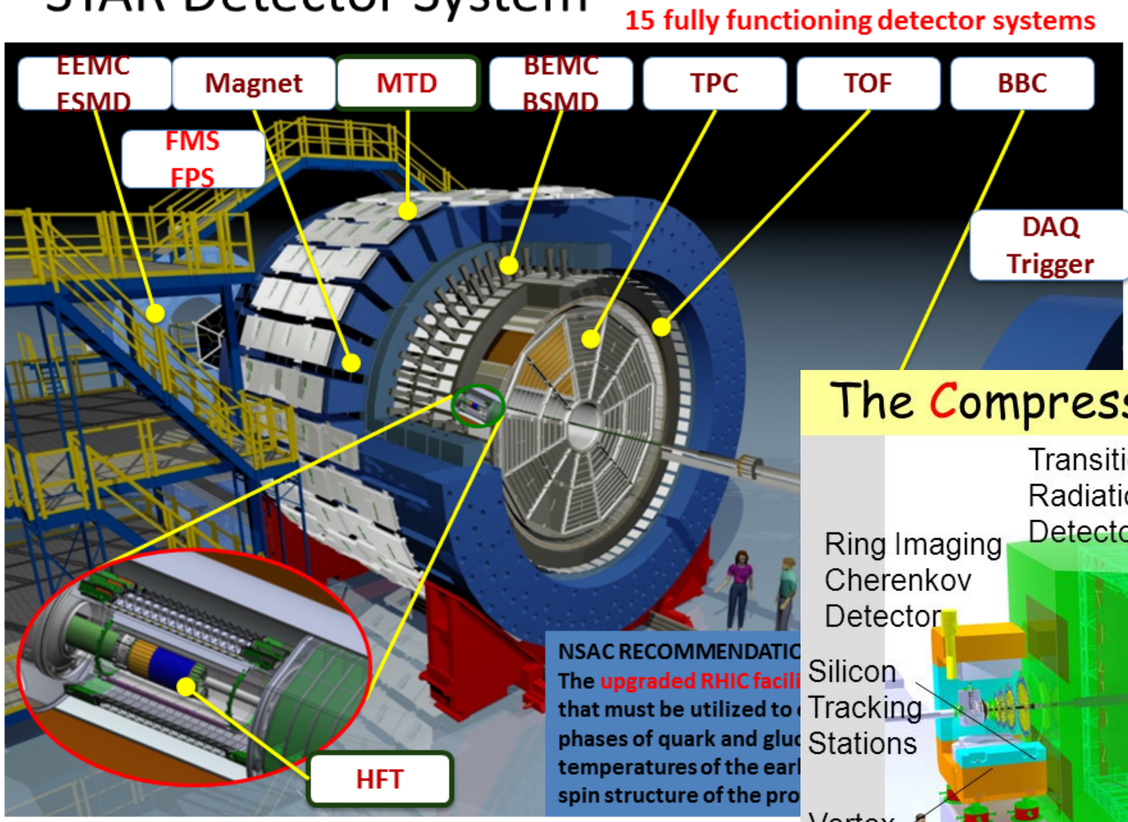
Speed up approaches in the Cellular Automaton track finder

G.Kozlov[1,2,3], I.Kisel[1,2]

