CO405H

Computing in Space with OpenSPL

Topic 10: Correlation Example

Oskar Mencer **Georgi Gaydadjiev**

Department of Computing
Imperial College London

http://www.doc.ic.ac.uk/~oskar/
http://www.doc.ic.ac.uk/~georgig/

CO405H course page: http://cc.doc.ic.ac.uk/openspl14/
WebIDE: http://openspl.doc.ic.ac.uk
OpenSPL consortium page: http://www.openspl.org

o.mencer@imperial.ac.uk g.gaydadjiev@imperial.ac.uk

Recap: Spatializing Loops

- Classifying Loops
  - Attributes and measures
- Simple Fixed Length Stream Loops
  - Example vector add
  - Custom memory controllers
- Nested Loops
  - Counter chains
  - Streaming and unrolling
  - How to avoid cyclic graphs
- Variable Length Loops
  - Convert to fixed length
- Loops with data dependencies
  - DFESeqLoop
  - DFEParLoop

This Lecture: Correlation Example

Let's create a spatial implementation of correlation:

$$r_{xy} = \frac{\sum^{n} x_i y_i - n\bar{x}\bar{y}}{(n-1)s_x s_y} = \frac{n\sum^{n} x_i y_i - \sum^{n} x_i \sum^{n} y_i}{\sqrt{n\sum^{n} x_i^2 - (\sum^{n} x_i)^2}\ \sqrt{n\sum^{n} y_i^2 - (\sum^{n} y_i)^2}}.$$

src: http://en.wikipedia.org/wiki/Correlation_and_dependence

r is the correlation coefficient for a pair of data timeseries *x,y* over a window *n*

Our example computes correlation for *N*=200-6000 data timeseries and returns the top 10 correlations at any point in time.

DFE Input: Interleaved stream of N timeseries, plus precalculations from the CPU

Original CPU Version for Correlation

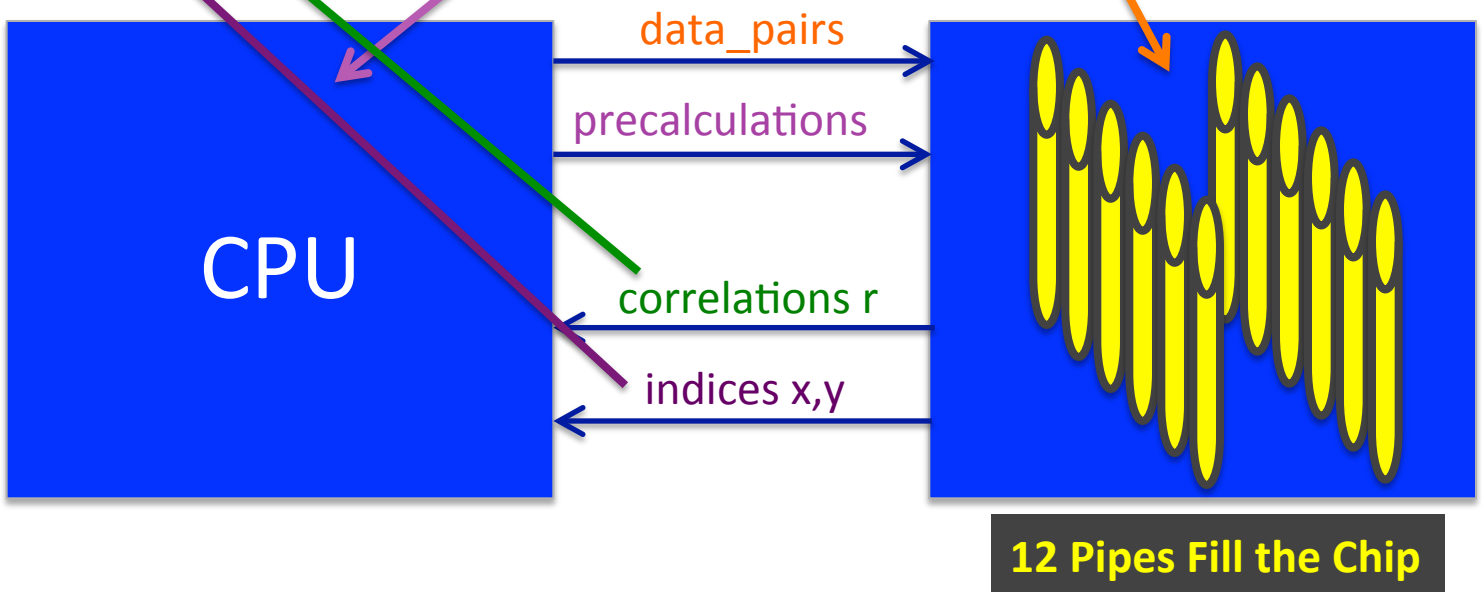
```
for (uint64_t s=0; s<numTimesteps; s++) {
    index_correlation = 0;
    for (uint64_t i=0; i<numTimeseries; i++) {
        double old = (s>=windowSize ? data[i][s-windowSize] : 0);
        double new = data [i][s];
        sums[i] += new - old;
        sums_sq[i] += new*new - old*old; // start and end of window => let's call them data pairs
    }
    for (uint64_t i=0; i<numTimeseries; i++) {
        double old_x = (s>=windowSize ? data[i][s-windowSize] : 0);
        double new_x = data [i][s];
        for (uint64_t j=i+1; j<numTimeseries; j++) {
            double old_y = (s>=windowSize ? data[j][s-windowSize] : 0);
            double new_y = data [j][s];

            sums_xy[index_correlation] += new_x*new_y - old_x*old_y;
            correlations_step[index_correlation] = (windowSize*sums_xy[index_correlation]-sums[i]*sums[j])/
                                                   (sqrt(windowSize*sums_sq[i]sums[i]*sums[i])*
                                                   sqrt(windowSize*sums_sq[j]-sums[j]*sums[j]));
            indices_step[2*index_correlation] = j;
            indices_step[2*index_correlation+1] = i;
            index_correlation++;
        }
    }
```

