Exercise 2 - PPLS
Spring 2012

Josh Reese
s1129936

March 28, 2012

BSP

The Bulk Synchronous Parallel (BSP) model is a model for parallel programming that allows the developer to abstract themselves from the specific parallel architecture they are using [1]. In other words this model provides a programming framework for which software can be developed and run on shared memory systems, message passing systems, and various other types of parallel architectures. BSP claims three important fundamental properties which are: simple to write, architecture independence, and the performance of a program is *predictable* [1]. To accomplish this the model looks at a program as a whole, rather than as individual processes, and determines what they call the *bulk* properties of a program. To simplify and make this possible BSP makes no attempt to use locality as a performance optimization. This means that concepts such as cache efficiency and data locality play no role in the BSP model.

The BSP programming style is broken into three main phases (called a *superstep*): local computations (values in each processes local memory), communication between processes, and a synchronization barrier [1]. In this style the interaction between processes occurs during the communication phase and all the processes are synchronized on the barrier at the end of the superstep.

To determine the performance cost BSP uses four parameters: h, g, l, and w. In every communication phase each process will be sending and receiving a fixed number of messages the maximum of which is represented by h and known as an h-relation. This term also has the size of the message folded into it. The parameter g represents the physical ability of the network to deliver messages. This is defined such that it takes time hg to deliver an h-relation. The value of l is determined empirically and is taken to be the cost of the barrier. The last parameter, w, is the time to process a local computation. Putting this all together yields:

$$cost\ of\ a\ superstep\ = \overset{MAX}{processes}\ w_i + \overset{MAX}{processes}\ h_i g\ +\ l$$

Where i is the range over all processes [1].

When considering *pipelined* parallel computations BSP seems to have a few issues. The main concern this model appears to have with pipelined processing is that it is limited by its barrier stage. In a standard pipeline computation whenever a process is finished with its data it is free to send data, receive data, and start working on any new data whenever it is ready. However, in the BSP model nobody is free to do anything with its received data until the last process has finished and thus the superstep has passed the barrier. This limits the pipeline process because each process in the BSP model will need to have a balanced workload otherwise the pattern will breakdown because all the processes will be waiting on one or more slow processes. The concept inherent in BSP in the global barrier breaks one of the main goals of pipelined computations which is that each process is basically an atomic unit which only depends on its predecessor.

Similarly, the *bag of tasks* computations are likely to cause some issues in the

BSP model. Since this model relies on communications between one process and all the other processes the way in which the BSP model handles communication will cause a serious problem for this pattern. Since all the processes will be trying to communicate with a single farmer there will likely be large floods of communication going to one process at one time. Conversely, if the farmer has received a large number of messages that process will likely have to send out a large amount of message directly afterwards. This puts heavy amounts of communication on one node in the process pool. In addition to this the BSP model breaks the fundamental concept of a bag of tasks which is that the workers should be completely independent from one another. The only similarity workers should have is that they communicate with the farmer. However, the BSP model forces the workers, and farmer, into sending messages simultaneously, and working in tandem as opposed to independently.

Bibliography

[1] D. B. Skillicorn, Jonathan M. D. Hill, and W. F. Mccoll. Questions and answers about bsp, 1996.