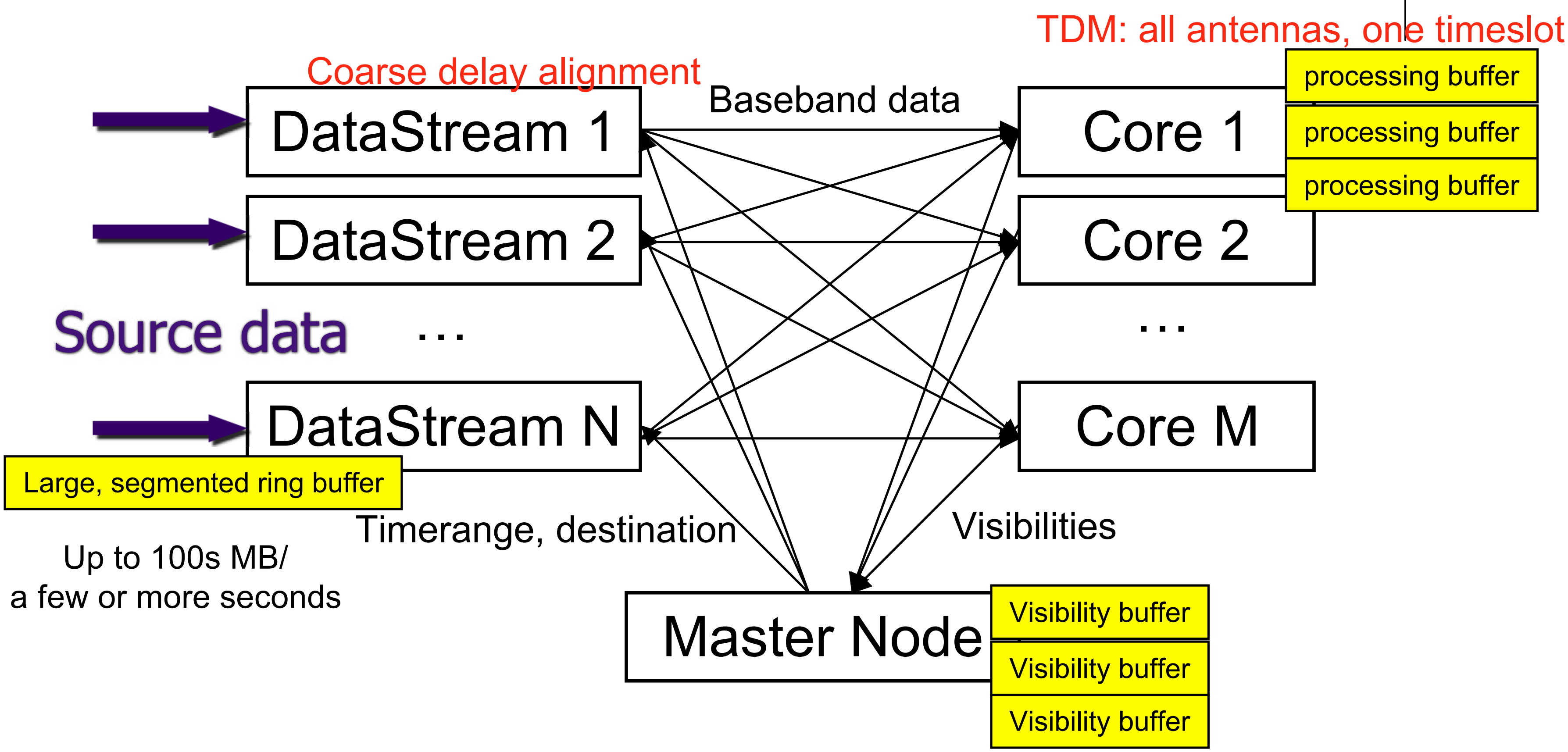
The DiFX software correlator: capabilities and performance

Adam Deller, NRAO/UC Berkeley

What is DiFX?

