AFRL-HE-WP-TR-2006-0160

# Distributed Cognition (DCOG): Foundations for a Computational Associative Memory Model

**Robert G. Eggleston**

**Cognitive Systems Branch**
**Wright-Patterson AFB OH 45433**

**Katherine L. McCreight**

**N-Space Analysis**
**306 Winding Trail**
**Xenia, OH 45385**

**August 2006**

**Interim Report for February 2001 to August 2006**





**Air Force Research Laboratory**
**Human Effectiveness Directorate**
**Warfighter Interface Division**
**Cognitive Systems Branch**
**WPAFB OH 45433-7604**

# NOTICE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory, Human Effectiveness, Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

**AFRL-HE-WP-TR-2006-0160** HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

**FOR THE DIRECTOR**

//SIGNED//

DANIEL G. GODDARD
Chief, Warfighter Interface Division
Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

# REPORT DOCUMENTATION PAGE

*Form Approved OMB No. 0704-0188*



| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From - To)* |
|---|---|---|
| August 2006 | Interim | February 2001 - August 2006 |

**4. TITLE AND SUBTITLE**
Distributed Cognition (DCOG): Foundations for a Computational Associative Memory Model

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**
61102F

**6. AUTHOR(S)**
[1]Robert G. Eggleston, [2]Katherine L. McCreight

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**
2313HC1A

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

[2]N-Space Analysis
306 Winding Trail
Xenia, OH 45385

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
[1]Air Force Materiel Command
Air Force Research Laboratory
Human Effectiveness Directorate
Warfighter Interface Division
Cognitive Systems Branch
Wright-Patterson AFB OH 45433-7604

**10. SPONSOR/MONITOR'S ACRONYM(S)**
AFRL-HECS

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**
AFRL-HE-WP-TR-2006-0160

**12. DISTRIBUTION / AVAILABILITY STATEMENT**

Approved for public release; distribution is unlimited.

**13. SUPPLEMENTARY NOTES**

AFRL/PA Cleared on 22 February 2007, AFRL/WS-07-0348.

**14. ABSTRACT**

Computational models of human behavior and performance continue to gain in importance for the development and use of tools to support Air Force personnel in warfighting operations. In this report, we describe the foundations of a different type of computational architecture; one that we believe will be less susceptible to cognitive brittleness and can better scale to complex and ill-structured work domains. More specifically, it describes an investigation into the design and software implementation of an associative style memory model. The memory model expresses knowledge at three fundamental levels of granularity as opposed to one found in other machine representations of cognition. The report describes the theoretical constructs used in the memory model; illustrates how they can be used to account for certain cognitive phenomena such as over generalization of categories by children and base rate neglect in decision making; develops these concepts mathematically and algorithmically; and illustrates how they capture cognition in a concept leaning task. It concludes with a software implementation plan that embeds the memory model into a software actor Distributed Cognition (DCOG) that employs a distributive software agent architecture.

**15. SUBJECT TERMS** Distributed Cognition (DCOG), Human Behavioral Representation, Cognitive Model, Human Performance Model, Computational Associate Memory Model

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Robert G. Eggleston |
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | SAR | 128 | 19b. TELEPHONE NUMBER *(include area code)* |

**Standard Form 298 (Rev. 8-98)**
**Prescribed by ANSI Std. 239.18**

THIS PAGE LEFT INTENTIONALLY BLANK

# Table of Contents

# List of Figures

# List of Tables

## ABSTRACT

Computational models of human behavior and performance continue to gain in importance for the development and use of tools to support Air Force personnel in war fighting operations. For example, software agents that emulate human cognitive abilities are embedded in many software applications that aid war fighter activities, and software actors are often critical components in simulations used for war gaming and weapon system acquisition activities. Current approaches to computationally model cognitive abilities and behavior can be used successfully for well defined, narrowly scoped, analysis, design aspects of software tool and simulation developments. However, these approaches appear to have difficulty scaling to provide broad, robust and intelligently adaptive cognitive behavior that is needed to meet all aspects of software tool and simulation development. The prevailing computational cognitive architectures currently require substantial hand crafting to meet the needs of a support tool or a simulation environment for use in war gaming or acquisition activities. If this additional tailoring is not done, then the cognitive models will evidence reduced cognitive ability and cognitive brittleness or both in the application context. Improved computational cognitive models are needed to redress these limitations.

In this report, we describe the foundations of a different type of computational architecture; one that we believe will be less susceptible to cognitive brittleness and can better scale to complex and ill-structured work domains. More specifically, it describes an investigation into the design and software implementation of an associative style memory model. The memory model expresses knowledge at three fundamental levels of granularity as opposed to one found in other machine representations of cognition. Moreover, these knowledge forms are all linked consistently through a common associative feature based architecture from which two of these knowledge forms are largely emergent. This provides a way to reduce cognitive brittleness and preserve the availability of full cognitive ability in complex and novel work situations.

The report describes the theoretical constructs used in the memory model, illustrates how they can be used to account for certain cognitive phenomena such as over generalization of categories by children and base rate neglect in decision making; develops these concepts mathematically and algorithmically; and illustrates how they capture cognition in a concept leaning task. It concludes with a software implementation plan that embeds the memory model into a software actor (DCOG) that employs a distributive software agent architecture.

## 1.0 INTRODUCTION

Computational models of human behavior and performance continue to gain in importance for the development and use of tools to support Air Force personnel in war fighting operations. Software agents that emulate human cognitive abilities, for example, are embedded in many software applications that aid war fighter activities. Cognitive factors may be used in software to aid the war fighter in many ways: e.g. to determine what, when, and where task-critical alerts need to be provided, to interpret user input to a software tool or weapon system and act on that interpretation, to assist in information retrieval over a distributed network, to assist general aspects of knowledge management in planning, coordination, and execution, and to aid specific task details included in war fighter jobs such as the cognitively demanding work activities found in a theater Air Operations Center. Other areas where cognitive models are of value to the war fighter include war game simulations, and simulations used in the acquisition of future weapon systems and for upgrades to existing weapon systems.

The degree of cognitive-based sophistication and automation needed in war fighter applications and simulation tools continues to grow. More cognitive ability of machine tools are needed to help the user handle higher volumes of available information and to handle more intricate and complex task scenarios against a smart, evolving adversary.

A recurring weakness with the current and past generations of cognitive models is that they tend to be brittle. The research and development community has the knowledge to build cognitive models to meet specific task goals, but the modeling solutions are rather limited in the degree to which they can adjust to evolving conditions in a manner that

preserves their level of intelligence when adapting to adversary actions. In other words, the cognitive models lose the quality of their cognitive ability and behave in a less expert, more predictable manner under 'novel' situations. This *cognitive brittleness* results in degraded performance of support tools and introduces unrealistic software agent behaviors in simulations that, in turn, adversely impact analytic assessments being made with them. To better meet the demand of the acquisition and war fighting communities, we need a new generation of cognitive models that are able to maintain their cognitive quality and avoid cognitive brittleness.

We believe that a new approach to the design of a computational cognitive architecture is needed to provide a persistent and robust solution to the cognitive brittleness problem. Knowledge representation is at the heart of the issue. Typical approaches to encoding knowledge use an encoding strategy that employs a single grain size knowledge element. Further, all knowledge is made explicit all of the time within this knowledge element structure. For example, a very popular cognitive architecture, ACT-R, uses a 'chunk' data structure as an atomic element of thought to encode declarative knowledge (Anderson and Labiere, 1998). One difficulty with this architectural approach is that knowledge storage and usage time increases exponentially as the work domain or situational complexity increases. This is a problem for all current computational cognitive architectures.

In this report, we describe the foundations of a different type of computational architecture; one that we believe will be less susceptible to the cognitive brittleness problem. Specifically, it addresses the principles and characteristics of an associational memory model as part of the Distributed Cognition computational framework. This

approach to encoding knowledge is able to handle multiple knowledge grain sizes and can keep most knowledge in a latent, implicit form until activated based on the characteristics of the emerging situation. Thus, storage demand and retrieval time can be better bounded.

Another important characteristic of our approach is a distributed cognitive architecture that consists of multiple functional regions that operate in parallel under local control. In other words, there is no single, central mechanism that provides control for the system. It is on the basis of this property, and the heavy emphasis on cognitive regions, that we identify our framework (and resultant computational models) by the label, *Distributed Cognition.* The first computational instantiation of this framework, known as DCOG-1, demonstrated the distributed knowledge and control characteristics (Eggleston, Young and McCreight, 2000, 2001) [A more detailed description of DCOG-1 is presented in Eggleston, McCreight and Young, 2005.] Due to schedule considerations, a flexible associative memory model was not included in DCOG-1. The research reported here addresses this aspect of the DCOG framework, which now has been demonstrated in our second DCOG model (DCOG-2).

The report begins with a detailed description of the DCOG associative memory model. We then briefly discuss the memory model in the context of different levels and types of views of human cognition and behavior. This is followed by a discussion of our memory model structure in relation to the cognitive work of solving a category learning problem. The report concludes with a discussion of the software implementation of the model.

## 2.0 DCOG Associative Memory Model

### 2.1 Introduction

The DCOG memory model uses distributed cognition to model human abilities of categorization and decision-making. The basis of the DCOG model is a non-linear, recognition-based model of associative memory, drawing on the framework of Q-analysis (Atkin, 1974). Input features are represented by mental nodes; co-active nodes become linked by associations of various strengths. These associations encode memory.

The problem for any model of associative memory is how do you preserve useful information without suffering from information overload?

One strategy is to reduce the input by representing several features with one symbol. For example, the perceptual input associated with seeing a cat (four legs, short ears, long tail, furry, black) might be summarized and stored by the symbol, cat. Problems arise when we try to decide just which input goes with what symbol, and when we want to retrieve information about a specific exemplar. A strictly specified set of abstract symbols will be brittle, unable to deal with exceptions and change over time.

Another strategy would be to keep all the information associated with each particular exemplar. In our example above, we might create the hyperedge, e1:{four legs, short ears, long tail, furry, black}, thus preserving co-occurrence information. Then we are left with the problem of how to generalize the information, as well as trouble explaining how it is that people forget.

Our approach does not abstract away from the input; nor do we keep all the information as hyperedges. We posit associational mechanisms which allow us to record co-occurrence information while building up prototypical concepts. Specific exemplars are

not retained per se, but information about specific exemplars may sometimes be retrieved through context and attention. Instead of looking up an old exemplar in memory, we simulate it by activating the same (or almost the same) set of nodes.

Our associative memory model demonstrates the effect of context on recognition (Barsalou, 1989), and provides an elegant explanation of the role of the unusual in retrieving memory of specific exemplars (what Schank, 1982/1990 terms failure-driven memory). The model supports both intuitive, sense-based concept recognition and analytic, symbol-based reasoning, providing insights into expert-novice differences. The model has interesting implications for critical learning periods, base-rate neglect in categorization (Kruschke 1992), and the issue of unidirectionality of association.

### 2.2 Activation and Association

The basis of the DCOG associative memory is the **backcloth**: a set of nodes connected by association links of varying strengths. In the framework of Q-analysis (Atkin, 1974), each node might represent a person, with the associations representing relationships between people. Alternatively, each node might represent a personal quality, with a cluster of connected nodes representing a particular person. Following Young (1998), we have adapted this system so that a cluster of connected nodes represents a mental concept.

Each node represents a feature that may be perceived, such as color, shape, etc. When a feature is perceived, the corresponding node is activated. Each node has a **firing threshold**, so that repeated input may be required for the node to become active. When a node becomes active, it fires, spreading activation energy to other nodes, in proportion to

the connection strength of the links between the nodes. Such **spreading activation** may in turn cause other nodes to fire. When two nodes are **co-active**, the connection between them is strengthened by a process of **association**. (See also Section 2.8., below, on **synchronization**.) Over time, the activation levels **decay**, so that a node representing an input which is not reinforced by either spreading activation or continued perceptual activation will drop below firing threshold and become inactive. Activation is thus a temporary phenomenon, which leaves traces in the form of associations between nodes.

### 2.3 Recognition

When an object is perceived, various features are perceived. These features activate the corresponding nodes. Some nodes go over threshold and fire. Connection strengths between the active nodes are strengthened. When a similar object is subsequently perceived, many of the same nodes become active, further strengthening connections among the typical features of this type of object. Although the exemplars of the concept have individual differences, over time the associational memory builds up a core of strongly connected nodes which represent the concept. We term this **emergent concept formation**. The concept core can be defined mathematically and identified by an algorithm (see Section 2.3 below).

Because concepts are built up from fuzzy sets of partially overlapping exemplars, they are free to evolve. Such **concept drift** allows the concept to continue to be the most useful representation for current circumstances. For example, my concept of cat will probably include as core features things like: {meows, furry, long tail, four legs}. If most of the cats I have encountered have been black cats, the feature {black} may also be

included in the core of strongly connected nodes. Suppose, however, that I begin to see more and more orange cats. Eventually, the feature {orange} will supplant the feature {black} among the core of strongly connected nodes.

Emergent concept formation and concept drift provide useful characterizations of overgeneralization, a typical stage in the development of category names in children's speech. A child's initial experience of a horse might include these features: {big, four legs, farm, hooves, long neck, the word "horse"}. There will be substantial overlap between the "horse" features and the features present when perceiving a cow, for example, {big, four legs, farm, hooves, black, short neck}. At this point, emergent concept formation may create one concept that includes the key features of both horses and cows. When attempting to associate a word with a cow, the child finds the word "horse" associated with the currently active features for the cow. So, cows, horses, and perhaps even goats come to be named "horse." Over time, the child sees more exemplars of both horses and cows, and the two separate concepts begin to emerge in the connection strengths.

We define **concept recognition** as the synchronous activation of the core features of a concept. (That is, all the nodes in the concept's core are activated above threshold and firing in synchrony.) Concept recognition may be the result of direct activation from perceptual input, or it may depend on both direct and indirect (spreading) activation. Initial activation of a subset of the core features may suffice for recognition, since activation will then spread among the strongly connected nodes.

For example, assume that the core of the {cat} concept is the set of features, {furry, four legs, long tail, meows, small pointed ears, graceful}. When I see a cat from my

window, I may not hear the cat meowing, but the {meow} node will be indirectly activated by my perception of the other features of the cat. The structure of the associational memory leads me to expect that the cat will meow. Similarly, we mentally reconstruct an object that is partially obscured from view, so long as we can see enough of the object to spread activation to the missing features.

We describe both emergent concept formation and concept drift mathematically in terms of an algorithm of **latent concept identification**. The algorithm identifies areas of strong structural association, regardless of current activation. We restrict our attention here to immediate links; although the algorithm can be generalized to include chains of links at greater distances (see Appendix A). Note that, while the algorithm uses a sequential process to discover the conceptual structures, the actual mental operations are state-change operations, and not dependent on sequential processing.

Latent Concept Identification Algorithm (1)

Take a set of nodes, A. Select a strength parameter, p.

For each node $n_i \in A$, find the set of nodes which share a link with $n_i$, with the connection strength of the link $\geq p$. Call this set $Rp(n_i)$, the p-range of $n_i$.

Start with the $n_i \in A$ which has the largest number of elements in its p-range. Call it $n_1$.

(If there are two $n_i$ with the same number of elements in their p-ranges, arbitrarily choose one to consider first.)

Let $T_1 = Rp(n_1) \cup n_1$ (this is the start of the first concept)

Take the next smallest p-range, call its source node $n_2$.

If $T_1 \cap Rp(n_2) = \varnothing$, then let $T_2 = Rp(n2) \cup n_2$ (this creates a second concept)

else let $T_1 = T_1 \cup Rp(n_2) \cup n_2$. (this adds the new nodes to the first concept)

Continue checking the p-range of each $n_i \in A$. The $T_i$ are the distinct latent concepts identified at strength p.

