A. V. Gerbessiotis
CIS 485/Spring 1999
Handout 10
LogP Model

The LogP Model

The LogP model (Culler, Karp, Patterson, Sahay, Schauser, Santos, Subramonian and von Eicken, 1993) has also been suggested as a realistic model for the design of parallel algorithms that work predictably well on a wide range of parallel machines. It models a parallel machine as a distributed memory multiprocessor in which processors communicate by point-to-point messages. It is an *asynchronous* model that does not enforce synchronization of the processors, as the BSP model does. If processor synchronization is required in a program, the programmer must provide it. A parallel machine under the LogP model can be characterized by the following tuple (p, L, g, o) of parameters. Each parameter is explained in more detail below (note that LogP may use the same notation for some of its parameters as the BSP model but these may have different meaning).

p: The number of processor/memory components (as in the BSP model).

L: an upper bound on the *latency* or delay of the machine, incurred while communicating a message of very small size (one or a small number of words) from a source to a destination component. In other words, such messages are delivered by the router within time L.

o: the *overhead*, defined as the length of time that a processor is engaged in the transmission or receipt of each message (i.e the time required for the submission of a message to the router or acquisition of a received message from it).

g: the *gap*, defined as the minimum time interval between consecutive message transmissions or consecutive message receptions at a processor. $1/g$ gives the bandwidth of the system per processor.

In addition, the communication capacity of the system is limited. This means that at most L/g messages can originate or be destined to any processor at any time. If a processor needs to transmit a message that would violate this condition, it stalls until this transmission is possible (i.e. there is some available capacity). Writing under the LogP model into a remote memory location takes time $L + 2o$.

Although BSP and LogP are similarly powered models (one can simulate a BSP computation on the LogP and the other way around), the BSP model seems to be more usable as a theoretical model for the design and analysis of parallel algorithms and as a programming paradigm for writing parallel software that is scalable and transportable (portable and efficient among a variety of hardware platforms).

The importance of BSP is that it introduces a new abstraction for communication in terms of the BSP parameters L and g so that considerations in programming parallel machines move from a local (detailed) level to a global (abstract) one. The two parameters abstract all communication and synchronization issues related to parallel computing and allow the design of software (a unifying property) that work on any machine independent of the underlying architecture (eg. shared memory vs distributed memory).

Under the BSP model, an algorithm designer describes the performance of a parallel algorithm in terms of p, L, g and problem size n. A collection of algorithms that solves a given problem can then be formed with varying performance characteristics (for example, one algorithm may be suitable for machines with small L, another for machines with small g or n, and so on). For a given machine whose BSP parameters are known or are measurable, that algorithm is chosen from the collection whose performance (computation and communication) on the particular machine is the best available.