Software Tools Compared To User Education in High Performance Computing

Lev Lafayette
V3 Alliance

The growth in datasets is well recognised due to the increasing ubiquity of information-gathering. In such an environment, unicore desktop applications and traditional file systems are not capable of providing researchers their needs within a reasonable time. However the necessary skillset for high performance computing is not common among researchers.

Two broad methods exist for bringing scientific and high performance computing together; (i) modify the HPC environment to suit the existing skillset or (ii) develop the skillset to match the HPC environment. There has been significant development in the former area, however even the provision of the most user-friendly and modular submission tools remains challenging because HPC requires a degree of understanding of the process. Without the grounded understanding the researcher will be learning (and relearning) applications.

The alternative is to provide a graduated training that provides both the skillset for HPC utilisation but also implicit learning adaptable for future situations. For past several years the Victorian Partnership for Advanced Computing (VPAC), and the successor organisation, V3 Alliance, have conducted a range of training courses designed to bring the capabilities of postgraduate researchers to a level of competence useful for their research. These courses make use of some of the key insights of andragogical education. This strategy has also resulted in significant increases in use on the 'Trifid' cluster.

The Challenge of Processing Increasingly Large Datasets

Under the popular science term 'big data' it is recognised that the volume, velocity, and variety of datasets are subject to significant increases with a rate identified 23% per annum increase in information storage from 1986 to 2007, and a 28% per annum increase in bidirectional telecommunications (Hilbert, López, 2011), due to increases in data collection opportunities and overall network connectivity (number of connections, range of connections, data rates etc). Likewise the benefits of such datasets are becoming well known. In a well-known example, search engine queries submitted to google were used to indicate the presence of influenza in a near real-time manner with a high degree of accuracy when compared to actuals from CDC data (Ginsberg, 2009). However, with these benefits significant problems have also arisen from a technical view on issues relating to storage, curation, and transfer, along with the legal and political issues surrounding data retention practises (De Zwart, Humphreys, van Dissel, 2014).

Future datasets are likely to be even more challenging; the proposed radio telescope, the Square Kilometre Array (SKA), will collect some 14 exabytes of data. This is ten times more data per day than the entire current Internet, which as a result has been described as the "Ultimate Big Data Challenge" (IBM Research, 2013). As one can reasonably expect, the raw data will processed locally into standardised "data cubes and images" as storage of raw daily data will be impractical, even considering likely improvements. Processing of such data is also considered problematic in its own right, as such a machine would most certainly rate at the top of standard supercomputing lists. This has led some (Newman, Tseng, 2011) to suggest distributed cloud-based alternatives. Speaking to Computerworld Australia, the director of the International Centre for Radio Astronomy Research (ICRAR), Professor Peter Quinn, gave the following simple comparison which elucidates the scale of the problem: "This telescope will generate the same amount of data in a day as the entire planet does in a year. We estimate that there will be more data flowing inside the telescope network than the entire internet in 2020." (Barwick, 2011). Of course, this is just one single future big data project by way of example and it can readily assumed that there will be a multitude of others.

Of the range of the aforementioned challenges concerning storage, curation, transfer, processing etc., only the issue of processing is directly addressed here, and indeed only one aspect of the processing challenge. It is useful to consider large dataset issues as a workflow; a decision is made to collect data, data collection points are distributed, the data is stored and curated, the data is processed, and an analysis is carried out, leading to improvements in knowledge and its applications under an evidence-based framework. The processing aspect of big datasets will typically be carried out by high performance computing systems. Distributed or loosely coupled computational systems, such as various grid computing architectures (e.g., SETI@home, folding@home) is arguably not the solution for large datasets except in cases where the dataset can be broken up, in which case it is a large collection of smaller datasets, bound together by logical interfaces. Whether or not this still constitutes a large dataset is moot (Beynon et.al., 2011), however much of the discussion carried out here can be applied equally with distributed or high performance computing systems.

The reasons for the need of distributed or high performance computing systems is well-known; unicore desktop applications are simply not capable of providing researchers their needs within a reasonable time (Jacobs, 2009). Whether conducted as task parallelism, where multiple processing units carried out tasks, or data parallelism, where different datasets are distributed simultaneously for processing, a theoretical speedup is available equal to the multiple of processing units dedicated to the task or dataset. Of course, this theoretical speedup is just that, as all forms of paralellisation have some serial component (Amdhal's Law) even if by increasing the quantity of tasks or the size of the dataset can reduce the serial proportion (Gustafon-Barsis' Law). Scalable task and data parallelism is thus a necessity as datasets increase in prevalence and size. Due to relative architectural simplicity and cost considerations, cluster and grid computing systems are the most common implementations of processor parallelism.