Move to DFE LMEM?

Precompute?

Split Correlation into CPU and DFE

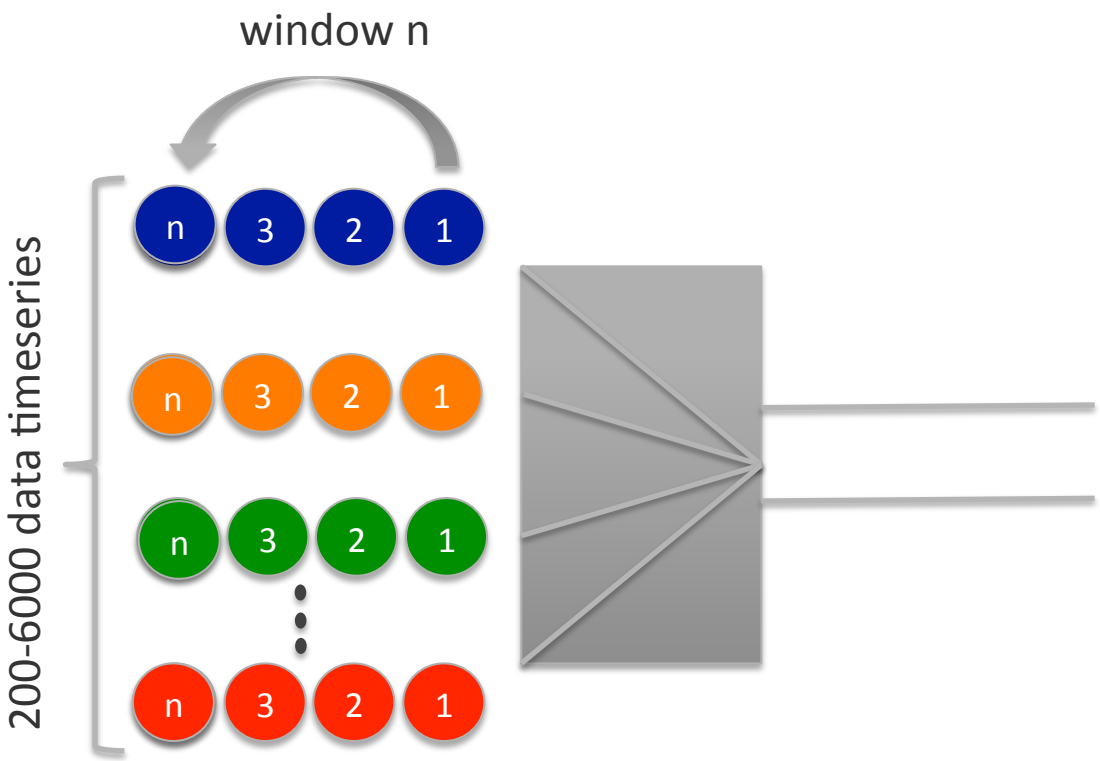
$$r_{xy} = \frac{\sum^{W} x_i y_i - n\bar{x}\bar{y}}{(n-1)s_x s_y} = \frac{n\sum^{W} x_i y_i - \sum^{W} x_i \sum^{W} y_i}{\sqrt{n\sum^{W} x_i^2 - (\sum^{W} x_i)^2}\sqrt{n\sum^{W} y_i^2 - (\sum^{W} y_i)^2}}.$$



