

# The Performance of Processor Co-Allocation in Multicenter Systems

A.I.D. Bucur

Faculty of Information Technology and Systems  
Delft University of Technology  
P.O. Box 5031, 2600 GA Delft, The Netherlands

D.H.J. Epema

## 1. The model

In systems consisting of multiple clusters of processors interconnected by relatively slow communication links, co-allocation may be required. We study its performance by means of simulations, depending on the structure and sizes of jobs, and the communication speed ratio.

We model a multicenter with  $C$  clusters of identical processors. The workload consists of rigid jobs that require fixed numbers of processors, possibly in multiple clusters simultaneously. A job is represented by a tuple of  $C$  values, each generated from a same distribution  $D$ . In an *ordered request* the positions of the components in the tuple specify the clusters from which the processors must be allocated. For an *unordered request*, by the components of the tuple the job only specifies the numbers of processors needed in the separate clusters. A *flexible request* specifies the total number of processors, obtained as the sum of the values in the tuple. For *total requests*, there is a single cluster and a request only specifies the total number of processors needed. All intracenter communication links have the same speed, as do all intercenter links. The *communication speed ratio* is the ratio between the time needed to complete a send operation between processors in different clusters and in the same cluster.

## 2. Simulating co-allocation

**The influence of the communication speed ratio.** We consider a multicenter consisting of 4 clusters with 8 processors each, and a single-cluster with 32 processors. We find that for ordered, unordered and flexible requests the increase of the communication speed ratio increases the response time (see Fig. 1).

**Co-allocation versus no co-allocation.** We consider a multicenter consisting of 4 clusters with 32 processors each. In the no-co-allocation case, we have single-component jobs with request sizes obtained as the sum of 4 components from  $D$ . At low utilizations avoiding co-allocation brings benefits for large speed ratios (see Fig.

2). However, for intermediate and high system loads, rather than waiting for enough idle processors in one cluster, spreading requests over clusters can improve the performance, despite the extra communication.

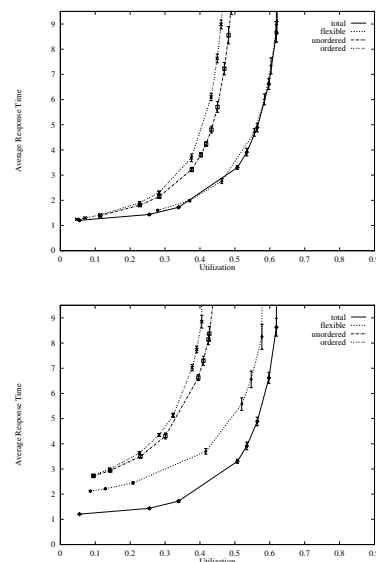


Figure 1. The performance for communication speed ratios 10 (top), and 100 (bottom).

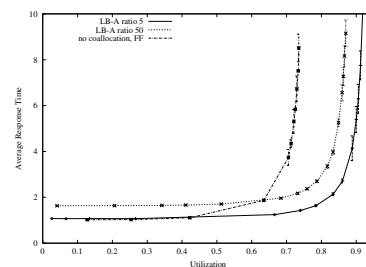


Figure 2. Comparing co-allocation for flexible requests with no co-allocation.