NON-VON's Applicability to Three AI Task Area[1]

David Elliot Shaw
Department of Computer Science
Columbia University

ABSTRACT

NON-VON is a massively parallel machine constructed using custom VLSI chips, each containing a number of simple processing elements A preliminary prototype is now operational at Columbia University The machine is intended to provide highly efficient support for a wide range of artificial intelligence and other symbolic applications This paper briefly describes the current version of the NON-VON machine and presents evidence for its applicability to the execution of OPS5 production systems, a number of low- and intermediate-level computer vision tasks, and certain "difficult" relational algebraic operations relevant to knowledge base management Analytic and simulation results are presented for a number of algorithms The data suggest that NON-VON could provide a performance improvement of as much as two to three orders of magnitude over a conventional sequential machine for a wide range of AI tasks

1 Introduction

A strong case can be made that the most formidable challenges now facing artificial intelligence researchers have little to do with problems of computational efficiency Indeed, the development of high-performance hardware is clearly not a *sufficient* condition for the implementation of systems having human-like cognitive capacities A number of researchers now believe, however, that as progress is made in understanding and mechanizing intelligent behavior, the availability of such hardware may ultimately prove to be a *necessary* condition in the case of certain AI tasks of practical importance

Several researchers [Fahlman, 1980, Shaw, 1980, Hilis, 1981; Stolfo and Shaw, 1982, Moto-Oka and Fuchi, 1983; Deenng, 1984] have proposed machines intended to accelerate various operations relevant to one or more AI applications The problem is complicated, however, by the fact that many integrated AI systems will ultimately require the performance of a number of *different* computationally intensive operations, ranging from the various signal processing algorithms employed in low-level computer vision and speech understanding through a wide range of inferencing and knowledge base management tasks that may be involved in high-level reasoning

The central goal of the NON-VON project is the investigation of massively parallel computer architectures capable of providing significant performance and cost/performance improvements by comparison with conventional von Neumann machines in a *wide range* of artificial intelligence and other symbolic applications In this paper, we provide some evidence of NON-VON's range of applicability within the space of AI problems by briefly describing certain algorithms, asymptotic results, analytical performance projections, and instruction-level simulation data involving three distinct AI task areas rule-based inferencing, computer vision, and knowledge base management

The following section provides a brief overview of the NON-VON machine family In Section 3, we report the results of a detailed projection of the machine's performance in executing OPS5 production systems, based on data derived from Gupta and Forgy's investigation of six existing production systems at Cargnegie-Mellon University [Gupta and Forgy, 1983, Gupta, 1984] In Section 4, we review a number of low- and intermediate-level computer vision algorithms for the NON-VON machine that have been developed and simulated at the functional and instruction levels. Section 5 presents the results of a projection of NON-VON's performance on certain AI-relevant database management benchmarks formulated by Hawthorn and DeWitt [1982], using the analytical techniques employed by those researchers The final section summarizes our results and attempts to characterize the essential architectural and algorithmic principles which would appear to be responsible for NON-VON's strong performance in these superficially dissimilar AI application areas

[1]This research was supported in part by the Defense Advanced Research Projects Agency, under contract N000390-84-C-0165 bv the New York State Center for Advanced Technology in Computers and Information Systems at Columbia University, and in part by an IBM Faculty Development Award.

2 The NON-VON Architecture

The name NON-VON is used to describe a family of massively parallel machines that have been under investigation at Columbia since 1980 The top-level organization of the current version of the general NON-VON architecture is shown in Figure 1 A number of distinct machine configurations have been defined and evaluated, however, based in part on the fact that some of the applications that have been investigated require only a subset of the subsystems comprised by the general NON-VON machine, and in part on the evolution of the architecture over time in response to experimental findings An overview of the NON-VON machine family is provided in [Shaw, 1984]

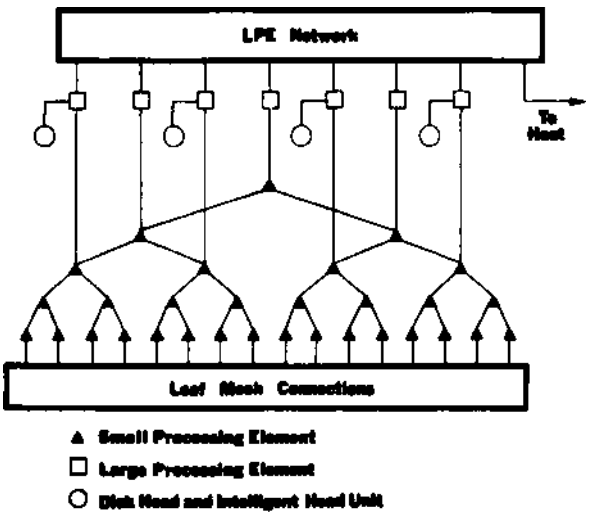

Figure 1: The NON-VON Machine

Central to all members of the NON-VON family is a massively parallel active *memory* The active memory is composed of a very large number (as many as a million, in a full-scale "supercomputer" configuration, and between 4K and 32K in the low-cost configurations evaluated in this paper) of simple, area-efficient *small processing elements* (SPE's), which are implemented using custom VLSI circuits. The most recently fabricated active memory chip contains eight 8-bit processing elements, a detailed description of this chip can be found in [Shaw and Sabety, 1984] Each SPE comprises a small local RAM[2], a modest amount of processing logic, and an *I/O switch* that permits the machine to be dynamically reconfigured to support various forms of inter-processor communication

In order to maximize circuit yield, each SPE was fabricated with only 32 bytes of local RAM in the current working prototype; in a production version of the machine, however, each SPE would probably contain the maximum amount of local RAM supported by the instruction set, which is 256 bytes.