each pipe has a feedback loop of 12 ticks => 144 calculations running at the same time

prepare_data_for_dfe() – The Movie

reordering for data_pairs

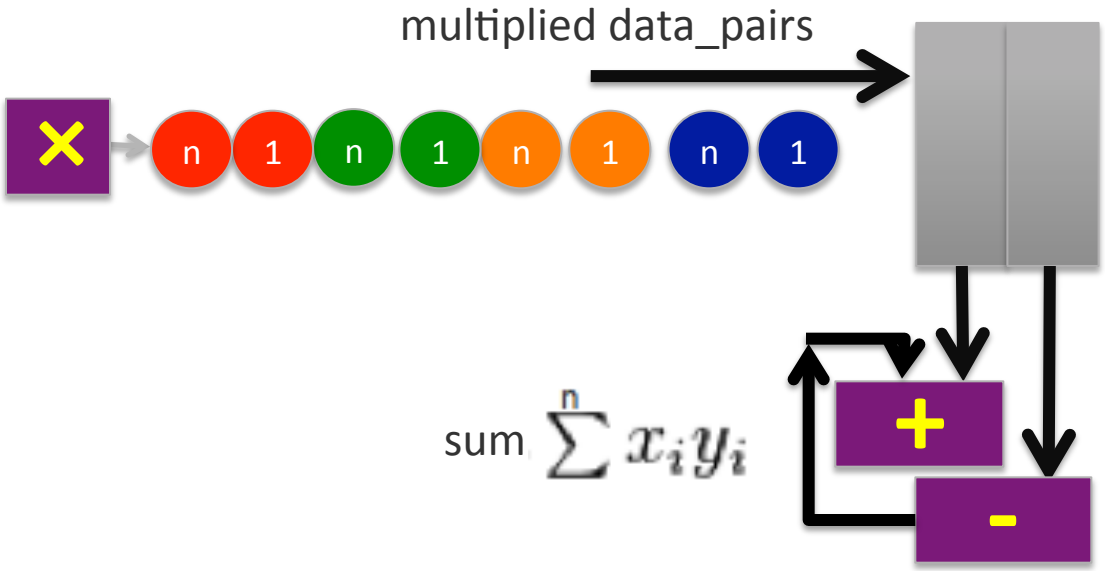


first and last element of each window creates a data_pair, then go round robin…

