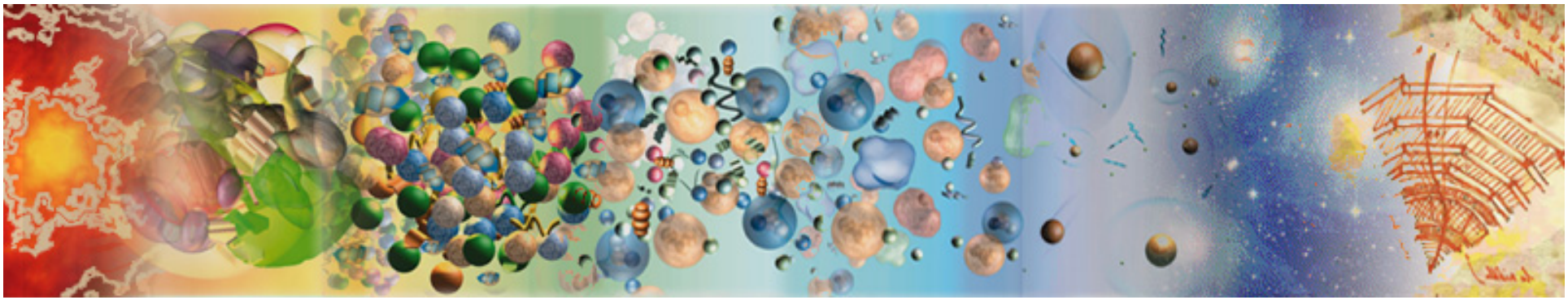
Study of track reconstruction with silicon detectors in very high multiplicity events

Ferenc Siklér

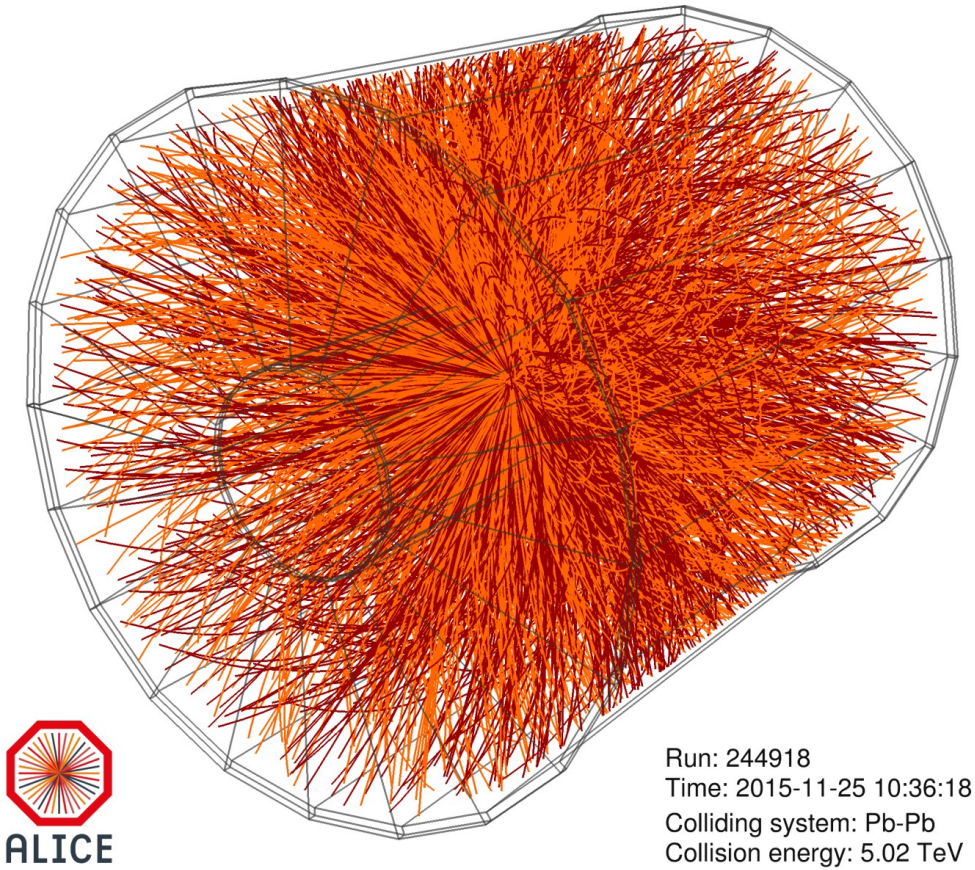
Wigner RCP, Budapest



Zimányi School '16

Budapest, 5 Dec 2016

Track reconstruction – single heavy-ion collisions

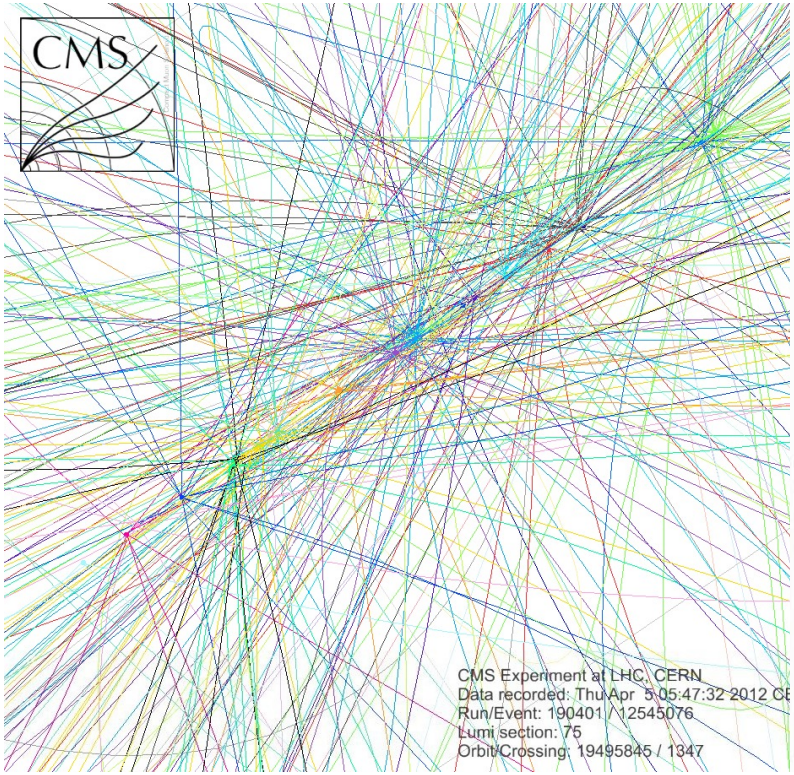


ALICE-PHO-GEN-2016-001-2

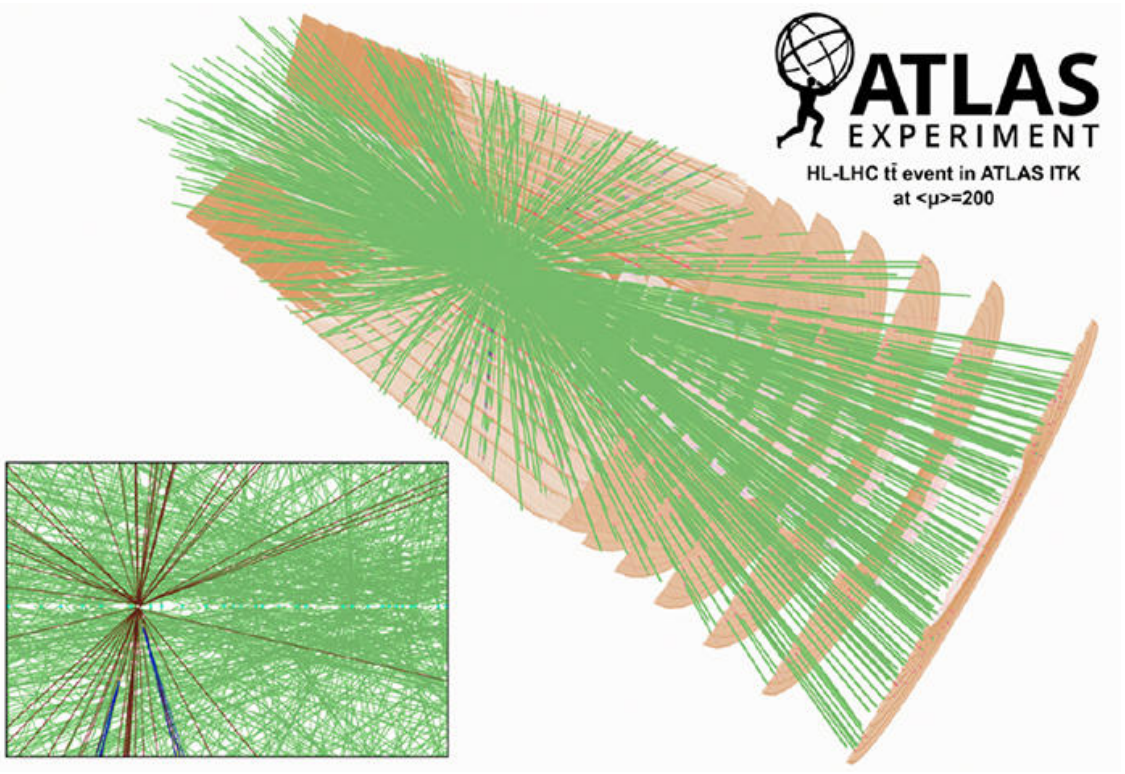
Pb-Pb collision at $\sqrt{s} = 5.02$ TeV

Very dense environment, challenging charged particle tracking

Track reconstruction – pileup of several p-p collisions



CMS Web



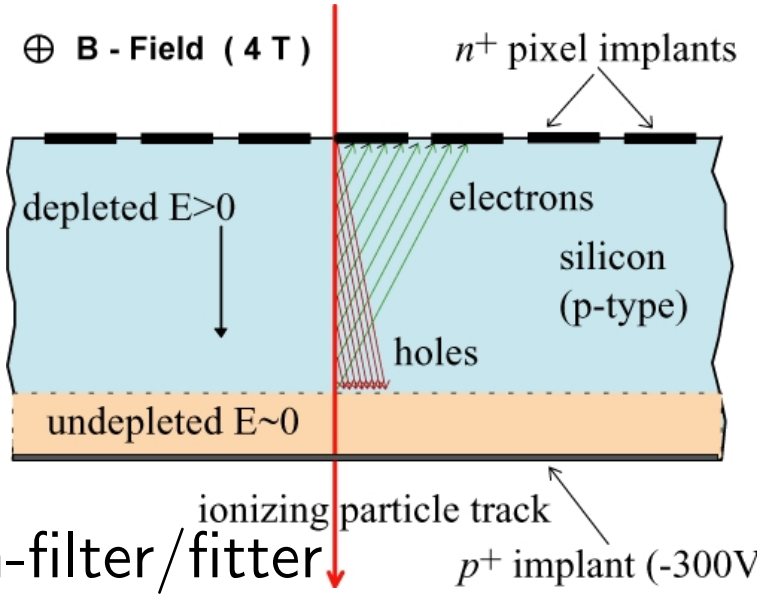
ATLAS Phase-II Tracker (Cern Courier)

