Where are the low hanging fruits?

A TALE BY DHCP

Image © Venables, Guy

One tough day at work…

```
const float det = (z1*z1)*z2 + z1*(z3*z3) + (z2*z2)*z3 - z2*(z3*z3) - z1*(z2*z2) - z3*(z1*z1);
const float det1 = (x1)*z2 + z1*(x3) + (x2)*z3 - z2*(x3) - z1*(x2) - z3*(x1);
const float det2 = (z1*z1)*x2 + x1*(z3*z3) + (z2*z2)*x3 - x2*(z3*z3) - x1*(z2*z2)  - x3*(z1*z1);
const float det3 = (z1*z1)*z2*x3 + z1*(z3*z3)*x2 + (z2*z2)*z3*x1 - z2*(z3*z3)*x1 - z1*(z2*z2)*x3 - z3*(z1*z1)*x2;
```

Ta-da!

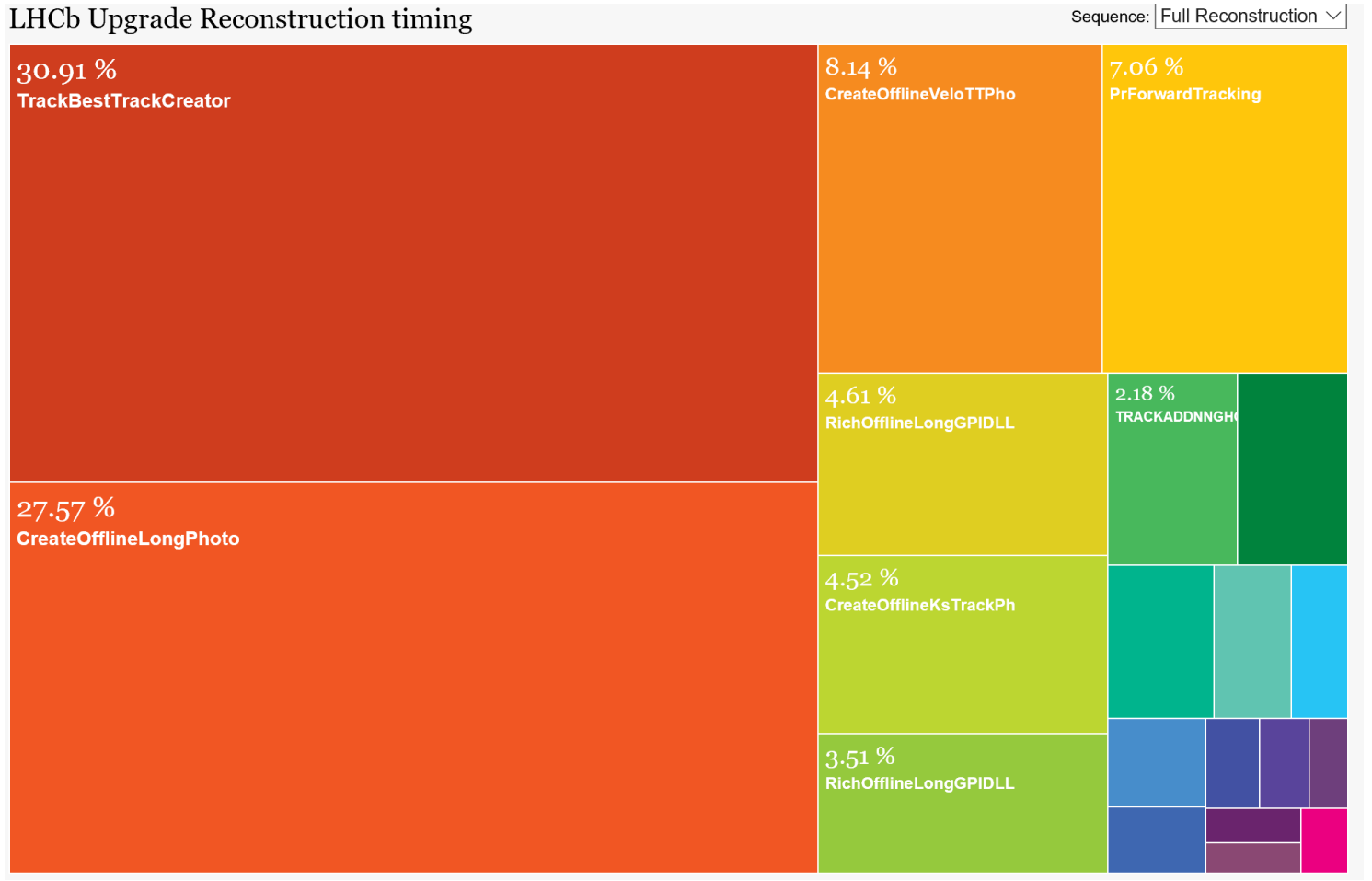
```
// const float det = (z1*z1)*z2 + z1*(z3*z3) + (z2*z2)*z3 - z2*(z3*z3) - z1*(z2*z2) - z3*(z1*z1);
// const float det1 = (x1)*z2 + z1*(x3) + (x2)*z3 - z2*(x3) - z1*(x2) - z3*(x1);
// const float det2 = (z1*z1)*x2 + x1*(z3*z3) + (z2*z2)*x3 - x2*(z3*z3) - x1*(z2*z2)  - x3*(z1*z1);
// const float det3 = (z1*z1)*z2*x3 + z1*(z3*z3)*x2 + (z2*z2)*z3*x1 - z2*(z3*z3)*x1 - z1*(z2*z2)*x3 - z3*(z1*z1)*x2;

const float det = ((z1*z1)*(z2-z3))+ ((z2*z2)*(z3-z1)) + ((z3*z3)*(z1-z2));
const float det1 = (x1)*(z2-z3) + (x2)*(z3-z1) + (x3)*(z1-z2);
const float det2 = (z1*z1)*(x2-x3) + (z2*z2)*(x3-x1) + (z3*z3)*(x1-x2);
const float det3 = (z1*z1)*(z2*x3 -z3*x2) + (z2*z2)*(z3*x1 -z1*x3) + (z3*z3)*(z1*x2 -z2*x1);
```