Example: Sum of a Moving Window

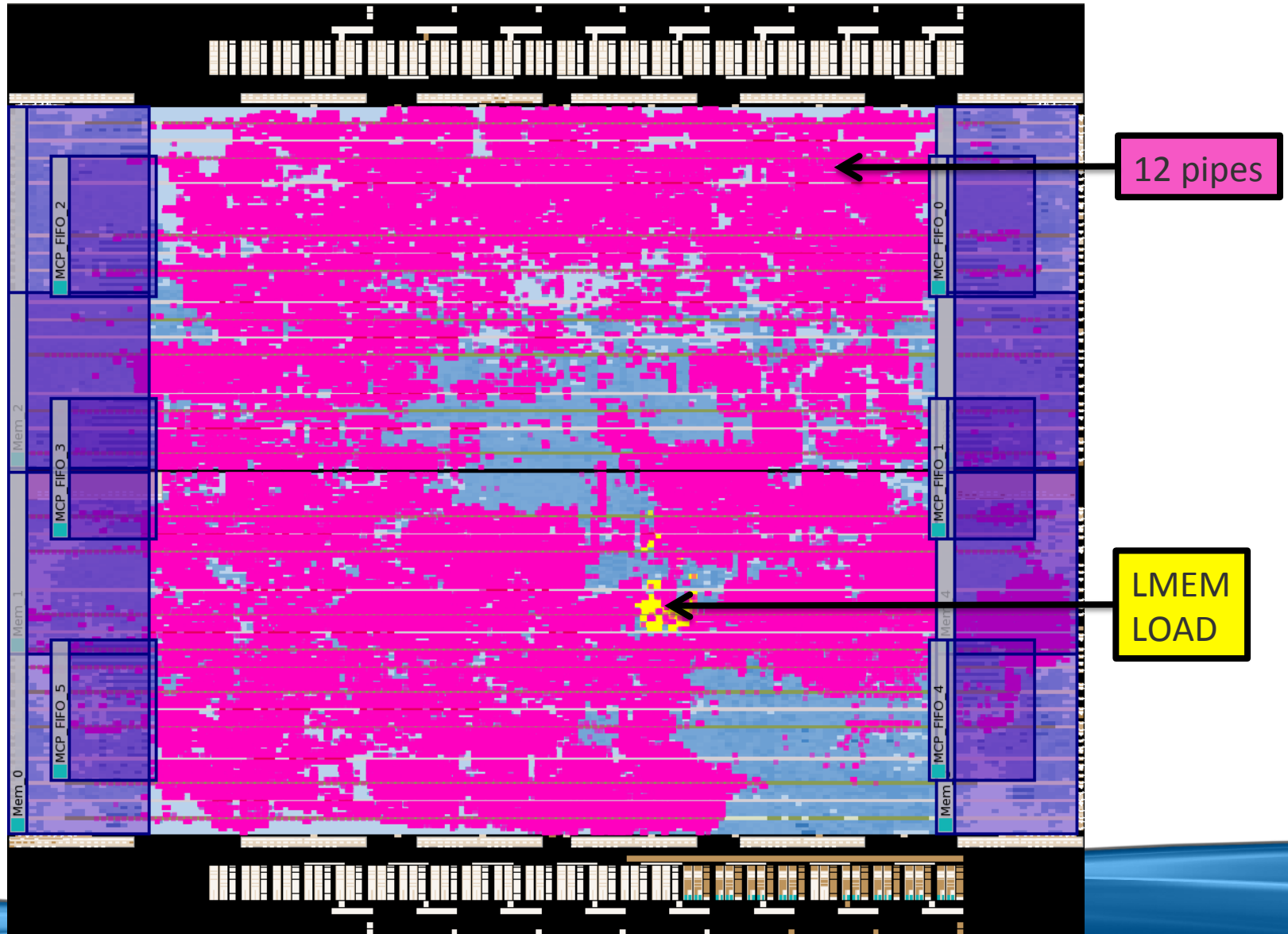
keep the window sum by subtracting the top and adding the new entrant

Note: this is a Case 2 Loop (Lecture 9), with a sequential streaming loop (par_loop)

