ExaScale High Performance Computing in the Square Kilometer Array

P. Chris Broekema[†,⋆]
broekema@astron.nl

Rob V. van Nieuwpoort[⋆,†]
rob@cs.vu.nl

Henri E. Bal[⋆]
bal@cs.vu.nl

[†]Netherlands Institute for Radio Astronomy (ASTRON)
Postbus 2, 7990 AA, Dwingeloo, The Netherlands

[⋆]VU University Amsterdam
De Boelelaan 1081, 1081 HV, Amsterdam, The Netherlands

ABSTRACT

Next generation radio telescopes will require tremendous amounts of compute power. With the current state of the art, the Square Kilometer Array (SKA), currently entering its pre-construction phase, will require in excess of one ExaFlop/s in order to process and reduce the massive amount of data generated by the sensors. The nature of the processing involved means that conventional high performance computing (HPC) platforms are not ideally suited. Consequently, the SKA project requires active and intensive involvement from both the high performance computing research community, as well as industry, in order to make sure a suitable system is available when the telescope is built. In this paper, we present a first analysis of the processing required, and a tool that will facilitate future analysis and external involvement.

Categories and Subject Descriptors

J.2 [**Physical Sciences and Engineering**]: Astronomy

General Terms

Design, Performance, Reliability

Keywords

ExaScale Computing, Square Kilometer Array, High Performance Computing, Streaming Computing

1. INTRODUCTION

The Square Kilometer Array (SKA) is a next-generation radio telescope currently entering its pre-construction phase. The central processor for the SKA will require computational resources well in excess of what even current top-of-the-line supercomputers can offer. Additionally, the processing done is quite different from conventional high performance computing applications, in computational intensity[1], in its streaming nature, and in its relative robustness against hardware failures.

In this paper, we present an analysis of SKA central processing on future supercomputer hardware, for as far as possible considering the limited information available. We also present an analysis tool that will allow the accurate and detailed analysis of SKA processing on a system level. This is intended to show not only architectural shortcomings of current or near-future high performance computing platforms, but also to identify design points that would make future systems particularly suited for radio astronomy.

This paper is structured as follows. First, we will briefly describe the Square Kilometer Array. In section 3, the computational requirements of the first phase of the SKA are identified, followed by an analysis of radio-astronomical algorithms, as compared to conventional HPC applications. We then present an analysis of the HPC roadmaps and identify where these fall short of our requirements. In section 6, we present a SKA analysis tool, followed by our conclusions. We end by briefly discussing future work.

2. THE SQUARE KILOMETER ARRAY

The Square Kilometer Array is a next-generation radio telescope, with a total collecting area of approximately one square kilometer. It will operate over a very wide frequency range, from 70 MHz to 10 GHz, which will require several different receiver types. It will be built in the southern hemisphere, either in South Africa or Western Australia. The SKA will be capable of extremely high sensitivity and angular resolution.

The extremely high sensitivity and angular resolution requires an array with a very large number of receivers, covering a very large area. SKA sites are projected to extend up to 3000 km from the central core and will contain millions of receivers.

The SKA will cover a frequency range from 70 MHz to 10 GHz. This can't be covered with a single antenna type, so arrays consisting of two, possibly three, different receptors will be built.

- The low end of the frequency range, from 70 to 450 MHz, will be covered by low-frequency sparse aperture arrays of simple dipole antennas (AA-low). Clusters of around 11,200 of these dipole antennas will be grouped into stations of about 180 meters in diameter. It is expected that 250 of these stations will be built. Each AA-low station will eventually produce between 10 and 30 Tb/s of data, to be transported to the central processor, depending on the number of simultaneous

[1]Computational intensity is defined as the number of floating point operations per byte of I/O. Normally I/O is considered to be a memory access, but for the SKA this is often a network read.

©ACM, 2012. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version is published in the proceedings.
AstroHPC'12, June 19, 2012, Delft, The Netherlands.
Copyright 2012 ACM 978-1-4503-1338-4/12/06 ...$10.00.

beams required. In phase 1 this will be limited to ~1 Tb/s per station. In total these stations will generate up to 7.5 Pb/s.

- The mid-frequency range may be covered by a dense aperture array design, using tiles grouped in stations of about 60 meters in diameter (AA-mid). Up to 250 of these stations may be built. AA-mid stations produce the same amount of data as AA-low stations, totalling ~2.5 Pb/s for 250 stations. The AA-mid concept is part of the Advanced Instrumentation Program (AIP). Technologies in this program will be assessed in terms of science impact, cost, and technical readiness, and deployed in SKA phase 2 if shown to be feasible and cost-effective.
- The high end of the frequency range, upward from 500 MHz, will be covered by relatively small dishes, with a Single Pixel Feed (SPF). These will be around 15 meters in diameter. A subset of the dishes may be equipped with a Phased Array Feed (PAF), or a Wide Band Single Pixel Feed (WBSPF). These advanced feeds are also part of the Advanced Instrumentation Program. Around 3000 dishes will be built. Each dish produces ~120 Gb/s, assuming single pixel feeds. PAFs will produce up 10 times more data, WBSPFs around twice the amount of a SPF. All dishes together produce ~360 Tb/s, if fitted with single pixel feeds.