In the current version of the general NON-VON machine, the SPE's are configured as a complete Unary tree whose leaves are also interconnected to form a two-dimensional orthogonal mesh Each node of the active memory tree, with the exception of the leaves and root, is thus connected to three neighboring SPE's, which are called the parent, *left child,* and *right child* of the node in question (Aspects of the chip-, board-, and system-level packaging of the active memory tree are discussed in [Shaw, 1985]) Each leaf is connected to its parent and to its four mesh-adjacent SPE's, which are called its *north, south, east,* and *west* neighbors In addition, the I/O switches may be dynamically configured in such a way as to support communication between any pair of nodes that would be adjacent in an inorder enumeration [Knuth, 1969] of the active memory tree

NON-VON programs are not stored within the small RAM associated with each SPE, but are instead broadcast to the active memory by one or more iarg-e *processing elements* (LPE's), each based on an off-the-shelf 32-bit microprocessor (a Motorola 68020, in the current design) having a significant amount of local RAM In the simplest NON-VON configuration, which was also the first to be implemented, the entire active memory operates under the control of a single LPE[3] which broadcasts instructions through a high-speed interface called the active *memory controller* for simultaneous execution by all enabled SPE's This simple configuration thus restricts NON-VON's operation to what Flynn [1972] has called the single *instruction* stream, *multiple data, stream* (SIMD) mode of execution

The current version of the general NON-VON design, however, provides for a number of LPE's, each capable of broadcasting an independent stream of instructions to some subtree of the active memory tree, as first described in [Stolfo and Shaw, 1982] Specifically, each SPE above a certain fixed level in the active memory tree is connected to its own LPE and active memory controller, which is capable of sending an independent instruction sequence to control the corresponding active memory subtree The LPE's in the general machine are interconnected using a high bandwidth, low latency two-stage interconnection scheme called a *rootpoint network* [Shaw, 1985a] The incorporation of a number of communicating LPE's gives the general NON-VON architecture the capacity for *multiple instruction stream, multiple data stream* (MIMD) and *multiple SIMD* (MSIMD) execution, multi-tasking applications, and multi-user operation (Further discussion may be found in [Shaw, 1984])

The prototype now in operation uses a VAX 11/750 in place of the Motorola 68020.

The general NON-VON architecture also includes a *secondary processing subsystem* based on a bank of "intelligent" rotating storage devices Specifically, each LPE at a particular, fixed level within the active memory tree is connected to a single-head Winchester disk drive, permitting high-bandwidth parallel I/O Associated with each disk head is a small amount of logic capable of dynamically examining and performing certain simple computations (hash coding, for example) on the data passing beneath it (Further discussion may be found in [Shaw, 1980] and [Shaw, 1982])

Only a subset of the features of the general NON-VON architecture are in fact required to perform the benchmark tasks described in this paper The particular configuration assumed for each task will be described in the three sections in which the results themselves are presented

One of the most important aspects of the NON-VON active memory is the highly efficient hardware support it provides for a set of associative processing primitives that have proven useful in a wide range of AI applications The machine's very fine granularity allows processing elements to be associated with individual data elements, offering the user the capabilities of an unusually powerful and flexible content-addressable memory Hardware features of the active memory support the massively parallel manipulation of data elements of arbitrary length Large data elements may span more than one SPE, or several small ones may be packed within a single SPE It is expected that systems software will ultimately hide the detailed mapping between processing elements and data elements from the programmer, allowing the user to deal with such abstractions as sets of "intelligent records" which may be thought of as capable of manipulating their own contents in parallel

Among the hardware features provided within the SPE's to support these associative processing capabilities are

1. Logic allowing a string broadcast by the controlling LPE to be very rapidly matched against the contents of all SPE's in parallel Using the special NON-VON string matching instructions, a single-instruction inner loop for string matching may be implemented, allowing matching to occur at approximately 3 million bytes per second per SPE in the new prototype now under construction A large active memory should thus be capable of matching at an aggregate rate of more than a trillion bytes per second

2 The inclusion of both eight-bit and one-bit registers and RAM, allowing sets of data elements to be conveniently marked and manipulated in parallel in various ways Such operations are facilitated by the ALU, which provides highly efficient support for logical operations on one-bit flags

3 A special one-bit register called the ENABLE flag All SPE's in which the ENABLE flag is set to zero are *disabled,* and ignore all instructions until the subsequent execution of an ENABLE instruction, which "awakens" all SPE's This mechanism, variants of which have been used in a number of other parallel machines, facilitates parallel operations on selected sets of data elements identified on the basis of some aspect of their contents

4 A multiple match resolution instruction called RESOLVE, which may be used to mark a single member (the one which would occur first in an inorder traversal of the active memory tree) of a specified set of SPE's Together with the REPORT instruction, which transfers data from a single enabled SPE to the controlling LPE, this technique may be used to sequentially enumerate the members of an arbitrary marked set of data elements in time linear in the cardinality of the set

