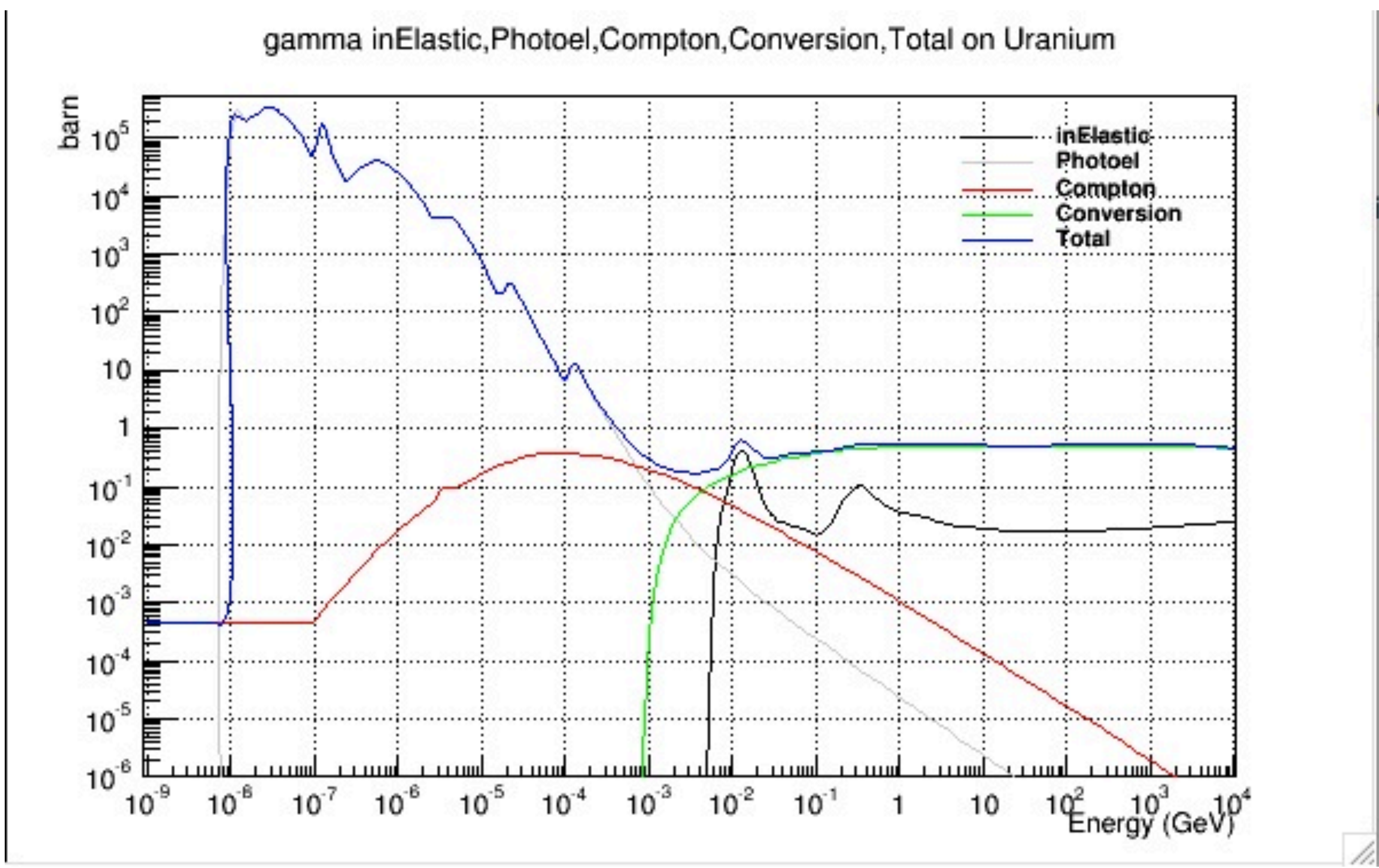


- excellent speedup for SSE4 version
- some further gain with AVX
- already gain considerably for small N
- there is an optimal point of operation (performance degradation for large N)