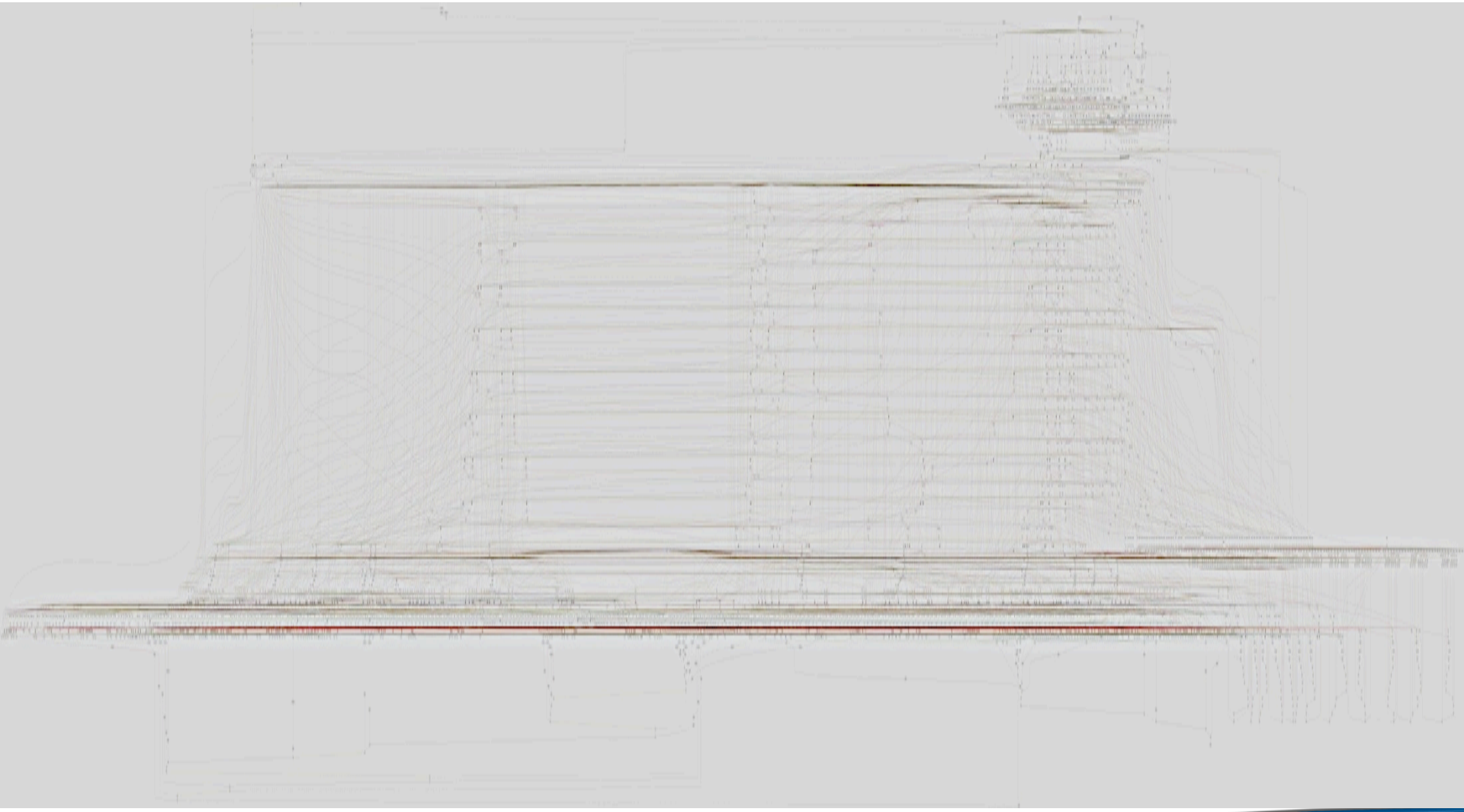



```
DFEParLoop lp = new DFEParLoop(this,“lp”);
DFEVector<DFEVar> data_pairs = io.input(”dp", dfeVector(…), lp.ndone);
lp.set_input(sum.getType(), 0.0);
   DFEVar sum = (lp.feedback + data_pairs[0]) - data_pairs[1];
lp.set_output(sum);
io.output(”sum", sum, sum.getType(), lp.done);
```

