PETASCALE COMPUTING CHALLENGES FOR THE SKA

ATHOL KEMBALL

NATIONAL CENTER FOR SUPERCOMPUTING APPLICATIONS

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

1 MAY 2008

ABSTRACT

The science goals of the Square Kilometer Array (SKA) dictate that it will be a telescope with data rates and associated processing demands that are in the petascale or exascale regime. Accordingly, the SKA computing problem, broadly defined, is widely recognized as a significant feasibility and construction cost equation risk. In this memo, we consider the specific challenges in computing on this scale, with particular reference to the likely architectures of petascale computing systems in the next decade and general issues that will affect the mapping of SKA calibration and processing algorithms to these systems. We provide approximate hardware cost scaling relations and re-examine preliminary assumptions about software cost models for the telescope.

INTRODUCTION

Many important key science goals of the Square Kilometer Array (SKA) require high-sensitivity wide-field imaging, or high time-resolution non-imaging analysis, over large cosmic survey volumes. These fundamental scientific requirements, in common with other contemporary telescopes such as the Large Synoptic Survey Telescope (LSST), inexorably require very high data acquisition rates and matching post-correlation processing performance in order to achieve the high-throughput survey modes that the science goals require. The SKA places us in the realm of petabyte (PB) data volumes and petaflop (PF) to exaflop (EF) processing rates (Cornwell 2008). The magnitude of the computing challenge for the SKA is sufficient for it to be a dominant cost and feasibility driver for the telescope. Considered in more detail, the general computing challenge for the SKA consists of several distinct issues, which can be posed broadly as the following questions:

a) Do calibration and processing algorithms exist to produce SKA imaging or non-imaging science products with the fidelity and dynamic range required for the telescope science goals ?

b) Can these algorithms achieve the required levels of computing performance for SKA data and processing rates ?

c) What is the construction and operations cost of the deployed computing hardware needed for post-correlation processing ?

d) What is the cost and scale of the software development effort needed for SKA post-correlation processing ?

In this initial memo, we review the dominant contributors to issues (b) through (d), as informed by past high-performance computing (HPC) experience in both radio astronomy as well as other disciplines in the physical sciences, and by current anticipated trends in high-end computing (HEC) architectures over the proposed SKA development and construction time-scale.

IMAGING AND CALIBRATION COSTS

As noted by Perley & Clark (2003), the intrinsic data rates for Large-N Small-Diameter (LNSD) SKA array designs are inherently large. From basic sampling considerations for full-field imaging at uniform resolution, the following relations can be derived for visibility output rates,

$$\left(\frac{\dot{V}_{vis}}{TBps}\right) = \frac{20 N_{chan} N_{ant} (N_{ant} - 1)}{10^{12} \Delta t} \tag{1.1}$$

$$\left(\frac{\dot{V}_{vis}}{TBps}\right) \sim 10^{-14} \left(\frac{N_{ant} B}{D}\right)^2 \left(\frac{\Delta \nu}{\nu}\right) \tag{1.2}$$

where D denotes dish diameter, B is the maximum baseline length, $\Delta\nu$ is the bandwidth, ν the observing frequency, N_{ant} the number of antennas, and $\dot{V}_{vis}$ the output visibility rate in TB per second. Example output data rates derived from this relation are shown in Figure 1.

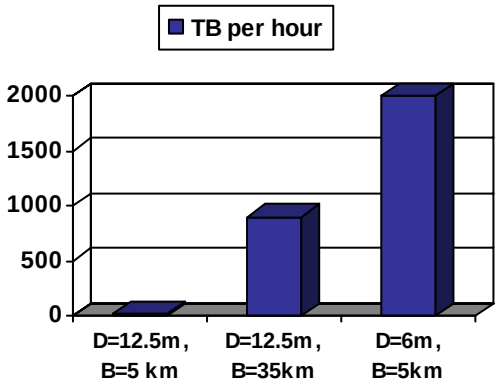

Figure 1. Projected data rates for full-field imaging at uniform resolution for LNSD SKA designs as a function of antenna diameter and maximum baseline length (from equation [1.2]).