Crossing of proton bunches: 50 or more simultaneous p-p collisions

Very dense environment, challenging charged particle tracking

Track reconstruction

- Silicon detectors
  - several, very thin layers $\mathcal{O}(100\ \mu\text{m})$
  - excellent spatial resolution $\mathcal{O}(10\ \mu\text{m})$
  - tolerates high collision rate (every 25 ns)



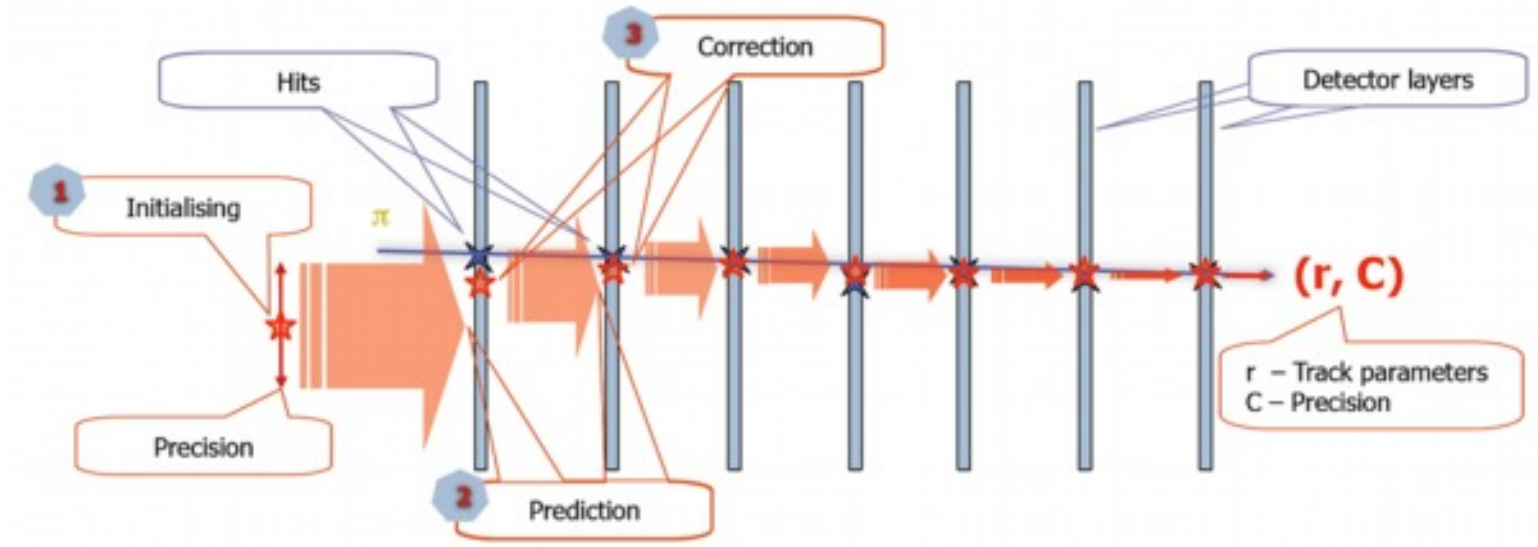
- Track reconstruction
  - combinatorial track finding using Kalman-filter/fitter
  - good efficiency and purity at high p_T
  - tolerates pileup (we are around 50)

- Room for improvement?
  - restore efficiency and purity for $p_\mathrm{T} < 1$ GeV/c
  - increase efficiency for high p_T
  - try to handle pileup expected for high-luminosity LHC (200?)

Any new ideas?

Track reconstruction

- Pattern recognition, now
  - find a track seed (usually two or three compatible hits)
  - try to extend that, pick up compatible points, build a trajectory
  - ☹ mostly uses local information
  - ☹ number of trajectory candidates must be limited at each step
  - ☹ keeping some of the best hit-candidates biases the result
  - ☹ decisions are made too early
  - ☹ trajectories are treated separately



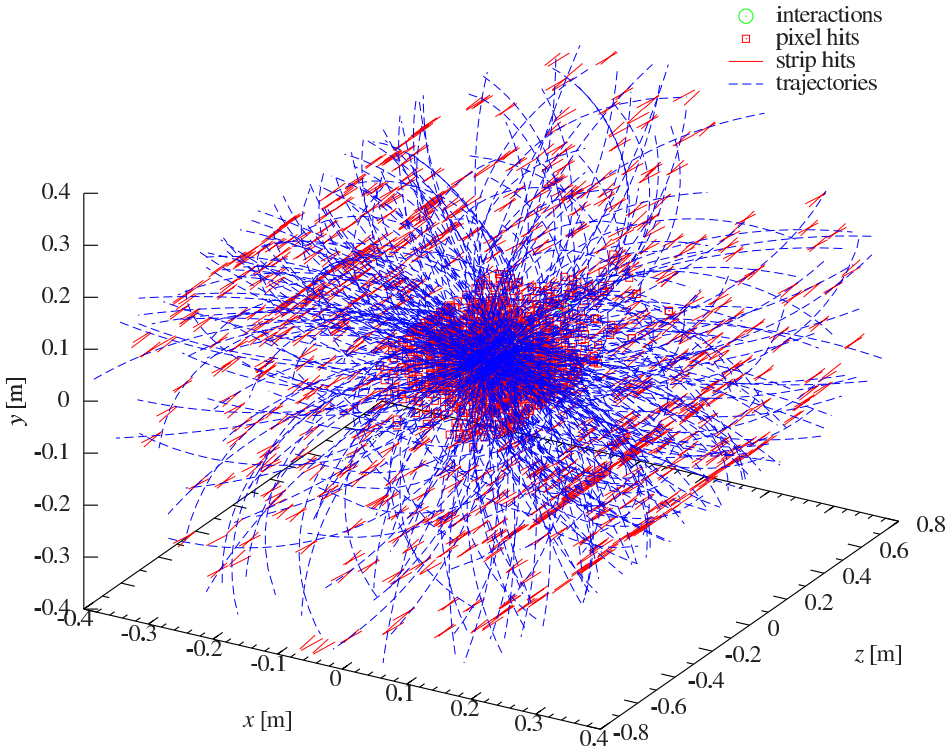
Track reconstruction – some ideas

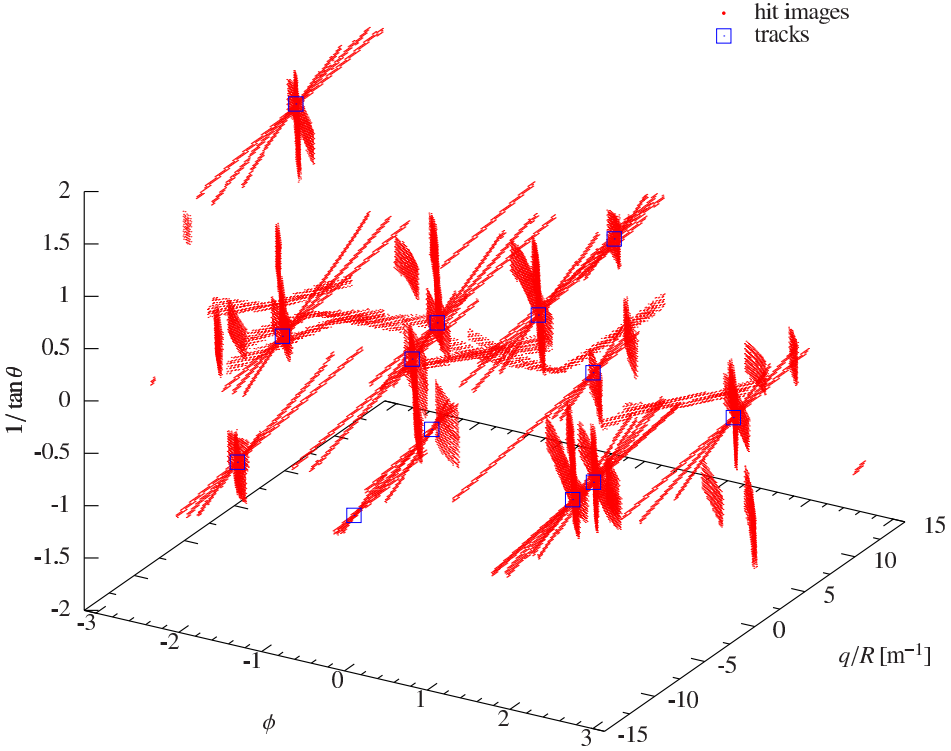
- How to identify track candidates?
  - try to use all available information
  - ☺ collect information globally and not locally

⇒ **image transformation (Hough transformation)**

  - several physics effects: multiple (Coulomb) scattering, energy loss

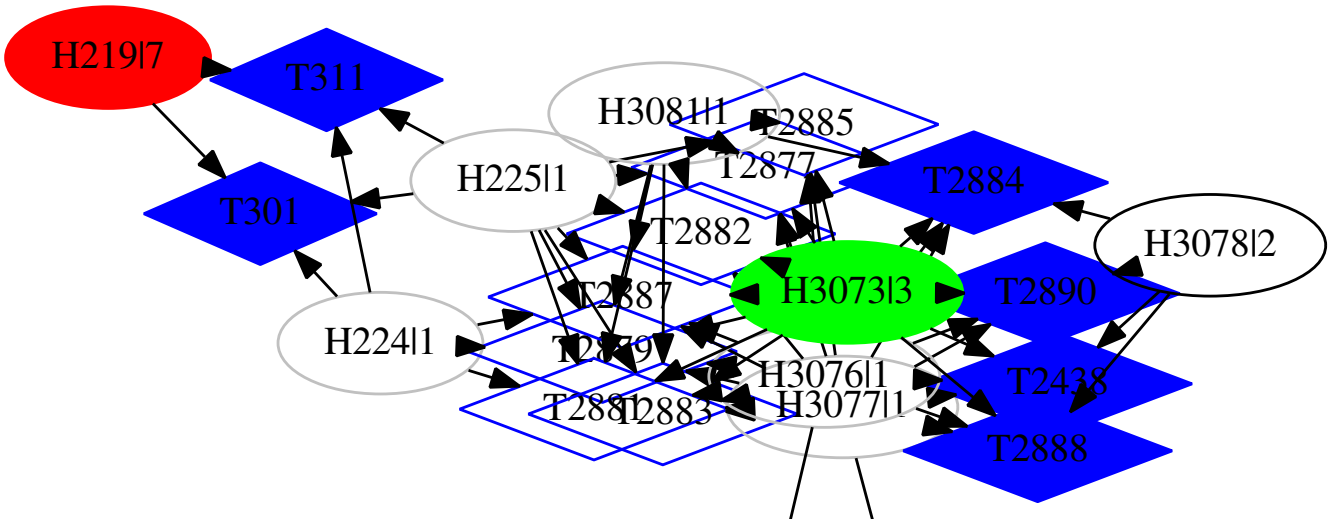
⇒ **use templates (with translation and rotational invariance)**