These associative processing mechanisms are employed in a number of the algorithms whose performance is reported in this paper Further discussion of the use of NON-VON's active memory to support various associative processing techniques may be found in [Shaw, 1982]

3 Production Systems

After several decades of research on artificial intelligence, rule-based *production systems* have emerged as one of the most important and widely employed tools for the implementation of expert systems and other AI software While the modularity, perspicuity, and ease of modification and augmentation which are characteristic of rule-based systems offer a number of potential advantages in the execution of many large-scale AI tasks, their use in such applications is in some cases very costly in execution time. This poses particular problems in the case of real-time applications and other tasks characterized by severe and inflexible time constraints The possibility

of using parallel hardware to accelerate the execution of production systems is thus of considerable interest

Forgy [1980] considered the problem of executing production systems in parallel on the ILLIAC IV, but was forced to significantly modify the production system paradigm in order to obtain reasonable performance Stolfo and Shaw [1982] subsequently proposed a highly parallel machine called DADO, which was intended specifically for the execution of production systems; an early prototype of the DADO machine is presently operational More recently, members of the DADO project have investigated a number of issues related to languages and algorithms for the parallel execution of production systems [Stolfo, 1984]

As a first step in evaluating NON-VON's performance in the domain of rule-based systems, we have implemented and analyzed an algorithm [Hillyer and Shaw, 1984] for the parallel execution of production systems implemented using the language OPS5 [Forgy, 1981], which was developed by researchers at Carnegie-Mellon University OPS5 was chosen as the vehicle for our investigations into parallel execution of production systems for several reasons

1 It is widely known, and has been evaluated favorably by other researchers [Hayes-Roth, et al , 1983]

2 It has been used to build a large and successful commercial production system [McDermott, 1980]

3 Static and dynamic characteristics of six substantial OPS5 production systems have already been measured [Gupta and Forgy, 1983]

4. Its speed can be increased significantly by parallel execution, despite the fact that the language was designed for sequential processing.

The NON-VON OPS5 algorithm may be regarded as a parallel version of Forgy's Rete Match [1982] algorithm, which exploits the observation that firing an OPS5 production causes only a few changes to working memory, and that these changes have few effects on the conflict set. This observation may be exploited, even on a sequential machine, if the production system is compiled into a dataflow graph, with state information saved at each node during execution. A change to working memory is entered into initial nodes of the graph Consequent state changes then propagate through the graph, updating information stored in intermediate nodes State changes in terminal nodes of the graph represent changes to the conflict set.

The execution cycle for an OPS5 production system has three phases match, select, and act In the Rete Match algorithm, the match phase has two steps First, *intra.-condition tests* are executed to check that attributes in a working memory element satisfy the specified relational operators, and that variables occurring more than once in a condition element are bound consistently Subsequently, *inter-condition tests* are performed to verify consistent binding of variables across multiple condition elements in a production's left-hand side The NON-VON OPS5 algorithm, which was implemented and tested by Bruce Hillyer using the NON-VON instruction-level simulator, achieves its performance and cost/performance advantages primarily by accelerating the execution of these two components of the match phase in the case of working memory additions, and by replacing them with a more highly parallel associative step in the case of working memory deletions

The NON-VON OPS5 algorithm is based on an algorithm by Gupta [1984] (his Algorithm 3) This algorithm was designed for the DADO machine [Stolfo and Shaw, 1982], in a prototype configuration consisting of 1023 identical "off-the-shelf" 8-bit microprocessors The NON-VON algorithm assumes 33 LPE's (or 32 LPE's and one dedicated host machine), each of which is somewhat more powerful than a DADO PE, and 16K SPE's, which make possible a greater degree of associative parallelism The 32 "non-host" LPE's are assumed to be attached at the fifth level of the active memory tree, the LPE's that would be associated with the first through fourth levels in a fully general NON-VON machine are not required for the execution of this algorithm The high-bandwidth LPE network is also not required, and could be replaced with a simple bus without significantly degrading performance in the production systems application

In the NON-VON OPS5 algorithm, intra-condition testing is performed in the active memory using two massively parallel SIMD computation steps The first step simultaneously evaluates individual terms of all condition elements. The second step determines the satisfaction of all condition elements in a single parallel communication step, requiring time proportional to the number of terms in the longest

condition element. The synchronous nature of SIMD execution was found not to limit the rate of processing in this phase

The more limited potential for parallelism embodied in the inter-condition testing step is captured using a multiple-SIMD execution discipline, in which independent instruction streams are broadcast to 32 separate active memory subtrees by the 32 LPE's Pure SIMD processing would have been a serious constraint during this portion of the execution, since evaluation of the Rete dataflow graphs presents a high degree of data sensitivity The use of active memory subtrees, on the other hand, significantly enhances the power of the LPE's by providing a fast associative search capability The select and act phases, which have little inherent parallelism in OPS5, are performed sequentially by a single LPE (or by an attached host processor), although a modest speedup is obtained by overlapping a portion of these computations with the subsequent match phase

