Domain Independent Perception in the LIDA Cognitive Software Agent

A Thesis

Presented for the

Master of Science

Degree

The University of Memphis

Robert Scott Brown

December 2006

Abstract

Brown, Robert S. M.S. The University of Memphis. December, 2006. Domain Independent Perception in the LIDA Cognitive Software Agent. Major Professor: Stan Franklin, Ph.D.

Perception, defined as the process of sensing, recognizing, categorizing and understanding the world around us, is a critical process in any cognitive system. Any autonomous agent, be it natural or artificial, which strives to behave intelligently in all but the simplest environments, must attain a robust level of perception. Such perception allows for the proper recognition of the agent's surroundings and the ability to assimilate, process and act upon sensory input from those surroundings. This paper will present an abstract software model representing a perceptual system that can be embedded in, and tailored for, virtually any environment for which sensory data can be digitally represented in a computer-based software system. The integration of this perceptual system into the existing LIDA cognitive software agent is used as the context for describing this model of perception as a crucial component of a comprehensive model of cognition.

Table of Contents

Table of Figures

1. Introduction

Autonomous Agents

Within the field of Artificial Intelligence, an autonomous agent is a system that is situated within and is a part of an environment; that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it senses in the future (Franklin & Graesser, 1996). Examples of agents span across both the biological and non-biological worlds. Biological agents include humans and most animals, while examples of non-biological agents include some robots and various software agents, including computer viruses. LIDA is an autonomous software agent built upon a theoretical cognitive model of mind.

2. Cognition and consciousness

The purpose of the LIDA system is to model minds; but what exactly do we mean by “mind”? A mind is a control structure for an autonomous agent (Franklin, 1995). In this regard, we are following the physicalist assumption that *minds are simply what brains do* (Minsky, 1985). Having based our system upon this definition of mind and the assumptions on which it is predicated (for a full discussion of this topic, see Franklin, 1995), our goal is then to formulate an architecture for cognition that is based upon – and supports through scientific experimentation – the known physical and functional architecture of the biological brain.

Although a formal treatment of human cognition and consciousness is beyond the scope of this paper, a brief description of these concepts and how this research has been applied to, and reflected in, the perceptual system described later may be useful.

Global Workspace Theory

To model the functional architecture of mind, LIDA is based on Baars' global workspace theory of consciousness (Baars, 1988). Like many others (e.g., Edelman 1987, Minsky 1985), Baars postulates that human cognition is the aggregate functionality of a multitude of relatively small, special purpose processes. Direct communication between these processes is rare and over a narrow bandwidth. This global workspace of processes serves as a means for the forming of coalitions of processes and the broadcast of coalition messages to all unconscious processes. This broadcast provides an opportunity for recruiting additional processes (i.e., schemas) for assistance in handling the current environmental situation. In this sense, consciousness provides a mechanism for dealing with novel or problematic situations that cannot be handled efficiently, or at all, by habituated unconscious processes.

All of this activity takes place under the auspices of contexts (see Figure 1): goal contexts, perceptual contexts, conceptual contexts, and/or cultural contexts. Baars uses goal hierarchies, dominant goal contexts, a dominant goal hierarchy, dominant context hierarchies, and lower level context hierarchies. Each context is itself a coalition of processes. Though contexts are typically unconscious, they strongly influence conscious processes. A key insight of global workspace says that each context is, in fact, a coalition of processors.

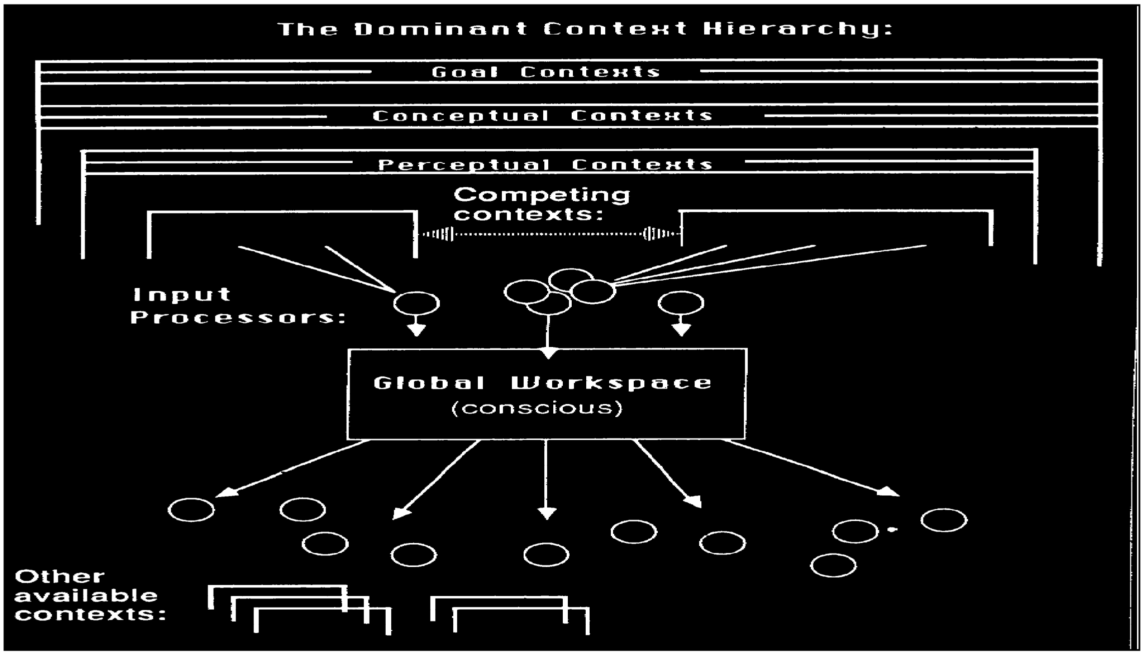

Figure 1. Global Workspace Theory

Baars postulates that learning results simply from conscious attention, that is, that consciousness is sufficient for learning. There's much more to global workspace theory, including attention, action selection, emotion, voluntary action, metacognition and a sense of self. One can think of it as a high level theory of cognition.