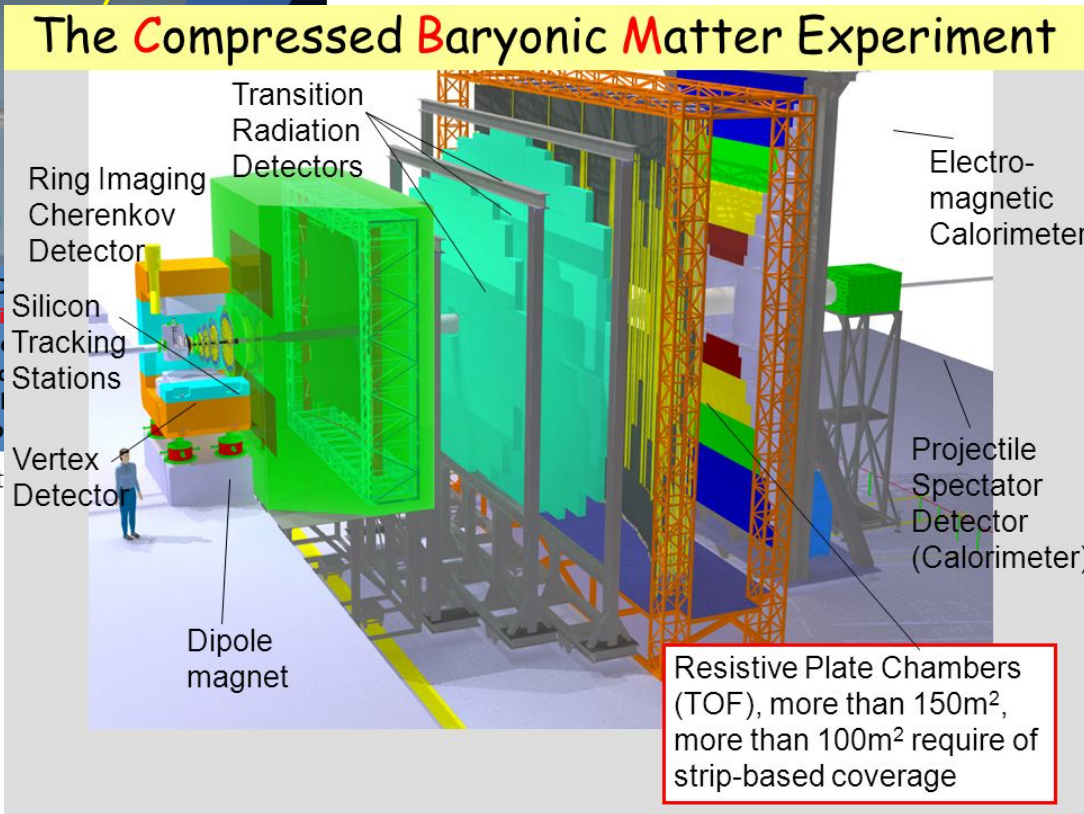
[1]Goethe-Universitat, Frankfurt, Germany
[2]FIAS, Frankfurt, Germany
[3]JINR LIT, Dubna, Russia

[FAIR phase 0]