Here is an example of latent concept identification. Take the set of nodes, {a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x}, with the connection strengths indicated by the following adjacency matrix:

**Table 1: Adjacency matrix for latent concept identification example**

| | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | | | 3 | | | | | | | | | | | | | | | | | | | | | |
| b | | | 2 | 1 | | | | | | | | | | | | | | | | | | | | |
| c | 3 | 2 | | | | | | | | 1 | | | | 1 | 3 | | | | | | | | | |
| d | | 1 | | | 3 | | | | | | | | | | 1 | | | | | | | | | |
| e | | | | 3 | | 2 | | | | | | | | | | | | | | | | | | |
| f | | | | | 2 | | 1 | | | | 1 | | | | | | | | | | | | | |
| g | | | | | | 1 | | | | | | 1 | | | | | | | | | | | | |
| h | | | | | | | | | 1 | | | 2 | | | | | 3 | | | | | | | |
| i | | | | | | | | 1 | | | | | 3 | | | | | | | | | | | |
| j | | | 1 | | | | | | | | | | | 2 | | | | | | | | | | |
| k | | | | | | 1 | | | | | | | | | | 1 | | 2 | | | | | | |
| l | | | | | | | 1 | 2 | | | | | | | | | 2 | | 3 | | | 3 | | |
| m | | | | | | | | | 3 | | | | | | | | 2 | | | | | | 1 | |
| n | | | 1 | | | | | | | 2 | | | | | 2 | | | | | | | | | |
| o | | | 3 | 1 | | | | | | | | | | 2 | | | | 3 | | 3 | | | | |
| p | | | | | | | | | | | 1 | | | | | | | | | | 1 | | | |
| q | | | | | | | | 3 | | | | 2 | 2 | | | | | | | | | | | |
| r | | | | | | | | | | | 2 | | | | 3 | | | | | | | | | |
| s | | | | | | | | | | | | 3 | | | | | | | | | 1 | 1 | | |
| t | | | | | | | | | | | | | | | 3 | | | | | | | | | |
| u | | | | | | | | | | | | | | | | 1 | | | 1 | | | | | 1 |
| v | | | | | | | | | | | | 3 | | | | | | | 1 | | | | 1 | |
| w | | | | | | | | | | | | | 1 | | | | | | | | | 1 | | |
| x | | | | | | | | | | | | | | | | | | | | | 1 | | | |

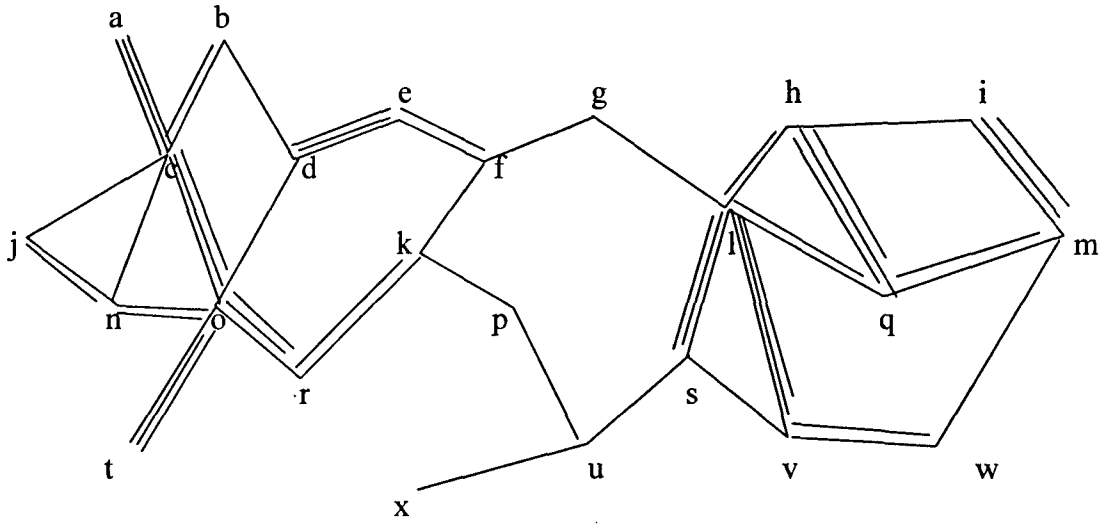



**Figure 1: Association diagram for latent concept identification example**

Consider first the concepts identified with strength $p \geq 3$. The p-ranges of each node are given below:

**Table 2: 3-ranges for latent concept identification example**

| node | $L_3$(node) | node | $L_3$(node) | node | $L_3$(node) |
|---|---|---|---|---|---|
| a | c | i | m | q | h |
| b | -- | j | -- | r | o |
| c | a, o | k | -- | s | l |
| d | e | l | s,v | t | o |
| e | d | m | i | u | -- |
| f | -- | n | -- | v | l |
| g | -- | o | c, r, t | w | -- |
| h | q | p | -- | x | -- |

The 3-range with the largest number of elements is $L_3(o)$: {c, r, t}. We set T1= {c, r, t} U {o}. The next largest 3-ranges are $L_3(c)$: {a, o} and $L_3(l)$: {s, v}. We start arbitrarily with c. The 3-range of c intersects with $T_1$, since they both contain the element, o. So $T_1$ becomes {c, r, t, o} U {c, a, o} = {c, r, t, o, a}. The 3-range of l does not intersect with $T_1$, so we create $T_2$ = {s, v} U {l}. The remaining nodes with non-empty 3-ranges all have one-element 3-ranges. We take them in arbitrary order, identifying 3 additional concepts. The results are summarized below:

**Table 3: Concepts identified by algorithm at p≥1**

| node | range | concepts | | | | |
|---|---|---|---|---|---|---|
| o | c,r,t | T1 = c,r,t,o | | | | |
| c | a,o | T1 = c,r,t,o,a | | | | |
| l | s,v | T1 = c,r,t,o,a | T2 = s,v,l | | | |
| a | c | T1 = c,r,t,o,a | T2 = s,v,l | [no change] | | |
| d | e | T1 = c,r,t,o,a | T2 = s,v,l | T3 = e,d | | |
| e | d | T1 = c,r,t,o,a | T2 = s,v,l | T3 = e,d | [no change] | |
| h | q | T1 = c,r,t,o,a | T2 = s,v,l | T3 = e,d | T4 = h,q | |
| i | m | T1 = c,r,t,o,a | T2 = s,v,l | T3 = e,d | T4 = h,q | T5 = i,m |

[no further changes for nodes m, q, r, s, t, v]

At strength p≥1, we thus identify five distinct concepts. At strength p≥2, we find only three concepts:

**Table 4: 2-ranges for latent concept identification example**

| node | $L_2$(node) | node | $L_2$(node) | node | $L_2$(node) |
|---|---|---|---|---|---|
| a | c | | | | |
| b | c | | | | |
| c | a, b, o | | | | |
| d | e | | | | |
| e | d, f | | | | |
| f | e | | | | |
| g | -- | | | | |
| h | l, q | | | | |
| i | m | | | | |
| j | n | | | | |
| k | r | | | | |
| l | h, q, s, v | | | | |
| m | i, q | | | | |
| n | j, o | | | | |
| o | c, n, r, t | | | | |
| p | -- | | | | |
| q | h, l, m | | | | |
| r | k, o | | | | |
| s | l | | | | |
| t | o | | | | |
| u | -- | | | | |
| v | l | | | | |
| w | -- | | | | |
| x | -- | | | | |

**Table 5: Concepts identified by algorithm at p≥2**

| node | range | concepts | | |
|---|---|---|---|---|
| o | c,n,r,t | T1= c,n,r,t,o | | |
| l | h,q,s,v | T1= c,n,r,t,o | T2=h,q,s,v,l | |
| c | a,b,o | T1= c,n,r,t,o,a,b | T2=h,q,s,v,l | |
| q | h,l,m | T1= c,n,r,t,o,a,b | T2=h,q,s,v,l,m | |
| e | d,f | T1= c,n,r,t,o,a,b | T2=h,q,s,v,l,m | T3=d,f,e |
| h | l,q | T1= c,n,r,t,o,a,b | T2=h,q,s,v,l,m | T3=d,f,e [no change] |
| m | i,q | T1= c,n,r,t,o,a,b | T2=h,q,s,v,l,m,i | T3=d,f,e |
| n | j,o | T1= c,n,r,t,o,a,b,j | T2=h,q,s,v,l,m,i | T3=d,f,e |
| r | k,o | T1= c,n,r,t,o,a,b,j,k | T2=h,q,s,v,l,m ,i | T3=d,f,e |

[no changes for remaining nodes]

The Ti identifies the **core**, or tightly connected nodes, of each latent concept. Other nodes will also be associated with various exemplars of a concept, but the core represents the greatest area of overlap among exemplars.

Once the latent concepts are identified, we can compare the active nodes to the latent concepts to effect **concept recognition**. Concept recognition is described mathematically with the definition of the term **example**:

Example: (2)

An exemplar, X, is an example of a concept, C, if the perception of X activates the core of C, either directly or by spreading activation, within some limit of time or distance.[1] When perception of X activates the core of C, we say that X is recognized as an example of the concept, C.

The example relationship between X and C is not static or permanent; rather, whether X is perceived as an example of C depends both on the structure of C and on the current activation of the nodes in C and in the backcloth. The structure of a concept can enable recognition if the core nodes have low firing thresholds, if there are many links to and among the core nodes, and if those links are of high connection strength. Conversely, a concept with a loosely connected core, or a concept with high firing thresholds in its core nodes, will be more difficult to recognize. Recognition is also subject to having sufficient activation, either in the core nodes themselves, or in neighboring nodes which can spread activation to the core. These limitations on recognition explain why different individuals perceive the same input differently, and why one individual may recognize input differently at different times.

We return to the definition of example in our discussion of dimensions in Section 2.5, below. We turn now to an examination of how context can influence recognition.

[1] We must set some time or distance limit, to distinguish between (a) recognition which occurs after one cycle of spreading activation, and (b) activation of a concept which is the result of continued spreading activation and possibly other perceptual input over a longer length of time (what we call a train of thought).

## 2.4 Context

### 2.4.1 Direct and Indirect Activation

When the core nodes of a concept are included in a perceptual exemplar, that input directly activates the concept, and the exemplar is recognized as an example of the concept (**direct activation**). If the exemplar fails to directly activate the complete set of core nodes, the exemplar may still be recognized as an example of the concept through **indirect activation**.

Indirect activation may come from perception of non-core features of the input, or from the context of previous perception and thinking. A set of context nodes may work together to provide sufficient activation to activate a target node. Barsalou (1989) notes the role of context, distinguishing three kinds of information used in thinking:

1. Context Independent information (CI)
2. Context Dependent information (CD)
3. Recent Context Dependent information (CD rec)

In our model, the context independent information is represented by the core nodes of a concept; the context dependent information refers to the non-core nodes; and the recent context dependent information refers to the active nodes in the backcloth. It is the combination of these elements that leads to recognition.

A shift in **attention** may, by creating a new context, trigger the recognition of a concept. A detailed example of the use of attention in reasoning is given in I.E., below.

### 2.4.2 Access Nodes, Typicality, and Specificity

How can we tell when a feature will activate a concept? We have stated above that the activation of a concept may depend on the combined effects of a set of nodes; however,

it is still possible to examine the usefulness of a particular node in activating a concept. A node that is useful in activating a concept is called an **access node** for that concept.

A good access node has two qualities: **typicality** and **specificity**. A node is typical of a concept if most of the exemplars of that concept contain that node. This relationship will be reflected in strong connection strengths between the nodes in the core of the concept and the proposed access node; in fact, a typical node is likely to be included in the core of a concept. A node is specific to a concept if activating that node tends to activate the target concept more than any other concept in some set of potential concepts. The set of concepts may include the entire backcloth, or it may be limited by context to a smaller set.

Nicknames provide an example of good access nodes: A nickname uses some quality that is typical of and specific to a person in order to identify that person. "Hotshot" would not be a good nickname for a modest person, since it invokes a quality that is not typical of that person. "Two-eyes" would not be a good nickname in general, since it describes a quality that is not specific to any one person.

A node may be typical without being specific. For example, blue is typical of blue jay, but not specific to blue jay, since there are blue jeans, blue sky, etc. If context restricts the set of potential concepts to types of birds, then blue becomes more specific for blue jay. Conversely, a node may be specific to a concept without being typical. Consider the word, runcible. A runcible spoon is one with notches in the bowl, allowing it to function as a fork. While runcible is highly specific for spoon, it is not typical of spoons.

So far, we have illustrated typical and specific features for a concept; in our formal definitions, below, we define the relationships of specificity and typicality between two features, and then extend those relationships to sets of features (i.e., concepts).

A feature X is **typical** of a feature Y if activating Y tends to activate X within a set of competing features. We write the relationship as, $Typ_J$ (X, Y), where J represents the set of features which includes X. This usually reflects a stronger connection strength between Y and X than between Y and the other features in the set. Context (concurrent activation of other, related feature nodes) can also influence the outcome, so we must say "tends to activate" X. See Section 2.4, on the role of context.

A feature X is typical of a set of features Y:{y1, ... yn} if activating the set Y tends to activate X within a set of competing features. The tendency to activate the typical feature may be due to connection strength or to the number of connections from the members of Y to X.

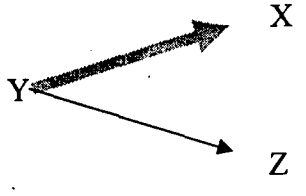



**Figure 2: Feature X is typical of feature Y**

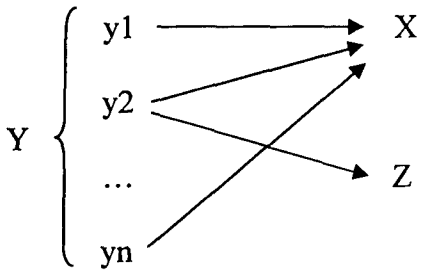



**Figure 3: Feature X is typical of the concept Y. Given the input Y: {y1, y2, ... yn}, X is preferentially activated over Z in the set {X, Z}**

A feature X is **specific** to a feature W if activating X tends to activate W within a set of competing features. We write, $Spec_J$ (X, W), where J represents the set of features which includes W. A feature X is specific to a set of features W:{w1, ...wn} if activating X tends to activate the entire set W within a set of competing sets of features. Again, the preference for W could be due to strength of connection or to number of connections.

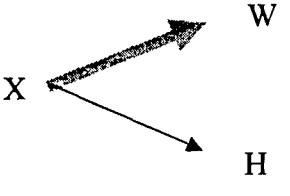



**Figure 4: Feature X is specific to feature W**

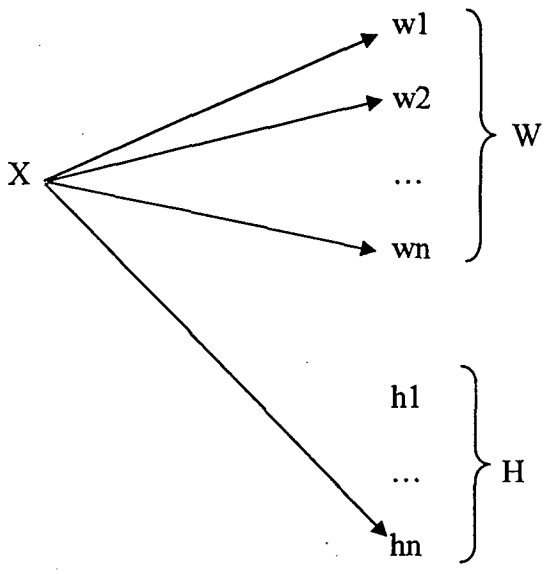



**Figure 5: Feature X is specific to concept W. Given the set of concepts, {W, H}, the feature X tends to activate W**

We must say, "tends to activate" the target node, because actual activation and recognition of the target depends on context, where by context we mean the existing activation of related nodes. A node which, by itself, provides poor access to a concept may be a good access node for that concept when sufficient contextual information is present.

Specificity and typicality are related functions. Define a set of features, J, such that Q is an element of J. If the feature, P is specific to the feature Q within J, then Q within J is typical of P.

Specificity and Typicality (3)

$Spec_J (P, Q) \longleftrightarrow Typ_J (Q, P)$

By limiting our consideration to the activation of Q compared to the activation of the other members of the set J, we are indicating the scope of our attention. This can be modeled by spreading activation or assuming existing levels of activation for the set J as a whole.

2.4.3 Other Treatments of Context

Our characterization of specificity parallels the definition of cue validity as discussed by Rosch (1978), "... the validity of a given cue x as a predictor of a given category y ... increases as the frequency with which cue x is associated with category y increases and decreases as the frequency with which cue x is associated with categories other than y increases." Rosch goes on to define cue validity for an entire category, creating a measure of cohesiveness and distinctiveness that is useful in establishing a basic level of categorization.

Barsalou (1989) also identifies characteristics similar to our notions of typicality and specificity. He states that high levels of accessibility increase concept stability (e.g., memorizing a frequently used phone number); this is reflected in our use of typicality. Barsalou also notes that highly specific cues focus on only a small amount of information, reducing variability in what is retrieved; this, of course, relates to our use of specificity. See also Young (1998) for a discussion of the role of contextual cues from path and place in determining recognition.

2.4.4 Base-Rate Neglect

Consider again the word runcible, which is specific for spoon, but not typical of spoon. If we consider the set {runcible, plastic} and the set {spoon, fork}, we can diagram this relationship as shown below (numbers are arbitrary):

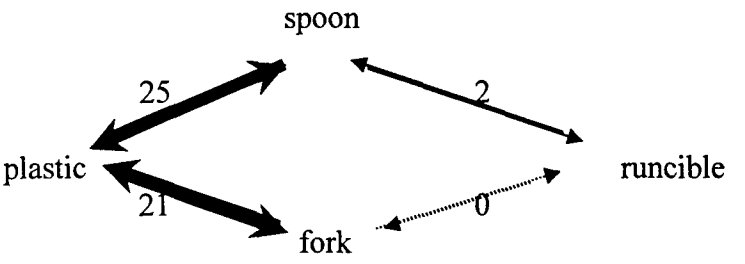



**Figure 6: Feature runcible is specific for feature spoon, but not typical of feature spoon**

Plastic will suggest either fork or spoon; spoon will strongly suggest plastic over runcible; but runcible will only suggest spoon, not fork.

This leads us to a discussion of base-rate neglect (Kruschke, 1992). Because runcible and plastic occur at different base rates in the exemplars we see for spoon, we build up stronger associations for plastic, the feature with the higher base rate. Starting at spoon and spreading activation, we spread considerably more activation to plastic than to runcible. However, if we consider the spreading of activation from the features plastic and runcible, we may be more successful in accessing spoon from runcible, despite the lower connection strength.

Kruschke (1992) recounts an experiment conducted by Gluck and Bower, in which subjects were asked to diagnose hypothetical illnesses based on various symptoms. When the illnesses differed in base rate, a situation was created in which a symptom led equally to both the rare and the common disease, yet the subjects favored the diagnosis of the rare disease.

