COGnitive ENGine Technologies (COGENT) – An Innovative Architecture for Cognitive Processing

Julius Bogdanowicz  Raytheon - Principal Investigator
John Granacki  USC-ISI - Co-Principal Investigator

September 20, 2006

This work was sponsored by DARPA IPTO in the Architectures for Cognitive Information Processing program with Dr. William Harrod as Program Manager; contract number FA8650-04-C-7126

Distribution Statement “A” (Approved for Public Release, Distribution Unlimited).
Agenda

- COGENT Team
- Goals
- Study Approach
- Requirement Drivers
- HW Philosophy
- Architecture Levels
- Agent Based Cognitive System Model
- COGENT Hardware & Software
- Performance
- Differences with Conventional Architectures
- Summary
COGENT Team Members

- Raytheon: Julius Bogdanowicz, Michael Vahey, Brad Miller, Bradley Norman, Matt Benjamin, Mark Redekopp, Doug Brink
- USC-ISI: John Granacki, Jeff LaCoss, Wei-Min Shen, Andrew Gordon, Jerry Hobbs, Mark Moll, Behnam Salemi
- Exogi: Craig Steele
- Mercury Computer Systems: Jim Kulp
- University of Pittsburgh: Daniel Mosse, Bruce Childers, Jonathan Misurda
- HRL: Howard Neely, Michael Daily
- Georgia Tech: Sudhakar Yalamanchili, Krishna Palem, Vincent Mooney, Santosh Pande
COGENT Goals/Objectives

- Broad Cognitive System Model based on spanning set of cognitive components that will efficiently implement current functions and enable new classes of cognitive algorithms
- Scalable computational fabric with morphable, heterogeneous hardware engines supporting multiple cognitive functions
- Extensible, open architecture allows general and special purpose accelerators for signal, data, and cognitive processing
- Communications network enables tight coupling of cognitive processing with classical signal, image, and data processing
- Instrumented hardware architecture for reacting to external environment and dynamic resource demands
- Self awareness, reacts to measured processor & memory activity patterns and the external environment to evaluate progress towards goals and achieves best results within time constraints
Flow from Challenge Problems to Architecture

- Challenge Problems
- Elementary Processes
- Elementary Transformations
- Reasoning and Learning Methods
- Mathematical, Logical, and Cognitive Classes of Methods

Cognitive System Model

- Abstraction of Elementary Process Form: Graph Representations
- Memory and Representation Architecture

- Hardware Technologies and Trades
- Derivations of New OPCODES

COGENT Architecture
Cognitive Problems Identify Computing Needs

- Process cognitive applications for military missions
  - **Recognition** of warfighter intent
    - Understanding of warfighter desires in context
    - Interaction driven by cognitive agents intentions
    - **Human problem understanding & decision making markedly improved**
  - **Analysis** of intelligence data
    - Detection of hidden relationships in very large knowledge bases
    - Slowly changing knowledge base
    - **Process very large problems**
  - **Planning** for wide range of missions
    - Single autonomous vehicles → battlefield
    - Rapidly changing working data
    - **Deliver real-time response in highly dynamic environments**

- Lessons learned
  - Applications need a robust Cognitive System Model
    - Adopted an Observe - Orient - Decide – Act + Learn (OODA+L) model based on a combination of research cognitive models
    - Need latency tolerant processing techniques with large memories
    - Need sophisticated memory management techniques for episodic and long term memory
  - Confirmed hardware support needed for agents, graphs, Bayesian networks & a wide range of computational kernels
  - To simplify application development we decouple the computational view from developers view
  - **New classes of algorithms are required to exploit the new computational fabric; must re-think the underlying computational model**
Cognitive Motivation

- **Cognitive applications are characterized by:**
  - Graph based operations and data structures
  - Sparse knowledge representation
  - “Inexact” Information
  - Very large amounts of parallelism at multiple levels
    - **Observe:** Input symbols distributed to multiple agents (sub/pub)
    - **Orient/Decide:** Competing Possible Worlds (OR-parallelism)
    - **Orient/Decide:** Searching and matching (Graph parallelism)
  - Potential for speculative processing – multiple predictive processes
  - Approximate solutions provided by “anytime” and best-available calculations
    - **Prioritize** promising processing contexts
    - **Filter/Prune** stale (too late) and ineffective (poor solution) processing
  - Learning - dynamic additions to knowledge base

