Mr. David G. Goldstein
2900B Cliffridge Lane
Fort Worth, TX 76116
Affiliation: University of Texas, Arlington

PRAIS: Distributed, Real-time Knowledge-Based Systems Made Easy

This paper discusses an architecture for real-time, distributed (parallel) knowledge-based systems called the Parallel Real-time Artificial Intelligence System (PRAIS). PRAIS strives for transparently parallelizing production (rule-based) systems, even when under real-time constraints. PRAIS accomplishes these goals by incorporating a dynamic task scheduler, operating system extensions for fact handling, and message-passing among multiple copies of CLIPS executing on a virtual blackboard. This distributed knowledge-based system tool uses the portability of CLIPS and common message-passing protocols to operate over a heterogeneous network of processors.

I. Introduction

"Real world", and especially real-time, artificial intelligence (AI) is an ideal application for parallel processing. Many problems including those in vision, natural language understanding, and multi-sensor fusion entail numerically and symbolically manipulating huge amounts of sensor data. Reasoning in these domains is often accomplished via specialized computing resources which are often (1) very difficult to use, (2) very costly to purchase (as in the $250,000 - $2,000,000 PIM [GL]), and (3) guarantee only fast- not guaranteed - performance.

This paper introduces PRAIS, the Parallel Real-time Artificial Intelligence System, a cost-effective approach to parallel and real-time computing. PRAIS embeds the 'C' Language Integrated Production System (CLIPS) into a blackboard architecture with artificial intelligence specific operating system extensions and standard communication mechanisms to provide a flexible development environment for distributed knowledge-based systems. The goals of PRAIS is to simplify parallelization, increase portability, and maintain a consistent knowledge representation throughout the system. Accomplishing these goals should dramatically lower the costs of developing and using sophisticated artificial intelligence software.

II. Blackboard Architectures

The blackboard architecture [Nii86] has probably been the most successful architecture for addressing complex problems. This architecture features multiple, independent knowledge sources (KS's) each of which reasons about a portion of the problem. Knowledge sources share a global data structure (the blackboard) to share information, in an analogy to experts examining data and hypothesizing solutions on an actual blackboard.

The blackboard architecture has been adopted for several reasons. First, each knowledge source has its own knowledge-base (KB, a database of knowledge driving reasoning), thereby partitioning the system's knowledge, reducing rule interactions, and making the system easier to understand and program. Blackboards also facilitate hierarchical problem-solving; results from lower level knowledge sources can be used to drive the reasoning of higher level knowledge sources.

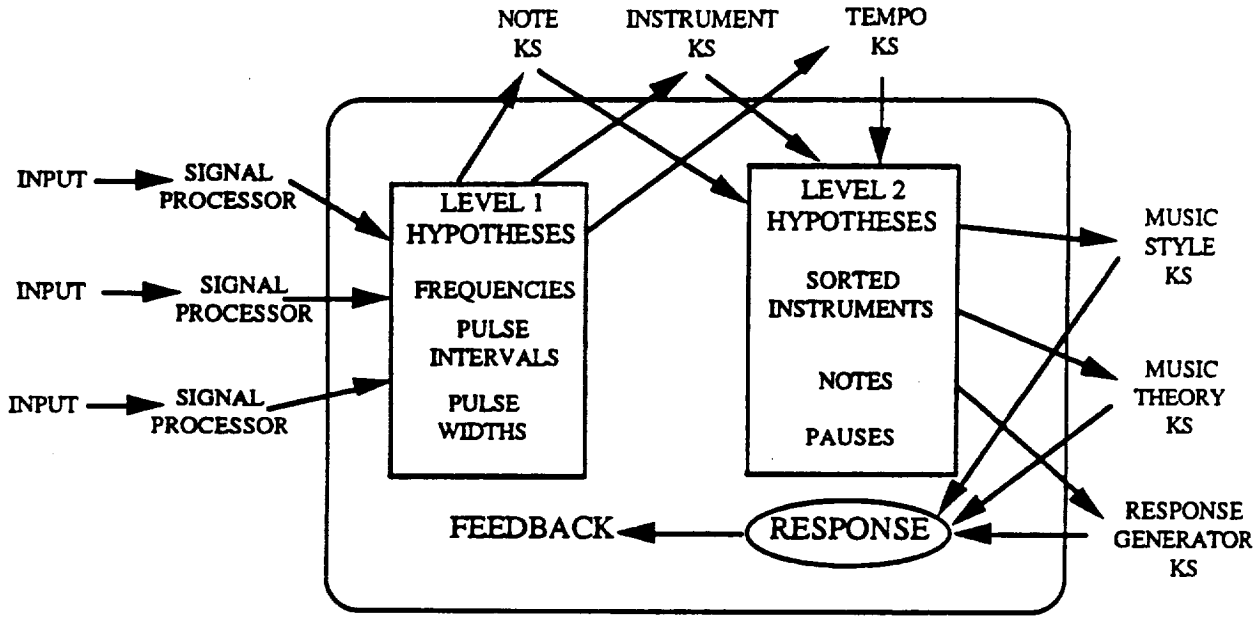
This hierarchical development of hypotheses is very useful, especially useful for problems where disparate data is encountered from multiple sources (e.g. vision, multi-sensor fusion).