Lonsdale et al. (2004) have raised the important possibility of reducing data rates by field-of-view (FOV) shaping; this research is continuing. Continuum wide-field imaging costs for the LNSD SKA design have been considered by Perley & Clark (2003) and Cornwell (2004ff). We derive a basic empirical relationship for continuum wide-field imaging costs, C, (including deconvolution) based the

scaling relations in Cornwell (2004). This is plotted for different SKA design parameters in Figure 2 below.

$$\left(\frac{C}{PF}\right) \approx 22.3 N_{ant}^2 \left(\frac{B}{D}\right)^2 10^{0.00273\frac{B}{D^2}-15} \tag{1.3}$$

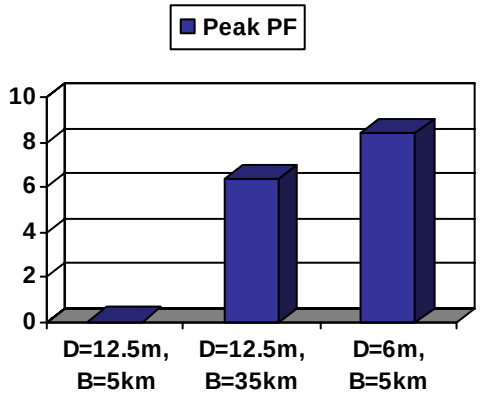

Figure 2. Predicted continuum wide-field imaging costs, following Cornwell (2004).

These continuum costs however are exceeded by spectral-line imaging costs, even without deconvolution, which scale by an additional factor of the number of channels, and produce estimates for the full SKA processing in the PF to EF regime (Cornwell 2008).

SCALABILITY

We consider now the question of whether existing radio-interferometric calibration and processing algorithms can achieve the level of performance required for the SKA data and processing rates described above, when deployed on HEC systems anticipated to be available at the time of SKA construction in the next decade. This requires algorithms with high scalability; such algorithms sustain a high ratio of actual deployed performance to nominal peak system performance, as HEC peak system performance is increased to the required scale. For the Phase 1 SKA at mid-decade, this required peak performance is in the petaflop (PF) regime. Scalability depends acutely on the efficiency with which the algorithm can be mapped to the underlying HEC architecture.

To understand this question, it is first important to consider the key elements of HEC systems and architectures likely in the SKA era. The peak performance of the fastest general-purpose computers is widely known to increase exponentially over time (see Figure 3), although this simple geometric relation hides underlying complexity that needs to be understood in order to answer the question of SKA algorithm scalability. HEC systems achieve their performance fundamentally through parallelism: the aggregation of large numbers of individual processors within an overall HEC architecture. At the highest level, an HEC architecture can be categorized by the design and performance of key sub-systems needed for processor aggregation, including: i) a memory access sub-system, defining the symmetry, latency, bandwidth, and CPU cache coherency of memory access by each processor (to either shared or local memory); ii) a system topology, defining the degree of

clustering of processors into individual nodes and their inter-connection; iii) a network inter-connect for processor or node communication; and, iii) an I/O system for local or shared file system access. The HEC architecture also depends on the intrinsic processor architecture, including the balance of vectorization, pipelining, scalar, and multi-core capabilities on each processor chip. The architecture is also defined by the degree of node-level parallelism, and whether the processors are augmented by heterogeneous processing elements, including FPGAs and GPUs. HEC architectures typically express a single dominant programming model, reflecting their underlying hardware, optimizations. This is usually characterized by the degree of heterogeneity of the processing and data distributed to each processor, e.g. Single Program Multiple Data (SPMD), or Multiple Program Multiple Data (MPMD), as examples. A given HEC design pattern typically has a lifetime of approximately a decade before it is overtaken by architectural and technological evolution.