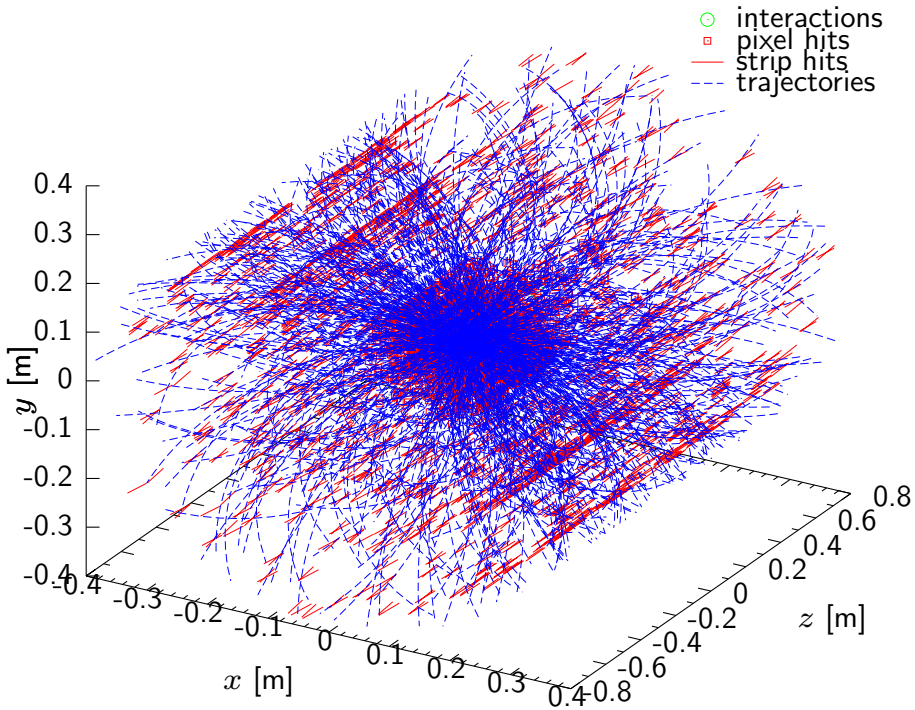
Track reconstruction – some ideas

- How to select the best set of tracks?
  - ☺ keep concurrent choices open; several hit-track assignments
  - ⇒ **treat the hits and track candidates as a (bipartite) graph**
  - – the graph can be highly connected; but has vulnerable components
  - ☺ disconnect it by looking for *bridges* and *articulation points*
  - – in the end each hit must belong to at most one track
  - ⇒ **solve subgraphs, game tree, deterministic single-player**
  - – maximize the number of hits on track, then minimize $\sum \chi^2$



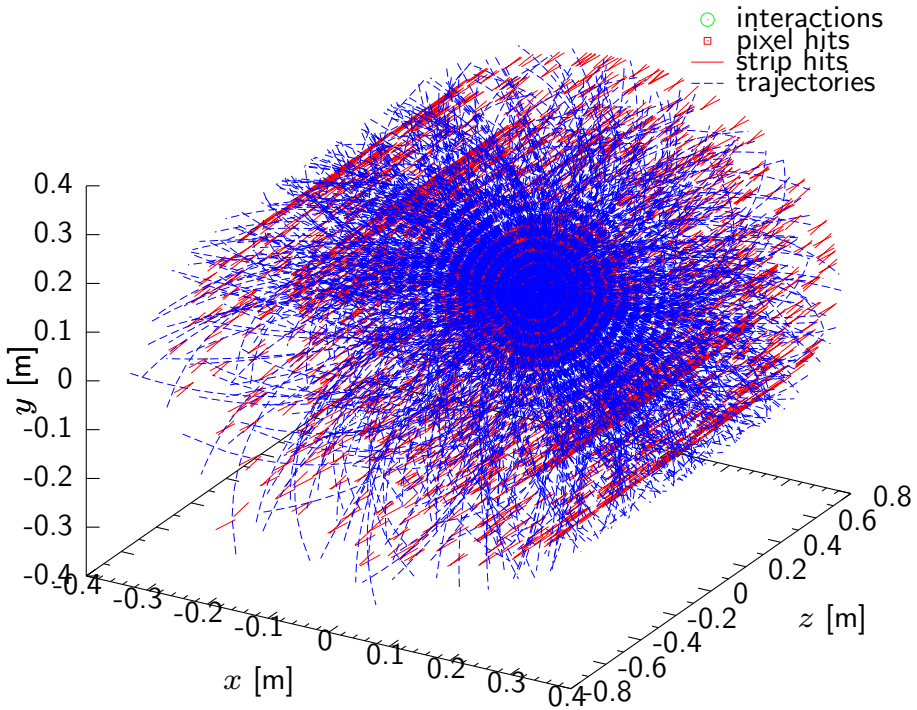
Simulated events – high multiplicity

30 simultaneous p-p



several distinct vertices

single Pb-Pb



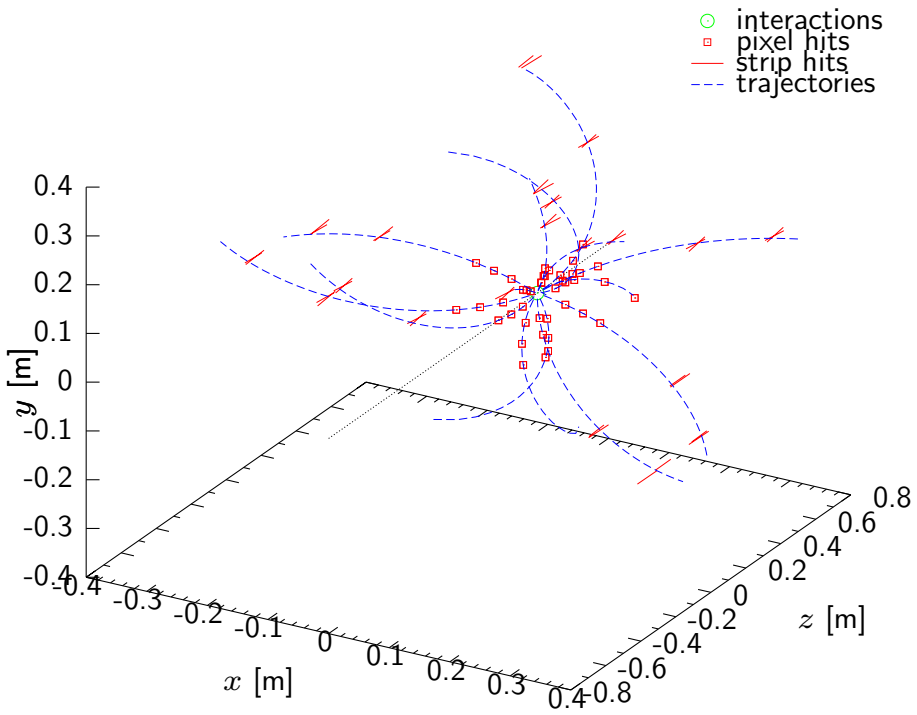
one vertex

Pretty dense

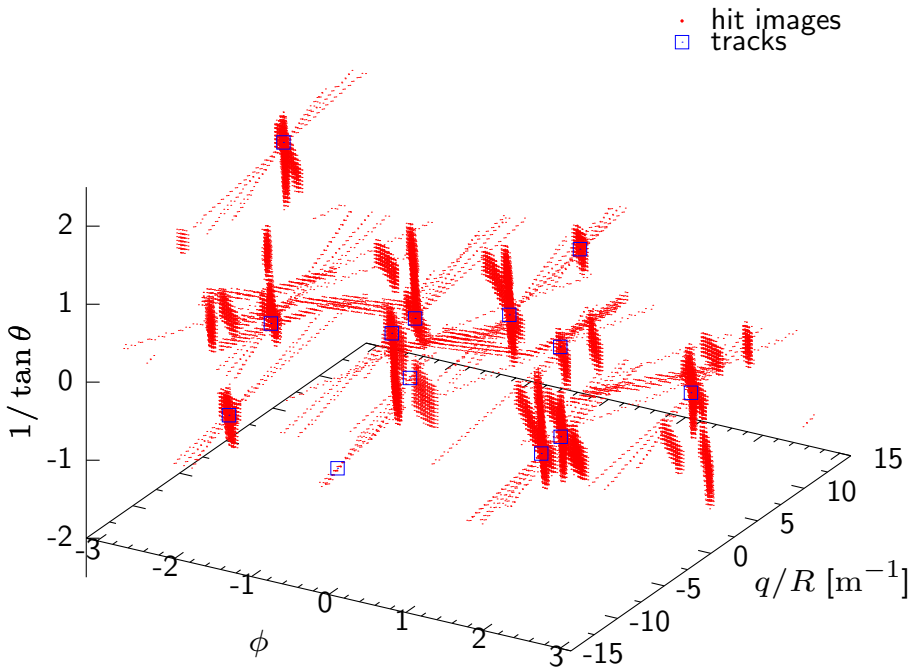
Inner pixel hits and outer single- or double-sided (stereo) strip modules