- **Cognition is poor match to conventional systems**
  - Limited parallelism with user specified management
  - Memory-intensive
    - Extensive pointer-chasing through graphs
    - Memory access is data dependent, limiting effective use of data caches
      - Profiling experiment: observed 1 IPC on 4-issue SGI system (80% data cache miss)
  - Processors optimized for numeric, not symbolic processing
Enable & exploit parallelism at all levels
- Multiprocessor system with very large distributed memory
- HW generates and manages parallelism
  - Independent agents and/or Possible Worlds running in parallel
  - HW-managed multicasting of inputs to agents via sub/pub mechanism
  - Graph operations spawn parallel search and match operations
- Mitigate cost of speculative computations
- Minimize user awareness of parallelism

Manage parallel processing in HW where possible
- Prioritizing – “promote” promising threads
- Filtering – quickly prune ineffective and “too late” threads
- Synchronizing – enforce “check-in”
- Enable anytime or earliest/best processing

Provide HW support for cognitive middleware
- Graph/Bayesian/HMM data structures
- Fast access of distributed objects, network routing, etc.
COGENT Architecture Levels

- Specify Mission goals & requirements
  - Performs goal selection, QoS
- HL OODA+L Cognitive Architecture with problem-solving rules specific to app
- Cognitive Algorithms & Agent Architecture
  - Newell 7+2
- Common Cognitive Services
  - Cognitive Agent Oriented Programming
  - Kernels (e.g., Probabilistic Reasoning)
- Run-time High Level Compiler
- Abstract Machine Models
- Run-time Low Level Compiler & Resource Mgmt
- Machine organization
  - Single Name Space
- Hardware micro-architectures
  - Graphs
  - Special-Purpose Accelerators (e.g. Bayes Nets)
- Optimized cognitive systems
  - Servers
  - Embedded/real-time
COGENT Hardware and Functional Organization for Embedded Applications

Sensors & Interfaces

Filters and Data Reduction

Object Recognition & Feature Classification

Coefficients, Algorithms, Etc (Predictions of what to look for)

Knowledge Based Processing & Learning

Output Processing & Interfaces

COGENT Cognitive Processor

"Signal Processing"

"Cognitive Processing"

Extremely High Data Rates

Moderate Data Low Information Rates

Low Symbolic Information Rates

Low – Moderate Status & Control Rates

Low – Very High Data & Control Rates

SAR

IR

Comm

Comm

Actuators

User Console

Coefficients, Algorithms, Etc (Predictions of what to look for)
We Implement the Cognitive Architecture Using Intelligent Agents

**Orientation Frame 1**

**Orientation Frame 2**

**Orientation Frame n**

**Actuator Agent**

**Action Manager Agent**

**Observation Agent**

**State Information**

Agent

Agent

Agent

**Orientation-Dependent Knowledge Base**

*Graphs, rules, etc. relevant to the orientation situation.*

**Subscription request**

**Symbolic & Numeric data**

**Actions**

**Actuators, Environment, etc.**

**Query interface**

**Prioritize, Fuse, Act**
We Implement the Cognitive Architecture Using Intelligent Agents

Perceptions are formed based on “what we are looking for” – i.e., they are driven by preconception of what we expect in the world, our current needs, etc.

1: Observe

Observation Agent

Symbolic & Numeric data

Subscription request

Orientation Frame 1

Orientation Frame 2

Orientation Frame n

Actions

Actuator agent

Actuators, Environment, etc.

Prioritize, Fuse, Act

Orientation Frame

State Information

Agent

Agent

Agent

API

API

API

Orientation-Dependent Knowledge Base

Graphs, rules, etc. relevant to the orientation situation.

Beliefs, hypothesis, goals, active/inactive, etc.

Actions

Manager Agent

Actions

Manager Agent

Actions

Manager Agent

Actions
We Implement the Cognitive Architecture Using Intelligent Agents

Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

Perceptions are formed based on “what we are looking for” – i.e., they are driven by preconception of what we expect in the world, our current needs, etc.

