Development of a distributed knowledge-based system

Conference or Workshop Item

Development of a Distributed Knowledge-Based System

K. C. P. Wong
Faculty of Technology,
The Open University,
Walton Hall, Milton Keynes,
MK7 6AA, UK

L. Nolle
Oxford Research Unit,
The Open University,
Boars Hill, Oxford,
OX1 5HR, UK

A. A. Hopgood
Department of Computing,
Nottingham Trent University,
Burton Street, Nottingham,
NG1 4BU, UK

Abstract - This paper describes the development of a distributed knowledge-based system. A software system, namely Distributed Algorithmic and Rule-based Blackboard System (DARBS), was developed from its predecessor ARBS, which lacked the distributed computing feature. ARBS has been used in solving a number of engineering problems [1-3]. DARBS now utilises client/server technology. It consists of a centralised database server, called the "Blackboard" and a number of Knowledge Source Clients (experts). It distributes the workload to a number of clients which are rule-based or other AI systems with specific knowledge in various areas. DARBS is being applied to automatic interpretation of non-destructive evaluation (NDE) data and control of plasma deposition processes.

Keywords: DARBS, blackboard system, distributing computing, knowledge-based system, rules.

1 Introduction

Over the years, ARBS, an in-house rule-based system, developed at the Open University, has been successfully applied to a number of projects for solving engineering problems, ranging from non-destructive evaluation (NDE), controlling plasma deposition processes and controlling telecommunication networks [1-3]. However, ARBS had some limitations: it was originally written in Poplog[1] and designed to operate on SUN SPARC workstations under Unix systems. In recent years, the Linux operating system has become an economical, reliable and flexible clone of the Unix system that can be run on inexpensive personal computers. Consequently, the authors decided to port the whole system to Linux. Nevertheless, because ARBS was written in Pop11, utilising the Poplog environment, the authors decided to re-design ARBS, using C++ and an object-oriented approach. In addition, ARBS was originally implemented as a single process, i.e. only one knowledge source could be active at one time. Each knowledge source had a set of preconditions that needed to be fulfilled before a knowledge source could be activated. A separate control module scheduled the knowledge sources using a first-come, first-served strategy. This was clearly a violation of the opportunistic idea of a Blackboard system [4]. In order to overcome this limitation, a distributed architecture has been used. A client/server technology is employed and the communication is through TCP/IP[2]. The new system consists of a Blackboard Server (BS) and a number of Knowledge Source Clients (experts). Workloads are distributed to a number of "experts" which are rule-based or other AI systems with specific knowledge in various areas. These "experts" communicate by adding or removing information to the blackboard. The concept of a blackboard system is analogous to a group of experts discussing a problem by writing and updating information onto the blackboard. The new system is now called Distributed Algorithmic and Rule-based Blackboard System (DARBS).

2 ARBS

ARBS was originally developed in 1990 with funding from the UK Engineering and Physical Sciences Research Council (EPSRC) and has been refined during several subsequent research projects [1-3,5]. ARBS was written in Poplog and designed to operate under Unix systems. The first version of ARBS was a rule-based system with embedded algorithms. Subsequently, ARBS was refined and became a blackboard system, which enables a variety of different computer techniques within one system. Specific tasks are handled by separate knowledge sources that communicate by adding information to an area of the blackboard. Rule-based, procedural, neural networks and genetic algorithm knowledge sources have been successfully integrated into the blackboard system (see Figure 1). However, the performance of ARBS is compromised because it requires Poplog to act as a virtual machine. Furthermore, ARBS lacks the distributed computing feature. For tackling increasingly complicated engineering problems, it was decided to re-design a new software system that facilitates the above requirements. The new system is called DARBS and is written in standard C++.

3 DARBS

The major improvement of DARBS compared with ARBS is the introduction of parallelism. The knowledge of the problem domain is distributed to a number of knowledge sources. These knowledge sources can be seen as experts having knowledge in specific areas. The experts are independent and can only communicate through the blackboard. Figure 2 shows the architecture of the blackboard system.

