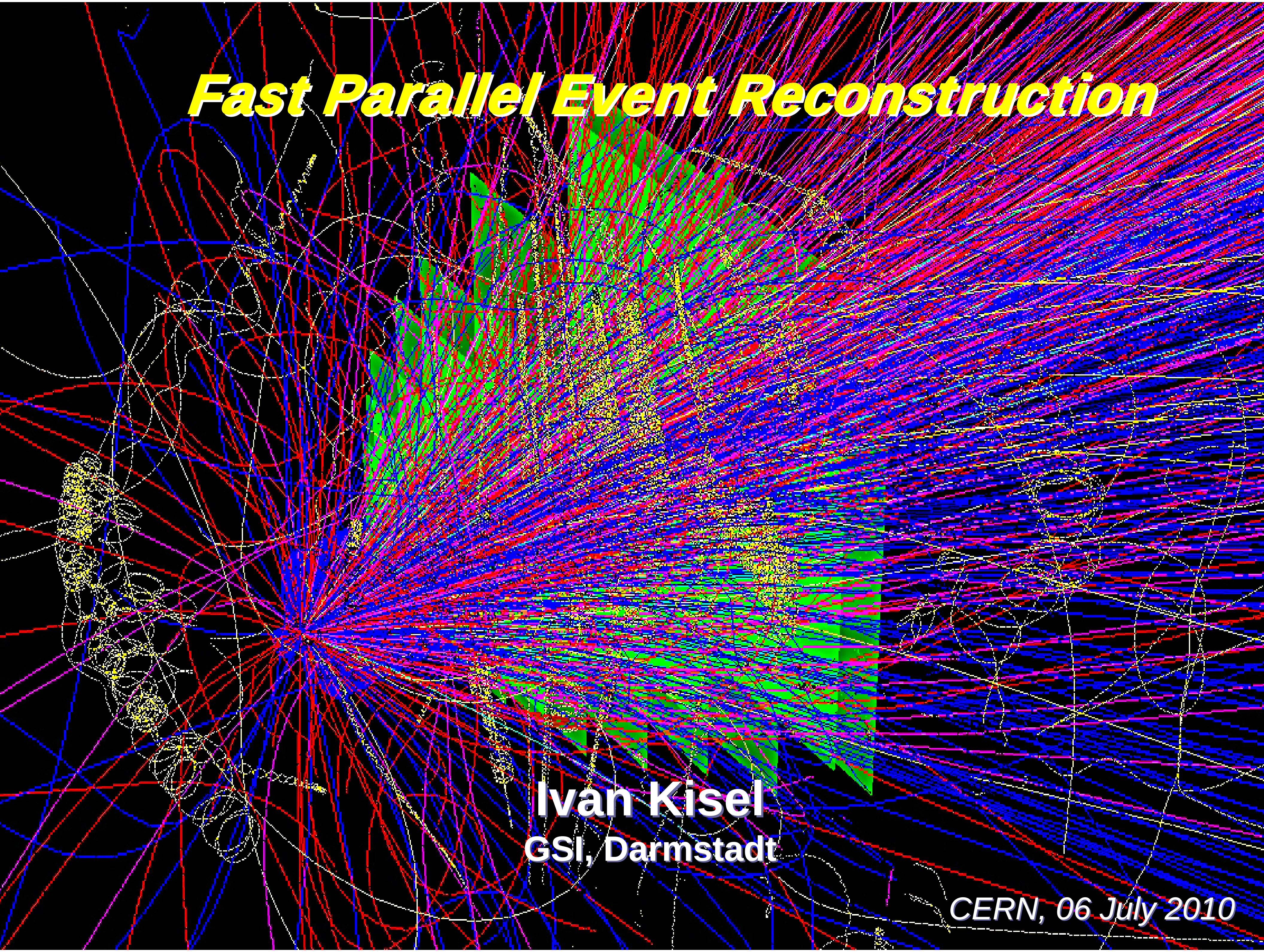
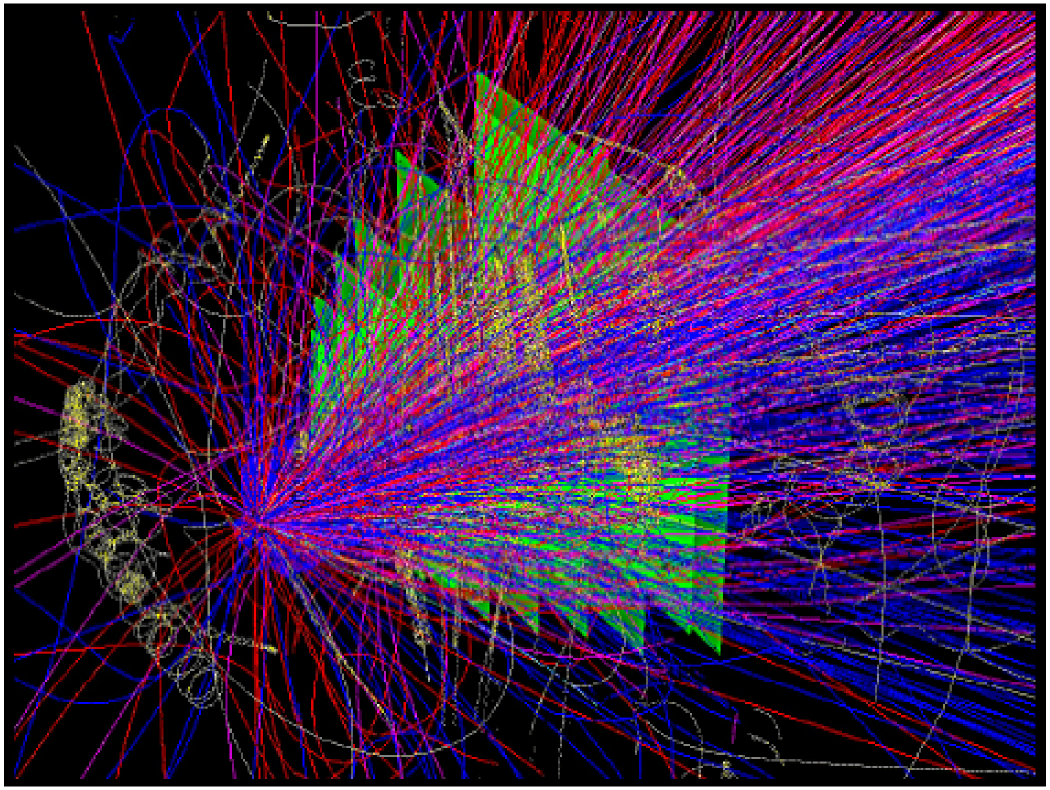


Fast Parallel Event Reconstruction

Ivan Kisel
GSI, Darmstadt

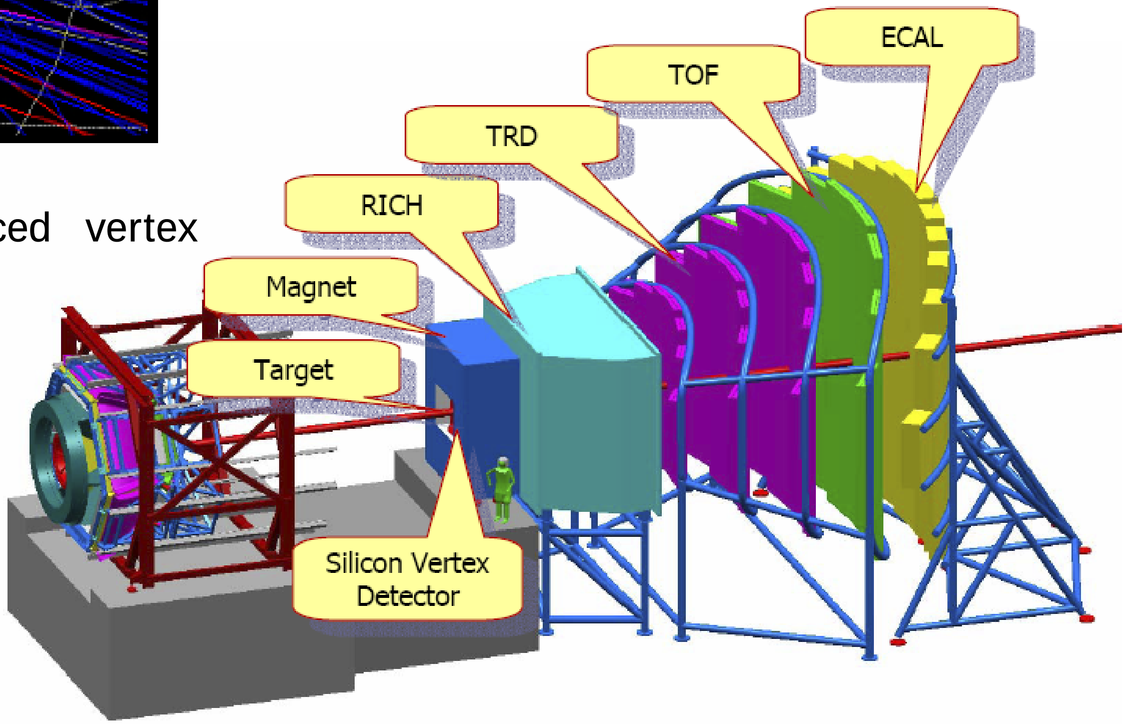
CERN, 06 July 2010

Tracking Challenge in CBM (FAIR/GSI, Germany)



- Fixed-target heavy-ion experiment
- 10^7 Au+Au collisions/s
- 1000 charged particles/collision
- Non-homogeneous magnetic field
- Double-sided strip detectors (85% combinatorial space points)

Track reconstruction in STS/MVD and displaced vertex search are required in the first trigger level.

