Outline of the lectures

- ***Filtering***
- ***Calibration and alignment***
- ***Event Reconstruction***
- ***Event Simulation***
- ***Physics Analysis***

What is Physics Computing?

- Input: A few petabytes of data (at most)
- Output: A few hundred papers (at least)
- Not yet *fully* automatized
- What happens to the data?

"HABENT SUA FATA DATA"

after Terentius Maurus, 2nd century AD

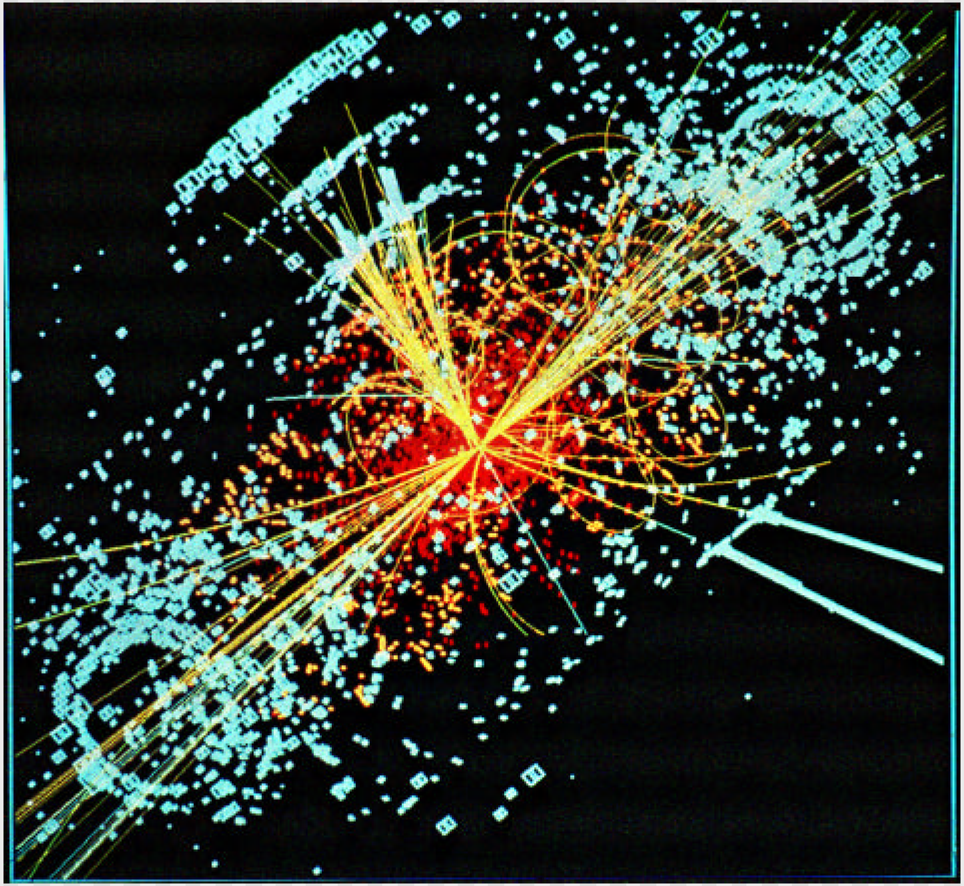
What happens to the data?

- Data filtering and storage
- Conversion, calibration, alignment
- Event reconstruction
- Event simulation
- Physics analysis
- In each step they get closer to be interpretable in physical terms

The challenge

- Very high event rate
- Large event size
- Large background of uninteresting events
- Large background in each event
  - many interactions in each BX
  - many low-momentum particles
- Large number of physicists doing analysis

The challenge



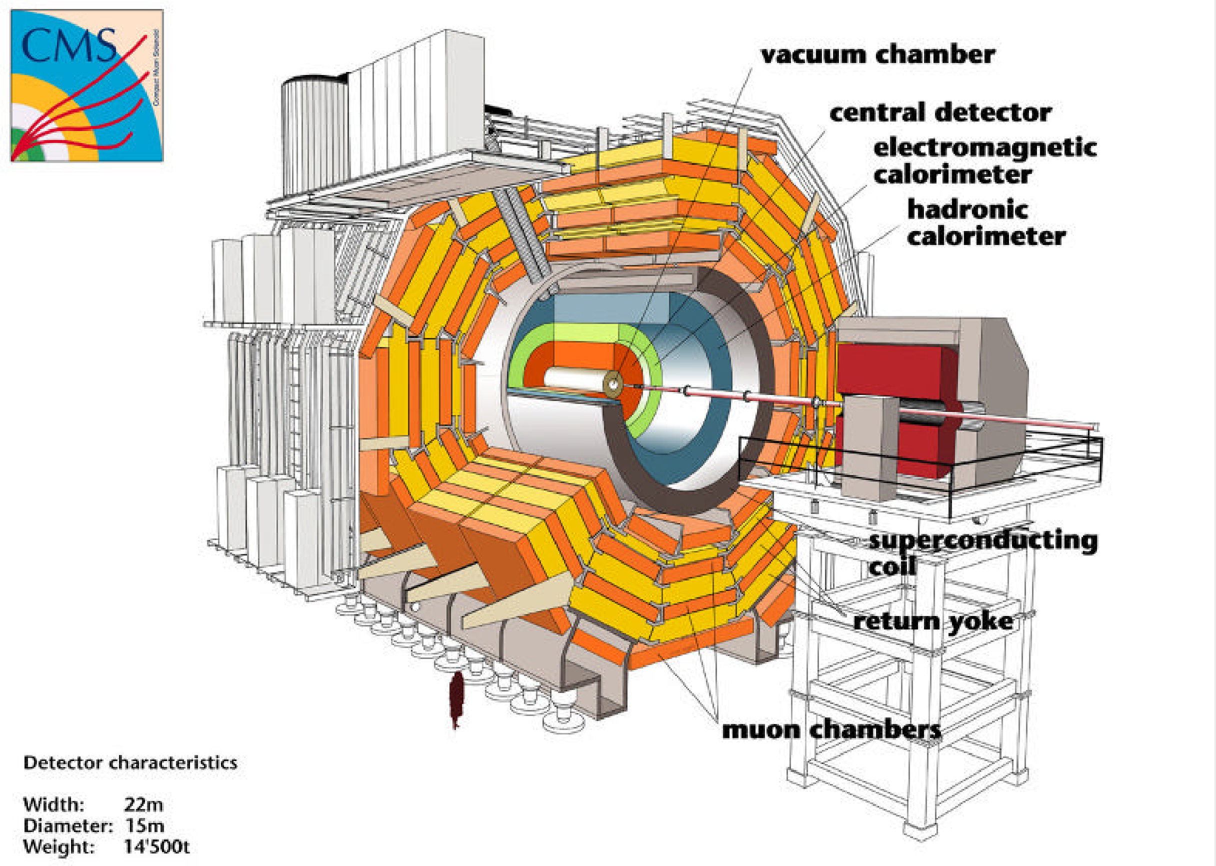
Data filtering

- Primary collision rate: 40 Megahertz
- Recording rate: 100 Hertz
- How is this achieved?
  - Multilevel trigger
  - Very fast first level: (Programmable) hardware
  - Slower second level: Software on fast processors
- Reliable: Rejected data are lost forever
- Conservative: Do not lose new physics

Multilevel trigger

- Dead-time has to be minimized
- Many collisions can be discarded very quickly – Level 1
- Only the surviving ones are scrutinized more carefully – High Level Trigger(s)
- Triggers are tailored to specific physics channels (Higgs, top, WW, ZZ, ...)