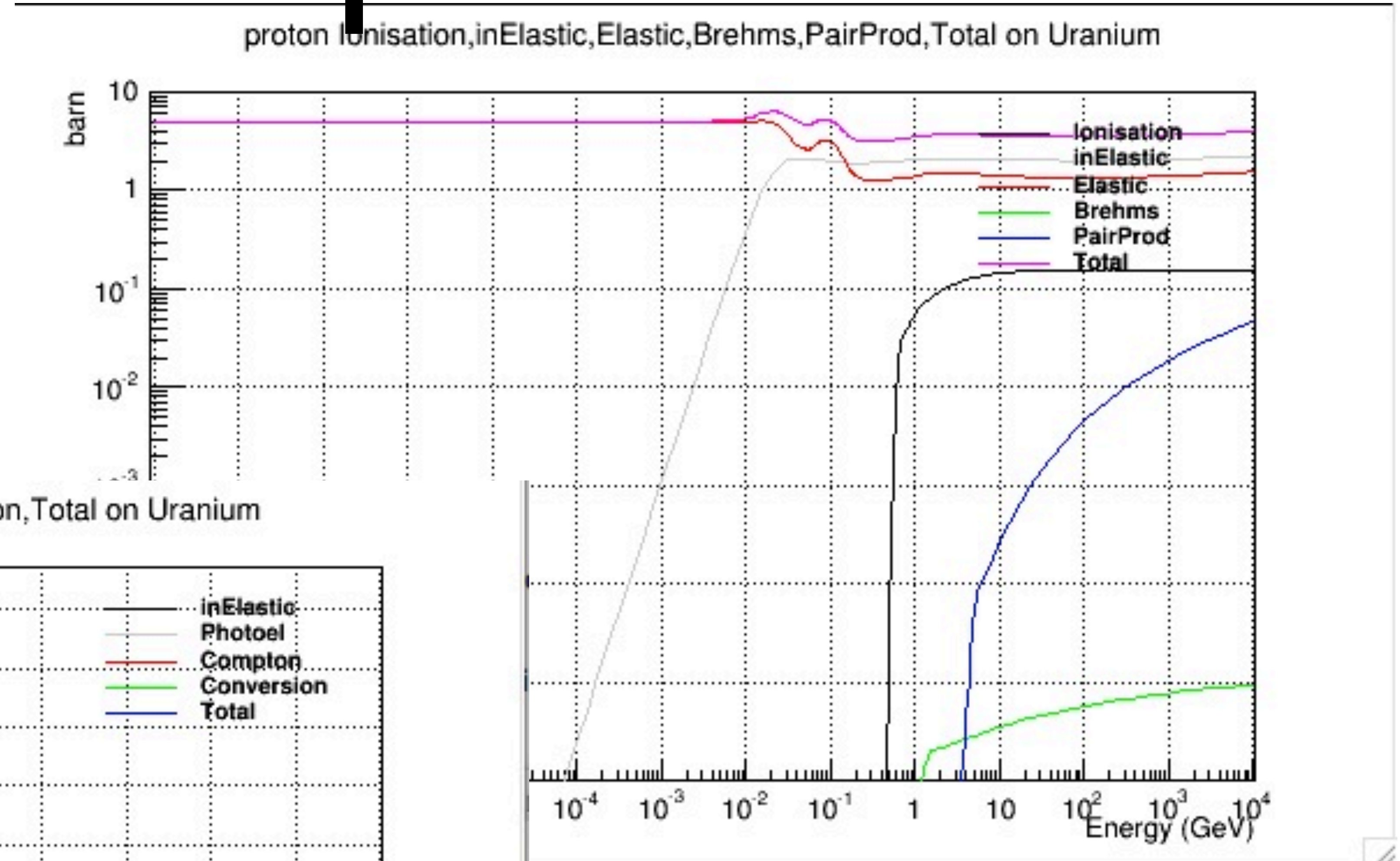
Physics

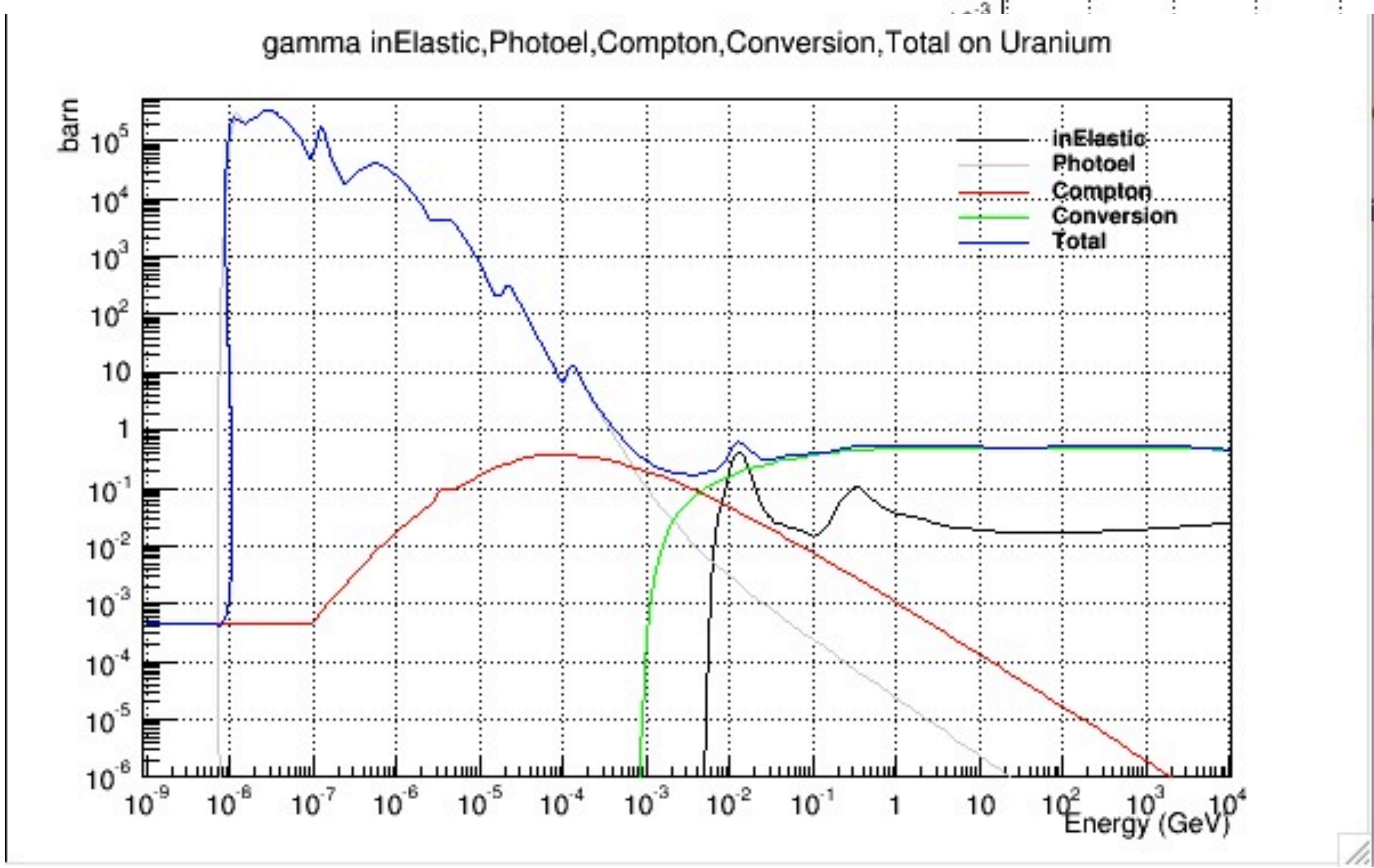
- We have selected the major mechanisms
  - Bremsstrahlung, e+ annihilation, Compton, Decay, Delta ray, Elastic hadron, Inelastic hadron, Pair production, Photoelectric, Capture
  - And of course energy loss and MS
- For each particle we have tabulated all G4 x-secs Z=1-92 (say E=100keV – 1TeV)
- For each reaction and each energy bin we generate N (10-50) final states with G4
  - In other words we generate a "database" of sampled products
- When a reaction is selected
  - Select the set of final states closer in energy
  - Randomly pick a final state & scale its energy (?), random rotate it around φ and rotoboost according to projectile
- This will give us a "near-realistic" shower development