Here is a diagram of the situation. The rare disease occured 25 % of the time, while the common disease occurred in 75% of the exemplars. The probability of symptom s1 with the rare disease was 60%, compared to 40% for its complement, symptom s1*. The

probability of symptom s1 with the common disease was 20%, compared to 80% for symptom s1*. If we assume a database of 400 exemplars, we can extrapolate the following connection strengths, with instances of each feature given in parentheses:

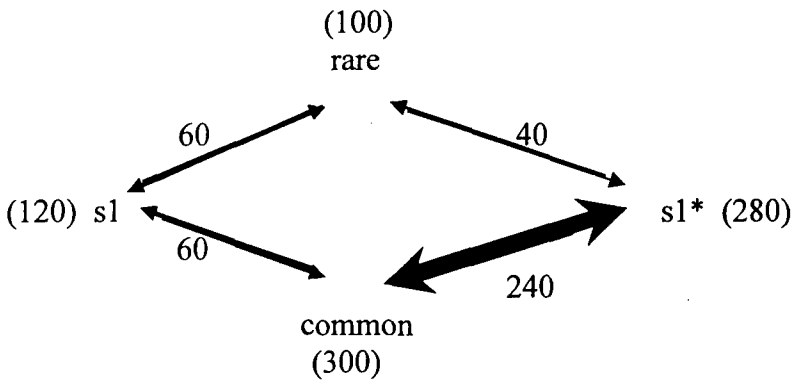



**Figure 7: Actual Base Rates**

Gluck and Bower term the subject's tendency to diagnose the rare disease on the basis of s1, base-rate neglect: If the base rates of the two diseases were equal, then the proportions of s1 and s1* would lead to the diagnosis of the rare disease. This can be seen in the diagram below.

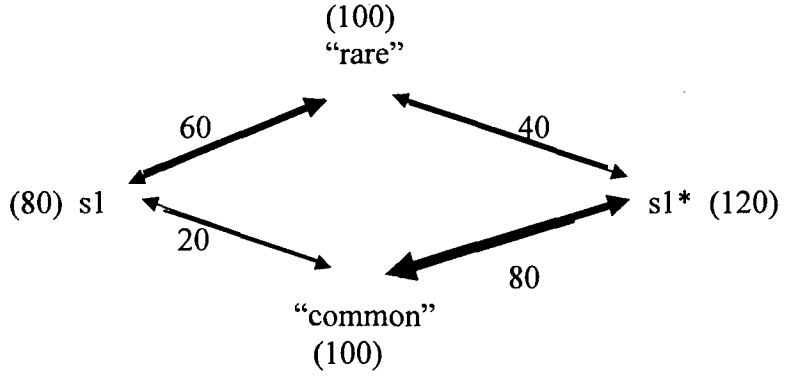



**Figure 8: False Base Rates**

Returning to Figure 7, we can explain base-rate neglect in terms of typicality and specificity. Symptom s1, while not specific to the rare disease, is still more typical of the rare disease than is symptom s1*. We believe that base-rate neglect reflects a shift in attention. Instead of simply spreading activation from the symptom, s1, the subjects must have also been supplying activation to the diseases, creating expectations to the effect, "if it's the rare disease, I should see symptom s1," and, "if it's the common disease, I should see symptom s1*."

Our analysis of base rate neglect has interesting implications for the issue of directionality of association. Many models of associational memory assume that links between features must be directional, under the assumption that a bidirectional association link would force associations to be reciprocal: if word a triggers word b, suggesting a strong association between a and b, then word b should equally trigger word a. Apparent counter-examples include antonyms such as black and white, which operate asymmetrically. Our analysis above suggests that such asymmetries in spreading activation can be built on bidirectional associations, with differences in base rate or context explaining the varying results.

2.4.5 Failure-Driven Memory

Schank (1982/1990) emphasizes the role of expectation failure in the memory of events. He notes that something which corresponds well to expectations is easily forgotten, while deviations from expectations are easier to retrieve. Schank encodes overlapping exemplars as scripts; he attributes failure-driven memory to special mechanisms that, upon a failure of expectations, create a new structure, indexed to the point of deviation from the script.

Our definition of access nodes explains this dichotomy without explicit encoding of expectation failures. Events which correspond to expectations build up stronger connections for a concept. Unusual events create new concepts with weaker connection strengths (because they occur infrequently), but better access nodes (because they are more specific to that concept). Thus, the details of common events are easily forgotten, while the details of unusual events are easily accessed, using one of the unusual features as an access node.

Here is an example: Once a waitress, reaching behind her ear for her pencil to take my order, accidentally flipped the pencil into my drink. That unusual occurrence provides specific access nodes from which to jog my memory of the rest of that experience. I can remember where I was sitting that day, for example, because the pencil incident connects only to that table, in that restaurant. No previous or subsequent associations overlap with that connection.

If, over time, the exception becomes the rule (all waiters begin flipping pencils into drinks), it will become more difficult to recall specific instances and details. For Schank's system, such changes require governing mechanisms that monitor exceptions and decide when an exception warrants development of a new script. In our system, the association and concept recognition mechanisms suffice: Unusual events create specific access nodes; subsequent, similar events obscure the initially unusual event by reducing the specificity of those access nodes.

### 2.4.6 Symbols

Another way to provide a specific access node for a concept is to give the concept a label. Concepts may be labeled with verbal or non-verbal symbols. Words and other

symbols tend to be good access nodes, because their arbitrary nature allows them to be specific to a particular concept. Symbols provide us a way to manipulate our otherwise intractable fuzzy concepts. Note that the symbol does not represent the concept in a hierarchical sense; the symbol is merely one of the many nodes that make up the core of the concept. See Section 2.6.3 below, for more information on the creation and use of symbols.

2.4.7 Layers

Sometimes, activating node a leads us to spread activation to node c, while at other times, activating node a leads us to spread activation to node b. That is, at some times we use a concept a—c, and at other times we use a concept, a—b. We term this **differential association**. How can this be, if there is one stable association strength for a—c and one stable association strength for a—b? In general, how can two or more concepts share the same set of nodes? Why don't the connections all blur together?

In addressing this issue of concept blur, it is tempting to create separate layers for the distinct concepts. In a layer analysis, each concept would use the same nodes, but the associations that make up that concept would exist on an independent mental plane. Another way to express this idea is to create different kinds of association links, so that a node might be linked to one target with one type of link, and simultaneously linked to a different target with a different type of link.

It is not clear, however, what the layers (or multiple association types) would correspond to in the brain. We will work through an example of a layer analysis here, and then show how the effects of layers can be simulated in our model with existing context mechanisms. We return to the issue of layers in Section 3.2 below.

In the example below, the two layers share nodes, but the associations exist only within each layer. We represent this idea here with two planes for the two layers. They are drawn separately, but keep in mind that the nodes are actually unitary objects.

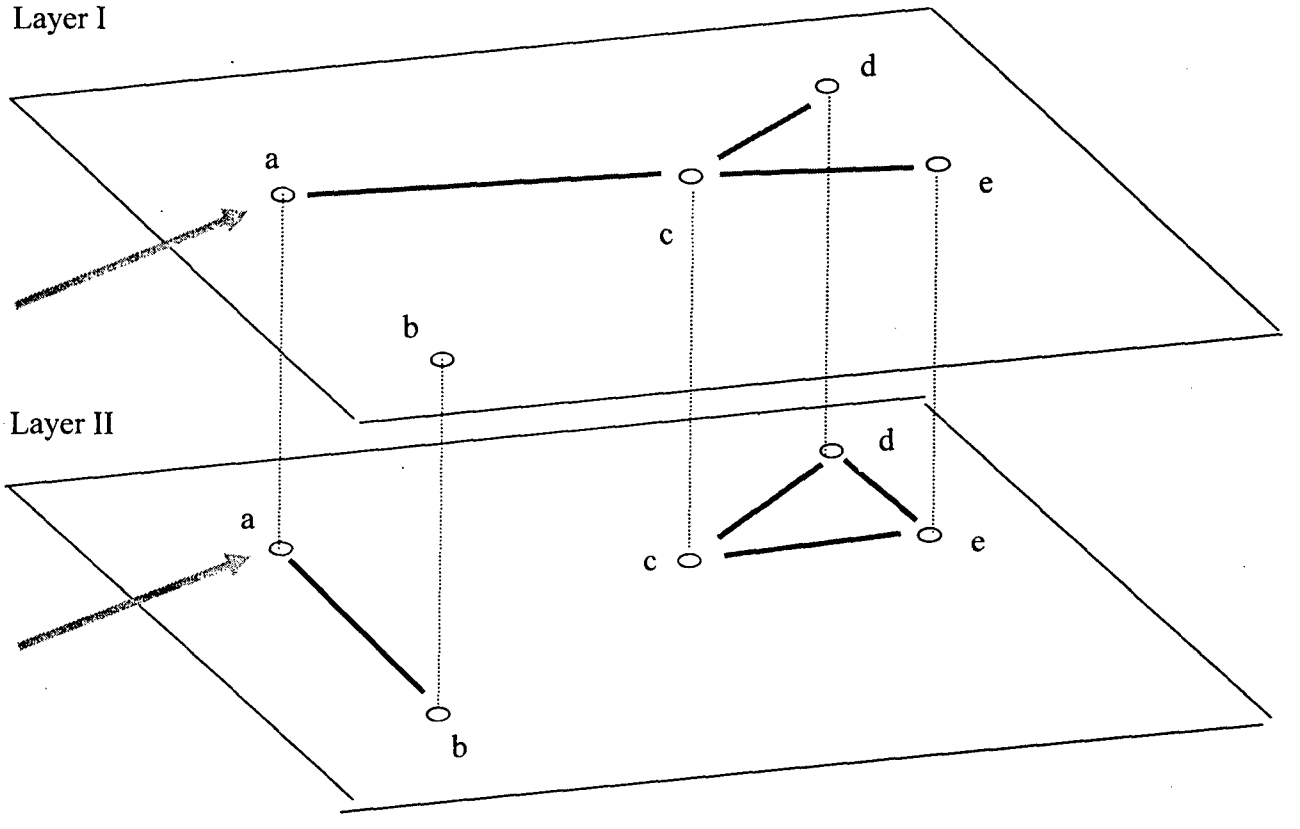



**Figure 9: Two-layer model of differential association**

In Layer I, node a is connected to node c, which is in turn connected to nodes d and e. There is no path connecting nodes a and b. In Layer II, node a is only connected to b. Assume input to node a (green). If we work in Layer II, we spread activation from node a to node b. We cannot get to nodes c, d, or e. If we shift to Layer I, however, we can spread activation from node a to node c, and then to nodes d and e.

Activation is a property of the nodes, and is not directly affected by shifting layers. Shifting layers may, however, enable new associations. Activation "left over" from working in one layer may be strong enough to yield new associations with active nodes in the current layer. In the scenario outlined above, we end up with activation of nodes a, b, c, d, and e. Coactivation of nodes a, b, and c would allow us to form new associations between a-b and b-c. This pattern of coactivation would not be possible from working with either layer independently.

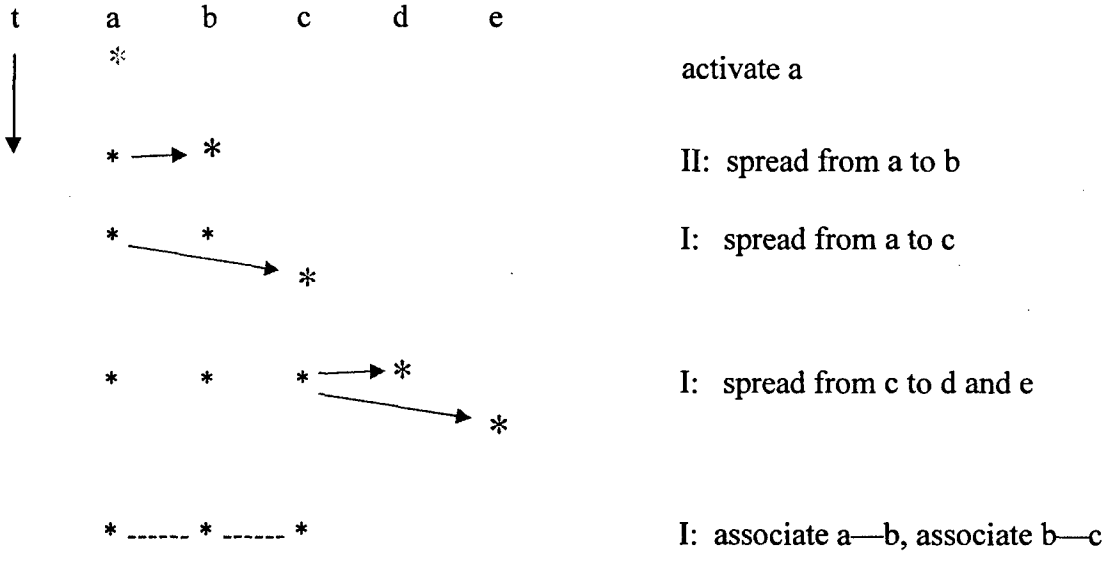



**Figure 10: Timeline for 2-layer model of differential association. (*= active node)**

Now consider this same example, on one plane, with an enriched context. We add nodes x, y, m, and n. Nodes x and y connect to node c; nodes m and n connect to node b.

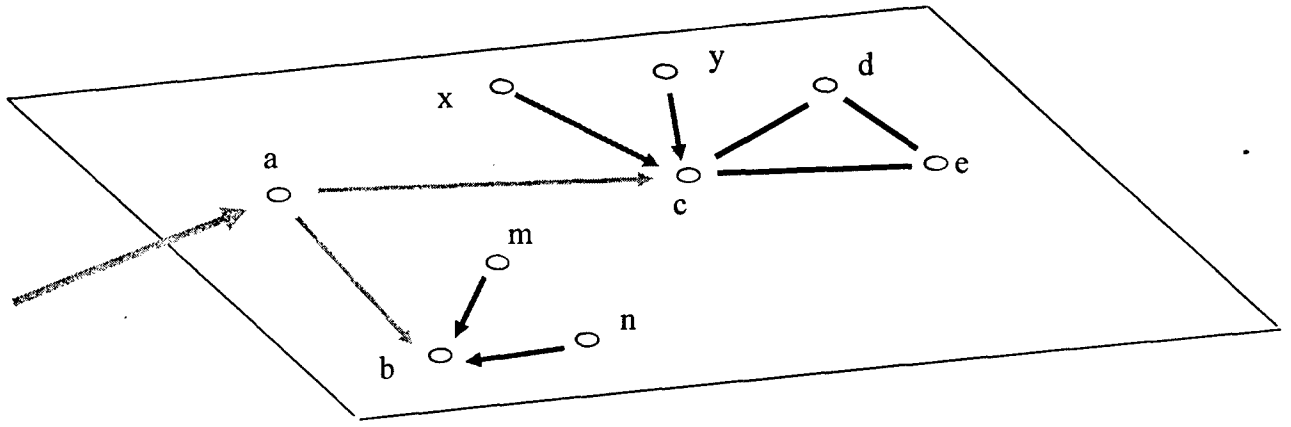



**Figure 11: Context model of differential association.**

Input to a node does not occur in isolation, but against a background of currently active nodes (and in concurrence with other new activation). Suppose the impedances between a—c and a—b are similar. Assume also that b and c each have relatively high activation thresholds. Activation energy to node a spreads to both b and c, but is not sufficient in itself to push either one over threshold.

If the context includes activation energy coming from x and y, then c will fire. If the context includes activation energy coming from m and n, then b will fire.
In effect, different contexts simulate different associational structures: activation of x and y simulates a lowered impedance from a to c; activation of m and n simulates a lowered impedance from a to b. Shifting between layers, as described above, is actually the result of shifting attention to other contextual features.

Layers, simulated by context, give us one way to describe what we perceive as changes in the level of abstraction. Our system also contains the constructs **type** and **dimension** to describe different levels of abstraction. We turn now to a discussion of these concepts.

## 2.5 Dimensions

### 2.5.1 Example, Type, and Dimension

Recall that an exemplar, X, is said to be an **example** of a concept, C, if the perception of X activates the core of C, either directly or by spreading activation. The example relationship is the mechanism that governs recognition.

Based on the definition of example, we can further define a **type**:

Type (4)

A concept, T, is a type of a concept, C, if
there is a non-empty set of exemplars X: {x1, x2, ... xn} such that

1. each xi is an example of C
2. each xi is an example of T
3. T is an example of C

We can represent the type relationship with a diagram, as shown in Figure 12.

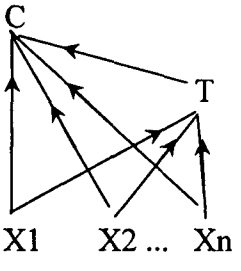



**Figure 12: T is a type of the concept, C**

Note. That this is not an association diagram: the nodes in this diagram represent concepts, and the arrows represent the example relationship

For example, if fruit tree is an example of tree, and we have the set X: {apple tree, cherry tree, pear tree}, in which each $x_i$ is an example of tree and an example of fruit tree, then fruit tree is a type of tree.

Based on the definition of type, we proceed directly to a definition of **dimension**:

Dimension (5)

A set of types T: {t1, t2, ... tn} define a well-formed dimension on a concept, C, if the set of types partitions the set of examples of C. That is, for each xi which is an example of C, xi is an example of one and only one type in the set of types.
(We must exclude here the types themselves from the set of xi.)

For example, the two types evergreen tree and deciduous tree form a well-formed dimension on the set of trees, but the two types deciduous tree and fruit tree do not, since they do not properly partition the set of trees (apple tree is both an example of deciduous tree and an example of fruit tree; pine tree is not an example of either fruit tree or deciduous tree).

Dimensions can be induced mathematically from a set of exemplars; we defer discussion of this technique until Section 4.6, where we apply it to the air traffic control task.

