Intelligent triggering: pattern recognition with Associative Memories & other tools (FPGAs) Fast TracKer & Hardware Tracking for the Trigger (HTT) examples in ATLAS



Kostas Kordas

Aristotle University of Thessaloniki





International School of Trigger and Data Acquisition



ISOTDAQ2022 Catania, Italy

June 22, 2022

Overview

- The need for tracking information at the Trigger of High Energy Physics experiments and how to do it fast
- We'll split the problem into "track finding" (define fast a "road" where a track can be) and "track fitting" (determine the track characteristics)
- The specific examples from ATLAS (FTK and HTT) use
 - Track finding with Pattern matching in Associative Memories , and Track fitting in FPGAs
- Basically you'll see that: if you want to avoid or cannot afford calculating something time consuming, split the problem and use pre-calculated patterns and quantities.
- We'll see also examples of other approaches, with both steps done in FPGAs.
- We'll also see examples beyond High Energy Physics

A. Introduction

Experiments at the LHC - pp collisions





ISOTDAQ2022, Catania, 22/6/2022 K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs

5

Looking at many & complex events every 25ns two proton bunches cross each other → a superposition of >25 pp collisions



Atlas event with a Z boson decaying to two muons and 24 additional interaction vertices.

The Trigger and Data Acquisition system,

* watches 40M such "events" (bunch crossings) / sec \rightarrow O(1) billion pp interactions per second

* select online "the most interesting" O(1k) events/sec (1 : 1 Million pp interactions deemed interesting enough to keep)

* and log them for offline use with a resolution of a
 ~100 Mpixel camera (100M channels: total ~1.5 MB/event)

Trigger at 2 stages: Level1 (L1: fast, no detailed info) & High Level Trigger (HLT: slower, using detailed info)

• Trigger & DAQ : Select events and get the data from the detector to the computing center for the first processing.



Example: Looking for Higgs

How do we see the Higgs?

→ from its children!

E.g., 4 muons traversing the detector (red lines here)



$$E = mc^{2} \longrightarrow E^{2} = m^{2}c^{4} + p^{2}c^{2} \longrightarrow E^{2} = m^{2} + p^{2} \longrightarrow m = \sqrt{(E^{2} - p^{2})}$$

Selection of the right events is essential (and selection of many of them too!) e.g., ATLAS: $H \rightarrow ZZ \rightarrow 4I$

./4I-FixedScale-NoMuProf2_400x300.gif

./4I-FixedScale-NoMuProf2_238x231.gif



9

https://twiki.cern.ch/twiki/pub/AtlasPublic/HiggsPublicResults//4I-FixedScale-NoMuProf2.gif

The more you know about the events, the easiest you select the "signal" and reject the "background"

When there is limited time budget (L1 trigger): typically, decide based only on the muon and calorimeter systems

But may need information from the inner tracker as early as possible to make an "educated" decision and keep as much signal as possible e.g., 2 "jets" of tracks, which are usually boring, they could actually be $H \rightarrow b b$ or $H \rightarrow \tau^+ \tau^-$

You will hear on the last day from Francesca Pastore the way various experiments trigger on the "interesting events"

Each charged particle leaves a trace ("a track") in the detector as it moves outwards



Atlas event with a Z boson decaying to two muons and 24 additional interaction vertices.



- Connecting the "hit" readout cells from one detection layer to the other
- traces the charged particle's path as it moves radially outward and its' position is measured in each detector layer

You have also noise and irrelevant hits on the same "event"/"picture"



Atlas event with a Z boson decaying to two muons and 24 additional interaction vertices.



2 "real tracks" + extra hits

Tracking is a combinatorics problem: which combinations of hits fit track hypothesis?



Atlas event with a Z boson decaying to two muons and 24 additional interaction vertices.



ISOTDAQ2022, Catania, 22/6/2022

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 13

Tracking is a combinatorics problem: which combinations of hits fit track hypothesis?



Atlas event with a Z boson decaying to two muons and 24 additional interaction vertices.



But when you look at this event/picture, **you just see hits!**

You have to find the tracks...

