Biologically Inspired Computing:
- The DARPA SyNAPSE Program &
- The Hierarchical Temporal Memory

Dan Hammerstrom
Electrical And Computer Engineering
Portland State University

Intelligent Computing

- There is one class of problems that we still do not solve well
- These problems involve the interaction of a computing system with the real world
- Which, in part, involves a transformation and understanding of data at the boundary between the real world and the digital world
- These problems occur wherever a computer is interacting with the real world – which includes almost every embedded application
- An interesting opportunity for specialized hardware
Our Focus: Intelligent Signal Processing (ISP)

- ISP augments and enhances traditional DSP (Digital Signal Processing) by incorporating contextual and higher level knowledge of the application domain into the data transformation process.

The “Front End” - DSP

1. Motor Control
2. Feature Extraction
3. Signal Processing
4. Classification

The “Back End” - ISP

1. Motor Control Subprogram
2. Motor Control Programs
3. Decision Making
4. Contextual Semantic Inference

The “Front End”

- Front end processing is well understood, it is the realm of traditional digital signal and image processing.

- Front end algorithms generally apply the same computation over large arrays of elements, they are data parallel, and communication tends to be local.
  - An excellent example of such an architecture is the CNN (Cellular Non-linear Network) developed by Chua, Roska et al.
  - Most “neuromorphic” VLSI operates at the front end.
But Then There’s The “Back-End” …

- In the early days of computing, “Artificial Intelligence” focused on the representation and use of contextual and semantic information
- Knowledge was generally represented by a set of rules
- However, these systems were “brittle,” exhibiting limited flexibility, generalization, and graceful degradation
- They did not scale
- And they were unable to adapt dynamically (i.e., learn) within the context of most real world applications

The ISP Toolbox – Still mostly empty after all these years …
Some Desirable ISP Characteristics

- Solving the problem
- Massively parallel and low precision
- Self-organizing – in fact, system design becomes more the provision of organizing principles (Prof. Christoph von der Malsburg), than the specification of all operational aspects of the models
  - [www.organic-computing.org](http://www.organic-computing.org)
- Generalization, and graceful degradation
- Low power - the processing power of the brain is roughly $10^{15}$ operations per second which it accomplishes at a power dissipation of about 25 watts
- Scales - The scaling limitations of both symbolic and traditional neural network approaches constitute one of their biggest shortcomings
- Adaptive - Consequently another important characteristic of real systems is incremental, integrative adaptation or learning during system operation

The Scope of Our Project

- Conceptually one can think of “computational intelligence” as a spectrum
- And though not universally accepted, it has been hypothesized that this spectrum is more or less continuous from one end to the other

---

Increasing Intelligence
There is increasing interest in using models from Computational Neuroscience, in particular cortical models as inspiration for new models of computation in general and intelligent computing in particular.

In Europe there is FACETS
- The goal of the FACETS (Fast Analog Computing with Emergent Transient States) project is to create a theoretical and experimental foundation for the realisation of novel computing paradigms which exploit the concepts experimentally observed in biological nervous systems.

And in the US there is the DARPA SyNAPSE Program

**Systems of Neuromorphic Adaptive Plastic Scalable Electronics**

Dr. Todd Hylton, Program Manager
DARPA DSO
The SyNAPSE program seeks to extend the development of modern electronics into a new revolutionary new era using a similar paradigm.
### Program Outline

<table>
<thead>
<tr>
<th>Phase 0</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
<th>Phase 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component synapse (and neuron) development</td>
<td>CMOS process and core circuit development</td>
<td>CMOS process integration</td>
<td>~10^6 neuron single chip implementation &quot;Mouse&quot; level</td>
<td>~10^8 neuron multi-chip robot at &quot;Cat&quot; level</td>
</tr>
<tr>
<td>Hardware</td>
<td>Architecture &amp; Tools</td>
<td>Emulation &amp; Simulation</td>
<td>Environment</td>
<td></td>
</tr>
<tr>
<td>Microcircuit architecture development</td>
<td>System level architecture development</td>
<td>&quot;Mouse&quot; level benchmark (~ 10^6 neuron)</td>
<td>Build Sensory, Planning and Navigation environments &quot;Small mammal&quot; complexity</td>
<td></td>
</tr>
<tr>
<td>Preparatory studies only</td>
<td>Simulate large neural subsystem dynamics</td>
<td>&quot;Cat&quot; level benchmark (~ 10^8 neuron)</td>
<td>Add Audition, Proprioception and Survival &quot;All mammal&quot; complexity</td>
<td></td>
</tr>
<tr>
<td>Preparatory studies only</td>
<td></td>
<td></td>
<td>Add Touch and Symbolic environments</td>
<td></td>
</tr>
<tr>
<td>Currently only Phases 0 and 1 have been funded</td>
<td>We are now in Phase 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Currently only Phases 0 and 1 have been funded
We are now in Phase 1