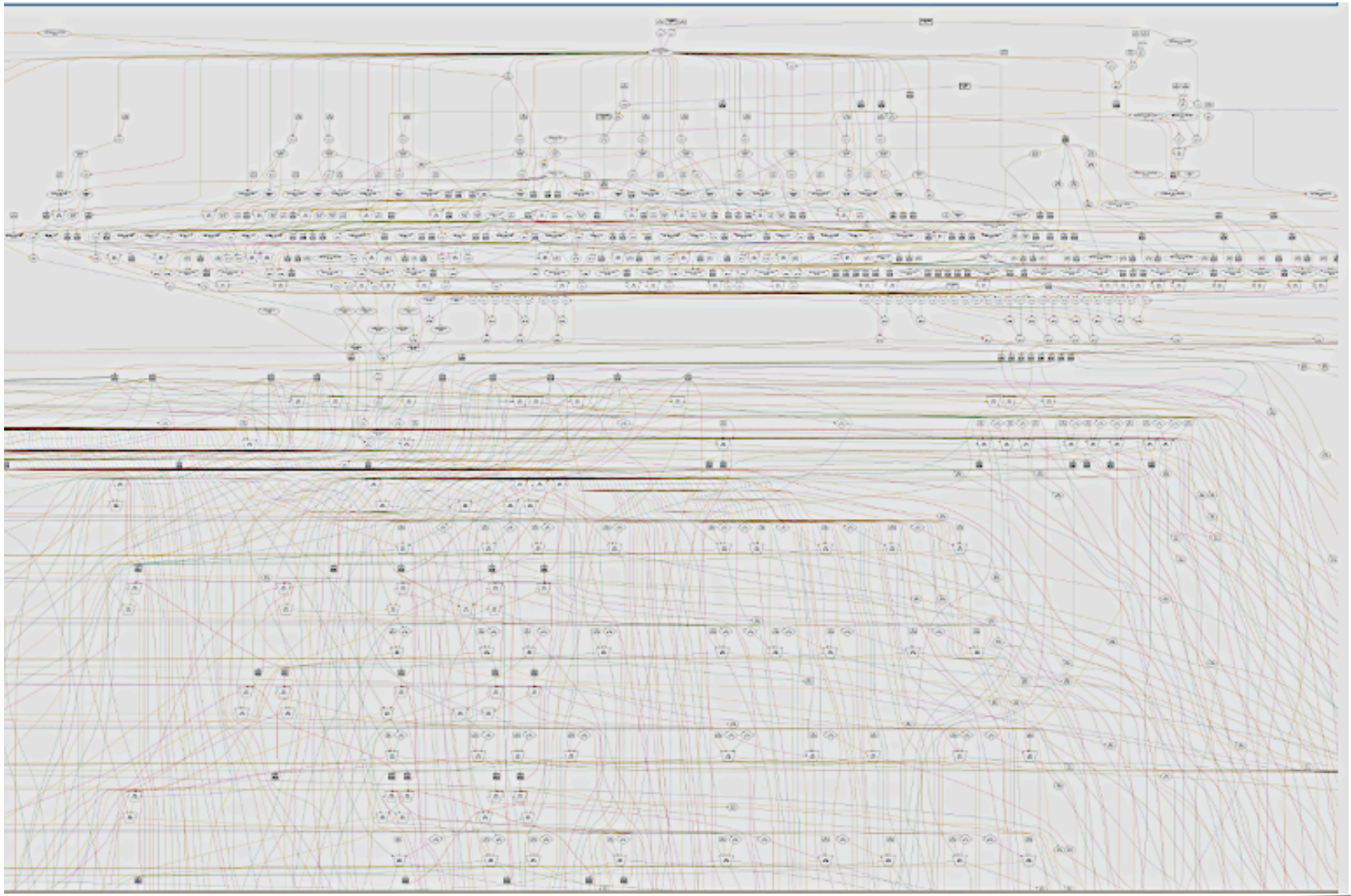
Correlation .max file plotted in Space



Correlation Pipes: Dataflow Graph



Correlation Dataflow Graph: Zoom In



File: correlationCPUCode.c
Purpose: calling correlationSAPI.h for correlation.max