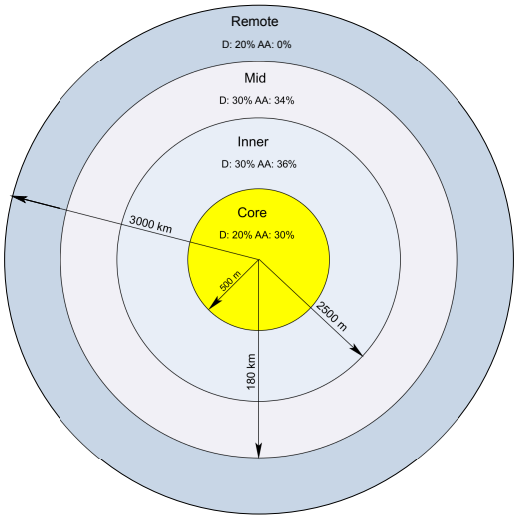

Figure 1: The distribution of collecting area for SKA phase 2

The SKA stations will be divided into four regions. Figure 1 shows the four regions of the SKA and the distribution of the collecting area.

- The core will have a diameter of around 1 km, for each of the receiver types. The dish core and the AA core will be spatially separated from each other.
- The inner region has a diameter of around 5 km. The core and inner regions together will contain more that half of the total collecting area of the SKA.
- A mid region, extending out to 100 km for phase 1, and up to 180 km for phase 2, will contain dishes and pairs of AA-low and, possibly, AA-mid stations. In each case, they will be randomly placed within the area, with the density of dishes and stations falling off towards the outer part of the region.
- The remote region extends from 180 km to 3000 km. This will comprise five spiral arms, along which dishes, grouped into stations of around 20, will be located. The separation of the stations increases towards the outer ends of the spiral arms.

The data from the receivers are transported, using long haul optical links, to a central processor facility. The aggregate data rate into the central processor will be in the order of 10 Pb/s, fittingly described as a data deluge. There is no known way around this central processing; the correlator requires data from all stations in an observation. Figure 2 shows a high level overview of the SKA system.

The central processor can conceptually be divided into the central signal processor, handling correlation and beamforming, visibility processors, responsible for gridding, and image formation, generating images and calibration solutions. The central processor takes data from the SKA stations as input, and delivers calibrated science data as output. The scientific data are stored in the science data archive, intermediate data are, in principle, not stored. The data rate out of the SKA central processor is several orders of magnitude smaller than that coming in.

The feature that sets the SKA central processor apart from conventional high performance computing applications, is its very high input data rate. Since it is unlikely that enough high performance storage will be available, or affordable, to allow batch processing, and the system needs to keep up with the input data stream or risk dropping data, the SKA central processor can be described as a pseudo real-time streaming processor. Note that there is no hard real-time deadline, apart from the requirement to keep up with the input data stream, making the real-time requirements less strict than in a classic real-time application.

An important consequence of the streaming nature of the SKA central processor, is the need to dimension the system to be able to comfortably handle the most demanding application. Since there is no intermediate storage available, insufficient resources will inevitably and immediately lead to data loss if the central processor is under dimensioned.

3. SKA PHASE ONE REQUIREMENTS

As a risk- and cost-reduction measure, the SKA will be built in two phases. Phase 1 (often identified as SKA_1), scheduled to start construction in 2016, will consist of two receiver types, low-frequency sparse aperture arrays and high-frequency dishes, and have a maximum baseline of around 200 kilometers. The preliminary specification of phase 1 defines 50 sparse aperture array stations and 250 dishes, although the science analysis may require more, but smaller, aperture array stations be built. It is likely that instead up to 250 smaller aperture array stations will be built. The cumulative data rate of these smaller stations will be similar to the original 50 bigger stations, but post-processing requirements will increase considerably.

Although the exact computational requirements of the SKA phase 1 are not well defined yet, it is clear from the preliminary requirements[4] that these are well beyond the capabilities of current HPC systems. A full analysis of the science requirements will be done in the pre-construction phase, starting in 2012. An early analysis of the current science requirements, based on the SKA phase 1 Design Reference Mission (DRM)[7], showed considerable differences with the preliminary requirements mentioned earlier[1].