STAR Detector System

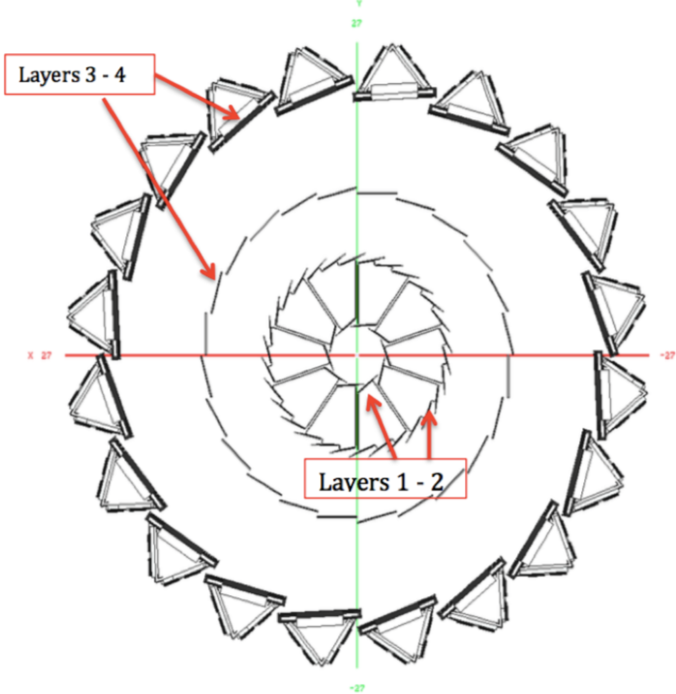


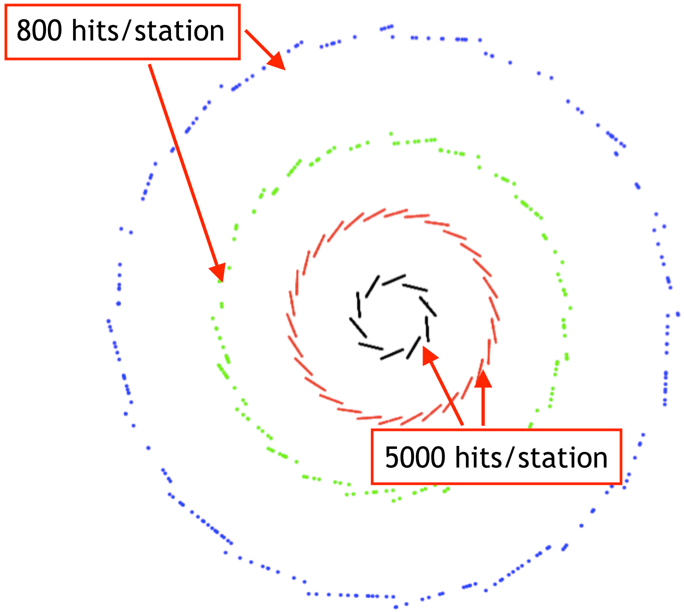
X10³ increases in DAQ rate since 2000, most precise Silicon Detect



HFT detector at STAR

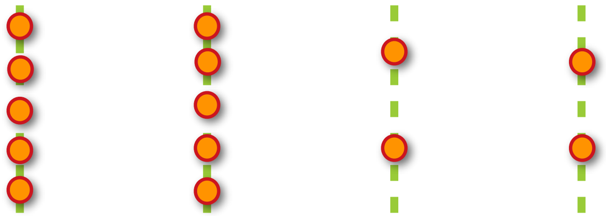
- 4-layers:
  - Layer 1 - 2: PIXEL (2.5 cm, 8 cm)
  - Layer 3: IST (14 cm)
  - Layer 4: SST (22.3 cm)
- Up to 800 tracks per event
- Pileup on each PIXEL layer is about 5000 hits
- HFT CA tracking procedure is similar to CBM L1 track finder





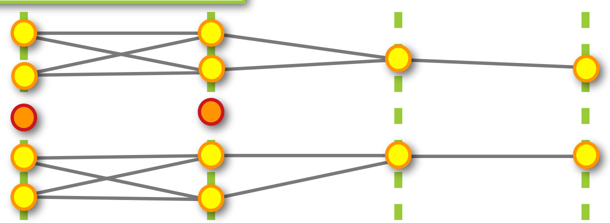
CA track finding procedure

1. Hits



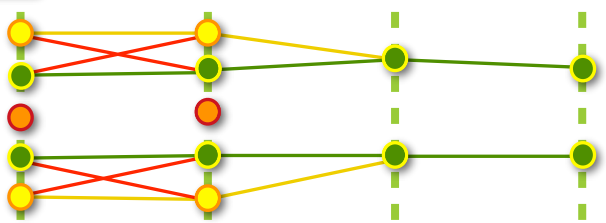
Read input hits, sort them in memory taking into account grid structure and associate them with grid.

2. Track segments



Find all possible track segments - triplets using grid structure, select those that meet the criteria, mark neighbours.

3. Tracks



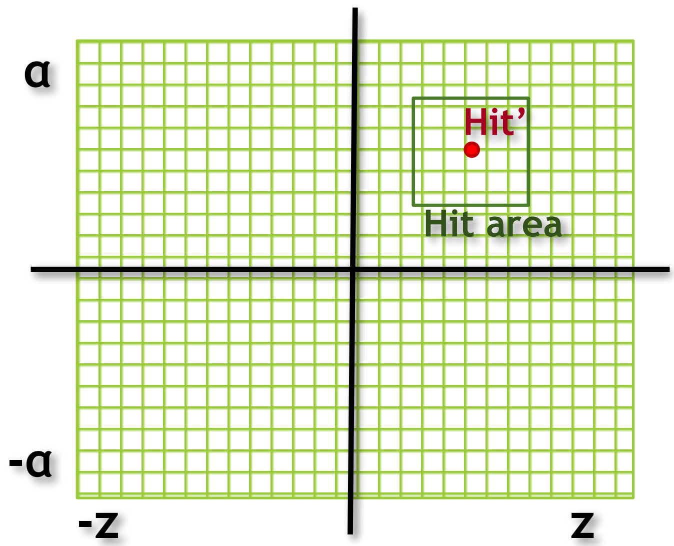
Collect track segments into track candidates, fit them, select the best candidates and save output tracks.