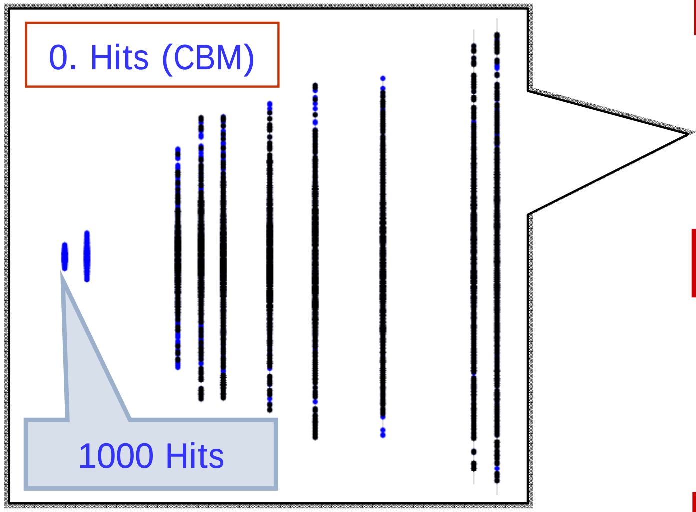


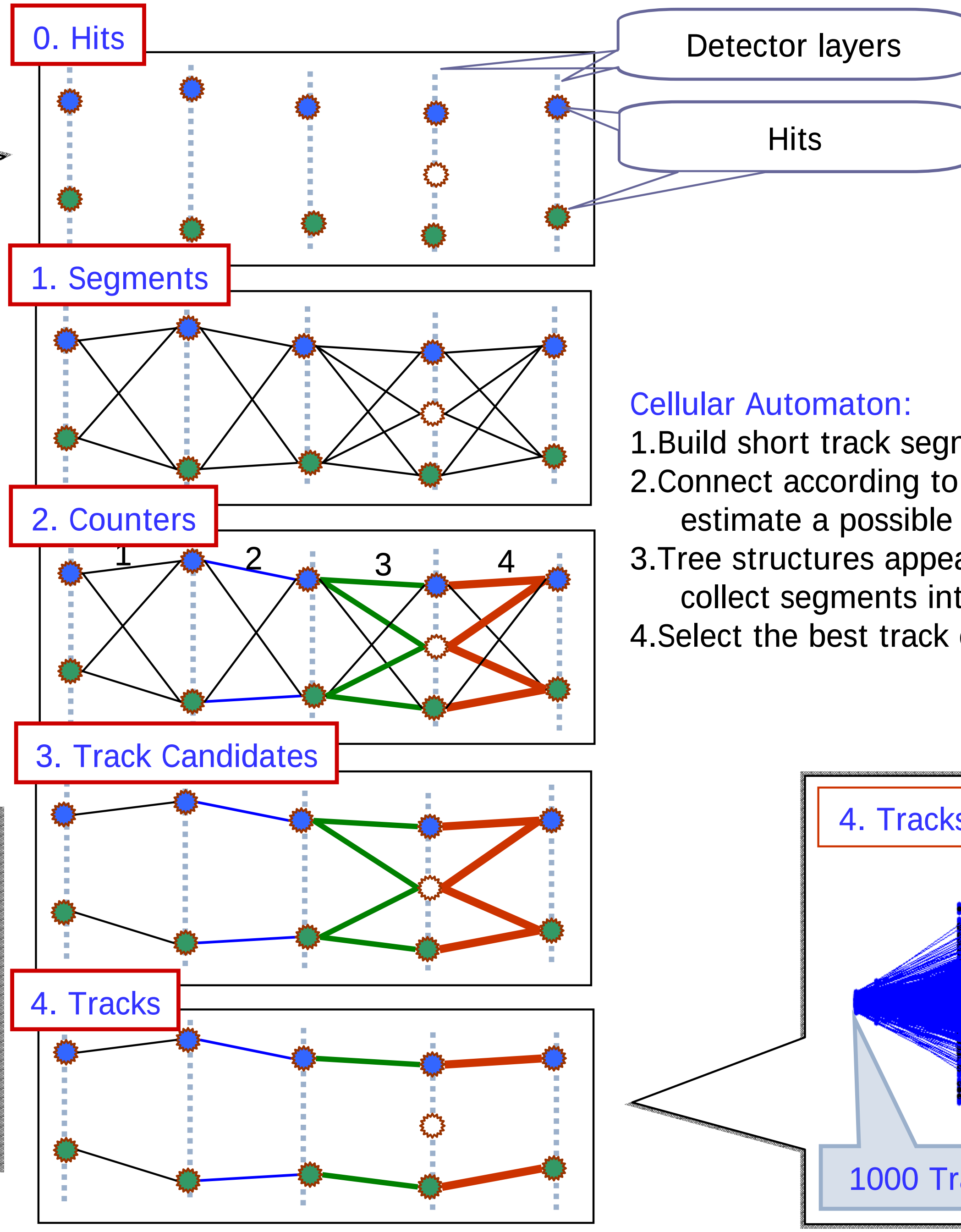
Reconstruction packages:

- track finding — Cellular Automaton (CA)
- track fitting — Kalman Filter (KF)
- vertexing — KF Particle

Cellular Automaton (CA) as Track Finder

Track finding: Wich hits in detector belong to the same track? – Cellular Automaton (CA)





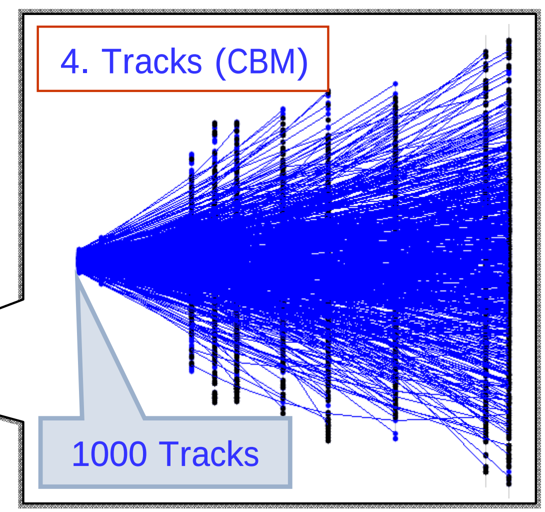
Cellular Automaton:

1. Build short track segments.
2. Connect according to the track model, estimate a possible position on a track.
3. Tree structures appear, collect segments into track candidates.
4. Select the best track candidates.

Cellular Automaton:

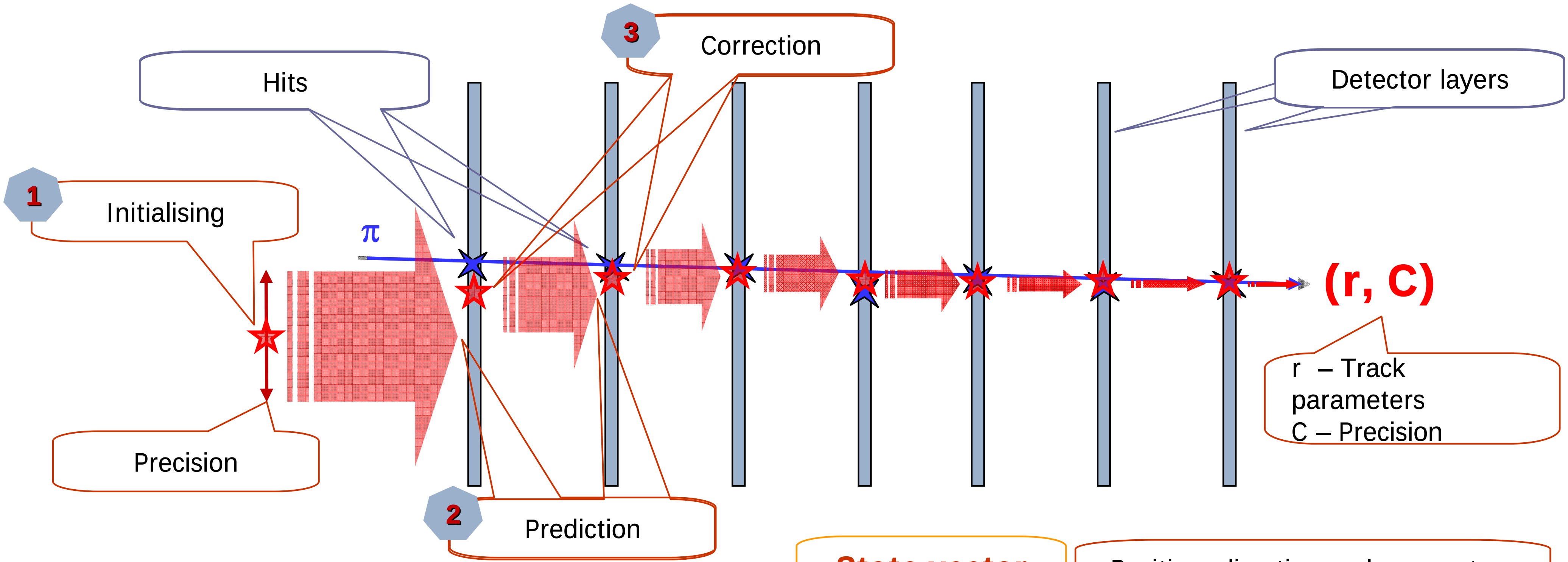
- local w.r.t. data
- intrinsically parallel
- extremely simple
- very fast

Perfect for many-core CPU/GPU !