It is of course unrealistic to expect the complete requirements for the software and computing component for a complex system like the SKA to be known at such an early stage. In fact, both software

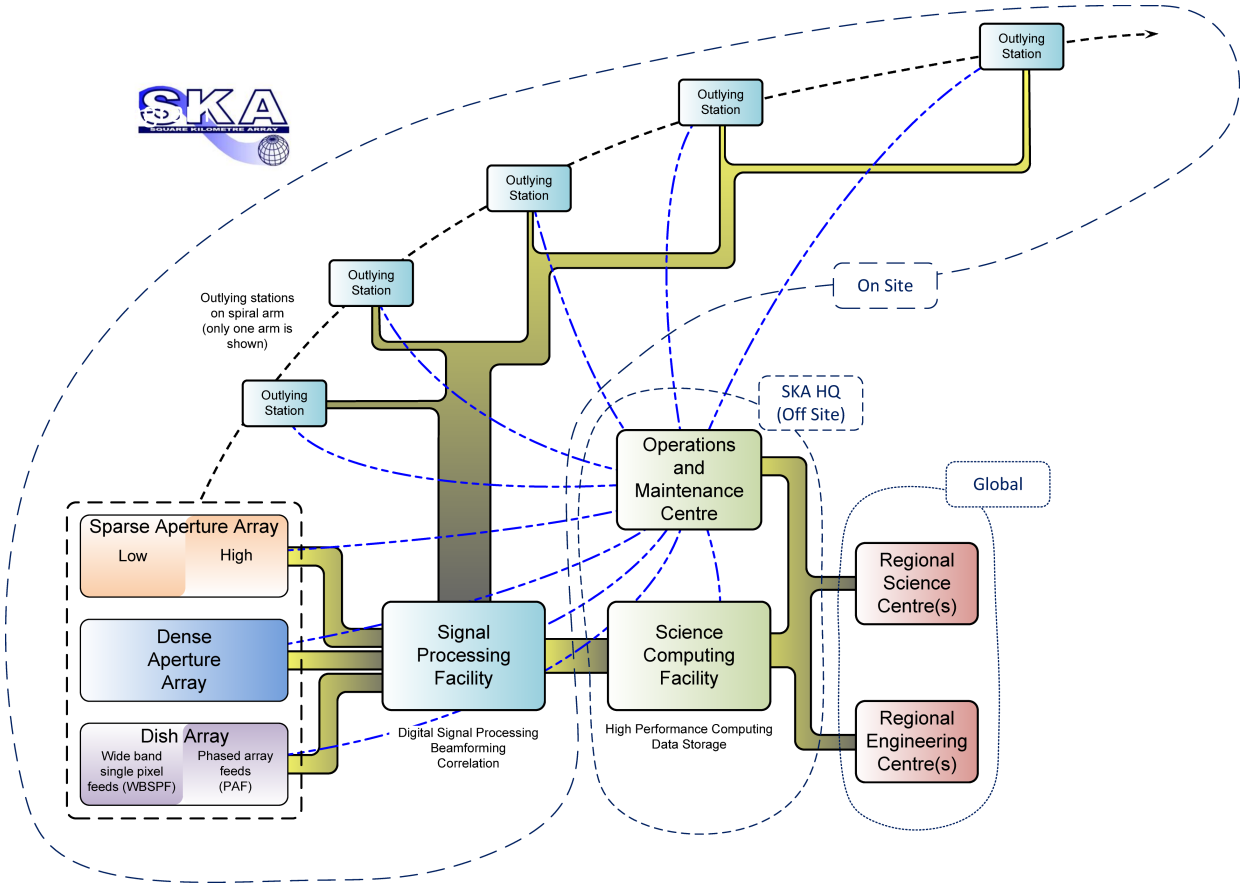

Figure 2: A high level overview of the Square Kilometer Array

engineering experience[12, 18], and experience from the only operational radio telescope comparable to the phase 1 SKA, LOFAR[2], show that requirements are likely to be subject to change. Having said this, the preliminary requirements currently available, give us a good first order estimate of the compute power needed for phase 1 of the SKA.

Figure 3 shows a schematic overview of the SKA phase 1 central processor, based on these preliminary requirements. Note that the minimum compute requirements shown are an absolute minimum. They assume a computational efficiency of 100%. Furthermore, recent experience has shown that the assumption that it takes 10^4 operations to completely process an input sample, may be a serious underestimate. If we assume a computational efficiency of 10%, quite reasonable in HPC, and 10^5 operations per input sample, we would need to scale the central processor to a peak performance of approximately 800 PFlop/s. This only covers the visibility processor and image generation parts of the central processor, it does not include the correlator and beamformer.

Output from the correlator for SKA phase 1 will be in the order of several terabytes per second. This extremely high data rate makes the SKA central processor quite unique among high performance computers. Most supercomputers in the Top500 are designed to handle complex simulations, which typically have a very high computational intensity. If we consider a software based correlator and beamformer, the input data rate increases to around 100 terabytes per second. Although the concept of a highly integrated software based central processor, including the correlator and beamformer, is intriguing and we will argue in section 7 that this may offer significant advantages, in the rest of this paper we will mostly ignore the beamformer and correlator, and concentrate on what figure 3 calls the visibility processors and image formation.

4. RADIO-ASTRONOMICAL HPC

When analyzing the computational feasibility of an instrument, like the SKA, the natural performance figure to look at centers around the floating point arithmetic performance, usually expressed as a LINPACK[11] performance figure. These figures form the basis for the Top500 list of the fastest supercomputers in the world[21].