Image transformation

hit/real space

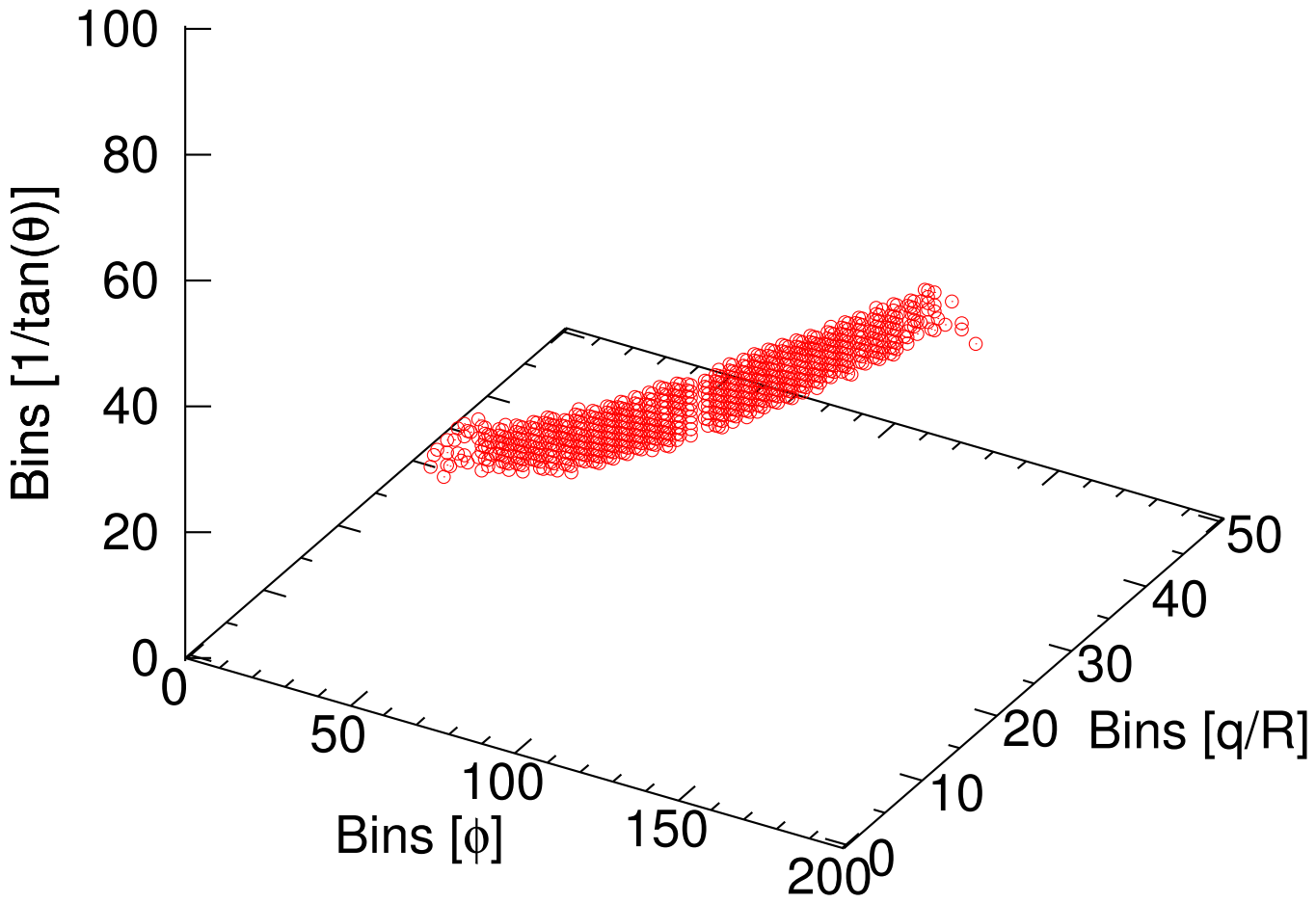


(binned) track/accumulator space



- From hit/real space to (binned) track/accumulator space
  - transform $(x, y, z) \rightarrow (\phi, q/R, 1/\tan\theta, z_0)$
  - each hit has a corresponding image (curve/band/volume)
  - hits on the same real track will have intersecting images
  - look for peaks in the track/accumulator space, ask for quality

Image transformation – templates

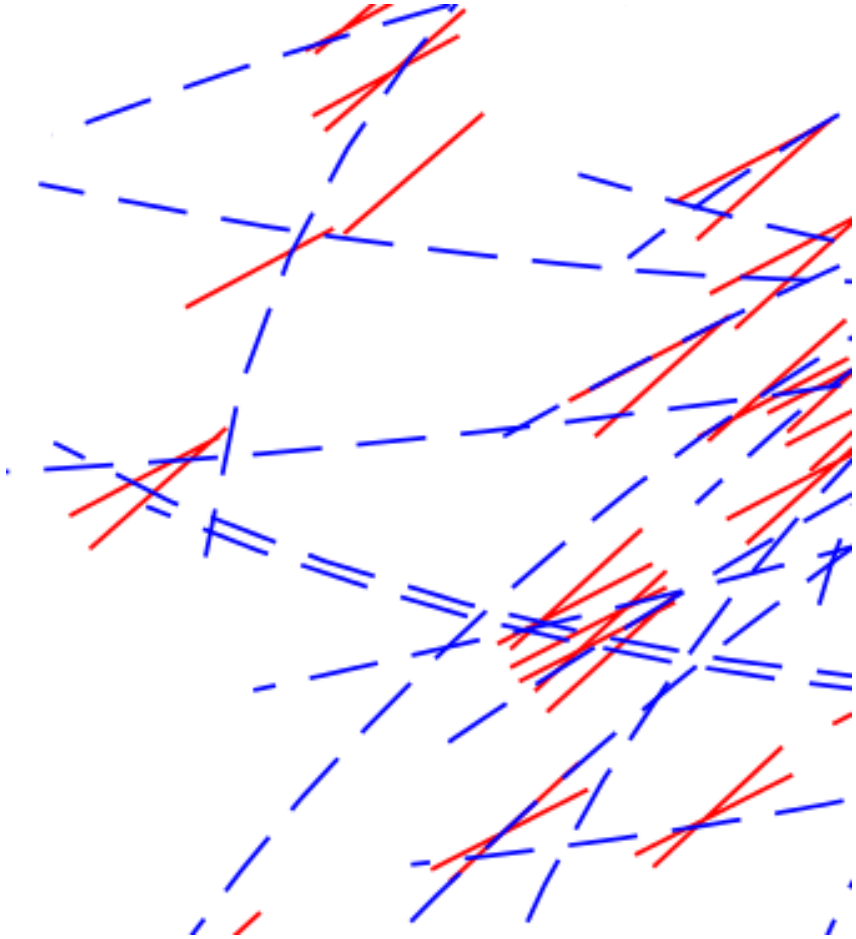


Could be nicely parametrized, but no: random processes

Generate templates with $10^6 - 10^7$ bins (p-p 4D, Pb-Pb 3D)

Check compatibility of cluster shape with direction of the trajectory

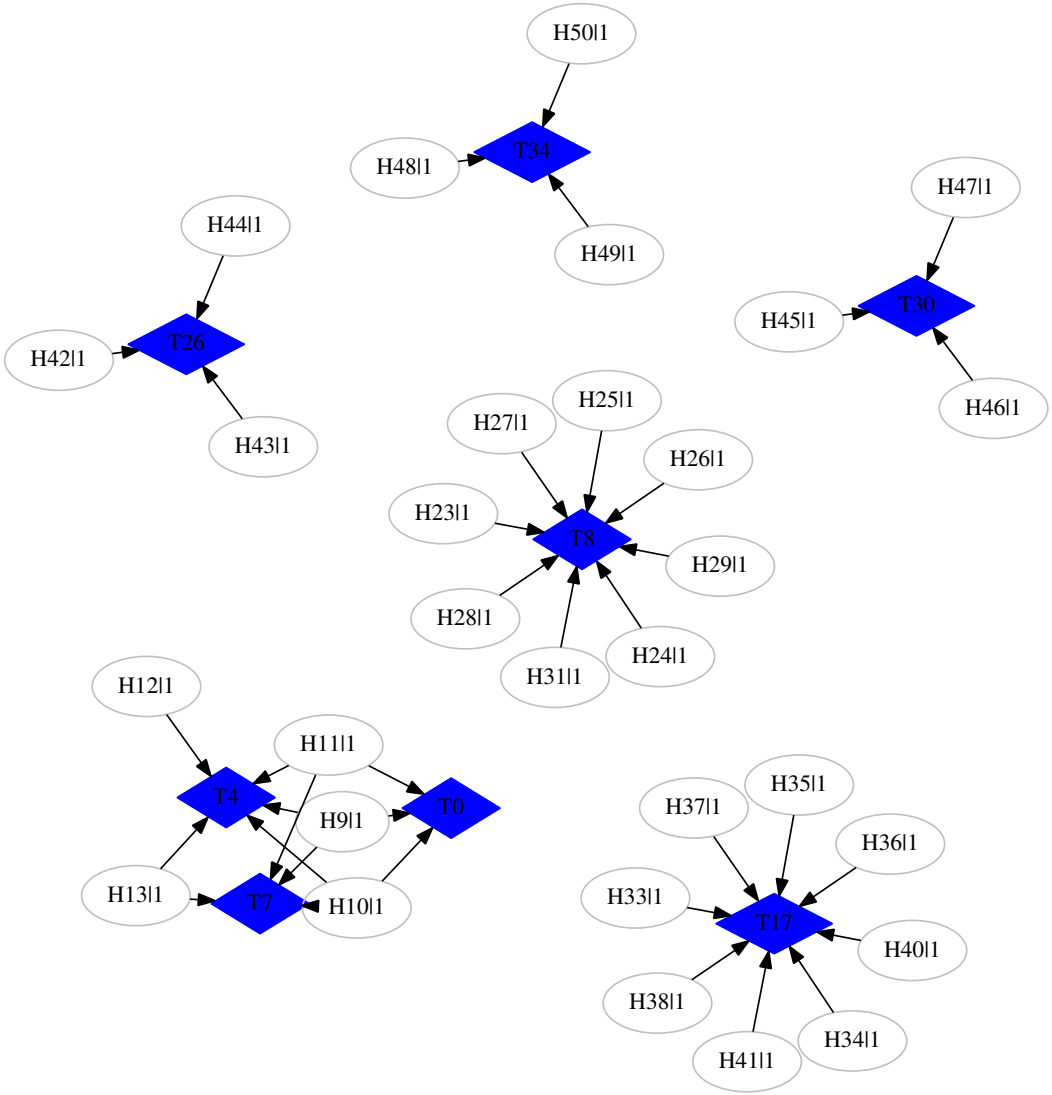
Fitting track candidates



- employ Kalman-filter/fitter; treating multiple (Coulomb) scattering, energy loss, measurement uncertainties
- check if χ^2 is reasonable for a given ndf at each stage of trajectory building
- check whether track candidate has enough hits wrt the number of potential hits
- output: list of track candidates with corresponding hits and quality measures

