A Paradigm for Reconfigurable Processing on Grid

Mahmood Ahmadi and Stephan Wong

GridSim Toolkit Simulator

In a Grid environment, it is hard and even impossible to perform scheduler performance evaluation in a controllable manner as resources and users are distributed across multiple organizations with their own policies. To overcome this limitation a Java-based discrete-event grid simulation toolkit has been developed that called **GrdiSim**. The main characteristics of GridSim are as follows:

- The toolkit supports modeling and simulation of heterogeneous grid resources, users and application models.
- It provides primitives for creation of application tasks, mapping tasks to resources, and their management.
- It investigates techniques to incorporate background traffic and network effects in GridSim.
- It supports modeling and simulation reconfigurable architectures and general purpose architectures (our contribution).
- It can support cooperative processing using **neighboring** concept (our contribution).

Our assumptions

- Each application can be broken as different subjobs that called gridlets. Each application is packaged as gridlets whose contents include the job length in MI (million instructions). The job length is expressed in terms of the items of the time it takes to run on a standard GPP with MIPS rating of 100.
- Each Reconfigurable element accelerates the submitted subjob in compared to GPP that this acceleration rate is represented by **speedup factor**.
- A grid consists of reconfigurable and general purpose elements.
- The waiting time for reconfigurable elements is summation of standard waiting time (queuing time) and reconfiguration time.
- There is not any background traffic on the network.

Methodology

Processing elements communicate and collaborate together based on **neighboring concept**. In the neighboring concept each processing element cooperates with their neighbors with minimum cost.

In other words, the network combined of some primitives. Each primitive includes a collaborator processing element and requester processing element. Basic primitives are depicted in Figure 1.