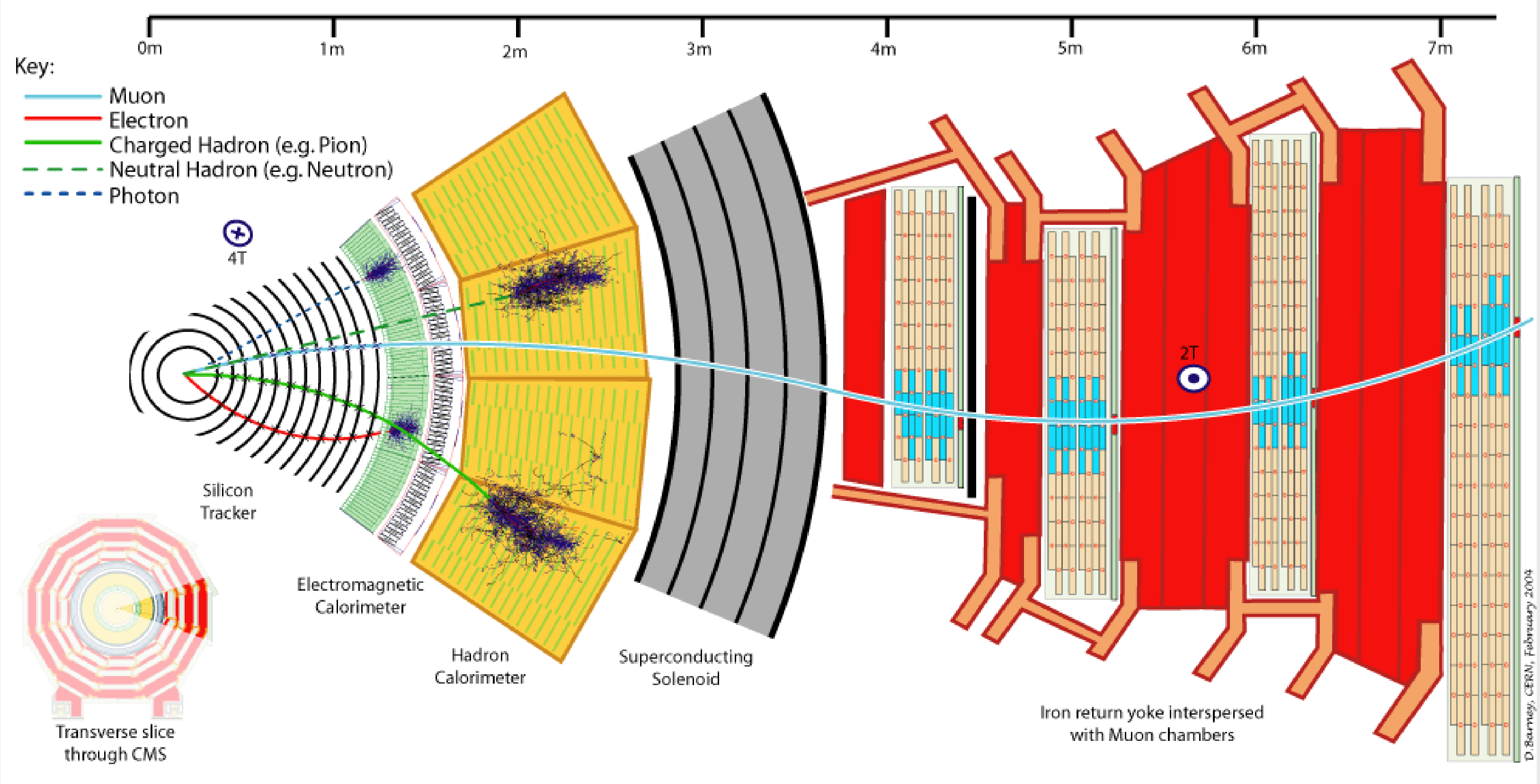
Example: CMS


CMS
vacuum chamber
central detector
electromagnetic calorimeter
hadronic calorimeter
superconducting coil
return yoke
muon chambers
Detector characteristics
Width: 22m
Diameter: 15m
Weight: 14'500t

What CMS subdetectors measure

- Inner tracker (pixels+strips)
  - Momentum and position of charged tracks
- Electromagnetic calorimeter
  - Energy of photons, electrons and positrons
- Hadronic calorimeter
  - Energy of charged and neutral hadrons
- Muon system
  - Momentum and position of muons

What CMS subdetectors measure



CMS L1 trigger

- Input rate: 40 megahertz
- Output rate: 30 – 100 kilohertz
- Latency: 3.2 μs (128 BX)
- Pipelined, dead-time < 1%
- Available time for calculations: 1.25 μs
- 2 detector systems: muons/calorimeters
- 3 main steps: local/regional/global

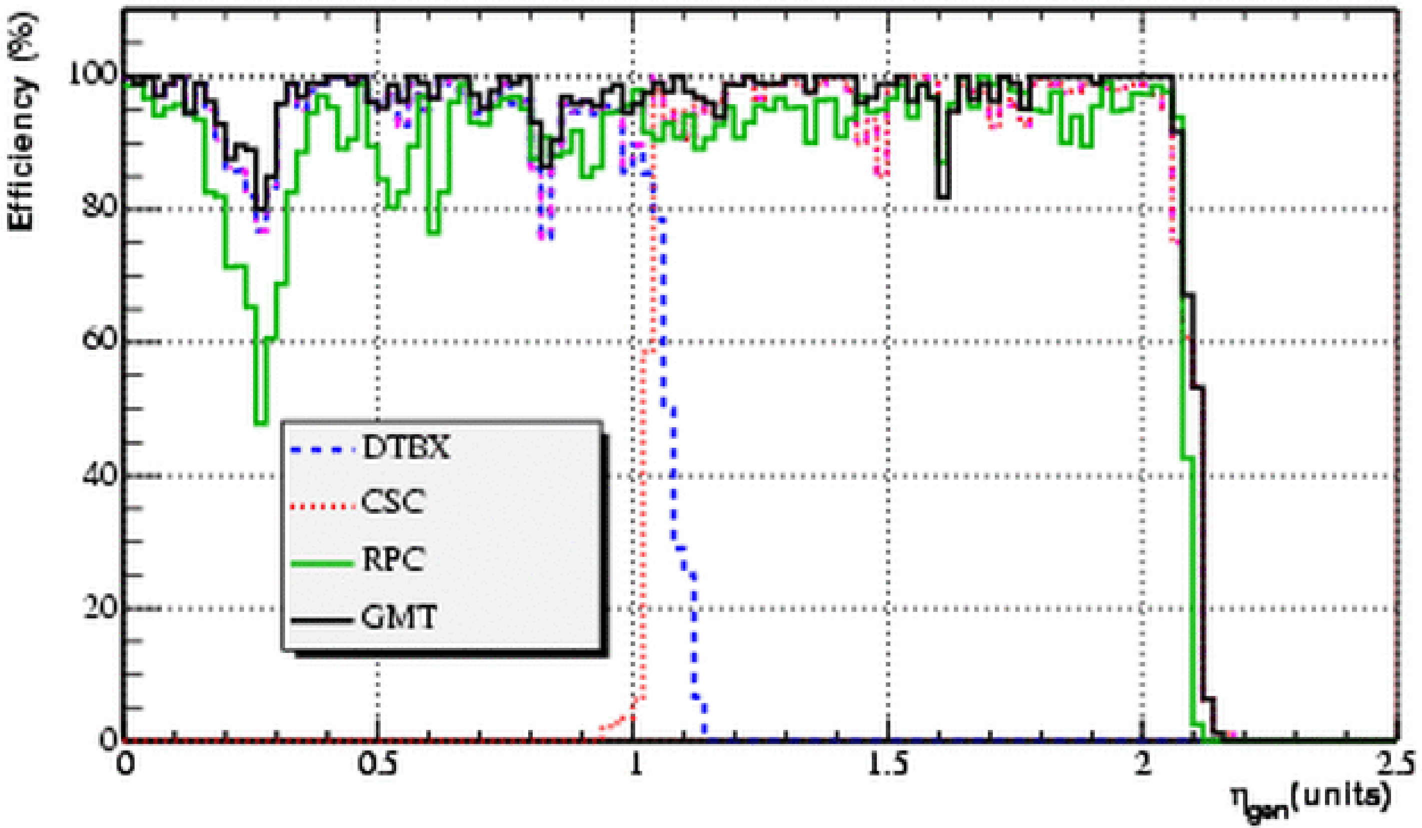
CMS L1 calorimeter trigger

- Calorimeter trigger:
  - Two types of calorimeters
  - Local: Computes energy deposits
  - Regional: Finds candidates for electrons, photons, jets, isolated hadrons; computes transverse energy sums
  - Global: Sorts candidates in all categories, does total and missing transverse energy sums, computes jet multiplicities for different thresholds