The exponential increase in HEC peak performance over time has been achieved by the exponential evolution in transistor chip density on integrated circuits (including CPUs) predicted by Moore's Law (Moore 1965), in concert with advances in HEC and processor architecture designs.

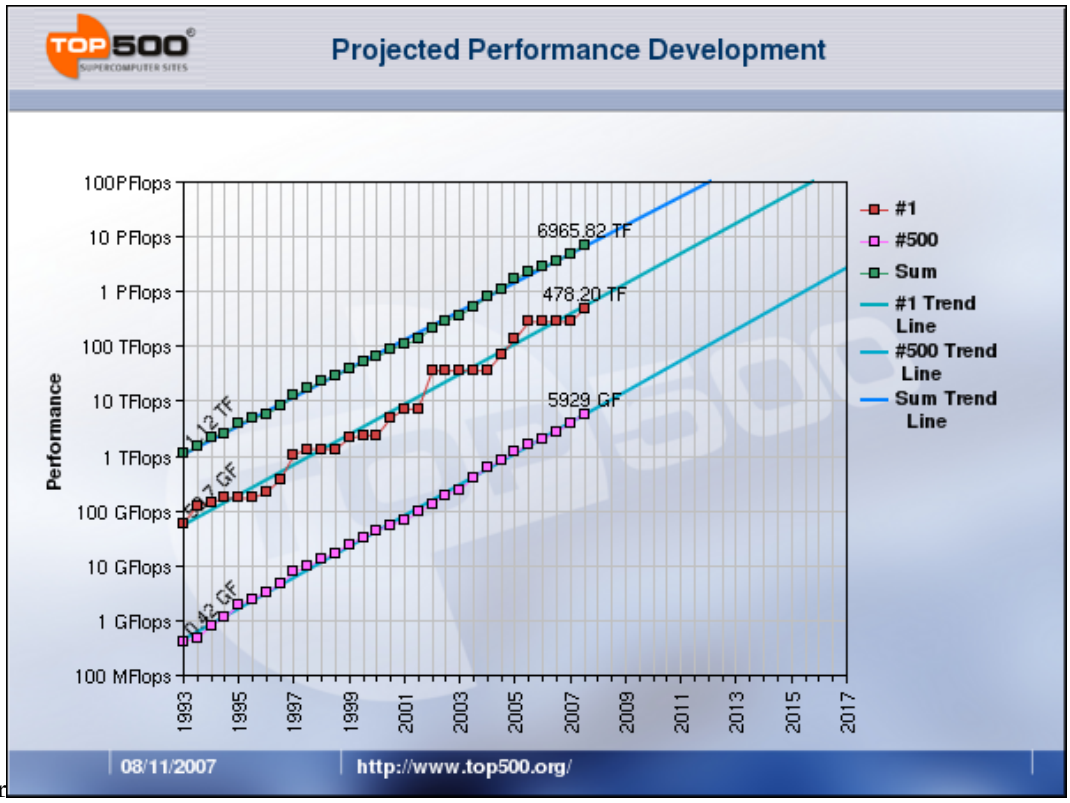

Figure 3. Projected performance development for the top 500 most powerful computer systems as maintained by the TOP500 project (http://www.top500.org). All performances figures are the best performance achieved with the floating-point linear algebra LINPACK benchmark (Dongarra, Luszczek, & Petitet 2001). The evolution of the LINPACK performance for the top 500 systems over time is plotted separately for the cumulative sum (upper), fastest system (middle), and slowest system (lower) on the list.

For an algorithm to scale efficiently on a given HEC system, experience across the physical sciences indicates that it will require substantial optimization targeted at the programming model and intrinsic capabilities expressed by the underlying HEC and processor architectures. In practice, the sustained performance of HPC codes relative to the peak nominal system performance is frequently

low. The degree of intrinsic parallelism in the computation sets an upper limit on the speed-up possible via Amdahl's Law. For a computation with parallel fraction p, that can be sped up by a factor s through parallel distribution, the speed-up for the overall computation will be:

$$\frac{1}{(1-p)+\frac{p}{s}}$$

Asymptotically, in the limit $s \to \infty$ (equivalent to the number of processors increasing without limit), the overall speed-up is still limited to $1/(1-p)$. The fraction of this speed-up that can be realized in practice at a given scale depends on the efficiency with which the algorithm can be mapped to the underlying HEC and processor architecture at that scale.