Approved for Public Release, Distribution Unlimited

---

- There are three contractors:
  - IBM - Dharmendra Modha
  - HP - Greg Snider
  - HRL (formerly the Hughes Research Lab) – Narayan Srinivasa

- The ultimate goal is to build a low-power, compact electronic chip combining a novel analog circuit design and a neuroscience-inspired architecture that can address a wide range of cognitive abilities—perception, planning, decision making and motor control

- "Our research progress in this area is unprecedented," says DARPA program manager Todd Hylton, Ph.D. "No suitable electronic synaptic device that can perform critical functions of a biological brain like spike-timing-dependent plasticity [an indicator of the capability to learn] has ever before been demonstrated or even articulated.”
HRL Team

- **Hardware:**
  - Analog circuits, HRL
  - Nano-devices, Wei Lu, Univ. of Michigan
  - Floating gate devices, Paul Hasler, Georgia Tech
  - Systems integration and global communication, Dan Hammerstrom, Portland State

- **Neuroscience**
  - Steve Grossberg, Boston University
  - Eugene Izhikevich, Jason Fleischer, et al., Neurosciences Institute
  - Jeff Krichmar, UC Irvine
  - Phil Goodman, University of Nevada, Reno
  - Giorgio Ascoli and Alexei Samsonovich, George Mason

- All three teams have completed phase 0 and are now in the middle of phase 1, which is scheduled to end in December 2010

---

**NEUROMORPHIC NETWORKS**

**Basic idea:**
CMOS "somas" + nanowire "axons" and "dendrites" + nanodevice "synapses"

Generic structure of a feedforward CrossNet

Likharev

S. Fölling, et al. (2001)
O. Turel et al. (2004)
Structural View of Mixed-Signal CMOL Design (Each CP) - Gao

CMOL Nanogrid

Nanowire-CMOS Pin

Driving current from CMOS to nanogrid

Injecting current from nanogrid to CMOS

Analog CMOS output neuron circuit e.g., I&F circuit

C. Gao

Work performed by HRL under DARPA contract HRL0011-09-C-001
However …

The “Gap”

- So do neural techniques lead to advanced ISP?

- Most Computational Neuroscience is weak in making the jump from spiking neurons with learning rules such as STDP to Cognition
  - The SyNAPSE program has this problem

- Even solutions to the more narrowly defined ISP back-end problem are not obvious
One way to possibly bridge the gap is to define a computational model that “spans” the gap.

A candidate has been proposed by Albus and many others:

The Cortical Computation Unit (CCU)

Desirable Characteristics

- Modular
- Distributed representation
- Hierarchical, bi-directional information flow
- Massively parallel
- Scales
- Learning / Self-organizing
- Does a kind of Bayesian inference

Solves the problem …
Albus: What is the path to success for reverse engineering the brain?

Pick the right level of resolution

- **overall system level** (central nervous system)
  - AI and Cognitive Neuroscience units (e.g., cortical regions)
  - macro-computational units (e.g., cortical hypercolumns & loops)
- **micro-computational units** (e.g., cortical microcolumns & loops)
- **neural clusters** (e.g., spinal and midbrain sensory-motor nuclei)
- **neurons** (elemental computational units) – input/output functions
- **synapses** (electronic gates, memory elements) – synaptic phenomena
- **Mainstream Neuroscience & Neural Nets** – molecular phenomena


---

Modular Hierarchies

- Computer engineers make extensive use of modular, hierarchical design, can we assume the same for these models?