CMS L1 muon trigger

- Muon trigger:
  - Three types of muon detectors
  - Local: Finds track segments
  - Regional: Finds tracks
  - Global: Combines information from all regional triggers, selects best four muons, provides energy and direction

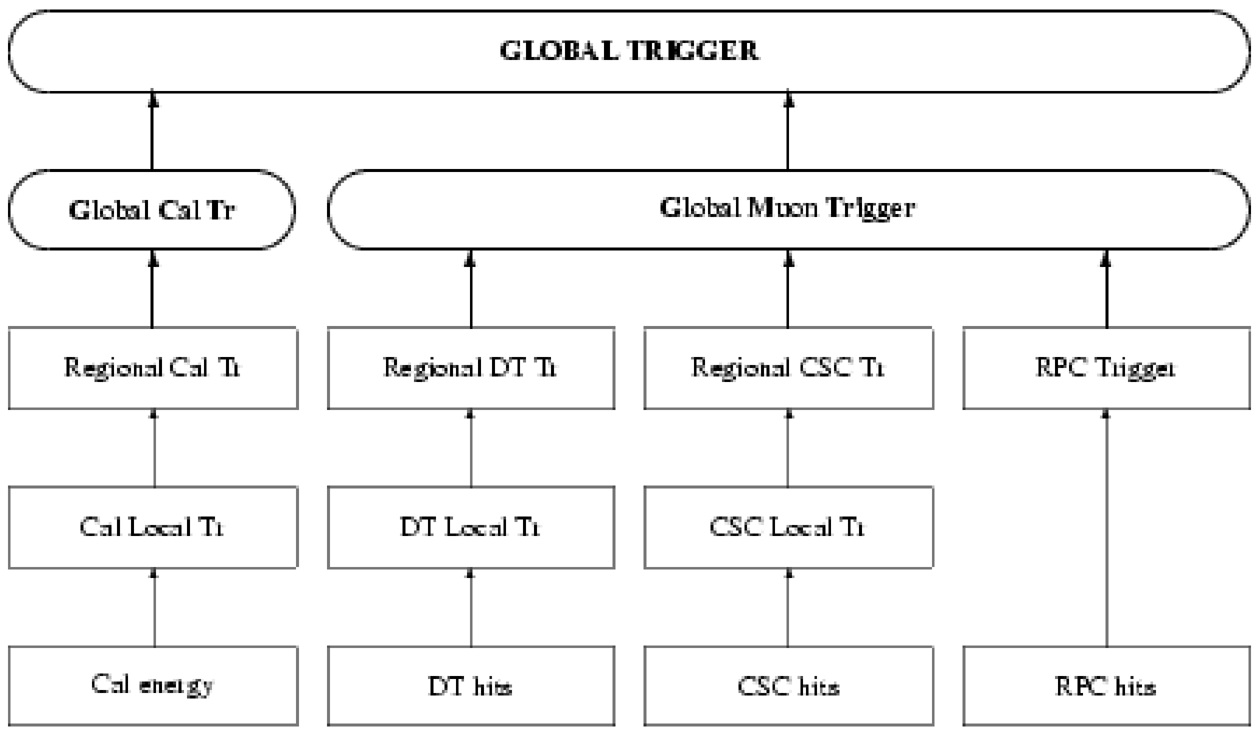
Efficiency of global muon trigger



CMS L1 global trigger

- Global trigger:
  - Final decision logic
  - 28 input channels (muons, jets, electrons, photons, total/missing E_T)
  - 128 trigger algorithms running in parallel
  - 128 decision bits
  - Apply conditions (thresholds, windows, deltas)
  - Check isolation bits
  - Apply topology criteria (close/opposite)

CMS L1 trigger



CMS L1 trigger software

- Algorithms are developed in C++
- They are tested by extensive simulation studies (🎭 Event Simulation)
- Manual translation into VHDL (***V***ery high speed integrated circuit ***H***ardware ***D***escription ***L***anguage)
- Comparison with C++ implementation

High level trigger

- Further data filtering:
  - 30 – 100 kilohertz input rate
  - 100 Hertz output rate
- Event tagging:
  - Reconstruct physics objects
  - Mark events having interesting features

High level trigger (cont)

- More detailed analysis of event and underlying physics
- Runs on standard processors (commodity PCs)
- Algorithms are similar to the ones used in event reconstruction (🎭 Event Reconstruction), but optimized for speed.

High level trigger (cont)

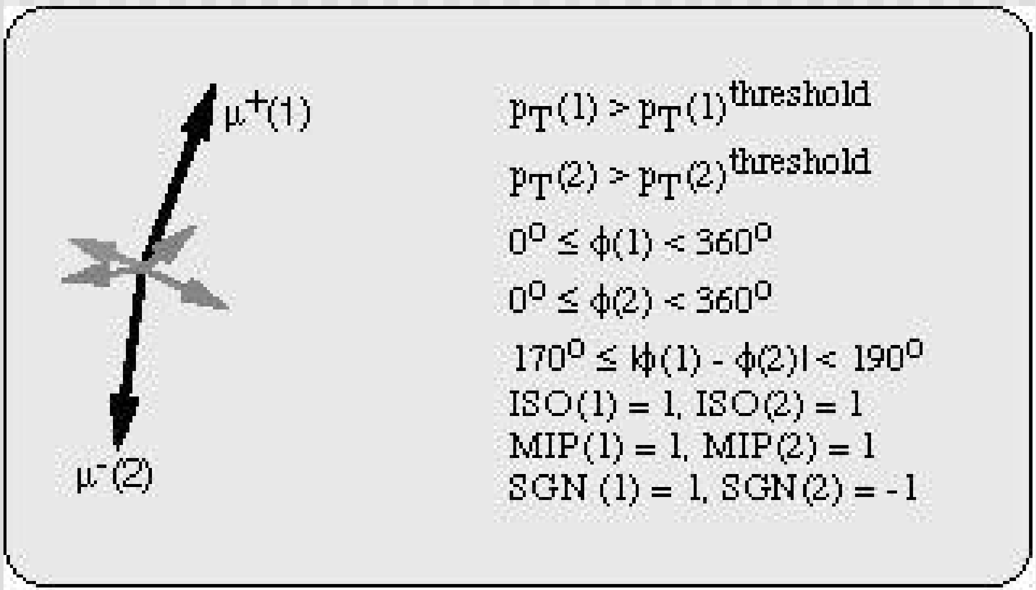
- Regional reconstruction
  - Concentrates on region(s) found by Level 1
  - Muons, electrons, jets, …
- Partial reconstruction
  - Abandon goal of optimal precision
  - Stop as soon specific questions are answered

CMS High level trigger

- Has to keep pace with the L1 Output
- Filter farm with about 1M SpecInt95
  - About 25000 Pentium III @ 1 GHz
  - About 2000 CPUs at startup (2007)
- Organized in subfarms
- Same software framework as in “offline” reconstruction
- Transparent exchange of algorithms

CMS HLT example

- back-to-back opposite sign isolated muons



After the high level trigger

- Data are written to mass storage
- 1 Megabyte @ 100 Hertz = 100 Megabyte/sec
- LHC runs for $\approx 2*10^7$ sec/year
- 2 Petabyte per year

From bits to GeV and cm

- Raw data are mostly ADC or TDC counts
- They have to be converted to physical quantities like energy or position
- Very detector dependent
- Every detector needs calibration
- Calibration constants need to be stored and updated

Calorimeter calibration

- Kinetic energy of incoming particle is converted into light or electric charge
- Destructive measurement
- Relation between deposited charge and energy needs to be known
- Long term drifts need to be monitored
- Huge amounts of data are accumulated

Silicon Tracker calibration

- Incoming particle creates electric charge in strips or pixels
- Charge distribution depends on location of crossing point and crossing angle
- Solve inverse problem: reconstruct crossing point from charge distribution and crossing angle
- Test beam, real data

Drift tube calibration

- Incoming particle ionizes gas in tube
- Electrons/ions drift to anode/cathode
- Drift time is measured
- Must be converted to drift distance
- Time/distance relation must be determined (not always linear)
- Test beam, real data

Where are the detectors?