Keeping multiple choices

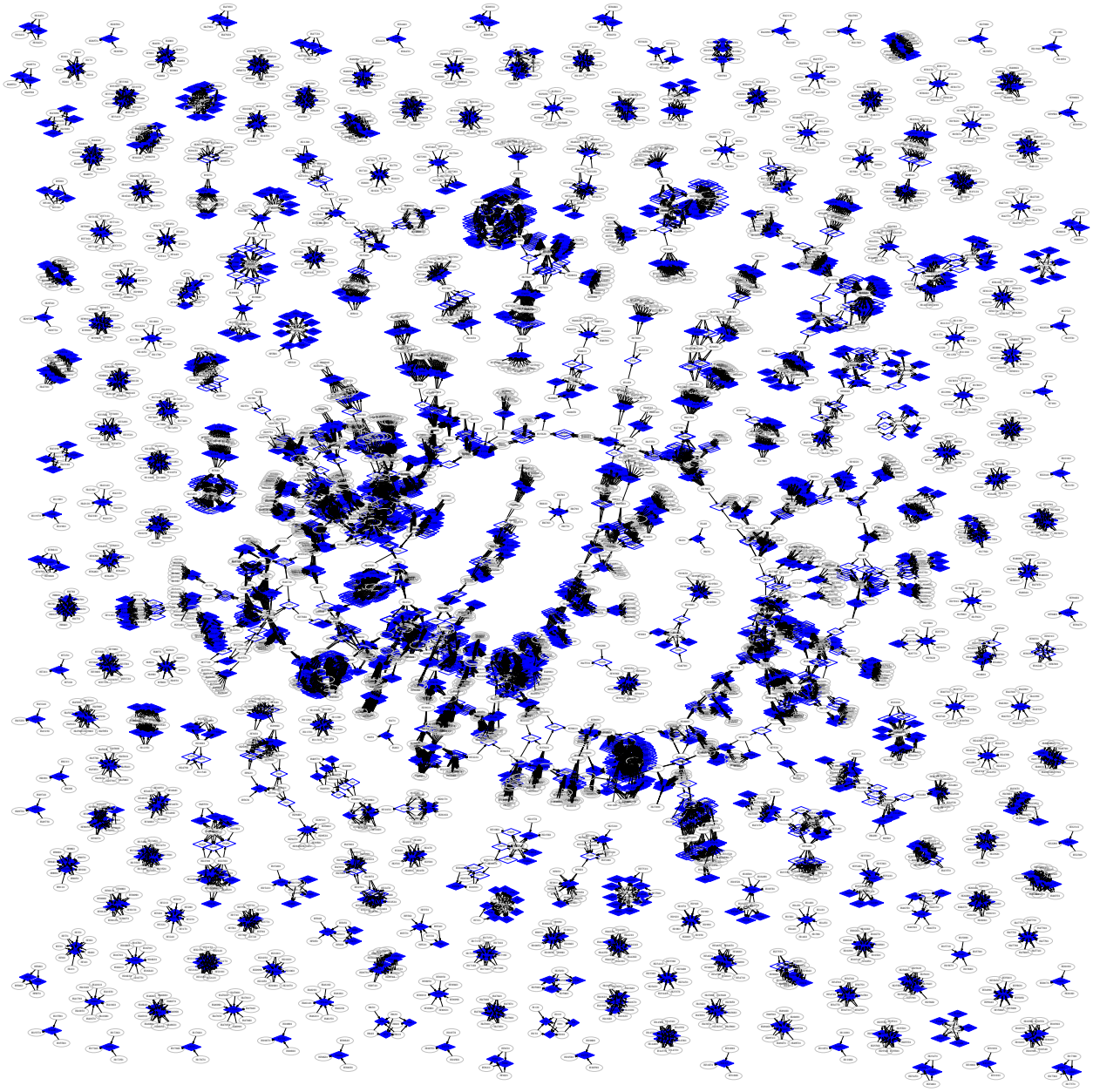
Convert to graph – single p-p



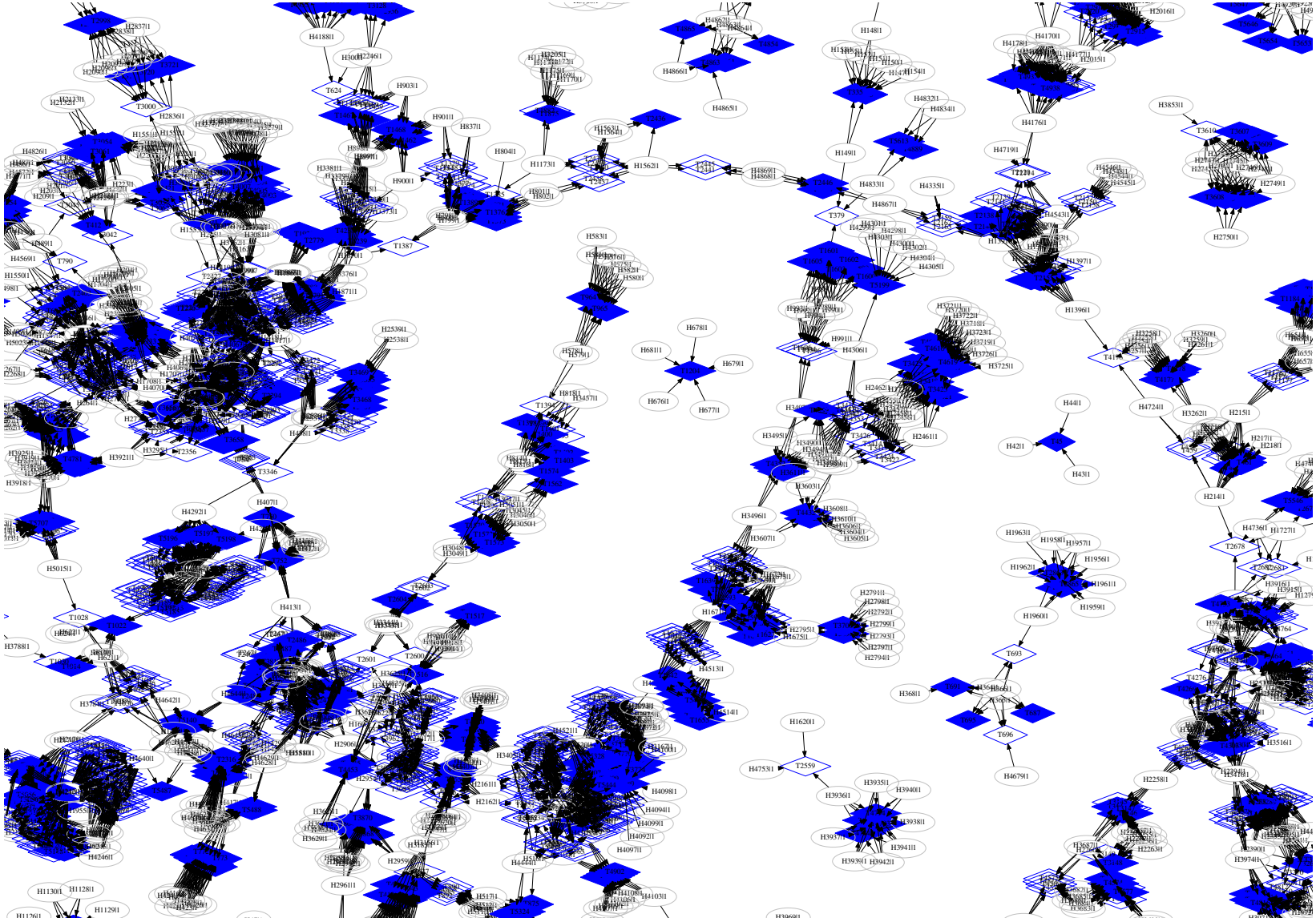
Nodes: hits (ellipses) and track candidates (blue diamonds)