- In neocortex the fundamental unit of computation appears to be the cortical 
  **minicolumn** (Mountcastle)
  - A minicolumn is a vertically organized group of about 80-100 neurons which traverses the thickness of the gray matter (~3 mm) and is about 50µ in diameter
  - Neurons in a column tend to communicate mostly vertically with other neurons in the different layers in the same column

- These are subsequently organized into larger columns variously called just 
  “columns”, “cortical columns”, “hypercolumns”, or sometimes “modules”
  - Note, columnar organization is not universally accepted in the neuroscience community
“Bayesian Memory” (BM) Building Block

- An approximation to a CCU
- A BM sees only a subset of its input BMs and each BM’s subset is slightly different
- Inference is performed over small sub-blocks
- The number of blocks increases linearly
- Relies heavily on sparse, distributed representations

From *Big Brain* by Gary Lynch and Rick Granger (Palgrave McMillan 2008):

- “Although the ‘front end’ circuits of the brain, with their point-to-point circuit designs, specialize in their own particular visual and auditory inputs, the rest of the brain converts these to random-access encodings in association areas throughout cortex. … these areas take initial sensory information and construct grammars

- “These are not grammars of linguistic elements, they are grammatical organizations (nested, hierarchical, sequences of categories) of percepts – visual, auditory, and other

- “Processing proceeds by incrementally assembling these constructs … these grammars generate successively larger ‘proto-grammatical fragments,’ eventually constituting full grammars"
“They thus are not built in the manner of most hand-made grammars; they are statistically assembled, to come to exhibit rule-like behavior, of the kind expected for linguistic grammars.

“Proto-grammatical fragments capture regularities that are empirically found to suffice both for recognizing and generating grammatical sequences.

“Auditory pathways in our brains grew and lengthened building voice-sounds into words, words into phrases, phrases into sentences.”

Information needs to flow both ways
Assume that conditional probabilities / priors - model the world
Bi-Directional Belief propagation – e.g., visual cortex model
Inference as the basic computation

Lee and Mumford Visual cortex model
A BM module then has two parts:

- **An input** that approximately learns the probability distribution of its inputs
  - A table of vectors, which is called a codebook and implements Vector functionality
  - Approximates the input probability distribution
  - Learns in an unsupervised manner by allocating new vectors and/or moving existing vectors

- A Vector Quantizer is an example of such a function – an “entropy” maximizing data reducer

- **An output** that creates a new representation of the codebook vectors to send up the hierarchy
  - A table of vectors, one for each codebook vector
  - The output vectors are sparse and are of a higher dimension than the space they span
  - In some implementations they are random, in Numenta's HTM they just pass up the index of the winning codebook vector
  - Ideally they would self-organize, as in a Self-Organizing Map (SOM) to capture the one or two dimensions of highest invariance of the input space
The winning codebook vector is the most likely given some input.

W is some vector weight.

In VQ terms it specifies the width of the region surrounding the codebook vector.

It can be thought of as the "prior" probabilities.

<table>
<thead>
<tr>
<th>Input Vector</th>
<th>W</th>
<th>i</th>
<th>Output Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000100111000</td>
<td>0.12</td>
<td>0</td>
<td>10010101000</td>
</tr>
<tr>
<td>0001101001010</td>
<td>0.05</td>
<td>1</td>
<td>01011100110</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111010101000</td>
<td>0.001</td>
<td>n</td>
<td>10001000100</td>
</tr>
</tbody>
</table>

Belief Propagation

The model looks good, but something missing.

It is generally assumed that biological systems perform a kind of inference over the knowledge they have learned.

If so, then perhaps Bayesian Inference.


Assume that our modular hierarchy is a directed Bayesian network.

- Vertices are objects which have local information and carry out local computations by updating of probability distribution via message passing.
Most Likely Computation – Influence From Above

- OK, I lied, the computation of the most-likely codebook vector is actually more complicated.

- The reason is that it is a result of the influence of “belief” propagated both from above and from below.