Kalman Filter (KF) based Track Fit

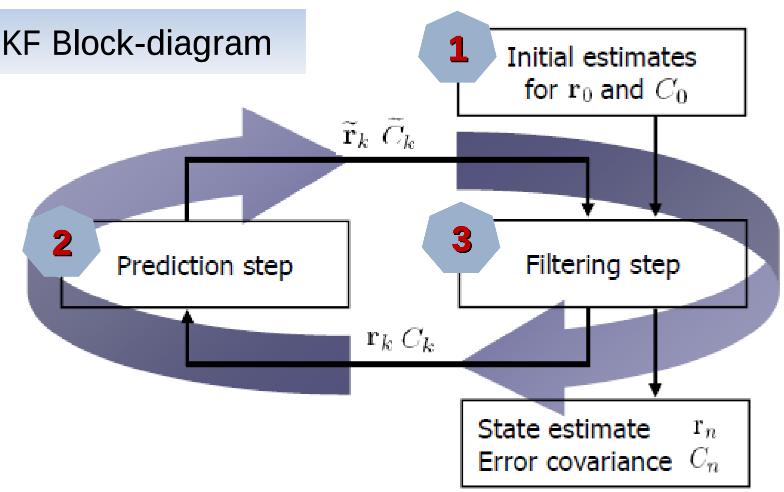
Track fit: Estimation of the track parameters at one or more hits along the track – Kalman-Filter (KF)



State vector – Position, direction and momentum

$$\mathbf{r} = \{\ x, y, z, p_x, p_y, p_z\ \}$$

KF Block-diagram



KF as a recursive least squares method

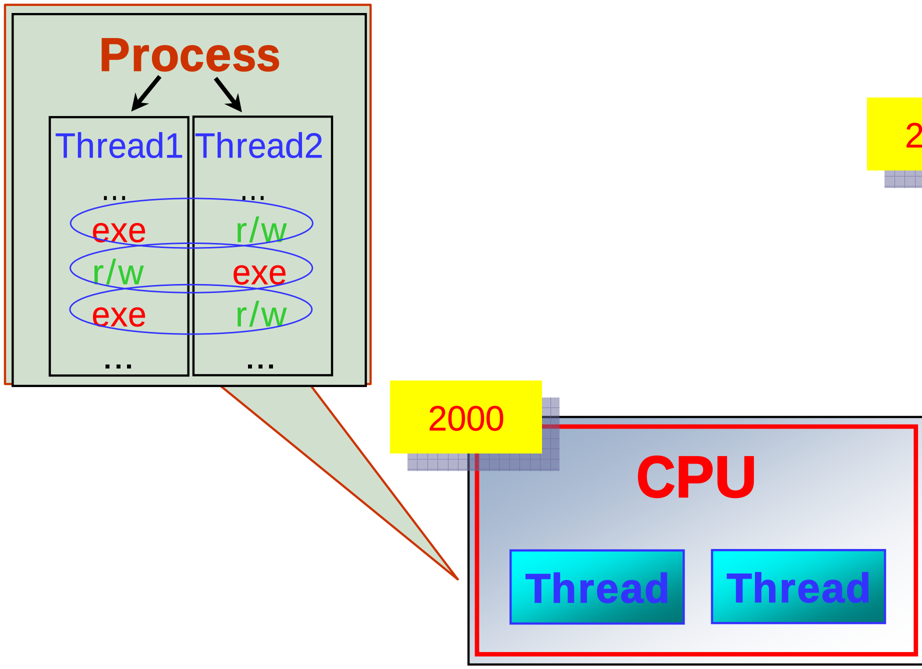
Kalman Filter:
1. Start with an arbitrary initialization.
2. Add one hit after another.
3. Improve the state vector.
4. Get the optimal parameters after the last hit.

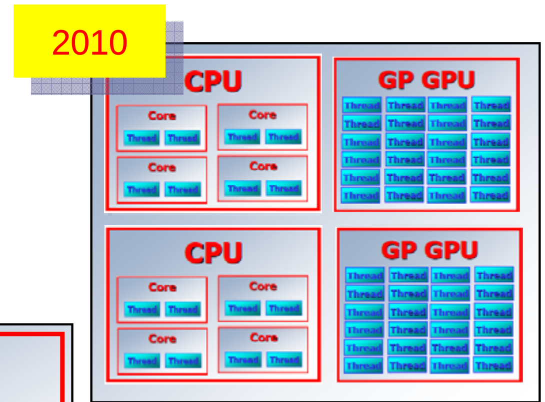
Nowadays the Kalman-Filter is used in almost all HEP experiments

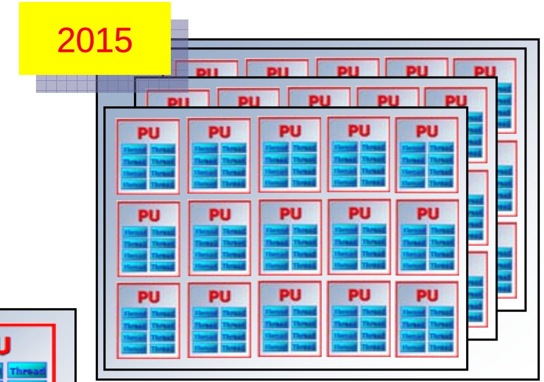
Many-Core HPC: Cores, Threads and SIMD

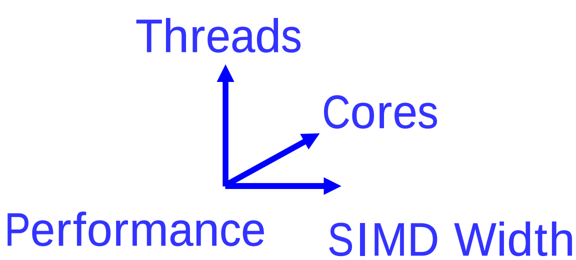
HEP: cope with high data rates !

Cores and Threads realize the task level of parallelism

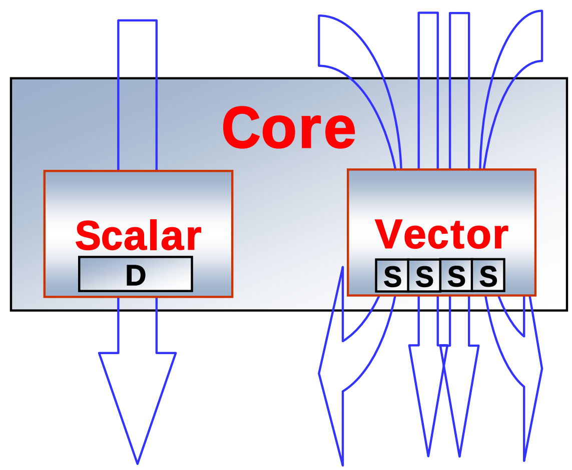








Fundamental redesign
of traditional approaches to data processing
is necessary



Vectors (SIMD) = data level of parallelism

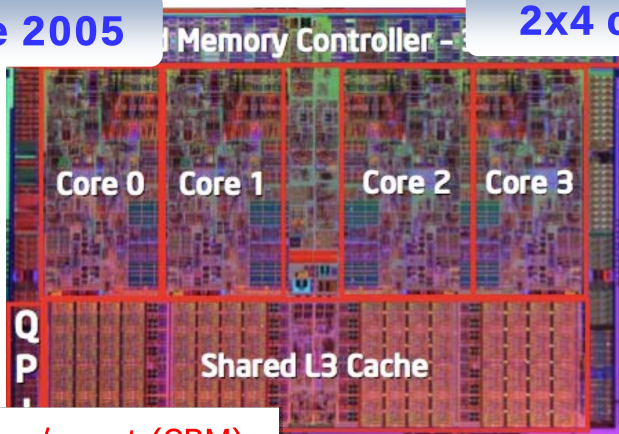
SIMD = Single Instruction, Multiple Data

Our Experience with Many-Core CPU/GPU Architectures

Intel/AMD CPU

Since 2005

2x4 cores

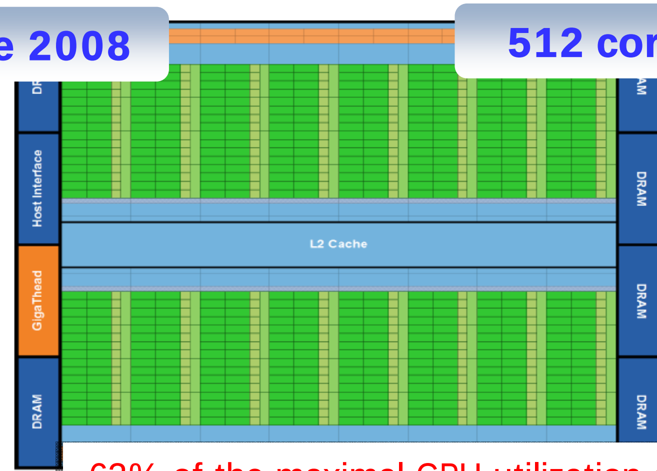


6.5 ms/event (CBM)