- Tracking detectors are very precise instruments
- Silicon strip detector: ~ 50 µm
- Pixel detector: ~ 10 µm
- Drift tube: ~ 100 µm
- Position needs to be known to a similar precision

Example: CMS tracker

Alignment

- Mechanical alignment
- Measurements taken before assembly
- Switching on the magnetic field
- Laser alignment
- Alignment with charged tracks from collisions, beam halo and cosmic rays
- Continuous process

Environmental data

- Calibration data
- Alignment data
- Temperatures, gas pressures, …
- Machine parameters
- Need to be made persistent

Detector related software

- Configuration
  - Load trigger files, set thresholds, set HV, set amplifier gains, …
- Slow control
  - Measure temperature, gas pressure, dark currents, …
- Monitoring
  - Check trigger rates, detector efficiency, cluster sizes, wire maps,

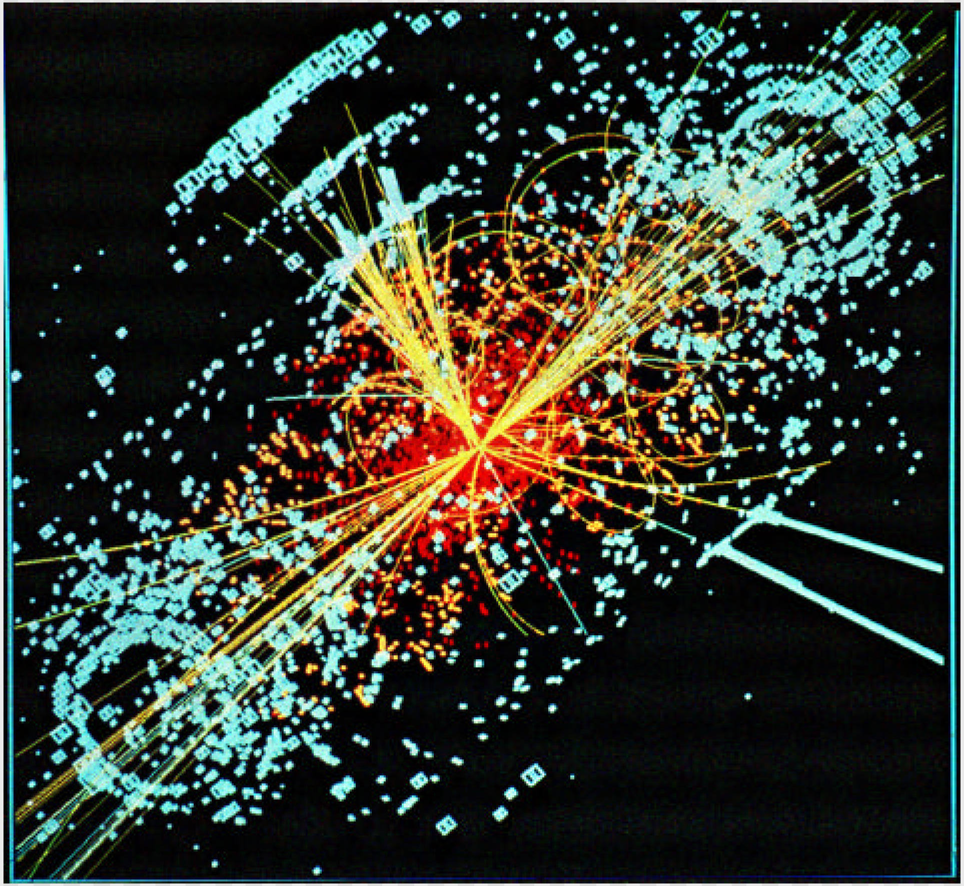
Event reconstruction

- Find out which particles have been created where and with which momentum
- Some of them are short-lived and have to be reconstructed from their decay products
- Some of them (neutrinos) escape without leaving any trace

Event reconstruction (cont)

- Reconstruct charged particles
- Reconstruct neutral particles
- Identify type of particles
- Reconstruct interaction points (vertices)
- Reconstruct kinematics of the interaction
- Not trivial …

CMS: Higgs decay into two jets



Charged particles

- Charged particles are detected by tracker and calorimeters
- Muons also reach the muon system
- Very high number of low-momentum charged particles
- Select by threshold on transverse momentum

Neutral particles

- Neutral particles are detected mainly by calorimeters (e.g. photons, neutrons)
- Some of them decay into two (or more) charged tracks which are detected by the tracker (e.g. K^0)
- Some of them escape without leaving a trace (neutrinos)

Reconstruction of charged particles

- Trajectory is curved because of the magnetic field
- Position is measured in a number of places –“hits”
- Determine track parameters (location, direction, momentum) from the position measurements
- Data compression

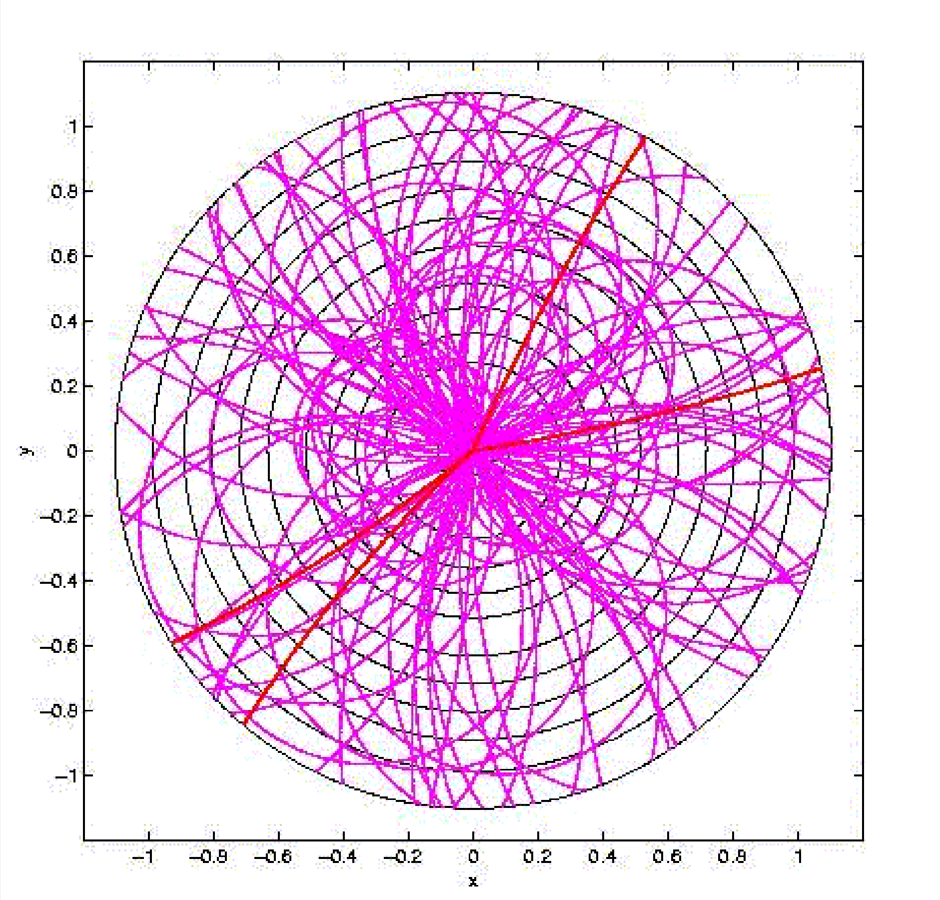
The difficulties

- Charged particles interact with all the material, not only the sensitive parts
- Multiple Coulomb scattering
  - Changes direction, but not momentum
- Energy loss by ionization
  - Changes momentum, but not direction
- Energy loss by bremsstrahlung
  - Only for electrons