Speed up approaches

- Direction of tracking from outer station
- Grid structure
- Exclude singlets from calculations
- Calculate triplets directly from hits without doublets
- Include neighbouring triplets finding into triplet calculation step
- Vectorization
  - Triplets
  - Tracks

Grid structure in HFT

Grid structure allows to establish compliance between hit coordinates and bins of this structure.



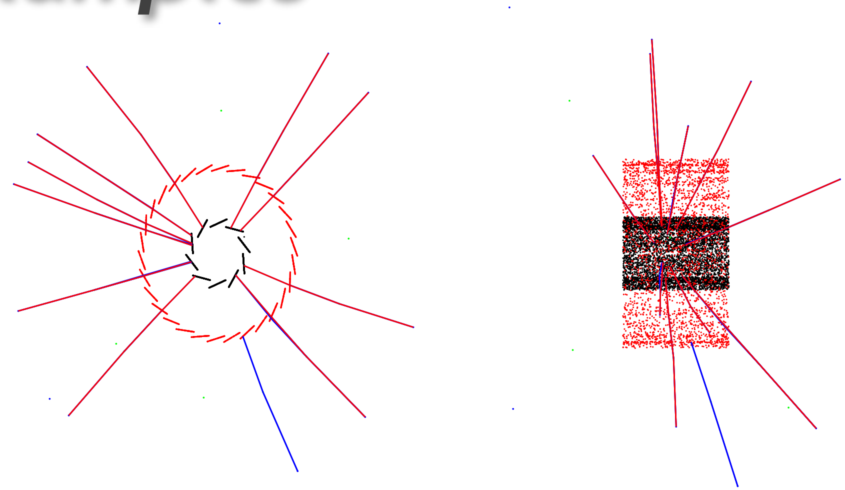
- ***Grid*** is based on *Z-coordinate* and *angle*.
- Track finding direction – *from outer station*.
- Main steps of grid usage:
  - Extrapolate hit -> hit' to the previous station in direction of primary vertex (*PV*);
  - Create *Hit area* around the hit' using ***dz*** and ***da***;
  - Search for the next hit inside of the *Hit area*.
- Number of bins in grid depends on the number of hits on station.