NVIDIA GPU

Since 2008

512 cores

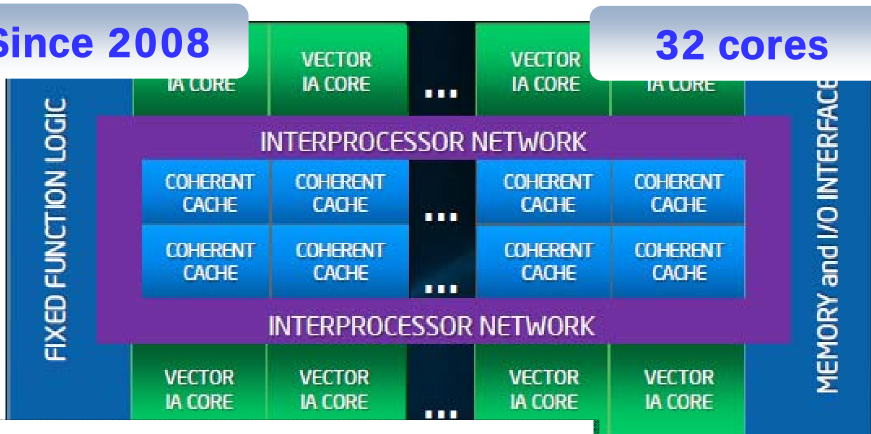


63% of the maximal GPU utilization (ALICE)

Intel MICA

Since 2008

32 cores

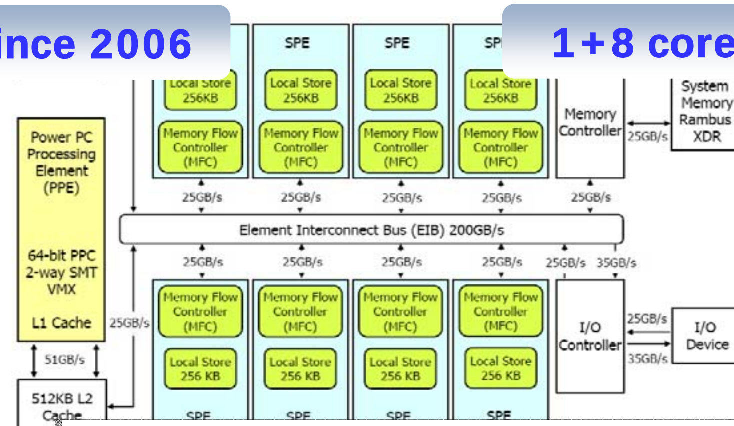


Cooperation with Intel (ALICE/CBM)

IBM Cell

Since 2006

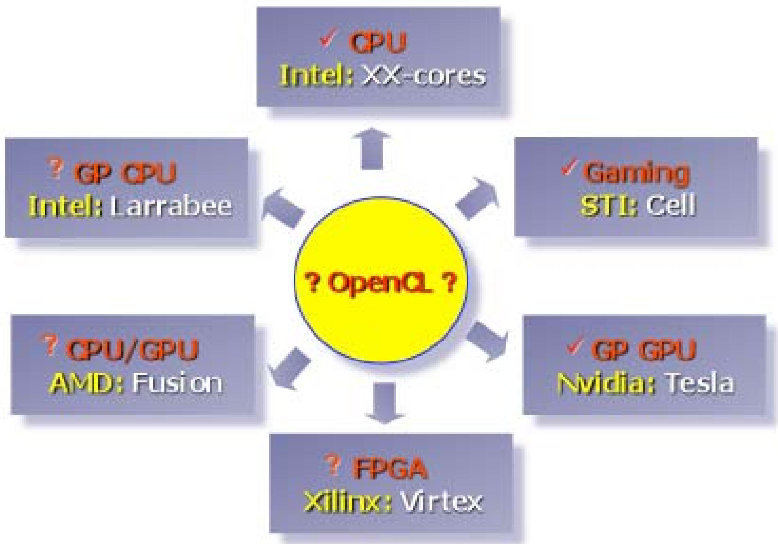
1+8 cores



70% of the maximal Cell performance (CBM)

Future systems are heterogeneous

CPU/GPU Programming Frameworks

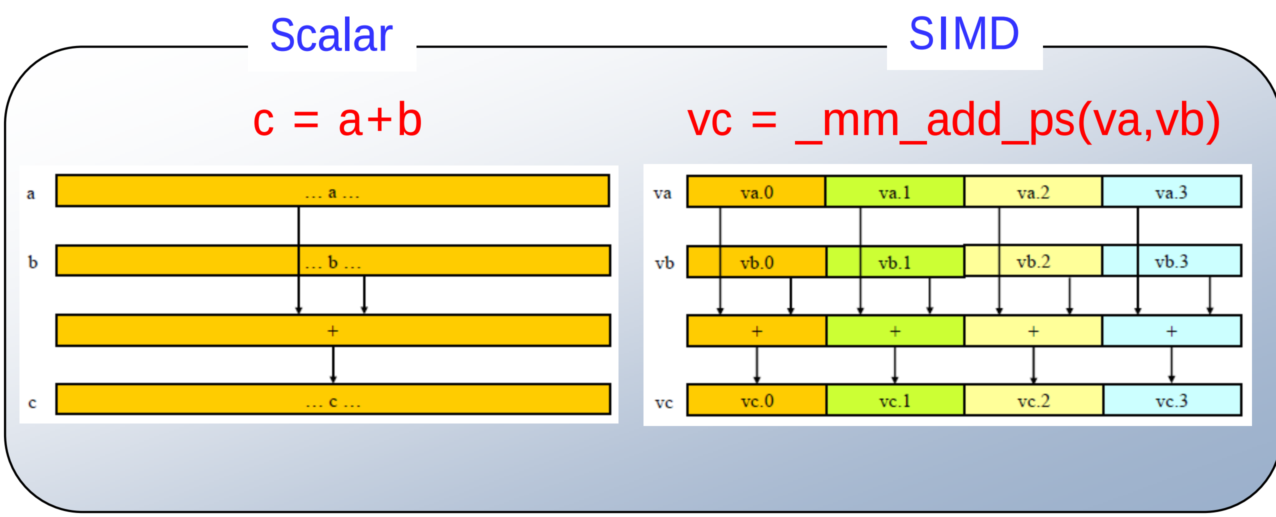


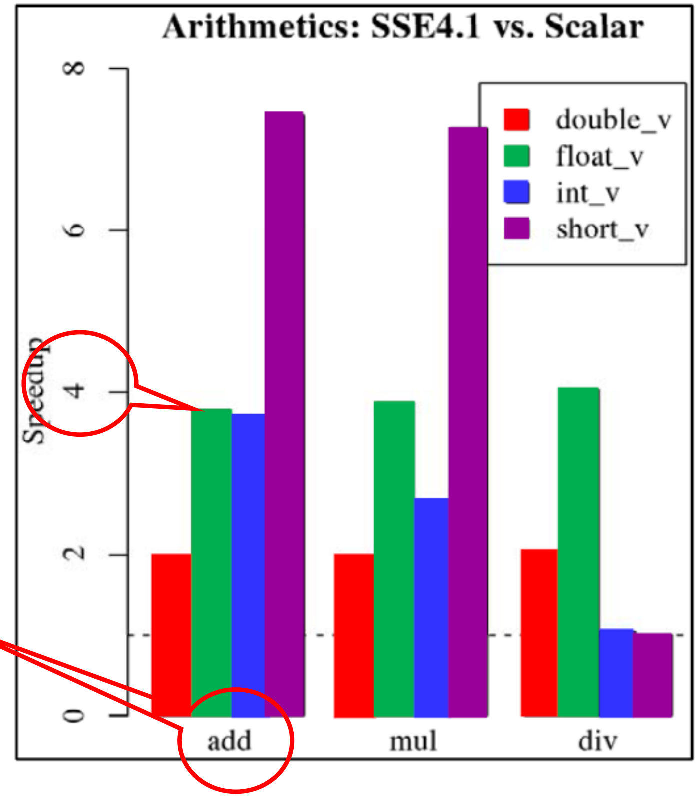
- Intel Ct (C for throughput)
  - Extension to the C language
  - Intel CPU/GPU specific
  - SIMD exploitation for automatic parallelism
- NVIDIA CUDA (Compute Unified Device Architecture)
  - Defines hardware platform
  - Generic programming
  - Extension to the C language
  - Explicit memory management
  - Programming on thread level
- OpenCL (Open Computing Language)
  - Open standard for generic programming
  - Extension to the C language
  - Supposed to work on any hardware
  - Usage of specific hardware capabilities by extensions
- Vector classes (Vc)
  - Overload of C operators with SIMD/SIMT instructions
  - Uniform approach to all CPU/GPU families
  - Uni-Frankfurt/FIAS/GSI

Vector classes: Cooperation with the Intel Ct group

Vector Classes (Vc)