The execution of knowledge sources are typically controlled by an external mechanism which activates knowledge sources based upon the blackboard's current state. The control module would normally be quite complex, since decisions it would have to make would include on which processor, for how long, and on what data a given knowledge source should execute. However, PRAIS simplifies control by determining during compilation what, when and for how long each knowledge source needs to execute, so no artificial mechanism for knowledge source activation is required. The distributed nature of the processing is accomplished by simply communicating facts asserted via either (1) a global memory, (2) messages, or (3) in the local fact database.

An illustration of a real time blackboard system for music generation is depicted in Figure 1. At any given time the system might receive a variety of auditory inputs. These inputs are examined by signal processing resources to extract and place on the blackboard primitives such as frequencies, pulse widths and pulse intervals. These primitives are then used by other processors to determine notes, "instruments", pauses, and durations, which are in turn combined to ascertain tempos, progressions, chords. At the highest levels of processing these deductions are combined with music styles, artistic profiles, scores and music theory to predict future sensor inputs and generate appropriate auditory output.

A distributed blackboard permits dedicated processors to handle specific tasks: sensor data can be filtered by signal processors, numeric computations on conventional processors, and symbolic reasoning on LISP machines. Information from one level of the hierarchy can be used at other levels of the hierarchy, and processors at different levels can explore different granularities of a solution in parallel.

Figure 1 - Hierarchical Blackboard Processing



III. Data Representations

The embedding of CLIPS in a blackboard architecture provides a tremendous degree of flexibility. However, choosing the proper data representation is possibly the most crucial aspect of any large KBS because as systems grow in size:

(1) data interactions become more subtle and difficult to predict,

(2) the entire collection of data may not be observable, and

(3) organizing even the initial state may become complicated.

Therefore PRAIS considers parallelization from a data-oriented perspective; facilitating CLIPS rule development drives the system's design. Productions in PRAIS appear almost identically as they do in CLIPS (see Figure 2). These productions are modified only to enhance real-time processing by adding the "importance" definition and a list of