Edges: hits belonging to one or more track candidates

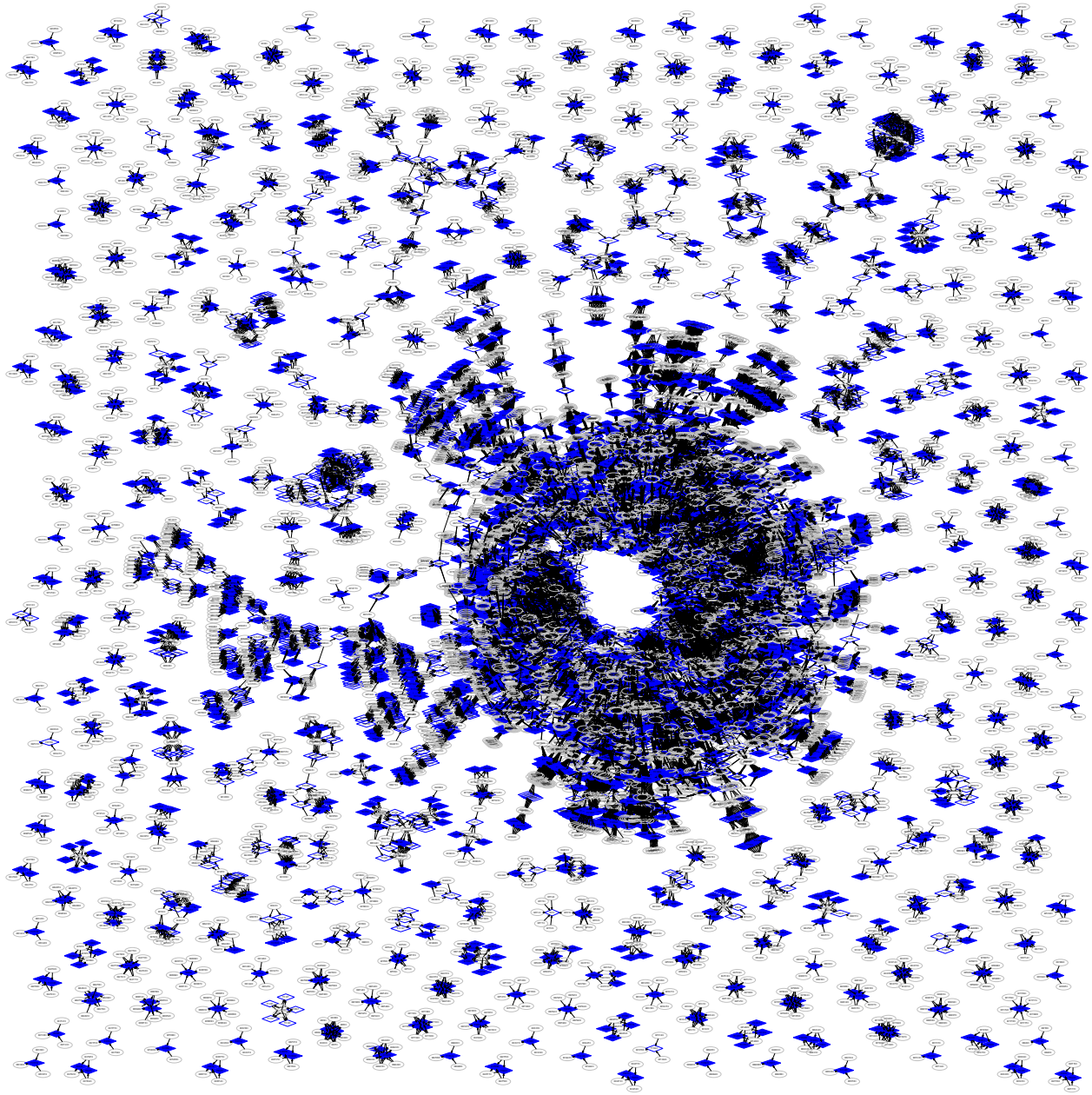
Disconnecting the graph – 30 p-p



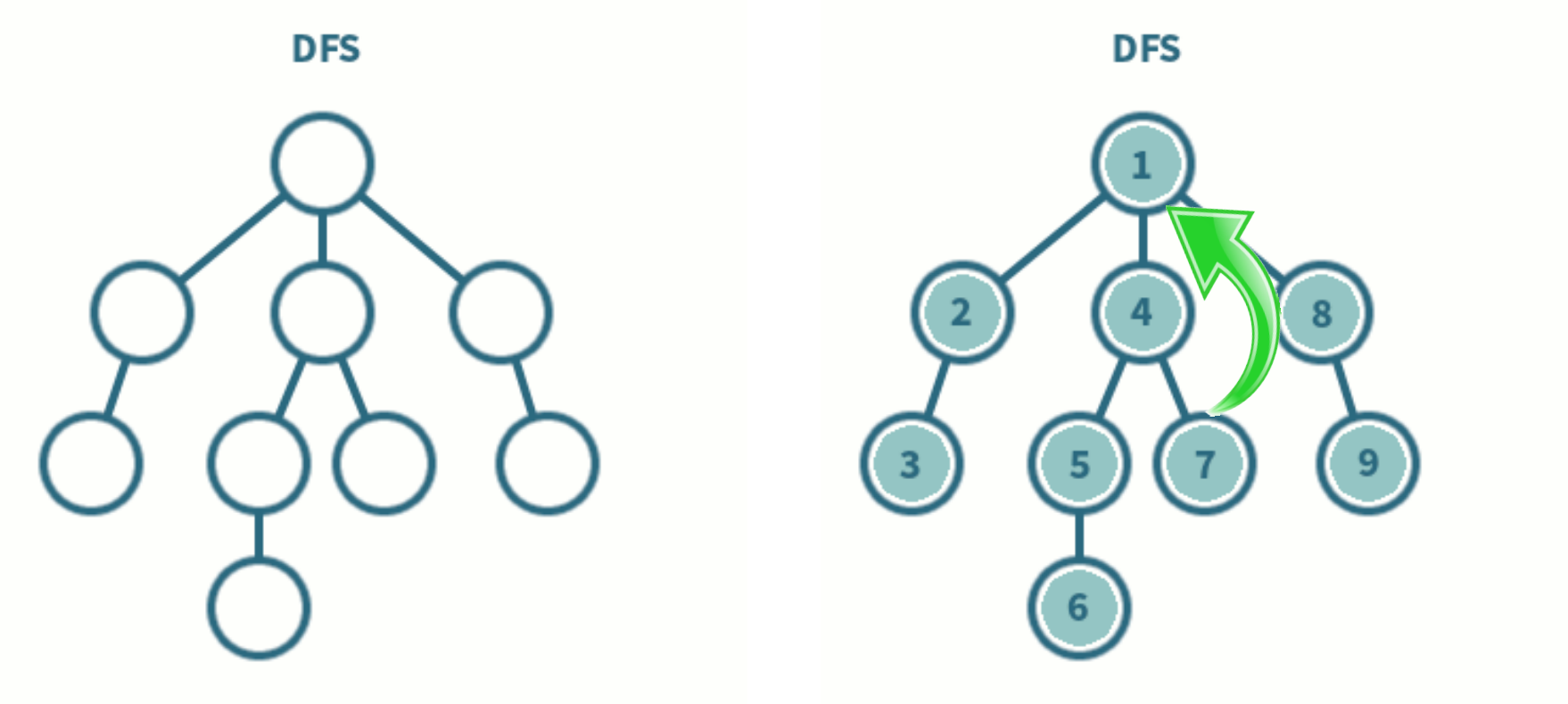
Disconnecting the graph – 30 p-p – zoom



Disconnecting the graph – single Pb-Pb

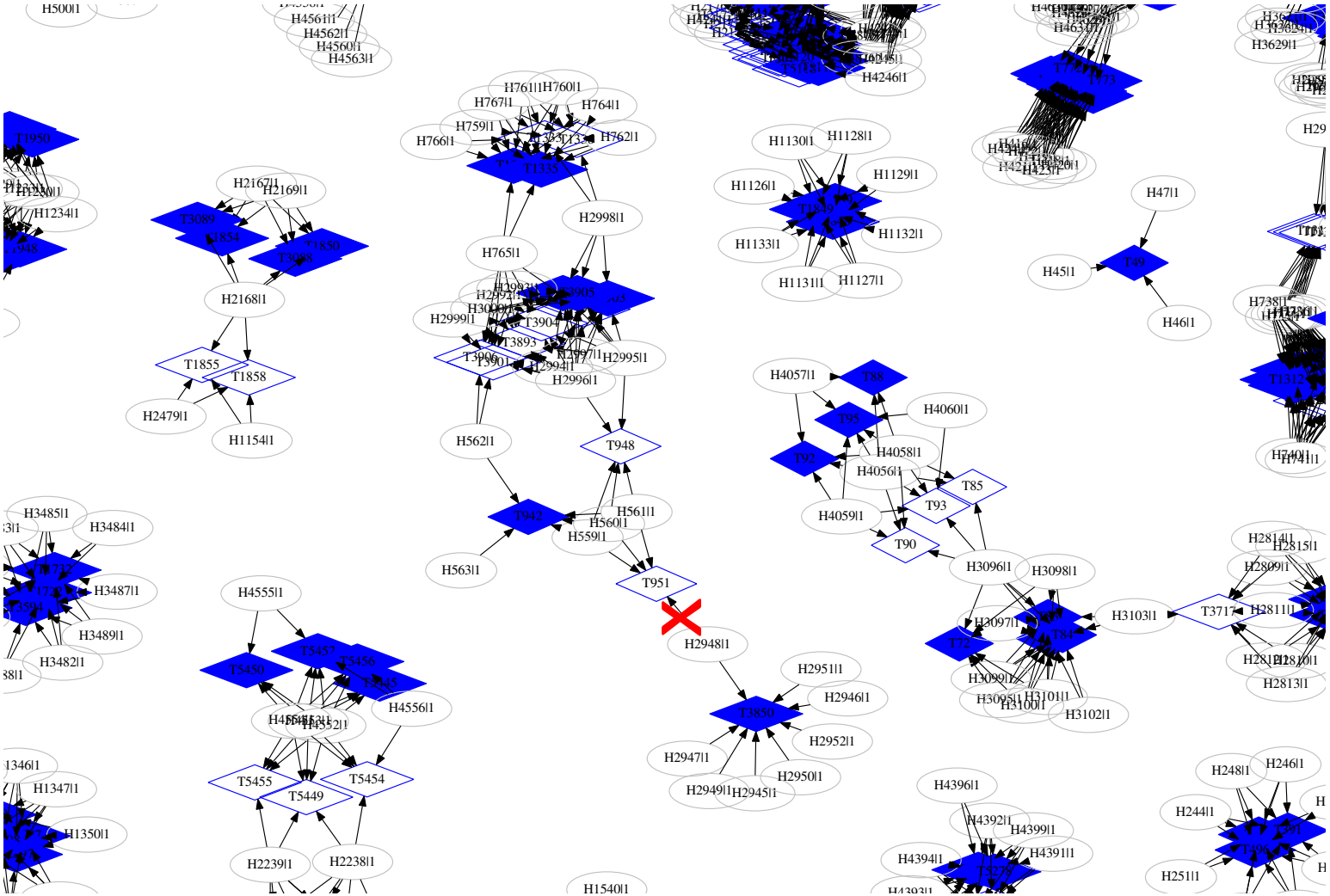


Exploring the graph



- Depth-first search ⇑
  - algorithm for traversing or searching graph structures
  - starts at the root (arbitrary node)
  - explores as far as possible along each branch before backtracking
  - back-edges may appear as well