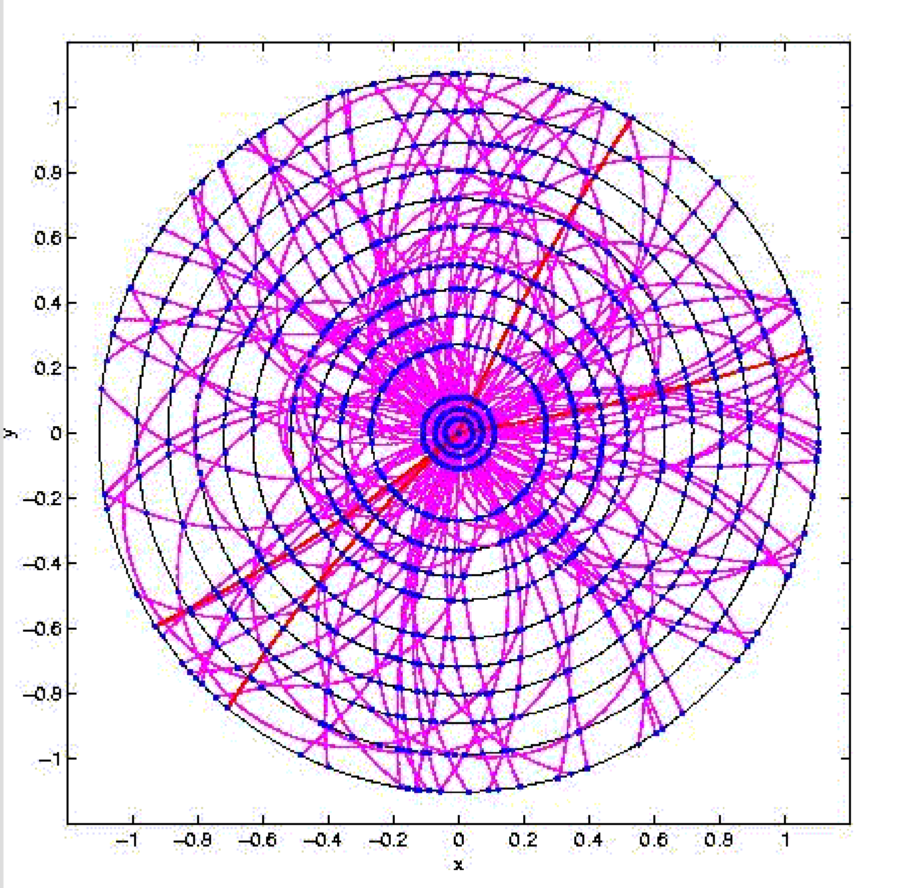
More difficulties

- Assignment of hits to particles is unknown
- Huge background from low-momentum tracks
- Additional background from other interactions in the same beam crossing and from adjacent beam crossings

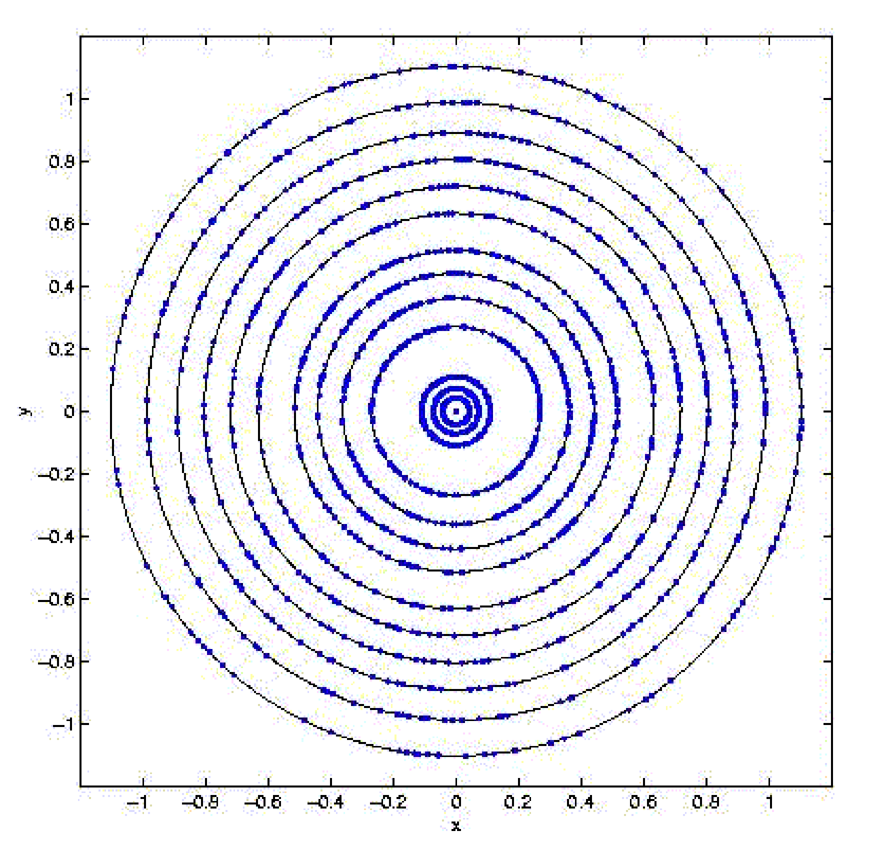
Tracks only



Tracks with hits



Hits only



Decomposition of the problem

- Pattern Recognition or Track Finding
  - Assign hits to track candidates
- Parameter estimation or Track Fit
  - Determine track parameters + covariance matrix
- Test of the track hypothesis
  - Check chi-square, residuals, remove outliers

Track finding

- Depends a lot on the properties of the detector:
  - Geometry, configuration
  - Magnetic field
  - Precision
  - Occupancy
- Many solutions available
- No general recipe

A few track finding algorithms

- Track following
- Simple Kalman filter
- Combinatorial Kalman filter
- Track road
- Hough transform
- Hopfield network

Track following

- Generate an initial track segment (“seed”)
- Extrapolate the seed
- Pick up hits close to extrapolated seed
- Can be done in a projection or in 3d
- Runs into trouble if too many competing hits are close to the extrapolation

Simple Kalman filter

- Generate an initial track segment (“seed”)
- Extrapolate the seed to the next layer
- Pick up closest compatible hit and update the seed (Track fit)
- Repeat until last layer
- Might pick up wrong hit somewhere and go astray

Combinatorial Kalman filter

- Generate an initial track segment (“seed”)
- Extrapolate the seed to the next layer
- Pick up all compatible hits and make branches for all of them (+1 empty)
- Extrapolate all branches and continue
- Number of branches must be limited

Track road

- Define a road using three hits or two hits plus a vertex (beam-line)
- Pick up hits in the road
- Can be done in a projection or in 3d
- Runs into trouble if too many competing hits are picked up in the road