The LINPACK benchmark is characterized by a large number of parameters, like matrix sizes, that can be freely chosen to allow the user to tailor or optimize the benchmark to the hardware being tested. On the one hand this allows a extensive exploration of the efficiency space, but it also makes LINPACK a highly unrealistic benchmark.

The computational performance of the LINPACK benchmark depends strongly on the double precision floating point performance of matrix-matrix operations, and interconnect latency. Compared to LINPACK, algorithms in radio astronomy are characterized by a very low computational intensity, expected to be in the order of 1 floating point operation per byte of I/O, often even less[8]. Data

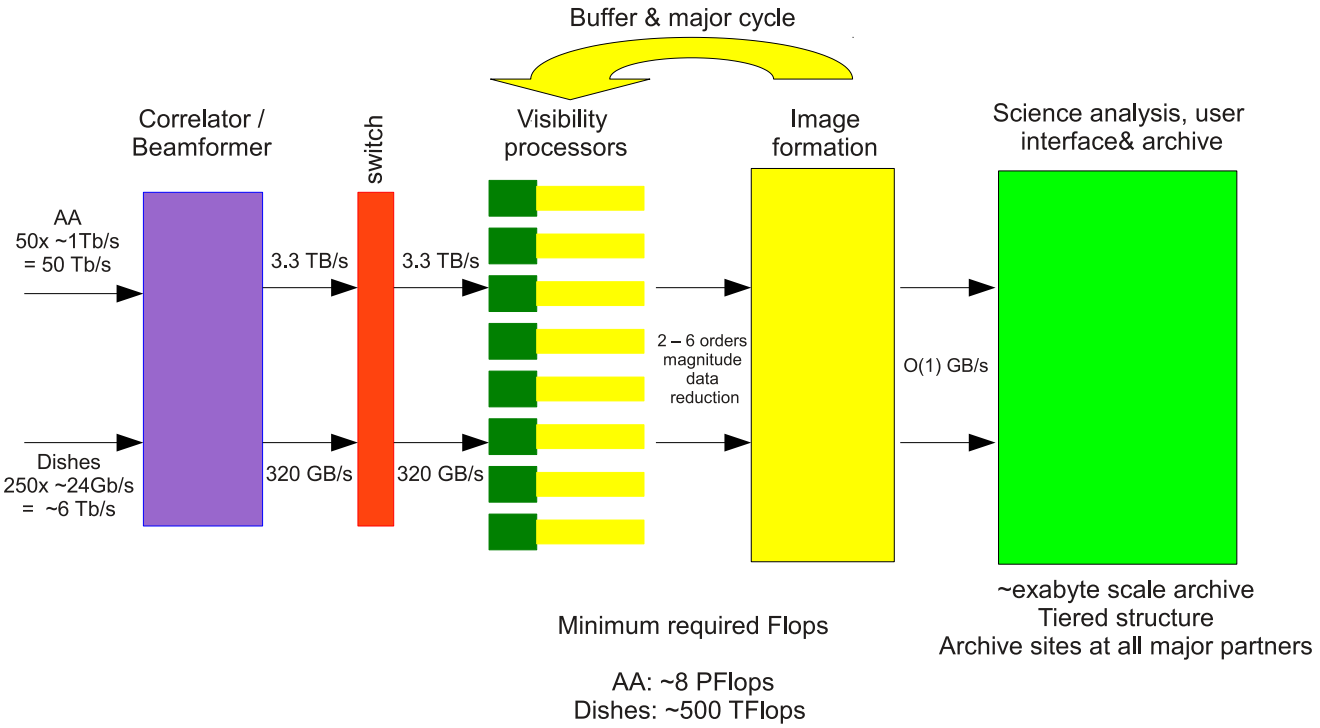

Figure 3: A schematic overview of the SKA phase 1 central processor

are mostly independent in frequency, which means that low latency in an interconnect is far less important than high bandwidth.

Figure 4 shows a flow diagram of the calibration and imaging steps in the SKA. This shows the current state of the art, which may obviously change. It is beyond the scope of this paper to go into the details, but it is important to note that, unlike current radio telescopes, data must be calibrated and imaged online, the data rates involved don't allow temporarily storing intermediate data products.

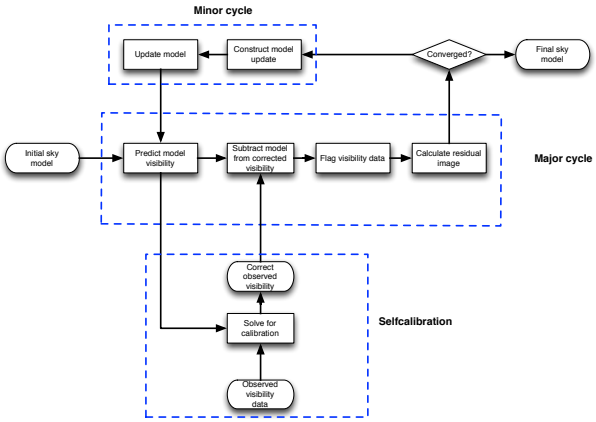

Figure 4: Flow diagram of SKA calibration and imaging

This exposes an interesting challenge. The very high input data rate demands a streaming processing model, without intermediate storage of data products. Unfortunately there is no known single-pass calibration method. Therefore, a very high performance buffer is required to store intermediate data, while the multi-pass calibration algorithm converges.