Figure 2 - Production Format

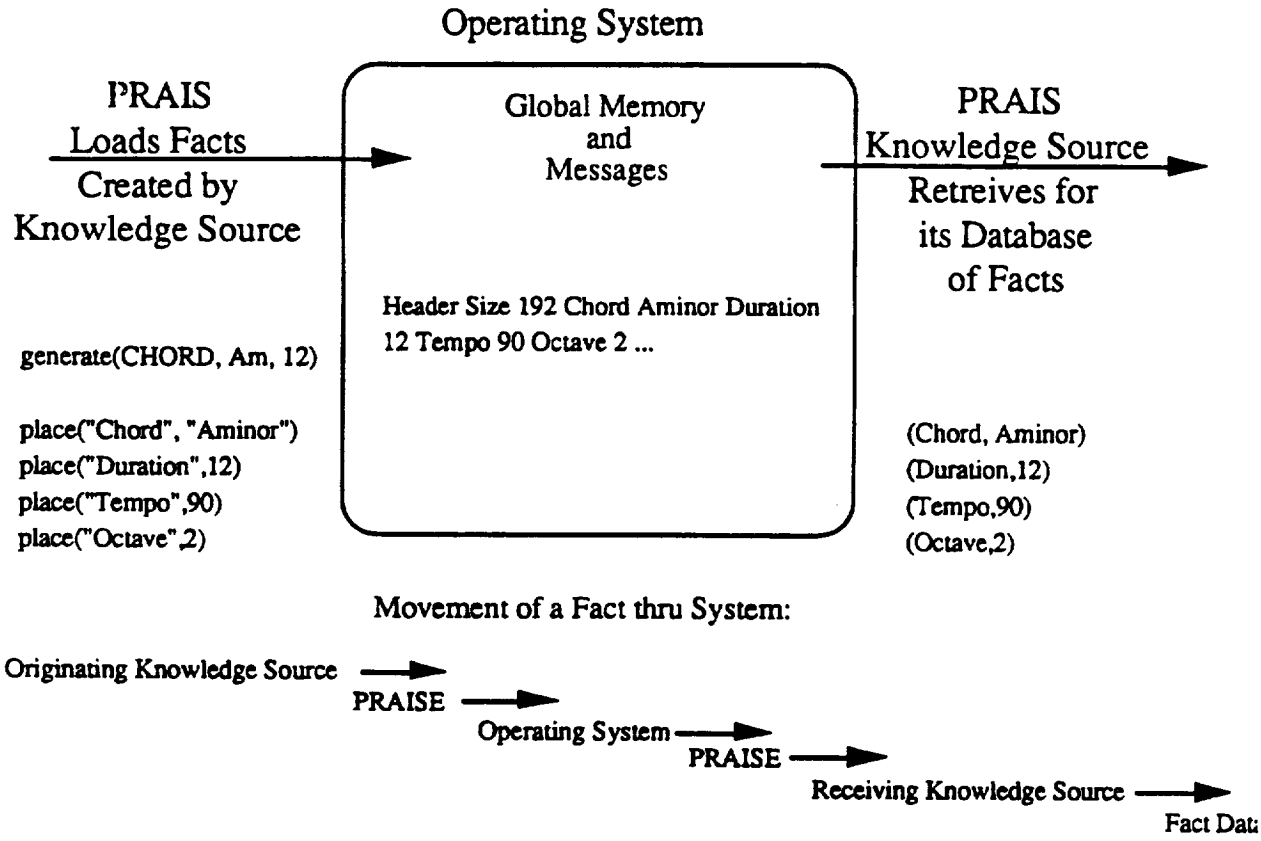
```
(defrule {rule-name}
 (salience {set of (times, priorities)})
 (importance {mandatory/optional/dropable})
  ({left-hand-side patterns})
=>
  ({right-hand-side actions})
)
```

salience values at specified times. Already developed CLIPS code can be used in PRAIS without change. Also, as information is used thruout the system, a syntax is used which resembles CLIPS facts as closely as possible (see Figure 3). This simplification of representation is especially useful in developing complicated, mixed-language, multi-platform applications; CLIPS is especially useful in such endeavors, since it supports 'C', FORTRAN, and Ada [CG].