 → Lots of hit combinations to try
 → a huge combinatorics problem
 → becoming worse and worse as luminosity increases
 → a big burden on CPUs

B. Associative Memories for track finding & FPGAs for track fitting



- The basic technique between the use in CDF II SVT, and the ATLAS FTK and HTT are the same for what I want to discuss here; so I give FTK as an example
- FTK is a hardware pre-*processor* finding tracks and storing them for further usage by the trigger
- HTT is a co-processor who is ordered by other components to the find tracks for them

FTK (Fast TracKer): dedicated hardware helping the HLT, by doing the tracking before the HLT

Permanent storage

Fast TracKer (FTK), a pre-processor for a CPU farm For each event accepted by L1 (100kHz), find all its tracks in <100 μ sec \rightarrow x1000 faster than the HLT farm of PCs

FTK

High Level Triggers: PC farms ~0.2s, ~1 kHz

L1: hardware {2.5μs, 100 kHz}

FTK: Tracking particles in the Silicon Detectors



ISOTDAQ2022, Catania, 22/6/2022

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 17

From hits to tracks in $< 100 \ \mu s$



1. Input & Data "Formatting": cluster adjacent hits, find the position of each cluster, forward them to the Processing Unit responsible for this geometrical η-φ region (64 η-φ towers)





2. Processing Unit: tracking in 2 steps (see analogy to the Trigger doing L1 & HLT)

 Find low resolution track candidates called "roads". Solve most of the combinatorial problem.



Pattern recognition w/ Associative Memory

Originally: M. Dell'Orso, L. Ristori, NIM A 278, 436 (1989)

Then track fitting inside the roads.
 Thanks to 1st step, this is much easier.



http://www.pi.infn.it/~orso/ftk/IEEECNF2007_2115.pdf

Excellent results with linear approximation!

2a. The coarse pattern matching first

In SVT, FTK and HTT: use 8 layers

The detector's finite resolution makes it "binned"→ finite number of "hit patterns"



Training: simulated tracks to find possible patterns



ISOTDAQ2022, Catania, 22/6/2022

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 24

Coarse track finding = pattern matching: does your event contain any of these patterns?



Compare ALL the hits in each event with ALL the stored patterns.



How to match data to patterns?



Number of patterns in your bank gets big easily

N_n = number of **straight lines** crossing the detector layers



For a detector with 8 layers, with 1M channels/layer, $N_p = 7 \ 10^{12}$!!!

(Re-bining with 2-channels per bin: $n \rightarrow n/2$ means Np $\rightarrow \frac{1}{4}$ Np)

patterns and search time are critical

- Need a lot of memory for the patterns:
 - OK, can use larger ("coarser") bins for 1st pattern matching (will come back to this later).
- But still, you have to match hits with patterns fast:
 - Linear search, of the pattern-table ("brute force") is the slowest.
 - If list of patterns is ordered, can do "binary" search:
 - Pick the middle element in the list,
 - Compare the data to the pattern to find the good half of the list,
 - pick the middle of the new (halved) list, and so on.

30

Example: The list to be searched: L = 1 3 4 6 8 9 11. The value to be found: X = 4.

Compare X to 6. X is smaller. Repeat with L = 1 3 4. Compare X to 3. X is bigger. Repeat with L = 4. Compare X to 4. They are equal. We're done, we found X.

Speed is extremely important at triggering. Find tracks at ultimate speed → Use "Associative Memories"

Ultimate speed for pattern matching: do it during the I/O, as the data go through the system \rightarrow no "processing time"

Associative Memory in a VLSI

436

Nuclear Instruments and Methods in Physics Research A278 (1989) 436-440 North-Holland Amsterdam

October 24, 1988

VLSI STRUCTURES FOR TRACK FINDING

Mauro DELL'ORSO

Dipartimento di Fisica, Università di Pisa, Piazza Torrizelli 2, 56100 Pisa, Italy

Luciano RISTORI

INFN Sezione di Pisa, Via Vecchia Licornese 582a, 56010 S. Piero a Grado (PE), Italy

Received 24 October 1988

We discuss the architecture of a device based on the concept of associative memory designed to solve the track finding problem, typical of high energy physics experiments, in a time span of a few microseconds even for very high multiplicity events. This "machine" is implemented as a large array of custom VLSI chips. All the chips are equal and each of them stores a number of "patterns". All the patterns in all the chips are compared in panillel to the data coming from the detector while the detector is being read out

1. Introduction

The quality of results from present and future high energy physics experiments depends to some extent on the implementation of fast and efficient track finding algorithms. The detection of heavy flavor production, for example, depends on the reconstruction of secondary vertices generated by the decay of long lived particles, which in turn requires the reconstruction of the majority of the tracks in every event.