3.1 Blackboard and Knowledge Sources

The blackboard has the appearance of a database. It can contain an unlimited number of partitions for knowledge sources (experts) categorising information. The knowledge sources can be seen as experts having knowledge in specific areas. Under a blackboard system, knowledge sources can only communicate through the blackboard. This ensures all the experts have a fair chance to access the information.

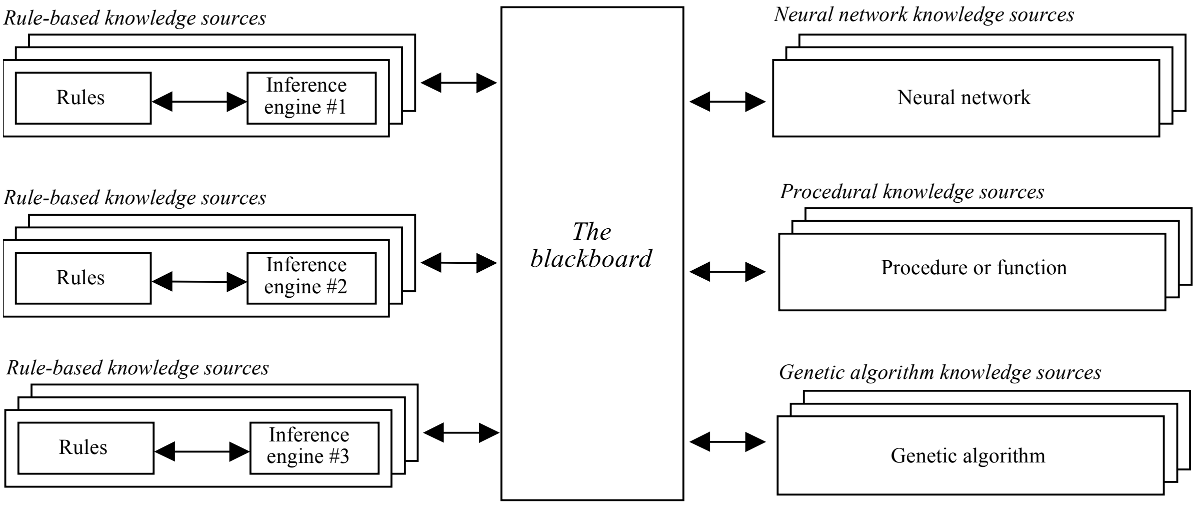

Figure 1 : The Blackboard architecture of ARBS.

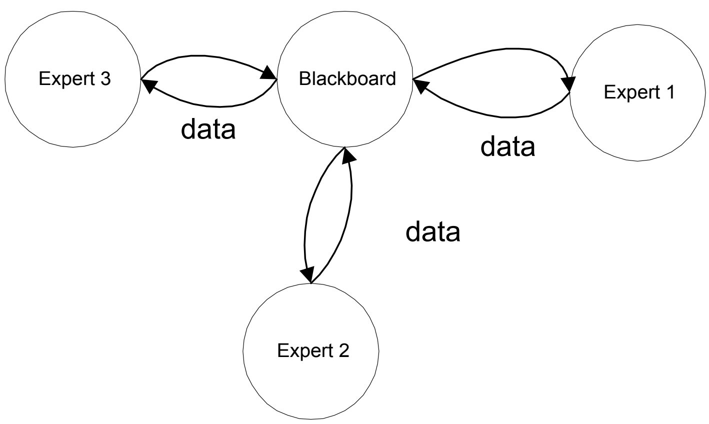

Figure 2: The architecture of DARBS.

3.2 Parallel Processing

In DARBS, a single blackboard and a number of "experts" co-operatively solve a problem. The experts observe the blackboard constantly and activate themselves when the information interests them. All the experts run in parallel. Whenever the content of a partition of the blackboard is changed, the blackboard server will broadcast messages to other experts. The experts themselves will decide how to react to this change. In other words, the rule-writer should consider how to deal with these broadcast messages when a rule is designed.

