?

Bare Demo of IEEEtran.cls for IEEE Journals

Ozlem Erkilic
Pawsey Supercomputing Centre Curtin University
Perth, Australia
Email:ozlem.erkilic@student.curtin.edu.au

***Abstract*—The best way to understand how to compute a task based on some resources, constraints and goals on advanced computing systems is to work through sample codes. A lot of expert users browse through the Internet to find these examples that they can slightly modify to achieve the results they expect. However, this can be very diffucult for some users who are new to this environment as there are always so many of these examples which are not always what they specifically need. This brings out the challenge of having to modify the example to suit their requirements on a particular computing resource, mainly the High Performance Computing systems (HPC) since the modified examples mostly do not work due to HPC systems having different set ups. Therefore, it is hard to tell what the issue is for inexperienced users. Quite often, the users maynot be able to tell if the example is broken or if the example is correct, but maybe the code requires some adjustments. This is a very common issue encountered by the users of Pawsey Supercomputing Centre. For this reason, this paper provides solutions to how to run and submit examples on different resources such as supercomputers of Pawsey Supercomputing Centre through the use of a simple tool ”getexample”. The ”getexample” is an effective tool that is developed for the supercomputers of Pawsey that are dependent on a command line interface and aims to help their users learn to how to run code mainly parallel programming examples, submit these examples to different operating systems of Pawsey such as Magnus, Zeus and Zythos. Although, the getexample is limited to these HPC resources, it has the potential to be expanded to other resources and interfaces.**

***Index Terms*—IEEE, IEEEtran, journal, LaTeX, paper, template.**

I. Introduction

EVEN though there are many sample codes on search engines for programming purposes, it is usually very rare that these codes work at the first try on different high performance computing systems without modifying them. This is mainly due to each advanced computing system being built up differently from each other to perform specific tasks for the needs of the users. At Pawsey Supercomputing Centre, there are various HPC resources where each of them are set up with slight variations from each other. Therefore, this becomes very challanging for the users, especially the beginners to change the example codes to display a working task on Pawsey's resources as there are many differences between each resources with many constraints. The most common differences between their systems are as follow:

1) Contrasting operating systems 2) Divergent program environments 3) Non-idendical compiler options with varying commands 4) Various compiler flags and wrappers for varying compiler options 5) Module systems and paths that are different 6) Disparate module naming conventions 7) Various library versions 8) Dissimilar personal scratch, group and home directories 9) Different scheduling policies

For this reason, when the user runs a sample code found from the Internet for each supercomputers (Magnus, Zeus and Zythos), the code fails to compile as expected and hence, does not run as each individual system has specific operating system and commands set up for them. To rectify these problems, some HPC resources provide their users websites with working examples that aim to assist them how to carry out their tasks step by step. However, sometimes the commands in these examples can be outdated due to the updates in compilers and the operating systems. They also can be challenging to follow and perform on the real systems for the users who are not very experienced with these resources. Although, one may find a well-written bash script to run these systems, may not know how to make this script executable by changing the permissions using the chmod commands. Furthermore, a user may use a sample source code such as a basic MPI code which includes some mpi libraries to perform a particular job on these supercomputers but may fail when compiled due to these libraries being no longer valid or upgraded to a different version. Or one may complete their task, but may not know in which filesystem to store their results, for example in scratch or their group because some supercomputers have policies on how long to keep the data on certain filesystems such as scratch. If they are not moved from that specific file system within a certain amount of time, the results get deleted and thus, the user loses their work. This is a major difficulty for the users, especially researchers and scientist since it takes very long time to collect their data and vital for their researches. It is also very crucial that these are stored correctly within these sources.

In order to minimise these problems, the getexample was developed to suplly examples which do not require further editing or modification and works when the executable is run. Thus, it aims to teach the users of Pawsey Supercomputing Centre how to run and submit tasks on the supercomputers without encountering issues. The rest of this paper explains how the getexample tool works.

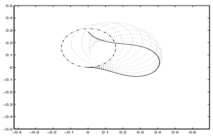

Fig. 1. elastic data set 1

A. Magnus

Magnus is described as a Cray X40 supercomputer that consists of many nodes that a$ofthemhas2socketsandeachofthesehas12cores.Magnusisalsospecifiedto$ the 1488 nodes.On Magnus, jobs run on the back-end of the system with the help of SLURM and ALPS $submittedtothequeuesystemonthefront-endfromthesbatchcommand.Whenitru$ which are the login nodes of Magnus. The aprun keeps running on these login nodes $jobalsocompletes.Asmentionedpreviously,thefocusofthegetexampletoolwasnotonlytoprovide$ on their scratch directories. Therefore, whenever the batch job was submitted to M$completed,theresultswerecarriedtothegroupdirectory.Intheend,thescract$ programming examples such as MPI, OpenMP and hybrid codes which a combination of O$Thesourcecodesusedfortheseexampleswerewritternincor Fortranandwerebas$ Each example was displayed in different environments on Magnus such as GNU, Intel $optionsofftn,mpif90andcc.Whenusingtheseenvironments,itwasimportantto$ codes and running them as each environment has specific compiler commands with dis

Subsection text here.

II. Conclusion

The conclusion goes here.

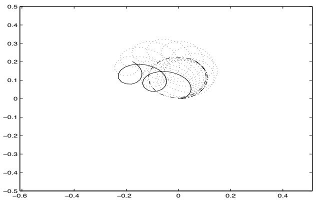

Fig. 2. elastic data set 2

Appendix A Proof of the First Zonklar Equation

Appendix one text goes here.

Appendix B

Appendix two text goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

PLACE PHOTO HERE

Ralph Bording Biography text here.

Jane Doe Biography text here.