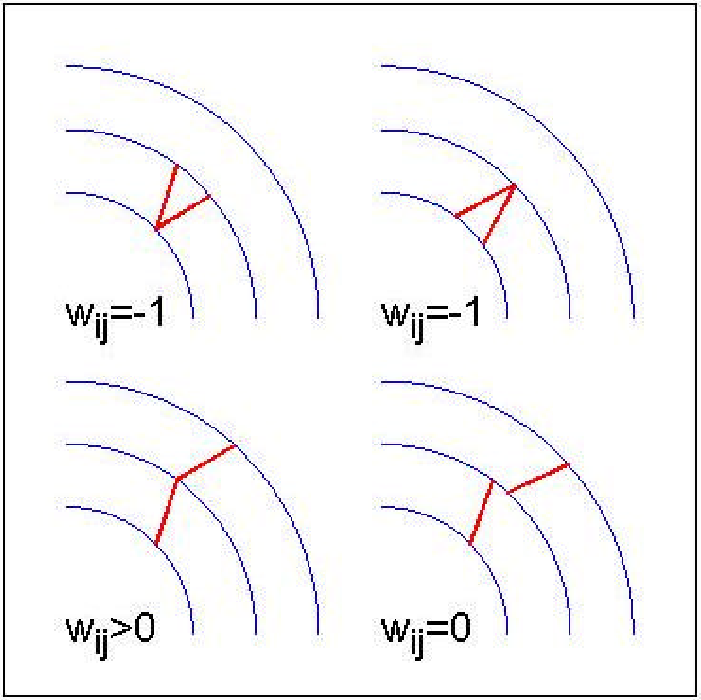
Hough transform

- Transform hits from “image space” to “parameter space”
- Hits on the same track cluster in parameter space
- Find clusters in parameter space
- Feasible only for very simple track models (line, circle)

Hopfield network

- Build recurrent neural network
- Neurons are track segments connecting hits in adjacent layers
- Weights reflect probability that two adjacent segments belong to the same track
- Minimize energy function of the network

Hopfield network (cont)



Hopfield network (cont)

- Track candidates are formed by neurons that are “on” in the final state
- No track model used
- Runs into trouble with high background and complicated detector geometry
- Successfully used in ALEPH TPC

Track fit

- Determine track parameters
- Determine covariance matrix
- Test track hypothesis
- Reject outliers
  - Distorted hits (cluster fusion, δ - electron)
  - Extraneous hits
  - Electronic noise

Ingredients

- Magnetic field
  - Constant or variable
- Track model
  - Solution of the equation of motion
  - Analytic (explicit) or numerical
- Error model
  - Observations errors
  - Process noise

Magnetic field

- Fast computation required
- Interpolation in a large table
- Global approximation by harmonic functions
- Local approximation by low-order polynomials
- Constant field plus correction terms

Track model

- Propagate track parameters from A to B:

$$p_B = f_{A\rightarrow B}(p_A)$$

- Provide Jacobian:

$$F_{A\rightarrow B} = \partial f_{A\rightarrow B} / \partial p$$

- Careful choice of track parameters important for linear approximation
- $f_{A\rightarrow B}$ is analytic only in very simple cases

Track model (cont)

- No field: straight line
- Constant field: helix
- Even in these cases, track model is analytic only for simple detector surfaces (plane, cylinder)
- In all other cases, $f_{A \rightarrow B}$ and $F_{A \rightarrow B}$ have to be computed numerically

Error model

- Observation error
  - Covariance matrix usually comes along with the hit
- Process noise
  - Mainly multiple Coulomb scattering, treated in Gaussian approximation
  - Bremsstrahlung (for electrons), treated in Gaussian or Gaussian mixture approximation
  - Energy loss by ionization, mostly mean only, no spread

Estimation of track parameters

- Most estimators minimize a least-squares objective function
- Least-squares estimation
  - Linear regression
  - Kalman filter
- Robust estimation
  - Adaptive filter

Linear regression

- Set up linear model:

$$m=Fp+\text{🎁},\ E(\text{🎁})=V,\ \mathrm{cov}(\text{🎁})=V=G^{-1}$$

- V describes observation errors and process noise
- Estimation of p:

$$p'=(F^{T}GF)^{-1}F^{T}Gm$$

- Covariance matrix of p:

$$\mathrm{cov}(p')=(F^{T}GF)^{-1}$$

Linear regression (cont)

- Chi-square statistic

  $$\chi^2 = (m-Fp')^T G (m-Fp')$$

- Outliers may be masked by multiple scattering
- Optimal estimate only at a single point

Kalman filter

- Iterative version of least-squares estimation
- Start with approximate track parameters and large errors
- Extrapolate to observation, adding up process noise
- Incorporate observation by weighted mean

Kalman filter (cont)

- Iterate until all observations are used
- Last estimate contains full information
- Propagate back the full information to all previous observations – "Smoothing"
- Optimal estimates are available anywhere along the track
- Full power for outlier search

Adaptive filter

- Robust version of the Kalman filter
- Compute Kalman filter/smoother
- Compute “assignment probability” of each observation to the track
- Re-compute Kalman filter /smoother, using assignment probabilities as weights
- Iterate until convergence

Adaptive filter (cont)

- Outliers are suppressed automatically
- Automatic choice between competing or ambiguous observations
- “Soft” assignment of hits to tracks: assignment probabilities can vary between 0 and 1

Reconstruction of neutral particles

- Neutral particles are only seen by the calorimeters
- Photons are absorbed in the electromagnetic calorimeter
- Neutral hadrons are absorbed in the hadronic calorimeter
- Neutrinos are not detected directly

Shower finding

- An incident particle produces a shower in the calorimeter
- A shower is a cluster of cells with energy deposit above threshold
- Various clustering techniques are used to find showers
- Overlapping clusters must be separated

Shower finding (cont)

- The algorithms depend on various characteristics of the calorimeter
  - Type (electromagnetic or hadronic)
  - Technology (homogeneous or sampling)
  - Cell geometry
  - Granularity

Particle identification

- Determining the type of a particle
- Dedicated detectors
  - Threshold Cherenkov
  - Ring imaging Cherenkov (RICH)
  - Transition radiation detector
  - Ionization measurements

Particle identification

- Combining information from several detectors
  - Shower in elmag calorimeter + no matching track in tracker 🎭 photon
  - Shower in elmag calorimeter + matching track in tracker 🎭 electron
  - Track in muon system + matching track in tracker 🎭 muon

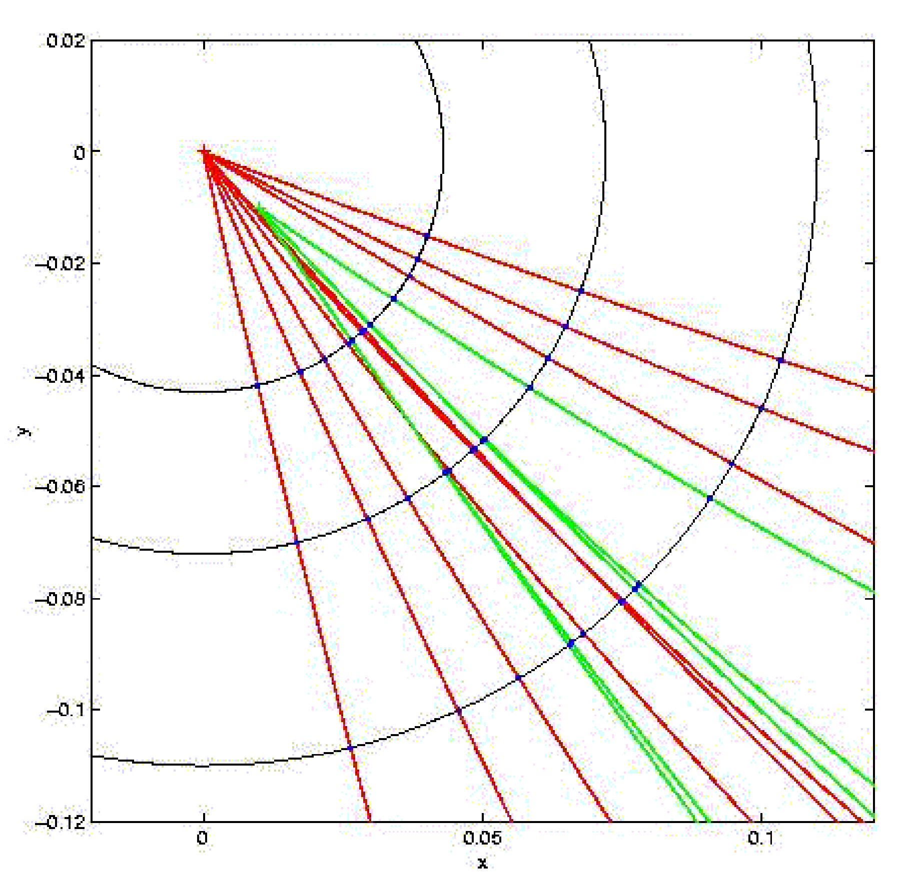
Vertex reconstruction

- Primary vertex: interaction of the two beam particles – easy
- Secondary vertices: decay vertices of unstable particles – difficult
- Emphasis on short-lived unstable particles which decay before reaching the tracker
- Data compression