2.5.2 Reasoning with Dimensions

A dimension is typically associated with a feature in the verbal/symbolic domain which characterizes or labels that dimension. (See Section 2.6.3 for more on the symbolic domain.) We demonstrate the usefulness of dimensions in illustrating the task of considering a set of objects and answering the question, "Which one doesn't belong?"

Suppose we are given a set of items and asked to choose one which does not belong. This amounts to identifying a dimension on which one member of the set has feature F, while all other members have feature $f^c$ or the equivalent. There are several ways in which we could reason about this problem.

For our example, consider toy vehicles that children can ride on. This is the category, C. The examples we are given to consider are a bike, a pedal car, a skateboard, and a battery-powered car. These are all examples of C. There are several dimensions (sets of types of C) that we can consider: number of wheels, powered vs. unpowered, etc. In the following diagram, the connecting lines in the verbal domain represent associations. In the non-verbal domain, the dashed lines represent the example relationship, and should be understood as pointing upward. (We omit here the verbal labels for the examples.)

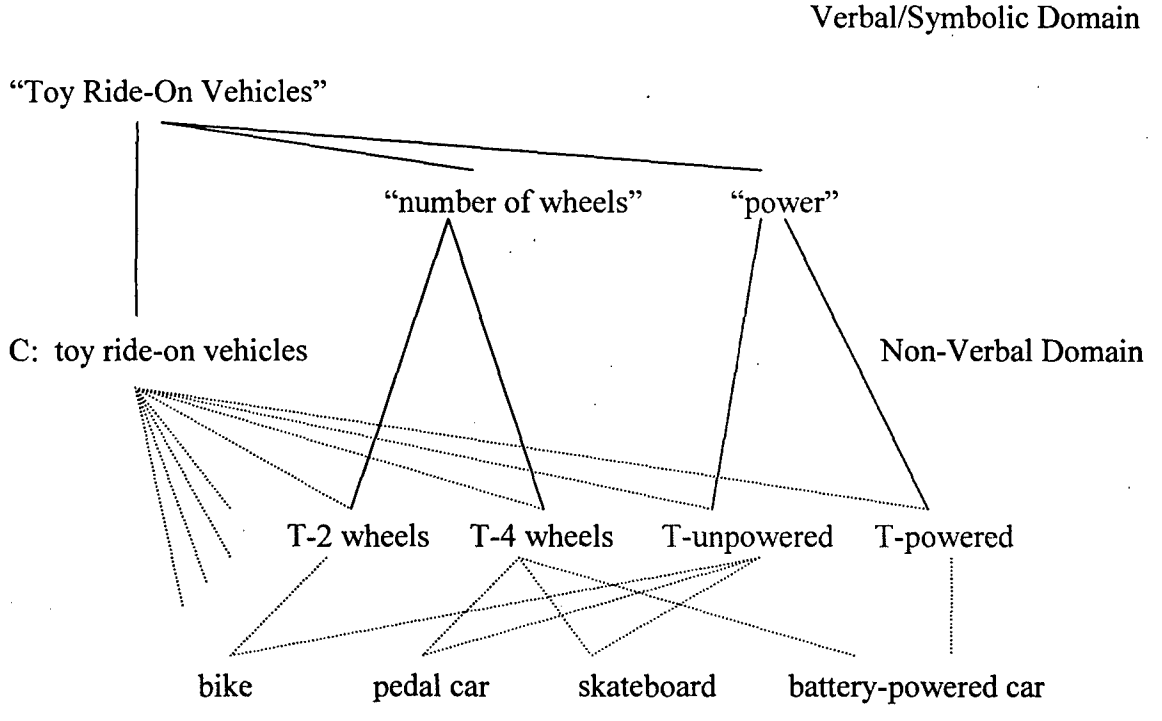



**Figure 13: Two partitionings of the category, {Toy Ride-On Vehicles}**

We can diagram our goal in finding "the one that doesn't belong" as follows. We are attempting to find a dimension D, with a Type Ti, such that Ti dominates only one of the examples Xj in the set X, and the other Type(s) of the dimension D dominate the remaining members of the set X, without dominating Xj.

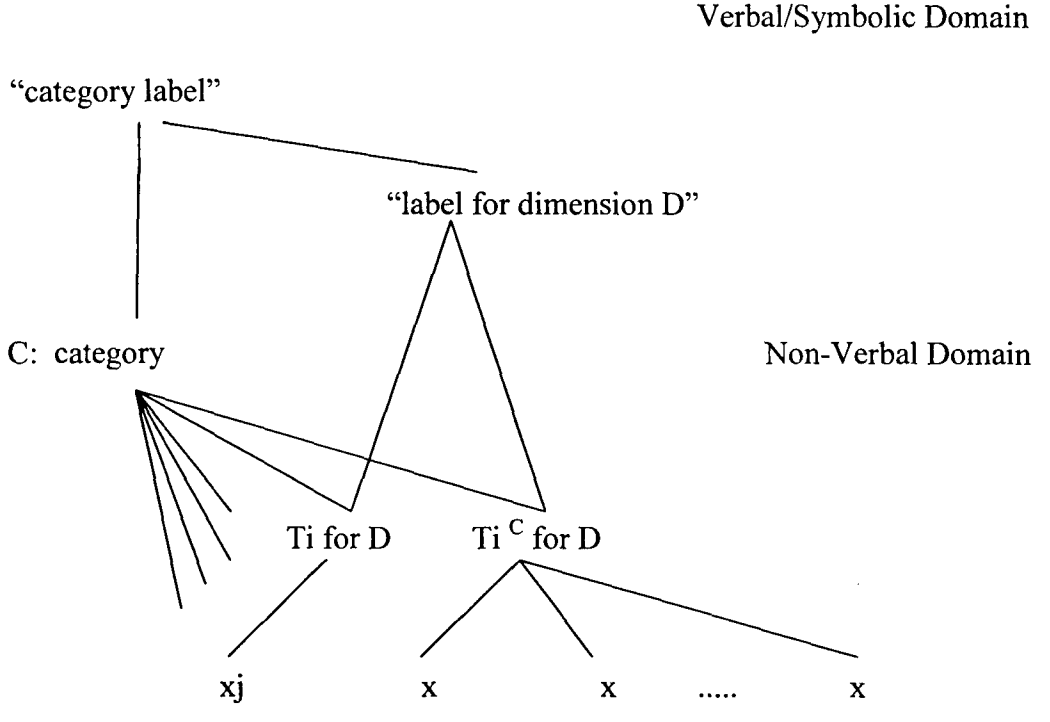



**Figure 14: Dimension D partitions the category and isolates one exemplar**

If we have already established types for this category, then we can start in the verbal domain, where we have access nodes (labels) for the various dimensions and types. We can send activation energy to the access node "number of wheels" to access the types, T-2 wheels and T-4 wheels. By activating these in turn, we activate the sets of examples, {bike} and then {pedal car, skateboard, battery-powered car}. This isolates {bike} as the one that doesn't belong. On the other hand, we can shift our attention by sending energy to

the verbal access node "power", activating the types T-unpowered and T-powered. These, in turn, activate the examples {bike, pedal car, skateboard} and {battery-powered car}. This line of thought isolates the {battery-powered car} as the one that doesn't belong.

We might, instead, start with the examples, evaluate eccentricity by comparing the number of shared features, and then attempt to justify or characterize the most eccentric one as "the one that doesn't belong" by spreading activation from the example up to the types. Say we happen to isolate the bike as the most eccentric. Then we spread activation to the concept, {bike}. This spreads activation to the types associated with {bike}, including T-2 wheels and T-unpowered. When we spread activation to T-2 wheels we recognize the situation we are trying to achieve: T-2 wheels only dominates {bike}, and the other Type in this dimension dominates the other examples.

A third way to reason about these types would be to consider features one at a time and propagate types for these features. We could start this process several ways: consider the features of the most eccentric example of the category; take one example and random and consider its features, then proceed to the features of the next example, etc.; take all the examples together and consider the most unusual features, or the features held most in common. At any rate, we identify a feature for consideration (e.g., having wheels). Propagate a type of C, T-f (here, T-having-wheels). Consider the divisioning of examples created by using T-f and T-$f^{C}$. In this instance, we notice that T-having-wheels is not useful, since all the examples are dominated by T-having-wheels. We continue this process until you find a feature which isolates one Xj.

For example, consider the {bike}. One of its features is {seat}. Propagate a type, T-seat. Notice that T-seat dominates {bike, pedal car, battery-powered car}, but not {skateboard}. This isolates the skateboard as the one that doesn't belong.

Certain automatic, attention-shifting mechanisms will be required in our model. We mention a few in passing here. In the reasoning example illustrated in Figure 13, above, we needed a mechanism to shift attention sequentially through a set of objects. Other candidate attention shifting mechanisms include paying attention when a node exhibits a large, sudden change in activation (whether or not it crosses its firing threshold); paying attention to a latent concept which has been structurally identified, but not related to a symbol; and paying attention to a newly created symbol. We propose below, in Section 4.3.2 that a conflict in feature values within a dimension also draws attention; in this task, attention is then shifted to other dimensions, enabling better hypothesis formation.

Depending on our motivation and current context of activation, we will use different dimensions to categorize concepts. In addition, people are able to make use of ill-formed dimensions. We might expect to find attentional mechanisms which direct us to spread activation to elements which cause a dimension to be ill-formed.

Consider the category, chair, with the types T-swivel chairs, T-rockers, and T-other chairs. This might be a well-formed dimension, partitioning your experience with chairs. Then someone gives you a swivel-rocker. Now your dimension is ill-formed, since this example is an example of both a swivel chair and a rocker. One response would be to classify this chair based on its most salient feature. That is, we pay attention (spread activation) to either its ability to swivel or its ability to rock, and classify it accordingly.

Another response would be to create a new type, T-swivel-rockers. Then we have the dimensioning, T-swivel chairs, T-rockers, T-swivel-rockers, and T-other chairs. The previous types, T-swivel chairs and T-rockers, must now be restricted to "pure" examples, excluding swivel-rockers. In the normal course of events, you might revert to your original classification of chairs. On encountering swivel rockers, you might shift to the second dimensioning.

Dimensions as defined here may provide a framework for examining other semantic relationships. We illustrate this possibility with a brief discussion of predication and opposition.

Predication can be understood in terms of dimensions. "X1 is small" implies the structure given below (arrows represent the example-of relationship): There is some category, C, for which some examples, including X1, are of Type small, and there are other examples which belong to a type which is not small. Note that the category is implied (X is small for a C).

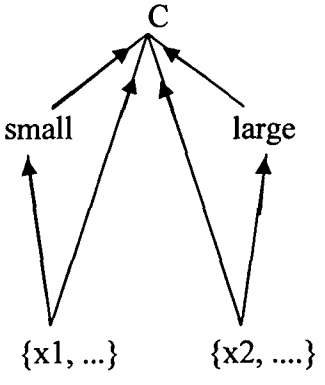



**Figure 15: Type representation of predication using two types**

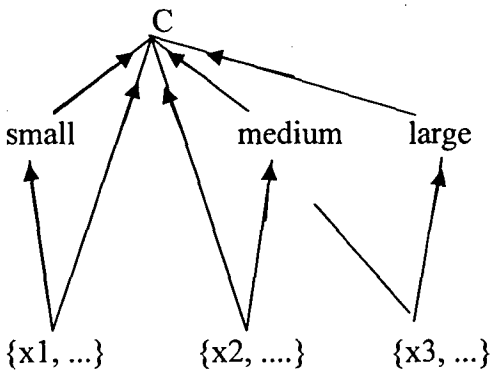



**Figure 16: Type representation of predication using more than two types**

Opposition also implies a dimensioning. "A and B are opposites" can be represented in the following structure (again, lines represent the example-of relationship). A and B are types, forming a dimension, and there are no other values (types) on that dimension.

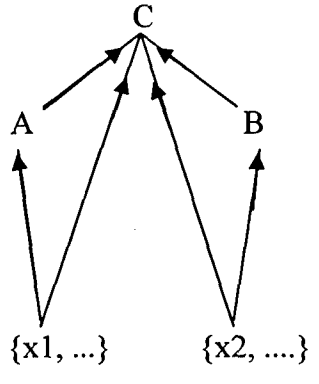



**Figure 17: Type representation of antonymy**

## 2.6 Domains

### 2.6.1 Explicit Concept Formation

We suggest that nodes are grouped into domains, including perceptual, symbolic, and motor domains. Each domain is expected to have unique capabilities and constraints, which influence what it treats as features, concepts, and relations. Nevertheless, concepts can form both across domains and within domains. Consider first symbols in the symbolic domain. Symbols, while related to their respective concepts, can undergo association and other manipulations with other symbols within the symbolic domain. One type of symbolic association is **explicit concept formation**, in which symbols are used to associate two concepts. While emergent concept learning builds up connection strength between nodes through repeated perception of exemplars (i.e., symbolic cognition based on perceptual domain processing), explicit concept learning uses access nodes to link concepts.

Consider a young child's concept of {whale}. It might include the features, {big, swims, tail, fins}. This might be very similar to the child's concept of {fish}. For this child, a whale is a fish. Explicit concept formation takes place when an adult tells the child, "Whales are mammals." Consider how the child must modify his or her concept. We cannot simply add a feature, mammal, to the {whale} concept, for {mammal} is a concept in its own right. Instead, we wish to form a link between the two concepts. Instead of creating some kind of abstract nodes to represent the concepts, we form a link in the symbolic domain between the verbal access nodes, "whale" and "mammal." This causes the concepts to begin firing in synchrony, thus temporarily forming a larger concept (See Section 2.8 on synchronization). Co-activation of {whale} and {mammal} causes the formation or strengthening of connections between specific nodes across the two concepts.

(Note that the child's initial connections between {whale} and {fish} are not lost; they are merely superseded by other, eventually stronger connections. Activation decays, but connection strength persists.)

### 2.6.2 Adult Learning vs. Child Learning

The distinction between symbolic and perceptual domains may explain some differences between adult and child learning. Here we sketch one possible interpretation. Certain nodes are built into the system, anticipating certain features. Nodes are added during infancy and childhood, as required to reflect sensory input. In the sensory domain, node creation ceases after childhood; new sensory input can no longer be encoded with a new sensory node. In the symbolic domain, however, we continue to create nodes throughout adulthood. In dealing with novel sensory input, we make our best match with existing sensory nodes, and then deal with disparities through symbol creation and manipulation. For example, an infant, hearing novel phonemes, creates nodes in the sensory domain to represent each phoneme. An adult, hearing novel phonemes, cannot create new sensory nodes, but must rely on symbolic processing to relate the new sound to sounds which do have sensory node correlates. Explicit concept formation and significant attention may be required to keep the new sound from being recognized as simply a variant of an existing sound. Similarly, in production of sounds acquired as an adult, the speaker must resort to symbolic processing to approximate the correct sound through the modification of sounds learned in childhood.

### 2.6.3 Symbolic Concepts

We assume that some concepts exist entirely within the symbolic domain. The exemplars for these concepts are sets of co-active symbols. While we have not yet fully

explored the implications of semantic concepts, we expect this to be the source of higher-level symbolic relationships. For example, we might find a symbolic concept "gift," which involves, as core nodes, symbols which fill the roles of giver, receiver, and given. When we perceive a situation in such a way that we activate symbolic nodes which are appropriate to these roles, we instantiate an exemplar of this symbolic concept.

### 2.6.4 Expert-Novice Differences

The use of symbols distinguishes three stages in expert-novice reasoning about categorization. Initially, the complete novice merely recognizes, relying on emergent concepts. To go beyond the emergent concepts, the novice creates and manipulates symbols. By shifting attention (spreading activation energy) to different symbols, the novice can create the context necessary for recognition. Such symbolic reasoning increases cognitive workload. Eventually, emergent concept formation provides the expert with the ability to bypass symbolic reasoning and once again simply recognize concepts.

## 2.7 Levels of Abstraction

### 2.7.1 Contextual Shifts

A shift in context may correspond to changing a level of abstraction. Suppose I glance at a table and I see a copy of *The Call of the Wild.* Depending on attention and context, I may recognize concepts at varying levels of abstraction, such as {my copy of *The Call of the Wild*}, {a book}, or {reading material}. For example, in the context of looking for a missing library book, I am more likely to identify the object as a specific title, rather than as just a book. There is no need in our system to represent the relationships between

these concepts hierarchically. The concepts are related because they share certain nodes; they represent different levels of abstraction by virtue of context.

2.7.2 Type Relationship

The type relationship provides us a more structured way to represent abstraction hierarchies. In Figure 12, repeated here as Figure 18, C, T, X1, X2, and Xn are all concepts; the arrows represent the relationship, "a is an example of b." C is at the highest level of abstraction, T is intermediate, and X1, X2, and Xn are at the lowest level of abstraction.

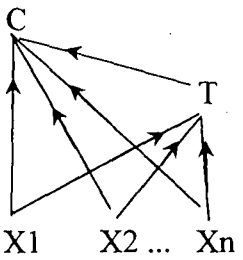



**Figure 18: T is a type of the category, C**

Perceptual input may match core nodes at all three levels of abstraction; context determines which concepts are recognized. Context can be provided by input or by thinking. Suppose an input, Q, is recognized initially as Xn, at the lowest level of abstraction in our diagram, based on the context of other active nodes which spread activation to Xn. By focusing our attention on an appropriate context, we can "see" Q at any of the three levels of abstraction. That is, by activating nodes appropriate to access T, we can recognize Q as an example of T, and by activating nodes appropriate to access C, we can recognize Q as an example of C.