<table>
<thead>
<tr>
<th>Input Vector</th>
<th>W</th>
<th>i</th>
<th>Output Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000100111000</td>
<td>0.12</td>
<td>0</td>
<td>10010101000</td>
</tr>
<tr>
<td>00011010010100</td>
<td>0.05</td>
<td>1</td>
<td>01011001000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111010101000</td>
<td>0.001</td>
<td>n</td>
<td>10001000100</td>
</tr>
</tbody>
</table>

A Simple Bayesian Network

\[
P(d|b,c) \quad P(b|a) \quad P(c|a) \quad P(d|b,c)
\]

CPT for node D, there are similar tables for B and C

| P(d|b,c) | d₁ | d₂ |
|---------|----|----|
| b₁, c₁  | 0.5| 0.5|
| b₂, c₁  | 0.3| 0.7|
| b₁, c₂  | 0.9| 0.1|
| b₂, c₂  | 0.8| 0.2|
Pearl’s Belief Propagation

The Evidence

- Evidence – values of observed nodes
  - $V_3 = T$, $V_6 = 3$
- Our belief in what the value of $V_i$ ‘should’ be changes
- This belief is propagated
- As if the CPTs became

<table>
<thead>
<tr>
<th>$V_3$</th>
<th>$V_5$</th>
<th>$P$</th>
<th>$V_2$</th>
<th>$V_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>1.0</td>
<td>T</td>
<td>0.0</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>1.0</td>
<td>F</td>
<td>0.0</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.0</td>
<td>F</td>
<td>0.0</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.0</td>
<td>F</td>
<td>0.0</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>1.0</td>
<td>F</td>
<td>1.0</td>
</tr>
</tbody>
</table>
The π Messages

- What are the messages?
- For simplicity, let the nodes be binary

<table>
<thead>
<tr>
<th>V₁=T</th>
<th>V₁=F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>0.2</td>
</tr>
</tbody>
</table>

The message passes on information

What information? Observe:

\[ P(V₂|V₁) = P(V₂|V₁=T)P(V₁=T) + P(V₂|V₁=F)P(V₁=F) \]

The information needed is the CPT of \( V₁ = \pi(V₁) \)

π Messages capture information passed from parent to child

The λ Messages

- What about λ?

Assume \( E = \{V₂\} \) and compute by Bayes rule:

\[ P(V₁|V₂) = \frac{P(V₁)P(V₂|V₁)}{P(V₂)} = \alpha P(V₁)P(V₂|V₁) \]

The information not available at \( V₁ \) is the \( P(V₂|V₁) \). To be passed upwards by a λ-message. Again, this is not in general exactly the CPT, but the belief based on evidence down the tree.

- The messages are \( \pi(V) = P(V|E^+) \) and \( \lambda(V) = P(E|V) \)
Combination of evidence

\[ P(V \mid E) = P(V \mid E^+, E^-) = \alpha' P(E^+, E^- \mid V) P(V) = \]
\[ \alpha' P(E^- \mid V) P(E^+ \mid V) P(V) = \alpha P(E^- \mid V) P(V \mid E^+) = \alpha \lambda(V) \pi(V) = \text{BEL}(V) \]

- \( \alpha \) is the normalization constant
- normalization is not necessary (can do it at the end)
- but may prevent numerical underflow problems

Messages to pass

- We need to compute \( \pi_{XY}(x) \)

\[ \pi_{XY}(x) = \alpha \pi_x(x) \prod_{k \neq j} \lambda_{k,x}(x) \]

- Similarly, \( \lambda_{XY}(x) \), \( X \) is parent, \( Y \) child
- Symbolically, group other parents of \( Y \) into \( V = V_1, \ldots, V_q \)

\[ \lambda_{Y,X}(x) = \sum_{y_j} \lambda_{Y_j}(y_j) \sum_{v_1, \ldots, v_q} P(y \mid v_1, \ldots, v_q) \prod_{k=1}^q \pi_{V_k Y_j}(v_k) \]
The Pearl Belief Propagation Algorithm

- Iterate until no change occurs
  - (For each node X) if X has received all the \( \pi \) messages from its parents, calculate \( \pi(x) \)
  - (For each node X) if X has received all the \( \lambda \) messages from its children, calculate \( \lambda(x) \)
  - (For each node X) if \( \pi(x) \) has been calculated and X received all the \( \lambda \)-messages from all its children (except Y), calculate \( \pi_{xy}(x) \) and send it to Y.
  - (For each node X) if \( \lambda(x) \) has been calculated and X received all the \( \pi \)-messages from all parents (except U), calculate \( \lambda_{xu}(x) \) and send it to U.