Nearly all computations are done on complex numbers. Fourier transforms, vector and matrix operations, and complex multiply-adds play an important role. In contrast to most conventional HPC applications, we can probably get away with single-precision floating point operations for a lot of our processing. This may significantly reduce data rates and, assuming appropriate hardware support, the size of the central processor. The exact ratio of single- and double-precision processing, as well as an accurate decomposition of the exact operations required and the exact computational intensity of the various processing steps, will be part of a detailed investigation in the upcoming pre-construction phase.

While the LINPACK benchmark gives us a reasonable idea of the performance characteristics of current state of the art supercomputers, it is of limited use for the evaluation of a system for the SKA.

5. HPC ROADMAP ANALYSIS

The most powerful supercomputers in the world, according to their performance in the LINPACK benchmark, are ranked in the Top500 list. This list is updated twice per year and goes back to 1993. Plotting the aggregate performance of the Top500 machines shows a steady increase in available compute power over the last two decades, mostly consistent with Moore's law.

A straightforward extrapolation of the past lists shows that a machine capable of handling the phase 1 central processor requirements, should be available by 2018 - 2019. This is illustrated in figure 5. This only shows the development in compute power, as measured using the LINPACK benchmark.

In 2008 the ExaScale panel published a report on the expected developments in high performance computing in the coming decade[15]. By looking at current, and expected developments in the

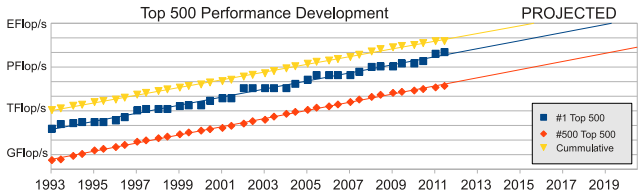

Figure 5: Top 500 extrapolation shows ExaScale systems available by 2018 - 2019

different components of a high performance computer, a projection was made of a feasible ExaScale system. Although one should carefully weigh the value of long term predictions like the ones in this study, they did highlight some disturbing trends.

One of the most obvious developments in HPC is the increase in overall concurrency. Moore's law, as interpreted as the doubling of the number of components per unit of area on a chip every 18-24 months, is expected to continue to hold for the next decade or so, which means that feature sizes in future processor will continue to decrease for the foreseeable future. Due to increased leakage power at small feature sizes, processor clock frequency has leveled off. Future systems are expected to continue to run at a clock frequency in the order of one to several Gigahertz.

The power budget available for a single processor socket has also leveled off. The practical limit for commodity cooling solutions is around 150W per socket. Water-cooling may raise this limit slightly. In the future we'll see aggressive and fine grained power gating shutting down unused parts of a CPU, allowing the remaining components to dynamically scale in performance to fill the available thermal budget. It is likely that the available thermal budget per socket will be insufficient to allow all components in a processor to run at full power simultaneously.

So the trend is that individual cores tend to not increase in performance very much, certainly not sufficiently to follow Moore's law. Shrinking feature sizes, however, allow us to add ever-increasing numbers of cores on a CPU. Additionally, many-core architectures, like GPUs and special purpose accelerators, can now be integrated into the CPU. These developments lead to a massive increase in required application concurrency to efficiently use the available resources. Nevertheless, these developments are, by themselves, not enough to reach the performance levels shown in figure 5. In order to bridge that gap, an increase in the total number of processors is also required, possibly with additional accelerator hardware. Regardless, both relative memory size and bandwidth are unlikely to keep up.

So ExaScale systems will be characterized by massive parallelism on many levels. Huge numbers of nodes, possibly of various types, will be connected to a cohesive but highly complex system. Within a node, and even within a processor, we'll see various levels of parallelism. It is probable that processors will be heterogeneous, consisting of both a smaller number of general purpose, complex, super-scalar, out-of-order cores, and many, much simpler, cores optimized for floating point operations. It is possible that these will be augmented by a number of special purpose accelerators. The heterogeneous nature of these processors makes them relatively hard to program, but the potential performance and efficiency of such a system is tremendous.

Conventional HPC applications are often relatively compute intensive; the number of Flops per bit of I/O is very large. SKA processing, in contrast, contains a significant portion of operations with very low computational intensity. The streaming nature of the SKA central processor emphasizes this. Although most HPC applications will notice the significantly reduced memory bandwidth per Flop available in future systems, the I/O bound and streaming nature of SKA processing makes this a particularly significant problem for us.