### 2.7.3 Basic Level

Rosch (1978, p. 32) suggests that we can identify a basic level of abstraction, "the most inclusive level of classification at which objects have numbers of attributes in common." Superordinate categories, above the basic level, have too few shared attributes, while subordinate categories, below the basic level, have too much overlap. Rosch points out (p. 42) that context can change the basic level; she gives the example of a man in a furniture store, for whom "chair" is no longer useful as a basic level category.

For us, the basic level would be that level of abstraction at which the exemplars form strongly connected concepts. For the man in the furniture store, context restricts the set of possible concepts to the concept chair, types of chairs and specific chairs. Within this context, recognizing something as a chair does not sufficiently distinguish the exemplars.

## 2.8 Synchronization

We have described the associative memory model in terms of "the set of active nodes," and below we lay out an implementation design that refers to a single set of active nodes. Eventually, the model must be extended to allow for several different sets of active nodes, distinguished by their rate of firing. Timing constraints will allow about seven distinct firing rates, so we might have up to seven different sets of active nodes. We assume that "co-active" means active and firing in synchrony, so that associations, for example, form only between synchronized nodes.

How do nodes come to fire in synchrony? We may assume that features which are perceived at the same time activate nodes in synchrony. However, for complex, simultaneous perceptual input, it may be desirable to sort the input into distinct pieces.

One way to accomplish this would be to apply the concept identification algorithm. Normally applied to the whole backcloth to identify latent concepts, regardless of activation, the concept identification algorithm (Section 2.3) may instead be applied to the input, picking out the closely related nodes and sorting them into likely concepts. Based on this sorting, the different concepts perceived in the input might come to fire at different rates, holding them distinct. An automatic mechanism might be proposed to vary the value of the strength parameter, p, so as to pick out the optimal set of concepts in the input. A high value of p reduces the input to a few small islands of tightly connected nodes; a low value of p coalesces the input into one large, connected group of loosely connected nodes. An optimal value for p would be one which sorts the input into one to seven concepts, maximizing the size and connectedness of each concept.

How do concepts which are initially firing at different rates come to fire in synchrony? First, we propose that the formation of a link between two active, asynchronous nodes will cause those nodes to begin firing in synchrony. We have suggested above that explicit concept formation can make use of symbolic nodes to indirectly couple concepts: symbol a from concept A is linked to symbol b from concept B, causing the concepts to fire at the same rate, and leading to the forming or strengthening of association links between the core nodes of A and the core nodes of B.

## 2.9 Cognitive Brittleness

The encoding and storage of knowledge in a memory system lies at the heart of the cognitive brittleness problem. Single coding approaches either must continuously add new knowledge elements and suffer information overload, or simplify by using knowledge reduction methods to handle a specific task and thereby lose some degree of cognitive

expertise. Our associative memory model approach is less susceptible to cognitive brittleness. This is achieved in part by encoding knowledge in three ways or levels of granularity: feature sets, type sets and concept sets, as described in this section. Because type sets and concept sets emerge from associative strength patterns of features, the space of concepts can grow in an essentially unbounded manner while the feature space can be bounded. This enables the memory system to be open to accumulating vast amounts of new knowledge within a fixed feature infrastructure without necessarily disturbing storage space and temporal processing performance of the model. Thus, this modeling approach can scale from providing memory services for small well-defined work domain to large, complex and ill-defined ones where more knowledge would come into play.

## 3.0 Putting the Associative Memory Model in Context

### 3.1 Abstraction Hierarchy

The focus of the DCOG framework is on knowledge-based behavior. Our associative memory model addresses the existence of knowledge and its use in actor behavior. A knowledge-based memory system such as this often dominates behavior of an actor. It is clearly dominant when an actor must solve a complex problem. It is useful for: (1) defining or framing the problem, (2) for making a decision about how to solve the problem, (3) for making a decision about how to proceed even when the solution is not yet known or adequately understood, and (3) for guiding actions implementing the solution. Up to this point, we have been expressing our memory system in a state-change form. It can be difficult to see the connection between this state-change type of model and the more apparent sequential steps an actor appears to take during complex problem solving. A four-level abstraction hierarchy that defines different ways of viewing knowledge-based human behavior and performance allows us to describe the sequential external reality in concert with the lower-level state-change processes. This hierarchy is depicted in Figure 19.

The top layer of the hierarchy represents the view of cognitive behavior noted by an outside observer, the second level represents the subject's own awareness of his/her cognitive processes, the third level lays out the structures and operations used to accomplish recognition, and the fourth level expresses the mathematical algorithms used to encode the operations.

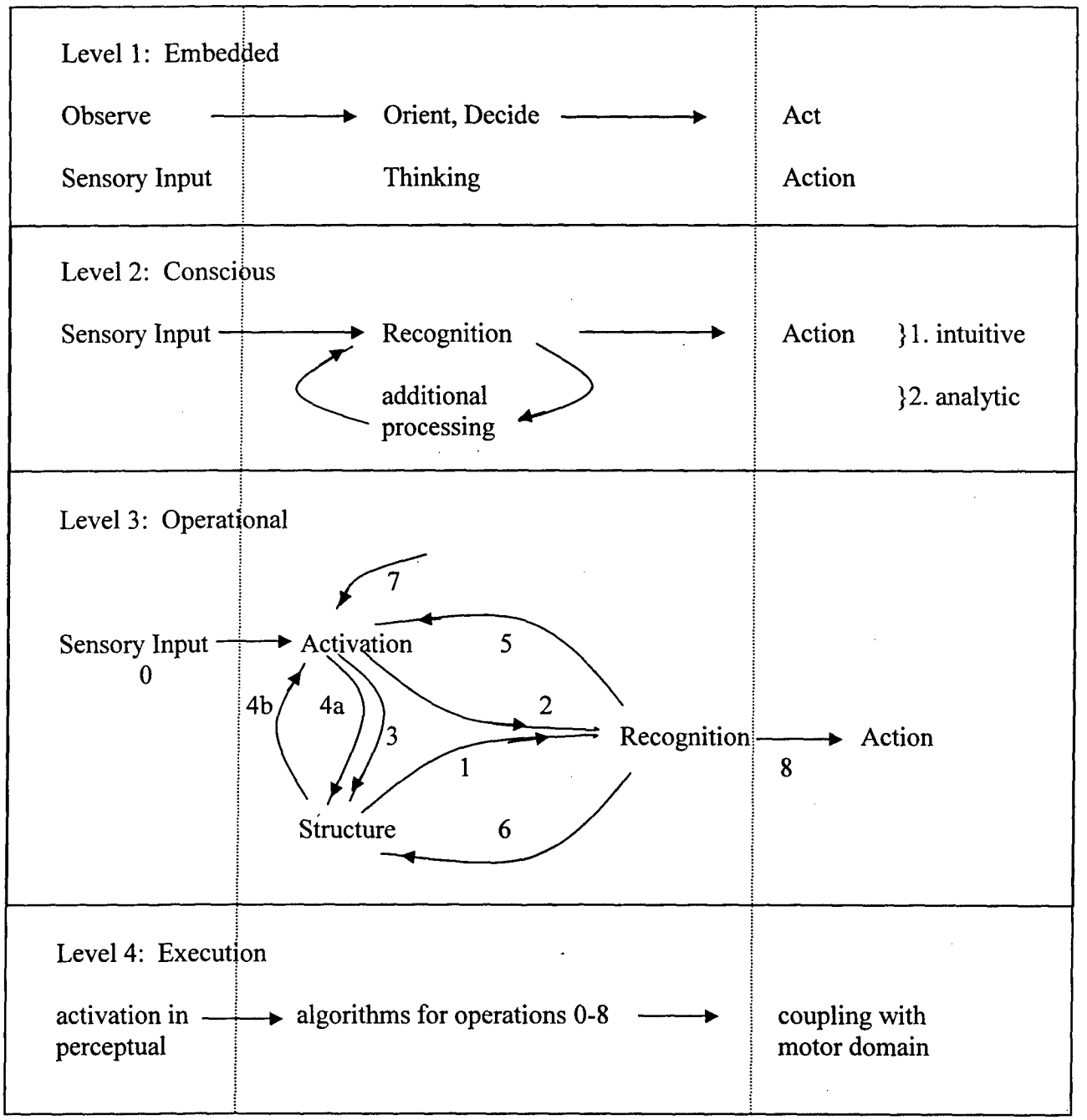



**Figure 19: Four level abstraction hierarchy**

### 3.1.1 Level 1: Embedded Level

This is the level of observation at which the subject's actions are embedded or situated in the world. This level takes the point of view of an observer external to the subject. To the observer, the subject appears to follow a sequence of perceiving input, thinking about it, and then acting. These stages correspond to the OODA loop (descriptive model): Observe, Orient/Decide, and Act, which is often used to describe decision making and complex problem solving.

The sensation that the subject follows a sequential process is an artifact. Sensory input continues to come in during the thinking process. Motor action continues as well, concurrently modifying sensory input. We retain the O-OD-A sequence, however, as a useful expository device.

### 3.1.2 Level 2: Conscious Level

The Conscious level represents the subject's understanding of his/her own cognitive states, and includes information about the subject's use of strategic processes.

At the Conscious level, we recognize that the thinking processes designated at the Embedded level may take two modes, 1. intuitive and 2. analytic. In the intuitive mode, the subject perceives sensory input, recognizes concepts, and takes action. The recognition process is automatic: The subject "just knows." In the analytic mode, we add additional processing, in the form of the subject shifting attention and/or creating and manipulating elements in the symbolic domain. The subject has the sensation of "reasoning it out." Since this is a state change model, the additional processes create an effective loop, feeding back into the recognition stage.

If we start with the additional processes, we can interpret the analytic mode as hypothesis formation and testing, as illustrated in Table 6:

**Table 6: Hypothesis formation and testing in the analytic mode**

Hypothesis formation

Generate Symbols, Shift Attention

Hypothesis testing

Recognition
Action
Sensory Input
Recognition

The shift between analytic and intuitive modes is sometimes caused by automatic processes (for example, the failure to recognize a coherent concept will invoke additional processing), but it may also be a result of subject-specific strategies. Some subjects may be more likely to rely on an intuitive mode, while others may be more given to using symbolic reasoning.

3.1.3 Level 3. Operational Level

At the Operational level, we decompose the subject's cognition into several specific operations within associative memory. The crucial problem for a model of associative memory is that of information overload. We must have methods that simplify the input enough to meet memory constraints, while still retaining important information. Therefore, in contrast to the emphasis on analytic vs. intuitive modes at the conscious level, at the

operational level we focus on distinguishing temporary states of mind from more stable structures which encode memory.

The operations are organized around the key elements of Activation, Structure, and Recognition, as shown in the diagram in Figure 19. Numbers refer to operations:

**Table 7: Operations**

0. perceptual activation

1. concept identification

2. activation slicing

3. association

4a. firing

4b. spreading activation

5. attention shifting

6. symbol creation

7. decay of activation

8. recognized concept

Activation reflects the temporary activation of mental nodes representing input features. Structure reflects the more stable associational links among nodes. Each node has an activation threshold, so that it may take repeated perception of a feature to push a node over its activation threshold. An active node then fires, spreading activation energy to its associated neighbors (4). While a node's activation level eventually decays so that the node becomes inactive (7), we retain information about the co-occurrence of features by forming or strengthening associations between co-active nodes (3).

Together, Activation and Structure combine to yield Recognition (1,2). The additional thinking processes of the Conscious level are reflected here as operations initiating at Recognition and affecting either Activation or Structure. Attention shifting spreads activation to different nodes (5); Symbol creation changes the structure by creating new nodes and associating them with currently active concepts (6). This is the source of explicit concept formation.

In relating Level 3 to Level 2, we can see that the operations 1, 2, 4a, and 4b of Level 3 fit the Recognition path in Level 2; while operations 5 and 6 of Level 3 reflect Additional Processing in Level 2. The remaining Level 3 operations, Decay (7) and Association (3), are time-dependent operations which influence both Recognition and Additional Processing. We take up the implementation issues of these operations in Part IV.

The preceding characterization of the operational level illustrates simple constructs (nodes and concepts). We must also address more complex structures: the coupling of concepts to produce threads, and the use of symbols to express relationships among concepts.

We can group operations according to two dimensions: temporary vs. stable, and simple vs. complex. This gives us the following arrangement:

**Table 8: Classification of operations as temporary or stable, simple or complex**

| | features | concepts |
|---|---|---|
| temporary changes<br><br>(synchronization) | co-activation<br><br>(features are active simultaneously) | coupling<br><br>(concepts firing in synchrony) |
| stable changes<br><br>(strengthened connections) | associations | no direct concept-to-concept associations<br><br>generation |

The fourth combination, stable U complex, has no direct instantiation: Since concepts are not represented hierarchically, there is not direct way to form an association between two concepts. Instead, the fourth cell is filled indirectly, by **generation**: To represent a stable link between two concepts, we generate symbols for each concept and form associations between the symbols. At the symbolic level, we may also form symbolic concepts representing semantic relationships. Suppose we have two concepts, A and B that participate in a semantic relationship, F. We represent this by generating a symbol $a_x \in A$, and a symbol $b_y \in B$, and noticing that $a_x$ and $b_y$ represent an exemplar of a concept Z in the semantic domain. The relationship F(A, B) is represented indirectly, through the

relationship $Z(a_x, b_y)$. Experts become adept at recognizing and using such symbolic concepts, so that their behavior can once again be described as following the intuitive mode. See Section 2.4.6 for additional discussion of the use of symbols in coupling concepts.

3.1.4 Level 4: Execution Level

The Execution level is the level at which we specify the mathematical algorithms that implement the operational constructs. Here we find definitions such as example, type, and dimension, and procedures such as spreading activation and association (See Section 2.2).

## 3.2 Layers

In parallel to the abstraction hierarchy, which describes the subject's states and actions at multiple levels of abstraction, we can characterize the organization of information at multiple levels. When the subject is merely waiting for information, the backcloth (set of all nodes) can be characterized as diffuse: There is a variety of low-level activation, but no nodes are strongly activated, and activation initiating at one node does not tend to spread strongly in any one direction. It is as though the association strengths between nodes are dampened. In contrast, when the subject is engaged in accomplishing a task, the backcloth can be characterized as focused: There is a specific area of several active, connected nodes, and activating one node within the task area tends to quickly activate one or more other task-related nodes. We characterize these changes in the backcloth as layers: the World Layer refers to the diffuse activation of the subject when not engaged in a task; the Work Layer refers to the tightly connected nodes used by the

subject to work on a task. In the Work layer, any of the nodes related to the task tend to be associated with other task-related nodes. Thinking of one of the task-related nodes tends to activate others within the task. In the World Layer, in contrast, the associations are broader. Thinking of one of the task nodes may lead to other task nodes, but is just as likely to lead to other, non-task-related nodes. The Work Layer enables concept recognition specific to a task, even with incomplete input; the World Layer enables flexible behavior in an unpredictable world.

The Work Layer enables concept recognition specific to a task, even with incomplete input; the World Layer enables flexible behavior in an unpredictable world. Intermediate layers could also be specified. In Section 3.2 above, we show that the layers are not physical objects, but a characterization of the results of changing contextual information. In effect, differences in context simulate changes in association strength.

# 4.0 Shepard, Hovland, and Jenkins Categorization Task

## 4.1 Introduction

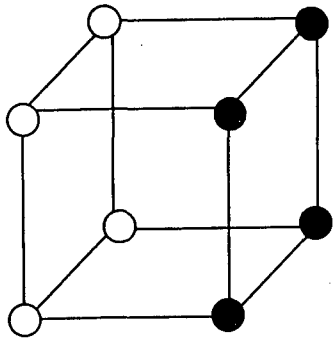
An air traffic control simulation was used as an initial testbed to demonstrate and analyze the performance of the DCOG-1 model (Tenney and Spector, 2001, Eggleston et al, 2000). This simulation testbed was modified to include a category learning problem within the air traffic control context. In its original form, a cognitive model (or a human) had to learn the basic air traffic control task. This involved acting as an airspace sector controller and learning how to process transiting aircraft in a manner that allowed fast and safe transit. The modified task will require additional real-time learning. The cognitive agent must also learn how to make correct responses to an altitude change request that involves three factors. The agent is given no prior information about what condition(s) of these three factors should result in granting a request. This modified testbed will be used to provide a richer test of cognitive behavior of computational cognitive models.

The parameters affecting the new altitude change request decision are divided into three binary dimensions: aircraft size (heavy or light), altitude (40,000 or 27,000), and turbulence (moderate or light). The subject must determine whether to accept or reject the altitude change request (initially, by guessing), and click the appropriate button. After a time interval, the system confirms or disconfirms the subject's decision, indicating whether the aircraft should have been accepted or rejected.

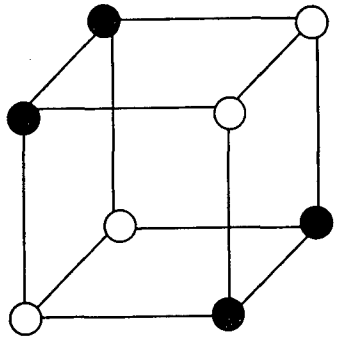
When processing the aircraft classification task, the subject must determine which aircraft are examples of type Accept and which are examples of type Reject. This is a form of concept recognition, and we process it with the mechanisms laid out in Section 2.3.