3. LIDA

Architecture

The LIDA architecture is composed of several more or less discrete modules described below.

- <u>Sensory Memory:</u>

  This is the area of the system in which sensory stimuli from both the internal and external environment are stored. This memory is extremely short term and represents, for example, patterns of sensation occurring on the receptive field of a particular sensor. This memory can be thought of as a volatile, short term buffer for perceptual memory. Sensory memory outputs direct activation patterns to the low level feature detector nodes of perceptual memory.

- Perceptual Memory:

  This area of the LIDA system is implemented as PAMnet and is therefore the subject of this paper. Perceptual memory allows for the recognition and categorization of information retrieved from sensory memory. The output of perceptual memory is a connected subset of the complete nodal structure of the network. This subset of linked nodes is called the percept and represents those perceptual elements recognized through sensation of the environment and becomes a candidate for being brought into consciousness.

- Working Memory:

  The contents of working memory are the current percept, as well as a set number of previous percepts. Working memory can be thought of as a preconscious buffer for LIDA’s various memory systems.

- Autobiographical Memory:

  This component represents those elements of memory for which the situational context of their learning can be recalled. For example, the first meeting of a close friend or the birth of a child would be examples of autobiographical memories. Unlike semantic memories, which do not retain this situational information related to the time of their being learned, autobiographical memories represent specific places and times in the life of the agent. Furthermore, these memories are very long term (they can span the entire life of the agent).

- <u>Transient Episodic Memory:</u>

  Similar to autobiographical memory, transient episodic memories retain the space and time elements of their situational context, but these memories are relatively short term (on the order of hours or days). Transient episodic memories that receive enough reinforcement – either through repetition or an emotional context – become candidates for long term autobiographical memory. An example of a transient episodic memory would be remembering where one parked one's car.

- <u>Long-Term Working Memory:</u>

  Long term working memory is so-called because it is a combination of those elements of long term autobiographical memory, as well as short term working memory combined to form a full-fledged memory construct that has relevance to the current situation.

- <u>Consciousness:</u>

  As discussed previously, consciousness in LIDA is based on the global workspace theory. It is therefore modeled as a number of sets of simple processing codelets, each of which is designed for a specific task. These codelets compete within the global workspace for an opportunity to come to consciousness based on their ability to comprehend and process the contents of long term working memory. The exact nature of this competition is described in section 5.2, *Cognitive Cycle*.

- Procedural Memory:

  Procedural memory contains sets of schemas representing those behaviors that have been previously learned by the agent. These behavioral schemas are generic, abstracted representations of actions and must be instantiated and bound before being executed.

- Behavior Net:

  The behavior net is the mechanism by which actions are selected and executed. Behavior streams (collections of associated actions with coupled pre- and post-condition values) are created from instantiated actions received from procedural memory. A behavior stream may represent a relatively long-term goal that will require several iterations of low-level action selection to successfully complete. New information coming to consciousness (and therefore recruiting conflicting coalitions of codelets) may cause a previously initiated behavior stream to be abandoned in favor of a new activity.

Cognitive Cycle

The functions of the LIDA system are enacted through the continuing iteration of a cognitive cycle. This cycle contains nine steps, though it is a cycle in the truest sense, with no set beginning or end point. As such, though ordered, any of the steps below can serve as an entry point for the cognitive process.

1) **Perception**. *Sensory stimuli, external or internal, are received and interpreted by perception producing meaning. Note that this stage is unconscious.*

a) ***Early perception**: Input arrives through senses. Specialized perception codelets descend on the input. Those that find features relevant to their specialty activate appropriate nodes in the PAMnet(a semantic net with activation).*

b) ***Chunk perception**: Activation passes from node to node in the PAMnet. The PAMnet stabilizes, bringing about the binding of streams from different senses and chunking bits of meaning into larger chunks of nodes which constitute the percept.*

Pertinent feeling/emotions are identified (recognized) along with objects, categories and their relations by the perceptual memory system.

2) ***Percept to Preconscious Buffer**. The percept, including some of the data plus the meaning, is stored in preconscious buffers of LIDA's working memory.*

In humans, these buffers may involve visuo-spatial, phonological, and other kinds of information. Feelings/emotions are part of the preconscious percept.

3) ***Local Associations**. Using the incoming percept and the residual contents of the preconscious buffers, including emotional content, as cues, local associations are automatically retrieved from transient episodic memory (TEM) and from declarative memory.*

The contents of the preconscious buffers together with the retrieved local associations from TEM and declarative memory, roughly correspond to Ericsson and Kintsch's long-term working memory (1995) and to Baddeley's episodic buffer (2000). These local associations contain records of the agent's past feelings/emotions, and actions, in associated situations.

4) ***Competition for Consciousness****. Attention codelets, whose job it is to bring relevant, urgent, or insistent events to consciousness, view long-term working memory. Some of them gather information, form coalitions and actively compete for access to consciousness.*

The competition may also include attention codelets from a recently previous cycle. Present and past feelings/emotions influence this competition for consciousness. Strong affective content strengthens a coalition's chances of coming to consciousness.

5) ***Conscious Broadcast****. A coalition of codelets, typically an attention codelet and its covey of related information codelets carrying content, gains access to the global workspace and has its contents broadcast.*

In humans, this broadcast is hypothesized to correspond to phenomenal consciousness. The conscious broadcast contains the entire content of consciousness including the affective portions. The contents of perceptual memory are updated in light of the current contents of consciousness, including feelings/emotions, as well as objects, categories and relations. Transient episodic memory is also updated with the current contents of consciousness, including feelings/emotions, as events. At recurring times not part of a cognitive cycle, the contents of transient episodic memory are consolidated into long-term declarative memory. Procedural memory (recent actions) is updated (reinforced) with the strength of the reinforcement influenced by the strength of the affect.