Vector classes overload scalar C operators with SIMD/SIMT extensions





Vector classes:

- provide full functionality for all platforms
- support the conditional operators

phi(phi<0)+=360;

Vc increase the speed by the factor:

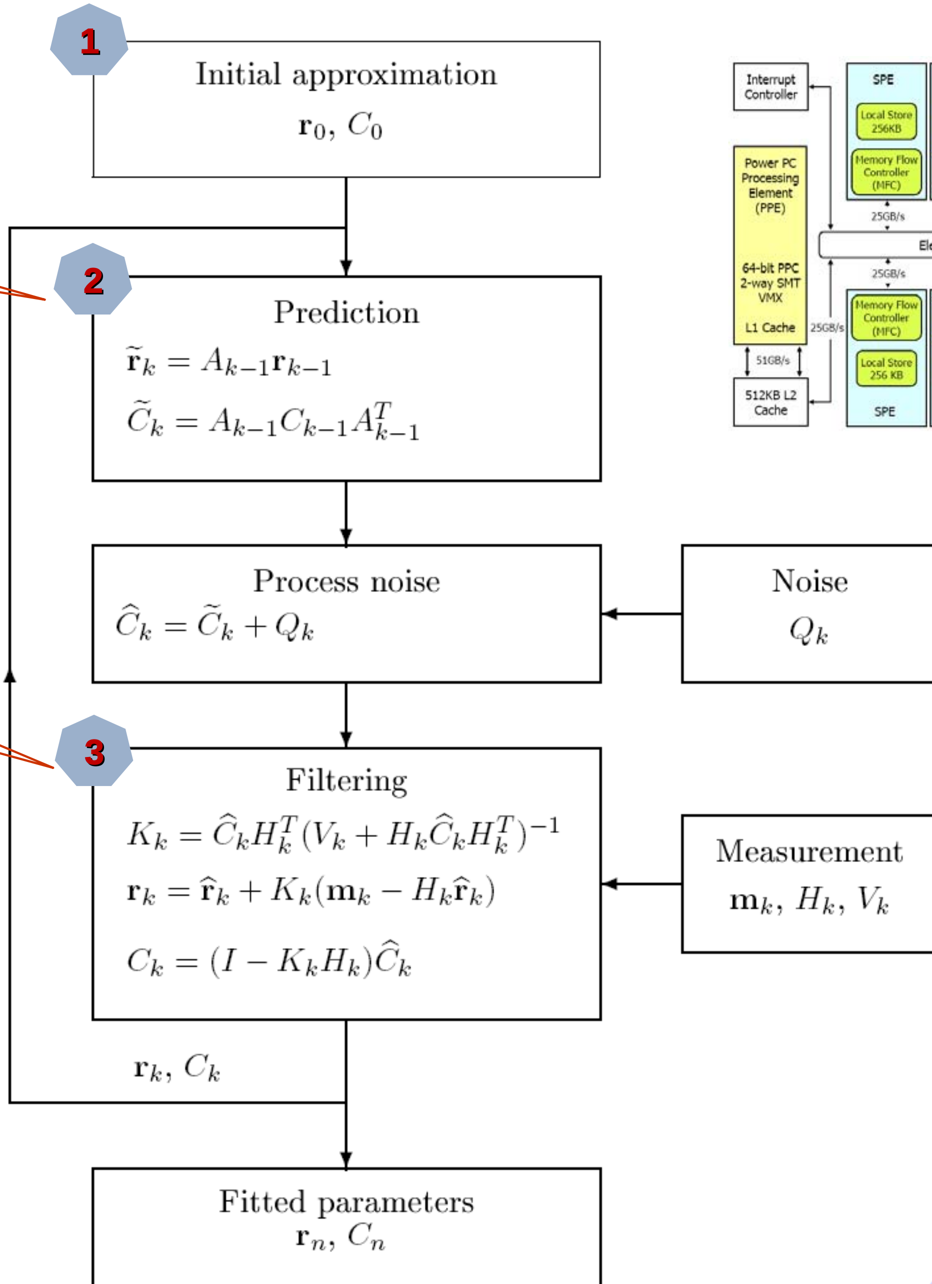
- ✓ SSE2 – SSE4 4x
- ✓ future CPUs 8x
- ✓ MICA/Larrabee 16x
- ➢ NVIDIA Fermi research

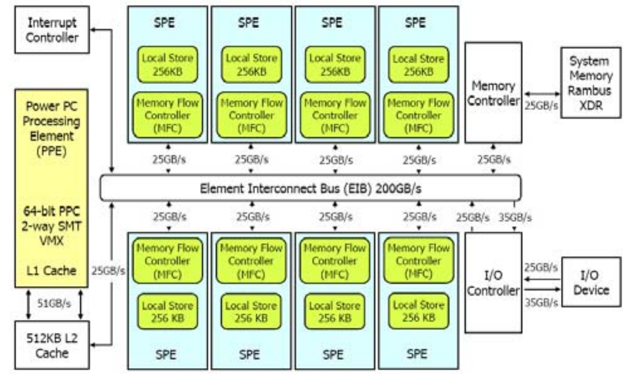
Vector classes enable easy vectorization of complex algorithms

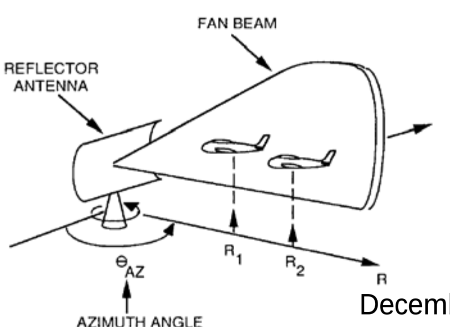
Parameterization of the magnetic field

KF was considerably reworked

Optimization of the algorithm







Apollo Flight Journal

December 21, 1968. The Apollo 8 spacecraft has just been sent on its way to the Moon.
003:46:31 Collins: Roger. At your convenience, would you please go P00 and Accept? We're going to update to your W-matrix.

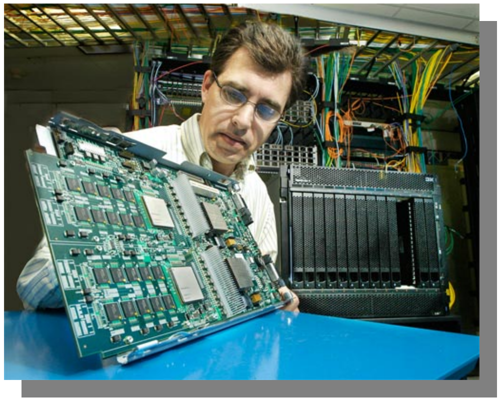
Kalman Filter Track Fit on Cell

Stage	Description	Time/track	Speedup
	Initial scalar version	12 ms	–
1	Approximation of the magnetic field	240 μs	50
2	Optimization of the algorithm	7.2 μs	35
3	Vectorization	1.6 μs	4.5
4	Porting to SPE	1.1 μs	1.5
5	Parallelization on 16 SPEs	0.1 μs	10
	Final simdized version	0.1 μs	120000

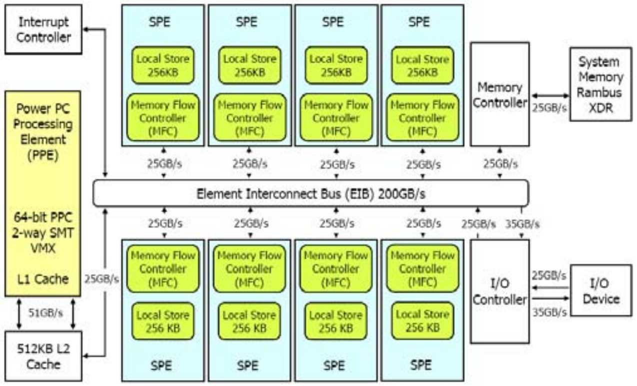
Intel P4 (stages 1–3) · Cell (stages 4–5)