Examples

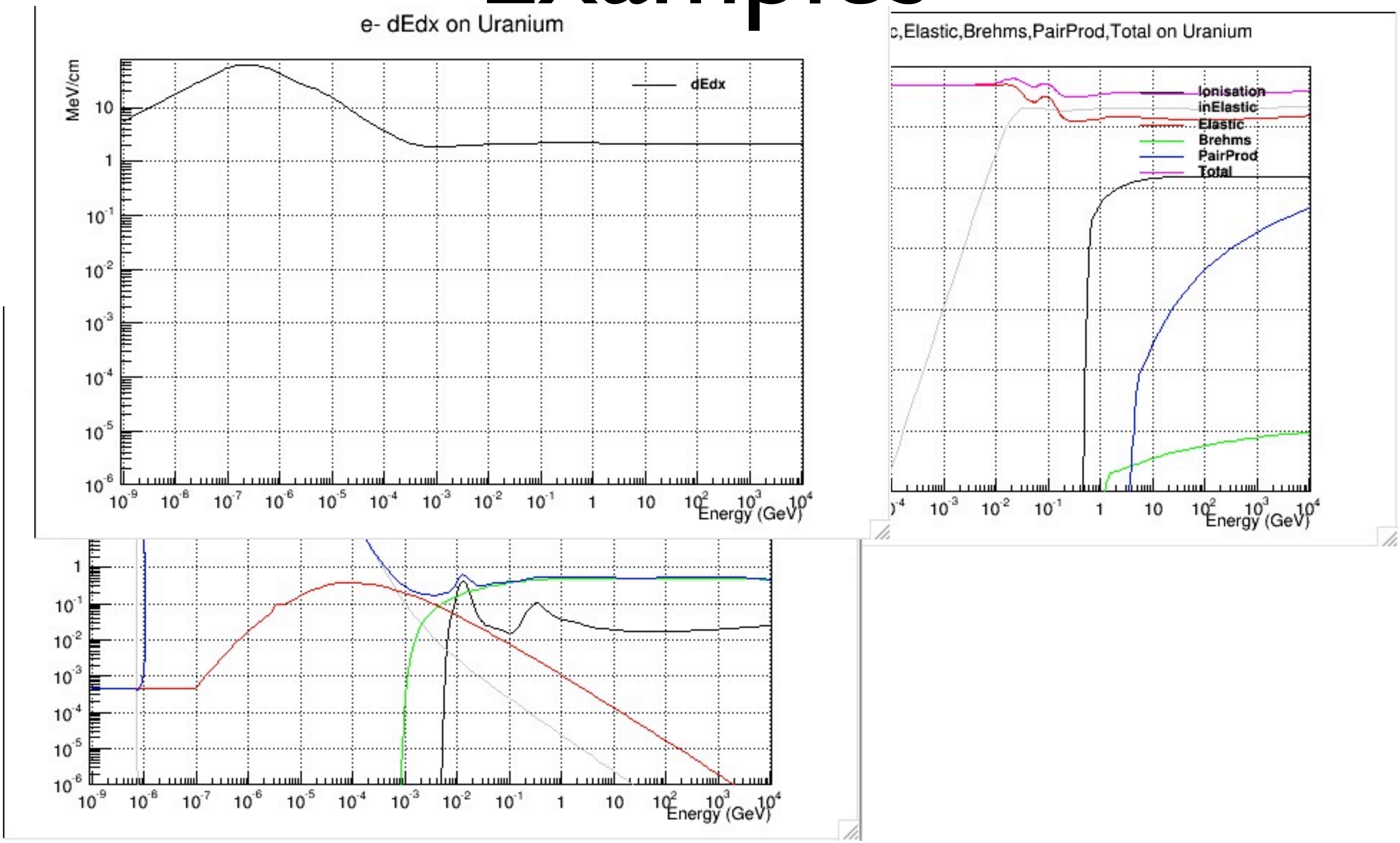

gamma inElastic,Photoel,Compton,Conversion,Total on Uranium
barn
inElastic
Photoel
Compton
Conversion
Total
Energy (GeV)

