**OSKAR Station Simulator:**

**Parallelism**

This document describes the approaches to parallel code design for the simulator. Taking module by module (as described in Figure 1) and considering the approximate operational cost and data rates the approach and degree of parallelisation is considered.

Sky

Antennas

Beamformer

Weights

Figure 1: Top level station processing modules and data paths for the station simulator. Module numbers correspond to section numbers in this document.

# Introductory notes

## Synchronous (end-to-end) operation?

Reading check pointed data will limit the input bandwidth of various sections of the simulator to that of the rate of disk I/O. As disk I/O read rates are lower than what is required by some sections of the simulator (see data interface document), the simulator will have to operate as a end-to-end simulation (i.e. from sky to beams) for some modes of operation.

## Data rates and simulator time

The simulated channeliser data rate in the station simulator will only be a small fraction of the data rate for an actual SKA station. This is due to limitations of processing and bandwidth on the platform the simulator will be run.

It is likely that the simulator will run with wall time channeliser data rate in the range 1-100Hz (although this will depend largely on the mode of operation. As a real SKA station is envisaged to run at a post channeliser data rate of around 106Hz the simulator can be considered to be run with time slowed down.

For example one second real time would correspond to simulator seconds; which for a simulator channeliser rate of 100Hz results in a time slow down of 104 and 1second of wall time corresponding to 10-4 seconds of simulator time.

## Consideration of modes of operation

Some modes of operation will be considerably more computationally or data intensive than others. As following through the details of each of these possibilities requires a huge undertaking this document attempts only to indicate the scaling for what might be a representative case.

# Sky processing

## Operations and data consideration

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Description | Processing | Data rate |
| Convert celestial sky to local sky | Perform a coordinate transform from the celestial frame to the local frame each time step for each source.  The coordinate transform takes the form of a rotation about an axis between the station phase canter and the position of the north celestial pole on the local sky. | 155 floating point operations per source per time step.  Total operation count of approx. 1.5Mflops | Output data: approx. 2.4MB/s.  (100 sources, 10channels, 2Polariations, 100Hz rate) |
| (Addition of source noise) | Add randomly generated amplitude and phase per source per time sample to each frequency channel and polarization |  | data rate unchanged |
| (Apply directional corruptions) | Evaluate the direction dependent corruptions and accumulate into the source map. |  | potentially very expensive (depends on form of corruption) |

Transform of celestial coordinates to local coordinates

The observer’s latitude is *φ* and the source declination is *δ*. Given a sidereal time *T,* observer’s longitude *λ* and source right ascension *α*, the hour angle of the source *H* is found using:

Then compute *X1* and *Y1* to find the azimuth, *A:*

Then compute *X2* and *Y2* to find the elevation, *e*:

## Possible operational shortcuts to be evaluated

* Reuse of trigonometric values.
* Performing the coordinate transform from celestial to local once and then applying a simple rotation.
* Remove the rotation of the earth. The celestial to local coordinate conversion can then be performed just once as the sky source distribution will be constant before corruptions/noise is added.

## Parallelisation

It is unlikely that this module will have to be parallelised in the first instance of the simulator. For the simple case of point-like constant sources, earth rotation and no corruptions both the data rates and computation required is extremely modest.

The addition of noisy sources increases the computational cost moderately but not the data rate and in this regime it is still unlikely to require parallelisation.

With the inclusion of direction dependent corruptions, both the data rate and computation are in general greatly increased, with a large dependence on the exact nature of the corruptions being applied (note the data format for corruptions allows almost complete flexibly in what this might be).

It is worth noting at this point that the sky processing scales linearly with the number of sources in the sky. For the purposed of determining the cost of this module we have assumed a sky of 100 sources however if this were to be increased considerably parallelisation of the sky processing by sources would be the natural approach as in this dimension the sky is trivially parallel.

# Antenna processing

## Operations

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Description | Processing | Data rate |
| Generate antenna signal for each time step | Evaluation of the complex amplitude at each antenna from which results from the source distribution above it. | 89 floating point operations per antenna per time step per source, per channel. | Output data rate: 1.1GB/s.  (65,536 antennas, 100Hz rate, 10channels, 2 polariations) |
| (Antenna noise) | Addition of both antenna element noise (from the electrical components) as well as evaluation of the total galactic noise. | will depend heavily on the implementation which still needs to be established | Data rate unchanged. |
| (Antenna corruptions) | Evaluation of the antenna signal due to various corruption effects. |  | Data rate unchanged. |
| (Antenna element pattern) |  |  | Data rate unchanged. |

Generate antenna signal for each time step.

This involves evaluating the complex amplitude at each antenna from the source distribution above it for that time step. The complex antenna signal takes the form

* A(source) is taken directly from the sky data and therefore takes no operations to evaluate in this module.
* phase(source) is calculated using the trigonometric relation which depends angular source position in the local sky for the particular time step.

Therefore for each antenna for each time step, generating the antenna signal requires evaluating the phase contribution from each source and evaluating the complex amplitude for the combination of sources.

The number of operations therefore scales as

for 65,536 antennas, 100 sources and a rate of 100Hz

## Possible operational shortcuts

* Remove sky rotation! The antenna signal is then constant over time steps prior to adding in corruptions and noise contributions.
* More approximate signal generation methods.

## Parallelisation

Antenna signal generation is both extremely expensive with respect to computation and data rate.

As a result in all cases antenna signal generation will have to be parallelised even for the simplest of cases. Fortunately the computation for antenna signal processing can easily be split by a number of dimensions each of which are trivially parallel; groups of antennas, channels, polarisations.

Antenna processing will at the first level of parallelism be split into logical tiles (antenna patches) which correspond closely to the logical tiles used in the first level of beamforming. A second level of parallelism can be made available where processing is also split along frequency channel although this is not likely to be required for the simulator.

Ultimately antenna processing will be limited by the output bandwidth available however depending on the consideration of noise and corruptions the computations rate will also limit what can be achieved in this module.

The choice antenna patch as the first level of parallelisation results from the understanding that the number of antennas is likely to be the largest parameter and also the least likely to change dramatically in various modes of operation.

# Beamformer processing

## Operations

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Description | Processing | Data rate |
| 1) Multiply weight matrix by detector element matrix and accumulate | simple matrix vector | depends on the number of levels of computation, number of beams ... | output data rate is reduced by the number of beams produced at each level. |
| 2) FFT |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Parallelisation

Less expensive than antenna signal generation!

The first level of parallelisation is likely to follow the antenna patches used for antenna signal processing.

Subsequent levels will require reordering and routing of the output beams.

# Weights processing

## Operations

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Description | Processing | Data rate |
| Generate a matrix of DFT weights |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Parallelisation