6) ***Recruitment of Resources****. Relevant behavior codelets respond to the conscious broadcast. These are typically codelets whose variables can be bound from information in the conscious broadcast.*

If the successful attention codelet was an expectation codelet calling attention to an unexpected result from a previous action, the responding codelets may be those that can help to rectify the unexpected situation. Thus consciousness solves the relevancy problem in recruiting resources. The affective content (feelings/emotions)together with the cognitive content, help to attract relevant resources (processors, neural assemblies) with which to deal with the current situation.

7) ***Setting Goal Context Hierarchy****. The recruited processors use the contents of consciousness, including feelings/emotions, to instantiate new goal context hierarchies, bind their variables, and increase their activation.*

It is here that feelings and emotions most directly implement motivations by helping to instantiate and activate goal contexts, and by determining which terminal goal contexts receive activation. Other, environmental, conditions determine which of the earlier goal contexts receive additional activation.

8) ***Action Chosen****. The behavior net chooses a single behavior (goal context), perhaps from a just instantiated behavior stream or possibly from a previously active stream.*

This selection is heavily influenced by activation passed to various behaviors influenced by the various feelings/emotions. The choice is also affected by the current situation, external and internal conditions, by the relationship between the behaviors, and by the residual activation values of various behaviors.

9) ***Action Taken****. The execution of a behavior (goal context) results in the behavior codelets performing their specialized tasks, which may have external or internal consequences, or both.*

This is LIDA taking an action. The acting codelets also include at least one expectation codelet (see Step 6) whose task it is to monitor the action and to try and bring to consciousness any failure in the expected results. We suspect that cognitive cycles occur five to ten times a second in humans, overlapping so that some of the steps in adjacent cycles occur in parallel (Baars & Franklin, 2003). Seriality is preserved in the conscious broadcasts.

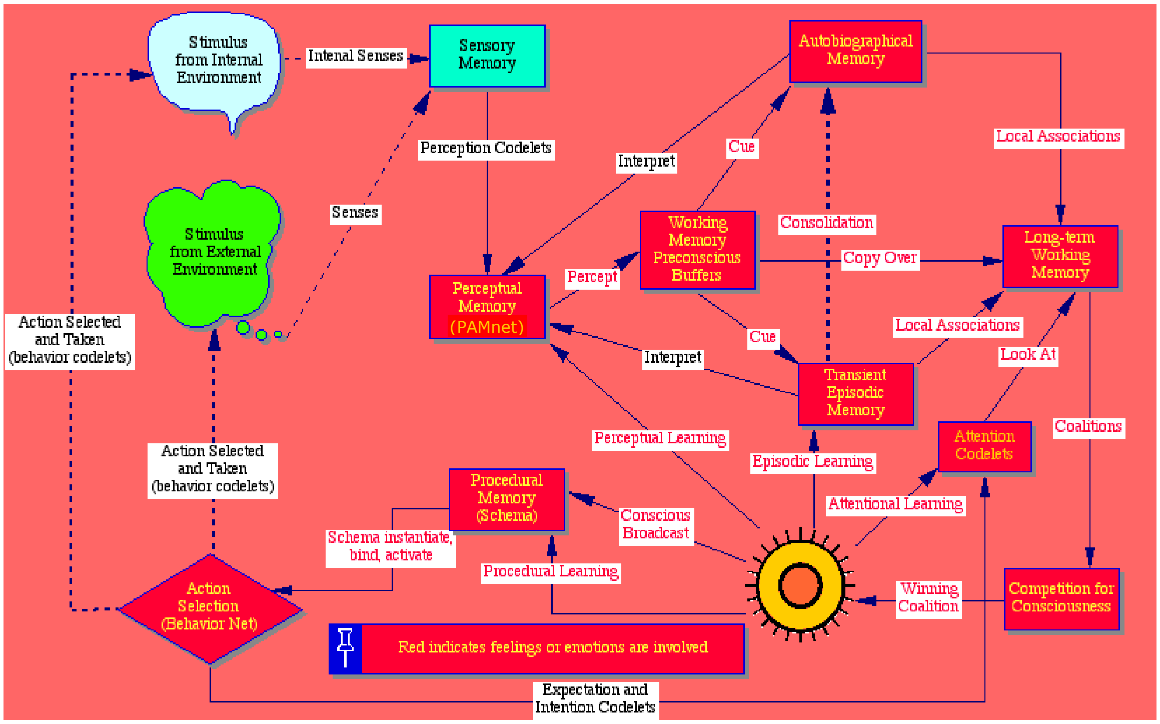
Stimulus from Internal Environment
Internal Senses
Sensory Memory
Autobiographical Memory
Perception Codelets
Interpret
Cue
Local Associations
Senses
Stimulus from External Environment
Working Memory Preconscious Buffers
Consolidation
Copy Over
Long-term Working Memory
Percept
Perceptual Memory (PAMnet)
Action Selected and Taken (behavior codelets)
Cue
Local Associations
Interpret
Look At
Transient Episodic Memory
Coalitions
Perceptual Learning
Attention Codelets
Episodic Learning
Action Selected and Taken (behavior codelets)
Procedural Memory (Schema)
Attentional Learning
Conscious Broadcast
Schema instantiate, bind, activate
Procedural Learning
Winning Coalition
Competition for Consciousness
Action Selection (Behavior Net)
Red indicates feelings or emotions are involved
Expectation and Intention Codelets

Figure 2. The LIDA Cognitive Cycle

4. Design and implementation

Overview

LIDA's perception module is known as the **P**erceptual **A**ssociative **M**emory **net**work, or PAMnet. Fundamentally, PAMnet is implemented as a semantic network with passing activation. A semantic network is a directed graph of nodes that represent definitional information about a particular domain (Sowa, 1992). In PAMnet, this graph of nodes is an abstract framework that is well-suited to building a semantic repository that can represent many different kinds of perceptual knowledge. Given a perceptual learning mechanism, the definitional context of the network can be built iteratively over time from a relatively sparse set of initial content. In practice, PAMnet can be connected to a domain-specific sensory system (e.g., a camera) and the nodes of the network represent both highly contextualized features (i.e., groups of colored pixels) as well as abstracted objects and concepts (e.g., a face or an apple).