Particularly appealing is the possibility of having detailed tracking information available at trigger level even for high multiplicity events. This information could be used to select events based on impact parameter or secondary vertices. If we could do this in a sufficiently short time we would significantly enrich the sample of events containing heavy flavors.

Typical events feature up to several tens of tracks each of them traversing a few position sensitive detector layers. Each layer detects many hits and we must correctly correlate hits belonging to the same track on different layers before we can compute the parameters

2. The detector

In this discussion we will assume that our detector consists of a number of layers, each layer being segmented into a number of bins. When charged particles cross the detector they hit one bin per layer. No particular assumption is made on the shape of trajectories: they could be straight or curved. Also the detector layers need not be parallel nor flat. This abstraction is meant to represent a whole class of real detectors (drift chambers, silicon microstrip detectors etc.). In the real world the coordinate of each hit will actually be the result of some computation performed on "raw" data: it could be the center of gravity of a cluster or a charge division interpolation or a drift-time to space conversion depending on the particular class of detector we are considering. We assume that all these operations are performed upstream and that the resulting coordinates are "binned" in some way before being transmitted to our device.



We discuss the architecture of a device based on the concept of associative memory designed to solve the track finding problem, typical of high energy physics experiments, in a time span of a few microseconds even for very high multiplicity events. This "machine" is implemented as a large array of custom VLSI chips. All the chips are equal and each of them stores a number of "patterns". All the patterns in all the chips are compared in parallel to the data coming from the detector while the detector is being read out.

Associative Memory (AM) = a kind of Content Addressable Memory (CAM)

- CAM = a memory that is accessed by its contents, not its location.
- E.g., while in a RAM we ask:

- what do you have in location **xyz**?

- In a Content Addressable Memory (CAM) we ask:
 - Are there any locations holding the value *abc*?

Binary CAMs

Binary CAM (simplest):

uses search words consisting entirely of "1" and "0"

Example:

stored word of

-----> "**10110**" ("one<u>pattern</u>")

It will be matched by the search word: "10110" ("the <u>data</u>")

Associative Memory: CAM cells contain pattern bank. <u>CAM cells</u> of same Layer are on a common bus



Associative Memory: CAM cells check the macthing of each hit independently



Associative Memory: CAM cells check the macthing of each hit independently


Associative Memory: CAM cells check the macthing of each hit independently



ISOTDAQ2022, Catania, 22/6/2022

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 38

Associative Memory: CAM cells check the matching of each hit independently



ISOTDAQ2022, Catania, 22/6/2022

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 39



Unique to AM chip: look for correlation of data received at different times.

AM evolution: ASICs (mainly)



- (90's) Full custom VLSI chip 0.7µm (INFN-Pisa)
- 128 patterns, 6x12bit words each, 30MHz
- F. Morsani et al., IEEE Trans. on Nucl. Sci., vol. 39 (1992)



Alternative FPGA implementation of SVT AM chip

- P. Giannetti et al., Nucl. Intsr. and Meth., vol. A413/2-3, (1998)
- G Magazzù, 1st std cell project presented @ LHCC (1999)



Standard Cell 0.18 μ m \rightarrow 5000 pattern/AM chip SVT upgrade total: 6M pattern, 40MHz A. Annovi et al., IEEE TNS, Vol 53, Issue 4, Part 2, 2006



AMchip04 –65nm technology, std cell & full custom, 100MHz Power/pattern/MHz ~30 times less. Pattern density x12. First variable resolution implementation!