	2009 Jaguar	2011 'K' computer	2018 (projected)	2009 vs 2018
System R_{peak}	2 PF	10 PF	1 EF	*O(1000)*
Node R_{peak}	125 GF	128 GF	1 - 15 TF	*O(10 - 100)*
Energy	6 MW	10 MW	20 MW	*O(10)*
Energy/Flop	3 nJ/F	1 nJ/F	20 pJ/F	*-O(100*
System memory	0.3 PB	1 PB	32-64 PB	*O(100)*
Memory/Flop	**0.6 B/F**	**0.1 B/F**	**0.03 B/F**	***-O(10)***
Memory bw/node	25 GB/s	64 GB/s	2 - 4 TB/s	*O(100)*
Memory bw/Flop	**0.2 B/s/F**	**0.5 B/s/F**	**0.002 B/s/F**	***-O(100)***
Total concurrency	**225,000**	**548,352**	**O(10^9)**	***O(10^5)***
MTTI[2]	days	days	hours	*-O(10)*

Table 1: A projected 2018 supercomputer compared to two current ones

In table 1 two recent top-of-the-line supercomputers are compared to the projected ExaScale machine as predicted by the ExaScale panel. A number of interesting features have been selected for comparison. Particular pain points are highlighted in bold face, the decrease in available memory bandwidth per Flop is especially worrying. In the next few sections we will investigate the impact of the various trends shown in this study.

Energy consumption

The ExaScale study used a total power consumption of 20MW as a design target for an ExaScale machine. They argued that this allowed some growth beyond that of todays largest systems, but still not be so high as to preclude it from deployment in anything other than specialized strategic national defense applications. The same study also concluded that even the most optimistic projections with respect to improvements in energy consumption per Flop, still fall short of this target by several factors.

The scale of the problem becomes clear when we look at the current state of the art. IBM's Blue Gene/Q prototypes are currently the most energy efficient machines in the Top500, by a significant margin[6]. These machines offer an efficiency of 2 GFlop/s/W. The design goal of an ExaScale machine within a 20 MW power envelope requires an efficiency of at least 50 GFlop/s/W.

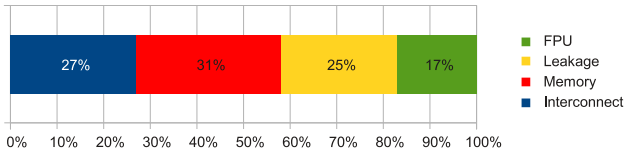

Figure 6: Energy distribution in an ExaScale system

Figure 6 shows the energy distribution in the most optimistic projection of a 2018 ExaScale supercomputer. Even in this very optimistic projection, more than half of the energy consumed is needed for I/O. It is worrying to note, though, that external I/O, i.e. data coming into or going out of the machine, is not taken into account at all. The ExaScale panel mainly considered conventional HPC applications, which are characterized by very little external I/O. The SKA central processor, in contrast, is characterized by a massive data stream into the system and a much smaller, but in comparison

[2]Mean Time To Interrupt

to conventional HPC applications still significant, stream of data out to the science data archive.

Input/output

The problem of input/output is closely related to that of energy consumption. As was shown in the previous section, a world-class supercomputer in the SKA time frame will consume the bulk of its energy moving bits around. Unfortunately, most HPC roadmaps limit I/O predictions to memory and interconnect bandwidth, and they often ignore external data transport.

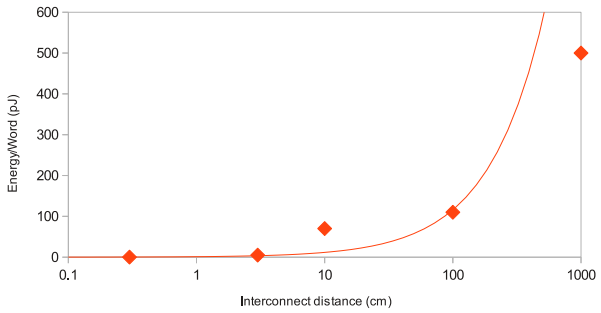

Figure 7: Energy required for I/O

Figure 7 shows the energy required to transport a word of data over a given distance[10]. As distance to the CPU increases, the energy required increases superlinearly. The streaming nature and the massive input data rate of the SKA central processor mean that the energy consumed for I/O in the SKA central processor will account for much more than the 58% shown in figure 6.

Experience with the LOFAR radio telescope has shown that, although a real-time streaming central processor is a feasible and very flexible proposition[13, 17], significant work is often needed to modify the system software designed and optimized for conventional HPC applications[9, 16, 22].

Programmability

In the next decade or so we'll see disruptive changes in the way compute hardware is designed and built. Massive parallelism, combined with heterogeneous and bandwidth starved architectures, will have to be handled by all HPC developers, but for the SKA the problem is amplified.

The streaming nature of the SKA central processor, as well as the tremendous input data rate, make it unique in high performance computing. This may mean that the programming models that will be developed to handle the features of future supercomputers are not suitable for our application. To make sure we can efficiently use future hardware, we may need to develop these programming models ourselves.

