How to Determine What to Do Next: A Concurrent Logic Programming Language ON Based on Declarative Priority

Keiji Hirata and Kenichi Yamazaki
Media Information Laboratory NTT Network Innovation Laboratories

Though computers may be versatile, they still lack many of the capabilities we associate with intelligent action. For example, it is difficult for computer to determine the priority of importance of a particular job. Imagine that the arm of a robot is going to hit an obstacle that may damage it. A sensor in the arm will signal to the robot brain that the distance between the arm and the obstacle is less than a certain threshold. Then the brain should therefore concentrate its resources on the job of stopping the arm, even if it has many other jobs awaiting execution. This situation is not a problem for a human but can pose a problem for a computer.

In a conventional prioritizing, a priority, represented by an integer, is assigned to each job, and the higher priority is, the more important the job is. In the above robot example, the arm sensor manager should be assigned the highest priority. This technique, which is very simple and easy to implement, works well if there are few jobs and the relationship between jobs is not complicated.

The tasks that now computer systems perform have become much more complicated, where many subjobs are created dynamically; thus, we can neither precisely predict how many jobs will be created nor control the relationships between jobs. It is, therefore, quite difficult to assign a relevant integer priority to every job.

We found that this problem arises from representing priority as a range of integer values. We introduce a new way of representing priority in which three kinds of relations exist between two priorities, i.e. one job has priority over the other, both have the same priority, and the two jobs are unrelated. In particular, we can not describe the *unrelated* relation with any of conventional priority techniques. We have proved that a computational model based on this type of priority represention has many good properties [1].

In addition, we have designed a new programming language, called ON, that incorporates our new priority scheme. We can describe concurrent programs in ON more naturally than with conventional programming languages.

[1] Hirata, K., and Yamazaki, K. Declarative Priority In A Concurrent Logic Language ON, *Proc. of International Conference on Logic Programming 1999*, pp.215–229 (1999).

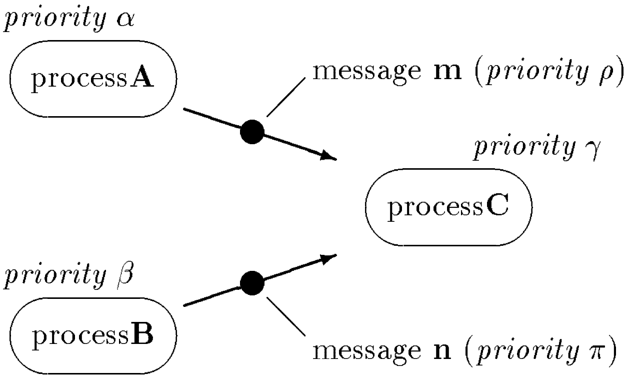

Fig. 1: Assigning priority in ON.

In ON, we can assign a priority to each process and message. We can declare the relation between two priorities using an operator $\succ$ as follows:

$\rho \succ \pi$

Note here that if we do not declare any relation between two priorities, the priorities are thought to be '*unrelated*'.