Individual Agents may be Reactive (propose an action in response to some observed situation) or part of a Deliberative process (perform some kind of formal or informal reasoning, including matching Episodic Memory, i.e., remembering when such a situation arose before, and what we did then, then using a process of Analogical Reasoning to apply it to the current circumstances.

Inexact Pattern-matching rules are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive “ways to think about a situation”, which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).
We Implement the Cognitive Architecture Using Intelligent Agents

Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

Each Frame proposes actions (which may be part of more elaborate plans) which are compared and correlated here. One action might benefit multiple orientations and goals; an agent may be able to perform more than one action at once, etc. Here, we pick a rational set of things to do now.

Inexact Pattern-matching rules are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive “ways to think about a situation”, which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).

Perceptions are formed based on “what we are looking for” – i.e., they are driven by preconception of what we expect in the world, our current needs, etc.

Beliefs, hypothesis, goals, active/inactive, etc.

Individual Agents may be Reactive (propose an action in response to some observed situation) or part of a Deliberative process (perform some kind of formal or informal reasoning, including matching Episodic Memory, i.e., remembering when such a situation arose before, and what we did then, then using a process of Analogical Reasoning to apply it to the current circumstances.)
We Implement the Cognitive Architecture Using Intelligent Agents

Each frame represents a different point of view – different assumptions about how the world really is, and therefore what is most important.

Each Frame proposes actions (which may be part of more elaborate plans) which are compared and correlated here. One action might benefit multiple orientations and goals; an agent may be able to perform more than one action at once, etc. Here, we pick a rational set of things to do now.

Individual Agents may be Reactive (propose an action in response to some observed situation) or part of a Deliberative process (perform some kind of formal or informal reasoning, including matching Episodic Memory, i.e., remembering when such a situation arose before, and what we did then, then using a process of Analogical Reasoning to apply it to the current circumstances).

Inexact Pattern-matching rules are used to mediate this process, similar to ACT-R. As we reuse analogies, we may induce new pattern rules to help us get to the more productive “ways to think about a situation”, which may ultimately lead to new orientation frames as well (if we learn to differentiate between situations).

Perceptions are formed based on “what we are looking for” – i.e., they are driven by preconception of what we expect in the world, our current needs, etc.

Orientation-Dependent Knowledge Base

Graphs, rules, etc. relevant to the orientation situation.

Beliefs, hypothesis, goals, active/inactive, etc.

Observation Agent

Query interface

Agent

API

API

Orientation Frame 1

Orientation Frame 2

...
COGENT: A Highly Parallel Cognitive Processor

- Concurrent agents drive multiple predictive “possible worlds” storage and relationships
- Universal name space enforced across computation fabric storage accelerates agents and prunes unnecessary work

**Agent-Based Programming Model**

**Agent Control Environment**

- **DDC** = Data Distribution Center - hardware for automatic “scatter” (distribution of computation to the relevant, distributed data)
- **CAGE** = Cognitive Agent & Graph Engine – form parallel computing fabric, accelerates primitive operations on graphs, supports probabilistic reasoning; uses distributed, scalable data storage
- **PFF** = Prioritize, Filter and Fuse – automatic gather, coalescing results, pruning of stale and unproductive computation, time based to assure timely results

**Cognitive API: Operators plus Utilities**

**Recirculating Computational Flow**

**Control**

**Answers**

**Observation**: External World (Environment)

**Orient**: Application-Independent Cognitive Architecture

**Act**: Programmer View

**HW-managed Parallelism & Namespace**

**Decide**: Computational View

**High Speed Input**

**Outputs**

**DDC** Single locus of global name translation

**CAGE** = Cognitive Agent & Graph Engine – form parallel computing fabric, accelerates primitive operations on graphs, supports probabilistic reasoning; uses distributed, scalable data storage

**PFF** = Prioritize, Filter and Fuse – automatic gather, coalescing results, pruning of stale and unproductive computation, time based to assure timely results
Graphs and Efficiency

- Highly parallel computation
  - Relational structure of graphs makes inherent parallelism apparent
- COGENT accelerates computation on large graphs:
  - Program instructions sent to location of data to minimize data movement optimizes bandwidth usage.
  - HW managed label-based routing of graphs and vertices (DDC)
  - Fast lookup of local graph vertices on a CAGE node
  - Wide-word memory access & processing of graph components on a CAGE node
- CAGE HW supports memory re-organization for efficient access to sparse to dense graph structures
  - Garbage collection performed during process
**COGENT HW Architecture**