High Performance Computing As a Subset of Scientific Computing

When maximising processing capacity the optimal user interface is the command-line shell, short of kernel-level programming itself. As the shell and applications interacts with the kernel, which itself is the interface layer between software and hardware, a very high level of resource efficiency is achieved. For this reason the command-line interface remains the dominant user interface in high performance computing, at least evident of a review of supersomputing metrics, with visualisation of the results of these computational tasks more typically available on desktop systems. However, the requisite skills - such as the Linux command line interface, PBS job submission, shell scripting, regular expressions - are not commonly taught at the university environment. It is almost as if it is expected, rather unfairly, that researchers will acquire them at some stage through their career through a combination of osmosis and necessity. This is optimistic at best when one considers that due to the prevalence of monopoly operating systems and the dominance of the graphic user interface on desktop systems, the general level of skills required for HPC systems among untrained researchers is lower than what it was twenty years ago.

A review of undergraduate course offerings from computer science departments from the Group of Eight Australian Universities illustrates the issue. The Australian National University offers a general Python programming course (COMP1730) for scientists, as does the University of Queensland (SCIE1000), although in this case the programming is a component of a general overview of a broad range of mathematical, analytical, conceptual, and computational tools used in science. The University of Adelaide offers a scientific computing (COMPSCI1102) which is almost entirely based around MATLAB. The University of Sydney offers one generic course, (INFO1003) in information technology which includes general purpose application skills in spreadsheets and databases as well as some basic website coding (html, javascript, and ajax). Monash University, the University of Melbourne, the University of New South Wales, and the University of Western Australia has no relevant undergraduate units.

The purpose of such a review of course is not meant as criticism of the courses offered, but rather to illustrate a challenging higher educational issue. The content level of basic training for high performance computing is mostly at the undergraduate level. However, in most disciplines the available curriculum space at this level is heavily contested. Thus where high performance or distributed computing is taught, it is not done so in the context of the application of other sciences utilising the technology but rather as a specialist subject of those advanced in the computer science major path (e.g., UWA's postgraduate CITS5507 High Performance Computing), which is far in advance of what most scientists need.

In this respect a common conflation of high performance computing with scientific computing is quite inaccurate. Past emphasis on high performance computing has given priority to computational metrics (e.g., number of floating point operations, number of cores etc), rather than the utilisation by the scientific community or the outputs generated (Wilson, 2008). Instead scientific computing should be reflects whatever scientists are using computing systems for within their scientific endeavour and if HPC systems are only used by a relatively small percentage of the scientific community then it is unreasonable to consider the two terms as analogous. With this view the software carpentry project (Wilson, 2006) makes use of immediate and practical lessons for computational tools such as the UNIX shell, version control (e.g., Subversion), build tools (GNU Make)., scripting (Python), and debugging. More recent courses have added SQL databases and mathematical programming using R.

This represents one approach to the distinction between scientific computing and high performance computing. It recognises that, at best, HPC computing can be considered a small subset of scientific computing. However such a situation cannot remain if research institutions are to continue with the challenge of larger datasets. Whilst analysis of processed data will remain viable with unicore desktop applications, it is implausible to suggest that this will be sufficient in many cases in the future. Viewed in this manner scientific computing *should* be analogous to high performance computing and the practical task is to raise the usage of HPC and indeed must be if an research organisation is to remain a viable entity as existing research shows a very strong correlation between provision and research output (Apon et. al, 2010) which, if expectations on datasets is confirmed, will increase over time. This suggests the need for an additional equivalent of "software carpentry" in high performance and distributed computing.

Limits of User-Friendly Interfaces for HPC Adoption

If there is a need for an increased usage of high performance computing by research scientists and existing skill levels are low, then two general options exist (i) modify the HPC environment to suit the existing skillset or (ii) develop the skillset to match the HPC environment. There has been significant development in the former area, especially championed by software developers and management who want to simplify job submission tools. Well-known examples include xpbs, grid computing interfaces such as the former Grisu project, distributed computing installers such as folding@home, web portals such as the Workflow Management System of BeesyCluster (Czarnul, 2014) or even from the direction of applications developing parallel capacity, such as Matlab's DCS and parallel computing toolbox. An Australian example which is reviewed is the implementation of Monash eResearch's STRUDEL (ScienTific Remote User DEsktop Launcher) which has been shown to usability and uptake of CQUniversity's High Performance Computing (HPC) facilities (Bell, Hines, 2014).