Network Structure

Architecturally, PAMnet is structured as a hierarchical network of nodes. Conceptually, PAMnet nodes can be thought of as analogous to Barsalou's perceptual symbols (Barsalou, 1999). Typically, this hierarchy of nodes is visually represented as a tree or forest structure and PAMnet is no exception (typical discussions of PAMnet behavior make liberal use of terms such as "top-down" or "bottom-up"). Often, PAMnet's hierarchy is represented as a radial network with the "root" being the center of an ever-growing circular structure of inter-connected nodes (Figure 3).

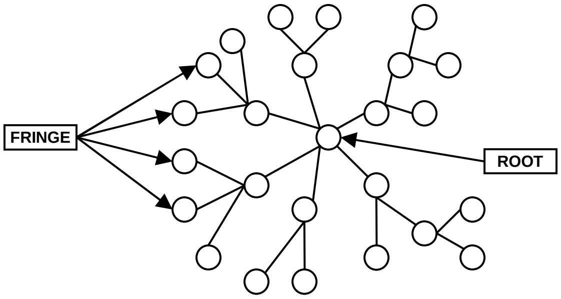

*Figure 3. C*onceptual view of PAMnet

PAMnet also makes use of the notion of conceptual depth (Mitchell, 1993). Originally used in Mitchell’s slipnet system, conceptual depth is a measurement used to objectively represent the level of complexity of a particular perceptual component. For example, low-level feature detector nodes that respond directly to sensory stimuli (e.g., visual stimulation representing a line segment at a particular angle) are considered to be at extremely low conceptual depth. Such nodes can be imagined to exist along the fringe of the network and, indeed, are located thusly in the PAMnet implementation. By contrast, a node representing a human face would be at very high conceptual depth and reside closer to the network’s center. Such high-concept nodes, in fact, aggregate many lower-level nodes which, in sum, feed connectivity into (“fan-in”) the higher-level object or category. For example, many different nodes representing line segments at various angles are connected to slightly higher-level nodes representing basic shapes (for example, three low-level feature detectors representing “╱” “╲” and “_” all might feed into a “triangle node”, representing the visual object “△”). These basic shapes feed into slightly more complex nodes, which can be thought of as analogous to geons in the sense of Biederman (1987), which are likewise connected to even higher concept nodes, which might represent real-world objects. As we traverse the network from lower to higher conceptual depth, the “meaningfulness” of nodes increases.

Primitive feature detectors – those nodes at the very lowest conceptual depth – exist at the extreme fringe of the network and are directly connected to sensory receptors. As such, activation levels are absolute for these nodes; i.e., they are either on or off depending solely on whether or not they are receiving stimulation from receptors (though this need not necessarily be the case; for other domain implementations, it may be useful to model even primitive feature detectors with continuous levels of activation). The sensory receptors that connect to primitive feature detectors are themselves directly coupled with the environment. Because of this explicit coupling with the environment, these areas of the system are, in fact, domain *dependent*. The domain dependent areas of the system include the sensory receptors, as well as the infrastructure that manages their coupling to primitive feature detectors. Further, the feature detectors themselves are domain dependent and their contents, configuration and topology are set permanently – they are not governed by the same learning processes that constantly alter the makeup of the rest of the nodes in PAMnet. This domain dependence is to be expected and, indeed, is required for any system designed to operate in an environment. For example, the human brain is remarkably flexible and adaptive, with neural pathways and connections being established, abolished and re-established constantly. However, our sensory organs, the central nervous system that directly connects to the brain, and the front-line neural cell assemblies that react directly to sensory input are governed by DNA rather than environment and are relatively unchanging and permanent. Likewise, any artificial system that operates in an environment (and satisfies the Franklin/Graesser definition of *agent*) will require some level of domain dependence in its sensory array.

Node Structure

Perceptual knowledge (that is, an abiding understanding of the environment in which the agent lives) is implicitly represented in PAMnet by the combination of the content of the nodes and the particular arrangement of those nodes in the network at any given time. Both the content and arrangement of nodes will be constantly changing, particularly in a learning-enabled version of PAMnet. The content of each node is composed of four elements: 1) the node identifier: a machine-level globally unique ID used to track and distinguish the node from others, 2) base level activation: a numeric value, learned over time, representing a node's default activation level, 3) current activation: a numeric value representing the total level of activation currently within the node and 4) a list of links for the other nodes in the network to which this node is connected.

