Understanding Multi-Transputer Execution

Susan Stepney

GEC-Marconi Research Centre, UK.

ABSTRACT

An understanding of parallel execution as good as, if not. better than, our current. understanding of conventional von Neumann execution is required in order to exploit the potential offered by multi-Transputer and other parallel systems.

ParSiFal's GRAIL display and occam profiling tools have been recognized as providing valuable visualization of how occam programs execute in Transputer arrays.

Here I explain how the GRAIL interface has been designed and used to enhance understanding of occam program structure and activity.

Introduction

Transputers (1) provide an ideal building block for parallel computers, all the way from small real-time control systems or personal computers, containing tens of processors, up to supercomputers with thousands of processors.

In order to program such systems effectively, it is necessary to have a good understanding of the execution of occam (2) programs. But even the static structure of an occam program is considerably more difficult to visualize and understand than that of a conventional, sequential one.

An occam program has many threads to the computation, and each thread can have all the complexity of structure of a sequential program. In addition, there can be a complicated communication structure between these threads. Pictorial methods can help make the structure more visible.

Once the static structure has been visualised, data about. program execution can be more readily grasped, especially if it is presented in a similar manner.

In this paper I describe GRAIL (Graphical Representation of Activity, Interconnection and Loading), one of the software tools being developed in the ParSiFal project to aid the design and understanding of occam programs.

Grail Display

The pictorial representation used in GRAIL is two dimensional. The vertical dimension (down the page) is used in the conventional way to represent. sequential execution, and deterministic choice (`IF`). The horizontal dimension (across the page) is used to indicate parallelism and non-deterministic choice (`ALT`). The various parallel threads of execution are drawn side by side.

The GRAIL display uses the process as its fundamental unit. An occam process is drawn in a rectangular box, and each process can be hierarchically composed of other processes.

Channels are drawn as arrows between the processes they connect, and show the direction of the communication. Folds are used to control the amount of information being displayed.

Figure 1 shows the GRAIL display of the well known processor farm pipeline:

```
PRI PAR
  PAR
    router(...)
    mixer(...)
  worker(...)
```

Susan Stepney. "Understanding Multi-Transputer Execution". IT UK 88, University College Swansea, 1988.

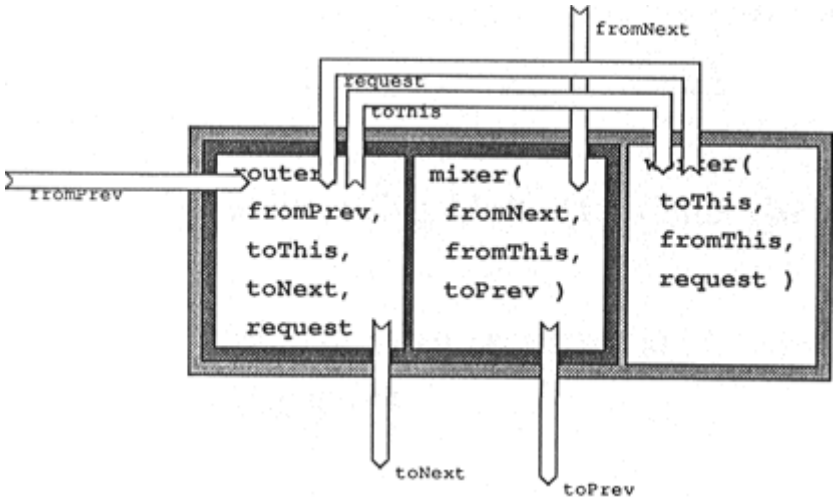

Figure 1 : GRAIL display of the program running on one Transputer in a processor farm