Correlation formula:

scalar r(x,y) = (n*SUM(x,y) - SUM(x)*SUM(y))*SQRT_INVERSE(x)*SQRT_INVERSE(y)
where:
x,y, ... - Time series data to be correlated
n - window for correlation (minimum size of 2)
SUM(x) - sum of all elements inside a window
SQRT_INVERSE(x) - 1/sqrt(n*SUM(x^2)- (SUM(x)^2))

Action 'loadLMem':
[in] memLoad - initializeLMem, used as temporary storage

Action 'default':

[in] precalculations: {SUM(x), SQRT_INVERSE(x)} for all timeseries for every timestep

[in] data_pair: {..., x[i], x[i-n], y[i], y[i-n], ... , x[i+1], x[i-n+1], y[i+1], y[i-n+1], ...} for all timeseries for every timestep

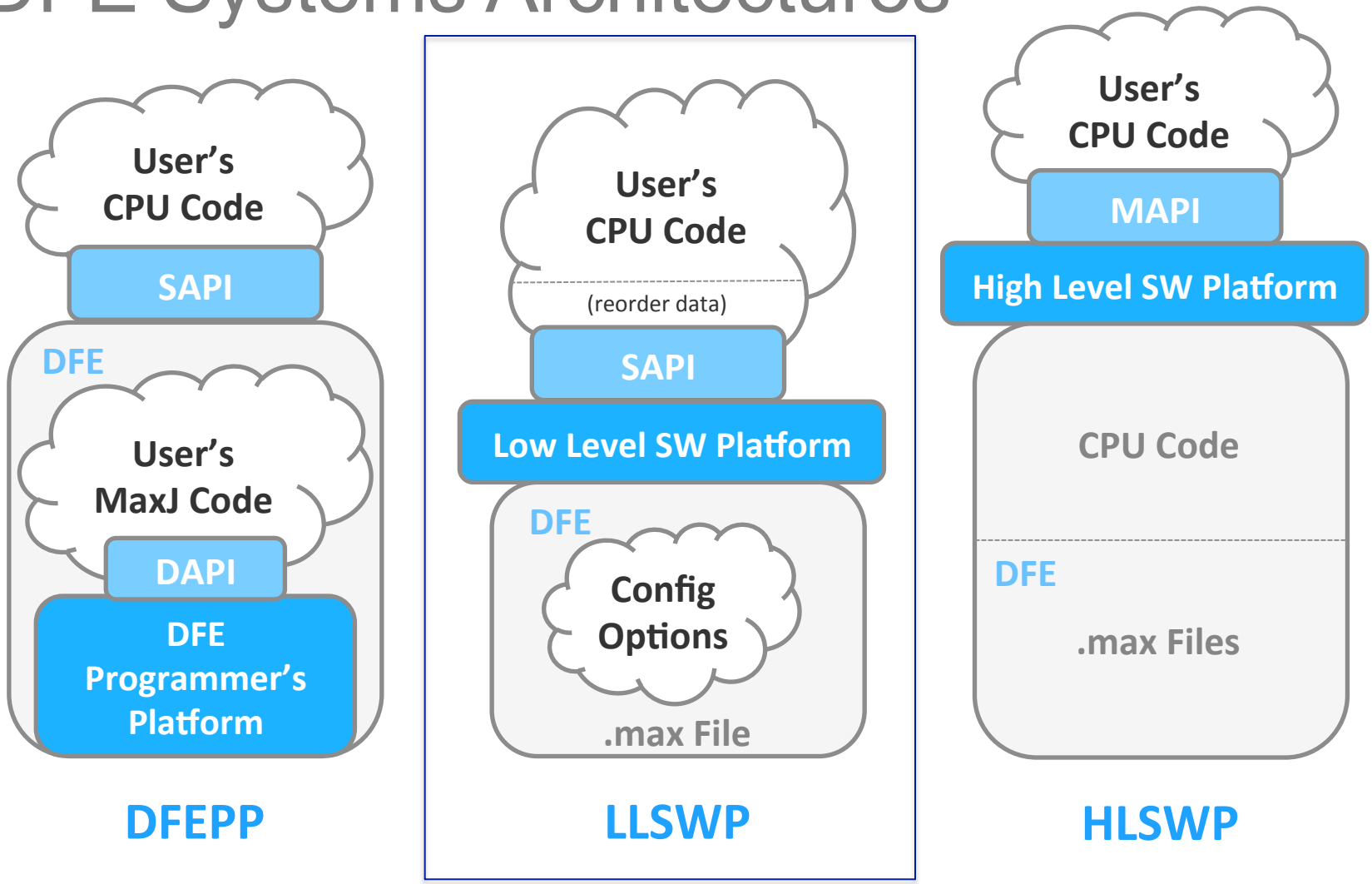
[out] correlation r: numPipes * CorrelationKernel_loopLength * topScores correlations for every timestep