An experimental compiler and runtime system for the execution of OPSS on a one-LPE NON-VON has been written and tested on an instruction-level simulator In order to predict the algorithm's performance when executing real production systems, its running time was calculated based on measurements obtained by Gupta and Forgy [1983] of the static and dynamic characteristics of six actual production systems, which had an average of 910 inference rules each By counting the number of NON-VON instructions, it has been possible to determine the number of clock cycles required for each of the SIMD processing steps under the assumptions implicit in Gupta and Forgy's data By adding approximate overhead values for non-overlapped execution in the LPE's[5], we have been able to predict with reasonable accuracy the time required per production firing cycle[6], and hence the rate of production system execution on NON-VON

According to these calculations, the NON-VON configuration outlined above should require 1108 microseconds per production firing, yielding an execution rate of 903 productions per second By way of comparison, A LISP-based OPS5 interpreter executing the sequential Rete Match algorithm on a VAX 11/780 typically fires between 1 and 5 rules per second, while a Bliss-based interpreter executes between 5 and 12 productions per second [Gupta, 1984 (personal communication)] The NON-VON machine configuration assumed in these calculations would embody 2048 custom chips, each containing 8 SPE's, and about 660 chips for the 33 LPE's The hardware cost of such a system is projected to be somewhere between that of a VAX 11/780 and a DECSystem-20 NON-VON's cost/performance ratio in executing OPS5 production systems is thus estimated to be approximately two orders of magnitude better than that of a conventional sequential machine in practice Furthermore, our analysis suggests that the machine's relative performance advantage should increase as the production system becomes larger

A detailed discussion of the assumptions underlying the above projections, and of the precise manner in which our results were derived, can be found in [Hillyer and Shaw, 1984]

4 Computer Vision

A number of operations performed by commercial and experimental image understanding systems are extremely time-consuming when executed on a conventional sequential machine For this reason, substantial efforts have already been made to develop special-purpose machines and algorithms adapted to various computer vision tasks The machines described in the literature may, for the most part, be divided into four general architectural categories

1 *Pipelined architectures* such as the NBS Pipelined Image Processing Engine [Kent, Shneier and Lumia, 1984] and the Cytocomputer [Sternberg, 1983], in which the task at hand is typically divided into a number of functions and each stage in the pipeline is configured by a control unit to perform one function

2 *Mesh-connected architectures* [Unger, 1958], such as CLIP4 [Duff, 1976], MPP [Potter, 1983], and the ICL DAP [Marks, 1980], which are based on a rectangular array of (typically one-bit) processing elements, each connected to its nearest four or eight neighbors, and each corresponding to a either a single pixel or a small rectangular

This can be improved to time logarithmic in the number of terms in the longest condition element with a worst-case 50% decrease in memory utilization, using techniques closely related to the allocation schemes for database records described by Shaw and Hillyer [1982].

The figures given for host and LPE processing are estimates, unlike the SPE figures, which are derived from actual code.

We have assumed that the only actions in the right-hand sides of productions are additions, deletions, and modifications of working memory elements. The right-hand side of a production expressed in OPS5 can cause arbitrarily large amounts of I/O and can call any function written in LISP, but the time consumed by such operations does not provide any information about the performance of the production system inferencing engine.

3 *Multiprocessor architectures,* examples of which include PASM [Siegel, et al, 1981] and ZMOB [Kushner, et al, 1982], in which a number of general-purpose microcomputers cooperate to solve a given image understanding task, communicating through some form of processor interconnection network.

4 *Hierarchical architectures,* including *cone* and *pyramid* machines [Uhr, 1978; Hanson and Riseman, 1980, Dyer, 1981, Tanimoto, 1983], in which the original image is progressively reduced through the application of successive transformations in a manner analogous to that apparently employed in the human visual system

The general NON-VON architecture provides efficient support for most of the parallel vision algorithms typically executed on each of the last three classes of architectures, in addition, many of the algorithms executed on pipelined vision machines may be transformed into SIMD algorithms amenable to execution within NON-VON's active memory While we expect NON-VON to prove highly useful for all levels of computer vision applications, from low-level signal processing and the extraction of primitive geometric properties through high-level object recognition and scene analysis, our work to date has concentrated largely on low- and intermediate-level computer vision tasks

A number of image understanding algorithms have been developed and tested using a functional simulator, and in some cases, the NON-VON machine instruction level simulator In particular, algorithms have been developed for

1 Image correlation

2 Image histogramming

3 Image thresholding

4 Union, intersection, and difference of two binary images

5 Connected component labeling

6 Euler number of connected components

7 Component area

8 Component perimeter

9. Component center of gravity

10 Component eccentricity

11 The Hough transform

12 The "moving light display" problem

These tasks were chosen to span several levels of image understanding applications, and to assess NON-VON's applicability to a number of different kinds of operations arising in the course of constructing integrated vision systems Preliminary analytical and simulation results suggest that NON-VON should provide significant performance and cost/performance advantages over sequential machines for each of these tasks, and should in a number of cases improve on the best results reported in the literature for special-purpose vision architectures and other highly parallel machines

Two hierarchical data structures that may be implemented naturally on the NON-VON machine are employed in the NON-VON algorithms for these tasks The first is the *multi-resolution pyramid* [Tanimoto and Klinger, 1980], which represents the original (binary or gray-scale) image at a number of different levels of detail, each containing one quarter of the number of pixels present at the next highest level of detail Multi-resolution pyramids are implemented on NON-VON by storing the original image in the (mesh-connected) leaf SPE's and the less detailed levels in the internal SPE's The latter process is based on a straightforward mapping of the natural quartic tree interpretation of the multi-resolution pyramid onto the active memory's binary tree structure