3.3 Client/Server Architecture using TCP/IP

DARBS utilises the client/server technology. The blackboard acts as a server and the knowledge sources act as clients. The communication is through standard TCP/IP (Figure 3).

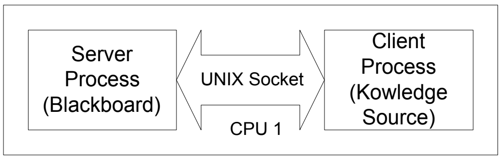

Figure 3 : Client/server communication using UNIX sockets.

With this approach, the server and clients operate as independent processes. The inter-process communications utilise the Internet sockets. This means DARBS can be operated over a wide area network (WAN) such as the Internet (Figure 4).

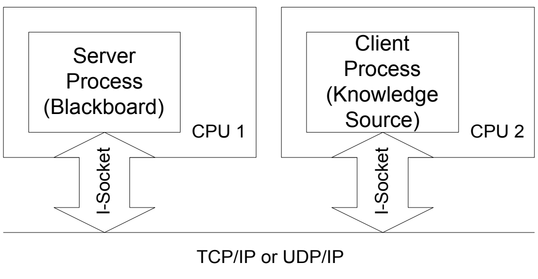

Figure 4 : Client/server communication using INTERNET sockets.

Furthermore, the system is platform-independent so that DARBS processes can be employed on various platforms. For example, the blackboard server can be run on a Unix machine and knowledge source clients can be run on a number of MS Windows, Mac, Linux and Unix machines. Another advantage of this approach is that the knowledge source client can be written in Java as a web-based applet and thus a knowledge source client can be operated under a web browser and no installation is required.

4 DARBS Knowledge Sources

DARBS integrates various computer techniques into one system. Particular tasks are co-operatively conducted by knowledge sources with specialised expertise. Rule-based and procedural knowledge sources have been integrated into DARBS and neural networks and genetic algorithm knowledge sources will be developed at later stage.

4.1 Knowledge Source Structure

Each knowledge source is encapsulated in a record, which contains seven fields (Figure 5). The first field specifies the knowledge source name. The second field is the knowledge source type. It can be either rule-based, procedural, neural network or other. For rule-based knowledge sources, there are fields for specifying inference mode (third field) and rules (fourth field). Field 5 is an activation flag that allows individual knowledge sources to be switch on or off. Field 6 is a set of preconditions, which must be satisfied before the knowledge source can be activated. The precondition may comprise sub-conditions joined with Boolean operators AND and OR. Consequently, an additional action field (field seven) states what actions are to be performed before the knowledge source is deactivated. For procedural, neural network and other non rule-based knowledge sources, the functions and procedures are listed in the action field.