Another job well done.

```
// const float det = (z1*z1)*z2 + z1*(z3*z3) + (z2*z2)*z3 - z2*(z3*z3) - z1*(z2*z2) - z3*(z1*z1);
// const float det1 = (x1)*z2 + z1*(x3) + (x2)*z3 - z2*(x3) - z1*(x2) - z3*(x1);
// const float det2 = (z1*z1)*x2 + x1*(z3*z3) + (z2*z2)*x3 - x2*(z3*z3) - x1*(z2*z2)  - x3*(z1*z1);
// const float det3 = (z1*z1)*z2*x3 + z1*(z3*z3)*x2 + (z2*z2)*z3*x1 - z2*(z3*z3)*x1 - z1*(z2*z2)*x3 - z3*(z1*z1)*x2;

const float det = ((z1*z1)*(z2-z3))+ ((z2*z2)*(z3-z1)) + ((z3*z3)*(z1-z2));
const float det1 = (x1)*(z2-z3) + (x2)*(z3-z1) + (x3)*(z1-z2);
const float det2 = (z1*z1)*(x2-x3) + (z2*z2)*(x3-x1) + (z3*z3)*(x1-x2);
const float det3 = (z1*z1)*(z2*x3 -z3*x2) + (z2*z2)*(z3*x1 -z1*x3) + (z3*z3)*(z1*x2 -z2*x1);
```

TrackFit solveParabola
gcc 4.8 reports a 1.18x speedup
Intel Xeon E5-2670 v2 @ 2.50GHz

Target: Optimise [insert large codebase]