The fastest system on the TOP 500 list shown in Figure 3 often increases in discrete steps, which may track architectural innovations or transitions in HPC funding levels. A simple regression over the time period 1993 to 2007 smoothes out these variations however, and produces a mean empirical relation:

$$\left(\frac{R_{\max}}{TF}\right) \sim 0.05555\, e^{0.6217(t_{yr}-1993)} \tag{1.4}$$

where R_{max} is the predicted peak LINPACK performance in TF, and t_{yr} is the year. Note that this fit suggests a doubling time of $\Delta t = \ln(2)/0.6217 \sim 1.11$ year. Using this relation, the extrapolated peak LINPACK performance of the likely fastest general-purpose supercomputer in the SKA era is plotted in Figure 4 below.

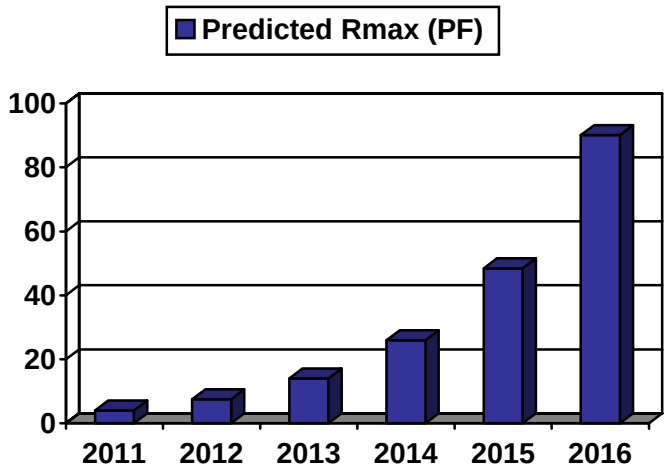

Figure 4. Projected peak LINPACK performance for the likely fastest TOP 500 supercomputer in the SKA era.

To understand the scalability of SKA calibration and imaging algorithms on such a early- to mid-decade PF system we need to consider the likely generic architecture, by extrapolating current architectural trends. Since approximately 2004, the relative increase in the clock speed of single CPUs has slowed dramatically, primarily due to on-chip power-dissipation limits. The additional transistor

count available via Moore's Law has been instead laterally expanded into multi-core CPU architectures. At present 8 cores per node are readily available in community CPUs; by 2015 this may be increased by a factor of 4 to 8 (tens of cores per node), but with only a modest increase in clock speed per core. To achieve a 10-20 PF peak in the 2012 time-frame, a generic PF system will require $O(10^5)$ cores; by mid- to late-decade that will likely be $O(10^{6-7})$. Note that there will be multiple threads of execution per core, increasing the level of parallelism (concurrency) by up to an order of magnitude above the extrapolated number of processor cores. This is a very high degree of hardware parallelism relative to current systems, as shown schematically in Figure 5. A typical teraflop (TF) application needs to scale over only $O(10^{3-4})$ cores.

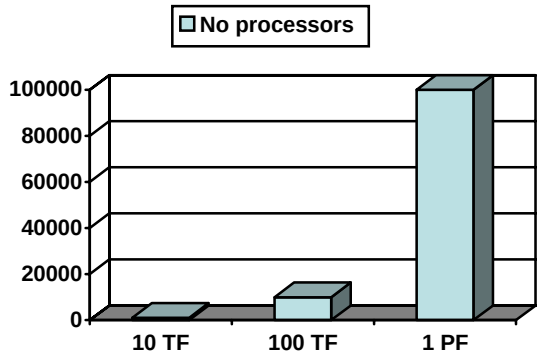

Figure 5. Order-of-magnitude scaling of the number of processor cores as a function of peak performance, if current architectural trends are followed.