10000x faster on each CPU

Comp. Phys. Comm. 178 (2008) 374-383



The KF speed was increased by 5 orders of magnitude

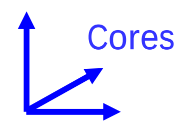


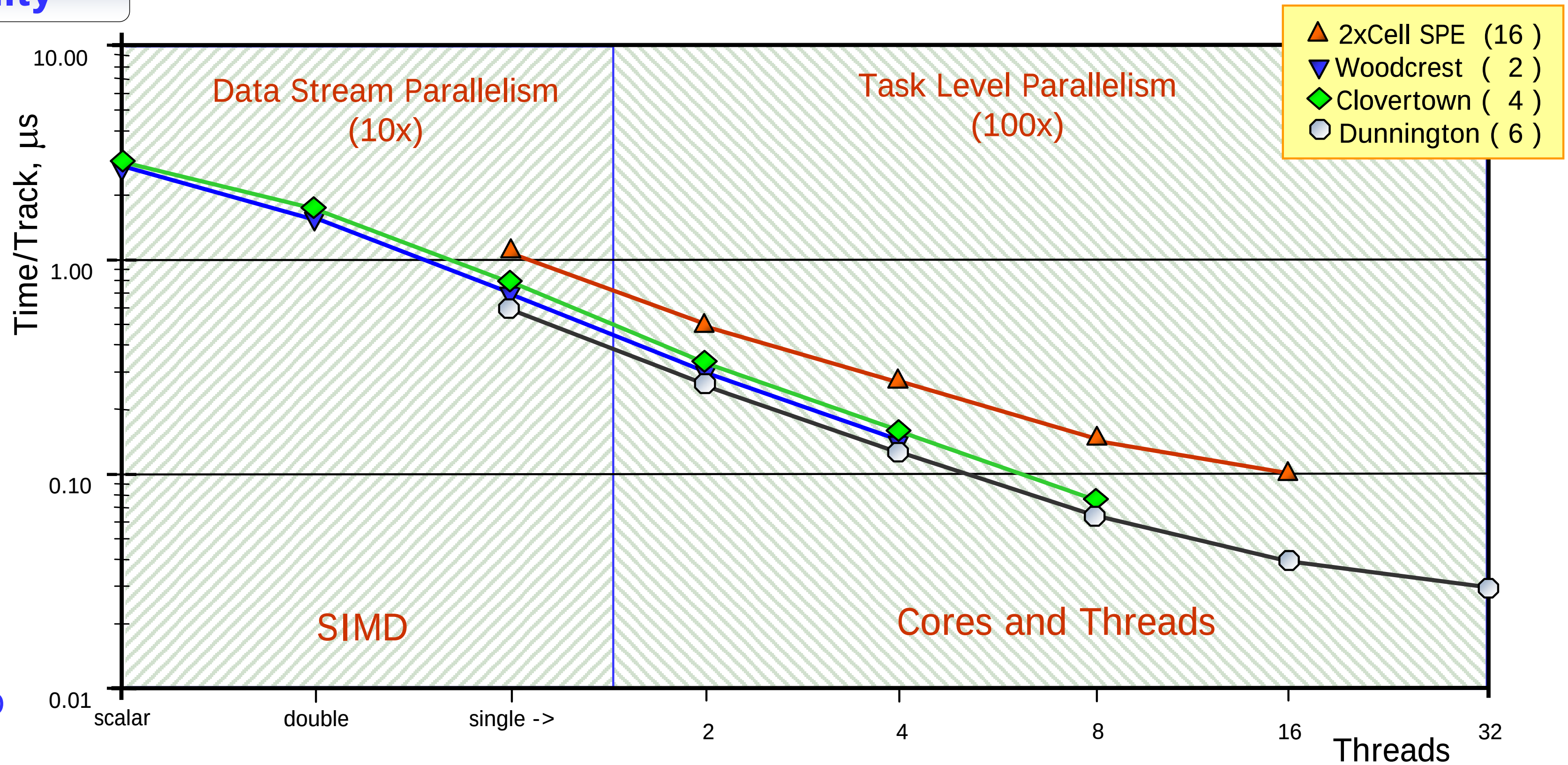
blade11bc4 @IBM, Böblingen: 2 Cell Broadband Engines with 256 kB Local Store at 2.4 GHz

Motivated by, but not restricted to Cell !

Performance of the KF Track Fit on CPU/GPU Systems

Scalabilty





Scalability on different CPU architectures – speed-up 100

CPU

Type	Cores	Clock, GHz	Time/track, ns
Core 2	2	2.66	260
Core i7	8	2.67	52

Real-time performance on different Intel CPU platforms

GPU

NVIDIA Unit	Clock, GHz	Throughput, 10^6 tr./s
8800 GTS 512	1.6	13.0
GTX 280	1.3	21.7

Real-time performance on NVIDIA GPU graphic cards

The Kalman Filter Algorithm performs at ns level

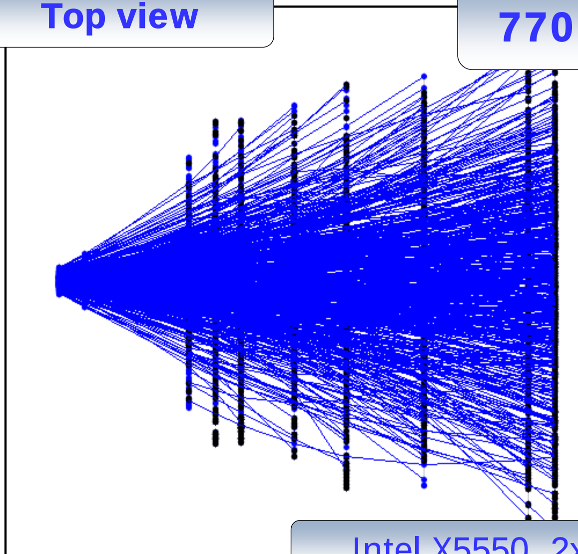
CBM Progr. Rep. 2008

CBM Cellular Automaton Track Finder

Problem

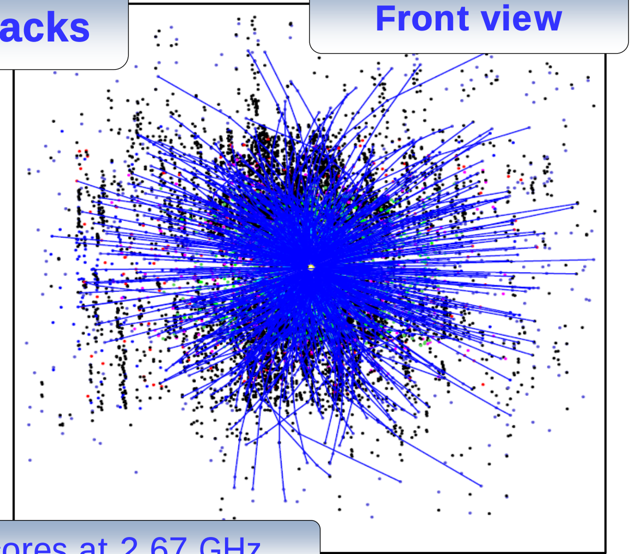
- Fixed-target heavy-ion experiment
- 10^7 Au+Au collisions/s
- 1000 charged particles/collision
- Non-homogeneous magnetic field
- Double-sided strip detectors (85% combinatorial space points)
- Full on-line event reconstruction

Top view



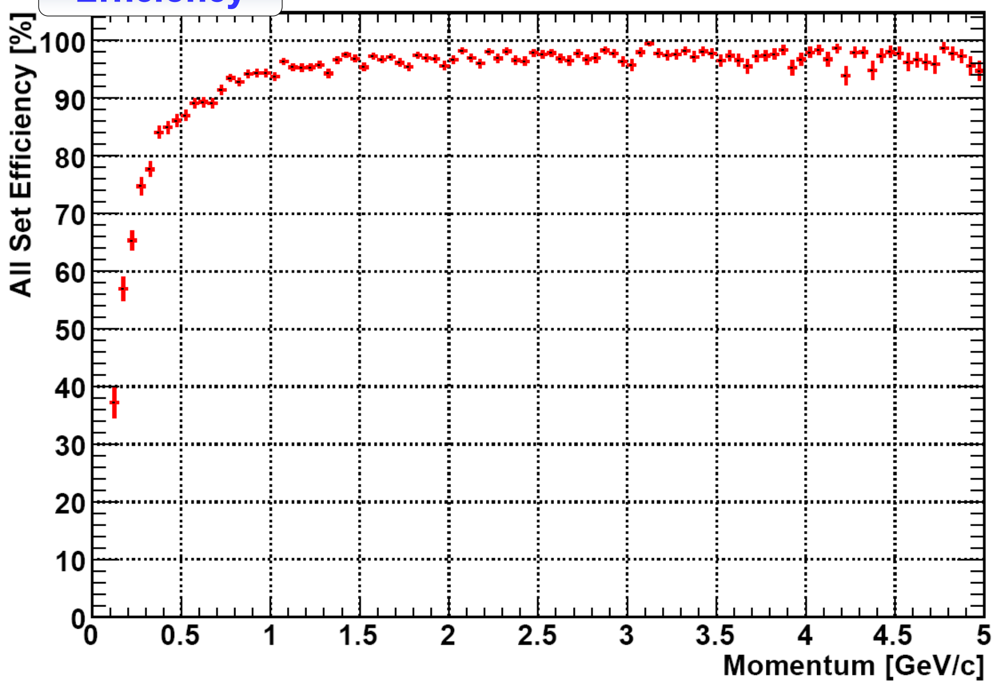
770 Tracks

Front view

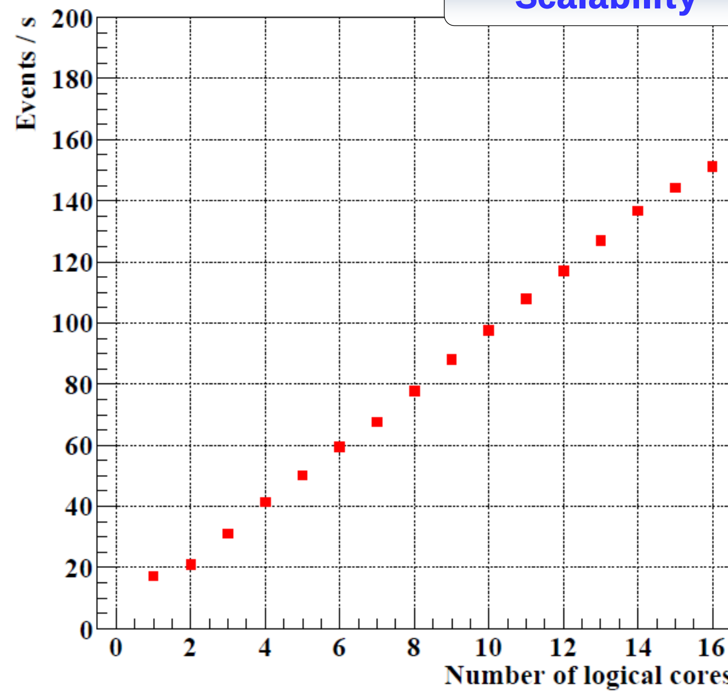


Intel X5550, 2x4 cores at 2.67 GHz

Efficiency



Scalability



Highly efficient reconstruction of 150 central collisions per second

Parallelization is now a Standard in the CBM Reconstruction

Algorithm	Vector SIMD	Multi-Threading	NVIDIA CUDA	OpenCL	Time/PC
STS Detector	+	+	+	+	**6.5 ms**
Muon Detector	+	+			**1.5 ms**
TRD Detector	+	+			**1.5 ms**
RICH Detector	+	+			**3.0 ms**
Vertexing	+		Future		**10 μs**
Open Charm Analysis	+				**10 μs**
User Reco/Digi		Future			
User Analysis					

