Astronomical Data Analysis Software and Systems VI
ASP Conference Series, Vol. 125, 1997
Gareth Hunt and H. E. Payne, eds.

The Grid Signal Processing System

I. J. Taylor[1]

Department of Physics and Astronomy, University of Wales, College of Cardiff, PO BOX 913, Cardiff, Wales, UK,
E-mail: Ian.Taylor@astro.cf.ac.uk

B. F. Schutz[2]

Albert Einstein Institute, Max Planck Institute for Gravitational Physics, Schlaatzweg 1, Potsdam, Germany.
E-mail: schutz@aei-potsdam.mpg.de

Abstract. In this paper, we present an interactive signal-processing environment called Grid. Grid enables users to create programs by graphically connecting a desired set of tools. It is written in Java and therefore can run on virtually any computer platform. Grid is being developed to provide a quick-look data analysis system for the GEO 600 gravitational wave detector, but we have found its use is much more general with currently many collaborators using it within their own specific fields.

1. Introduction

Signal-processing systems are fast becoming an essential tool within the signal-processing community. This is primarily due to the need for constructing large complex signal-processing algorithms which would take many hours of work to construct using conventional programming languages. Here, we present such a signal-processing system, called Grid. Grid is a graphical interactive *multi-threaded* signal-processing environment which allows the creation of complex arrays of algorithms (called units within Grid) by simply choosing the desired units from a selection of toolboxes then graphically wiring them together.

2. An Overview

When Grid is run three windows are displayed. A *ToolBox* window, a *MainGrid* window and a *Dustbin window*. Briefly, the ToolBox window shows the various tools available within Grid, the MainGrid window allows algorithms to be connected together and the Dustbin window allows unwanted units to be discarded.

---

[1] A post doctoral researcher at Cardiff who has been developing Grid since January 1996. For more information, see http://www.astro.cf.ac.uk/pub/Ian.Taylor/Grid.

[2] Professor Schutz is a Director of the Albert Einstein Institute

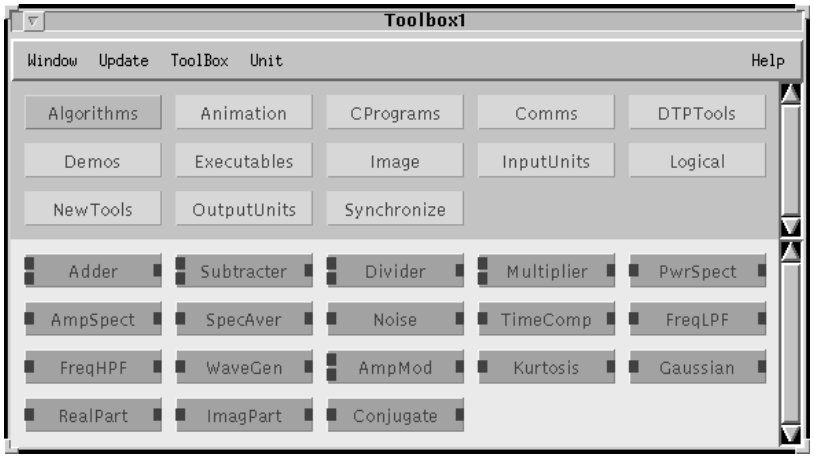

Figure 1. Grid's ToolBox window consists of a section which lists the available toolboxes and another showing its associated tools.

2.1. Distributed ToolBoxes

Figure 1 shows the *ToolBox* window which is divided into two sections. The top section shows the available toolboxes and the bottom section shows the tools which are contained within the selected toolbox. When Grid is run, it scans the toolbox paths (specified by the *TOOLBOXES* environment variable) and shows the detected toolboxes on the upper part of the *Toolbox* window. Toolboxes (and the associated tools) can be stored on a local server or distributed throughout several network servers. The network addresses are specified in the *TOOLBOXES* and the standard Java *CLASSPATH* environment variables. Grid uses its own *class loader* which makes the loading in of classes via the Internet or a local server totally transparent.

2.2. Programming Within Grid

The MainGrid window (see Figure 2) gives a typical example of an algorithm constructed within Grid. Within Grid, instead of re-coding computer algorithms when the specific connectivity needs to be altered slightly, we simply wire the algorithm in different way. Units are created by *dragging* them from the ToolBox window to the desired position in the MainGrid window and then connected together by dragging from a socket on the right of a sending unit to the socket on the left of a receiving unit. Once the desired connectivity is in place, the algorithm can be started by clicking on the *start* button and run in a *single step* fashion (i.e., one step at a time) or continuously, i.e., where an algorithm may be applied to continuous data (for example, in analysing the output from a gravitational wave detector or when animating the formation of star clusters, etc.). Each unit is run as a separate *thread* and therefore automatically load-balanced by the specific operating system.