F. Alberti et al 2013 JINST 8 C01040, doi:10.1088/1748-0221/8/01/C01040

AM chip for FTK: AMchip06



- 90's Full custom VLSI chip 0.7mm (INFN-Pisa) 128 patterns, 6x12bit words each (F. Morsani et al., The AMchip: a Full-custom MOS VLSI Associative memory for Pattern Recognition, IEEE Trans. on Nucl. Sci., vol. 39, pp. 795-797, (1992).)
- 1998 FPGA for the same AMchip (P. Giannetti et al. A Programmable Associative Memory for Track Finding, Nucl. Intsr. and Meth., vol. A413/2-3, pp.367-373, (1998)).
- 1999 G. Magazzù, first standard cell project presented at LHCC
- 2006 Standard Cell UMC 0.18 µm 5000 pattern/AMchip for CDF SVT upgrade total: 6M patterns (L. Sartori, A. Annovi et al., A VLSI Processor for Fast Track Finding Based on Content Addressable Memories, IEEE TNS, Vol 53, Issue 4, Part 2, Aug. 2006)
- 2012 AMchip04 8k patterns in 14mm2, TSMC 65nm LP technology Power/pattern/MHz 40 times less. Pattern density x12. First variable resolution implementation. (F. Alberti et al 2013 JINST 8 C01040, doi:10.1088/1748-0221/8/01/C01040)
- 2013-2014 AMchip MiniAsic and AMchip05

a further step towards final AMchip version. Serialized input and output buses at 2 Gbs, further power reduction approach. BGA 23 x 23 package.

2014-2015 AMchip06: final FTK version of the AMchip for the ATLAS experiment .

AMchip06: the FTK AM chip has 128k patterns/chip AMchip08→09: the AM chip for HTT: ~400k pat./chip

ISOTDAQ2022, Catania, 22/6/2022

Pattern matching not restricted to trackers. Applies everywhere there is correlated behaviour. e.g:



The values in X1 and X2 are correlated: their values define the possible {X1,X2} patterns

> For example, task = Associate the measured X1 and X2: e.g., X1 = 5 with X2 = 8



2b.

Now that we have a system that does pattern matching as the data are coming in,

storage problem: how do we deal with the number of patterns which can be big in the high-granularity detectors?

N_{patterns} depends on the "bin size", i.e, the granularity with which we want to look at the detector



High efficiency with large bins (small number of patterns) → but, lots of fake tracks found → lots of work for detailed track fitting!

We want "few patterns"-"few fakes" scenario



Use the feature of "ternary CAMs"

- **Ternary CAM**: added flexibility to the search
 - allows a third matching state of "X" or "Don't Care" for one or more bits in the stored pattern word: one pattern matches various data words
- Example: a ternary CAM might have a

stored word of

"**10XX0**" ("one<u>pattern</u>")

This will match any of 4 search words: "10000" ("the data")

---->

"10010" ("the <u>data</u>")

"10100" ("the <u>data</u>")

"10110" ("the <u>data</u>")

The added flexibility comes at additional cost:

 the internal memory cell must now encode three possible states instead of the two of binary CAM. This additional state is typically implemented by adding a mask bit ("care" or "don't care" bit) to every memory cell.

Variable resolution (bin sizes) with "Don't Care" (DC) bits

Alberto Annovi

 * ANIMMA - A new "Variable Resolution Associative Memory" for High Energy Physics ATL-UPGRADE-PROC-2011-004, doi:10.1109/ANIMMA.2011.6172856
 * "Variable resolution Associative Memory for the Fast Tracker ATLAS upgrade", ICATTP 2013

- For each layer: a "bin" is identified by a number with DC bits (X)
- Least significant bits of "bin" number can use 3 states (0, 1, X)
- The "bin" number is stored in the Associative Memory
- The DC bits can be used to OR neighborhood high-resolution bins, which differ by few bits, without increasing the number of patterns

Pixels:

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

Refinements: majority & variable widths

- Majority Logic: Only require N out of M layers have a match
 - Gains efficiency
- Variable Resolution Patterns (Don't Care Bits)
 With 2 DC bits: Apart from reduction in fakes (factor 7), we save also a factor 5 in the size of the pattern bank!



Technique can be exploited by any coincidence based trigger!

Alberto Annovi

 * ANIMMA - A new "Variable Resolution Associative Memory" for High Energy Physics ATL-UPGRADE-PROC-2011-004, doi:10.1109/ANIMMA.2011.6172856
 * "Variable resolution Associative Memory for the Fast Tracker ATLAS upgrade", ICATTP 2013

So, we have found possible tracks (the matched patterns)

Each matching pattern defines a "road" for the refined tracking fetch all the (few now) hits in the road fit them to a helical track to measure the track parameters precisely

Track fitting in FPGAs: 1st stage, 8 layers



Track fitting in FPGAs w/ many Digital Signal Processors (DSPs)
BUT: Linear approximation: get a set of linear equations
"each parameter depends linearly on the hits" → fast
multiplications with pre-computed constants ~1 Gfits/s per FPGA
constants in memory & speed of retrieval limiting factor
TSOTDAQ2022, Catania, 22/0/2022
K. Kordas - Pattern Recognition W/ Associative Memories & FPGAs
52

Track fitting in FPGAs: 2nd stage (12 layers in FTK, 13 layers in HTT)



Done on FPGAs, on a "2nd stage" board

Approximations and Tables with constants are very common calculation tools. Don't be afraid of them!