IV. Operating System Extensions

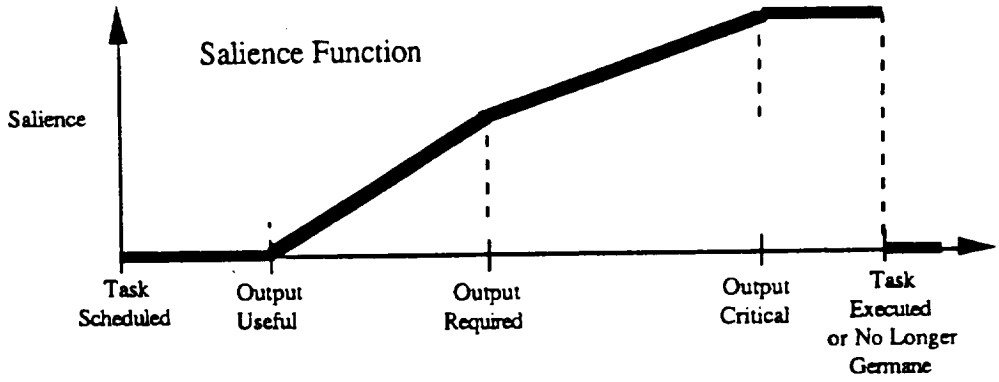
PRAIS provides real-time control, reasoning specification and interruption, and process migration of reasoning tasks as extensions to the operating system . The real-time control mechanisms incorporate salience functions, generated at compile-time, which dynamically reflect the system's current state by considering the timeliness of a given task/rule. A task's salience is initially low, and increases as the task becomes more important, until it become mandatory (see Figure 4). Untimely or lesser important tasks can be dropped by the system to provide more processing power for more important tasks. This allows the system to prioritize an approaching anti-tank missile over determining the optimal path across current terrain.

Figure 3 - Uniform Knowledge Representation



A producer/consumer model has been adopted (since facts typically travel by messages) and facts migrate with a process capable of reasoning. If processor A is overwhelmed with the amount of reasoning it has to perform it broadcasts a request for assistance. If processor B is the least loaded processor in the system, it is also the most likely to respond soonest, and so the first responding task receives both the necessary rules and facts to perform the reasoning task, performs the reasoning, and sends the results to processor A. Other extensions

Figure 4 - Time-Varying Salience Function



include signal-based functions to interrupt reasoning at any time and remote procedure calls to transmit information or to control the amount of time with which a knowledge source may reason.

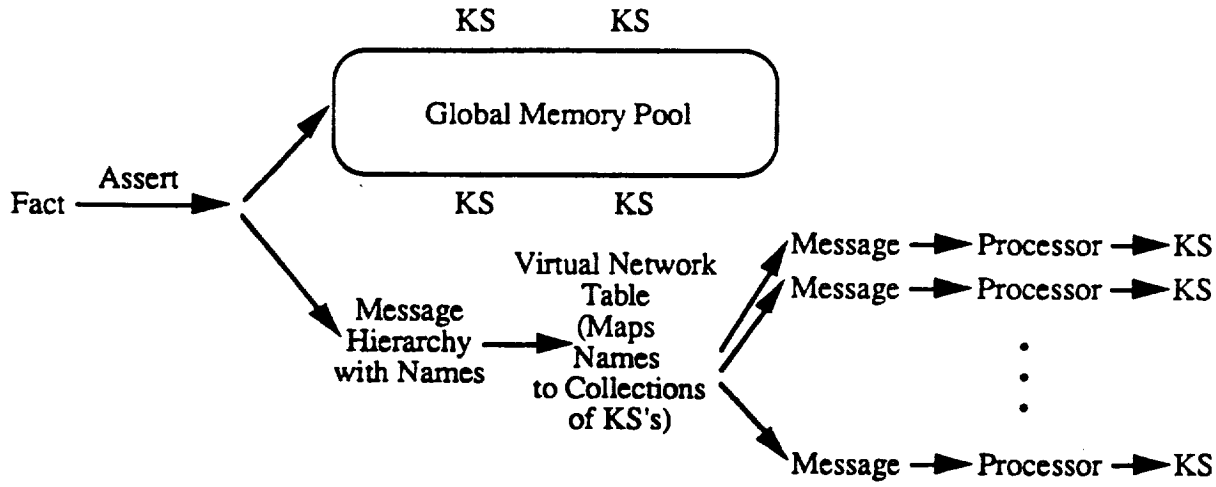
V. Communication Mechanisms

PRAIS provides coarse-grained parallel execution based upon a virtual global memory. PRAIS is also both language and system independent, simply providing the user with a global assert command to enact parallel processing. The system is economical since: (1) relatively few changes must be made to parallelize existing software, (2) inexpensive, commonly available processing resources can be used, and (3) few hardware-specific details must be considered by users. Because PRAIS is easy for its users to work with and will operate on a variety of platforms, PRAIS offers inexpensive parallelism.