The test data will deal with six category types, based on Shepard, Hovland, and Jenkin's (1961) classification. The six types can be represented diagrammatically using x, y, and z axes. Human data show that Type I categories are easiest to learn, followed by Type II; Types III, IV, and V are intermediate in difficulty, and Type VI is very difficult to learn (ibid.). We must attempt to model these learning differences.

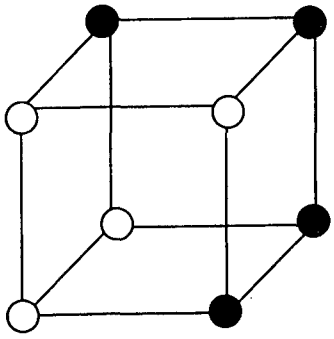
Type I



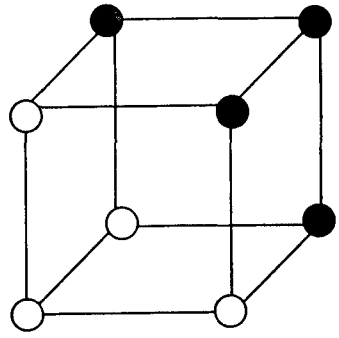
Type II



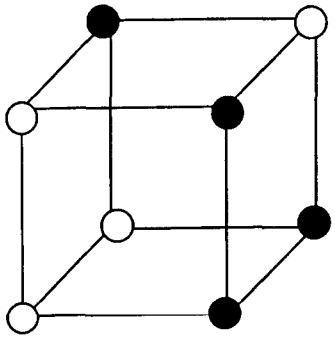
Type III



Type IV



Type V



Type VI

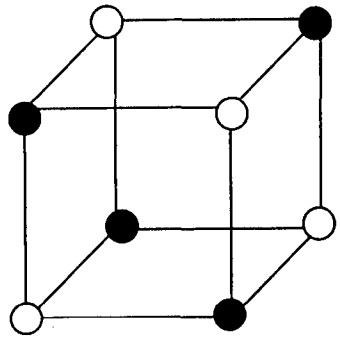


**Figure 20: The six Shepard, Hovland, and Jenkins (1961) classification types. Some of our models are rotations or reflections of the originals, but the relationships are maintained**

In Section 2.8, we related the expert's advanced performance to the ability to use concept recognition in place of symbolic reasoning. The Shepard, Hovland, and Jenkins categories simulate such expert-novice differences. The Type I category is a simple category based on one value of one dimension, and can be detected through emergent concept formation. In effect, everyone is an expert with Type I. The intermediate types (II, III, IV, and V) involve some complications or exceptions, and require some symbol creation. Type VI, the hardest to process, requires the creation of a separate symbol for each of the eight potential exemplars. Everyone must use novice-like symbolic reasoning to handle Type VI.

### 4.2 Emergent Concept Formation

We treat each value (e.g., moderate turbulence) as an independent feature. We treat "accept" and "reject" as additional features. For each aircraft perceived, we co-activate the nodes corresponding to the features. For the purposes of explaining the mathematics, we use the following notations. The initial input features, grouped into three dimensions with two values per dimension, are represented arbitrarily as x, $x^c$ (x-complement), y, $y^c$, z, and $z^c$. For example, if x is small aircraft size, then $x^c$ (x-complement) is large aircraft size. We use the labels A and R to refer to the input features for Accept and Reject. When necessary, we generalize A and R to T and $T^c$(target and target-complement). We represent statements of the type "Accept aircraft with the values large aircraft size and high altitude" with a function notation: A(x,y).

Within this system, the concepts of Accept and Reject are built up from exemplars of x, y, z, and T values. The Accept concept includes, as one of its nodes, the A (accept)

input feature. For Type I, the Accept and Reject concepts develop naturally through emergent concept formation. Suppose the correct generalization is A(f), an instantiation of Type I. Over time, we build up stronger associations between A and the crucial feature, f. At the same time, we build up stronger associations between R and the complement feature, $f^c$. Once the concepts have emerged, subsequent input with f will spread activation to A, and the subject guesses Accept. In reasoning about the problem, the subject may spread activation to A, and activate f.

We can represent the associations with an eight-node or "octagonal" association diagram, below. Given an input {x, y, z}, for example, association forms links between x-y, x-z, and y-z. Subsequent input indicates that this aircraft should be accepted; this activates the node A. The nodes A and x, y, z are coactive, and we form links between A-x, A-y, and A-z.

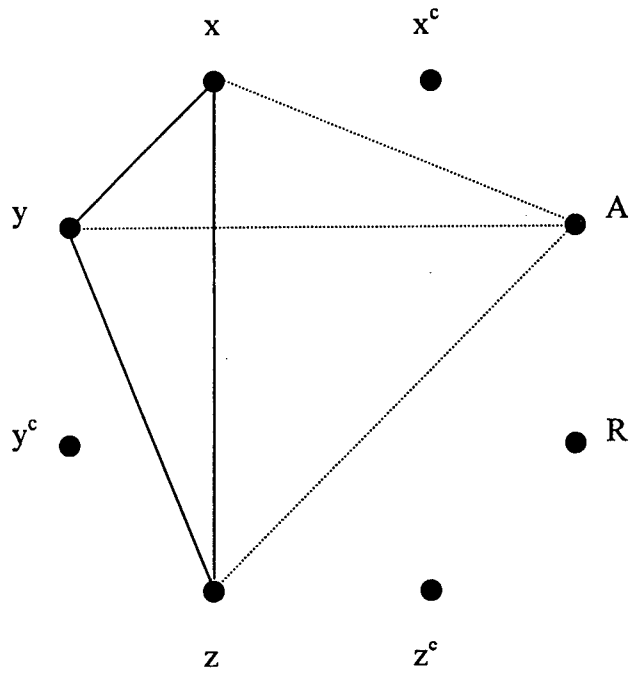



**Figure 21: Octagonal Association Diagram for {x, y, z} with subsequent input {A}**

We assume that the subject has been able to hold {x, y, z} active during the interval between the initial request and the confirmation of accept; if the interval is too long or something too distracting intervenes, the nodes x, y, and z might decay below firing threshold and become inactive. In that case, the activation of A would not create new associations.

If the subject views a balanced presentation of all possible aircraft types in Type I, we build up the associations shown in the octagonal diagram below. (Associations with R (reject) have been colored red here, but it is important to note that the subject does not distinguish the links in this way.) We can use the concept identification algorithm with strength $p \geq 3$ to isolate two concepts: $\{x - A\}$ and $\{ x^c - R\}$.

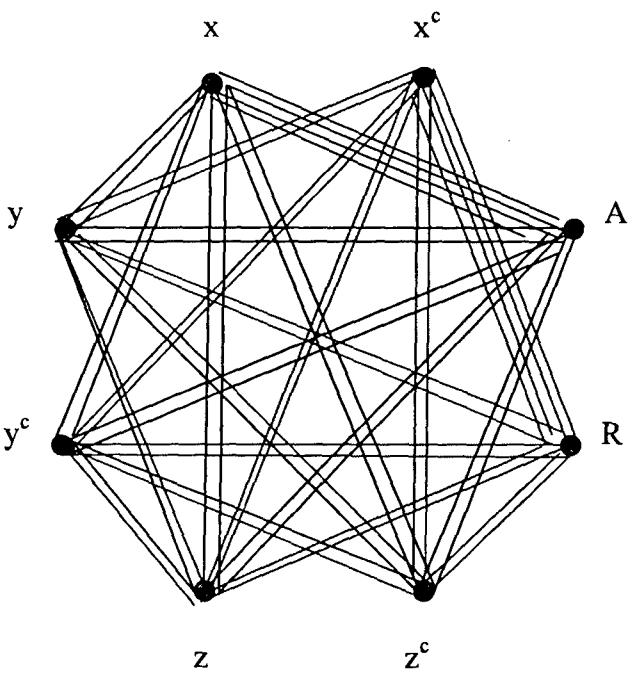



**Figure 22: Association diagram for Type I, A(x)**

The total input from all three aircraft features presents a slightly more complicated picture. This information can be represented in input-association charts, where the associations from each input feature are totaled to give an accept:reject ratio for the exemplar. For example, for Type I, A(x), the input {x, y, z} yields the ratio 8:4. The input, x, contributes 4:0 towards accept, but the inputs y and z each contribute 2 to accept and 2 to reject, since they are equally associated with accept and reject in the exemplars. The reverse situation holds for the input $\{x^c, y, z\}$.

**Table 9: Complete input-association chart for Type I, A(x)**

| | x | $x^c$ | y | $y^c$ | z | $z^c$ |
|---|---|---|---|---|---|---|
| Accept | 4 | 0 | 2 | 2 | 2 | 2 |
| Reject | 0 | 4 | 2 | 2 | 2 | 2 |

**Table 10: Input-association chart for {x, y, z} in Type I, A(x)**

| input: | x | y | z | total |
|---|---|---|---|---|
| Accept | 4 | 2 | 2 | 8 |
| Reject | 0 | 2 | 2 | 4 |

**Table 11: Input-association chart for $\{x^c, y, z\}$ in Type I, A(x)**

| input: | $x^c$ | y | z | total |
|---|---|---|---|---|
| Accept | 0 | 2 | 2 | 4 |
| Reject | 4 | 2 | 2 | 8 |

While emergent concept formation yields the correct results for Type I, it is not as useful in Types II – VI. Assuming a balanced presentation of exemplars, Type II and Type

VI will build up equal associations for each pair of features. For these types, moderate turbulence, for example, will be equally associated with A (accept) as with R (reject).

**Table 12: Complete input-association chart for Type II, $A(x,y^c) \cup A(x^c,y)$**

| | $x$ | $x^c$ | $y$ | $y^c$ | $z$ | $z^c$ |
|---|---|---|---|---|---|---|
| Accept | 2 | 2 | 2 | 2 | 2 | 2 |
| Reject | 2 | 2 | 2 | 2 | 2 | 2 |

**Table 13: Complete input-association chart for Type VI, $A(x^c,y^c,z) \cup A(x^c,y,z^c) \cup A(x,y^c,z^c) \cup A(x,y,z)$**

| | $x$ | $x^c$ | $y$ | $y^c$ | $z$ | $z^c$ |
|---|---|---|---|---|---|---|
| Accept | 2 | 2 | 2 | 2 | 2 | 2 |
| Reject | 2 | 2 | 2 | 2 | 2 | 2 |

In Type VI this happens because the exemplars are evenly distributed among the dimension values: there are no generalizations to find. In Type II this happens because the "good" value for one dimension appears equally with A and R, depending on the values for the second relevant dimension. So, while x is a good quality when paired with $y^c$, x is a bad quality when paired with y, and we build up equal associations between x—A and x—R.

Here are the exemplar tabs, the adjacency matrix, and the octagonal association diagram for Type II. In the octagonal diagram, the associations due to rejected aircraft are given in red to illustrate the source; the subject, however, does not retain this information.

**Table 14: Exemplars for Type II, A(x, $y^c$), A($x^c$, y)**

| AC # | Altitude x:Low | $x^c$:High | Turbulence y:No | $y^c$:Yes | AC Size z:Sm | $z^c$:Large | Accept | Reject |
|---|---|---|---|---|---|---|---|---|
| 1 | x | | x | | x | | | x |
| 2 | x | | x | | | x | | x |
| 3 | x | | | x | x | | x | |
| 4 | x | | | x | | x | x | |
| 5 | | x | x | | x | | x | |
| 6 | | x | x | | | x | x | |
| 7 | | x | | x | x | | | x |
| 8 | | x | | x | | x | | x |

**Table 15: Adjacency matrix for Type II, A(x, $y^c$), A($x^c$, y)**

| | x | $x^c$ | y | $y^c$ | z | $z^c$ | A | R |
|---|---|---|---|---|---|---|---|---|
| x | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| $x^c$ | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| y | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| $y^c$ | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| z | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| $z^c$ | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| A | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |
| R | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| | \|\| |

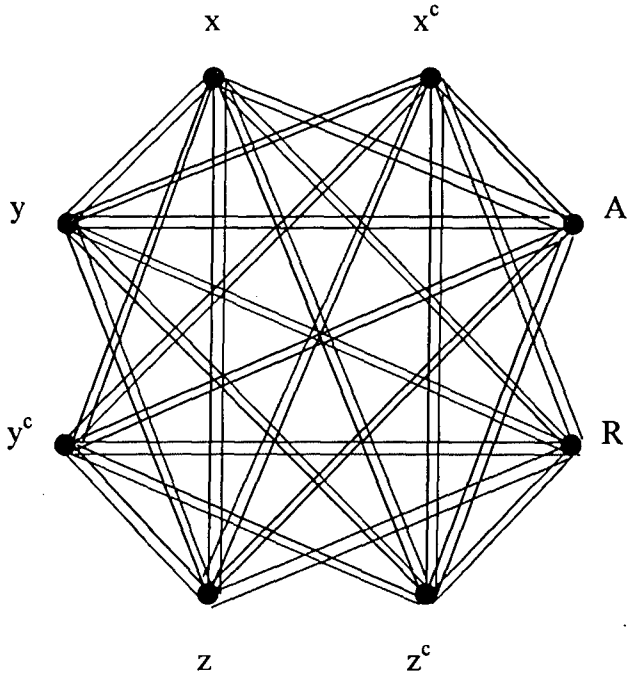



**Figure 23: Association diagram for Type II, A(x, $y^c$), A($x^c$, y)**

We observe that emergent concept formation alone will not provide the generalizations we need. We will explain how our system handles categorization in Types II-VI in Section 4.3 below. In this section, we summarize the effects of emergent concept formation on all six types. Here are the input/association charts for all six types, again, assuming a balanced presentation of exemplars.

**Table 16: Complete input association chart for Type I, A(x)**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 4 | 0 | 2 | 2 | 2 | 2 |
| Reject | 0 | 4 | 2 | 2 | 2 | 2 |

**Table 17: Complete input association chart for Type II, $A(x,y^c)$ U $A(x^c,y)$**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 2 | 2 | 2 | 2 | 2 | 2 |
| Reject | 2 | 2 | 2 | 2 | 2 | 2 |

**Table 18: Complete input association chart for Type III, $A(x)$ U $A(x^c,y,z)$ U $R(x,y,z^c)$**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 3 | 1 | 2 | 2 | 3 | 1 |
| Reject | 1 | 3 | 2 | 2 | 1 | 3 |

**Table 19: Complete input association chart for Type IV, $A(x)$ U $A(x^c,y,z)$ U $R(x,y^c,z^c)$**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 3 | 1 | 3 | 1 | 3 | 1 |
| Reject | 1 | 3 | 1 | 3 | 1 | 3 |

**Table 20:Complete input association chart for Type V, $A(x)$ U $A(x^c,y,z)$ U $R(x,y,z)$**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 3 | 1 | 2 | 2 | 2 | 2 |
| Reject | 1 | 3 | 2 | 2 | 2 | 2 |

**Table 21: Complete input association chart for Type VI, $A(x^c,y^c,z)$ U $A(x^c,y,z^c)$ U $A(x,y^c,z^c)$ U $A(x,y,z)$**

| | x | xc | y | yc | z | zc |
|---|---|---|---|---|---|---|
| Accept | 2 | 2 | 2 | 2 | 2 | 2 |
| Reject | 2 | 2 | 2 | 2 | 2 | 2 |

Below we present a table of the results from applying emergent concept formation to each of the six types. The exemplars are labeled a through h, according to the following arrangement (Figure 24):

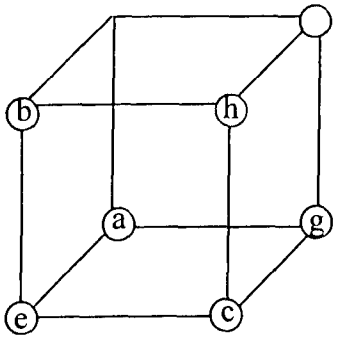



**Figure 24: Alphabetic labels for nodes taking on values 0 or 1 on the x, y, and z axes. For example, node a represents the values (0,0,1)**

The exemplars in the table are grouped according to Kruschke's (1992) classifications of central, peripheral, and exceptional exemplars for Types III, IV, and V. A central exemplar is one which fits the main generalization of the category; a peripheral exemplar shares some values with the central node(s); an exceptional exemplar violates the generalization. Kruschke claims that central exemplars are learned fastest, followed by peripheral exemplars, and finally exceptional exemplars.

Next to each group of exemplars is a ratio that indicates the proportion of Accept to Reject associations which will emerge, given a balanced presentation of exemplars. When the ratio is 1, a question mark indicates that no decision can be made on this basis. An exclamation point is used to show when the ratio indicates the wrong answer.

**Table 22: Association strengths for Accept or Reject for central, peripheral, and exceptional elements**

| Type | A / R | Central | Peripheral | Exceptional |
|---|---|---|---|---|
| I | accept | h,d,g,c 8:4 | | |
| | reject | b,f,e,a 4:8 | | |
| II | accept | c,g,b,f 6:6 [?] | | |
| | reject | h,d,e,a 6:6 [?] | | |
| III | accept | g,d 8:4 | f,c 6:6 [?] | |
| | reject | e,b 4:8 | a,h 6:6 [?] | |
| IV | accept | d 9:3 | f,h,g 7:5 | |
| | reject | e 9:3 | b,a,c 5:7 | |
| V | accept | c 7:5 | h,g 7:5 | f 5:7 [!] |
| | reject | e 5:7 | b,a 5:7 | d 7:5 [!] |
| VI | accept | | | a,b,c,d 6:6 [?] |
| | reject | | | e,f,g,h 6:6 [?] |

Many of the types can be described by other arrangements of central generalizations and peripheral exceptions. Type V, for example, does not have to be analyzed as central and peripheral. It can also be considered a variant of Type I, A(x), with two exceptional exemplars (d and f). This fits our prediction, that the "peripheral" exemplars (h, g, b, a) show the same accept:reject ratio as the "central" exemplars (e, c), but it goes against Kruschke's learning data.