Most attempts to implement a GIU alternative or 'skin' to HPC interfaces have been less than successful. This is generally because HPC systems and the various options available for job submission are not designed for user friendliness, but rather for efficiency. In order to utilise HPC successfully, an understanding of the process is required which is not something which is immediately discernible from existing tools. Applications such as xpbs and Grisu, for example, require the user to set the usual resource requests (e.g., memory, system units, processors, walltime etc) as well as organising their remote connection. In other words, the GUI does not make the relatively complex task any less complex, which it can and does succeed in doing in other application environments. In the main, the provision of user-friendly graphic user interfaces for high performance computing remains inevitably unsuccessful because the environment requires a deeper level of understanding of the process involved, whereas in other areas the GUI is able to simplify both the process and move from understanding to a more intuitive response.

An interesting alternative and counter-example is STRUDEL at CQUniversity, derived from the the Multi-modal Australian ScienceS Imaging and Visualisation Environment (MASSIVE) desktop Characterisation Virtual Laboratory launcher from Monash University and the Australian Synchrotron. It shows a mediating point between those most familiar with applications that already have a graphic user interface and the additional complexity that is achieved through command-line knowledge. STRUDEL provides a virtual desktop, similar to cloud environments, but with the performance improvements achieved by an HPC system and without user maintenance issues. In the particular instance of CQUniversity, a default configuration was established that allow for such applications to run in an interactive mode, in addition to significant automation in connectivity such as checking to see whether an existing VNC desktop is already running and configuring an SSH tunnel. This level of automation allows users to have a transitional stage between the desktop application and the HPC application as structured knowledge and also as an opportunity to witness the improvements in job speed. As the user becomes more familiar with the options available then they will be able to approach the new options with a higher degree of self-efficacy.

However, even the provision of the most user-friendly and modular submission tools remains limited to those tasks which are subject to effective simplification. Beyond this limited range educational intervention is required for two major reasons. The first relates to the well-established educational principle of a zone of proximal development, an educational stage between where a learner cannot carry out a task and where a learner can carry out a task. In this zone, the learner can carry out the task with the assistance of the educator, whose role is to bring the learner to a point where they can carry out the task independently. Because high performance computer tasks are strongly bound to context whilst it is theoretically possible to provide an automated user interface that helps in this regard (perhaps rather like a suggestive spell checker), there has not been much development to date.

The second reason relates to disciplinary-based learning style advantages. Whilst popular (and even some educational) theory presupposes that individual learning styles have educational outcomes, the evidence for this is questionable at best. Instead, it would appear that some individuals may have style preferences, but effective educational outcomes are more tied to disciplinary and sub-disciplinary environments. A promising example in high performance computing (Zarestky, J., Bangerth, W., 2014) is the use of a "flipped classroom" and project-based approach, whereby learners take up the learning experience in a connectivist and collaborative manner and the educator becomes more of a facilitator and guide (Hughes, H. 2012). Such an approach is difficult to introduce through a user interface as it relies on the motivation generated through social interaction.

Postgraduate Education and Andragogical Techniques: The V3 Alliance Experience

The Victorian Partnership for Advanced Computing (VPAC) is a not-for-profit registered research agency established in 2000 by a consortium of Victorian Universities. Research institutions required the performance of many core clustered computing systems, but lacked the finances and expertise to individually purchase their own system. Collectively however they could do so, and through a proportional of cycles equal to contribution, VPAC provided several systems over the years, including two systems that were in the Top 500. By the mid-2000s the organisation had introduced a number of training courses both in general HPC usage and in MPI programming. It is fair to describe these courses as being technically correct, but lacked the integrated education that was required. These courses were conducted by people with excellent technical knowledge, but whose education and training capabilities were more a case of natural ability rather than systemic knowledge.

More recently VPAC, and the successor organisation following a merger with the Victorian eResearch Initiation (VeRSI), V3 Alliance, have changed their training courses to make use of some of the key insights of andragogical techniques in postgraduate education, addressing learners, content, and delivery. Even more than other adult learners, postgraduates especially tend towards more voluntaristic engagement, rather than compulsory requirements. They are more self-motivated and orientated towards independent learning, and have a greater range of experiential resources. Certainly for the case of training in high performance computing, they are usually attending because they have specific problems relevant to their research that they want solved. The learner is autonomous and usually self-determining with internal rather than external motivations. There is a greater level of equality between learner and educator, and greater opportunity for the learner to express knowledge from their own experiences, and with the flexibility to engage in study directions suited to their own practical tasks and interests.

