

Workshop on Research Directions in Architectures and Systems for Cognitive Processing

July 14 (full day) – 15 (morning), 2005
Ithaca, NY

RESEARCH GAP ANALYSIS

Computer Systems Laboratory (CSL)
Intelligent Information Systems Institute (IISI)
Cornell University

Workshop Organizers

José F. Martínez
Cornell U. (CSL & IISI)
Editor

Carla P. Gomes
Cornell U. (IISI)

Richard W. Linderman
AFRL

Attendees

Nael Abu-Ghazaleh
Binghamton U.

Ed Addison
Lexxle

Dave Albonesi
Cornell U. (CSL)

Paul Allopenna
Aptima

Jim Anderson
Brown U.

Manny Aparicio
Saffron Tech.

Julius Bogdanowicz
Raytheon

Bob Bond
MIT-LL

Rich Caruana
Cornell U. (IISI)

Dan Corkill
UMass

Steve Crago
USC-ISI

Julius Ettl
Lockheed Martin

Dennis Fitzgerald
ITT/AES

Chris Flynn
AFRL

Nort Fowler
AFRL

Kanad Ghose
Binghamton U.

Mary Hall
USC-ISI

Jon Hiller
ST Associates

Doug Holzhauer
AFRL

John Laird
U. Michigan

Amy Magnus
AFOSR

Rajit Manohar
Cornell U. (CSL)

Mike Moore
ITT/AES

Keshav Pingali
Cornell U. (CSL & IISI)

Mark Richards
Gatech

Jon Russo
Lockheed Martin

Bart Selman
Cornell U. (IISI)

Wilmar Sifre
AFRL

Carl Stern
Mgt. Sciences

Paul Yaworsky
AFRL

Executive Summary

The *Workshop on Research Directions in Architectures and Systems for Cognitive Processing* (July 14 and morning of July 15, 2003) was jointly organized by Cornell University's Computer Systems Laboratory (CSL) and Intelligent Information Systems Institute (IISI) on behalf of the Air Force Research Laboratory (AFRL). Attendees were asked to assess the current state of the art and identify research gaps on the subject of architectural and system support to cognitive computing.

The main findings and recommendations for architectures and systems are the following.

- * Conventional von Neumann micro- or multi-processor architectures of coarse-grain parallelism are power- and performance-inefficient for cognitive computing. There is opportunity in new approaches such as (1) very fine-grain, low-power parallel architectures; (2) memory-centric approaches; (3) general-purpose + cognitive co-processing architectures; and (4) unconventional circuit design that is (i) asynchronous, (ii) analog, (iii) approximate.
- * Cognitive processing algorithms and data structures are generally not locality- or context-aware, which limits performance. We advocate development of context/locality-aware algorithms and data structures for cognitive processing.
- * Cultural gap between architecture and cognitive communities hinders blue-sky thinking and slows down innovation. Significant leaps in cognitive systems are more likely if more fundamental, sustained interaction between these two communities is encouraged.
- * Traditional separation of symbolic and connectionist approaches may represent an important missed opportunity. We believe a hierarchical/layered integration of both should be explored, with the hardware possibly serving as facilitator by, for example, narrowing the speed gap between them.

The main findings and recommendations on benchmarks, metrics, tools, and infrastructure are the following.

- * The diversity and little interaction between cognitive areas makes it difficult to develop full-system benchmarks. We propose a two-phase approach: (1) In the short term, identify benchmarks separately for each application area. (2) In the long term, as cross-disciplinary efforts bear fruit, develop more wholesome benchmark suites that accommodate the emerging needs.
- * We see the need to develop principles and tools for observation, testing, and verification of cognitive systems, particularly in the presence of (1) inexact, non-repeatable computation; (2) complexity; and (3) time-dependent/emergent behavior.
- * Traditional metrics seem generally adequate for a fair range of quantitative analysis of cognitive processing: execution time, power consumption, mean time between failures or MTBF, etc. However, understanding higher-level behavior, which is expected of cognitive systems, requires seeking other forms of measurement.