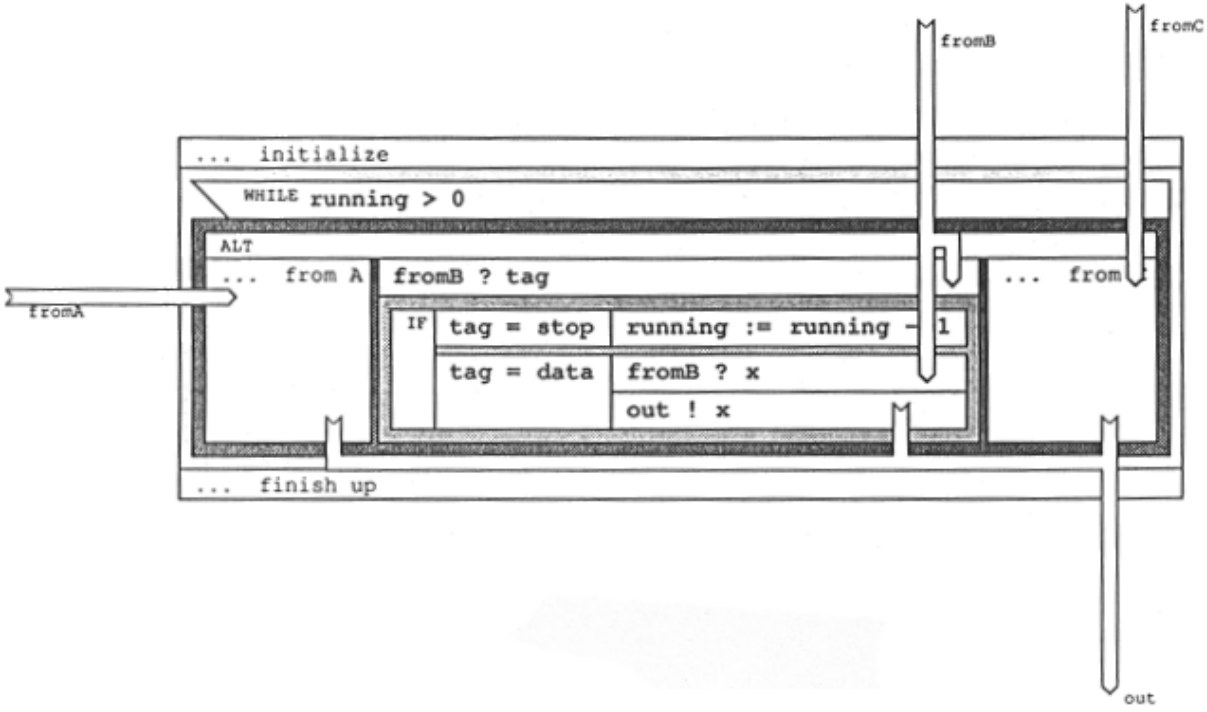

Figure 2 : GRAIL display of a 3-to-1 multiplexer

`router(...)` collects work from the previous Transputer in the pipe, and sends it to its own worker, if requested, or to the next Transputer. `mixer(...)` collects results from its own worker and the next Transputer and sends them to the previous Transputer.

Figure 2 shows the GRAIL display of a simple channel multiplexer, (similar to the mixer process of Figure 1, except. that it has three input. channels). It. is an `ALT` inside a `WHILE` loop. The `ALT` waits for input from the three channels `fromA`, `fromB` and `fromC`, and merges them onto the channel out. This structure is clear from the picture, but not from the occam text.

The GRAIL display is described in more detail in (3), and is formally specified in (4).

GRAIL (currently running under SunView on a Sun Workstation) is interactive; the user can select. which procedure to look at, can manipulate folds, and can display or hide channels, using the mouse.

Static Profiling

Another tool developed in ParSiFal is an occam profiler running on an array of Transputers. As the program executes, the profiler counts statement. executions, and when the program has terminated, the profile results are wormed out. of the array.

GRAIL was originally developed in order to display this profiling data. Activity information is given as a colour overlay. Blue indicates inactive processes, with a gradual change in colour to red for the most active processes.

This paper's medium of static, black and white text is not a very satisfactory way of describing an interactive colour graphical display. The colour in the display allows interesting areas of the program to be found "at a glance"; blocks of a particular colour are much easier to identify than one anomalous number out of a list of many. Worryingly hot, or unexpectedly cold, areas of the program can be identified with ease. The interactive aspect allows the user to examine these areas in more detail, by entering folds and zooming in, or to step back and look at the overall, global pattern of activity.

Colour is used to identify the interesting areas, but once quantitative results are required, the actual profiling information can be displayed on the screen in a textual form. Thus GRAIL uses graphics to display structure and behaviour, but can revert to text when detailed data is required. Using GRAIL for displaying static profiling data is described in more detail in (5).

Dynamic Profiling

The static profiler, although useful, does not. give enough information for a complete understanding of program execution. Transient. bottlenecks and trouble spots are averaged out. So the monitoring work is being extended to monitor activity in a program as it is running, and to display activity dynamically.

The static profiling mechanism can be used on any network of Tranputers - the links are used to gather the data after the program has finished executing. Dynamic monitoring requires the extraction of data at the same tine as the program is executing. Using the links would cause unacceptable distortion. The backplane bus of the ParSiFal T-rack is used as the route to extract this data.

Any other Transputer hardware with a similar structure can use the same scheme.

GRAIL is being extended to cope with the extra dimension of information - time - that is being displayed. This leads to interesting problems in how to display such information. The obvious way is to have a time varying display. This works for spotting sudden changes in an otherwise quasi-static system, provided the user is given a rewind option. However, if the whole system is very dynamic, the main reaction is confusion (if not sea-sickness)! A static display of graphs (for example, plotting colour against. time), also has its own advantages and disadvantages. In keeping with the idea that different displays are suitable under different circumstances (and for different users) a range of options is being provided.

This dynamic monitoring information is being used to highlight transient features of program execution, and to help gain greater understanding of how parallel components of occam programs interact and work together.

Understanding

The aim of this work is to develop understanding, in order to identify various standard solution frameworks to certain identified classes of problems. This is being done as a set. of case studies within ParSiFal, with applications including neural networks, vision processing, finite element analysis, and network simulation.

Conclusions

Occam programs can be complicated, not. only in their structure, but. in their execution patterns. Graphical tools can aid understanding, by providing a high-density display of structure and activity data in a readily graspable form. GRAIL is one such tool.

References

1. Inmos Ltd., *The Transputer Family*, product information, 1987.
2. Inmos Ltd., *occam 2 Reference Manual*, Prentice-Hall International, 1988.
3. S. Stepney, "Pictorial Representation of Parallel Programs", in A.C. Kilgour and R. A. Earnshaw, eds. *Graphics Tools for Software Engineering*, BCS Proceedings, Cambridge University Press, 1989.
4. C. Rees, "Z Specification of the GRAIL Display", PSF/GEC/1JP3/88/9
5. S. Stepney, "GRAIL - Graphical Representation of Activity, Interconnection and Loading", in T. Muntean, ed. *7th Technical Meeting of the occam User Group*, IOS Amsterdam, 1987.