PRAIS also has a variety of features that make it appropriate for distributed knowledge-based system development. First, a deployed knowledge source communicates via message-passing to hierarchies of names knowledge sources (see Figure 5). By partially ordering the classes of message recipients communications can be minimized while: (1) replicated copies of knowledge sources can be treated uniformly, (2) processing tasks can be referenced by classes of knowledge sources

(without refering to processors) and (3) virtual communications networks can be established.

Figure 5 - Passing Facts Among Knowledge Sources



Another enhancement is that tasks can be forked to use either the dynamically changing global memory or to use the global memory available at the time of the fork, without incorporating any future updates. This feature is important in applications such as game-playing and certain types of simulations where all relevant details up to the time of the fork are important, but any future details would corrupt the hypothetical universe under consideration.

Other functions incorporated in the system include load balancing, fault-detection, and fault-recovery algorithms.

VI. Status and Results

PRAIS is currently being developed under the auspices of the University of Texas at Arlington. At the time of this writing the knowledge-based system shell has been modified for real-time processing with portions of the dynamic scheduling implemented. The

distributed communication mechanisms have been implemented via sockets, but is being transferred to TCP/UDP datagram RPC's for hardware transparency (via XDR) and to accommodate an unlimited number of processors. Aspects of the system that are either not implemented or are untested include those for fault recovery recovery and insufficient memory detection. It is hoped to port the system to a supercomputer as funding becomes available.

Metrics that have been used to evaluate the system include thruput, efficiency, message load, and "quality" of reasoning which has been interrupted. As the actual immediate application is real-time sensor processing, sensor inputs missed is also an extremely important. Finally, as the the system's goal is cost-effective parallel processing (via cycle-stealing from underutilized processors transparently to the programmer), programmability and maintainability are two highly desired qualities. These qualities are estimated by "system observability", or how easily a person can understand the system. A mathematical representation for system observability is currently being developed, and will hopefully be presented at next year's International Joint Conference on Artificial Intelligence.

As of the time of this writing, the application on which this research has been tested is a high-fidelity simulation featuring many parameters which themselves be connected to sensors or even higher fidelity simulations. The speedup was such that, while efficiency was not as high as hoped, the thruput provided surpassed the benchmark (running on a multi-processing mini-supercomputer) was exceeded by five Sun workstations by over 50%.

VII. Conclusions

PRAIS offers a variety of features including:

- heterogeneous hardware capability,
- real time control via dynamic saliences, and
- incorporating 'C', 'C++', FORTRAN, and Ada, with productions.

PRAIS strives to lessen the user's effort needed to build a parallel, real time KBS by incorporating many knowledge-base system functions as extensions to the operating system.

Actual knowledge-based systems often fail simply because of cost; a more cost-effective approach for developing parallel, real time knowledge-based systems is crucial for bringing AI to the "real world". Systems requiring AI are typically very complex, so sophisticated tools providing simpler solutions must be used to reduce programming costs by .

References

[CG88d] C. Culbert and J. Giarrantano, CLIPS Reference Manual Version 4.2, Artificial Intelligence Section Lyndon B. Johnson Space Center, Houston, Texas, April 1988.

[GL] A. Gupta and T. Laffey, Real-Time Knowledge-Based Systems, Tutorial: MA3, Eleventh International Joint Conference on Artificial Intelligence, Detroit, 1989.

[Kit] J. Kitfield, Military Forum, National Journal, Inc., Washington, D.C., July 1989, pp. 28-35.

[Nii86] P. Nii, "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures", The AI Magazine, Summer 1986, pp. 38 - 53.