- Compute \( \text{BEL}(X) = \lambda(x)\pi(x) \) and normalize

Most Graphs are not Polytrees

- Cutset conditioning
  - Instantiate a node in cycle, absorb the value in child’s CPT
  - Do it with all possible values and run belief propagation
  - Sum over obtained conditionals
  - Hard to do
    - Need to compute \( P(c) \)
    - Exponential explosion - minimal cutset desirable (also NP-complete)

- Clustering algorithm
- Approximate inference
  - Sampling methods
  - Loopy BP
Now To Add BBP to The BM

- A node in our hierarchy then represents a variable and is part of a larger, acyclic graph
- Child regions $Y_1$ and $Y_2$, parent $U$

1. $\lambda(x_j) = \prod_j \lambda_j(x_j)$
2. $\lambda(u_m) = \sum_x \lambda(x)P(x | u_m)$
3. $\pi(x_j) = \sum_u P(x_j | u)\pi_x(u)$
4. $BEL(x_j) = \alpha \lambda(x_j)\pi(x_j)$
5. $BEL(x_j) = \alpha \pi(x_j) \prod_{i\neq j} \lambda_i(x_i)$

Voilá - A “Bayesian Memory”
Neural Network Equivalent of BBP-PA

- For 4K CB entries
- No. Neurons ~ 32e3
- Synapses ~ 34e6
- NN derived from Hawkins' paper

BM-y

Parent-z

Repeat for all childs

HTM – Hierarchical Temporal Memory – Version 2
Numenta

- The HTM algorithm is the work of Jeff Hawkins and Dileep George
  
  - Jeff (Palm Pilot inventor) founded the Redwood Neuroscience Institute, [http://redwood.berkeley.edu](http://redwood.berkeley.edu)
  
  - From which has emerged a synthesis of a number of existing and new ideas of cortical operation
  
  - These are highlighted in his book, “On Intelligence”
  
  - The models have worked so well that he has now spun out a company, Numenta, Inc., [www.numenta.com](http://www.numenta.com)
  
  - Our work has borrowed heavily from Jeff and Dileep

Based on neuroscience principles, Jeff proposed that Cortex performs the following:

1. Learns sequences of patterns
2. Operates auto-associatively
3. Captures invariants
4. Is organized hierarchically, and
5. Based on fundamental Bayesian principles

- The George / Hawkins model starts with a fairly general Bayesian module, very similar to the BM presented earlier
  

- These modules then are combined into a hierarchy to form the Numenta *Hierarchical Temporal Memory* (HTM)
Hierarchical -- HTMs are organized as a tree-shaped hierarchy of nodes. Each node implements a learning and memory function, that is, it encapsulates an algorithm
- Lower-level nodes receive large amounts of input and send processed input up to the next level
- In that way, the HTM Network abstracts the information as it is passed up the hierarchy

Temporal -- During training, the HTM application must be presented with objects as they change over time
- For example, during training of the Pictures application, the images are presented first top to bottom, then left to right as if the image were moving over time
- Note that the temporal element is critical: The algorithm has been written to expect input that changes gradually over time

Memory -- An HTM application works in two stages, which can be thought of as training memory and using memory
- During training, the HTM Network learns to recognize patterns in the input it receives. Each level in the hierarchy is trained separately
- In the fully trained HTM Network, each level in the hierarchy knows -- has in memory -- all the objects in its world
- During inference, when the HTM Network is presented with new objects, it can determine the likelihood that an object is one of the already known objects.
- Level by level learning
- Each node
  - Stores co-occurrence patterns
  - Learns sequences

Markov chains (sequences)

