Accelerating Automated Extraction of Radio Astronomical Sources from Overseation Data with GPU Accelerators , University of Cape Town

YASEEN HAMDULAY, JARRED DE BEER

1. PROJECT DESCRIPTION

Source finding is the detection of galaxies from blind neutral hydrogen surveys of the sky.

Duchamp and Sofia are source finding packages that implement various source finding algorithms.

1.1. The problem

MeerKAT and ASKAP are future radio interferometers that are going to survey the sky. The expected dataset is many orders of magnitude larger than any other dataset we currently have. We expect our current source finders to take hours to days to run on this data. This is too slow.

1.2. Why is it important?

waiting days for source catalogues from source finders is unacceptable. This is expensive and wastes the time of the scientists running the experiments. It is a huge bottleneck and should be cut down.

1.3. Issues

current implementations are too slow

1.4. difficulties

gpu's are difficult to code

2. PROBLEM STATEMENT

2.1. Aim

our aim is to accelerate the process of source finding to the point where it is feasible to run on meerkat datasets.

2.2. Research question

can we accelerate source finding using gpu's?

2.3. Work division

Yaseen will implement 2d-1d wavelet reconstruction on the GPU in Duchamp.

Jarred will implement S+C on the GPU in Sofia.

3. PROCEDURES AND METHODS

3.1. Implementation strategy

(1) We will reimplement a single threaded version of our algorithm in C/C++.
(2) Implement naive GPU version of algorithm in CUDA.
(3) Compare two implementations output for correctness
(4) Attempt to optimise naive GPU version (using different memory hierarchy strategies)

3.2. Expected challenges

(1) Understanding the source finding algorithm from the source code written by the astronomers.
(2) Confirming correcness of source finding implementation.
(3) GPU's are difficult to code
(4) data dependencies making dividing (and merging) data into subcubes difficult
(5) memory bandwidth between GPU and CPU is limited and source finders are memory bandwidth limited (minimising memory congestion).
(6) maximising GPU core occupancy.
(7) Getting access to GPU cluster in order to test our implementations.
(8) Getting original Sofia and Duchamp creators to integrate our new accelerated algorithms back into sofia and duchamp.

3.3. Evaluation

— We can run the reference implementation (the original found in Duchamp and SoFiA) on a datacube. Then we run our implementatin on the same data cube and compare the results. They should be exactly the same.

4. ETHICAL, PROFESSIONAL AND LEGAL ISSUES

SoFiA is GPL version 3.0
DUCHAMP is GPL version 2.0
We will need to check if re-integrating our code into these repositories will have legal issues with the NRF who is funding our work.

5. RELATED WORK

— A comparison of Source finding algorithms. [**?**]
— Parallel Gaussian source finder in OpenCL and CUDA. [**?**]
— K-Means implementation on the GPU. [**?**]
— NVidia paper on convolution. [**?**]

6. ANTICIPATED OUTCOMES

6.1. Expected impact

— We anticipate the GPU implementation to be considerably faster than the single-threaded implementation.
— Faster turn-around time for astronomers when source finding is required.

6.2. Key success factors

— Algorithm speed-up.
— Being merged into the original source finding packages (Sofia and duchamp respectively),
— Used in production by astronomers.

7. PROJECT PLAN

7.1. risks

Risk	P	Mitigation
Data format from semi-realistic data cubes is incompatible with SoFiA/DUCHAMP packages.	7	Convert it to FITS
Unable to get access to HIPASS data cubes	6	Email Paalo Serra and ask for his datacubes
Algorithm is memory bound to the point where it is unable to be sped up over the CPU.	6	Choose another algorithm
Integrating GPU code with Sofia and Duchamp is infeasible requiring us to reimplement key aspects of the source finder framework	6	Rewrite input, merging sources etc
GPU programming is difficult and we may not have sufficient experience to optimise the GPU algorithm to the point of achieveing a speedup.	6	Consult Chris Laidler, look at Nvidia papers on similar topics, e.g. their convolution paper ¡cite here¿
The part of the pipeline we choose to implement turns out not to be the bottleneck.	6	Profile the execution of the single threaded version before we start implementing it
A team member drops out of the course.	6	Design the implementation strategy such that each member has a similar procedure, but the code, framework and implementation are entirely independent, and can be tested, compiled, integrated individually. Yaseen has chosen DUCHAMP package, Jarred has chosen SoFiA package.
Processing time takes many hours to complete and power outages become a problem.	8	Gain access to UCT's cluster and run it on that
GPU Kernels take too long to process and are interupted by the Watchdog timer.	6	Break the cubes up into smaller subcubes, or write smaller CUDA kernels

7.2. work allocation diagram

7.3. gantt chart

7.4. resources required

— Generated semi-realistic data cubes from astronomer.
— Access to the GPU cluster.
— HIPASS Parkes All Sky Survey data cubes as well as the catalogues for testing on real data and comparing reliability.
— Duchamp and Sofia code.
— NVidia graphics cards for local testing before sending to the cluster.