P. Primet et al. (Eds.): GridNets 2008, LNICST 2, pp. 259–262, 2009.

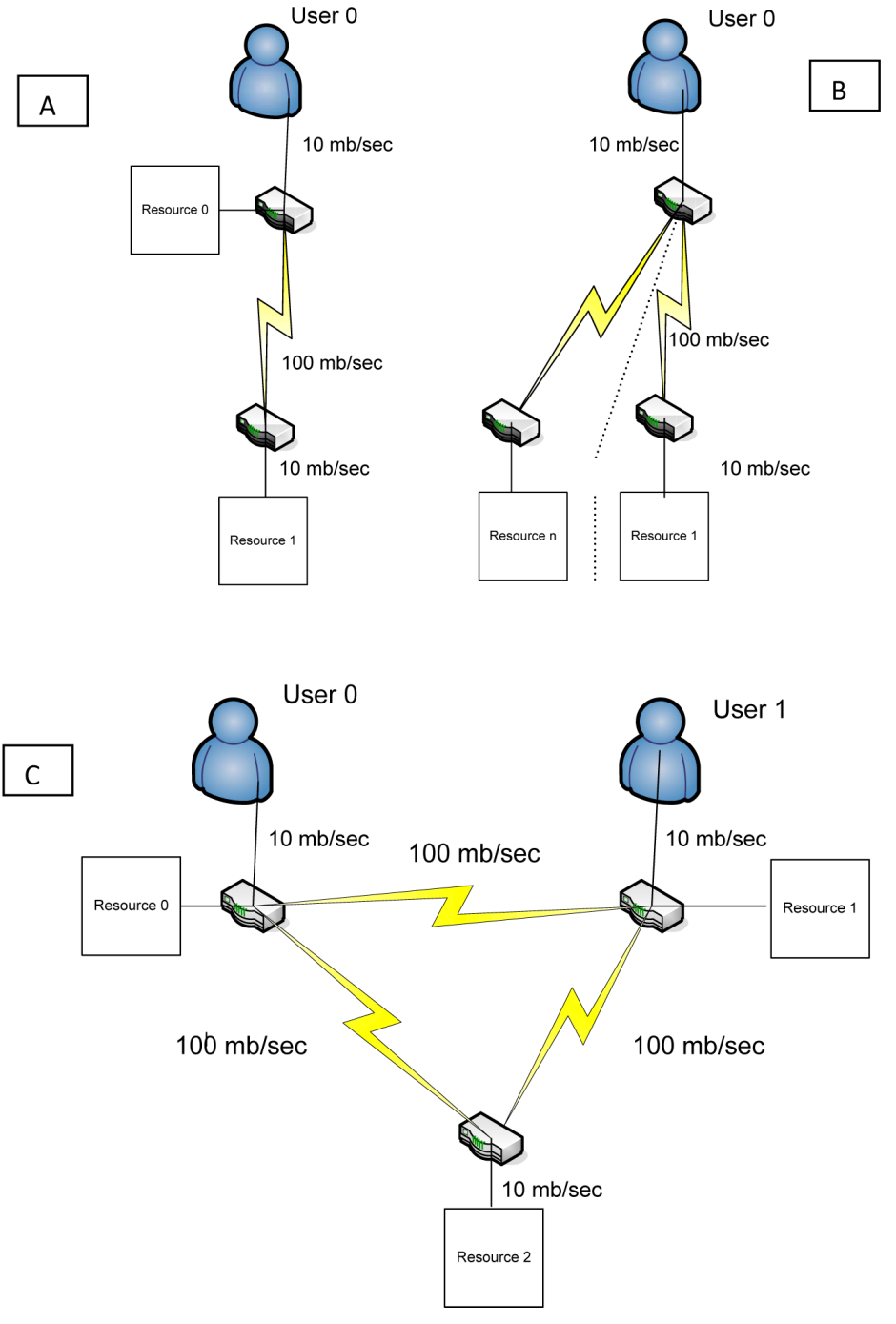

Fig. 1. Basic primitives (A)1-array primitive (B) n-array primitive (C) Complex primitive

Case studies

Basic primitives in Figure 1 are investigated as case studies with following specifications.

- maximum packet size =1500 byte/sec
- user-router bandwidth =10 mb/sec
- router-router bandwidth=100 mb/sec
- number of gridlets =10
- size of gridlets=5000 MI
- size of all gridlets is same
- MIPS rating of GPP=377 mips
- Minimum speedup for reconfigurable elements=2
- Maximum speed up for reconfigurable elements =10
- Reconfiguration file size =6 mb
- Reconfiguration speed=2
- Reconfiguration time= reconfiguration file size/reconfiguration speed=3 sec
- Users submit their job to related resource after that each resource finds the neighbor resource to cooperative processing.
- Each resource can be included GPP, reconfigurable element (RE) alone or different combination such as cluster of processing elements. In this case we utilize only one GPP or Re in each resource.

Results

The results show that utilization of reconfigurable architectures increase performance.

In this case, the best performance is achieved when all of elements are REs and speed up factor set to 10.

When the combination GPP and RE is utilized, a variation in waiting time and turnaround time is observed that is due to the MIP difference in GPPs and REs.

To achieve the maximum performance some significant factors are:

- Application type: cooperative processing is useful for computational-intensive applications.
- Reconfiguration time: another important factor is reconfiguration time. It should be noted the summation of reconfiguration time and accelerated CPU time must be less than normal CPU time (the job execution time on GPP).
- The results can generalize when the size of gridlets and processing elements MIPS are random.

Mathematical Analysis

This analysis is performed to show the effect of different parameters on the processing time in collaborative and non-collaborative systems.

To describe the related equations, the following notations are defined.

N= number of subtasks
S=size of subtask= MI instruction
T_{GPP}= processing time for each instruction by GPP
T_{RE}=processing time for each instruction by RE=$\frac{T_{GPP}}{k}$

$T_{Communication}$= communication time between different processing elements in collaborative systems.
k=speedup factor
N_1=number of subtasks that can process by neighbor
Pkt=packet size
Bw=network bandwidth
N_{pkt}= number of packets

$$T_{Noncollaboration} = \frac{\sum_{i=1}^{N} MI_i}{T_{GPP}}$$

$$T_{Collaboration} = \frac{\sum_{i=1}^{N-N_1} MI_i}{T_{GPP}} + \frac{\sum_{i=1}^{N_1} MI_i}{T_{RE}} + N_1 T_{Reconf} + T_{Communication}$$

$$T_{Collaboration} = \frac{\sum_{i=1}^{N-N_1} MI_i}{T_{GPP}} + \frac{\sum_{i=1}^{N_1} MI_i}{T_{GPP}} k + N_1 \frac{Reconf.\ file\ size}{Reconf.\ speed} + T_{Communication}$$

$$N_{pkt} = N.S/Pkt$$

$$T_{Communication} = packet\ transmision\ time.N_{pkt} = \frac{Bw}{Pkt}.N.S/Pkt$$