The difficulties

- Association of tracks to vertices is unknown
- Secondary tracks may pass very close to the primary vertex
  - Especially if decay length is small
- Track reconstruction may be less than perfect

Primary and secondary tracks



Decomposition of the problem

- Pattern Recognition or Vertex Finding
  - Assign tracks to vertex candidates
- Parameter estimation or Vertex Fit
  - Determine vertex location + covariance matrix, update track parameters
- Test of the vertex hypothesis
  - Check chi-square, residuals, remove outliers

Vertex finding

- Almost independent of the detector geometry
- Secondary vertex finding may depend on the physic channel under investigation
- Essentially a clustering problem
- Many solutions available

A few vertex finding algorithms

- Hierarchical clustering
  - Single linkage, complete linkage,…
- Non-hierarchical clustering
  - k-means, robust location (mode) estimation, iterated vertex fit
- Neural network/physics inspired
  - Competitive learning, deterministic annealing, superparamagnetic clustering, quantum clustering,…

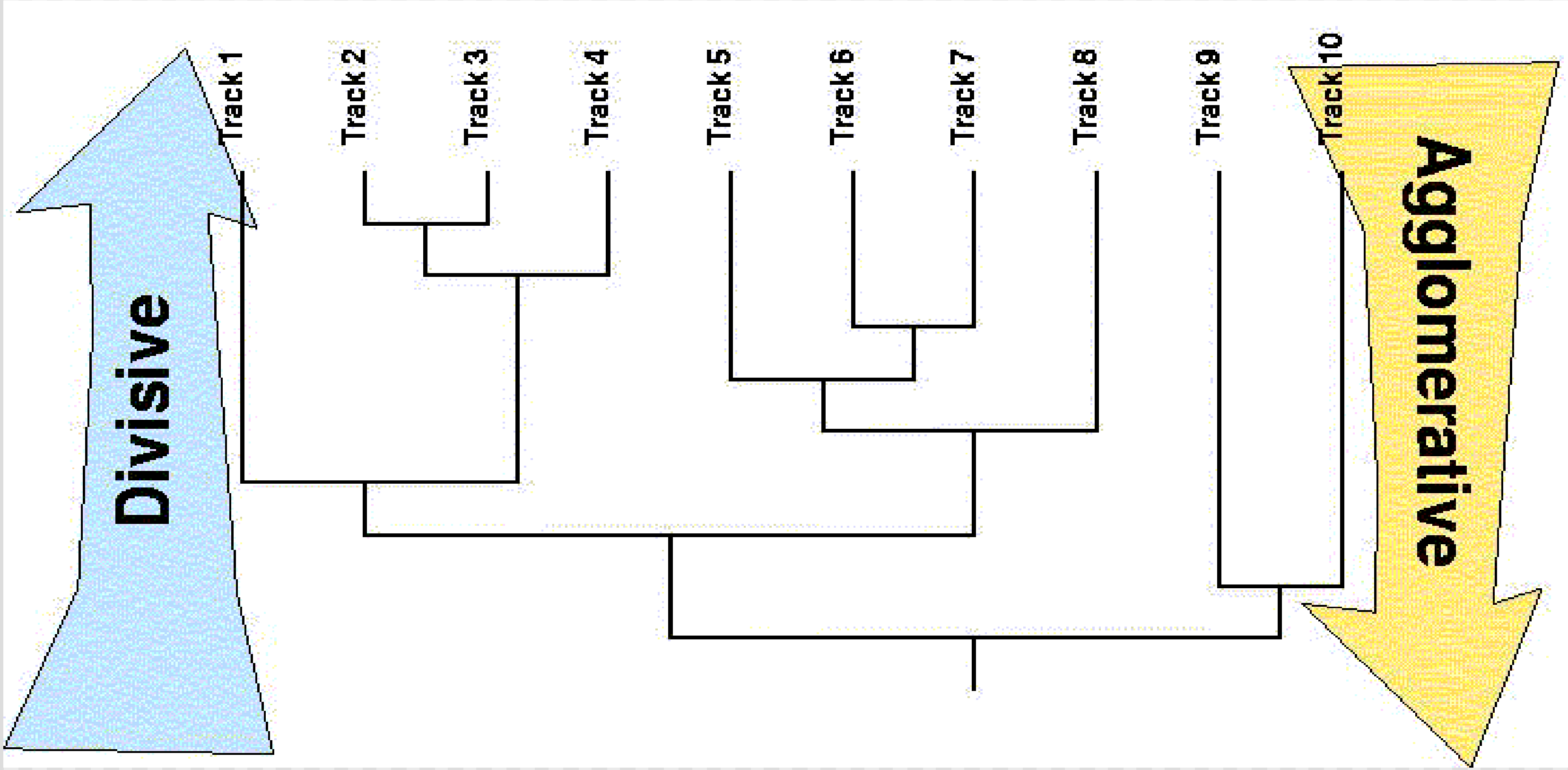
Hierarchical clustering

- Agglomerative
- Works with distances between objects (tracks) and clusters
- Single linkage
  - Distance between clusters is minimum of pair-wise object distances
- Complete linkage
  - Distance between clusters is maximum of pair-wise object distances

Hierarchical clustering (cont)

- Start with singleton clusters
- Merge clusters with the smallest distance
- Iterate until smallest distance exceeds some threshold
- Fast, but explores only a very small subset of possible clustering
- Complete linkage works quite well

Hierarchical clustering (cont)



Non-hierarchical clustering

- K-means
  - Start out with a number of prototype vertices
  - For each prototype, find all tracks for which this is the closest prototype
  - Update the prototype by doing a vertex fit using only those tracks
  - Requires careful initialization
  - Number of prototypes fixed

Non-hierarchical clustering (cont)

➢ Robust location estimation

- Represent each track by a space point (“apex point”)
- Find accumulation point of apex points using location or mode estimators with high break-down point
- Remove apex points in the vicinity of the estimated accumulation point
- Iterate
- High break-down point estimators have low precision

Non-hierarchical clustering (cont)

Iterated vertex fit

- Vertex fit with all tracks
- Find and remove outlier(s)
- Redo vertex fit until no outliers are found
- Iterate procedure on all remaining tracks
- Very powerful
- Can be speeded up by using a robust vertex fit

Neural network inspired clustering

- Competitive learning
  - Start out with a number of prototype vertices
  - Prototypes are attracted to the tracks according to some learning rule
  - Iterate learning steps until convergence
  - Can be made more “just” by letting all prototypes learn at about the same rate

Neural network inspired clustering

➢ Deterministic annealing

- Start out with a single prototype at a certain "temperature"
- Prototype is attracted to the tracks according to some learning rule
- Compute largest eigenvalue of spread around the prototype
- Split prototype if eigenvalue is too large for the current temperature
- Iterate for each new prototype

Estimation of vertex parameters

- Most estimators minimize a least-squares objective function
- Least-squares estimation
  - Linear regression
  - Kalman filter
- Robust estimation
  - Adaptive filter