- Constants can be coefficients in Taylor expansions, Fourier series, etc. e.g.,
 - $sin(x) = Taylor expansion gives a polynomial to calculate <math>sin(x) \simeq x x^3/6 + x^5/120$
- Or, use Look-Up Tables (LUTs = pre-calculated values stored in tables) → interpolate between stored values to get value of sin(x) you ask for



FTK : working configuration

- High resolution patterns: (15x36)_{pix}x16_{sct}
 - Pixels: 15 channels along $\phi,$ 36 ch. along η
 - Strips: 16 strips
- Background events with 69 superimposed pp collisions
 - Instantaneous luminosity 3*10³⁴ Hz/cm²
- Hardware constraints (for each of 64 η - ϕ towers)
 - # AM patterns < 16.8 * 10⁶
 - # roads/event < 16 * 10³
 - # fits/event < 80 * 103

channels together an
increase the pattern
resolution

Work load for track fitter

DC bits group detector

	Coarse resolution roads	Max # DC bits / layer	# AM pattern * 10 ⁶	Efficien cy %	roads / evt * 10 ³	fits / evt * 10 ³					
Barrel	(30x72) _{pix} x32 _{sct}	$2_{pix} x 1_{sct}$	16.8	93.3%	3.2	26					
Endcap	(30x72) _{pix} x32 _{sct}	$2_{pix} \times 1_{sct}$	16.8	91.2%	6.9	55					

Alberto Annovi

* ANIMMA - A new "Variable Resolution Associative Memory" for High Energy Physics ATL-UPGRADE-PROC-2011-004, doi:10.1109/ANIMMA.2011.6172856 * "Variable resolution Associative Memory for the Fast Tracker ATLAS upgrade", ICATTP 2013

55

Some documentation for details

FTK Technical Design Report (TDR): https://cds.cern.ch/record/1552953?In=en https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/UPGRADE/CERN-LHCC-2013-007/index.html

HTT described (some changes since then) in: ATLAS Trigger and Data Acquisition Phase-II Upgrade Technical Design Report. Tech. rep. ATL-COM-DAQ-2017-185. https://cds.cern.ch/record/2296879

FTK Public results: https://twiki.cern.ch/twiki/bin/view/AtlasPublic/FTKPublicResults

<u>A word on strategic decisions. FTK is not used now:</u>

the luminosity expected for the Run3 of LHC (2022-2025) will not be a factor of 3 higher than Run2, as thought at the time FTK was proposed and designed. So, this did not make FTK a necessity. Given the complexity of the system and the number of experts needed to run it, the tracking wil be done in a larger CPU farm.

The final FTK paper:

The ATLAS collaboration (G. Aad et al.), "The ATLAS Fast TracKer system", 2021 JINST 16 P07006 (<u>DOI:10.1088/1748-0221/16/07/P07006</u>).

Some Refs: Content Addressable Memory, the Associative Memory & FPGAs

- K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," in IEEE Journal of Solid-State Circuits, vol.41, no.3, pp. 712-727, March 2006
- M. Dell'Orso and L. Ristori, "VLSI Structures Track Finding", Nucl. Instr. and Meth. A, vol. 278, pp. 436-440, 1989.
- W. Ashmanskas et al., "*The CDF online Silicon Vertex Tracker*", Nucl. Instr. and Meth. A, vol. 485, pp. 178-182, 2002.
- A. Annovi, et al., "Associative memory design for the Fast Track processor (FTK) at ATLAS," in IEEE NSS/MIC, 2009, Orlando, pp. 1866 – 1867.
- C.-L. Sotiropoulou, S. Gkaitatzis, A. Annovi, et al. "A Multi-Core FPGA-based 2D-Clustering Implementation for Real-Time Image Processing", in IEEE Trans. on Nuclear Science, vol. 61, no. 6, pp. 3599 - 3606, December 2014.