Node ID

The node's ID is a unique identifier and is used programmatically for tracking, organizational and lookup purposes. This ID should not be thought of as the "name" of the node. For purposes of discussion, it is often much easier for humans to refer to named nodes in the PAMnet, such as the "chair node" or the "apple node". However, in a fully embodied learning agent employing a PAMnet implementation for its perceptual system, the meaning of any particular node (from an outside observer's perspective) may change over time because the contextual landscape of the network will be constantly changing due to the learning process. However, *a node's ID remains static and* *unchanging*. For example, a node that starts out as a newly-learned object node will relatively soon be placed under the context of a category node, as other members of the category are learned and assimilated. This can be thought of as analogous to the way small children learn. A young child, upon first encountering and learning to recognize a pet dog, will then apply the label "dog" to many similar four-legged creatures. However, over time the child learns the difference between dog, cat, cow, pig, etc (this is analogous to the notion of "constructivism" from Cognitive Psychology (Piaget, 1972). In this context, what starts out as a "dog" category might evolve over time to become "pet", "mammal", ultimately even "animal", with more specific category and object nodes added as children in the subtree extending beneath the node at lower conceptual depth. From an implementation perspective, the node's ID has remained constant and unchanging, even though the meaning of the node has evolved over time, as the learning process changes the network structure.

Base Level Activation

Base level activation is used in perceptual learning and, as such, the values are static in the current PAMnet implementation. Nodes with a high base level activation represent those elements of perceptual memory that have been reinforced over time, through learning, such that they achieve higher total activation levels (total activation being a function of a node's base and current activation levels) – and thus, inclusion in the percept – much easier than newer, relatively unlearned, nodes. Through the learning process, those nodes which make it to consciousness again and again have their base level activation reinforced. All nodes have their base level activation decay over time. The decay rate follows an inverse sigmoidal function of the current value of the base level activation.

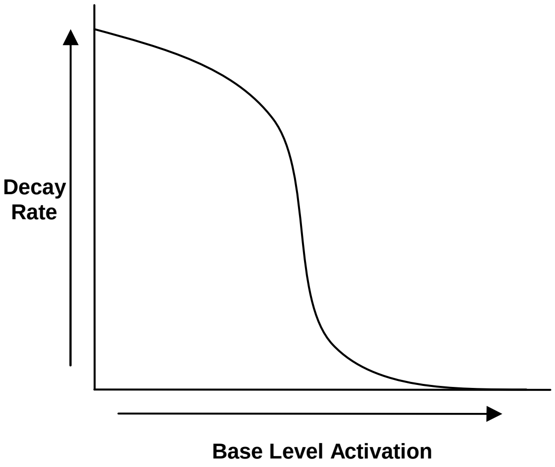

Figure 4. Behavior of Base Level Activation Decay Function

The higher the base level activation (and thus, the more reinforcement a node has received), the slower its decay rate. This behavior implements the notion that those nodes which receive reinforcement over and over are more likely to be useful, until they reach a point of relative permanence in perceptual memory. Conversely, those nodes which are created through the profligacy of learning, but are ultimately not useful (indeed, the vast majority of nodes created through perceptual learning fall into this category) will decay away and will be removed from the network. This "survival of the fittest" approach echoes evolutionary processes found throughout nature.

Current Activation

A node's current activation level is constantly changing as activation is received from input connections, and quickly decays over time. At any given point in time, a node's total activation (α) is equal to the sum (or some other function) of its base level activation (β) and current (γ) activation.

$$\alpha = \beta + \gamma$$

Figure 5. Node Total Activation

Like the base level decay rate, a node's current activation also experiences decay. However, the decay rate for a node's *current* activation is very swift, with most nodes losing current activation in less than one second. Through computational metering, the current implementation of PAMnet experiences a single cognitive cycle every 200 milliseconds. With each cycle, all nodes in PAMnet lose a percentage of their current activation as a function of the value of the nodes' conceptual depth (δ). The value of a node's current activation cannot fall below zero.

$$\%\ decay = 100 - \delta$$

Figure 6. Percentage of Node Current Activation Decay Each Cycle

Conceptual Depth

Conceptual depth (δ) is represented in PAMnet as a numerical value with a minimum of zero (indicating the node as a primitive feature detector) and a normalized maximum value of 100 (there is no upper bound to the non-normalized raw value of a node's conceptual depth). As stated previously, a node's conceptual depth represents its level of complexity, abstractness and meaning. These are relatively fuzzy concepts, but the general notion is that the higher a node's conceptual depth, the more meaningful that node is from a cognitive perspective. A node's conceptual depth is calculated as the product of its level of input linkage (κ) and its "layer depth" (λ).

$$\delta = \kappa \cdot \lambda$$

Figure 7. Calculation of Conceptual Depth

A node's input linkage, κ, represents the number of links feeding into this node, both directly and indirectly. Therefore, the κ value for a particular node is the sum of the κ values of all of its children, plus the number of its children (since each child node will provide exactly one input link to the parent).

The layer depth, λ, of a node is simply a measurement of a node's distance from the fringe (i.e., how many "layers" deep the node resides). It is simply calculated as the fewest number of steps needed to traverse the network from any fringe node to the node in question. As the structure of the network changes, a node's linkage value and layer depth are re-calculated, resulting in a dynamic update of the node's conceptual depth value.

Activation and Activation Passing

Nodes at higher conceptual depth require more activation than lower-concept nodes in order to be above threshold and eligible for inclusion in the percept. Therefore, the activation threshold (τ) of a node is calculated as a function of the node's conceptual depth.

$$\tau = \delta \cdot U \cdot R$$

Figure 8. Calculation of Activation Threshold

In the formula above, U is the activation base unit and R is the threshold increase rate. Both U and R are simple constants used as scalar values for dynamically calculating a node's activation threshold. In the current implementation of PAMnet, U is set to 50 and R is set to 1.05. These values were arrived at through experimentation (see section 7, *Testing*).

When a node's activation level rises above threshold, it is considered to be a part of the percept. Conceptually, it is often convenient to think of nodes in PAMnet as being "on" or "off", depending upon whether the node's activation level is above or below threshold (and, in fact, nodes are visually represented this way in the ALife environment created for prototyping and experimentation – see section 7, *Testing*). However, it is important to remember that, although nodes can be represented in a binary state – "on" or "off" (i.e., above or below threshold) – they actually reflect a range of activation values of zero and above. This makes PAMnet nodes well-suited for representing continuous or "fuzzy" domains that demand subtle representations of environmental qualia (for example, "tallness" or "coldness").