Linear regression

- Set up linear model:
  $p_i = c_i + A_i v + B q_i + \varepsilon_i$, $E(\varepsilon_i) = V_i$, $\mathrm{cov}(\varepsilon_i) = V_i = G_i^{-1}$
- V_i describes track errors of track i
- Estimation of vertex v:
  $$v' = C \left[\sum A_i^T G_i^B (p_i - c_i) \right]$$
- Covariance matrix of p:
  $$\mathrm{cov}(v') = C = \left(\sum A_i^T G_i^B A_i \right)^{-1}$$

Linear regression (cont)

- Track residuals:

$$r_i = p_i - c_i - A_i v'$$

- Chi-square statistic

$$\chi^2 = \sum r_i^T G_i r_i$$

- Multiple outliers may be masked

Kalman filter

- Iterative version of least-squares estimation
- Start with approximate vertex position and large errors
- Add one track after the other
- For each track, check compatibility with current vertex estimate
- Remove outliers immediately

Kalman filter (cont)

- Iterate until all tracks are used
- Last estimate contains full information
- Update all track parameters – “Smoothing”
    - Improvement of track parameters by vertex constraint

Adaptive vertex fit

- Robust version of the linear vertex fit (regression/Kalman filter)
- Make linear vertex fit
- Compute “assignment probability” of each track to the vertex
- Re-compute linear vertex fit, using assignment probabilities as weights
- Iterate until convergence

Adaptive vertex fit (cont)

- Outliers are suppressed automatically
- “Soft” assignment of tracks to vertices: assignment probabilities can vary between 0 and 1
- Assignment can be made “hard” by cooling down to low temperature
- Adaptive filter can be reapplied to rejected tracks – vertex finding

Kinematical fit

- Impose constraints on a reconstructed vertex
  - Momentum conservation
  - Energy conservation (if masses are known)
  - Invariant mass of mother particle
- Put constraints into a Lagrange multiplier
- Construct least-squares objective function

Kinematical fit

- Taylor-expand objective function with respect to all momentum vectors
- Minimize objective function
- Neutral particles have to be included (calorimeter information)

Persistency

- Event reconstruction produces physics objects
  - Tracks
  - Vertices
  - Identified particles
  - Jets
  - Tags
- Need to be made persistent

Persistency (cont)

- Physics objects depend on
  - Alignment
  - Calibration
  - Version of the reconstruction program
  - Algorithm parameters
- Must be made persistent as well
- About 200 kbyte per event (CMS)
- Tools: Objectivity, ROOT, POOL

POOL

- Common persistency framework for physics applications at LHC
- Part of the LCG (LHC Computing Grid)
- Data are stored in a distributed and grid-enabled fashion
- You will hear a LOT about POOL later!

Simulation

- Why do we need simulation?
- Optimization of detector in design phase
- Testing, validation and optimization of reconstruction algorithms
- Computation of reconstruction efficiency
- Computation of acceptance corrections
- Background studies

Simulation steps

- Event generation
- Tracking through the detector, using detector geometry and magnetic field
- Interaction of particles with matter
- Signal generation in sensitive volumes
- Digitization (simulate ADC or TDC)
- Digitized data and truth information are made persistent

Event generation packages

- PYTHIA/JETSET
  - Also known as “Lund Monte Carlo”
  - General purpose event generator
  - Collisions of electrons, positrons, protons and antiprotons in various combinations.
  - “Together they contain theory and models for a number of physics aspects, including hard and soft interactions, parton distributions, initial and final state parton showers, multiple interactions, fragmentation and decay.”

Event generation packages (cont)

- HERWIG
  - Hadron Emission Reactions With Interfering Gluons
  - General purpose event generator
  - Hard lepton-lepton, lepton-hadron and hadron-hadron scattering and soft hadron-hadron collisions
  - Current version in FORTRAN, HERWIG++ planned

Event generation packages (cont)

- PANDORA
  - Physics event generator for linear collider studies
  - Collisions of electrons, positrons and photons
  - Current version in FORTRAN, HERWIG++ planned
  - Written in C++
  - Interface to PYTHIA

Event generation packages (cont)

- LOTS of specialized generators for
  - Electroweak physics
  - QCD
  - Higgs
  - Supersymmetry
  - Exotic physics

Detector simulation

- Was frequently (and still sometimes is) experiment-specific
- Nowadays there is a widely used standard: GEANT
  - GEANT3: FORTRAN
  - GEANT4: C++
- You will hear a LOT about GEANT4 later!

GEANT4 functionality

- Description of geometry and materials
- Particle tracking and interactions with matter
- Generation of the detector response
- Bookkeeping, metadata management
- Visualization of geometry, tracks and hits

User responsibility

- Link to the event generator
- Description of the detector
- Setting of physics processes and cuts
- Code for digitization of the detector response and generation of noise
- Tuning – simulated data should resemble real ones as closely as possible

Detector description

- Geometry
  - Shape
  - Placement relative to mother volume
  - Symmetries
- Material
  - Composition
  - Density
  - Radiation length, interaction length, …

DD examples

- CMS
  - XML Schema detector description database
  - Derive detector descriptions for simulation (GEANT4), reconstruction and visualization
- ATLAS
  - Primary Numbers stored in relational database
  - GeoModel C++ library
  - Derive detector descriptions for simulation (GEANT4), reconstruction and visualization

DD examples

- Alice
  - ROOT classes
  - Used for simulation and reconstruction
  - Invoke physics processes from GEANT and FLUKA
- LHCb
  - XML DTD (Document Type Definition)
  - Interpreted by GAUDI plug-ins to build detector representations for simulation and reconstructions

Physics analysis

- Event selection
  - Multidimensional criteria
- Signal extraction
  - Study background
  - Determine significance of signal
- Corrections
  - Detector acceptance, reconstruction efficiency, …
  - From simulated data

Physics analysis (cont)

- Computation of physical quantities
  - Cross sections, masses, decay widths, …
- … and of their errors
  - Statistical errors: uncertainty because of limited number of observations
  - Systematic errors: uncertainty because of limited knowledge of key assumptions (beam energy, calibration, alignment, magnetic field, theoretical values, background channels, …)

Analysis tools

- Need versatile tools for
  - Multidimensional selection
  - Event display and interactive reprocessing
  - Histogramming
  - Plotting
  - Fitting of curves and models
  - Point estimation and confidence intervals
  - …

Analysis tools (cont)

- ROOT
  - Builds on HBOOK and PAW (CERN)
- JAS
  - Java Analysis Studio (SLAC)
- OpenScientist
  - “Open, modular, free, portable, efficient and collaborative” (LAL Orsay)
- AIDA
  - common interfaces for data analysis and visualisation

Distributed analysis

- Physics analysis will take place in many labs all over the world
- Physicists need access to event data and corresponding calibration, alignment and bookkeeping data … and to simulated data
- We need the grid!
- You will hear a LOT about the grid next week!

Reconstruction on demand

- Objects are reconstructed if and when required by the user
- User requests fitted vertices
  - triggers vertex reconstruction
  - triggers track reconstruction
  - triggers hit reconstruction
  - hit reconstruction is done
  - track reconstruction is done
  - vertex reconstruction is done

Metadata challenge

- Metadata are data describing other data
- Distributed analysis needs lots of metadata to track the location and validity of alignment constants, calibration constants, reconstructed objects, …
- Frequent updates
- Frequent access

Summary

- Physics computing involves:
  - Event filtering with multilevel trigger
  - Persistency of raw data
  - Calibration and alignment
  - Persistency of calibration, alignment and environmental data
  - Event reconstruction
  - Persistency of reconstruction objects and metadata

Summary (cont)

- Physics computing involves:
  - Event simulation
  - Persistency of simulated raw data and truth information
  - Reconstruction of simulated events
  - Persistency of reconstruction object and truth information
  - Distributed physics analysis
  - Persistency of high-level physics objects

Outlook on the track

- Data bases and persistency
  - 3 L, 3 E
- Experiment simulation
  - 4 L, 3 E
- Physics in GEANT4
  - 2 L (optional)