Things are not so simple in real-*er* life

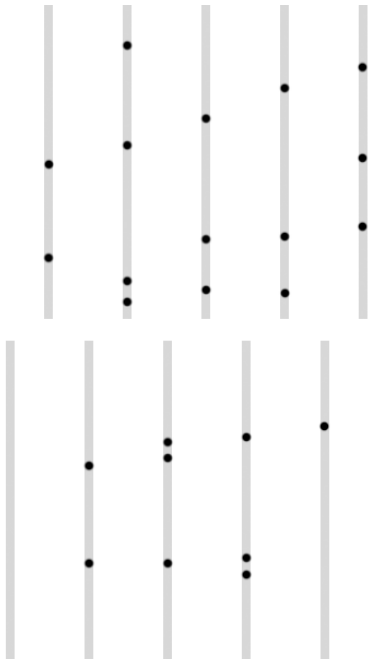
- Think of data
  - Locality, cache usage
  - Data format – AoS vs SoA – Hiding under templates
  - Multi-threading support
  - Data alignment – Cache line splits
  - Coalesced data accesses

- Think of instruction flow
  - Locality
  - Virtual tables
  - Nested calls / Function calls
  - Dependency graphs

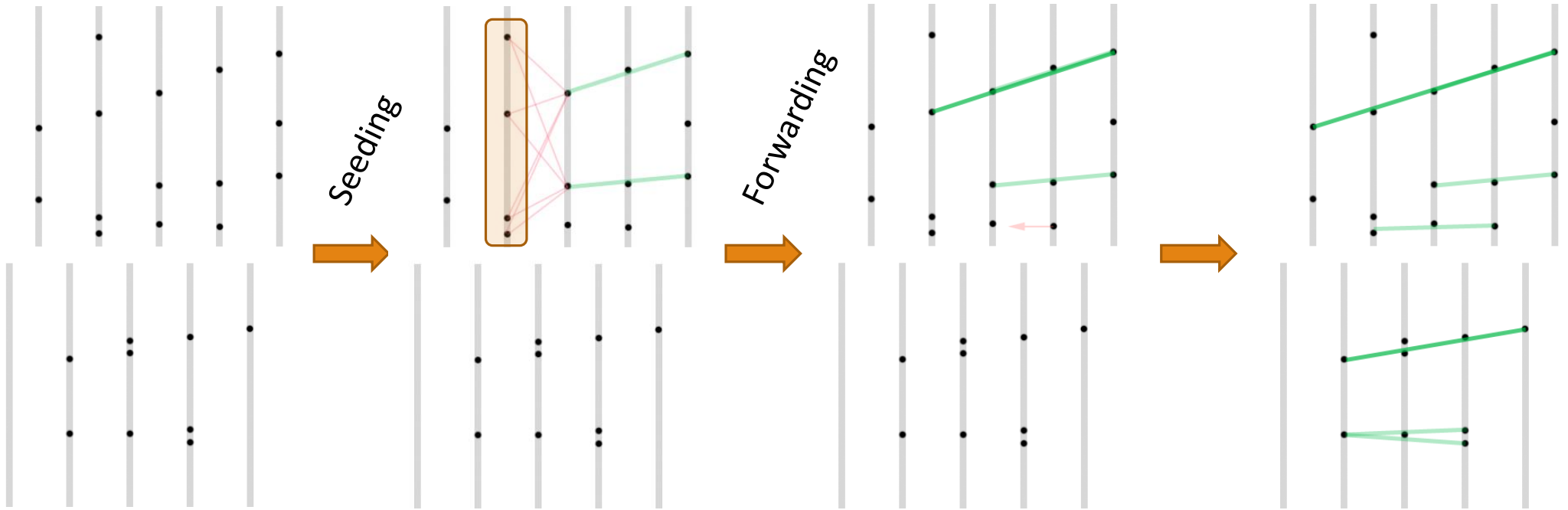
Is that all?

- Think of data
  - Locality, cache usage
  - Data format – AoS vs SoA – Hiding under templates
  - Multi-threading support
  - Data alignment – Cache line splits
  - Coalesced data accesses
- Think of instruction flow
  - Locality
  - Virtual tables
  - Nested calls / Function calls
  - Dependency graphs
- Hardware constraints
  - Memory hierarchy
  - SIMD / SIMT
  - Core count
  - Vector register count
  - NUMA nodes
- Good design
  - Maintainability
  - Flexibility
  - Readability
  - Usability
  - Integration

Hm. Yet another tracking instance.