+ 2009
+ 2010

The CBM reconstruction is at ms level

Intel X5550, 2x4 cores at 2.67 GHz

Workshop for Future Challenges in Tracking and Trigger Concepts

June 7-11, 2010, GSI, Darmstadt, Germany

HIC for FAIR – Helmholtz International Center

HadronPhysics2

Topics:

Fixed-Target Experiments (CBM, HADES, PANDA)

Collider Experiments (ALICE, STAR)

Reconstruction Methods (Finding/Fitting)

Computer Architectures (CPU/GPU)

Software Architectures (Framework/Standalone)

Training:

Vector Classes/SIMD

Multi-Threading

Intel's Ct

CUDA/OpenCL

45 participants from Austria, China, Germany, India, Italy, Norway, Russia, Switzerland, UK and USA

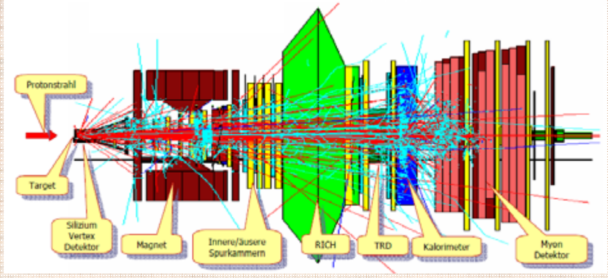
Workshop Program

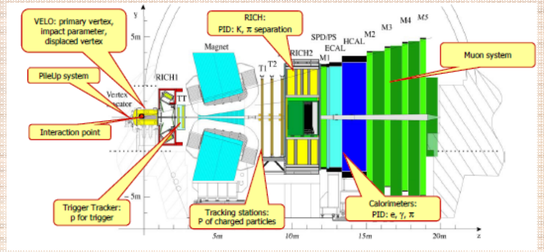
Wednesday 09.06	Thursday 10.06	Friday 11.06
09:00-13:00 Fixed-Target Experiments 09:00 P. Senger CBM Experiment 09:10 C. Höhne CBM Physics 09:30 V. Friese CBM Software 09:50 I. Kisel CBM Reconstruction 10:10 J. Markert HADES Tracking 10:30 R. Karabowicz PANDA Tracking 11:00 --- Coffee break --- 11:30 Discussion	09:00-13:00 Software Architectures 09:00 V. Friese CBM Framework 09:20 A. Lebedev CBM MUCH Tracking 09:40 S. Lebedev CBM Ring Finder 10:00 J. Lauret STAR Framework 10:30 S. Gorbunov CA HLT Tracking 11:00 --- Coffee break --- 11:30 I. Kulakov Porting CA to STAR 11:50 M. Zyzak CA Merger 12:10 Discussion	09:00-13:00 Reconstruction Methods 09:00 D. Rohr ALICE GPU Tracking 09:20 I. Kisel Track Finding 10:00 R.Frühwirth Adaptive Methods 10:45 I. Kulakov SIMD KF Track Fit 11:00 --- Coffee break --- 11:30 M. Bach GPU KF Track Fit 11:45 M. Zyzak KFParticle 12:00 Discussion
14:00-18:00 Collider Experiments 14:00 V. Lindenstruth ALICE Experiment 14:10 J. Thäder ALICE HLT 14:50 J. Lauret STAR Experiment 15:10 Y. Fisyak STAR Tracking 15:30 H. Qiu STAR HLT 16:00 --- Coffee break --- 16:30 Discussion	14:00-18:00 Computer Architectures 14:00 S. Jarp Future CPU/GPU 14:30 K.-D. Örtel Intel CPU 15:00 H. Pabst Intel Ct 15:30 M. Al-Turany GPU Tracking 16:00 --- Coffee break --- 16:30 M. Kretz Vc Classes 16:50 I. Kulakov CBM CA Track Finder 17:10 Discussion 18:30 --- Dinner ---	14:0 0-18:00 General Discussion, Future Plans 14:00 Discussion 16:00 --- Coffee break --- 16:30 Discussion 17:00 I. Kisel Summary

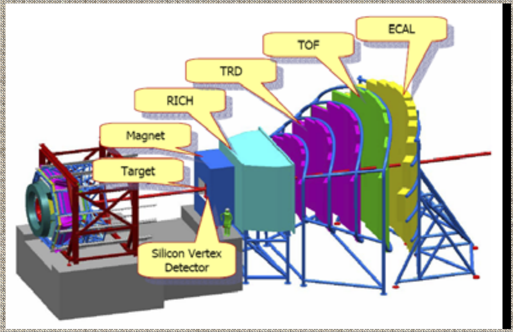
Software Evolution: Many-Core Barrier

Scalar single-core OOP

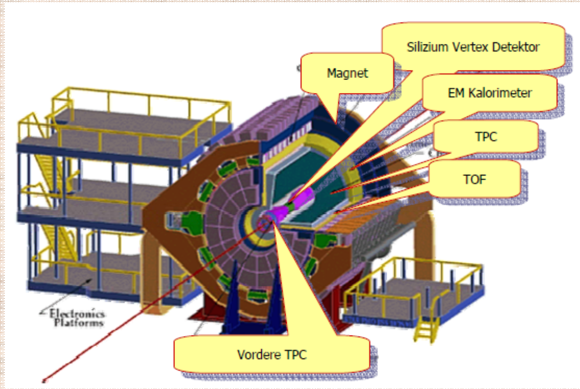
Many-core HPC era

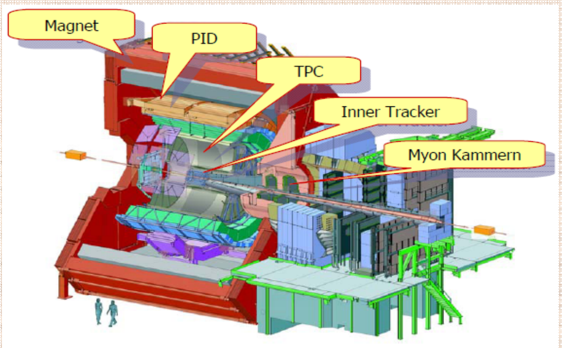






1990 2000 2010 t



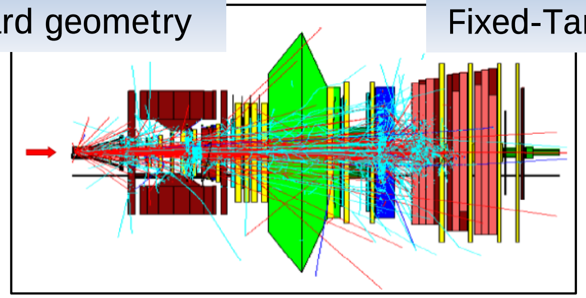


Consolidate efforts of:
- Physicists
- Mathematicians
- Computer scientists
- Developers of parallel languages
- Many-core CPU/GPU producers