**Recirculating Computational Flow**

- **High Speed Input**
- **DDC**
- **CAGE**
- **PFF**

**DDC** = Data Distribution Center - hardware for routing messages (data or computation tasks) to CAGE “worker” processors. Routing is equivalent to automatic “scatter” (distribution) of computation to the relevant, distributed data located in 1 or more CAGE nodes.
CAGE = Cognitive Agent & Graph Engine(s) – form parallel computing fabric, hosts agents, augmented ISA accelerates primitive operations on graphs, supports probabilistic reasoning, uses distributed, scalable data storage; Agents distributed to 1 or more CAGE nodes, Agents spawn cognitive functions located on 1 or more CAGE nodes, Computations requests on non-local data are automatically sent to proper CAGE node via PFF & DDC.
COGENT HW Architecture

Recirculating Computational Flow

High Speed Input

Outputs

DDC

CAGE

CAGE

CAGE

PFF

PPF = Prioritize, Filter and Fuse – automatic “gather”/coalescing results from CAGE(s), pruning of stale and unproductive computation, re-circulates uncompleted computations through DDC, time based to assure timely results
Agents/Components

Agent-based Cognitive Architecture: OODA, Orientation frames etc.

AOP/CAOP

Cognitive Agent Abstraction   Library Agents

Agent Abstraction

Agent abstraction implemented as assemblies of components

Distributed/Parallel

Component-based Kernels

Library Components/Assemblies

Components Abstraction: active/autonomous/connected

Component Infrastructure   Cogent 50+ Fundamental Operations

Cogent Node

Local RTOS Services   Local Comm Services   Local Cog50 Elementary Building Blocks   Local Libraries (e.g. math)

Node Comm Primitives   Node Processing ISA
Example of Simulated Performance

- Surrogate for UAV ISR planner using analogical reasoning
- Problem size: 6600 propositions represented as a graph structure with 40K vertices & 80K edges
- Extrapolated performance on Intel Pentium 4 was 40 hours
- Goal: 800X speedup
- Minimum of 32 CAGE processors needed to exceed goal

![Graph showing speed up over Intel Pentium 4 Baseline]

**Speed up Over Intel Pentium 4 Baseline**

- **GOAL:**
- Number of CAGE nodes:
  - 16
  - 32
  - 64
  - 128

<table>
<thead>
<tr>
<th>Number of CAGE nodes</th>
<th>Speed up</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Conventional Processor</td>
<td>COGENT Processor</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Relies heavily on cache: dynamic access patterns make cache ineffective</td>
<td>No cache – large knowledge bases won’t fit Large on chip memories with good BW – per processor memory BW &gt; 100 GB/S</td>
</tr>
<tr>
<td>Load/store access to memory</td>
<td>HW manipulation of memory access, global ID vs. address, publish and subscribe to information sources</td>
</tr>
<tr>
<td>Word focused – no semantics</td>
<td>Semantically accessed memory: Graph based representation of knowledge – HW optimized access and manipulation mechanisms</td>
</tr>
<tr>
<td>General computing – RISC instruction set – register file focus – compiler driven program</td>
<td>Optimizations for the cognitive operations – intensive memory focus – staging and location of data HW optimized</td>
</tr>
<tr>
<td>Exact, repeatable deterministic functions – low level semaphores</td>
<td>Probabilistic representations, reconstructive memory – runtime synthesized data representations</td>
</tr>
<tr>
<td>Ops concept is driven from program counter, interrupts, etc.</td>
<td>Ops concept is driven from data flow and probabilistic data relationships – dynamically adjusted based on experience</td>
</tr>
</tbody>
</table>
Summary

- COGENT is an innovative recirculating computational architecture for cognitive processing
- COGENT architecture is being driven by application & cognitive system model requirements
- A single unified hardware & software structure has been defined
  - Hardware directly supports the management of parallelism
  - Agent based software structure provides foundation for OODA based cognitive system model to implement the applications
- Simulations are being used to refine the architectural details