The second data structure used to represent images in the NON-VON machine is a variant of the *quadtree* called the *binary image tree,* which was originally proposed by Knowlton [1980] as an encoding scheme for the compact transmission of image data The original binary image is again stored in the leaf SPE's Internal nodes, however, can take one of three values, which may be thought of as "black", "white" and "gray" An internal node has the value black if its two children are both black, white if its two children are both white, and gray otherwise This data structure is used in most of the NON-VON algorithms that operate on binary image data

A number of aspects of the NON-VON machine architecture, exploited as part of several frequently used algorithmic techniques, are responsible for the machine's performance and cost/performance advantages in computer vision applications The simplest technique, which has also been employed on

the original image. The connected component algorithm enumerates all of these black rectangles (using the RESOLVE and REPORT instructions) one by one in order of decreasing size (or equivalently, of increasing tree level)

As each such rectangle R_i is enumerated, it is assigned a new component number if not already labelled Any rectangle R_j. that is adjacent to R_i is then assigned the same label, as are all other rectangles previously assigned the same component number as R_j (Note that this last operation requires only a single broadcast and assignment step, independent of the number of rectangles having the same component number as R_j) Since only one pass through the set of black rectangles in the binary image tree is required, Ibrahim's algorithm executes in time proportional to the number of rectangles, which in real images is much smaller than the number of pixels (Details of this and other geometric algorithms for NON-VON may be found in [Ibrahim, 1984])

The importance of NON-VON's mesh connections in computer vision applications is illustrated by such low-level signal processing operations as image correlation and various types of two-dimensional filtering The NON-VON algorithms for these operations rely on a considerable amount of communication among adjacent and nearly-adjacent neighbors in the mesh Although Ibrahim [1984] has developed techniques to perform such operations on a "pure" tree machine that tend to minimize the need for such communication, NON-VON's mesh connections make possible a considerable increase in performance

Other vision algorithms make use of the MSIMD execution capabilities of the general NON-VON architecture to partition a problem into a number of subproblems, each of which is solved in a different active memory subtree In many cases, several forms of parallelism are exploited in the same algorithm Although work on high-level image understanding on the NON-VON machine is still in its earliest phases, our experience with the parallel execution of production systems suggests that significant opportunities exist for accelerating such applications as well

5 Knowledge Base Management

A number of "real world" AI applications require the construction and manipulation of large *knowledge bases* Such applications tend to be characterized by the need to access the same data in a number of different ways, often in conjunction with some form of such machines as MPP and the ICL DAP, involves the use of ordinary content-addressable arithmetic and logical operations to perform a single operation in parallel on all pixels This technique is illustrated in its simplest form by the algorithm for image thresholding, in which a gray-scale image is transformed to a binary image by setting all pixels whose intensities exceed a specified threshold value to one, and all other pixels to zero A NON-VON machine of size sufficient to store each pixel in its own processing element is capable of thresholding an entire image in less than two microseconds, independent of the size of the image, by broadcasting the threshold value to all pixels and executing a single parallel comparison followed by two parallel assignment operations, each executed by a separately enabled set of SPE's

Like a number of highly parallel machines developed by other researchers, NON-VON is thus able to achieve a speedup of many orders *of* magnitude, and a cost/performance improvement of approximately three orders of magnitude, by comparison with a VAX 11/780 on simple associative operations of the kind exemplified by image thresholding The same general technique is used in the NON-VON algorithm for image histogramming [Ibrahim, 1984], in which the bounds of a given intensity range are compared in parallel with all pixel values to identify those pixels whose values fall within a given intensity range (Histogramming is often performed prior to the conversion of a gray-scale to a binary image in order to choose an appropriate threshold value)

Unlike the massively parallel mesh-connected machines, however, the tree-structured connections embedded in the NON-VON active memory permit a major reduction in the time required to determine the number of pixels falling within each histogram bin In the NON-VON algorithm, this quantity is computed through an O(log n) level-by-level addition procedure, where n is the number of pixels in the image The running time of the algorithm is further reduced, however, by pipelining this computation for successive intensity ranges, resulting in an $0(b + \log n)$ algorithm, where b is the number of histogram bins The NON-VON histogram algorithm is easily modified to produce a cumulative or normalized histogram

A different use of the active memory tree is illustrated by Ibrahim's connected component algorithm for NON-VON [Ibrahim, 1984] The algorithm begins by constructing the binary image tree representation of the original image data, which requires time logarithmic in the number of pixels It will be recalled that the binary image tree includes explicit representations of black rectangles of various sizes from

inferencing. The need to flexibly access and deductively manipulate large knowledge bases has led a number of researchers to construct logic-based and knowledge-oriented database systems [Gallaire and Minker, 1978, Wiederhold, et al, 1980, Shaw, 1980] whose primitive operations are based on the relational model of data [Codd, 1972] In addition, many knowledge-based systems that do not make explicit use of relational data structures or access primitives nonetheless employ equivalent or closely analogous representation formalisms and operations at some level

Unfortunately, knowledge base management systems tend to rely heavily on such "difficult" operations as the relational join, whose execution may be extremely time-consuming on a sequential machine, particularly in the case of large databases Several researchers [Shaw, 1980, Moto-oka and Fuchi, 1983] have thus proposed using special hardware supporting the rapid execution of such database primitives in constructing systems designed for AI applications Algorithms for a number of database primitives have been developed for the NON-VON machine, including

