

Joshua Nasman

Summer Research Outline

PetaApps Project

One week timetable:

- Ensure code so far operates as intended
 - Testing – Rerunning tests as described in semester paper
 - Verifying against runs on Levi
 - Writing up reasons why results may not be as good as desired
 - Commenting the code sufficiently
- Find ways to make collective calls be more accurate (likely will begin in the week timetable and go beyond).
 - MPI calls are taking longer within PHASTA than in the MPI call tester
 - Please see Appendix A. for a further discussion of this
- Once these goals are completed the one to one emulator should be able to be put aside for a while

One month timetable:

- The second goal in the first section should definitely be done by this point.
- During this time it should be investigated how using threading will interact with MPI. By the end of the month timetable toy programs should be completed which demonstrate that collective calls can be done across MPI processes with more than MPI process per node of a system and that the timings can be nearly as accurate as that of the one to one emulator.
 - This may or may not include support for doing this on the Blue Gene/ L

Summer timetable:

- The research into how MPI processes can be done for multiple MPI processes per node will be extended to include PHASTA.
 - Investigate if this necessitates a notion of GVT (Global Virtual Time).
 - Figure out if this can be done using round robin or must have a more advanced scheduling algorithm
 - i.e. If some processes have completed their work and are simply waiting for an MPI call, do we still want them eating up CPU time? Could be dependent upon if we care if the simulator finishes quickly or not
 - When MPI calls occur is it necessary to preemptively switch processes?
- Investigate (the more simple) ways to make the emulator more accurate

- Go beyond a simple ratio of CPU speeds. Use analysis of the size of cache as well as the amount of memory PHASTA is using to get a more accurate ratio of how much more quickly computation will occur

Beyond:

- Actual model the network of a machine (as opposed to just lookup tables) to improve emulation of the network
 - This will specifically give more accurate results as far as the effects of congestion in a network.
- Simulate actual cache performance

Other ideas for the future:

- Given that whatever stage the emulator is at it can trivially collect a type of profiling data for each process, it could be useful to actually get out this data (either for a simulated or even simply the actual system) and put it into a visual form useful for the analysis of PHASTA
- Steps:
 - As discussed before find a way of segmenting the code (it may ultimately be necessary to insert a few dummy calls into PHASTA to separate more effectively).
 - Using this information give the emulator information to find information out about the loops/overall flow of the program
 - Have a text/binary file outputted containing this information
 - On a separate (non-supercomputer system) have a program which creates a graphical representation of this data showing computation vs. io (blue and red) with various shades indicating where there are bottlenecks

Appendix A:

It has been noted that MPI calls take longer in PHASTA than in a simple program used to test times. This should not have been surprising considering collective calls always are forced to wait for the slowest process to arrive. For this reason MPI_Barrier, MPI_Waitall, and MPI_Allreduce will take varying times to complete depending on where various processes are in their execution. In a fully balanced program, this might not be much of an issue, but in the case of PHASTA there are often imbalances, particularly when it comes to communication.

When computation is done in PHASTA to find either the LHS or the RHS, the partitions are often imbalanced. This is a result of partitions often being adjacent to different numbers of other partitions. (If it's useful I'll look in my notes later to find the appropriate methods where this occurs). Also, for each shared part between partitions there is a master and a slave. What this means is that only one does the computation for the shared part and sends it back to the other. As the number of neighboring partitions is not forced to be even, there are often also imbalances in the number of sends/receives per partition.

As much as I can see figuring out how to get more accurate data for the calls is not doable by using information known about PHASTA, it is simply useful in diagnosing the problem. It would seem the way to get more accurate MPI results would be by forcing the program to wait in the collective calls for the rest to get there (presumably using an MPI_Barrier). The time waited would then be added to the looked up time of the particular call (Barrier, Waitall or allreduce). After which the time to actually compute the barrier would have to be subtracted. I have not thought through all the details of this yet, but it seems it may be a viable approach.

Appendix B: (Questions of devotion/excitement about project)

Questions have been voiced as to the excitement about this particular project and the reasons for the rather slow progress to this point. I will take this question in several parts.

Much of the reasons initial progress on the project was slow was a difficulty in getting acquainted with the project. In order for the emulator to make any headway it was necessary to deal with a few difficulties which were not always easy to get quick answers to. These included getting PHASTA to compile (with slightly different instructions for each machine). Figuring out how to get a wrapper to compile against PHASTA. Discovering how wrappers would work for both C and Fortran. While many of these tasks seem rather trivial in retrospect, it was difficult to find time to get fully engaged in a project while learning many of the idiosyncrasies of working on a large program designed for supercomputer use and not having officemates to struggle through things through.

I anticipate it should be easier to make headway when full days are available to work on the project without the interruption of schoolwork. I would add that although I'm not particularly excited about the debugging and investigation of anomalous results that currently has to take place the goals of the project do excite me. As you can tell, learning the specifics of a CFD solver does not particularly interest me, but if I can see where it fits into making progress, I assure you, it will not stand in the way.

I do wish to continue on this project and feel it might be a useful exercise to evaluate 'enthusiasm' once again later in the summer after having a chance to more fully engage the project without distractions. I appreciate any feedback on this as far as what are good and attainable goals, flaws in the plan, or thoughts as far as ways to accomplish goals. One thing that I would ask is for more frequent checks as far as what making good progress looks like.