Some improvements on this method of tracking

- **Pattern matching:** Towards AM09: the 400k patterns per chip:
 - "The AM08 Associative Memory ASIC Design, Architecture and Evaluation methodology",
 - A. Vgenopoulos et al., to be published as procceedings in MOCAST2022
- Track Fitting: We saw that we can make the Track Fitting very fast with linearized equations (track parameters expressed as linear functions of hit positions). But, need small regions for the linear-equations-approximation to be true → many sets of equation parameters (one set per region).
 - Can make a coordinate transformation to an "idealized geometry" and there the detector regions where the linear approximation holds are much larger: much fewer regions → fewer equations → fewer sets of parameters → much smaller memory needs on the FPGAs:

"A High-performance Track Fitter for Use in Ultra-fast Electronics", E. Clementa et al., <u>arXiv:1809.01467</u>

The point to take home:

• Split the problem in a fast (coarse) one, and a refined one working with much reduced data.

(you know now that we do this all the time in the trigger)

 Use pre-calculated patterns & values wherever you can: if you get the desired precision, you gain a lot in time

...And time is precious in the online world!

- We saw the example of the Fast TracKer upgrade in ATLAS, using
 - AM-based pattern matching with "AM chip" (ASIC),
 - refined track-fitting and almost everything else needed (from formatting to smart databases, to I/O) in powerful modern FPGAs (recall Hannes Sakulin's & Manoel Barros Marin' talks)

C. Other examples

Other examples

- What was presented here is not the only way to solve the tracking problem fast. Other solutions exist, e.g:
 - <u>Hough transforms</u> in FPGAs,
 - Other algorithms in FPGAs (e.g., <u>Retina algorithm</u>: Luciano Ristori, NIM A 453 (2000) pp. 425-429)
 - GPUs for the HLT farms etc.... (→ You heard from Gianluca Lamanna on Friday 17/6/2022)
 - Machine Learning for TDAQ, particularly on FPGAs (→ You heard from Sioni Summers on Tuesday, 21/6/2022)
- But nothing can be as fast as doing the tracking while reading your data, as they pass through the system.
 - If you can not afford to be slower, then you'll probably use an Associative Memory.
 - For commercial solutions (e.g., CPUs, FPGAs, GPUs, etc). can overcome slower speed with high parallelism → it's all a matter of cost at the end...

Tracking at HEP in the future: AM+FPGAs, FPGAs, CPUs, and GPUs

- V. Halyo, et. al., "GPU Enhancement of the Trigger to Extend Physics Reach at the LHC," Journal of Instrumentation 8 P10005, 2013.
- C. Gentsos, F. Crescioli, P. Giannetti, D. Magalotti, S. Nikolaidis, *"Future evolution of the Fast TracKer (FTK) processing unit"*, PoS (TIPP2014) 209
- A. Annovi, et al., "Associative Memory for L1 Track Triggering in LHC Environment," in IEEE Trans. on Nuclear Science, Vol. 60, No. 5, pp. 3627 – 3632, 2013.
- G. Hall, et al., "A time-multiplexed track-trigger for the CMS HL-LHC upgrade", in NIM A, Vol.824, 11 July 2016, pp. 292–295
- A. Abba et al., "Simulation and performance of an artificial retina for 40 MHz track reconstruction", in JINST 10 C03008 (2015)

HighLuminosity-LHC (HL-LHC): pile-up of ~140-200 events/crossing will be typical; up to 200 events per crossing are considered likely. At L1: need tracking in <10 μ s

L1 Track Trigger at CMS : starts with a much cleaner picture than ATLAS (not just hits)

- ~99% of tracks have $P_T < 2$ GeV/c ; interesting things have higher P_T tracks.
- CMS makes the detector itself selective on such tracks by finding track "stubs" on closely spaced layers



• At B=4 Tesla, you have for each stub (straight line) of angle φ , at some double layer at a radius r, originating from a track generated with φ_0 and P_T



Can do "Hough Transform" for track finding, on FPGA

• This $\varphi(\mathbf{r})$ behaviour is a straight line: $\varphi = \mathbf{m} \mathbf{r} + \mathbf{c}$

→ So, could do a "Hough transform": map individual {r, φ } measurement points to a whole-line characteristic in 2D space: **the slope (m) & the intercept (c)**