In addition to the very high level of concurrency, a generic petascale system in the SKA era will also very likely have a heterogeneous processor architecture, with integrated hardware accelerators such as FPGAs and GPUs. These systems are attractive for reasons discussed further below. A generic PF system will also likely have a highly-clustered topology, with shared-memory nodes, and deep memory and I/O hierarchies, i.e. highly asymmetric memory and resource access times.

These generic architectural characteristics present a significant feasibility and cost risk for the scalability of SKA calibration and imaging algorithms. SKA algorithms need to scale over 10^{5-6} processor cores without significant loss of performance. Also, as the probability of failure increases with the number of processor cores, this will require new robust approaches to reliability and fault tolerance, particularly in a near real-time environment. In addition, the large degree of concurrency will require care in load-balancing and adaptive scheduling of individual execution threads. If a parallel application has a fixed and predictable execution time per parallel block, it can be distributed over a large parallel system without appreciable wasted time on individual cores as all cores should complete each assigned work segment at nearly the same time. If the pattern of computation is dynamic however, adaptive load-balancing is required to allocate the work evenly across the system.

Given the arguments outlined in this section, it is clear that the scalability of SKA calibration and imaging algorithms needs to be proven as a key element of the telescope design, development, and costing study. Task decomposition by spectral channel is an attractive approach, and has previously been used in the parallelization of radio interferometric imaging (Roberts & Crutcher 1996; Roberts et al. 1999), and it remains highly-preferred for the SKA and SKA pathfinders (Cornwell 2008). However, SKA will only have 16k or 32k spectral channels in spectral-line mode, several orders of

magnitude lower than the anticipated level of hardware concurrency, which is $O(10^{6-7})$. Hierarchical parallelism will clearly be needed; however further work is required to determine the exact degree of parallel coupling between adjacent processes or threads within an overall hierarchical decomposition. Wide-field imaging using facet-based tessellation has previously been parallelized per imaging facet (Golap et al. 2001) but this falls 4-5 orders of magnitude beneath the anticipated hardware parallelism so could only be used within an hierarchical parallel distribution. A full analysis and demonstration of SKA calibration and processing algorithm scalability is essential to the assessment of SKA feasibility and costing. This is an active area of current and future work for the SKA and SKA pathfinders; for example, a component-based HPC application architecture for radio-interferometric imaging is discussed by Kemball, Crutcher, & Hasan (2008).

HARDWARE COSTS

Computing hardware system costs vary primarily over two axes: i) time, with a general exponential reduction in cost with a Moore's Law halving time of approximately 13-18 months; and ii) the level of commoditization, i.e. the size of the market into which the vendor is selling the computer hardware, determined largely by market demand.

We can estimate the approximate commoditization factor by considering order-of-magnitude estimates (at a fixed epoch of late 2007) for three categories of systems: i) a GPU-based workstation based on video processors with a high degree of commoditization; ii) a high-end CPU-based workstation, with a modest level of commoditization, and iii) the approximately top 10 HEC systems on the TOP 500 list at this epoch, using available (though unquestionably incomplete) public cost data for these systems. We stress that given the rapid evolution in prices, and the wide variability in HEC system costs for different architectures, *these numbers are order-of-magnitude estimates only,* and are quoted only to show the trend in reduced cost with higher levels of commoditization.

Cost per unit TF c_{TF}	Peak performance	$1000 per TF
GPU-based workstation	500 GF	15
CPU-based workstation	100-1000 GF	100
HEC system	100-1000 TF	300

Table 1. Order-of-magnitude estimates of hardware costs at a fixed epoch (end 2007), for three broad levels of component commoditization: i) GPU-based workstations; ii) CPU-based workstations; and iii) HEC systems, as estimated from average cost data as publicly reported (though unquestionably incomplete) for the approximately top 10 systems on the TOP 500 list at this epoch.