Both analyses of Type V designate d and f as exceptions. The presence of these exceptions leads to the close 7:5 ratio for the other exemplars, as shown in these input/association charts comparing exemplar a in Type I and Type V.

**Table 23: Input-association chart for a={ $x^c$ $y^c$ z } in Type I, A(x)**

| input: a: | xc | yc | z | total |
|---|---|---|---|---|
| Accept | 0 | 2 | 2 | 4 |
| Reject | 4 | 2 | 2 | 8 |

**Table 24: Input-association chart for a={ $x^c$ $y^c$ z } in Type V, A(x) U R(x,y,z) U $A(x^c,y,z)$**

| input: a: | xc | yc | z | total |
|---|---|---|---|---|
| Accept | 1 | 2 | 2 | 5 |
| Reject | 3 | 2 | 2 | 7 |

Here, the presence of associations from $A(x^c, y, z)$ changes the result of the input $x^c$ from 0 Accept, 4 Reject in Type I to 1 Accept, 3 Reject in Type V. The total number of associations still correctly favors reject.

For the Type V exceptions themselves, d and f, the input association charts show how the preponderance of associations leads to the wrong ratio:

**Table 25: Input-association chart for f={ $x^c$ y z } in Type V, A(x) U R(x,y,z) U $A(x^c$, y,z)**

| input f: | xc | y | z | total | |
|---|---|---|---|---|---|
| Accept | 1 | 2 | 2 | 5 | [!] |
| Reject | 3 | 2 | 2 | 7 | |

**Table 26 : Input-association chart for d={ x y z } in Type V, A(x) U R(x,y,z) U $A(x^c$, y,z)**

| input d: | x | y | z | total | |
|---|---|---|---|---|---|
| Accept | 3 | 2 | 2 | 7 | [!] |
| Reject | 1 | 2 | 2 | 5 | |

Type IV actually emerges to some extent. This is because Type IV has an Accept exemplar, c: {x, y, z}, which can be viewed as the prototype of a fuzzy concept: The other acceptable exemplars are similar in two dimensions to the prototype: h: {x, y, $z^c$}, g: {x, $y^c$, z}, and f: {$x^c$, y, z}, contributing two 3:1 accept inputs and one 1:3 reject input. This parallels Kruschke's analysis of c as central, with f, h, and g being peripheral.

**Table 27: Input-association chart for d={ x y z } in Type IV, A(x) U A($x^c$,y,z) U R(x,$y^c$,$z^c$)**

| input d: | x | y | z | total |
|---|---|---|---|---|
| Accept | 3 | 3 | 3 | 9 |
| Reject | 1 | 1 | 1 | 3 |

**Table 28: Input-association chart for g={ x $y^c$ z } in Type IV, A(x) U A($x^c$,y,z) U R(x,$y^c$,$z^c$)**

| input g: | x | yc | z | total |
|---|---|---|---|---|
| Accept | 3 | 1 | 3 | 7 |
| Reject | 1 | 3 | 1 | 5 |

In the following section, we will see how symbol generation helps us detect non-emergent categories, and explains the various learning rates for the six types.

## 4.3 Hypothesis Formation

### 4.3.1 Hypothesis Formation

We have seen that, while Type I categories can be detected by emergent concept formation, the other Types do not emerge, or emerge imperfectly. We claim that the subject must engage in symbolic reasoning to detect the categories in Types II – VI. It is

possible that a person may presume, after some experience, that only imperfect answers (i.e. a probabilistic solution) are achievable, in which case the emergent concept may be deemed acceptable for Type II problems. It is unlikely that such a solution based only on emergent concept formation would be satisfying to the performer for Type III – Type 6 problems, because probabilistic-based performance would be very low. Thus, symbolic reasoning, as a minimum, must be a dominate factor in solving these category learning problems.

Symbolic reasoning involves both symbol creation and attention shifting (see Section 5.0). The symbols that are created constitute hypotheses about the Accept and Reject concepts. For example, if we see that an aircraft with low altitude, small size, and moderate turbulence is accepted, we may guess that low altitude leads to acceptance. In this situation, the subject creates a symbol (call it "Accept low altitude") and links it to both the feature (low altitude) and the target (Accept).

For Type I categories, the formation of a symbol is not necessary to detect the category, since emergent concept formation will identify the relevant feature. However, as we saw in the input/association charts, above, the fact that the irrelevant dimension values are evenly distributed between Accept and Reject means that there are some misleading associations. Given the category A(x), for example, the input {x,y,z} is associated 8:4 with accept over reject. By using the hypothesis, A(x), we feed additional attention to Accept, and make it easier to classify exemplars.

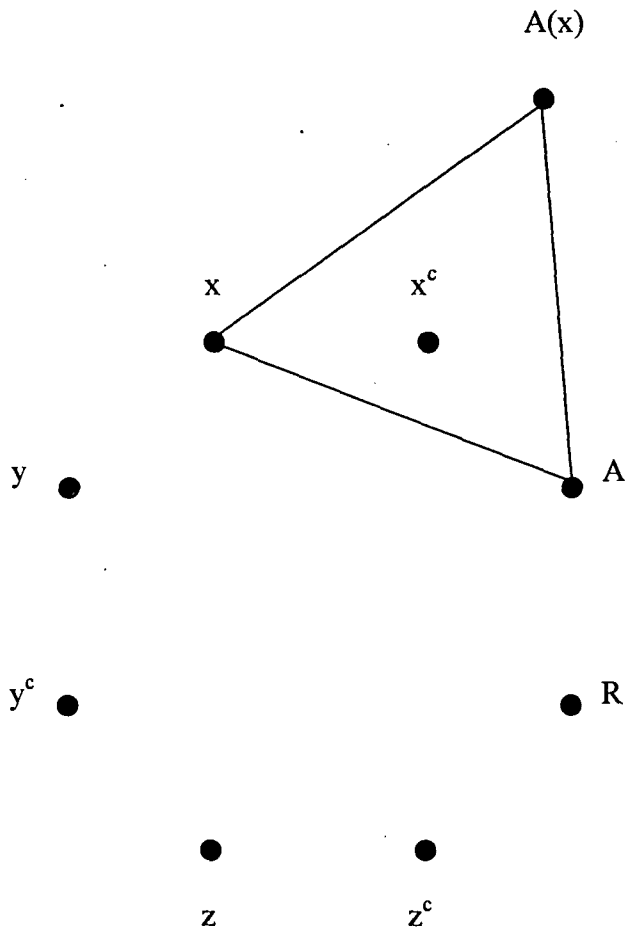



**Figure 25: Hypothesis Formation 1. Given the input x—A, we propose the hypothesis, "accept x."**

We posit three hypothesis formation procedures:

Hypothesis Formation 1 (6)

If the value x and the target T are co-active, create the node T(x) and link it to x and to T.

Hypothesis Formation 2 (7)

If the values x and y and the target T are all co-active, create the node T(x,y)

and link it to x, y, and T.

Hypothesis Formation 3 (8)

If the values x, y, and z, and the target T are all co-active, create the node T(x, y, z) and link it to x, y, z, and T.

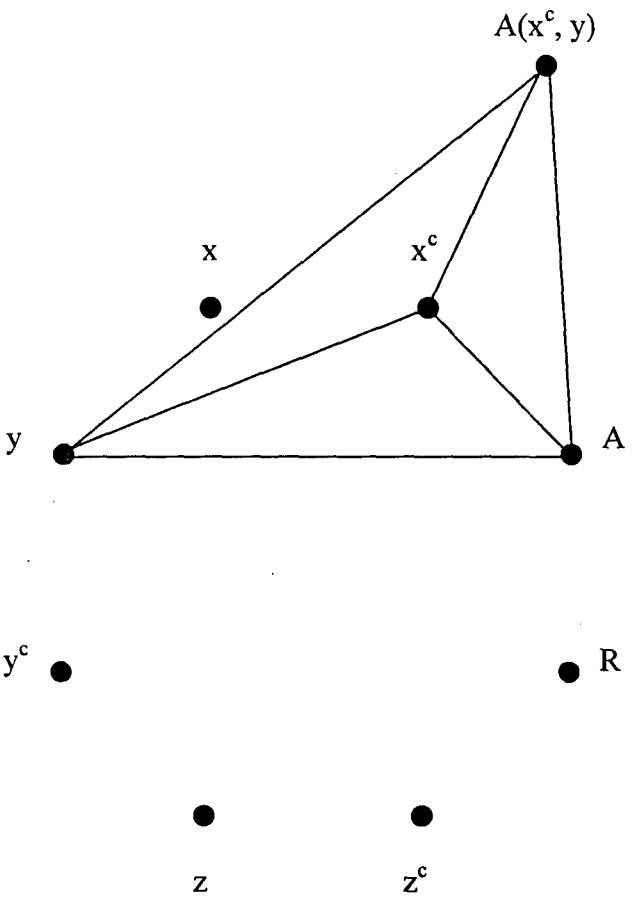



**Figure 26: Hypothesis Formation 2. Given input of y—A and $x^c$—A, we propose the hypothesis, "accept a combination of $x^c$ and y"**

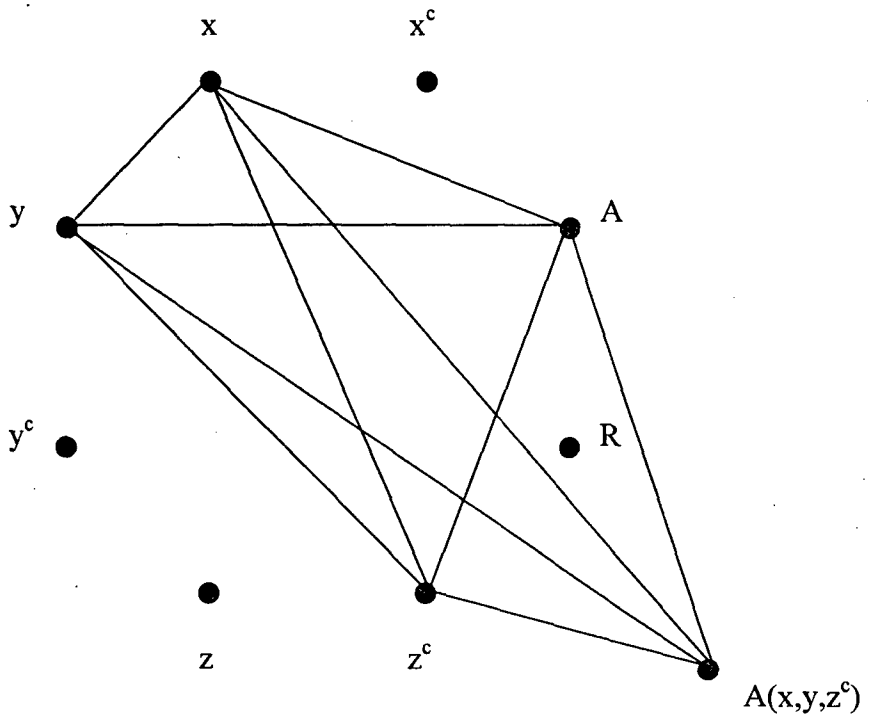



**Figure 27: Hypothesis Formation 3. Given the input, x—A, y—A, $z^c$—A, we propose the hypothesis, "accept the combination, x, y, $z^c$"**

One possibility we should consider is that the existing aircraft labels (e.g., VIR199) may serve as symbols, being co-active with all three input features. A subject may be able to hold the input features active by feeding attention to the aircraft symbol. The existence of the aircraft symbol may facilitate creation of a three-dimensional hypothesis symbol, such as the one shown above.

Given the input x, y, z from any one aircraft, there are seven potential hypotheses. We assume that the subject chooses one hypothesis, based on two factors: contextual activation and subject-specific strategies. Contextual activation preferentially activates one dimension (e.g., x), leading the subject to form hypothesis on that dimension. Subjects may also follow individual strategies: One subject may prefer single-dimensioned

hypotheses like T(x); another subject may prefer to try multiple-dimensioned hypotheses like T(x, y, z).

In addition, subjects might form complementary hypotheses. That is, if the subject decides to work with the hypothesis "accept low altitude", the subject may also make use of the complementary hypothesis, "reject high altitude." We suppose that the typical subject will use the complementary hypothesis in the single dimension, but that multiple dimension complementary hypotheses are available on a subject-specific basis.

Complementary Hypothesis Formation 1 (9)

If T(x) is posited, also posit $T^c(x^c)$

4.3.2 Hypothesis Rejection

Attention shifting comes into play when the subject has made an incorrect hypothesis. In this case, the hypothesis will activate one target, T, and the eventual feedback will activate the other target, $T^c$. We claim the subject, on perceiving this conflict, will shift attention to another dimension, guiding the choice of the next hypothesis. For example, suppose we saw that low altitude, small aircraft, moderate turbulence was accepted, and formed the hypothesis "accept low altitude". When subsequently seeing the input, low altitude, large aircraft, moderate turbulence, activation would spread from low altitude, to the hypothesis symbol, and thence to Accept. If we receive feedback to Reject that aircraft, we then shift attention away from the altitude dimension and instead concentrate on either turbulence or aircraft size as the relevant dimension. We propose to guide hypothesis shifting by an ordered list of dimensions, for

example: {altitude, turbulence, aircraft size, altitude and turbulence, altitude and aircraft size, turbulence and aircraft size, altitude and turbulence and aircraft size}. Once the single dimensions are exhausted, we turn to double dimensions and finally consider triple dimensions. This attention list might vary for different subjects, as mentioned above.

We make use of the dimensions in shifting attention. For example, if the hypothesis "accept aircraft with high altitude" is discredited, the model shifts attention to either the turbulence dimension or the ac size dimension, by feeding activation energy to the dimension node. On the next input, the increased activation to that dimension will lead the model to create its next hypothesis based on the value for that dimension. In the example below, input of {x,y,z,A} led to the formation of an hypothesis, A(x); this was discredited by subsequent input of $\{x,y^c,z,R\}$. Attention now shifts away from DimX, to DimY. Activation spreads from DimY to y and $y^c$, picking out $y^c$, which is still active from the last input. The co-activation of $y^c$ and R allows us to form the new hypothesis, $R(y^c)$.

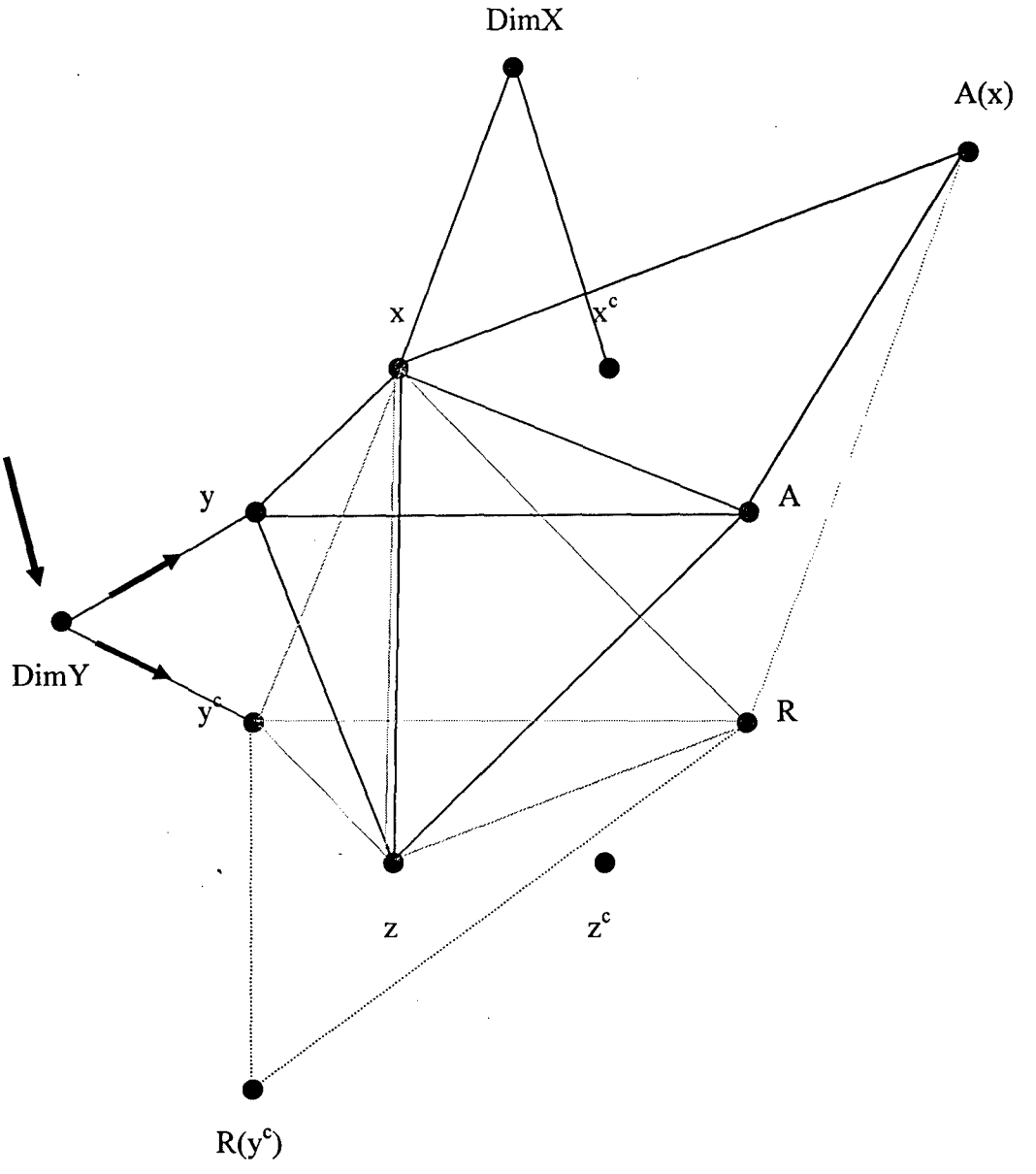



**Figure 28: Hypothesis A(x) discredited in favor of hypothesis $R(y^c)$**

Instead of erasing or directly inhibiting bad hypotheses, we use association to render them harmless. Consider again the example in which we thought the correct hypothesis was "accept low altitude." If an aircraft with low altitude is rejected, then we link Reject to the still-active hypothesis, so that the future instances of low altitude activate

the bad hypothesis to no avail: It contributes nothing to the classification of the aircraft, since it equally activates both Accept and Reject.