Content in such an environment must be provided in a structured manner which suggests that understanding is achieved when knowledge exists in an embedded organised context with other knowledge which is differentiated from mere recognition. Thus content cannot be expressed in just facts alone, even an intergrated set of facts, but rather facts that are backed up with grounded reasons. The content provides the learner a conceptual organising structure for the body of knowledge which allows the learner to elaborate their knowledge into new constructions. Structured knowledge thus adds to learner self-efficacy, their internal recognition of the capacities in an allocated task (a performance outcome), but also their confidence in carrying out future tasks. Content is thus provided initially in the most necessary task - knowledge of the Linux command-line interface, then on to PBS job submission and review commands, and on to range of job submission tasks across multiple disciplines and with different submission configurations. More advanced courses extend the command-line knowledge to include scripting, the incorporation of scripting into job submission, and job submission using arrays, dependencies, and interactive jobs, with a further course in MPI programming from the basic compilation process and review of simple examples, to an exploration of the essential routines, and then elaboration into the more task-specific examples, concluding with various tools for profiling and debugging MPI code.

Finally, there have been changes to the delivery of the material. Noting that learning styles are biased in disciplinary and subdisciplinary form, and that there are individual learning style preferences, the content is presented in a multimedia form. For reference and for the opportunity to work through the material, the training provided in a hardcopy manual, ordered according to the structured content. The introduction to the material is presented in a lecture form, explaining the importance of high performance computing, an overview of the architecture, the reasons why particular operating systems and user interfaces are used, an explanation of system architecture, and limitations of parallelisation. The objective here is to provided grounded reasons and motivation to the learner, so that they have an understanding of the content, rather than just the ability to recall the facts of the content. This emphasis on understanding is continued through the extensive hands-on exercises, which the learner both carries out individually, and with small-group assistance, and with the exercises repeated by the trainer through overhead projection, noting the importance of visualisation technology to illustrate concepts. Each exercise is relatively small, providing the opportunity for immediate feedback and guided, taking into account proximal development for the learners, with each activity providing a foundation for the next. Throughout the days, extensive opportunities are provided for dialogue with the learners to express their issues, examples, and experiences, in preference to verbatim lecture.

Results

The validity of the approach can be indicated by changes in usage. An excellent starting point for this consideration for V3 Alliance is the end of 2012, when the former cluster (Tango) was in the process of being shut-down (final shutdown was not until April 12 2013), a new cluster has just been introduced (Trifid, November 29, 2014), and most importantly, the course content and material had just undergone a thorough revision to align itself in accord to the established methods for adult and advanced education, and was notably increased in the number of classes conducted.

From the August 2012 to December 2014, fifty course days were conducted at V3 Alliance premises for the Trifid cluster. This cluster is equally owned by La Trobe University and the Royal Melbourne Institute of Technology (EDIT Day of the Trifid), with a smaller share owned by the V3 Alliance itself. Due to proximity, the overwhelming majority of attendees at these courses were from RMIT. A comparison between enrolments and cluster usage indicates a strong correlation; more training in HPC computer results in greater cluster usage.

Table 1.
Trifid Usage (CPU Hours) to December 31st 2014

Year	RMIT	La Trobe	Cluster
2012	1,729,837h	1,719,554h	Tango
2013	8,108,695h	3,301,052h	Trifid
2014	9,760,919h	4,964,297h	Trifid

Table 2.
Trifid Course Enrolments to December 31st 2014

RMIT enrolments	229 (62.74%)
La Trobe enrolments	29 (7.96%)
University of Melbourne	38 (10.41%)
DEPI/DPI	28 (7.67%)
Swinburne University	16 (4.38%)
Deakin University	15 (4.11%)
Victorian Univeristy	7 (1.92%)
Commercial	3 (0.82%)
Total enrolments	365

(NB: The University of Melbourne has its own cluster, Edward, operated by V3 Alliance. The Department of Environment and Primary Industries also has a cluster that was installed by V3 Alliance).

With positive results further opportunities present themselves. Most obvious is increased enrolments from other institutions. For 2015 this has already began with La Trobe University taking up some 57 places in opening months, 15 at the University of Western Australia, and another 130 at the University of Sydney with their new HPC system using PBSPro as a job scheduler (EDIT Sydney press release), whereas existing classes had concentrated on Moab and Torque as a job scheduler and resource manager respectively. There is also demand for elaboration on the courses offered, especially for scientific programming and utility tools, such as those offered by the Software Carpentry group. Further it is hoped to significantly improve our post-course mentoring and introduce more project-based hackathons for high performance computing.