- A Distributed FX software correlator
- Written in C++, designed to run on x86 CPUs
- >95% of computation spent in vector performance library - currently Intel Performance Primitives is used
- Data distribution performed using MPI (double buffered asynchronous transfers)
- Fast, flexible and widely adopted in the VLBI community (LBA, VLBA, MPIfR, Haystack)

Mpifxcorr architecture



TDM: all antennas, one timeslot

Coarse delay alignment

Baseband data

DataStream 1

DataStream 2

Source data

…

DataStream N

Large, segmented ring buffer

Up to 100s MB/
a few or more seconds

Core 1

processing buffer

processing buffer

processing buffer

Core 2

…

Core M

Timerange, destination

Visibilities

Master Node

Visibility buffer

Visibility buffer

Visibility buffer

MPI is used for inter-process communications

Each data transfer is double buffered

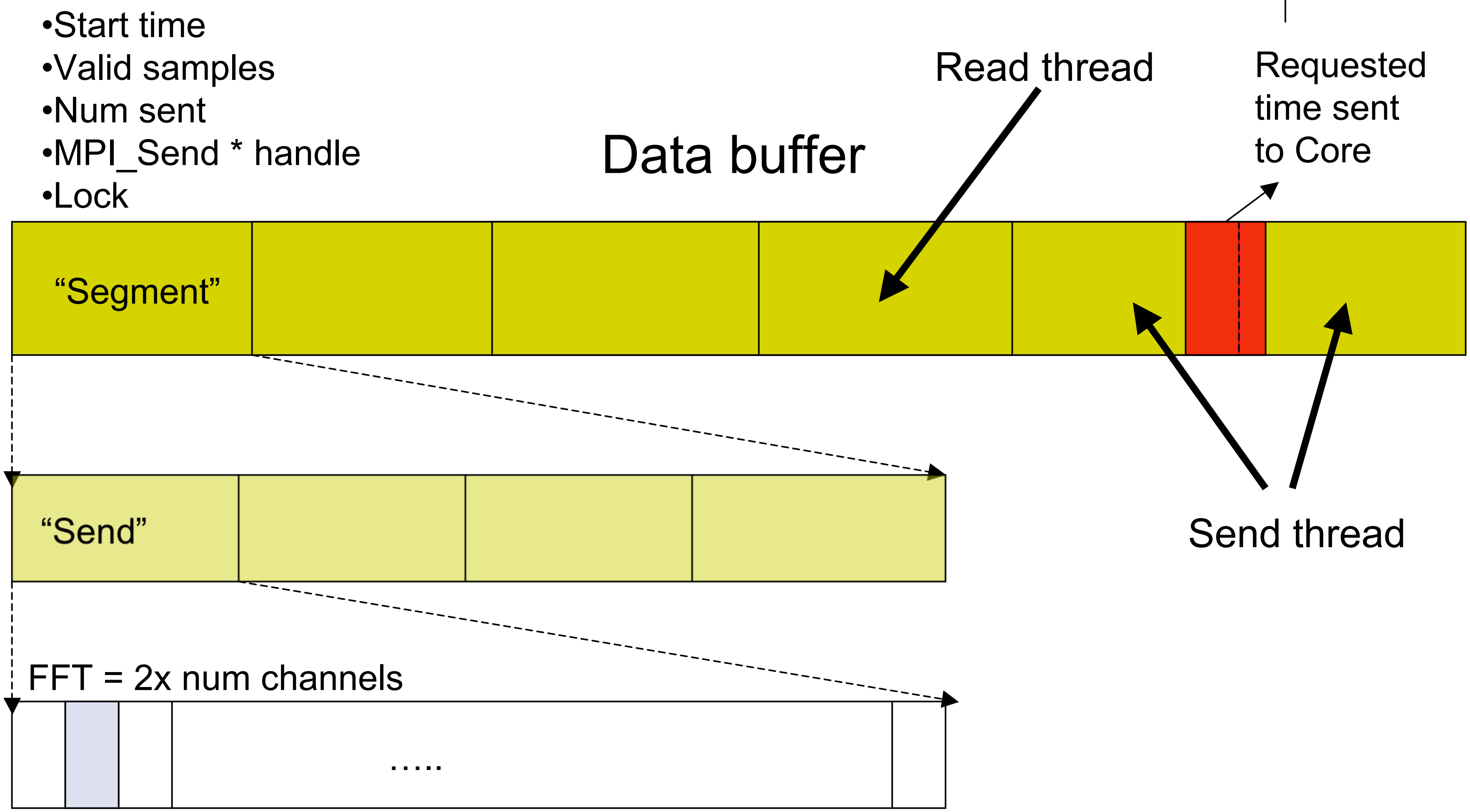
FxManager correlation flow

- Start at the requested time, step one block of FFTs at a time until end of correlation
- After the initial filling of buffers, sit in a loop receiving subintegrations, adding the to visibilities and sending commands off for fresh data to be sent to processing nodes
- As visibilities are completed, release lock on visibility buffer slot (second thread writes out)

Datastream correlation flow

- Two threads: Main (receives instructions, sends data) and read (fills the buffer)
- Each maintains a lock on at least one segment of the databuffer at all times
- While data remains, the read thread will keep populating the data buffer until told to stop
- Main thread just dumbly fulfils requests until told to stop by Manager
- Sends a short flag to Core if no valid data