Architectures and Systems for Cognitive Processing

In this section we describe a number of relevant problems identified by the workshop attendees, the limitations of current approaches, and the potential research opportunities. We classify research opportunities into short-, mid-, and long-term, depending on the expected lapse until first tangible results. This is not necessarily indicative that funding should be short-, mid-, or long-term, respectively, but rather than funding for these initiatives may be distributed in time and prioritized according to such predictions.

Problem: Performance Limitations

A quick look at traditional cognitive processing algorithms and data structures reveals seemingly high memory, communication, and input/output requirements. Many existing algorithms are not aware of the spatial/temporal locality properties of the underlying architecture. Also, while the notion of context is important in cognitive processing, contextual representation does not exploit such architectural locality. This absence of locality/context awareness is further magnified when algorithms are parallelized.

Examples of current limitations in this regard include large objects (e.g., sparse graphs) that exceed cache capacity; algorithms that do not organize data accesses in a cache-aware fashion; objects that are spread across large, even geographically distributed locations; not well-represented context.

Workshop attendees see a research opportunity in developing context/locality-aware algorithms and data structures for cognitive processing, where data accesses are organized in accordance to the underlying architecture's spatial/temporal locality properties. Given the ever-increasing speed gap between processor and memory in current architectures (two orders of magnitude or more), the potential for performance gains is large. Moreover, this effort would facilitate development of context/locality-aware parallel codes, which would further boost performance.

This research opportunity should be able to produce initial tangible results short-term.

Problem: Programming Model and Environment

Cognitive computing has traditionally followed one of two programming models: One that uses general-purpose programming environments (e.g., C), and another one that favors programming environments more specifically tailored to cognitive computing (e.g., Prolog). Prototyping languages, such as Matlab, are found somewhere in the middle. Currently, many of AI's key problems require speed, and thus the tendency is to work with general-purpose languages.

While general-purpose environments typically benefit from extensive investments in research and development, and are able to deliver higher performance than higher-level environments, such as prototyping languages or cognitive environments, they lack

awareness of the specifics of the problem, particularly in the case of cognitive processing: convergence flexibility, emergent behavior, etc.

Workshop attendees believe there may be a research opportunity in exploring “cognitive techniques” for programming environments, compiler, and run-time support, (e.g., code optimization that is tailored at cognitive programs). Not only does this possess the potential to improve cognitive computing performance, it could also generate ideas that could be applicable to general-purpose computing.

This research initiative should yield initial tangible results short- to medium-term.

Problem: Inefficiency of Current Architecture Approaches

Most architectures devoted to cognitive processing today are based on an array of conventional processors. This is inefficient for a variety of reasons: (1) Coarse-grain parallelism, as provided by conventional multiprocessors or processor clusters, is not a good mapping to the more fine-grained parallelism present in some cognitive processing tasks. (2) Current memory hierarchies, which revolve around the von Neumann concept, limit performance. (3) The power efficiency and scalability of these architectures exceeds, in general, the performance delivered.

Workshop attendees perceive a research opportunity in exploring two architectural concepts that would depart from the current approaches: (1) very fine-grain, low-power parallel architectures with flexible, efficient mechanisms for concurrency and co-ordination, and (2) memory-centric approaches. The success of some of these architectures would be tied to the advances in the programming model and environment problem described earlier. Some specific architectures to explore include processing in memory, associative memory organizations, etc. A fundamental part of this research should include exploration of the right primitives, structure, and functionality that the hardware should provide.

This research initiative should yield initial tangible results short- to medium-term.

Problem: Cognitive/Architecture Cultural Gap

One recurring concern in the investigation of architectures and systems for cognitive processing is the fact that each community is generally unaware of the challenges and opportunities present in the other one. Architects often do not know what the cognitive processing community *wants* (not to confuse with *has*), and the cognitive processing community is not well aware of the (missed) opportunities in current and upcoming architectures.

DARPA’s ACIP program—possibly the best funded and most well-regarded program in computer architectures for cognitive processing—includes a two-year “blue-sky” Phase I. Workshop attendees consider this time span too short and too unique to

generate radically new ideas and spur sustained collaboration between the two communities. In fact, the focus of Phase I toward “making it” into Phase II possibly limits creativity and innovation further.