Assuming that the commoditization effect is expressed as a factor η in [0,1], increasing to unity for leading-edge HEC systems with the lowest level of commoditization, and incorporating the anticipated Moore's Law evolution in cost per performance, we can derive a crude expression for computing hardware unit costs per TF over the two axes of time and level of commoditization:

$$c_{TF}(t+\Delta c) \sim \eta c_{TF}(t_0) e^{-\frac{(t-t_0)}{\Delta t}\ln(2)} \quad (1.5)$$

where t is time, Δt is the adopted Moore's Law doubling time (1.11 yr from equation [1.4] above), Δc is the construction lead time Δc ~ 1-2 years, $c_{TF}(t_0)$ ~ \$300k/TF, and t_0=2007. With all the caveats noted above, the resulting cost estimates are plotted in Figure 6 rounded to the nearest order-or-magnitude; as expected there is a sharp dependence on how close the required system is to the leading-edge peak performance at the extrapolated epoch (see Figure 4 above).

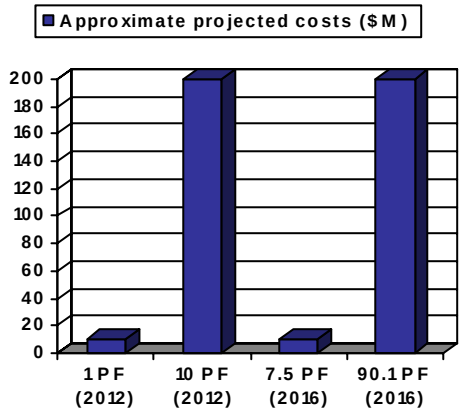

Figure 6. Very approximate estimates of HEC system cost, derived from equation [1.5], rounded to the nearest order-of-magnitude

Power and memory costs have become increasingly important in HEC construction costs, and these trends are expected to continue. Environmental and facility requirements for petascale systems in the next decade are set fundamentally by the number of processors cores and their power and cooling requirements. Machine room floor space of $O(10^4)$ ft^2 will be required with overall power consumption in the range of tens of MW. Chilled-water cooling will likely be needed for many systems. Green innovations will be essential for these facilities, both morally and possibly mandated by legislation. For example, burgeoning data center energy costs in the US prompted Public Law 109-431, mandating a study of rising US computing and data center energy costs by the US Environmental Protection Agency. Their report (EPA 2007) noted that current US data centers consume 61b kWh annually, with an expected doubling by 2011 to a total \$7.4b annual electricity cost. SKA processing facility energy efficiency is likely to be a important design cost driver, and should be factored in to the overall telescope construction cost equation as early as possible.

We estimate approximate annual HEC system operating costs here at ~10-20% of system construction cost.

SOFTWARE COSTS

The question of software costing for the SKA has been considered previously by Kemball & Cornwell (2004), based on a complexity extrapolation relative to software costs for the Atacama Large Millimeter Array (ALMA). This yielded a software cost estimate of approximately 20% of total

SKA construction cost. These estimates remain considerably uncertain however, given that the full SKA design is not yet complete and that system complexity is the fundamental driver of overall software costs. Basic software cost models, such as COCOMO (Boehm 1981), illustrate this scaling with problem size as:

$$\left(\frac{COST}{FTE-months}\right) \sim a\,\beta\left(\frac{Lines\ of\ code}{1000}\right)^{b} \tag{1.6}$$

where a ~ 2.4-3.6 and b ~ 1.05-1.2, depending on the nature and scope of the project (the upper end of this range is applicable to SKA), and β is a correction factor applied in this memo to account for typical under-funding of academic research software projects relative to their commercial counterparts (β~ 0.5-0.7 perhaps). The fact that the exponent b is greater than unity reflects the fact that as software projects increase in size, the number of components and complexity of their interactions increases geometrically. The fact that this is not a linear relation is the fundamental risk factor affecting all large software projects. For the SKA, where digitization is moved much closer to the receptors than in many contemporary telescopes, costs are displaced from electronic hardware to software, constituting an important cost risk factor. We note as an aside that software cost estimation is an established discipline in computer science and informatics, and considerably more detailed cost models are available than the relation presented in equation [1.6], incorporating information of team and process maturity, hardware constraints, and a host of other factors.