Datastream correlation flow



- Start time
- Valid samples
- Num sent
- MPI_Send * handle
- Lock

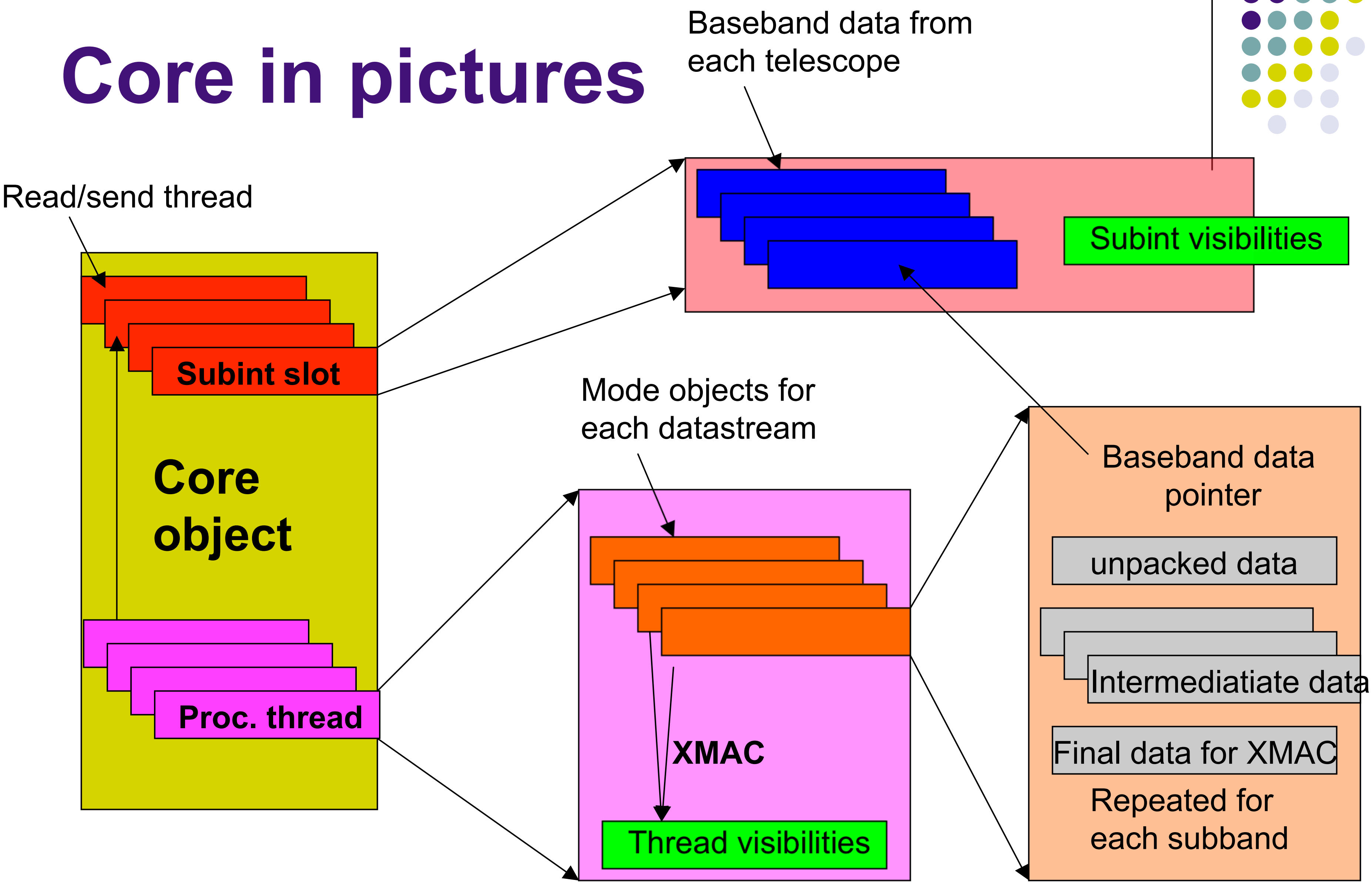
Core correlation flow

- N+1 threads: 1 for send/receive, the rest to do actual correlation
- One buffer slot is processed at a time - each process thread gets 1/Nth of the FFTs
- More locking is required so the threads can aggregate their results, which are stored in one long array (for ease of sending back)
- Keeps looping until a terminate message is received from FxManager

Under the hood in Core

- Each thread is identical, and has an array of “Mode” objects, which handle the station-based processing for each Datastream
- Mode knows how to unpack the different formats, and then handles fringe rotation, FFT and fractional sample correction
- After telling each Mode to do its thing, the thread grabs the appropriate results and XMACs

Core in pictures



What do you get out?

- A dumb binary format is written by DiFX
- Essentially arbitrary time/frequency resolution, start pay the cost in performance as you push away from <100ms, <100 kHz
- Translated post-correlation to FITS-IDI (AIPS)
- Translation to other formats like Mark4/HOPS (geodesy)
- Fast filterbank dump (separate over UDP)

DiFX advantages

- C++ is (reasonably) comprehensible and its easy to add new features
- Intel updates IPP all the time to take best advantage of new processors - someone else is doing the optimization
- Essentially the same architecture has been used since the beginning of the project and it still seems very efficient
- 32-bit throughout (delay calculations 64 bit) so no artifacts