The algorithm, shown in Figure 2 implements a simple algorithm which compares two signals. The basis of the signal is formed by contaminating a sine wave (of 2 kHz) with Gaussian noise (5 times its amplitude) and transforming

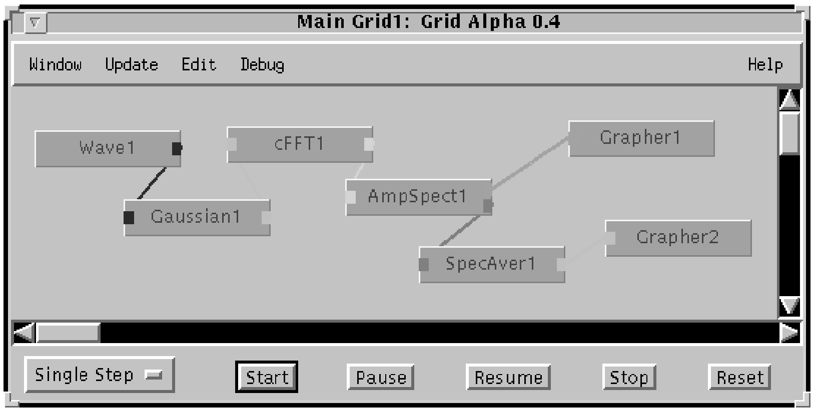

Figure 2. Grid's programming window, called the MainGrid.

this into the frequency domain by using an FFT.[3] The output of the Spectrum is then split into two, with one copy being sent straight to the first Grapher and the other sent via a spectral averaging unit to the second Grapher. The spectral averager receives a specified number of spectra (in our case 20) and outputs their average. The signals are displayed using two Graphers (but we could have just as easily have used one). As many concurrent signals as the user wants can be displayed on the same Grapher, each having its own drawing colour and line style). The result of the second Grapher is given in Figure 3. The first Grapher simply shows each new incoming signal.

Once the signal is displayed, it can be investigated further by using one of the Grapher's various zooming facilities. Zooming can be controlled via a zoom window which allows specific ranges to be set or by simply using the mouse to *drag* throughout the image. For example, by holding the control key down and dragging down the image is zoomed in vertically and by dragging across from left to right zoomed in horizontally. The reverse operations allow zooming out. Also once zoomed in, by holding the shift key and the control key down the mouse can be used to move around the particular area you are interested in. We also have another powerful zooming function which literally allows the user to drag to the position of interest and the image will zoom in accordingly.

3. For What Platforms is Grid Available?

Grid is written mostly in Java (e.g., there are about 33,000 lines of Java source code/documentation and about 500 lines of C code). A previous version was written in C++ using InterViews (Taylor & Schutz 1995) but was abandoned earlier in the year.[4] Grid is now in its fourth *alpha* testing stage with many

[3] the FFT is written in C for efficiency and dynamically linked into Java at run time

[4] We are grateful to Justin Shuttleworth who implemented the original design of Grid as the ideas presented in this version provided a strong foundation for the current design of Grid's graphical user interface

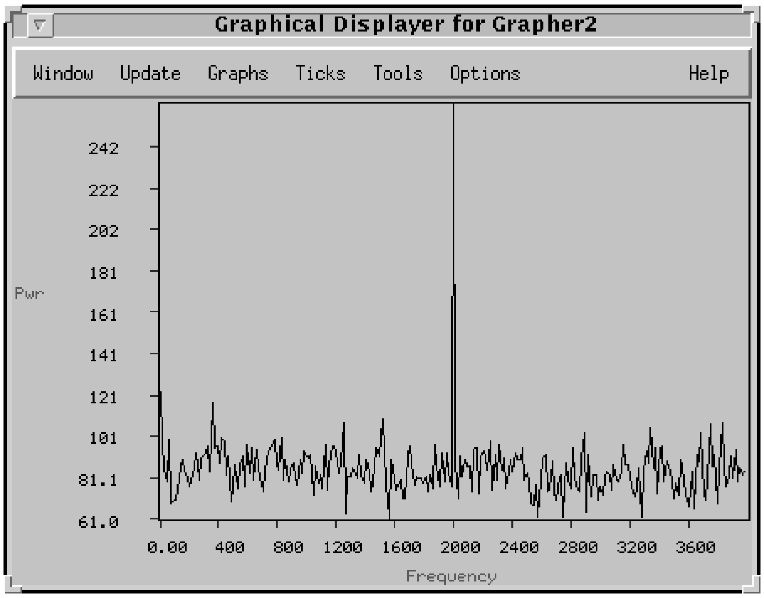

Figure 3. The Grapher can simultaneously display as many signals as required each with its own colour and line style but here we show just one display of a 2 kHz sine wave with Gaussian noise added.

more features being planned for the coming year. Grid is being made available via the World Wide Web.[5,6]

4. Current Applications of Grid

Although Grid is being developed to provide a quick-look data analysis system for the GEO 600 gravitational wave detector (expected to be built by mid-1999) its use is much more general. For example, we have created toolboxes for animation, image viewing, and certain desk-top publishing tasks. Collaborators are already working on units which can manipulate FITS images, apply a variety of signal and image processing algorithms to a variety of sources, and provide multi-media teaching aids; there is also a project to construct a musical composition system. In the future, we anticipate that Grid will be applied to many other varied subject areas.

References

Taylor, I. J. & Schutz, B. F. 1995. in Proceedings of the International Computer Music Conference 95, ed. R. Bidlack (San Francisco, CA: ICMA), 371

---

[5] http://www.astro.cf.ac.uk/pub/Ian.Taylor/Grid

[6] We do not discount the possibility of a version of Grid being made commercially available but none-the-less we will always provide a free version for download.