Current radio telescopes use a significant number of legacy codes unsuitable for deployment on an ExaScale system. Some of these handle relatively simple things, like coordinate transformations and so on. These, casacore[3] and wcslib[5] are some examples, will have to be rewritten for future systems. This is time consuming, but the algorithms are well known and this should not be a significant risk. Others deal with processing the massive data stream from the correlator and beamformer. For these codes, merely porting existing codes used by the pathfinder and precursor instruments is not enough. Significant algorithm development is needed as well, to make sure it can handle the data volumes, achieve the science requirements, and efficiently use future ExaScale systems. This must be considered a significant risk.

Reliability

In the next few years we'll also see a tremendous increase in the number of components in a supercomputer, as illustrated by the total concurrency row in table 1. Since reliability per component is not expected to increase significantly, it is inevitable that total reliability of the system will decrease. In fact, one can say that the system must be considered somewhat broken all of the time.

In contrast to conventional HPC applications, SKA processing is relatively robust against failures. Within reason, the data are embarrassingly parallel, and loss of a small portion of data is often quite acceptable. As long as the system software is capable of detecting and reporting failed nodes, this reduced system reliability should not present a serious problem.

6. HPC INVOLVEMENT

The SKA community has been remarkably successful in attracting interest in the project, both from industry, and from the HPC research community. Since the SKA community itself lacks much of the experience and, frankly, critical mass required to handle the disruptive technology changes expected to occur in the next couple of years, it is essential to leverage this generated interest and get an increased involvement of industry and academia in the project.

One way of doing this is demonstrated by Lawrence Livermore National Lab. For their new supercomputer, Sequoia, they've published a set of sixteen representative benchmark codes[19]. These codes give a useful insight into the scalability and efficiency problems these applications face, but the obvious goal is of course to enable industry to optimize their architectures, software, and programming models to perform optimally for these applications.

A SKA analysis tool

ASTRON, VU University Amsterdam, and Delft University of Technology have in the past few years concentrated a lot of research on various computational kernels on many-core architectures. These were optimized for the specific architecture targeted and showed in great detail what the bottlenecks for that particular platform are[14]. This work will continue for future architectures.

Additionally, we've now produced the beginnings of a framework and a small set of reference codes that are far more generic, but should show in detail what operations are required in SKA processing. These are written for clarity, and show not only the computational processing, but, more importantly, also the data-flow through the system. It is possible, in theory, to model the entire life cycle of the data processing, from instrument to final image, using this framework.

The first version, presented in this paper, will concentrate on the framework, providing datastructures and streaming datacommunication primitives. A small set of computational codes are provided and more will be added as they become available[20].

The main goal of this effort is to provide the broader community with a representative and coherent set of sample codes that allow them a more detailed insight into the challenges that the SKA faces.

We present a first set of codes that are heavily based on the operational and proven framework of the LOFAR real-time central processor. All platform specific code was removed, as was a lot of the LOFAR infrastructure not needed for a simple analysis tool. Datastructures and sample codes are provided in the initial version, as well as primitives for a small number of communication interfaces. There are no external dependencies, apart from some POSIX headers, some Boost libraries, and, optionally, MPI.

Initially we will provide a number of simple algorithms, a correlator, a beamformer and a polyphase filter, characterized by their streaming nature and very low computational intensity. When they become available, more complex algorithms, image formation, gridding and dedispersion for instance, will be added. All of these will be reference implementations, written for clarity not for performance.

7. CONCLUSIONS

It is clear that the central processor of the SKA is quite different from conventional HPC applications. Even though the LINPACK benchmark is already of limited value for normal HPC applications, its emphasis on low-latency interconnects and high computational intensity matrix-matrix operations means LINPACK figures are often misleading for our application.

This highlights the underlying problem: due to the markedly different properties of normal HPC applications versus radio-astronomical reductions, conventional future HPC installations are ill suited for SKA processing. How this can be improved should be part of a detailed study in the oncoming SKA pre-construction phase.

The very high data rates involved in the central processor are a major concern, especially considering the fact that moving bits around is going to account for the bulk of the energy consumed in a future supercomputer. Any design or technique that limits the amount of data movement should be seriously considered. This also means that optimization of code should no longer focus on maximum utilization of the computational resources, but instead try to minimize energy consumption. In practice this will often mean minimizing I/O.

A highly integrated central processor, combining correlator, beamformer and science data processing in a single integrated solution that avoids highly inefficient switches in favor of integrated backplane communication, may well be more energy efficient than separate components. It is also important to realize that, although the efficiency in Joules/Op of custom hardware, FPGAs for instance, is still, and will probably continue to be, unbeatable for simple operations, like the correlator or beamformer, this is offset by power hungry data transport between the dedicated hardware solution and the general purpose science data processor. In other words, Joules/Op is really not a suitable metric to evaluate the efficiency of a system, since this ignores data transport. Instead a measure for the energy consumed by the entire central processor for a unit of scientific data, Joules per generated image pixel for instance, should be adopted.

Streaming processing support, both in hardware, and in software, is, and will continue to be, limited. Unfortunately, the SKA is quite unique in HPC in its requirement for very high input data rate and streaming processing of data. This means that we cannot expect any off the shelf solution to be immediately suited for SKA central processing. LOFAR experience has shown that significant work, especially streaming I/O related, is often needed to optimize an otherwise excellent HPC platform for radio astronomy.