→ Given an $\{r, \varphi\}$ pair, try values for m, get c: $c = -m r + \varphi$ and put $\{m, c\}$ in a 2-dimensional histogram



- {r,φ} measurements from same track will populate same {m,c} bin
- Most populated bin = characterises whole track
- Note: Small |m| values: $|m| = 0.006/P_T \rightarrow |m| < 0.003$

Can do "Hough Transform" for track finding, on FPGA

• This $\varphi(\mathbf{r})$ behaviour is a straight line: $\varphi = \mathbf{m} \mathbf{r} + \mathbf{c}$

→ So, could do a "Hough transform": map individual {r, φ } measurement points to a whole-line characteristic in 2D space: **the slope (m) & the intercept (c)**

→ Given an $\{r, \varphi\}$ pair, try values for m, get c: $\mathbf{c} = -\mathbf{m} \mathbf{r} + \boldsymbol{\varphi}$ and put $\{m, c\}$ in a 2-dimensional histogram



- {r,φ} measurements from same track will populate same {m,c} bin
- Most populated bin = characterises whole track
- Note: Small |m| values: $|m| = 0.006/P_T \rightarrow |m| < 0.003$

Pileup events: {m,c} array heavily populated and such peaks are not initially prominent.

But, by requiring e.g., all stubs in the (m,c) histogram bin to be from different radial layers, significantly reduces the background

Hough transform

- P. V. C. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3,069,654, 1962.
- R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures" Communications of the ACM, vol. 15, no. 1, pp. 11–15, 1972.
- J. Illingworth and J. Kittler, "The Adaptive Hough Transform", IEEE Trans. On Pattern Analysis and Machine Inteligence, Vol PAMI-9, No. 5, Sept. 1987, pp. 690-698.
- Xin Zhou, Yasuaki Ito, and Koji Nakano. "An FPGA Implementation of Hough Transform using DSP blocks and block RAMs." Bulletin of Networking, Computing, Systems, and Software, Vol 2, No 1 (2013), pages 18–24.

....etc....

 On FPGAs: important to adapt the algorithms to the constraints of FPGA operation. Algorithms can overflow the capacity of even a very large FPGA because of timing constraints or routing congestion → see from Manoel Barros Marin, on Monday

CMS L1 track approach is "local tracking" in both stages

- What we've seen so far is "global tracking": all hits available simultaneously (pattern matching and linear approximation wanted all hits present to work with the patterns and constants needed).
- "Local tracking" (~progressive tracking): add hits on the way
- Track finding at CMS:
 - stubs in adjacent layers form "tracklet seeds" \rightarrow growth of tracks by projection to next layers and χ^2 test for adding the stubs

• Track fitting at CMS:

 - "Kalmam filter" → project the helix parameters of the tracklet to each next layer, recalculate hit positions based on extrapolation and observed hits, recalculate and extrapolate helix parameters and so on.

E.g, see (and references therein): T. James, "Level-1 Track Finding with an all-FPGA system at CMS for the HL-LHC", arXiv:1910.12668 https://arxiv.org/abs/1910.12668

A. Hart, "Level 1 Track Finder at CMS" arXiv:1910.06614 https://arxiv.org/abs/1910.06614

Beyond High Energy Physics applications: E.g., image processing with pattern matching

- Hough transform is one of the classic techniques used in generic image processing to do first an "edge-detection"
- Let's see another interesting example on image perception:

M. Del Viva, G. Punzi, and D. Benedetti. *"Information and perception of meaningful patterns."* PloS one 8.7 (2013): e69154.

"... models describe the initial processing of visual information as the extraction of a simplified "sketch" based on a limited number of "salient features" [11], [12], that therefore contains a much reduced amount of information."

"We adopt the **principle of maximum entropy** as a measure of optimization: we ask what is choice of the **pattern set** producing the largest amount of entropy allowed by the given limitations of the system. We will see that this simple requirement, together with the imposed strict limitations to the computing resources of the system, allows to completely determine the choice of the pattern set from the knowledge of the statistical properties of the input data."