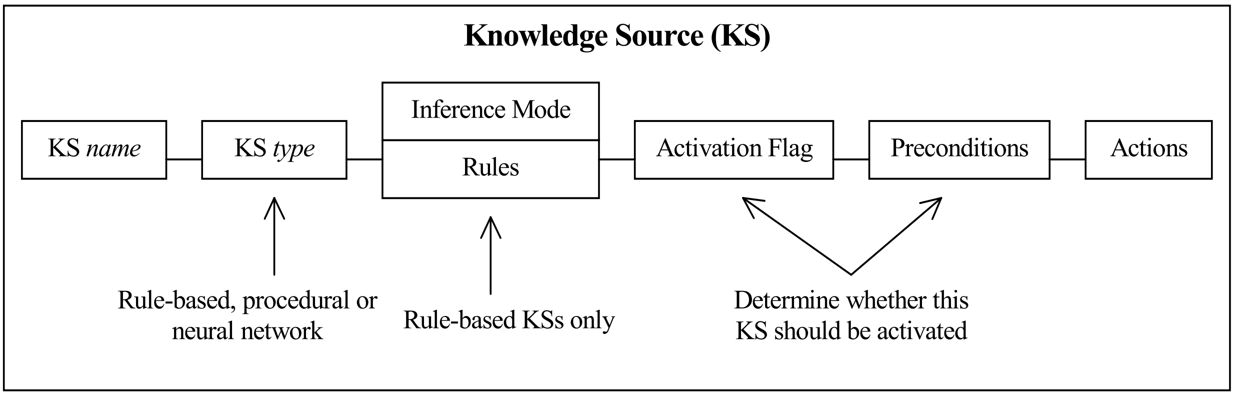

Figure 5 : Knowledge Source Structure.

In DARBS, all knowledge sources run in parallel. They constantly check the blackboard for activation opportunity. Knowledge sources can read data from the blackboard simultaneously. However, to avoid deadlock, only one knowledge source is allowed to write data to the same partition of the blackboard at one time. Whenever the content of the blackboard changes, the blackboard server broadcasts a message to all knowledge sources. The knowledge sources themselves decide how to react to this change, e.g. restart the knowledge source. With this approach, knowledge sources are completely opportunistic, i.e. activating themselves whenever they have some contributions to make to the blackboard.

4.2 Rule Structure

In DARBS, rules can be used for looking up information on the blackboard, making deductions about that information and posting new information on the blackboard. The rule structure is simple, but complex conditions and conclusion can be constructed (Figure 6). It consists of four fields. The first field is the rule number. It is followed by a set of preconditions and conclusions. Like the precondition field in knowledge sources, the precondition may comprise sub-conditions joined with Boolean operators AND and OR. The precondition must be satisfied before the conclusion field can be activated. In the conclusion section, information can be added to or deleted from the blackboard and local and external functions can be called. Finally, the last field is for explanation. It permits the rule writer to explain why the rule is executed.

4.3 Inference Engines

DARBS offers two types of inference engines, which are based on the principle of multiple and single instantiation of variables [5]. Within a rule, sharing of information can be achieved by employing variables (the blackboard is used for sharing information between rules). Under multiple instantiation, all possible matches to the variables are carried out with a single firing of a rule. In contrast, only the first match to the variables is found when single instantiation is used. Usually, the multiple instantiation inference mechanism is preferred because it is straightforward and efficient. However, single instantiation might be preferred in circumstances where a time constraint is imposed [2]. An example of a rule is shown in Figure 6.

5 Applications

DARBS is currently employed for automatically interpreting ultrasonic non-destructive evaluation (NDE) data and controlling plasma deposition processes. The applications are summarised in the following subsections.

```
RULE ghost_echo_prediction_rule                                        conditions
  IF
  [
   [on_partition [?centre1 is the CENTRE of the AREA == corners ~area1] setsoflinechars]
   AND
   [on_partition [?centre2 is the CENTRE of the AREA == corners ~area2] setsoflinechars]
  ]
  THEN                                                                       conclusions
  [
   [add [ghost echoes for centres ~centre1 and ~centre2 expected to pass thru
        ~[run_algorithm [ghostecho_predict [~centre1 ~centre2]] coords]] prediction_list]
   [report [ghost echoes for centres ~centre1 and ~centre2 expected to pass
          thru ~coords] nil]
  ]
  BECAUSE [~centre1 is the centre of the area]  <------------------  Explanation
END
```

Where:
The match variable, which is prefixed by a "?", will be looked up from the blackboard;
The insert variable, which is prefixed by a "~", will be replaced by the instantiations of that variable.

Figure 6 : A typical DARBS rule.

5.1 Ultrasonic Non-Destructive Evaluation

Previously, ARBS has been successfully applied to the interpretation of B-scan images from weld defects in flat ferritic steel plates [1]. However, the geometry of the specimen is relatively simple compared with real industrial components. Extensive work is being undertaken for interpreting ultrasonic images of turbine disks and blades, provided by Rolls-Royce. New knowledge sources and rules are under development to cope with the more complex geometric reflections. Artificial neural networks will be developed to classify type of defects.