In addition to becoming part of the percept when its activation rises above threshold, a node also passes on its activation level to its parent node(s). In the radial diagram shown in figure 1 (see section 6.2, *Network Structure*), this can be envisioned as activation passing inward from the fringe of the network (lower conceptual depth) towards the center (higher conceptual depth). In the current PAMnet implementation, each node passes its full activation value to its parent node(s).

The processes described above – current activation decay, current activation excitation and activation passing – occur each and every cognitive cycle as part of a synchronous and serial process. In biological systems, cognitive cycles are thought to be asynchronous and cascading, with consciousness being serial. Future implementations of PAMnet could take advantage of a multi-threading approach in order to achieve some level of parallelism in node processing. However, in any non-trivial environment (especially if perceptual learning is enabled), the likely number of extant nodes will dictate some “grouping” of nodes per thread. It’s very unlikely that each node would have its own thread devoted to its individual computational needs.

Domain Interoperability

A key component of the domain independence achieved by PAMnet is the ***easy*** and ***rapid*** application of the system across multiple domains. In order to facilitate this interoperability between domains, the PAMnet system was designed from the beginning to have a static, serialized XML representation. This file-based representation allows for the easy transfer of domains between systems, offline manual editing of domains for knowledge engineering purposes, as well as the easy application of PAMnet to several different domains with minimal re-tooling necessary on the part of the user.

Import and export features are built directly into the PAMnet system. The export feature allows the user to create a “snapshot” of a running network at any given time. Importing from a previously saved (or manually created) XML representation will clear current memory contents and load an entirely new PAMnet structure. In addition to allowing for the storage and reloading of the network’s structure, all node contents are also serialized to file, including both current and base activation levels. In this way, the actual state of perceptual memory at any given moment in time can be exported, stored and retrieved later allowing for great flexibility in testing and experimentation.

The structure of the offline file representation is a proprietary XML format reflecting the structure and architecture of the PAMnet implementation (see Figure 9).

```
<PAMnetXML>
  <NodeXML>
    <nodeID>d4103b32-1b6b-4b78-b1c5-2f72b2f5979b</nodeID>
    <nodeLabel>LEFT_SEMI</nodeLabel>
    <baseLevelActivation>0</baseLevelActivation>
    <currentActivation>0</currentActivation>
    <linkedNodeId>0861658c-8744-4e0b-9255-9d63160e2e74</linkedNodeId>
  </NodeXML>
  <NodeXML>
    <nodeID>481e77da-9a51-47a9-9c8b-791c33a5f2bc</nodeID>
    <nodeLabel>RIGHT_SEMI</nodeLabel>
    <baseLevelActivation>0</baseLevelActivation>
    <currentActivation>0</currentActivation>
    <linkedNodeId>0861658c-8744-4e0b-9255-9d63160e2e74</linkedNodeId>
  </NodeXML>
  <NodeXML>
    <nodeID>af8b93e9-6622-4868-afc0-64476a8abf60</nodeID>
    <nodeLabel>HORIZ_LINE</nodeLabel>
    <baseLevelActivation>0</baseLevelActivation>
    <currentActivation>0</currentActivation>
    <linkedNodeId>513a8365-2b0a-49f4-af27-e5a98d820327</linkedNodeId>
    <linkedNodeId>63cd341e-681f-4279-b2e4-27c7bd18535f</linkedNodeId>
  </NodeXML>
…
</PAMnetXML>
```

Figure 9. Sample PAMnet XML File

The root node of the PAMnet'x XML representation is named `<PAMnetXML>` and contains one or more `<NodeXML>` elements. Each `<NodeXML>` element contains several elements, each pertaining to the logical contents of a PAMnet node, as well as two additional data elements. A node "label" is a human-readable name for the node. This label is strictly optional and has no bearing on PAMnet's runtime functionality. If present, however, the label will be used in the visualization feature in the current PAMnet Java application. This visualization feature displays a network representation of PAMnet to the user for informal inspection of the network contents and structure. If present in the XML, the label of each node will also be displayed. In addition, one or more `<linkedNodeId>` elements may be present within a node's XML representation. These elements represent the identifiers of all nodes to which this node is connected in a feed-forward fashion (i.e., increasing conceptual depth). In this way, the PAMnet application can traverse the network in a uni-directional fashion by reading the contents of the XML file in order to re-construct the network. Though this reconstruction is done by traversing the XML contents serially and building the network from the bottom up, do note that, once constructed, the current PAMnet implementation does store links as bi-directional connections and, thus, also allows for a top-down network traversal.

Development Environment

PAMnet was implemented entirely in Java, specifically using Sun Microsystem's Java Development Kit (JDK), version 1.5.

5. Testing

This section describes the application environment used to test the PAMnet implementation, as well as difficulties encountered during the implementation, testing process used, and parameter tuning done as part of the project's development.

Domain

Testing of PAMnet was done using a simple Artificial Life (ALife) domain specially constructed for the purpose of providing a framework for experimentation. This ALife domain consists of a 4x4 tiled gridworld containing a simulated agent program and several simple domain world objects.

In this ALife implementation, the agent itself has no automated behavior mechanism or any other cognitive functionality besides perception via PAMnet. A human user must control the agent's movements through the gridworld using computer keyboard controls.