Examples

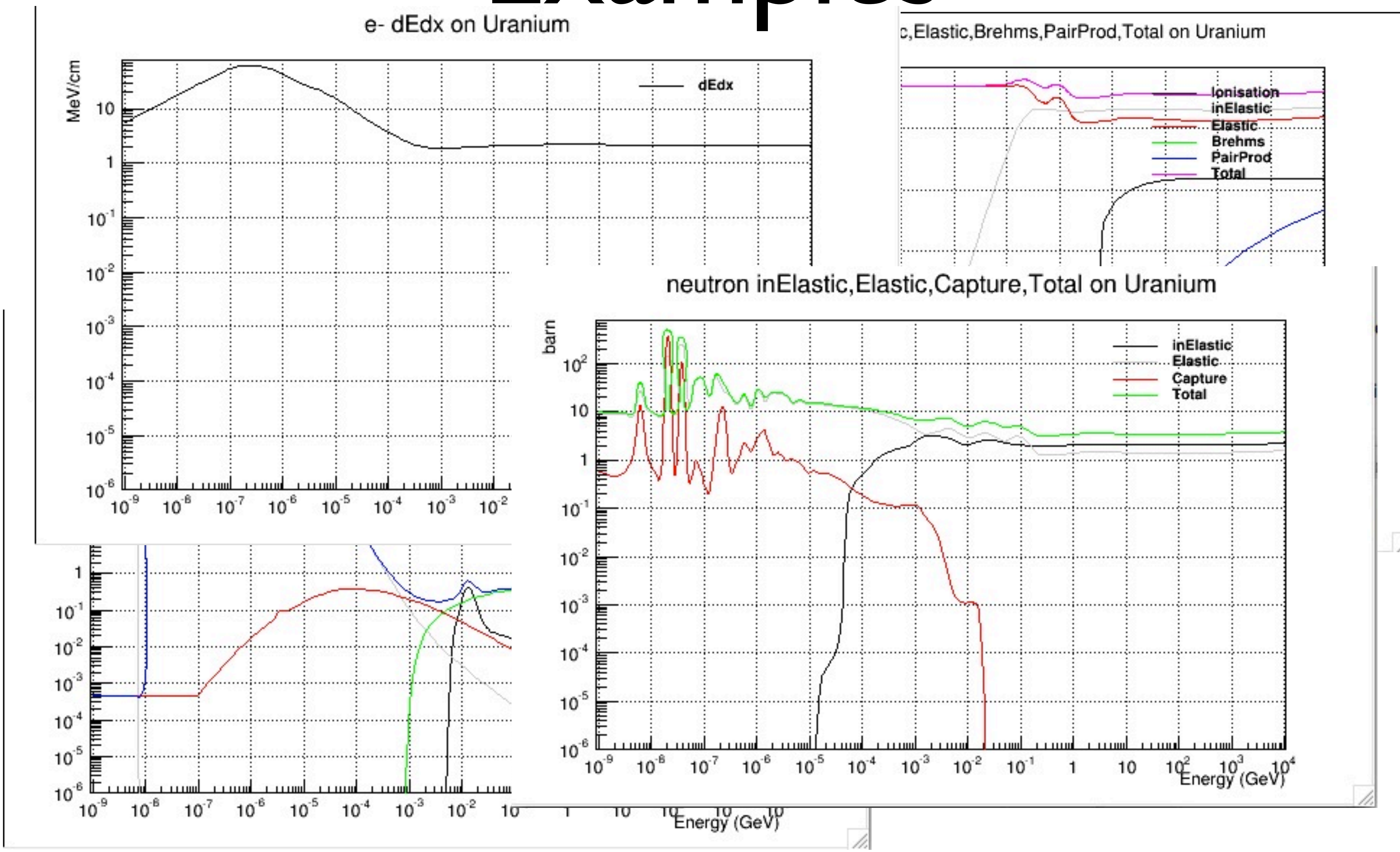

proton Ionisation,inElastic,Elastic,Brehms,PairProd,Total on Uranium
barn
Ionisation
inElastic
Elastic
Brehms
PairProd
Total
Energy (GeV)


gamma inElastic,Photoel,Compton,Conversion,Total on Uranium
barn
inElastic
Photoel
Compton
Conversion
Total
Energy (GeV)

Examples


e- dEdx on Uranium
MeV/cm
dEdx
Energy (GeV)
c,Elastic,Brehms,PairProd,Total on Uranium
Ionisation
inElastic
Elastic
Brehms
PairProd
Total
Energy (GeV)
Energy (GeV)