5.2 Plasma Process Control

Previous work has involved using artificial intelligence (AI) techniques to control plasma deposition processes from pump-down to switch-off [3]. The AI approach has involved the use of rules and fuzzy logic to mimic the actions of a skilled operator. However, the system is unable to determine for itself the optimum plasma operating conditions. Our current work involves extending the system so that it can explore the parameter space in order to determine the optimum operating conditions. The new system will therefore design the fluxes of species towards surfaces to match the particular process requirements.

6 Further Development

DARBS is still in the development phase. The core architecture has been built and tested with simple examples. The results are satisfactory. However, for combating more advanced problems, specialised knowledge sources including neural networks and genetic algorithms need to be built. Initially, DARBS will be run and be tested on a small Intranet. It is anticipated that DARBS will subsequently be operated on the Internet so that knowledge sources developed from different parts of world can be connected to the central blackboard server.
Currently, the blackboard server and knowledge source clients of DARBS start and terminate manually. It is proposed to develop a DARBS project manager, which will keep track of the project files and the status. It will also launch and terminate the knowledge source clients automatically. A full-featured graphical user interface will also be built.

7 Discussions and Conclusions

The aim of this project was to develop a blackboard system with distributed computing features. The project employs client/server technology using TCP/IP, allowing the system to be operated on a LAN, WAN or the Internet. DARBS was designed to be adaptable so that various kinds of AI approaches can be docked to DARBS. Each knowledge source client is equivalent to an agent, with its own specialism. DARBS is therefore equivalent to a distributed agent-based system, with the proviso that all communication is via the blackboard. The core of DARBS has been built and tested with satisfactory results. To demonstrate the genericity of the system, DARBS is being applied to two different kinds of AI applications, i.e., for automatically interpreting non-destructive evaluation (NDE) data and controlling plasma deposition processes.

Acknowledgements

This work was funded by the Open University Strategic Research Initiative and the Engineering and Physical Sciences Research Council (EPSRC). The authors are grateful to Mr. Graham Porter of Rolls-Royce for provision of NDT data and advice on its interpretation.

References

[1]. Hopgood, A.A., Woodcock, N., Hallam, N.J. and Picton, P.D. "Interpreting ultrasonic images using rules, algorithms and neural networks" European Journal of Non-Destructive Testing, 2, (1993), 135-149. UK

[2]. Hopgood, A.A. "Rule-based control of a telecommunications network using the blackboard model" Artificial Intelligence in Engineering, 9, (1994), 29-38. UK.

[3]. Hopgood, A.A., Phillips, H.J., Picton, P.D. and Braithwaite, N.S. "Fuzzy logic in a blackboard system for controlling plasma deposition processes", Artificial Intelligence in Engineering, 12, (1997) pp 253-260. UK.

[4]. Engelmore, R.S., Morgan, A.J.: "Blackboard Systems", Addison-Wesley, (1998), ISBN 0-201-17431-6, UK.

[5]. Hopgood, A.A.: "Intelligent Systems for Engineers and Scientists", 2nd edition, CRC Press, (2000), ISBN 0-8493-0456-3, Boca Raton, FL.

[6]. Matthew N, Stones R "Beginning Linux Programming", Wrox Press Ltd., (1999), ISBN 1-861002-97-1. UK.

---

[1] Poplog grew out of the Pop-11 language produced at Sussex University derived from Pop2 language, originally developed at Edinburgh University in the late 1960s and early 1970s. Poplog provides support for multi-paradigm software development in a rapid prototyping environment. It has been widely used for AI applications and research. Detailed information can be found from http://www.cs.bham.ac.uk/research/poplog/poplog.info.html

[2] IP (Internet Protocol) handles the routing from one computer to another, where TCP (Transmission Control Protocol) handles sequencing, flow control and retransmission to ensure successful delivery [6].