Track finding examples

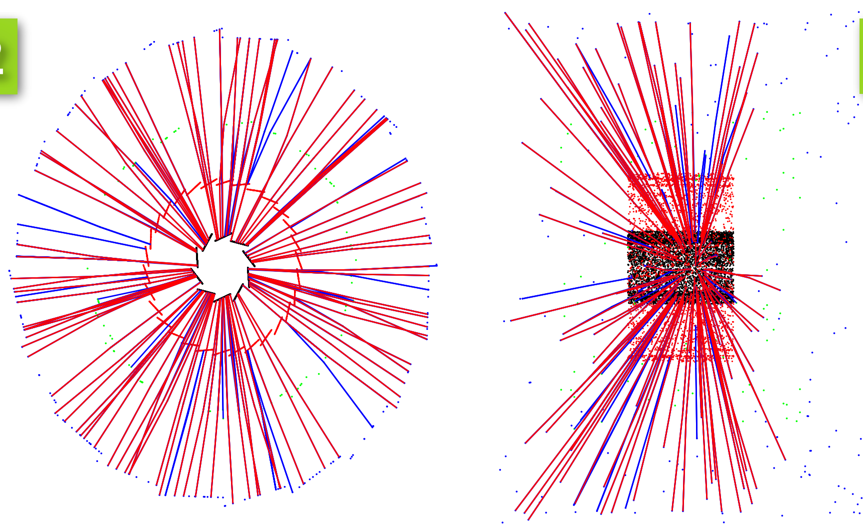
Triplets

Tracks

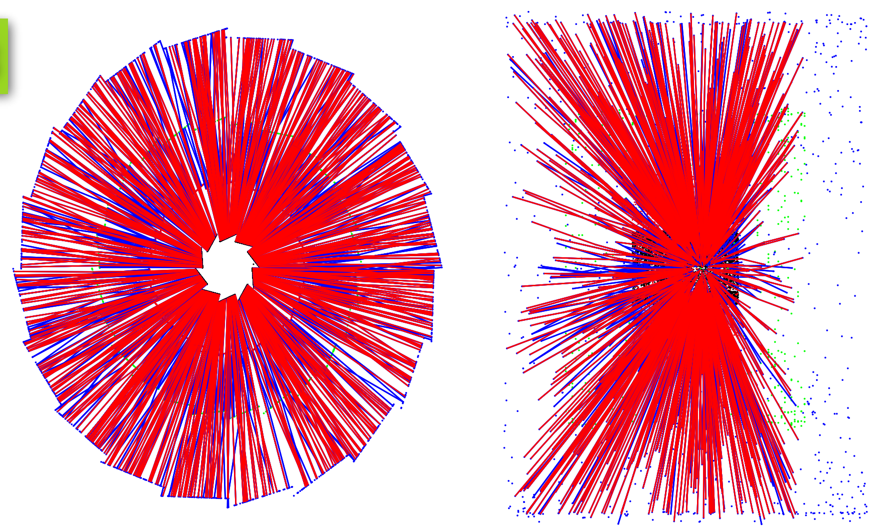
1



2



3



Data structures

1. Hits are located in memory as an array of structures

Hit 1				Hit 2				Hit 3				Hit 4				Hit 5				Hit n			
X	Y	Z	…	X	Y	Z	…	X	Y	Z	…	X	Y	Z	…	X	Y	Z	…	X	Y	Z	…