Examples

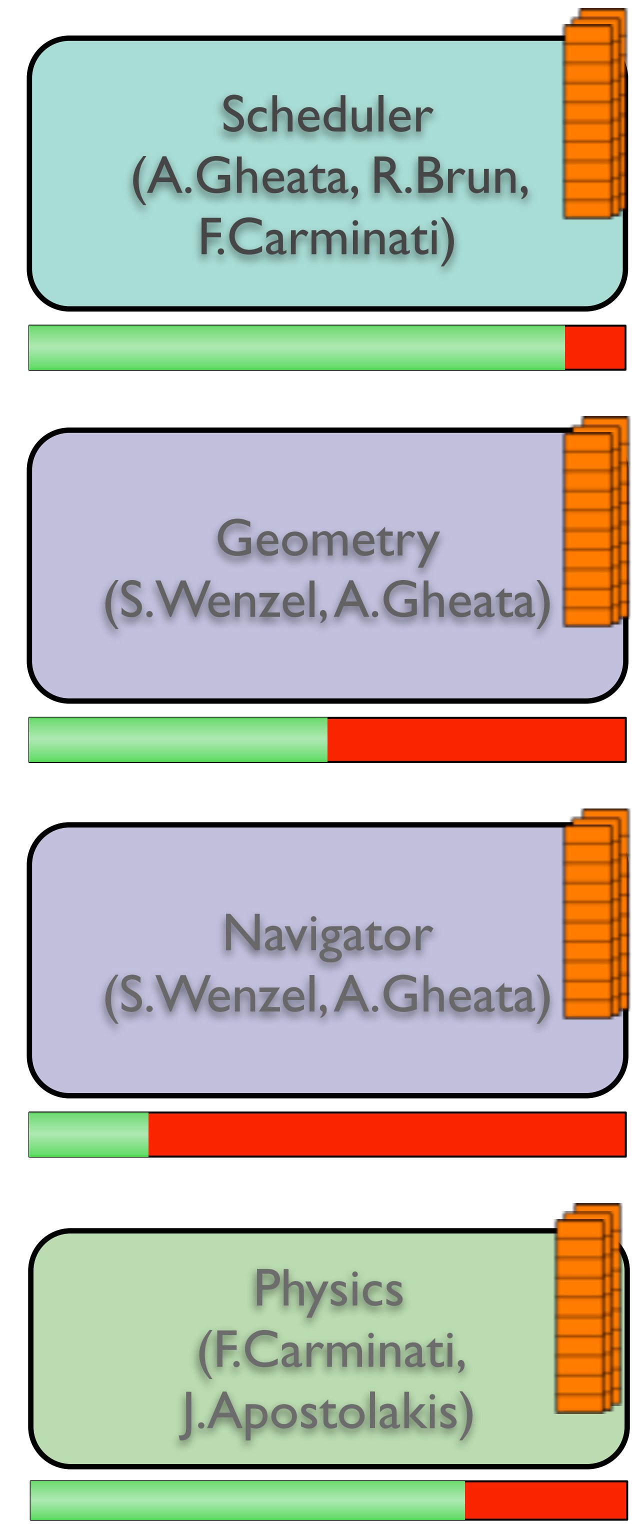

e- dEdx on Uranium
MeV/cm
dEdx
c,Elastic,Brehms,PairProd,Total on Uranium
Ionisation
inElastic
Elastic
Brehms
PairProd
Total
neutron inElastic,Elastic,Capture,Total on Uranium
barn
inElastic
Elastic
Capture
Total
Energy (GeV)
Energy (GeV)

Physics expectations

- Cross sections are precisely calculated
- Final states will give a precise description of the multiplicity
- It will be interesting to see how "good" will the physics description be
  - Of course we do not expect it to be as good as G4
  - But it could be the seed of an interesting fast simulation option
- The size of all x-sec (100 bins, 10keV-10TeV) is 90MB
- The size of the sampled final states (10 final states) is
  - ~2.2MB for H, ~3.5MB for U

Where are we now?

- Scheduler
  - The new version, hopefully improved of the scheduler has been committed and we are testing it
- Geometry
  - The proof or principle that we can achieve large speedups (3-5+) is there (see S.Wenzel's talk), however a lot of work lays ahead
- Navigator
  - "Percolating" vectors through the navigator is a difficult business. We have a simplified navigator that achieves that (S.Wenzel), but more work is needed here
- Physics
  - Can generate x-secs and final states and sample them, but there are still many points to be clarified with Geant4 experts



Targets

- By the end of the year we should be able to “glue” the different pieces together
- Target is to measure the evolution of the memory footprint and the performance of the code at least in terms of hardware counters
- Absolute performance measurements will be much harder
    - Difficult compare apples to apples
    - Probably we need to develop dedicated benchmarks
- It will be interesting to compare physics performance

The full picture

- The objective of this work is to demonstrate the potential for a substantial speedup thanks to MT, improved locality and SIMD
- For the moment we concentrated on Xeon architecture for the SIMD part, but we intend to extend this to GPU and to Xeon PHI
- We are working closely with Geant4 for the physics tables
- Once the prototyping phase over, we will have to sit down with the stakeholders and decide how to proceed from there

The larger picture

- We expect the findings (code, algorithms) to be used for all programs in HEP and to be contributed to the HPC community
- We expect this experience to benefit from the other HPC initiatives in High Energy Physics
- It is clear that the danger of local developments...
  - Being lost to the community
  - Reinventing the wheel
- ...for lack of communication is very high
- We need a concerted community effort

Summary

- HEP needs all the cycles it can obtain, nowadays this means using parallelism and SIMD
- Simulation is the ideal primary target for investigation for its relative experiment independence and its importance in the use of computing resources
- The GeantVector project aims at demonstrating substantial speedup (3-5+) on modern architectures
- The work is done in close collaboration with the stakeholders and with Geant4

Thank you!