Implementing Distributed Computing Abilities of a Statistical System

Yoshikazu Yamamoto, *Tokushima Bunri University, Faculty of Engineering*
1314-1 Shido, Okawa, Kagawa 769-2193, Japan, yamamoto@is.bunri-u.ac.jp

Junji Nakano, *the Institute of Statistical Mathematics*
4-6-7 Minami-azabu, Minato-ku, Tokyo 106-8569, Japan, nakanoj@ism.ac.jp

Takeshi Fujiwara, *Graduate University for Advanced Studies*
4-6-7 Minami-azabu, Minato-ku, Tokyo 106-8569, Japan, fuji@ism.ac.jp

Ikunori Kobayashi, *Tokushima Bunri University, Faculty of Engineering*
1314-1 Shido, Okawa, Kagawa 769-2193, Japan, ikunori@es.bunri-u.ac.jp

1. Introduction

Computer technologies have been developed dramatically rapidly in this decade. The dissemination of cheap but powerful personal computers and computer networks is especially influential for all the human activities, including statistics. Here, we mention two important changes brought to the statistical environment by it. Firstly, we are able to develop and execute complicated computer intensive methods such as nonparametric modeling, resampling methods, or simulation experiments by using powerful computers at hand. Secondly, the amount of data we have to handle becomes tremendous, mainly because many computers gather data automatically and continuously and we can access them through Internet easily. Such huge datasets require another type of new methodologies for analyzing them, for example, so called data mining techniques. As a result, although we have far more powerful computer performance by faster CPUs or larger storage amount than before, required calculations for statistical works still go beyond the usual single computer ability.

Parallel processing or distributed computing is one solution for it. For example, almost all recent expensive high performance computers have the multiple processor architecture. Also as we have many cheap personal computers connected by networks, it is natural to perform calculations simultaneously on them. For realizing these tasks, some distributed computing technologies are available, for example, MPI (Message Passing Interface) and PVM (Parallel Virtual Machine). However, they are too general and not easy to use for usual statisticians. It is heavily required to have easy distributed computing abilities in statistical systems.

We realize simple functions for distributed computing in order to execute calculations independently on remote computers in the statistical system Jasp (Nakano et al., 2000), which is

designed mainly for utilizing various new technologies in the recent computational environment.

2. Distributed computing functions

Jasp adopt a client/server architecture for distributed computing. The client program works as a user interface (UI), and the server program takes part in calculations, in other words, a language interpreter. As we use the Java language to implement Jasp, this architecture is also useful to execute the client UI program as a Java applet on Web browsers. As an applet can communicate only with programs on the computer from which the applet was transferred, the client UI applet can use the server on the Web server computer. We call the server to which the client connects firstly as a main server, and let a main server communicate with other remote servers. Thus communications are performed among servers and between a client and a main server.

As Jasp has a function based language enhanced by an object oriented framework, we realize distributed computing abilities as simple functions given in Table 1. They are easy to use and sufficient for strongly independent statistical works such as grid searches for maximizing complicated likelihood functions or large simulation experiments.

Table 1: Functions for distributed computing

Function name	Usage and description
`remote`	`remote(command, serverName)` executes `command` on the server whose name is `serverName`
`remotes`	`remotes(commands)` executes `commands` on the specified servers and waits until all the calculations will be done, where `commands` is an ArrayList whose elements are "`command`, `serverName`" like in the function `remote`
`send`	`send(value, variableName, serverName)` substitutes the value of `value` to `variableName` on the server whose name is `serverName`
`receive`	`variableName1 = receive(variableName2, serverName)` substitutes the value of `variableName2` on the server whose name is `serverName` to `variableName1`

REFERENCES

Nakano, J., Fujiwara, T., Yamamoto, Y. and Kobayashi, I. (2000). A statistical package based on Pnuts. In: *COMPSTAT2000 Proceedings in Computational Statistics*, 361-366. Heidelberg: Physica-Verlag.