1 Select

2 Project

3 Join

4 Union

5 Intersection

6 Set Difference

7 Aggregation

8 Various statistical operations

To evaluate NON-VON's applicability to the kinds of database operations most relevant to AI applications, a detailed analysis was performed of the machine's projected performance on a set of benchmark queries formulated by Hawthorn and DeWitt [1982] These authors presented an analysis of the predicted performance of a number of database machine architectures (specifically, RAP [Ozkarahan, et al, 1974], CASSM [Copeland, et al, 1973], DBC [Banerjee, et al, 1979], DIRECT [DeWitt, 1979], CAFS [Coulouns, et al , 1972], and associative disks [Langdon, 1978, Lin, et al, 1976, Slotnik, 1970]) on three relational database queries.

In evaluating NON-VON, we used the same three queries and similar techniques for performance analysis and for the adjustment of machine configuration to control for hardware cost. In particular, we assumed a NON-VON configuration having 4K SPE's (512 8-processor chips), 16 disk drives, and a single LPE -- a configuration comparable in each relevant dimension to that of the other machines considered in the Hawthorn and DeWitt analysis

Hawthorn and DeWitt's first query is a simple relational selection operation Their analysis predicted that all six of the architectures they examined would have roughly comparable performance on this query Our analysis, using the same techniques, predicts that NON-VON should achieve slightly, but not significantly faster performance on this simple selection query The lack of a significant performance difference between NON-VON and the six database machines evaluated by Hawthorn and DeWitt is attributable to the fact that all seven machines are capable of the kinds of simple associative processing techniques required to achieve close to the maximum performance permitted by the aggregate disk transfer rate of their (cost-normalized) banks of disk drives It is on the more demanding queries frequently found in AI-onented applications that NON-VON is able to offer a greater relative advantage Table 1 summarizes our results and those of Hawthorn and DeWitt for the first sample query.

	Response times in seconds	
	Best	Worst
Associative Disks, CAFS	0.089	0.267
CASSM	0.158	0.334
DBC	0.095	0.273
DIRECT	0.066	0.490
RAP	0.066	0.493
NON-VON	0.042	0.267

Table 1: Query 1, Hawthorn and DeWitt

The second query, which may have more immediate relevance to complex knowledge base management applications, involves a relational join operation On this query, NON-VON achieves a performance improvement of approximately an order of magnitude over the six database machines evaluated by Hawthorn and DeWitt, as indicated in Table 2

There are three reasons for NON-VON's surprising speed, even by comparison with DIRECT, which is the fastest of the other database machines on this query The first is that DIRECT stores entire records in its cache when processing, while NON-VON

	Response times in seconds Best	Worst
Associative Disks	5.04	5.90
DBC	5.04	5.90
CASSM	15.50	15.83
DIRECT	2.37	3.24
RAP	7.06	14.18
CAFS	14.48	14.81
NON-VON	0.21	0.67

Table 2: Query 2, Hawthorn and DeWitt

projects over those attributes that are of interest, discarding the rest Second, DIRECT reads data from one of its 8 (faster) disk drives, while NON-VON loads one of the argument relations in parallel through all 16 heads of its (slower) drives (Unlike NON-VON, DIRECT would not benefit greatly from the use of parallel disk transfers, DeWitt and Hawthorn [1982] estimate that parallel disk input would improve the join performance of architectures like DIRECT by only *4%*) Third, DIRECT performs a sequential search through one of the argument relations for each tuple of the other relation it considers, in contrast with NON-VON's associative matching followed by fast enumeration of responders using RESOLVE and REPORT instructions

The final query considered by DeWitt and Hawthorn involves an aggregation operation in which the tuples in the argument relation are divided into groups according to the values of two attributes, and the sum of the values of a third attribute is computed separately for each group Using associative processing techniques (both internal and external), together with the capability for fast addition provided by the active memory's tree connections, NON-VON again achieves significant performance improvements over the other architectures evaluated by Hawthorn and DeWitt While the authors do not explicitly present the times required by the machines they studied for this query, a graph is provided from which these times were estimated Comparative results for this query are presented in Table 3

Further details of the assumptions and analysis underlying the results presented above may be found in [Hillyer, Shaw and Nigam, 1984]

	Response times in seconds Best	Worst
Associative Disks	1.6	2.2
CAFS	1.6	2.2
CASSM	0.8	0.8
RAP	0.8	1.5
DBC	0.7	1.6
DIRECT	0.2	0.5
NON-VON	0.05	0.27

Table 3: Query 3, Hawthorn and DeWitt

6 Conclusions

It is dangerous to attempt to draw firm conclusions about the range of applicability of a novel computer architecture over a wide range of applications on the basis of a relatively small number of benchmark tasks Based on our research to date, however, it would appear that the general NON-VON architecture may offer significant performance and cost/performance advantages over both conventional and, in a number of cases, special-purpose machines in at least three superficially disparate artificial intelligence task areas