At present an accurate software cost estimate for the SKA will require a more detailed analysis and greater finality in the reference design. However, contemporary large survey instruments in the optical under design at present provide independent estimates of overall data management cost, including hardware and software, that may lie in the 25-30% range of their total construction cost. Whether the metric of fractional construction cost transfers directly to SKA is not yet clear, but these complementary, independent estimates confirm that software and data management will be a non-negligible cost for the SKA and an important consideration when making hardware design choices.

We draw attention here also to an important dependency in overall hardware and software cost estimation. Hardware accelerators, such as GPUs, are highly attractive components in the overall SKA computing design due to their high level of commoditization, as noted above, and their significantly lower hardware unit cost as a result. However, it is important to note that GPUs and other hardware accelerators present a significantly more complex programming environment due to the aggressive hardware optimization designed-in for their primary role (e.g. processing graphics primitives) and the cost equation contribution for development with these devices will be significantly higher than conventional development. The issue of programmer productivity and code optimization when programming hardware accelerators is an active area of current computer science research and significant advances may be expected on the time-scale of the SKA.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. AST 0431486. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Boehm, B.W. 1981, *"Software Engineering Economics"*, Prentice-Hall: Englewood Cliffs, NJ.

Cornwell, T.J. 2004, *"EVLA and SKA Computing Costs for Wide-Field Imaging (Revised)"*, EVLA Memo 77, http://www.nrao.edu.

Cornwell, T.J. 2008, *"Computing Challenges for the Square Kilometer Array Pathfinders"* in Deep Surveys of the Radio Universe with SKA Pathfinders, http://ska2008.ivec.org/.

Dongarra, J.J, Luszczek, P., & Petitet, A. 2001, "*The LINPACK Benchmark: Past, Present, and Future",* http://www.netlib.org/utk/people/JackDongarra/PAPERS/hpl.pdf.

EPA 2007, *"Report to Congress on Server and Data Center Energy Efficiency, Public Law 109-431"*, US Environmental Protection Agency (EPA), ENERGY STAR program, http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf.

Golap, K. et al. 2001, *"Parallelization of Wide-Field Imaging in AIPS++"*, in Astronomical Data Analysis and Software Systems X, **238**, 408.

Kemball, A.J., & Cornwell, T.J. 2004, *"A Simple Model of Software Costs for the Square Kilometer Array"*, Experimental Astronomy, **17**, 317.

Kemball, A.J., Crutcher, R.M., & Hasan, R. 2008, *"A Component-Based Framwork for Radio-Astronomical Imaging Software Systems"*, Software: Practice and Experience, **38**, 493.

Lonsdale, C.J., Doeleman, S., & Oberoi, D. 2004, *"Efficient Imaging Strategies for Next-Generation Radio Arrays"*, Experimental Astronomy, **17**, 345.

Moore, G.E. 1965, *"Cramming More Components onto Integrated Circuits"*, Electronics, **38(8)**, 114.

Perley, R., & Clark, B. 2003, *"Scaling Relations for Interferometric Post-Processing"*, EVLA Memo 63, http://www.nrao.edu.

Roberts, D., & Crutcher, R. 1997, *"IMAGER: A Parallel Interface to Spectral Line Processing"*, in Astronomical Data Analysis Software and Systems VI, ASP Conference Series, **125**, 120.

Roberts, D. et al. 1999, *"Status and Future Plans for Parallelization of AIPS++"*, in Astronomical Data Analysis Software and Systems VIII, ASP Conference Series, **172,** 15.