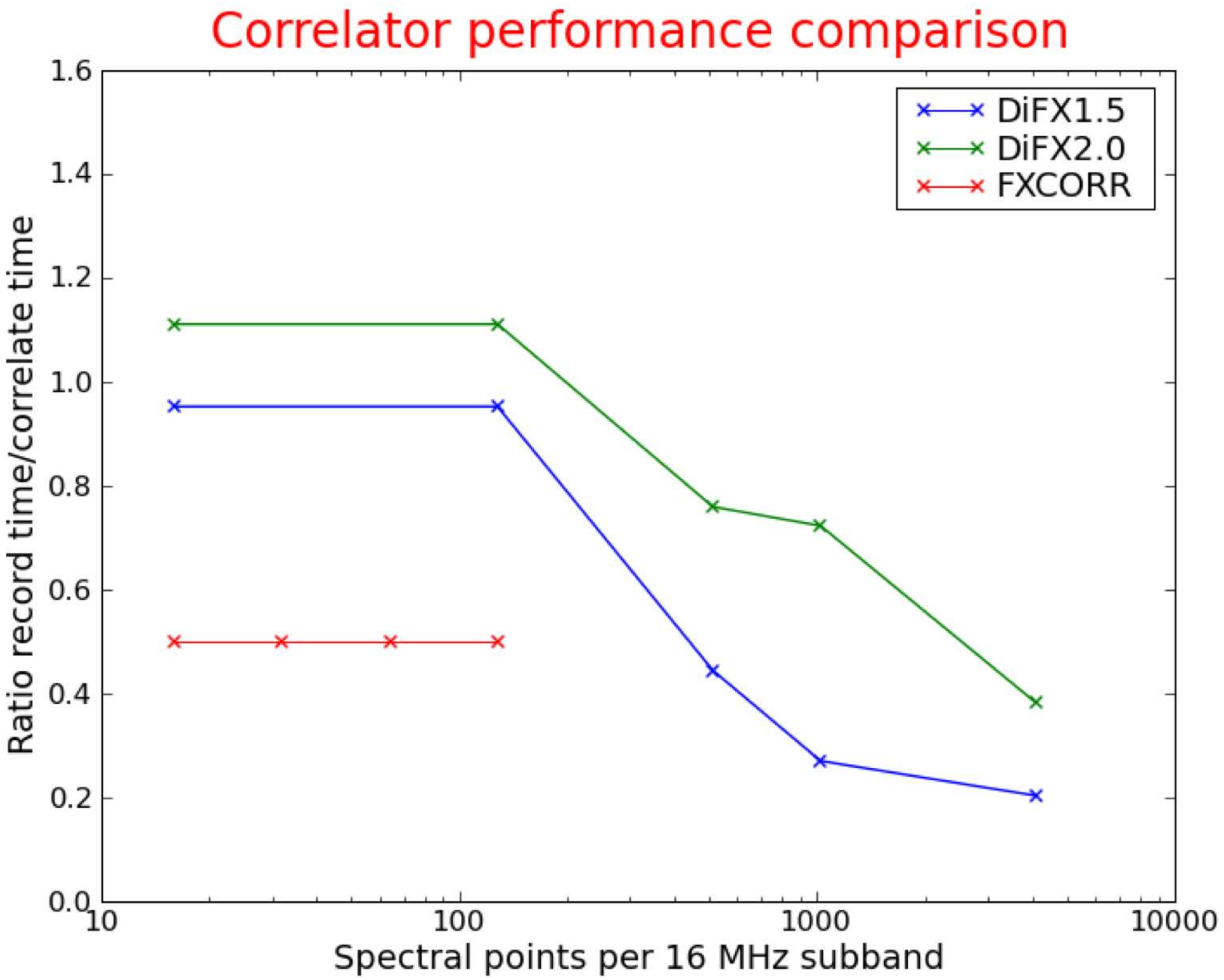
DiFX features

- Reads a large number of input formats (and its easy to add more)
- Does phase calibration tone extraction
- Produce multiple phase centers in a single correlator pass (uv shift inside the correlator)
- Can correlate mis-matched bands (e.g. 2x16 with 1x32, USB with LSB)
- High time res. filterbank dump, transients
- Many more; see: **http://cira.ivec.org/dokuwiki/doku.php/difx/start**