In particular, we have presented a number of algorithms, analytic performance projections, and simulation results suggesting that NON-VON is capable of rapid and cost-effective performance on a number of tasks in the areas of production system execution, low- and intermediate-level image understanding, and knowledge base management Although the magnitudes of the projected performance and cost/performance advantages are different for different tasks, performance improvements of as much as two to five orders of magnitude, and cost/performance advantages of as much as two to three orders of magnitude, have been predicted in many cases

Different aspects of the NON-VON architecture appear to be responsible for the machine's advantages in different problem areas It is nonetheless possible to identify a relatively small number of features, several of which are typically operative in the case of any single application, to which the machine's advantages may be attributed

1 The effective exploitation of an unusually high degree of parallelism, which is made possible by the very fine granularity of the active memory

2 The extensive use of broadcast communication, high-speed content-addressable matching, and other associative processing techniques

3. The use of the active memory tree to execute algebraically commutative and associative operations (such as sum and maximum) in logarithmic time

4 The exploitation of other physical and logical interconnection topologies to support a number of problem-specific communication functions

5 The capacity for SIMD, MIMD and MSIMD execution, and for a mixture of synchronous and asyncronous execution within a single algorithm

6 The simplicity and cost-effectiveness with which the machine can be implemented using currently available technology

Although not all of these architectural features are relevant to every AI application, each is exploited in different ways in a number of different algorithms While it is difficult to rigorously explicate the intuitive notion of computational generality, NON-VON would seem to have many of the properties of a rather general machine having a number of widely applicable architectural features, as opposed to an amalgam of special-purpose subsystems, each performing a fixed, application-specific task

It is of course of interest that NON-VON may be able to exceed the performance of the fastest special-purpose machines described in the literature for the execution of some of the tasks considered in this paper Of potentially greater practical significance than NON-VON's superior performance in any one particular task area, however, is the range of diverse AI tasks that would appear to be efficiently executable within a single machine. While there is still insufficient evidence to adequately evaluate the extent to which the NON-VON architecture might serve as the basis for a high-performance "general AI machine", the work described in this paper may represent an early, preliminary step toward the ultimate development of such machines.

ACKNOWLEDGEMENTS

Bruce Hillyer was directly responsible for implementing, simulating, and analyzing most of the database management and production system algorithms discussed in this paper The NON-VON image understanding work we have described is largely attributable to Hussein Ibrahim and John Render More generally, the author wishes to acknowledge the direct and indirect contributions of the NON-VON hardware implementation team, under the direction of Ted Sabety, of Dong Choi, who was responsible for much of the simulation software used to obtain the results we have reported, and of the other Ph D students and research staff members associated with the NON-VON project, without whose efforts this work would not have been possible

REFERENCES

Banerjee, J, R I Baum, D K Hsiao, and K Kannan, "Concepts and Capabilities of a Database Computer", *ACM Transactions on Database Systems,* vol 3, no 4, 1978

Codd, E F , "Relational Completeness of Data Base Sublanguages", in R Rustin (ed), *Courant Computer Science Symposium 6: Data Base Systems,* Prentice-Hall, Inc , 1972

Copeland, G P., G J Lipovski, and S Y W Su, "The Architecture of CASSM A Cellular System for Nonnumeric Processing", *Proceedings of the First Annual Symposium on Computer Architecture,* 1973

Coulouns, G F , J M Evans, and R W Mitchell, "Towards Content Addressing in Databases", *Computer Journal,* vol. 15, no 2, 1972

Deering, M. F , 'Hardware and Software Architectures for Efficient AI", *AAAI-84, Proceedings of the National Conference on Artificial Intelligence,* pp 73-78, August, 1984

DeWitt, D J, "DIRECT--A Multiprocessor Organization for Supporting Relational Database Management Systems", IEEE *Transactions on Computers,* vol. c-28, no 6, June, 1979

DeWitt, D J. and P. B Hawthorn, "A Performance Evaluation of Database Machine Architectures", *Journal of Digital Systems,* 1982

Duff, M J B., "A Large Scale Integrated Circuit Array Parallel Processor", *Proceedings of the IEE*

Conference on Pattern Recognition and Image Processing, pp 728-733, 1976

Dyer, C R, "A VLSI Pyramid Machine for Hierarchical Parallel Image Processing", Proceedings of the IEEE Conference on Pattern Recognition and Image Processing, pp 281-386, 1981

Fahlman, S E , "Preliminary Design for a Million-Element NETL Machine", Technical Report, Department of Computer Science, Carnegie-Mellon University, 1980

Flynn, M J , "Some Computer Organizations and their Effectiveness", IEEE Transactions on Computers, vol c-21, September, 1972

Forgy, C L , "Note on Production Systems and Illiac IV", Technical Report, Computer Science Department, Carnegie-Mellon University, July 1980

Forgy, C L, "OPS5 Users' Manual", Technical Report CMU-CS-81-135, Carnegie-Mellon University, 1981

Forgy, C L, "Rete A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", Artificial Intelligence, September 1982

Gallaire, H and J Minker (eds), Logic and Data Bases, New York, Plenum Press, 1978

Gupta, A, "Implementing OPS5 Production Systems on DADO", Proceedings of the 1984 International Conference on Parallel Processing, Aug 21-24, 1984

Gupta, A and C L Forgy, "Measurements on Production Systems", Technical Report, Carnegie-Mellon Computer Science Department, 1983.