Conclusions and Opportunities

The pressure on research institutions to process ever-increasingly large datasets shows no signs of abetting. Research institutions that fall behind on their capacity in this regard will face challenges to their relevance and survival. In order to process these datasets, not only are more powerful high-performance computing facilities required but so is the need to increase the competency of researchers. Only a portion of the HPC job submission processes can be simplified and made intuitive, and thus only a portion is particularly well-suited to GUI tools. For most of the procedure a deeper level of trained understanding in command-line tools is requisite and this knowledge is best transferred through the adoption of a range of particular androgogical techniques best suited to intellect of the participants. The empirical results from of two years of coursework on the Trifid cluster, and a comparison between two institutions that began on equivalent terms but with divergent training enrolments is indicative of the success of training programme which can serve as a model for other institutions which seek to emulate its success.

A particularly challenging issue, especially considering the current financial and funding environment for higher education in Australia (The Commonwealth of Australia, 2014), is the provision for the necessary skill training and mentoring for research institutions to make of high performance computing. A comparison can be made with the European institutions who, keenly aware of the economics of socially positive externalities, have seen their advanced computing partnership make a powerful case of the provision of high performance computing for scientific research (Guest, 2012). This case also raises the issues of user support and training, which is defined as including user support, researcher awareness of facilities, instructional courses, optimisation for data processing, and development. In order to derive similar positive externalities and their own institutional benefit, the Australian higher education sector should also engage in such training programs. Financial pressures may enforce cooperation and collaboration with specialist organisations, such as the V3 Alliance, with funding derived from the National Collaborative Research Infrastructure Strategy research facilities, a $150 million commitment for Australian researchers with access to national, strategically important research facilities.

References

Apon A., Ahalt S., Dantuluri V, et. al., (2010) "*High Performance Coputing Instrumentation and Research Productivity in U.S. Universities*", Journal of Information Technology Impact, Vol 10, No 2, p87-98

Barwick, H., (2011), "*SKA telescope to generate more data than entire Internet in 2020*", Computerworld Australia

Bell J., Hines C., (2014) "*Improved High Performance Computing usability and uptake through the utilisation of Remote Desktops*", eResearch Australasia

Beynon, M.D., Kurc, T., Catalyurek, U., Chang C., Sussman A., Saltz J., (2011) "*Distributed processing of very large datasets with DataCutter*", Parallel Computing Vol 27 No 11, p1457-1478

The Commonwealth of Australia (2014), "*Budget 2014-15: Higher Education*"

Czarnul, P., (2014) "*A Workflow Application for Parallel Processing of Big Data from an Internet Portal*", Procedia Computer Science, ICCS 2014. 14th International Conference on Computational Science, Volume 29, p499-508

De Zwart, M., Humphreys, S., & van Dissel, B. (2014). "*Surveillance, big data and democracy: Lessons for Australia from the US and UK*", The University of New South Wales Law Journal, Vol. 37, No. 2

Ginsberg, J., Mohebbi, M., Patel, R.S.., et. al., (2009), "*Detecting influenza epidemics using search engine query data*". Nature 457 (7232): 1012-1014. doi:10.1038/nature07634

Guest, M (2012) "*The Scientific Case for High Performance in Europe 2012-2020 : From Petascale to Exascale*", Partnership for Advanced Computing in Europe (PRACE)

Hilbert, M., López, P. (2011) "*The World's Technological Capacity to Store, Communicate, and Compute Information*", Science Vol. 332 no. 6025, pp. 60-65

Hughes, H. (2012). "*Introduction to Flipping the College Classroom*", in T. Amiel & B. Wilson (Eds.), Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2012 (pp. 2434-2438). AACE.

IBM Research (2013), "*Ultimate Big Data Challenge*"

Jacobs A, (2009) "*The Pathologies of Big Data*", Queue, Association for Computer Machinery

Newman, R., Tseng J., "*SKA Memo 134 : Cloud Computing and the Square Kilometre Array*", Oxford University

Wilson, G. (2008), "*High Performance Computing Considered Harmful*", 22nd International Symposium on High Performance Computing Systems and Applications, 2008

Wilson, G. (2006). "*Software Carpentry*", Computing in Science & Engineering, 8, 66.

Zarestky, J., & Bangerth, W. (2014). "*Teaching High Performance Computing: Lessons from a flipped classroom, project-based course on finite element methods*", in Proceedings of the Workshop on Education for High-Performance Computing (pp. 34-41). IEEE Press.

Corresponding author: Lev Lafayette**,** lev@v3.org.au

Please cite as: Lafayette, L. (2015, May). *Software Tools Compared To User Education in High Performance Computing*. Paper presented at the THETA 2015 Conference, Gold Coast, Australia.

Note: All published papers are refereed, having undergone a single-blind peer-review process.