The objects that exist in this ALife world are simple and discrete software structures that follow a very basic pattern of composition. Each object is composed of a set of primitive features that correspond exactly to an array of fringe nodes (primitive feature detectors) in the PAMnet structure. For example, the “Rock” object (see Figure 10), the most basic object in the domain, is composed of two semi-circles (◖,◗). An “Apple”, by comparison, is also composed of two semi-circles, but also has a vertical line. Color, though shown for clarification, is not a feature of the domain world – that is, the agent does not sense colors. As the agent moves around the domain world, it senses the features of objects anytime it enters a square occupied by an object. The primitive features that the agent can sense include semi-circles, vertical lines, horizontal lines and smells (olfactory stimulation). The smells have no visual representation in the worldview above (though they could be given one, if desired), but consist of five different odor primitives, combinations of which are given off by the tree, apple and orange objects.

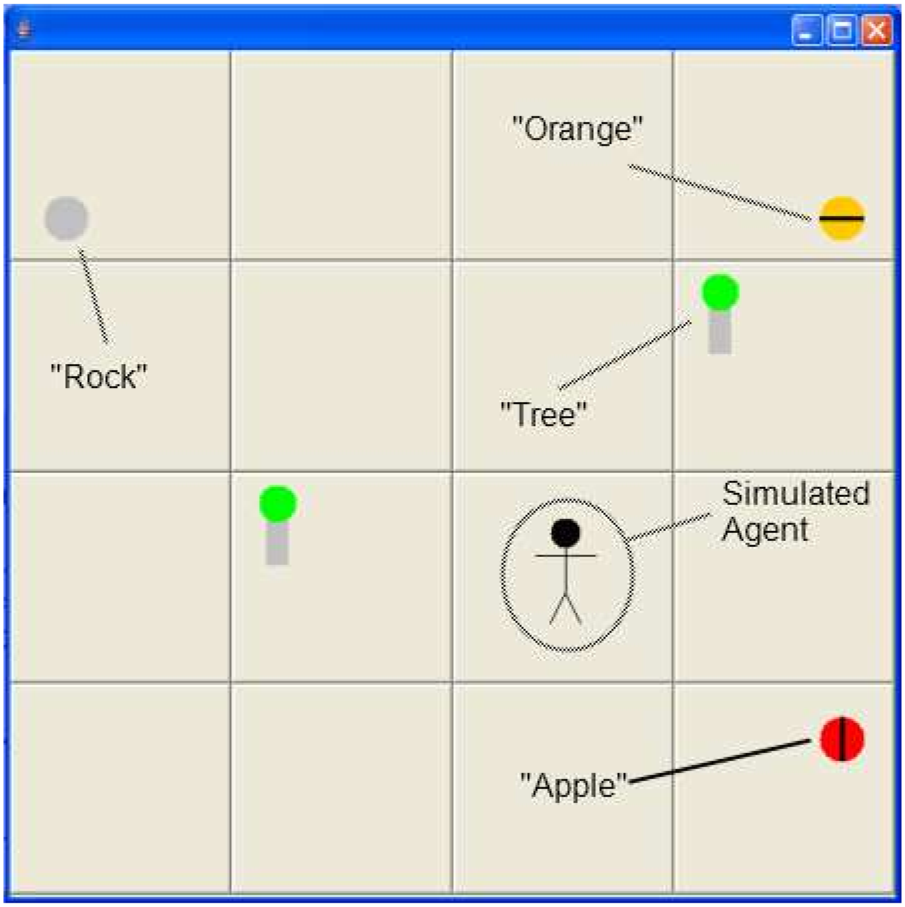

Figure 10. Experimental ALife Domain with Labels for World Objects

When the agent enters a grid space containing an object, the agent senses the object through the activation of feature detectors corresponding to the constituent features of the object encountered. For example, when a space containing a "Rock" object is encountered, the "left semi-circle" and "right semi-circle" primitive features are sensed and activated.

Since learning has not been enabled for this version of PAMnet, a network structure that corresponds to this domain world was constructed manually. It is this PAMnet structure that was used as the basis for testing, experimentation and parameter tuning as part this project (see Figure 11).