Coincidence patterns

\[ N_{2,1}^{2,1} \]

\[ g_{1}^{2,1} \]

\[ g_{2}^{2,1} \]

\[ g_{3}^{2,1} \]

\[ g_{4}^{2,1} \]

\[ c_{1}^{2,1} = \begin{bmatrix} g_{1}^{1,1} & g_{2}^{1,2} \\ g_{2}^{1,1} & g_{2}^{1,2} \end{bmatrix} \]

\[ c_{2}^{2,1} = \begin{bmatrix} g_{1}^{1,1} & g_{1}^{1,2} \\ g_{2}^{1,1} & g_{2}^{1,2} \end{bmatrix} \]

\[ c_{3}^{2,1} = \begin{bmatrix} g_{1}^{1,1} & g_{2}^{1,2} \\ g_{3}^{1,1} & g_{3}^{1,2} \end{bmatrix} \]

\[ c_{4}^{2,1} = \begin{bmatrix} g_{1}^{1,1} & g_{3}^{1,2} \\ g_{2}^{1,1} & g_{2}^{1,2} \end{bmatrix} \]

\[ 0 \]

\[ c_{5}^{2,1} = \begin{bmatrix} g_{3}^{1,1} & g_{1}^{2,2} \\ 0 & 0 \end{bmatrix} \]

\[ c_{6}^{2,1} = \begin{bmatrix} g_{3}^{1,1} & g_{1}^{2,2} \\ 0 & 0 \end{bmatrix} \]

(2) Calculate likelihood of Markov chains
(3) Calculate probability of coincidence patterns
(1) Calculate likelihood of coincidence
(4) Calculate probability of Markov chains
What is the likelihood of Markov chain 1?

Dynamic programming calculates these likelihoods efficiently.

\[
\alpha_t(c_i, g_r) = P(-c_i|c_i(t)) \sum_{c_j(t-1) \in C^k} P(c_i(t)|c_j(t-1), g_r) \alpha_{t-1}(c_j, g_r)
\]
Hierarchy in space and time
- Evidence for biology abstracting in space and time as signals proceed up the hierarchy
- Feed-forward and feedback connections
- Common cortical algorithm
- Inference using Bayesian belief propagation
- Sparse Distributed Representations
- Prediction using temporal context
- Biologically accurate

Several computational vision applications
- www.numenta.com
In the PLOS paper, “Towards a Mathematical Theory of Cortical Microcircuits,” they also speculate on mapping the algorithms to cortical circuitry:

- Use known facts about cortical organization to map belief propagation to cortical layers
- “The vertical dimension of the cortical rectangle is only a few layers deep, the horizontal dimension is variable”
- The states of the region are represented by neurons along the horizontal dimension of the cortical region
- They then divide the horizontal dimension of the cortical region into a number of compartments, where each compartment corresponds to a particular state of the region
- This subdivision corresponds to a columnar organization of the cortex
Cortical Layers: Cells and Connectivity

Region-Region Pathways
- Feedforward
- Feedback
- Feedforward (gated)

From: Alex Thompson and Peter Barrister
It is an interesting time!

- SyNAPSE probably won’t meet its original goals, but it will push the field forward – assuming there is no catastrophic failure – or too much hype …
  - IEEE Tech Blog, "Cat Fight Brews Over Cat Brain"

- I personally believe that Jeff is on the right track
- And he is in this for the long haul, like the Terminator he will just keep on attacking this problem …
- And for the work I do, there are all kinds of interesting hardware possibilities – especially with nano and molecular electronics!

A Path From Nanowires to ISP

- Our approach is top-down, not bottom up
- There is a large range of implementation options
  - 1000 Atom processors
  - Neuromorphic VLSI
  - Nano-grids
  - Other nano …

Intelligent Signal Processing
Hierarchical Bayesian Network
Modular Bidirectional Spiking Associative Memory
Mixed Signal Nano-Scale Devices
Our Goal – A Commercial Product: The Field Adaptable Bayesian Array (FABA)

Each Square is a single Bayesian Memory Node

Nanoscale Analog Associative Memory

Nanoscale Analog Associative Memory

Thousands of nodes with full connectivity

CMOS provides sparse inter-module connectivity, I/O, signal amplification

Bayesian Memory Inside!

FABA – Long Term Goal

- A roughly 1 inch die containing several billion CMOS transistors and close to a trillion molecular devices
- Operating at over 10 Tera-Ops
- Extensive fault / defect tolerance
- Performs real-time, adaptive bayesian inference over very complex spatial and temporal knowledge structures
- Available in a portable, hand-held, low power devices