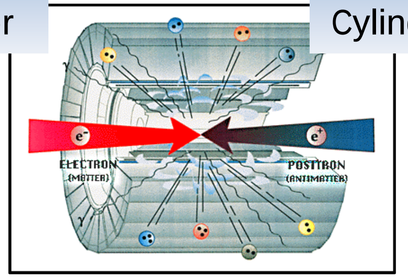
1990 2000 2010 t

Software redesign can be synchronized between the experiments

Forward geometry | Fixed-Target

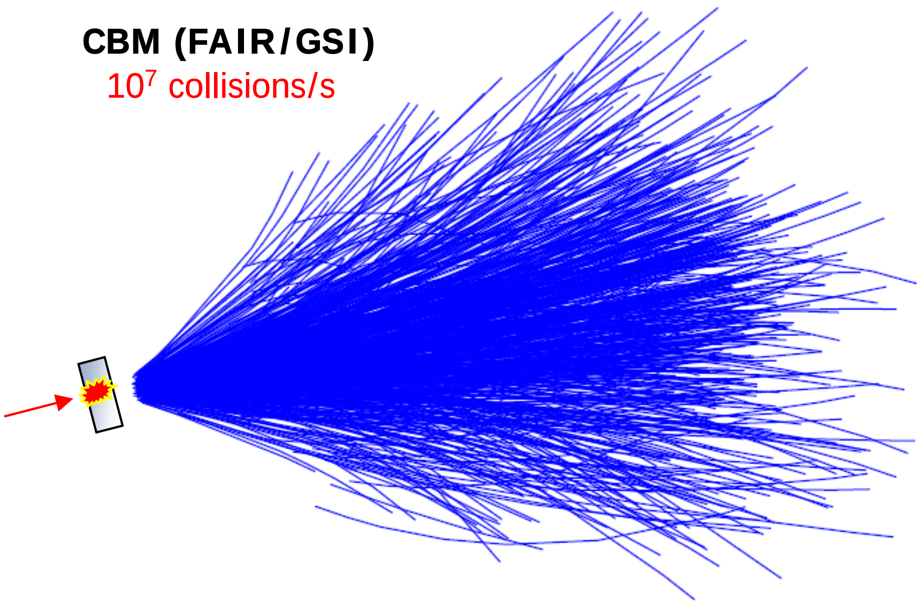


Collider | Cylindrical geometry



CBM (FAIR/GSI)

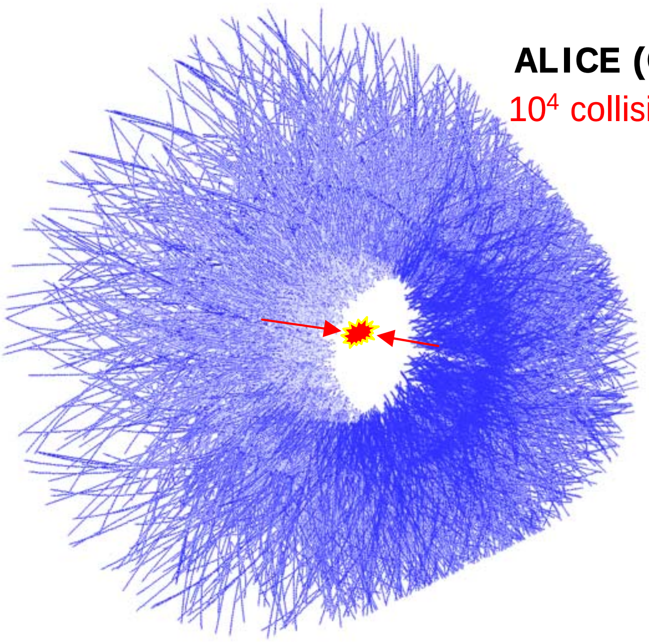
10^7 collisions/s



Intel CPU 8 cores (CBM Reco Group)

ALICE (CERN)

10^4 collisions/s



NVIDIA GPU 240 cores (ALICE HLT Group)

Different experiments have similar reconstruction problems

Track reconstruction is the most time consuming part of the event reconstruction, therefore many-core CPU/GPU platforms.

Track finding is based in both cases on the Cellular Automaton method,
track fitting – on the Kalman Filter method.

Stages of Event Reconstruction: To-Do List

Detector dependent

- Generalized track finder(s)
- Geometry representation
- Interfaces
- Infrastructure

Track model dependent

- Kalman Filter
- Kalman Smoother
- Deterministic Annealing Filter
- Gaussian Sum Filter
- Field representation

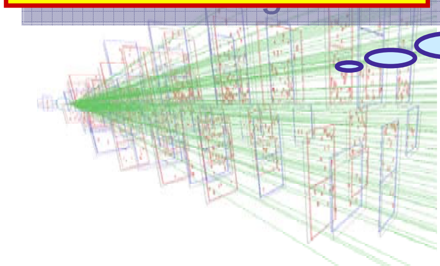
Detector/geometry independent

- 3D Mathematics
- Adaptive filters
- Functionality
- Physics analysis

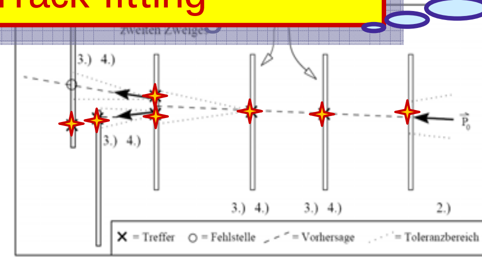
RICH specific

- Ring finders

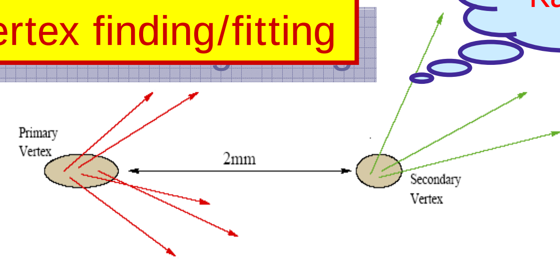
Track finding — Time consuming!!!



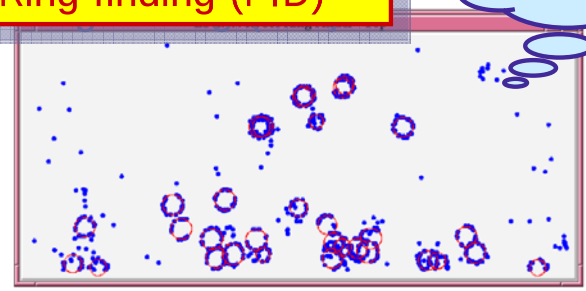
Track fitting — Kalman Filter



Vertex finding/fitting — Kalman Filter



Ring finding (PID) — Combinatorics



Consolidate Efforts: Common Reconstruction Package

Uni-Frankfurt/FIAS:
Vector classes
GPU implementation

GSI:
Algorithms development
Many-core optimization

OpenLab (CERN):
Many-core optimization
Benchmarking

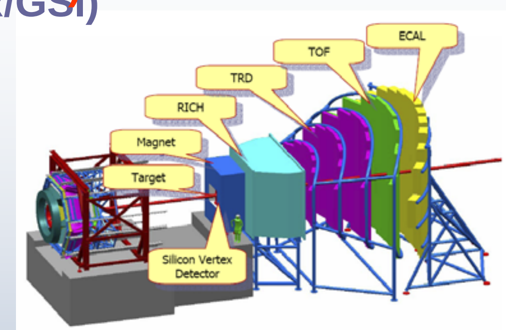
HEPHY (Vienna)/Uni-Gjovik:
Kalman Filter track fit
Kalman Filter vertex fit

Common Reconstruction Package

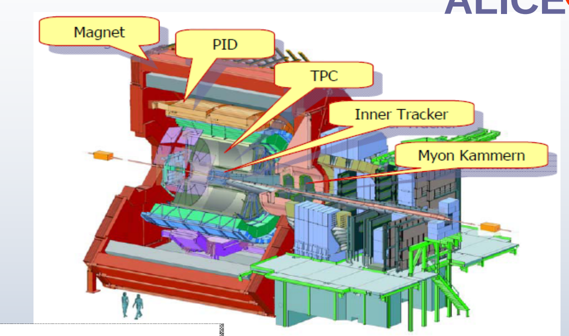
Intel:
Ct implementation
Many-core optimization
Benchmarking

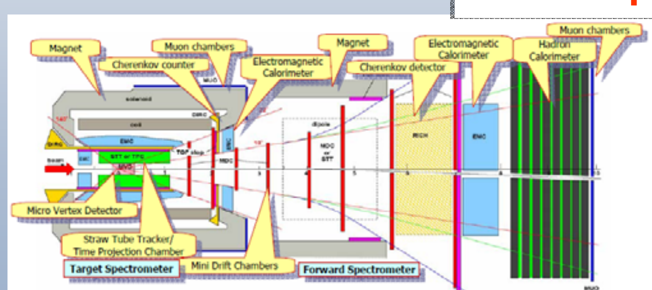
Host Experiments:

CBM (FAIR/GSI)

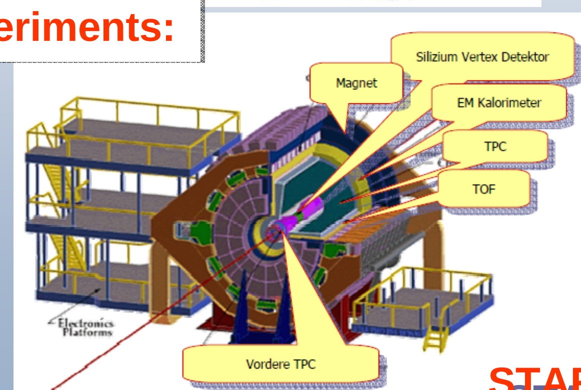


ALICE (CERN)





PANDA (FAIR/GSI)



STAR (BNL)

Workshop for Future Challenges in Tracking and Trigger Concepts

Follow-up Workshop: November 2010 – February 2011 at GSI or CERN or BNL ?

Topics:

Fixed-Target Experiments (CBM, HADES,PANDA)

Collider Experiments (ALICE, STAR)

Reconstruction Methods (Finding/Fitting)

Computer Architectures (CPU/GPU)

Software Architectures (Framework/Standalone)

Training:

Vector Classes/SIMD

Multi-Threading

Intel's Ct

CUDA/OpenCL