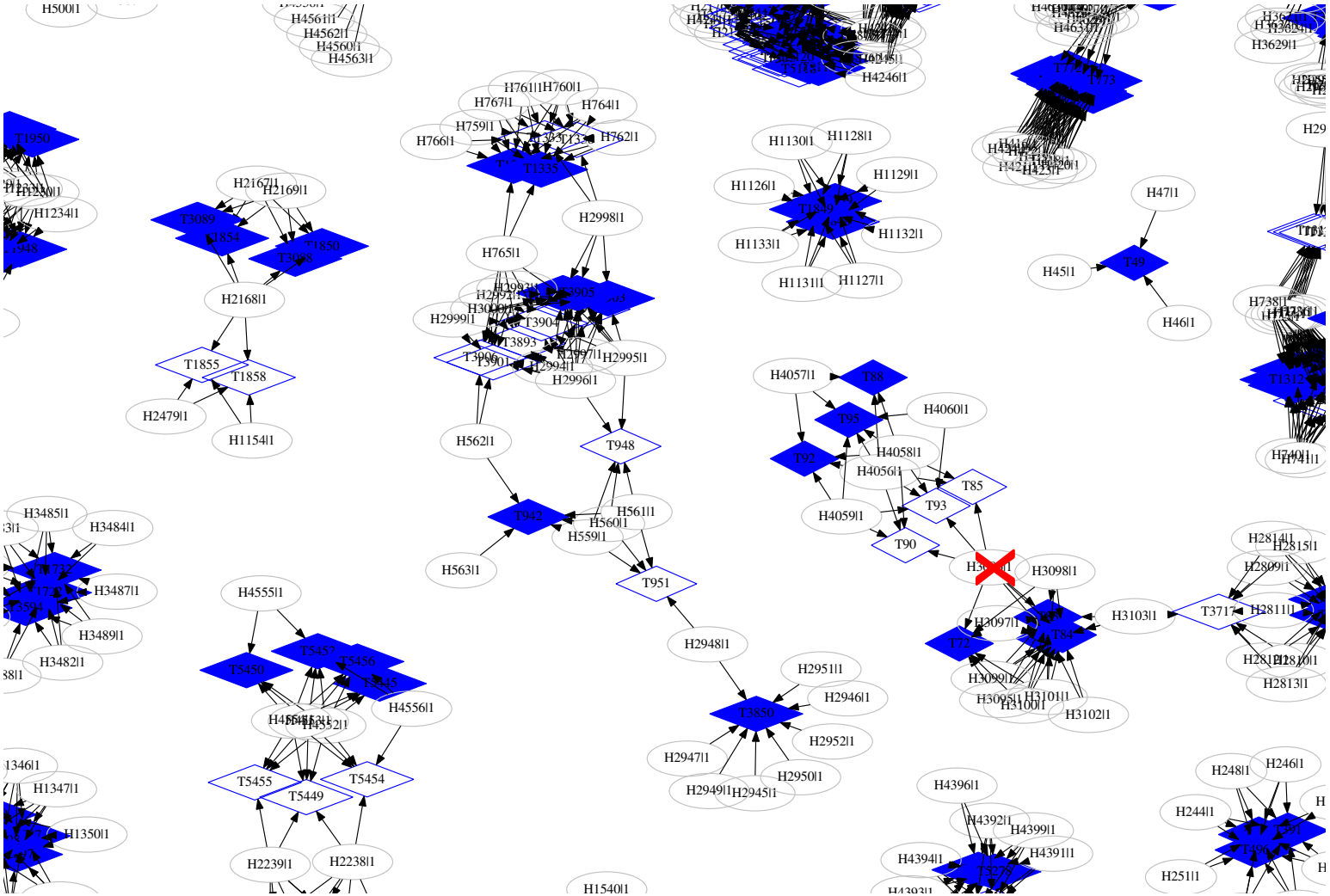
Disconnecting the graph – bridges



Its deletion increases the number of connected components

The children of its node do not have a back-edge

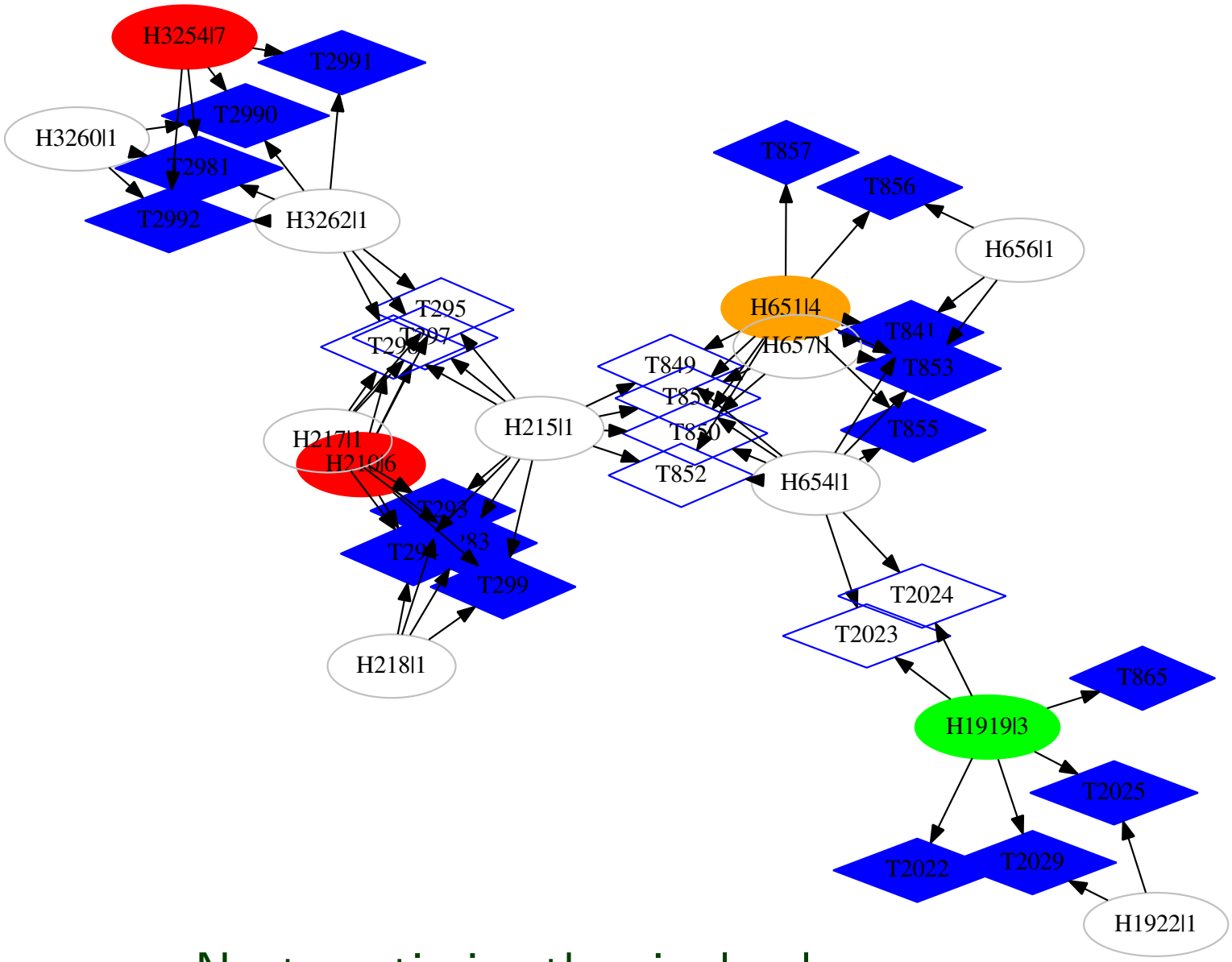
Disconnecting the graph – articulation points



Its removal increases the number of connected components

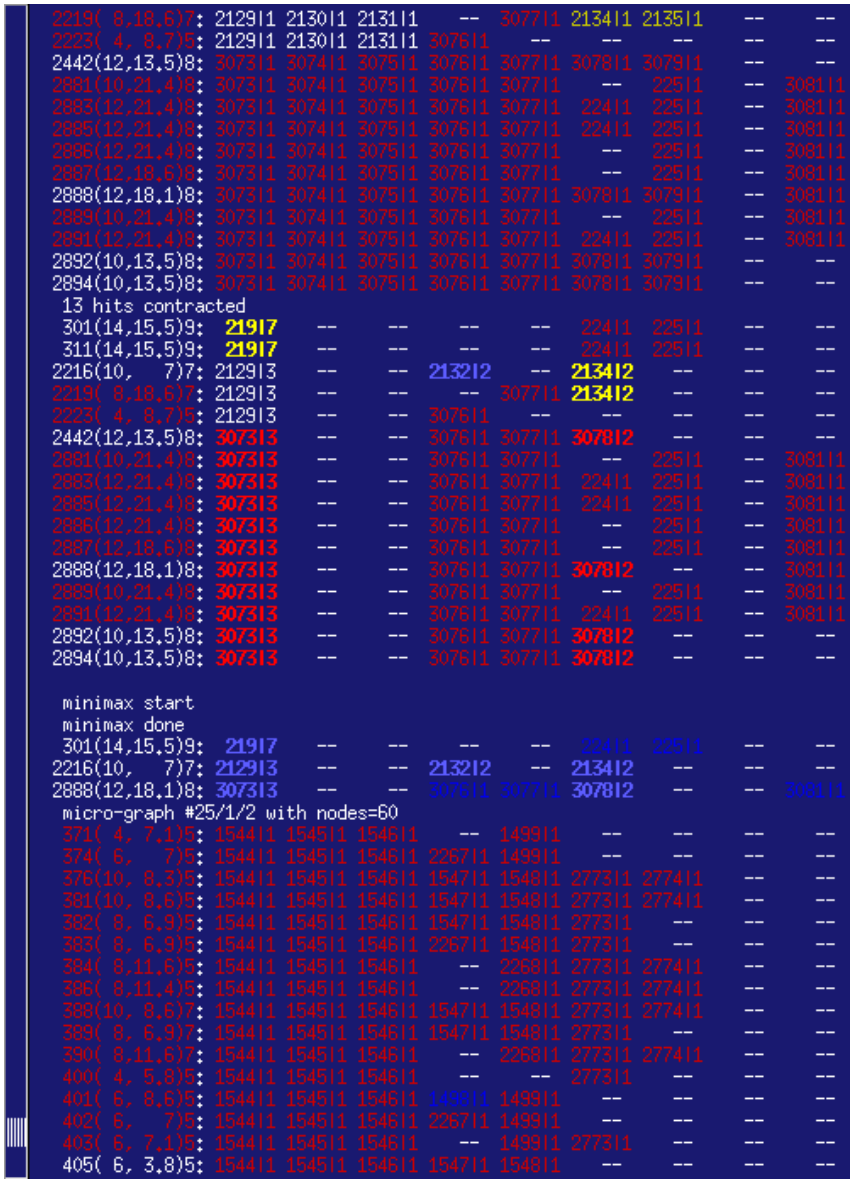
Might look for double-edges (2-edge connectivity) as well