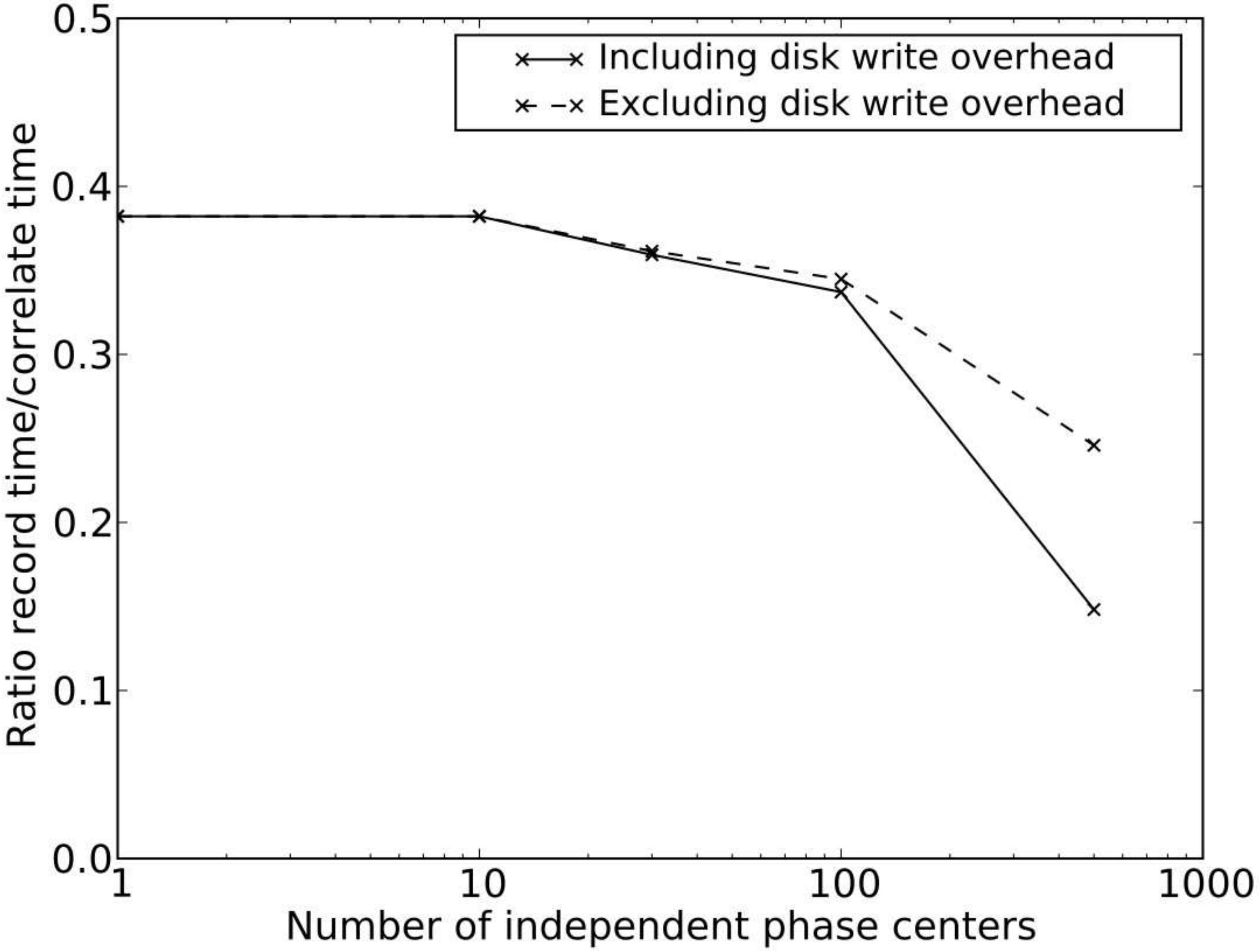
Performance

- Convenient benchmark is the VLBA cluster: 10x dual quad core Intel 5420s @ 2.5 GHz, cost around $20k in 2008
- Nowadays 2, maybe 3 dual 6 cores would do
- On this cluster DiFX sustains 512 Mbps (128 MHz bandwidth single pol, 64 MHz dual pol) from 10 stations
- This is the small-N regime where station based costs dominate, baseline based becomes equal at roughly 20 antennas

Performance



Performance



Conclusions

- DiFX is a flexible and mature software correlator
- It already has lots of nice features, but its easy to add in more
- Performance is good - will never be a match for GPUs, FPGAs or ASICs but can be installed/used much quicker and is more transferable
- Check out the wiki and sign up for the mailing list if you’re interested