[out] indices x,y: pair of timeseries indices for each result r in the correlation stream

prepare_data_for_dfe() – The Code

```
// 2 DFE input streams: precalculations and data pairs
for (uint64_t i=0; i<numTimesteps; i++) {

        old_index = i - (uint64_t)windowSize;

        for (uint64_t j=0; j<numTimeseries; j++) {
                if (old_index<0) old = 0; else old = data [j][old_index];

                new = data [j][i];

                if (i==0) {
                        sums [i][j] = new;
                        sums_sq [i][j] = new*new;
                }else {
                        sums [i][j] = sums [i-1][j] + new - old;
                        sums_sq [i][j] = sums_sq [i-1][j] + new*new - old*old;
                }

                inv [i][j] = 1/sqrt((uint64_t)windowSize*sums_sq[i][j] - sums[i][j]*sums[i][j]);

                // precalculations REORDERED in DFE ORDER
                precalculations [2*i*numTimeseries + 2*j] = sums[i][j];
                precalculations [2*i*numTimeseries + 2*j + 1] = inv [i][j];

                // data_pairs REORDERED in DFE ORDER
                data_pairs[2*i*numTimeseries + 2*j] = new;
                data_pairs[2*i*numTimeseries + 2*j + 1] = old;
        }
}
```

DFE Systems Architectures



EXAMPLES

Max MPT
Video transcoding/processing

Correlation App

Risk Analytics Library
Video Encoding

Correlation Lab Exercise (due Dec 1, 2014)

- Open correlation project at openspl.doc.ic.ac.uk/correlation/static/index.html

- Three directories:
  1. **ORIGINAL:** original software
  2. **SPLIT_CONTROL_DATA:** software split into dataflow and controlflow
  3. **DFE:** CPU code plus DFE SAPI interface plus correlation.max file for simulation and for running on DFE.

Task 1: run software versions (1+2) and report all timings for correlating 256 time series for 10, 100, and 1,000 time steps. Why are the runtimes different.

Task 2: Then run 256, 512, 1024, 2046 and 4096 time series for 10, 100 and 1,000 time steps on a real MAX4(maia) DFE. If any combination does not work, run the closest possible combination. Collect output of the programs into a spreadsheet report and compare speed per correlations of the various versions.

Task 3: run CPU+DFE code with 256 time series in simulation for 10 time steps and report the time. How much slower is the simulation compared to running on the DFE? How much slower is it than the software versions from task 1?

Task 4 (optional): Change input timeseries from purely random to series 100 and series 200 being perfectly correlated (same random numbers shifted by 10) and run the DFE version on 256 timeseries again and show output with 100/200 ontop.