MPPs versus Clusters

Charles L. Seitz
Myricom, Inc.

Abstract

In coming years, if not already, the parallel-processing community can expect to hear regularly from MPP advocates and cluster advocates about why their approach is better. Either pitch is apt to be a hard sell: hard to sell to an informed audience or reader, and dull. The attempt to distinguish between MPPs and clusters is in some cases an empty subject. By the term “cluster,” I mean a group of high-performance computers—"commodity" PCs, workstations, or SMPs—connected by a fast network, and employing software layers for low-latency message handling and a standard interface for multiple-process application programs. Indeed, today's multicomputers (distributed-memory MPPs) fit this definition of a cluster except for being packaged in a cabinet rather than distributed across a room or building.

MPP bigots can be expected to present positive arguments of tight coupling and the virtues of nodes employing processor types not used in commodity computers. They may also present negative arguments of the limits imposed eventually by the speed-of-light delays in clusters. They will be right on all points, at least eventually. Cluster bigots can be expected to present different arguments involving the limited size of the MPP market in comparison with the PC market, and the resulting differences in investment, technology-insertion schedule, and performance per unit cost. They will also be right on all points, at least empirically, and will have persuasive benchmarks to prove these points.

The obvious trouble with blind advocacy is that it distracts from the opportunities for the parallel-processing community to embrace both approaches, and thereby to extend the applicability of parallel processing. In certain application domains, such as military systems, it becoming commonplace for people to build systems composed of specialized, “embedded” MPPs clustered together with conventional computers. On the software side, a number of streamlined, message-handling, software layers, such as the University of California, Berkeley, Active Messages (AM) layer and the University of Illinois, Urbana-Champaign, Fast Messages (FM) layer, operate interchangeably on both clusters and MPPs. At the level of application programming and libraries, standard application-programming interfaces such as MPI are likewise portable between MPPs and clusters.

There are many other needs and opportunities for research that will unify rather than divide MPPs and clusters. Let me join you in hoping that this is the last talk that you will have to endure on MPPs “versus” clusters.