Good for sorting

2. For vectorized calculations hits are stored as a structure of arrays

X1	X2	X3	X4	X5	X6	X7	Xn	…
Y1	Y2	Y3	Y4	Y5	Y6	Y7	Yn	
Z1	Z2	Z3	Z4	Z5	Z6	Z7	Zn	
…								

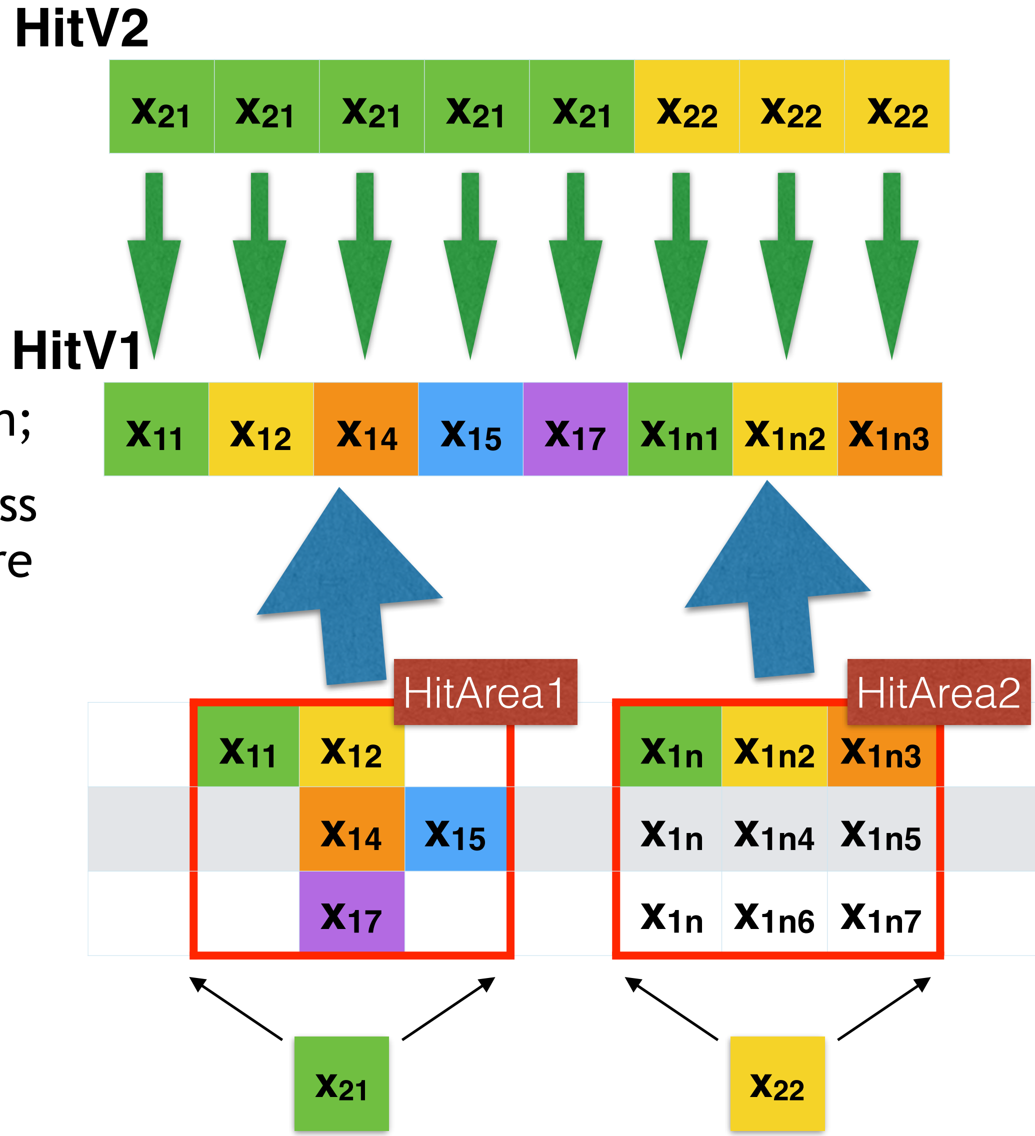
Good for vectorisation

3. Within vectorized calculations every portion of hits is used as a structure of SIMD vectors

Hit															
X1	X2	X3	X4	Y1	Y2	Y3	Y4	Z1	Z2	Z3	Z4	…	…	…	…

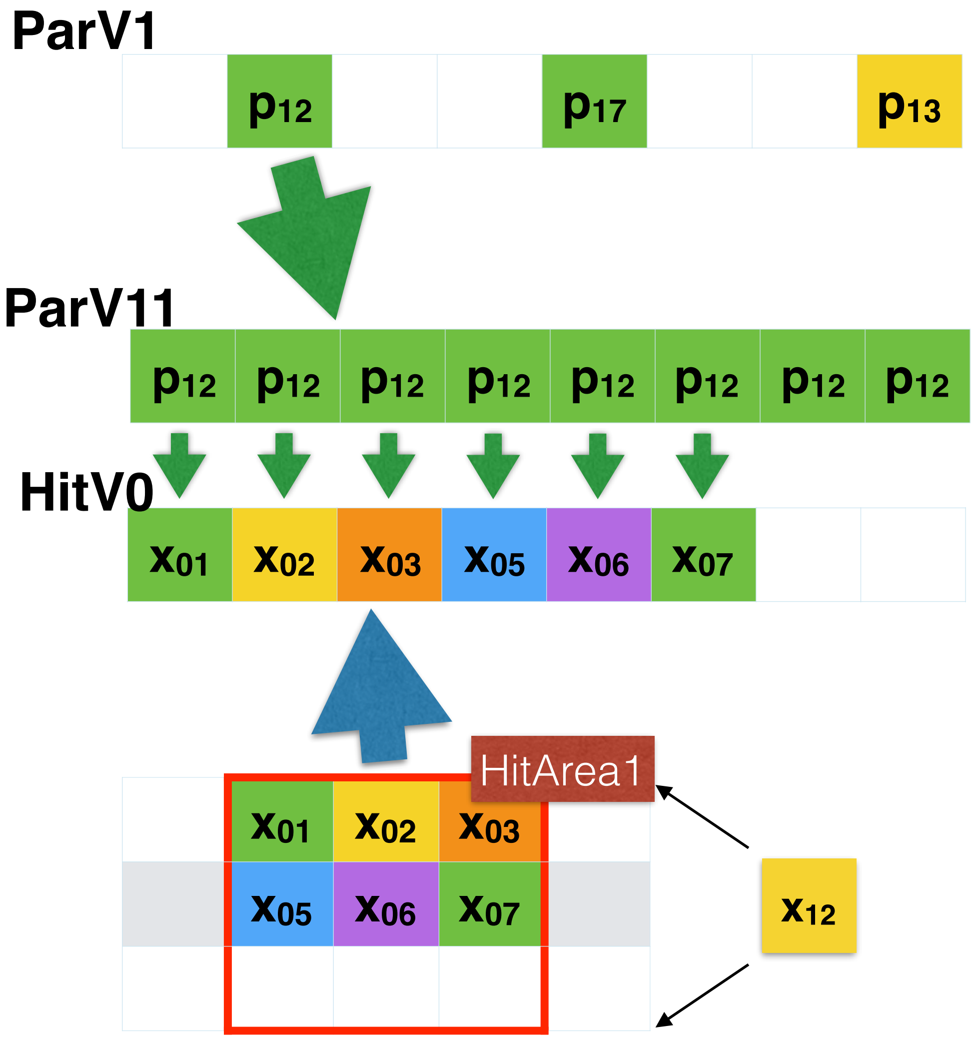
Triplet finding vectorization

- ***First segment calculation:***
  - Take hit from outer station;
  - Extrapolate it to PV, create HitArea and look over the hit candidates from middle station;
  - If the number of candidates less than vector size, take one more outer hit;
  - Create maximal filled hit vectors;
  - Initialise vector of parameters for the outer hit;
  - Transport it to the middle station in vector mode, check cuts.



Triplet finding vectorization

- ***Second segment calculation:***
  - Take active element from parameters vector and fill new parameters and hits vectors by this element;
  - Extrapolate corresponding middle hit to PV, create HitArea and look over the hit candidates from inner station;
  - Create vector of inner hits;
  - Transport to the inner station in vector mode, check cuts;
  - Save triplets and go to the next active element from parameters vector.