Image processing, an interesting example with patterns: M. Del Viva, G. Punzi, and D. Benedetti. Fig. 1



Constrain #1: storage N = number of patterns Constraint #2: output bandwidth W = reduction factor (e.g., $W=0.001 \rightarrow 1/1000$ can be selected with this pattern set)

p = probability that the given
pattern matches the
(sub)image we check

$$f(p) = \frac{-p \log(p)}{\max(1/N, p/W)}$$

unit cost for each pattern

Image processing, an interesting example with patterns: M. Del Viva, G. Punzi, and D. Benedetti, Fig. 3 and 4









ISOTDAQ2022, Catania, 22/6/2022

In a 3x3 grid:

- 512 possible patterns

Green:

- the "best" 50 of them

(use them in the images below)

Blue:

the "best" 15 of them

K. Kordas - Pattern Recognition w/ Associative Memories & FPGAs 70

Image processing, an interesting example with patterns: M. Del Viva, G. Punzi, and D. Benedetti. Fig. 2

Our tracking detectors also produce "images" (= the set of hits), and we select events based on them



Of course, we know that all these zig-zag lines are meaningless Training on simulated events, to get the patterns with max. entropy, picks up the patterns we also select when we do simulations to define the pattern bank.

Summary

- Show that need fast tracking information at the Trigger of High Energy Physics experiments
- We split the problem into "track finding" (define fast a "road" where a track can be) and "track fitting" (determine the track characteristics)
- Show in some detail the ATLAS (FTK and HTT) case, using
 - Track finding with Pattern matching in Associative Memories , and Track fitting in FPGAs
- Basically we saw that: if we want to avoid or cannot afford calculating something time consuming, we can split the problem and use pre-calculated patterns and quantities.
- We saw also examples of other approaches, with both steps done in FPGAs (e.g., CMS L1 track finder)
- We saw an example of patterns used in image processing



Acknowedgments: Alberto Annovi, Francesco Crescioli, Mauro Dell' Orso, Paola Giannetti, Andrea Negri and FTK members
Extras...



FTK - Fast Tracker for Hadron Colliders

An FP7 IAPP project (February 1, 2103 - January 31, 2017)





http://ftk-iapp.physics.auth.gr/

This project aims to develop an extremely fast but compact processor, with supercomputer performances, for pattern recognition, data reduction, and information extraction in high quality image processing.

The proposed hardware prototype features flexibility for potential applications in a wide range of fields, from triggering in high energy physics to simulating human brain functions in experimental psychology or to automating diagnosis by imaging in medical physics. In general, any artificial intelligence process based on massive pattern recognition could largely profit from our device, provided data are suitably prepared and formatted.

The project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement n.324318 Participants (2 SMEs and 4 Academic Institutions)





Extras

- D. Emeliyanov, et al., "GPU-based tracking algorithms for the ATLAS high-level trigger" in Journal of Phys. Conf., Ser. 396, 012018, 2012.
- J. Mattmann, et al., "Track finding in ATLAS using GPUs," in Journal of Phys. Conf., Ser. 396, 022035, 2012.
- Y. Ago, Y. Ito, and K. Nakano, "An FPGA implementation for neural networks with the FDFM processor core approach," International Journal of Parallel, Emergent and Distributed Systems, vol. 28, no. 4, pp. 308–320, 2012.



* Red: involvement of the group



FTK Schedule

All boards for the full detector coverage are available at 2018



2009



FTK Latency

FTK has enough processing power at L=3x1034cm-2s-1 (operating rate ~60%)

Latency was rise-up by heavy event, but after such an event the latency quickly return to the typical range.



Averagely latency is ~50 μ sec and maximum on tail is ~ few handed μ sec. It is enough speed for HLT requirement.

FTK Track performance

All results are base line of FTK performance!



Difference is :

- Algorism of hit clustering
- Lack of Low Pt patterns
- Broken of linear approximation.
- No TRT, not δray correction, etc

More than 90 % efficiency with respect to offline.

Beyond FTK

- FTK was using the AMchip06, an Associative Memory with
 - 128k patterns of 8 words × 18 bits each word
 - high speed serial links
 - variable resolution (up to 6 ternary bits)
 - low power
 - 8 \times 16 bit comparisons at 100 MHz
- The ATLAS Hardware Track Trigger (HTT) for the HL-LHC era, was to use an AM chip (AMchip09) with many more patterns (~400k patterns/chip).
 - Applications outside HEP (medical imaging, smart cameras, genomics, ...)