As the price/performance ratio of parallel computers continues to fall many applications developers such as scientist and engineers with large computational problems are looking to non von Neuman computer technology to provide powerful throughput platforms. While the hardware technology behind such platforms are well matured the software environments do not provide the functionality necessary to allow ordinary applications developers to utilize them efficiently. One particular problem, which requires the development of new and innovative techniques, is the mapping of the work of a potentially concurrent computation to the processors of a multiprocessor system. Adjusting this mapping in order to complete the workload in the minimum possible time (i.e. to share the workload among the processors evenly and minimize inter processor communication) is known as load balancing. Deciding on the mapping before execution based on compile time information is known as static load balancing while adjust the mapping during the execution (via process migration) is known as dynamic load balancing.

The papers in this session incorporate techniques for mapping processes both before and during execution in order to maintain an effective load balance. However one of the papers deals with the distributed computing environment while the other is in the area of object oriented programming environments for multiprocessors.

The first paper in this session, entitled “The Efficient Management of Task Clusters in a Dynamic Load Balancer”, describes work being carried out on load balancing of multiuser distributed systems. A novel technique is proposed which deal with groups of subtasks, known as task clusters rather than single task units. This provides the advantage of letting the user submit several tasks, in a script type format, to the load balancer. The load balancer, which consists of a load manager running on each node in the system, can then distribute the subtasks across the nodes thus executing the task cluster in parallel.

Allowing task clusters to be submitted to the load balancing system gives increase efficiency over submitting tasks separately. Currently two strategies for task cluster management are being considered by the authors. The two alternatives are based on extensions to either a bidding strategy or a probing strategy. The paper will compare the use of these two options. In addition the load balancing scheme has been implemented across a network of workstations and performance results from experiments which compare the scheme described in the paper with an old scheme will be included.

The second paper which is entitled “The Benefits of Migration in a Parallel Objects Programming Environment” deals with load balancing of distributed memory multiprocessors used for object oriented programming. The parallel objects environment is based on the active object model. Parallel object applications can be highly dynamic as new objects and new threads of execution within objects can be created at run time. A scheme which automatically handles the load balancing of parallel object applications, using both static and dynamic techniques, is presented.

Objects available at the start of the execution are grouped in clusters according to communication between them and then allocated to processors so that each has approximately the same computation and memory load. This is optimized using a branch and bound algorithm. During execution each node has a monitoring manager, allocation manager and creation manager as well as a router which handles the placement of new objects as well as the migration of existing objects. The paper compares the performance of parallel objects applications with and without the load balancing system running on a Meiko computing surface.

This session will be of value to anyone who is interested in the issues involved in making efficient use of the potential power offered by parallel processing.