Workshop attendees believe there is an opportunity in funding more long-term, collaborative research in more fundamental problems related to the connection between computer architectures and cognitive processing. The outcome of this long-term research is likely to be more innovative and forward-looking, it may spur other more targeted programs, and it may cement collaborative ventures between the two communities. Furthermore, to reinforce sustained collaboration, a research forum where to exchange ideas and assess progress should be set up.

This research initiative should yield initial tangible results short- to mid-term.

Problem: Symbolic vs. Connectionist Approach

Symbolic and connectionist approaches have largely diverged over the last years in terms of domain of application. Today, most of the research in higher-level cognitive computing per se is done taking a symbolic approach, and some regard connectionist approaches “slow” or “inefficient” at that level. On the other hand, connectionist approaches typically offer a much better match to hardware by their own nature.

Workshop attendees perceive a missed opportunity in the separation of symbolic and connectionist approaches, and suggest that a hierarchical/layered integration of both be explored, where the hardware could serve as facilitator by, for example, accelerating connectionist structures and algorithms to the point that they could be useful as part of a hybrid cognitive architecture.

This research initiative should produce initial tangible results mid-term.

Problem: Computational Model Mismatch

There is good reason to believe that general-purpose and cognitive computational models are fundamentally different. Computers today excel at solving amazingly quickly problems that humans are terrible at in comparison. On the other hand, general-purpose computing—and, by extension, general-purpose architectures—probably make a poor match to cognitive computing. Yet to date most efforts in computer architectures for cognitive processing focus on mapping cognitive algorithms and data structures to general-purpose computing/architectures.

Workshop attendees agree that, while it seems reasonable to conduct exploration of general-purpose architectures to run cognitive computations, particularly before long-term research can bear fruit, this combination is inefficient by nature. In particular, the mapping of cognitive tasks to general-purpose hardware seems unlikely to yield

unconventional speedups. At the same time, general-purpose architectures are excellent at what they do, and thus they are likely to be part of any future cognitive system.

Attendees see a research opportunity in exploring a co-processing approach: a general-purpose CPU side-by-side with a “cognitive CPU.” The challenges in this approach are numerous, from the description of the cognitive computational model, to realization of the cognitive CPU itself, to the exploration of task division and efficient, meaningful information exchange between the two CPU types.

This research initiative constitutes a long-term effort before tangible results can be obtained.

Problem: Digital Computational Substrate

Digital, synchronous logic is inherently precise: results are computed using exact arithmetic, and produced at regular clock cycles. This precision-oriented approach seems to be a clear mismatch to cognitive computing. Even if neuronal activity can be understood as fundamentally digital (transmission/inhibition), cognitive outcomes have little to do with exact arithmetic or logic.

Workshop attendees believe that constraining research to architectures that use a traditional digital substrate may fundamentally limit its impact and scope. They observe an opportunity in pursuing unconventional circuit design with one or more of the following properties: (1) asynchronous, (2) analog, (3) approximate. Specifically, workshop attendees encourage the exploration of “neuromorphic” circuit design. While the idea itself is not new, current technology advances and the possibility of a “multi-prong” approach makes it appealing.

This research initiative represents a long-term effort before tangible results can be obtained.

Problem: Non-integrative Research

The traditional partitioning of the different cognitive disciplines (e.g., knowledge representation, machine learning, language processing, reasoning, etc.) makes it hard to come up with architectural solutions that can help cognitive computing in a broad sense. While it is possible to develop different hardware solutions for each area, the ultimate impact is greatly diminished by this separation. Furthermore, later attempts at integration may find interfacing and compatibility issues across the different hardware components.

At the same time, current hardware efforts do not generally provide a mechanism for quick prototyping of hardware ideas, so that validation or benchmarking can be explored in a timely fashion.

Workshop attendees encourage support for hardware studies targeted at cross-disciplinary efforts in cognitive computing (such as the CALO project), where a holistic approach is more viable, and the hardware study itself can foster communication across cognitive computing discipline, for example via hardware prototyping.

The time to tangible results depends largely on the ambition of the underlying cognitive computing project. In the next section we revisit this problem and provide a timeframe.

Benchmarks, Metrics, Tools, and Infrastructure

In this section we describe a number of relevant problems in the areas of benchmarks, metrics, tools, and infrastructure, as identified by the workshop attendees, as well as the limitations of current approaches and the potential research opportunities.