7.5. deliverables

— Final product with the source finding algorithms integrated into the respective frameworks. Buildable and runnable.

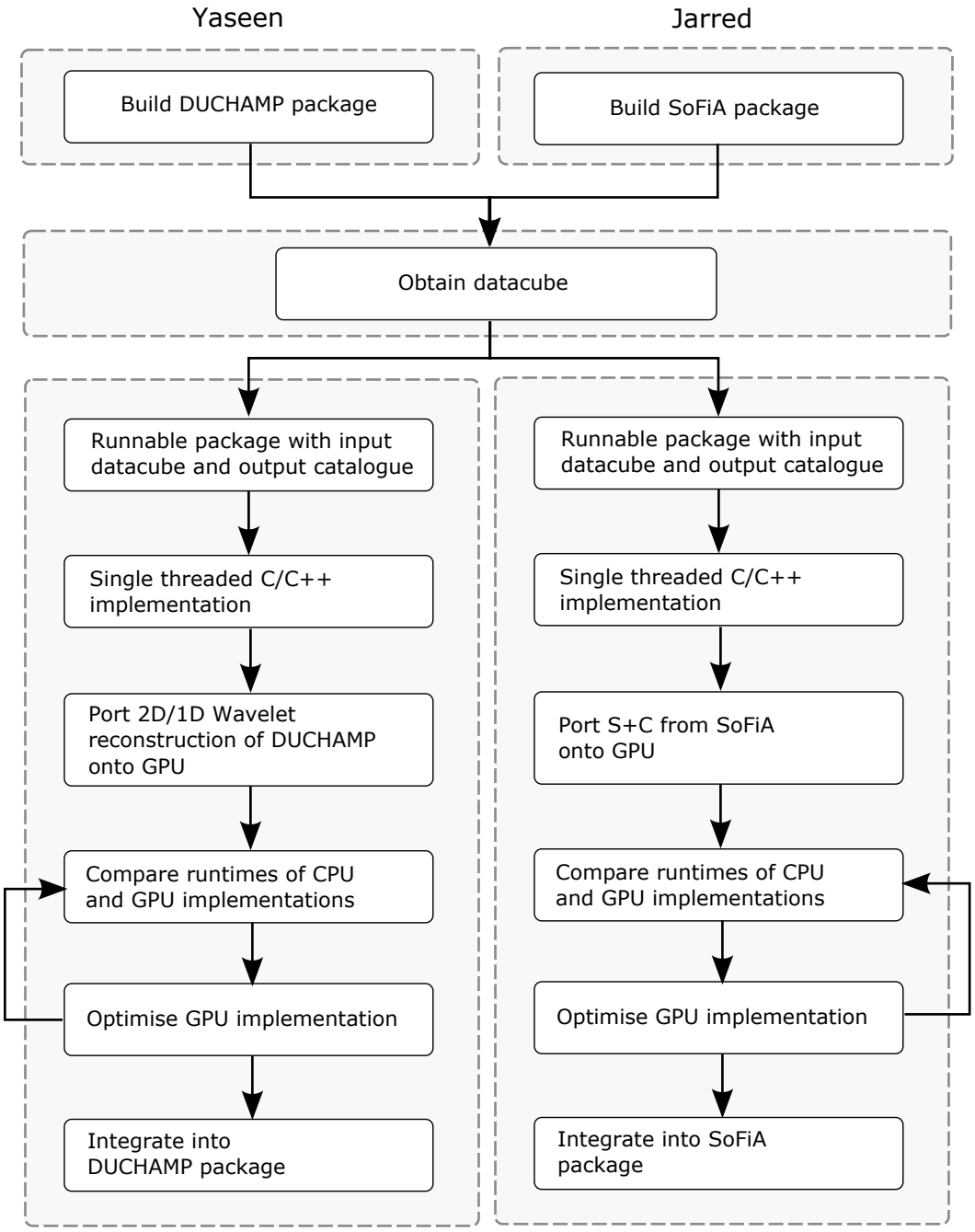

Fig. 1. Work allocation between Yaseen and Jarred

— Final report with work completed, our findings and results.
— Graphs comparing datacube size with execution time on the original framework implementation, our CPU implementations and GPU implementations.

7.6. milestones

— Have checked out and built the relative packages
— Have obtained datacubes and run these against the built packages
— Single threaded implementation
— Have run performance tests on single threaded implementation
— Naive GPU implementation
— Have run performance tests on GPU implementation
— Optimised GPU implementation
— Have run performance tests on Optimised GPU implementation
— Have generated graphs from comparison tests
— Have integrated our code back into the package source repositories