Contracting hits with identical role



Next: optimize the single-player game

Solve the subgraphs – game tree



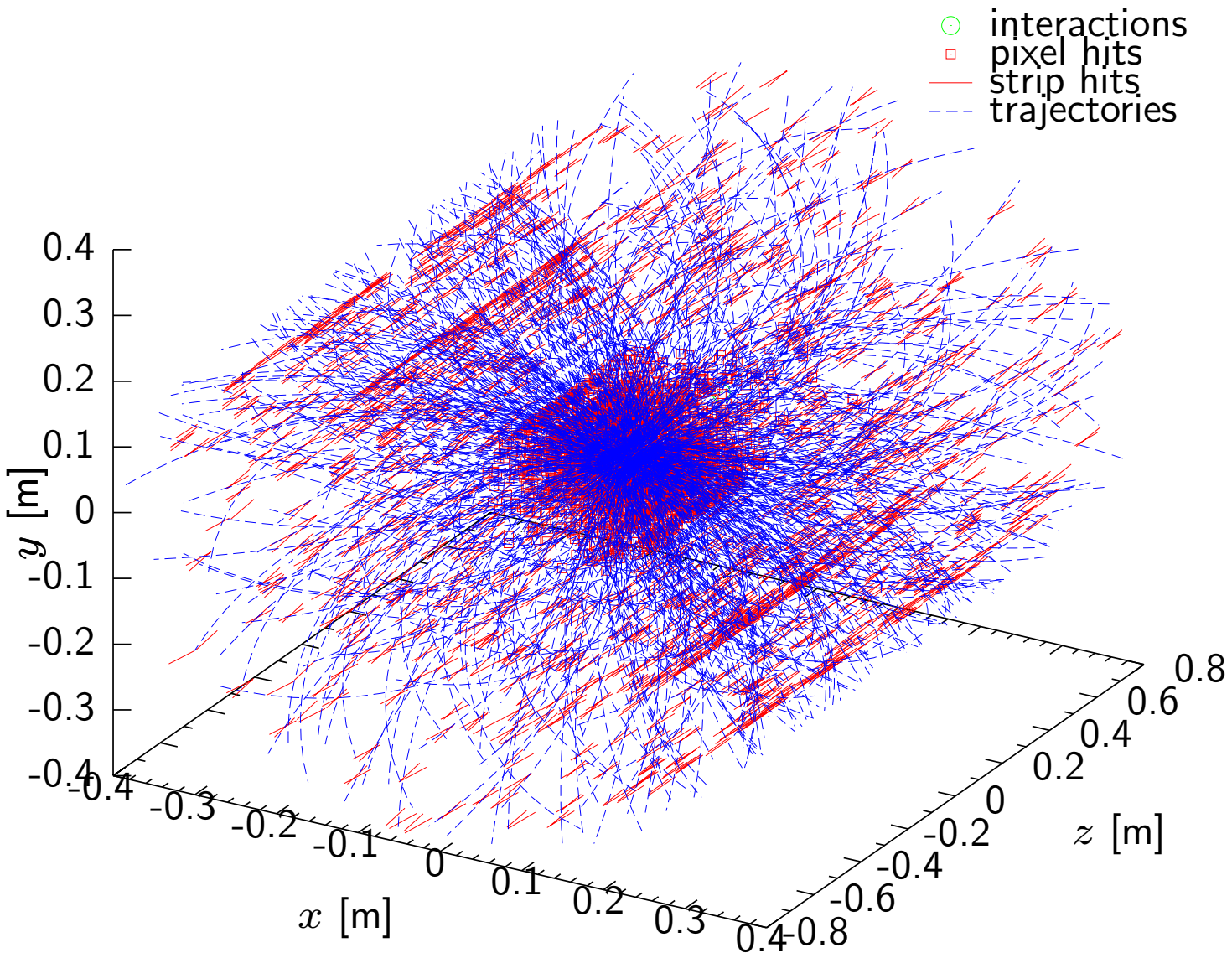
- Explore
  1. choose the most wanted hit
  2. assign it to a track (loop over)
  3. store tracks with at least two lonely hits (those not requested by other tracks)
  4. remove tracks with too few hits
  5. if there are hits left, go to $\rightarrow$ 1.
  6. evaluate # of hits on track, $\sum \chi^2$
  7. if best score so far $\rightarrow$ take note

Recursive

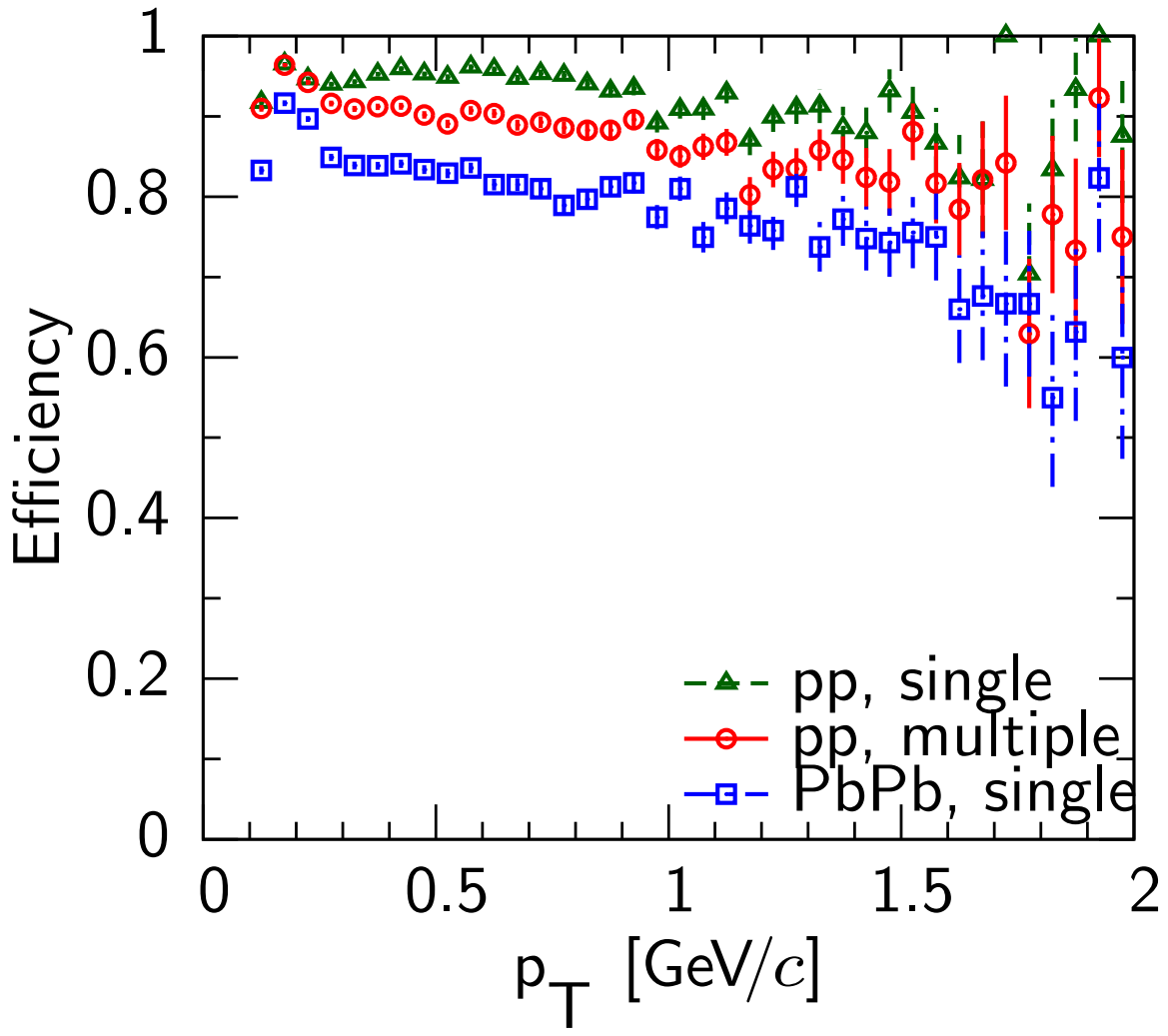
Depth-limited, horizon ($\sim$chess)

Gives optimal distribution of hits

Result

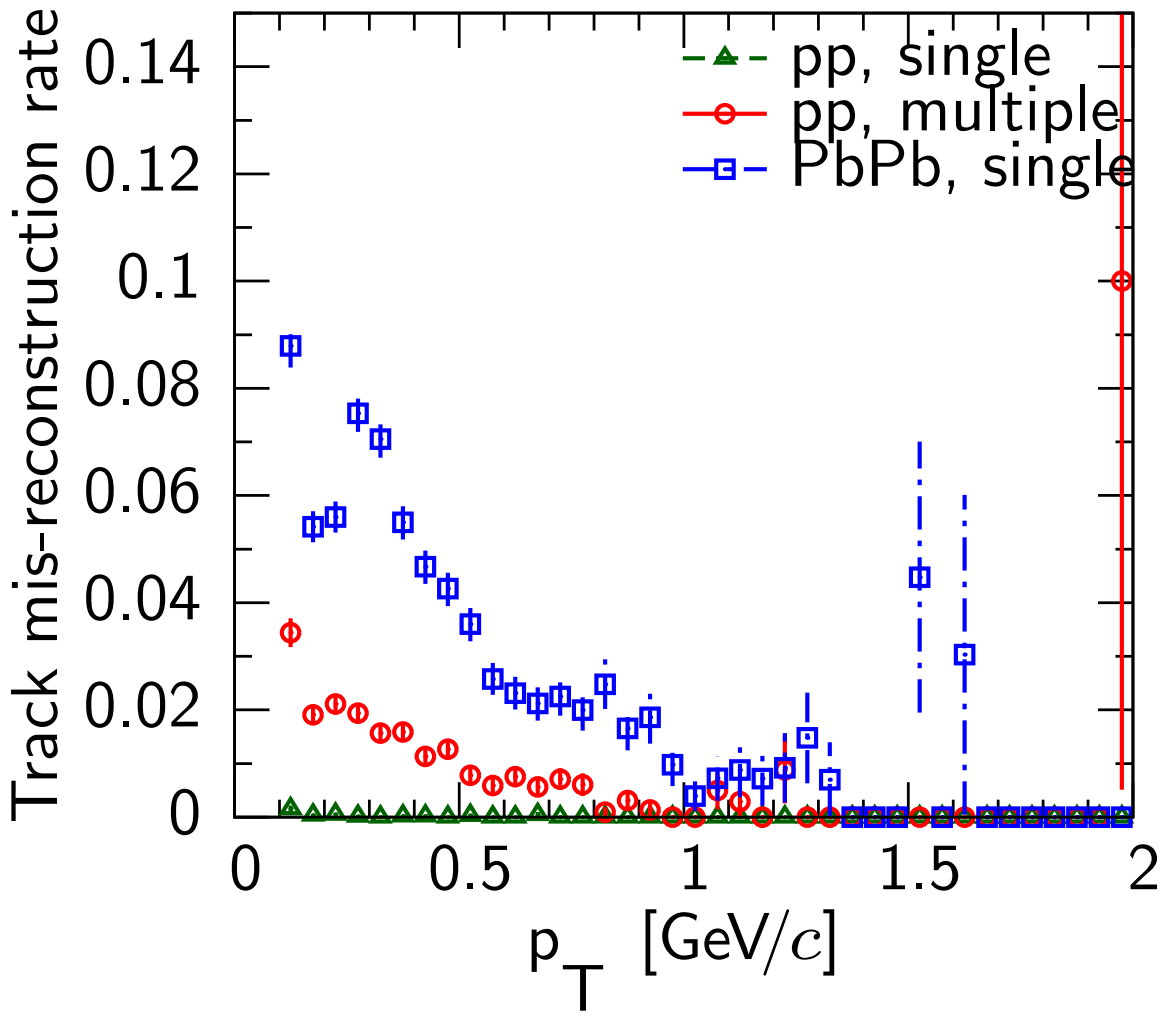

interactions
pixel hits
strip hits
trajectories
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
y [m]
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
x [m]
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
z [m]

Performance – track finding efficiency



Looks good

Performance – fake track rate



Nice, event at low p_{T}

Summary

- Several key ideas
  - use global information: image transformation with templates
  - keep choices open: graph representation
  - find vulnerable edges and nodes, remove them
  - analyze disconnected parts as a deterministic single-player game
  - maximize the number of hits on track, then minimize $\sum \chi^2$
  - many steps employ parallel computing, good timing
  - ☺ looks promising: down to very low p_{T} and also in high pileup

- Status and plans
  - ☺ several mathematical concepts and algorithms at work, fun!
  - standalone simulation-reconstruction software package exists
  - paper draft very soon

Thanks for your attention