Benchmarks

Benchmark creation/selection for the purposes of evaluating computer architectures and systems for cognitive processing is critical to promote faster, more robust development, prototyping, and implementation cycles. We must balance factors such as the number of benchmarks, their complexity, and coverage of important phenomena.

Currently, the diversity and relatively little interaction between the different cognitive processing communities (a concern highlighted in the earlier section on architectures and systems) makes it more difficult to come up with a common set of benchmarks that can provide a “complete” picture. Instead, in the short-term, carefully identifying benchmarks separately for each application area seems to make sense

As efforts are devoted to increasing interaction between the different communities, better benchmarks will be possible. In terms of complexity, initial benchmarks may be small kernels, although the final goal ought to be the use of full applications—as cognitive systems mature.

The stratified nature of cognitive systems may require multiple-level benchmarks. For example, some benchmarks may be developed to evaluate the computer architecture for cognitive processing, while a different set of benchmarks may be used to assess the cognitive methods themselves. The identification of key layers for benchmarking, as well as the intersection between these layers, is currently not well understood.

Because environment conditions, input data, or metrics for cognitive processing may not be as simple as benchmarks for general-purpose computing, *self-contained* benchmarks may be developed to contain meta-processing code that can select, build, and manage the appropriate benchmarking environment, and collect and process results at the end of the test.

Metrics

Traditional metrics used in general-purpose computing, such as those related to performance and power, seem generally adequate for a fair range of quantitative analysis of cognitive processing systems. Specifically, in regards to the impact of computer architectures and systems, the quantitative metrics used to evaluate software response to hardware mechanisms (e.g., execution time, power consumption, mean time between failures or MTBF, etc.) are likely to provide useful information at that level.

A cognitive system, however, is able to process information at a significantly higher level. Moreover, cognitive systems are often not “Boolean” or otherwise exact. One desirable feature of cognitive systems in certain domains, for example, is robustness: The system offers certain guarantee of providing an approximate answer in any situation. Quantification of characteristics such as robustness is harder—is it more robust, for example, if it provides a response faster? What is the quality of that faster response?

To identify the right metrics, workshop attendees see a need to explore answers to questions such as: What is the desirable behavior of a cognitive system? Which attributes can be quantified and used as a metric? Is it possible to quantify behavior via indirect, observable metrics? In particular, how do these attributes connect with traditional metrics for performance, real-time constraints, power consumption, etc.?

As in the case of benchmarks, the answer to these questions may come more easily through specialization, by looking for metrics for each cognitive domain, and thus integration remains a problem for metrics as well.

Tools and Infrastructure

As indicated in the architecture and systems section, workshop attendees recognize the need to develop tools and infrastructure for relatively rapid prototyping of architectural ideas for cognitive processing. This is a relatively short-term goal that would accelerate innovation and integration of architectures and cognitive systems.

The development of such tools and infrastructure are likely to impact longer-term research as well. Workshop attendees believe that this effort should include a definite focus toward the establishment of meaningful middleware, libraries, etc.

In particular, the use of a well-defined middleware or virtualization layer may allow some degree of independence in the design details for architectures and cognitive software, while still abiding by a predetermined interface.

Another important problem is that of providing tools for observation, testing, and verification. Cognitive systems are complex in nature, do not follow an exact computation, or provide fully predictable/repeatable answers. This problem becomes more complex as the system evolves over time, and/or if emerging behavior is present.

How can we guarantee that a cognitive system's operation, or the system itself, is safe? Trustworthy? Correct?

Workshop attendees see the need to explore this dimension—determine what kind of tools should be developed to accomplish testing/verification. Approaches to this problem may include possibilities such as: Should the system be able to explain its own reasoning (for auditing purposes, for example)? If so, how? Should the system contain axiomatic safeguards (perhaps reminiscent of “I, Robot”)? What sort of guarantees, if any, should the hardware provide?

More fundamental questions on algorithmic bounds, such as providing a guarantee of convergence, may require development of fundamental models (e.g., mathematical framework).

The issue of emergent behavior also raises questions about sustainability of performance. Tools and infrastructure should be able to accommodate such changes, and provide the system with the ability to self-optimize in order to sustain a certain level of performance, particularly in the case of real-time cognitive systems.

* * *