Hanson, A R. and E. M Riseman, "Processing Cones. A Computational Structure for Image Analysis", in S Tanimoto and A Klinger (eds), Structured Computer Vision, Academic Press, 1980

Hawthorn, P B. and D J DeWitt, "Performance Analysis of Alternative Database Machine Architectures", IEEE Transactions on Software Engineering, Vol SE-8, No 1, January 1982, pp. 61-75.

Hillis, W. D, "The Connection Machine", AI Memo No. 646, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, September, 1981

Hillyer, B K, D E Shaw, and A Nigam, "NON-VON's Performance on Certain Database Benchmarks", Technical Report, Department of Computer Science,

Hillyer, B K and David Elliot Shaw, "Execution of Production Systems on a Massively Parallel Machine", Technical Report, Department of Computer Science, Columbia University, September, 1984

Ibrahim, H A H , Image Understanding Algorithms on Fine-Grained Tree-Structured S1MD Machines, Ph D Thesis, Department of Computer Science, Columbia University, October, 1984

Kent, E", M Shneier, and R Lumia, "NBS Pipelined Image Processing Engine", Workshop on Algorithm-Guided Parallel Architectures For Automatic Target Recognition, Leesburg, VA July 1984

Knowlton, K , "Progressive Transmission of Gray-Scale and Binary Pictures by Simple, Efficient, and Lossless Encoding Schemes", Proceedings of the IEEE, vol 68, no 7, July, 1980

Knuth, D E, The Art of Computer Programming, vol 1: Fundamental Algorithms, Addison-Wesley, 1969

Kushner, T, A U Wu, and A Rosenfeld, "Image Processing on ZMOB", IEEE Transactions on Computers, vol 31 , no 10, October, 1982

Langdon, G G , "A Note on Associative Processors for Data Management", Transactions on Database Systems, vol 3, no 2, June, 1978

Lin, C S, D C P. Smith and J M Smith, "The Design of a Rotating Associative Memory for Relational Database Applications", Transactions on Database Systems, vol 1, no 1, March, 1976

Marks, P , "Low Level Vision Using an Array Processor", Computer Graphics and Image Processing, vol 14, pp 287-292, 1980

McDermott, J "RI A Rule-Based Configurer of Computer Systems", Technical Report CMU-CS-80-119, Computer Science Department, Carnegie-Mellon University, April 1980

Moto-oka, T and K Fuchi, "The Architectures in the Fifth Generation Computers", in R E A Mason (ed), Information Processing 83, September 1983

Ozkarahan, E. A., S. A Schuster, and K. C. Smith, "A Data Base Processor", Technical Report CSRG-43, Computer Systems Research Group, University of Toronto, September, 1974

Potter, J L, "Image Processing on the Massively Parallel Processor", IEEE Computer, vol 16, no 1, January, 1983

Shaw, D E, *Knowledge-Based Retrieval on a Relational* Database *Machine,* PhD Thesis, Department of Computer Science, Stanford University, 1980

Shaw, D E, "The NON-VON Supercomputer", Technical Report, Department of Computer Science, Columbia University, August, 1982

Shaw, D E, "SIMD and MSIMD Variants of the NON-VON Supercomputer", *Proceedings of COMPCON Spring '84,* San Francisco, February, 1984

Shaw, D E, "Applications of VLSI Technology in a Massively Parallel Machine", *Circuits, Systems and Signal Processing,* 1985 (to appear)

Shaw, D E , "The Rootpoint Interconnection Network", Technical Report, Computer Science Department, Columbia University, 1985a (in preparation)

Shaw, D E and B K Hillyer, "Allocation and Manipulation of Records in the NON-VON Supercomputer", Technical Report, Computer Science Department, Columbia University, 1982

Shaw, D E and T M Sabety, "An Eight-Processor Chip for a Massively Parallel Machine", Technical Report, Computer Science Department, Columbia University, July, 1984

Slotnik, D L, "Logic Per Track Devices", in F Alt (ed), Advances *in Computers,* vol 10, Academic Press, New York, 1970

Sternberg, S R, "Biomedical Image Processing", *IEEE Computer,* vol 16, no 1, January, 1983

Stolfo, S J , "Five Algorithms for PS Execution on the DADO Machine", Technical Report, Computer Science Department, Columbia University, [AAAI-84, Proceedings of the National Conference on Artificial Intelligence], August, 1984

Stolfo, S, J. and D. E Shaw, "DADO, a Tree-Structured Machine Architecture for Production Systems", *National Conference on Artificial Intelligence,* 1982

Siegel, H J, L J. Siegel, F C Kemmerer, P T. Mueller, H. E Smalley, and S. D Smith , "PASM A Partionable SIMD/MIMD System for Image Processing and Pattern Recognition", *IEEE Transactions on Computers,* vol. 30, no 12, December, 1981

S Tanimoto, "A Pyramidal Approach to Parallel Processing", Technical Report, Computer Science

S Tanimoto and A. Klinger, *Structured Computer Vision,* Academic Press, 1980

Uhr, L, "Recognition Cones, and Some Test Results", in A R Hanson and E M Riseman (eds), *Computer Vision Systems,* Academic Press, 1980

Unger, S H , "A Computer Oriented Towards Spatial Problems", *Proceedings of the IRE,* p 1744, 1958

Wiederhold, G , "Knowledge and Database Management", *IEEE Software Magazine,* vol 1 no 1, January, 1984, pp 63-73