Calculation speed

Efficiency: 90.3% (75% correct hits)

Efficiency: 81.8% (100% correct hits)

Calculation time in ms/event

	Scalar	Vector SSE(gcc)	Vector SSE(icc)	Vector AVX-1(icc)
-Input hits	*0.6*	*0.4*	*0.4*	*0.4*
Sort and fill grid	4.7	1.4	2.0	2.0
Triplets	27.1	4.9	3.9	3.4
Tracks	4.7	1.3	1.2	1.1
-Output tracks	*0.8*	*0.7*	*0.7*	*0.7*
Total	37.0	7.7	7.1	6.5

Efficiency

	75% correct hits	100% correct hits	MC Tracks / event
All tracks	90.3%	81.8%	237.2
Primary high-p	98.1%	93.7%	27.4
Primary low-p	92.2%	83.3%	198.1
Secondary high-p	83.6%	76.0%	0.7
Secondary low-p	36.5%	26.1%	10.9
Ghost level	17.5%	36.9%	
Correct tracks	215	194	
Time / event	6.5 ms		

Summary

- Optimal direction of tracking allows as to reduce an effect of pile up on the inner stations.
- Grig structure strongly decreases the combinatoric level within track segments calculation.
- Memory usage was strongly decreased by excluding of singlets and doublets from calculations.
- Neighbours finding was combined with triplets calculation.
- Vectorization of triplets and tracks calculation gives us speed up with factor up to 4.8 on SSE and 5.7 on AXV.

Next steps

- Developed approaches will be integrated into CBM L1 track finder to additionally speed up it and make CA track finder uniform for both experiments.