Data access patterns solved the day



Focusing on one thing at a time

```
void fitAndSelect(
    TrackData& track) const;
```

Focusing on ~~one~~ *many things* at a time

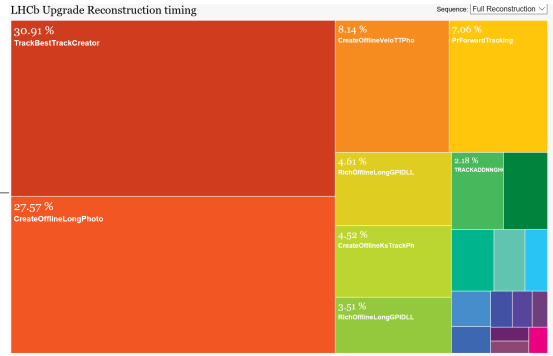
```
// void fitAndSelect(
//     TrackData& track) const;

void fitAndSelect(
  const std::vector<std::reference_wrapper<TrackData>>::iterator& alltracksBegin,
  const std::vector<std::reference_wrapper<TrackData>>::iterator& alltracksEnd) const;
```

- This small change implied O(1000) line changes in the code (*and growing*)
  - Things are not always easy to test!
  - But oh when they work – Roughly 10% speedup, opens up avenues for further parallelization

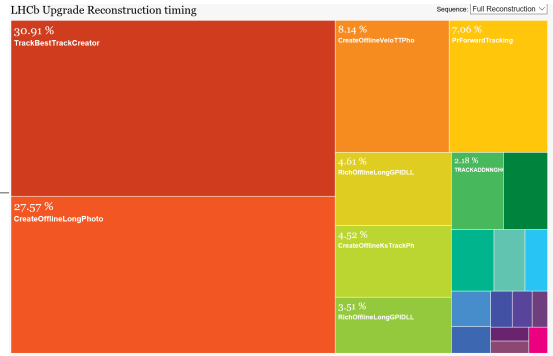
- **Choose your fights**
  - Amdahl's law is a good indicator
  - Profiling – absolutely essential!

I have so much time

LHCb Computing Upgrade – 4 MCHF



I have so much time



LHCb Computing Upgrade – 4 MCHF

Whole reco:

- HLT1 50%
- HLT2 50%

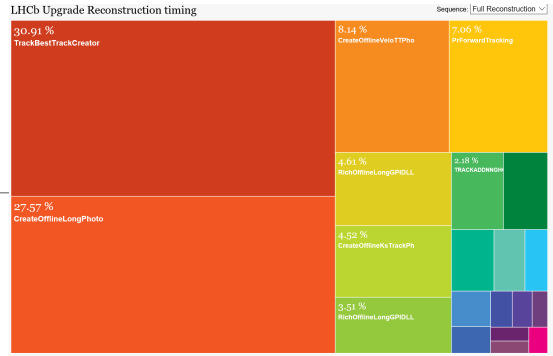
HLT 1:

- Kalman Filter 60%
- Tracking 38%
- TES data 2%

Kalman Filter:

- Predict Filter 60%
- Material corr 20%
- Other stuff 20%

I have so much time



LHCb Computing Upgrade – 4 MCHF

Whole reco:

- HLT1 50%
- HLT2 50%

HLT 1:

- Kalman Filter 60%
- Tracking 38%
- TES data 2%

Kalman Filter:

- Predict Filter 60%
- Material corr 20%
- Other stuff 20%

- A 5% improvement in TES manipulation means saving 2kCHF
- A 5% improvement in Predict Filter means saving **45kCHF** !

How much time is it worth spending on profiling wrt coding?

- **90% +**

Where are the low hanging fruits?

Low hanging fruits may be closer than you think!