Proposal for PyCBC restructuring: - Gergely Debreczeni

Batch execution

The goal is to **saturate** the compute cores. Segmenth length will not increase much in the following ~10 years, but global memory and number of compute cores will **do**. Processing series of segments (batches) in one go **does better** and batch size can be changed any time. An iterator is not good, segments should be memory-continous arrays to reach coalesced reading. Changes necessary:

- The datavectorXXX should describe a batch not a segment.
- Thus segmenting should be a bit different
- New meta_data elements are necessary
  - meta_data.number_of_segments - how many segment is contained in the datavactor
  - meta_data.length_of_segment - size of one segment (number of elements)

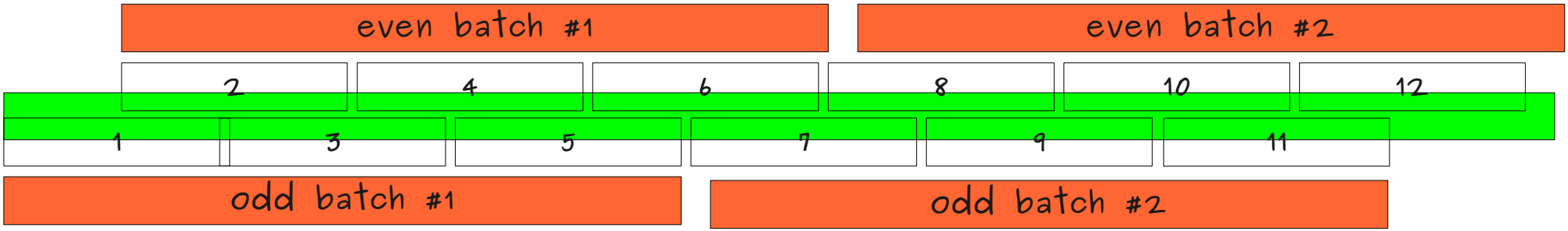
Limiations:

- MAX_MEMORY_ALLOC_SIZE limits the buffer size on the global memory (typically one batch could contain ~20 segment

Hot loop:

- The hot loop should loop on batches and not on segments. The 'real hot loop' should be inside the MatchedFilterXXX object. There implementation can decide whether or not use batch execution.

Segmenting



- Odd segments are forming a batch
- Even segments are forming a batch
- Instead of using an iterator through the X segment, the batches should be processed. If there are too much segment – more than one batch – then iterator should be looped on the batches.
 (FFT, iFFT, and scalar product with the template, can be be performed with one kernel call in batch mode, and calculates various quantities for the whole batch right away.)

- Current FFT size (128 sec) is just on the limit, for the next generation of GPUs this will be very useful !