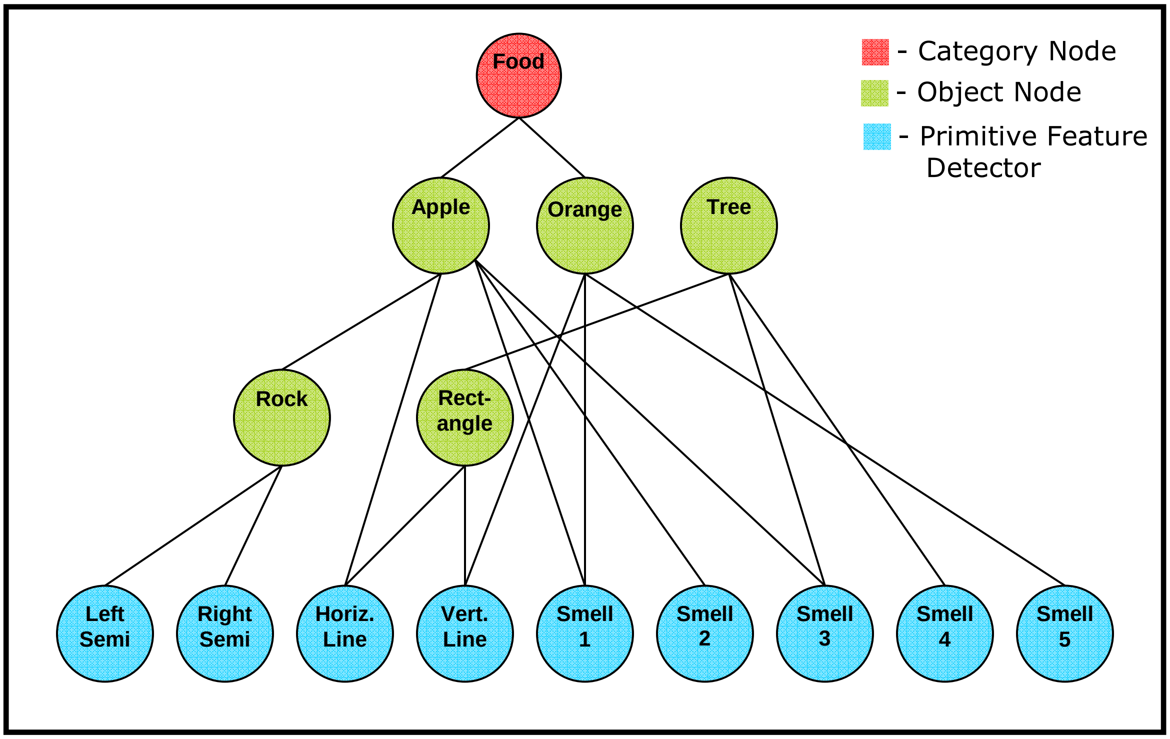

Figure 111. PAMnet Structure for ALife Domain

Note that the rectangle object, though represented within the PAMnet, is not actually an extant object itself in the domain world (it's a higher-level feature detector). It only exists as a component of the "Tree" object and is used as an internal representation and abstraction of a "Tree" object's more primitive features (i.e., lines and intersections).

Difficulties

During the implementation of PAMnet, a few difficulties had to be overcome. Much of PAMnet's structure and functionality was based on the Slipnet component of Hofstadter and Mitchell's Copycat architecture (Mitchell, 1993). However, Mitchell's initial Slipnet implementation was done in a domain-specific way, intended for application as an analogy-making inference engine. As such, several modifications were necessary in order for this approach to be viable in a domain-independent fashion. Specifically, Mitchell's calculation of conceptual depth for each node of the Slipnet was hard-coded and arbitrarily assigned, based on the context of the node within the larger problem space. For this domain independent version, a conceptual depth calculation was arrived at (see section 6.3.4), such that the conceptual depth value for any given node could be calculated automatically.

In addition, two free variables were introduced into the PAMnet system that required considerable tuning. These are U, the activation base unit and R, the threshold increase rate. Both of these variables are involved in the activation passing function and required tuning through trial and error, in order to strike the correct balance with PAMnet's current activation decay function. When tuned improperly, two erroneous behaviors were observed:

1. When R was tuned too low, the decay function dominated, resulting in few or no nodes ever achieving threshold and, subsequently, the object/category nodes being sensed never achieved activation.
2. When U was tuned too high, the activation passing function dominated. In turn, not only did the relevant objects receive activation above threshold, so did every other connected node in the network, disabling the agent's recognition capabilities.

When both variables were properly tuned, only the relevant nodes corresponding to the domain objects being perceived in the ALife world (as well as the lower-concept nodes feeding into them) received activation above threshold. Though activation was still being iteratively injected into the network, the system stabilized very quickly into the correct configuration.

6. Conclusion

Through the application of concepts, models and theories from multiple scientific disciplines, including Computer Science, Artificial Intelligence, Cognitive Psychology and Neuroscience, the LIDA system is taking a deliberate step towards the goal of creating a unified cognitive model within the context of an autonomous software agent. As part of that effort, PAMnet provides a robust, scaleable and domain-independent perceptual mechanism upon which future work may be based. Most importantly, it forms the framework with which a perceptual learning mechanism can be built. The nodal structure of PAMnet also provides the common currency for processing throughout the rest of the LIDA system, including its memory and behavior modules. In this way, PAMnet provides a clean, stable and uniform architecture for LIDA, as well as a functional perceptual system that can be applied to virtually any domain or autonomous agent system.

7. References

Baars, B. J. (1988). *A Cognitive Theory of Consciousness*. Cambridge: CambridgeUniversity Press.

Baars, B. J. (1997). *In the Theater of Consciousness*. Oxford: Oxford University Press.

Baars, B. J., and S. Franklin. (2003). *How conscious experience and working memory interact.* Trends in Cognitive Science 7:166–172.

Baddeley, A. D. (2000). *The episodic buffer: a new component of working memory?* Trends in Cognitive Science 4:417-423.

Barsalou, L. W. (1999). *Perceptual symbols systems*. Behavioral & Brain Sciences, **22**, pp 577-660.

Biederman, I. (1987). *Recognition-by-Components: A Theory of Human Image Understanding*. Psychological Review, 94, 115-147.

Edelman, G. M. (1987). *Neural Darwinism*. New York: Basic Books.

Ericsson, K. A., and W. Kintsch. (1995). *Long-term working memory*. Psychological Review 102:211-245.

Franklin, S. (1995). *Artificial Minds*. Cambridge MA: MIT Press

Franklin, S. (2005). *Perceptual Memory and Learning: Recognizing, Categorizing, and Relating*. Symposium on Developmental Robotics; American Association for Artificial Intelligence (AAAI). Stanford University, Palo Alto CA, USA. March 21-23, 2005.

Franklin, S., and Graesser, A. 1999. *A Software Agent Model of Consciousness.* Consciousness and Cognition 8:285-305.

Franklin, S., and Graesser, A. (1996). *Is it an Agent, or just a Program? Taxonomy for Autonomous Agents*, Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Vienna, Austria: Springer-Verlag.

Kanerva, P. (1988). *Sparse Distributed Memory*. Cambridge MA: The MIT Press.

Minsky, Marvin. (1985). *Society of Mind*. New York: Simon and Schuster.

Mitchell, M. (1993). *Analogy-Making as Perception*. Cambridge, MA: MIT Press/Bradford Books.

Piaget, J. (1972). *Science of education and the psychology of the child* (Rev. ed.). New York: Viking.

Sowa, J. F. (Eds.). (1992). *Principles of Semantic Networks*. San Mateo, CA: Morgan Kaufmann Publishers.