The unique nature of our problem should be leveraged as an interesting case study for industry and research alike. Getting a challenging streaming application to work efficiently on a platform requires all components involved, hardware, operating system, communication middleware, and software, to work together in the most optimal way possible. In other words, if a streaming pseudo real-time application, like the SKA central processor, works efficiently on a platform, most other more conventional applications will also benefit from the optimizations required to get to that point.

By publishing a relatively simple, but representative set of codes, we give researchers and industry representatives the opportunity to get a more accurate idea of the processing challenges faced by the SKA. This will hopefully help turn the significant academic and industry interest generated by the project into useful research.

8. FUTURE WORK

With the analysis framework introduced in this paper we intend to start an in depth analysis of the computational requirements of the SKA. We will show, by optimizing parts of the system on a modern and challenging platform, what architectural choices work well for our application, and where that platform can be improved. This will result in a more generic set of design points for a system optimized for SKA like operations.

This is meant to expose architectural design points particular to the SKA. It is hoped that, with a detailed and comprehensive analysis of the SKA system, industry can provide an architecture particularly suited for our application.

Acknowledgments

Tim Cornwell (CSIRO/CASS) provided the flow diagram showing imaging and calibration (figure 4). Figures 2 and 1 were kindly provided by Peter Dewdney of the SKA Project Office (SPO). This work is supported by the SKA-NN grant from the EFRO/Koers Noord programme from Samenwerkingsverband Noord-Nederland, and the ASTRON / IBM Dome project, funded by the province Drenthe and the Dutch Ministry of EL&I.

9. REFERENCES

[1] P. Alexander et al. Analysis of requirements derived from the DRM, August 2011. SKA Software and Computing CoDR.

[2] H.R. Butcher. LOFAR: First of a New Generation of Radio Telescopes. *Proceedings of the SPIE*, 5489:537–544, October 2004.

[3] casacore. http://code.google.com/p/casacore/.

[4] P. Dewdney et al. SKA phase 1: Preliminary system description, 2010.

[5] FITS world coordinate systems. http://www.atnf.csiro.au/people/mcalabre/WCS/.

[6] The Green500 list. http://www.green500.org.

[7] SKA Science Working Group. The square kilometre array design reference mission: SKA phase 1 v. 2.0, September 2011.

[8] Ben Humphreys and Chris Broekema. HPC technology roadmap. Technical report, SPDO, December 2011. SKA Software and Computing CoDR.

[9] Kamil Iskra, John W. Romein, Kazutomo Yoshii, and Pete Beckman. ZOID: I/O-Forwarding Infrastructure for Petascale Architectures. In *ACM Symposium on Principles and Practice of Parallel Programming (PPoPP'08)*, pages 153–162, Salt Lake City, UT, February 2008.

[10] Peter M. Kogge. Energy at exaflops. Supercomputing, 2009. The ExaScale Panel.

[11] The LINPACK benchmark. http://www.netlib.org/benchmark/hpl/.

[12] Steve McConnell. *Code Complete, Second Edition*. Microsoft Press, Redmond, WA, USA, 2004.

[13] Jan David Mol and John W. Romein. The LOFAR Beam Former: Implementation and Performance Analysis. In *EuroPar'11*, volume LNCS 6853, Part II, pages 328–339, Bordeaux, France, August 2011.

[14] R. V. van Nieuwpoort and J. W. Romein. Correlating Radio Astronomy Signals with Many-Core Hardware. *International Journal of Parallel Processing*, 1(39):88–114, February 2011.

[15] Peter M. Kogge et al. ExaScale Computing Study: Technology Challenges in Achieving ExaScale Systems, September 2008.

[16] John W. Romein. FCNP: Fast I/O on the Blue Gene/P. In *Parallel and Distributed Processing Techniques and Applications (PDPTA'09)*, volume 1, pages 225–231, Las Vegas, NV, July 2009.

[17] John W. Romein, P. Chris Broekema, Jan David Mol, and Rob V. van Nieuwpoort. The LOFAR Correlator: Implementation and Performance Analysis. In *ACM Symposium on Principles and Practice of Parallel Programming (PPoPP'10)*, pages 169–178, Bangalore, India, January 2010.

[18] Winston Royce. Managing the development of large software systems. volume 26 of *WESCON*. IEEE, August 1970.

[19] ASC Sequoia Benchmark Codes. http://asc.llnl.gov/sequoia/benchmarks/.

[20] A SKA analysis tool. http://www.exaska.org.

[21] The Top500 list. http://www.top500.org.

[22] Kazutomo Yoshii, Kamil Iskra, Harish Naik, Pete Beckman, and P. Chris Broekema. Performance and Scalability Evaluation of 'Big Memory' on Blue Gene Linux. *International Journal of High Performance Computing Applications*, 25:148–160, May 2011. first published online on May 12, 2010.