4.3.3 Workload

Hypotheses are symbols and associated links to input and target nodes. Creating and maintaining an active symbol requires work. We claim that the dimensionality of a hypothesis affects the workload required to maintain it. The simple, one-dimensional hypothesis, A(x), requires less work to maintain than the two-dimensional hypothesis, A(x,y), while the three-dimensional hypothesis A(x,y,z) imposes the heaviest workload. When we consider the six types in terms of hypothesis workload, we can characterize the desired learning differences. We treat Types III, IV, and V as variants of Type I, having a general, one-dimensional hypothesis with two three-dimensional exceptions. We assign a weight of 1 to a one-dimensional hypothesis, 2 to a two-dimensional hypothesis, and 3 to three-dimensional hypotheses. We assume here that there is no cost assigned to complementary hypotheses.

**Table 29: Workload as a function of hypothesis dimensionality**

| | Hypotheses | | | |
|---|---|---|---|---|
| Type | T(x) | T(x,y) | T(x,y,z) | Total Dim |
| I | 1 | | | 1 |
| II | | 2 | | 4 |
| III | 1 | | 2 | 7 |
| IV | 1 | | 2 | 7 |
| V | 1 | | 2 | 7 |
| VI | | | 4 | 12 |

Note. Only positive hypotheses are counted (e.g., Type I is considered fully identified by A(x), without reference to the complementary hypothesis, $R(x^c)$.)

The hypothesis workload analysis matches the general learning pattern for the six types: Type I is the easiest to learn, followed by Type II; Types III, IV and V are intermediate in difficulty; and Type VI is considerably harder.

There are alternative ways to analyze some of the types, but they all fall into the same approximate workload categories: I < II< III, IV, V < VI. For example, Type IV can be described by three overlapping two-dimensional hypotheses, A(x,z) U A(x,y) U A(y,z), giving it a dimensionality weight of 6.

An alternative approach to determining workload for the six types is to appeal to the conflict in hypotheses in Types III, IV, and V. Type III, for example, is analyzed is the union of A(x) with $R(x, y, z^c)$ and $A(x^c, y, z)$. The input $\{x, y, z^c\}$ causes a conflict between the two hypotheses, A(x) and $R(x, y, z^c)$, which is resolved in favor of the more specific hypothesis.

While Type II involves the multiple hypotheses, $A(x^c, y)$ U $A(x, y^c)$ U R(x, y) U $R(x^c, y^c)$, there is no conflict: Any input is uniquely specified by one hypothesis. If we attribute a cognitive cost to the conflict resolution process, we can better distinguish Type II from the more difficult Types III, IV, and V.

**Table 30: Workload as a function of hypothesis dimensionality and conflict resolution**

| Type | Hypotheses T(x) | T(x,y) | T(x,y,z) | Conflict | Total Dim |
|---|---|---|---|---|---|
| I | 2 | | | no | 2 |
| II | | 4 | | no | 8 |
| III | 2 | | 2 | yes | 8 + C |
| IV | 2 | | 2 | yes | 8 + C |
| V | 2 | | 2 | yes | 8 + C |
| VI | | | 8 | no | 24 |

Note. Both Accept and Reject Hypotheses are counted (e.g., Type I includes A(x) and $R(x^c)$ as hypotheses.) The symbol, C, represents the additional cognitive workload needed to resolve conflicting hypotheses.

In this second calculation of workload, we suggest that the Reject hypotheses count toward workload when they are required to fully specify the status of the exemplars. Type II, for example, cannot be simply A(x,yc) U A(xc, y), because that would leave exemplars like {x, y, z} unspecified.

Some alternative analyses of Types IV and V create no conflict, but, because of the full specification requirement, these analyses level a high workload. For example, Type IV could be A(x,z) U A(x,y) U A(y,z), with a workload of 2 x 3 = 6, but, because we would also have to include R(xc, yc), R(yc, zc), and R(xc, zc), the total workload is 12. Kruschke's analysis of Type III corresponds to A(x,z) U A(x,yc,zc) U A(xc,y,z), which would give a workload of 2 + 3 + 3 = 8, but the addition of R(xc, zc), R(xc, yc, z) and R(x, y, zc) raises the workload to 16.

### 4.4 Combining Direct and Indirect Activation

There are three sources of information about whether a given aircraft should be Accepted or Rejected: Direct activation from previous experience, indirect activation through hypotheses, and confirmation or disconfirmation from the system. We consider each in turn.

1. Direct spreading activation from input to target: x --------> A

Each time a feature is co-active with the target, the association link between that feature and the target is strengthened. Subsequent activation of the input feature spreads activation directly to the target, leading to the guess ("Accept x"). This is an example of emergent concept formation.

2. Indirect spreading activation from input to symbol to target: x ------> A(x) ------> A

When a feature (or set of features) is co-active with the target, we may create a symbol which connects to both the feature (or set of features) and the target. The symbol and its association links constitute a hypothesis ("Accept x").

Subsequent activation of the feature (or set of features) spreads activation to the symbol, which then spreads activation to the target, leading to the guess ("Accept x"). We claim that all input features must be present to activate the hypothesis: $A(x,y,z^c)$ will not be activated by the input {x, y, z}.

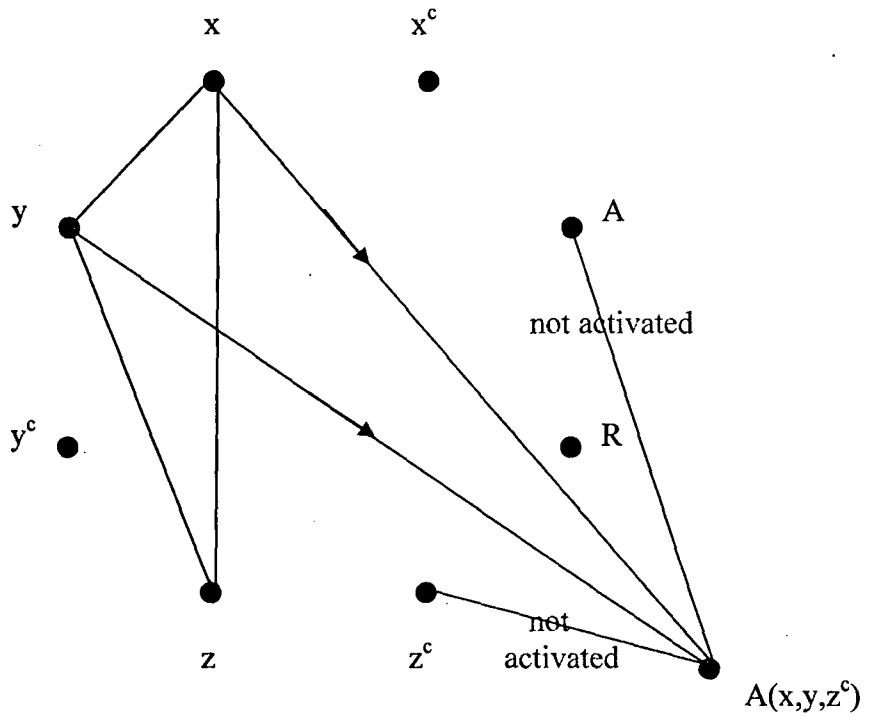



**Figure 29: Partial input {x, y} is insufficient to activate hypothesis $A(x, y, z^c)$**

3. (Dis)confirmation in the second stage of input, when the subject is told that the aircraft should have been Accepted or Rejected. This provides independent activation of either the node A (accept) or the node R (reject). Association then forms or strengthens links between that node and the other currently active nodes.

There are several ways in which these three sources of information can provide conflicting information. There are three possible kinds of conflict in arriving at a guess, and one further possible conflict after the guess has been formed:

- 1 vs. 1 : the input feature is associated equally with each of the two targets
- 1 vs. 2 : the direct and indirect associations (hypotheses) give different answers
- 2 vs. 2 : there are multiple hypotheses which may apply to the input

- 1 & 2 vs. 3: Once the guess has been determined, there may be a conflict between the guess and the third source of information (i.e., we may guess wrong).

We now consider in detail how to resolve each of the four conflict types.

For 1 vs. 1, where the input feature is associated equally with each of the two targets, we require the subject to make a guess. The guess may be random; the guess may be influenced by the subject's existing activation due to context or attention; or the guess may be derived via subject-specific strategies, such as "when in doubt, guess Reject."

For 1 vs. 2, where the direct associations and the hypothesis symbols give different answers, we assume that the subject makes a guess based on the total activation energy supplied to the two possible targets, summing up both direct and indirect activation. We assume that hypothesis formation establishes associations between the feature and the symbol, and between the symbol and the target, at a certain strength, given by the parameter, h. We tentatively set h at 3, reflecting the attention being fed to the hypothesis.

For 2 vs. 2, where there are multiple applicable hypotheses, we assume that the most specific active hypothesis (the one with the greatest number of input features) will be applied. (Recall that, to be activated, a hypothesis must receive activation from all relevant input features.) In Figure 30, the appropriate input is present to activate both R(x) and $A(x,y,z^c)$. We resolve the conflict in favor of the most specific hypothesis, $A(x,y,z^c)$.

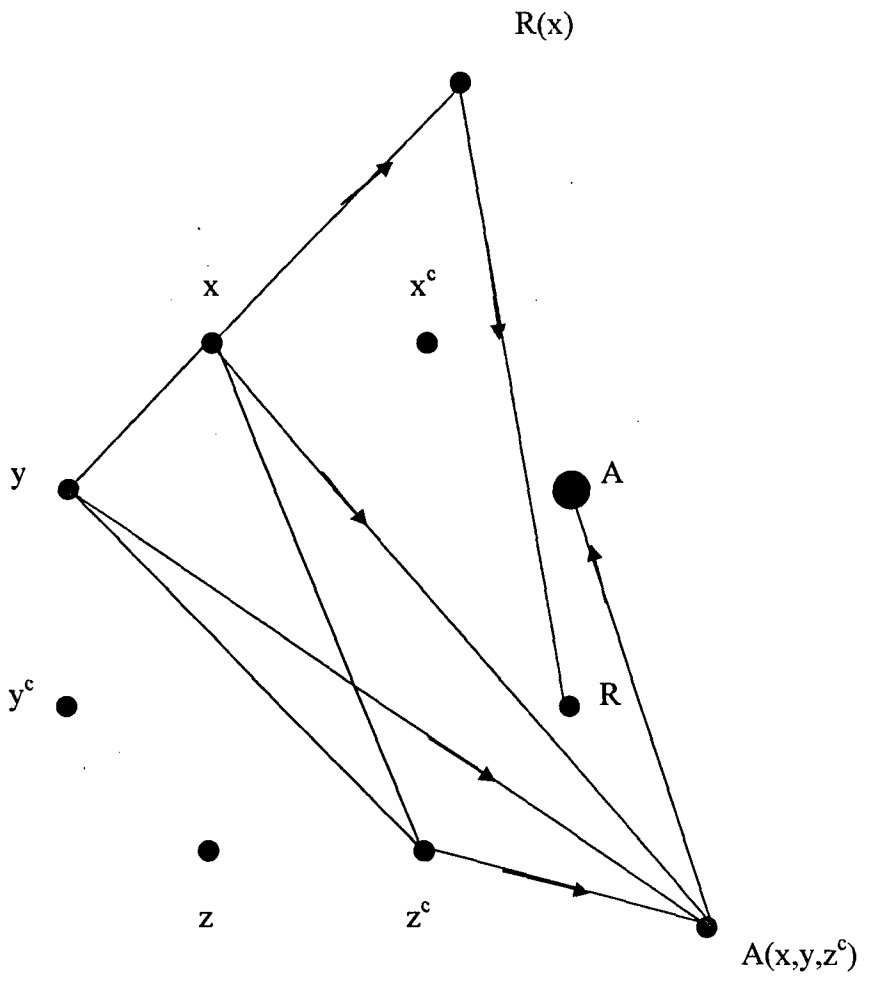



**Figure 30: Resolving a conflict in favor of the most specific hypothesis. Here, the hypothesis $A(x, y, z^c)$ prevails over the less specific hypothesis, R(x)**

When we guess wrong on the basis of an hypothesis, there are three possibilities.

1. The hypothesis is just wrong.
2. The hypothesis is OK, but there are some exceptions.
3. The hypothesis is OK; it has apparent exceptions, but these are data errors.

Corresponding to these three possibilities are three desired outcomes:

1. Abandon the hypothesis.

2. Keep the hypothesis, and make note of the exceptions.

3. Keep the hypothesis, and learn to ignore the errors.

Here are examples of the way our system handles these three circumstances.

**Table 31: Initial incorrect hypothesis is corrected by subsequent data**

The generalization is Accept y.

Input x y $z^c$; confirmation: Accept

Subject creates hypothesis, A(x)

Subject creates complement hypothesis, R($x^c$)

Input x $y^c$ z

Subject guesses Accept

Disconfirmation: !Reject

Subject shifts attention to Dim-y

Subject creates hypothesis, R($y^c$).

Subject creates complement hypothesis, A(y).

**Table 32: Initial hypothesis is correct, but exceptions are noted over time**

The generalization is Accept x, with exceptions Reject x y z, Accept $x^c$ y z.

Input x $y^c$ $z^c$; confirmation: Accept.

Subject creates hypothesis, A(x)

Subject creates complement hypothesis, R($x^c$)

Input x $y^c$ z.

Subject guesses Accept.

Confirmation: Accept.

Hypothesis is strengthened.

Input x y z.

Subject guesses Accept.

Disconfirmation: !Reject.

Subject keeps strong hypothesis, A(x).

Subject adds hypothesis, R(x y z)

**Table 33: Initial hypothesis is correct, but errors in the data cause some confusion**

The generalization is Accept x, with some errors in the data.

Input x y z; confirmation: Accept.

Subject creates hypothesis, A(x)

Subject creates complement hypothesis, R($x^c$)

Input x $y^c$ z.

Subject guesses Accept.

Confirmation: Accept.

Hypothesis is strengthened.

Input x y $z^c$.

Subject guesses Accept.

Disconfirmation: !Reject. [this is the error in the data]

Subject keeps strong hypothesis, A(x).

Subject adds hypothesis, R(x y $z^c$).

...

Input x y $z^c$.

Subject guesses Reject.

Disconfirmation: !Accept.

Subject shifts attention away from Dim(x,y,z)

A good hypothesis will be supported over time by many data points. Each time the hypothesis is confirmed as providing the correct data, the association strengths for the hypothesis are strengthened. On the other hand, a wrong hypothesis will not be well-supported by the data, and will have only weak connection strengths. So, in resolving conflicts, we appeal to the strength of these connections.

If a hypothesis is weak (has weak connection strengths), we shift attention to a different dimension (e.g., x, y, or z), avoiding the dimension associated with the discredited

hypothesis. The shift in attention is implemented by feeding activation energy to the new dimension or dimensions. This energy ensures that the next hypothesis proposed will involve the new dimension(s).

The shift in attention may also vary, depending on subject-specific strategies. For example, some subjects might assume a strategy of considering all three dimensions (xyz), while other subjects might prefer to first consider the more general, single dimension hypotheses. Here, we assume that the order of attention shift is x, y, z, xy, xz, yz, and xyz. Finally, note that attention shift may be affected by other, contextual, activation.

Note that the erroneous hypothesis is not erased or blocked. It continues to participate in categorization decisions, but it is typically overruled by stronger hypotheses. We assume that new hypotheses are temporarily strongly activated, reflecting the attention applied to them; this helps them overrule older hypotheses.

If a hypothesis is strong (has strong connection strengths), we keep the hypothesis and deal with exceptions by adding an additional hypothesis which corresponds to the exception.

If the exceptional data were in error, the new hypothesis will be erroneous. In this situation, new data will reveal the error: We will eventually derive a contradiction between the (dis)confirmation of Accept or Reject and the guess based on the hypothesis. Since the erroneous hypothesis is based on one example, it is weak, and the contradiction will cause us to shift attention away from it.

## 4.5 Possible Extensions

The air traffic control task may be extended in the test phase to include probabilistic data, additional dimensions, and non-binary dimensions (dimensions with either continuous values or dimensions with greater than two discrete values).

We have seen in Section 4.2 above, that emergent concept formation is well-suited to handling data with errors. We predict that errors in the data will deflect subjects for a time, as they consider false hypotheses, but eventually, the associations formed by correct data will be stronger than the spurious associations derived from the errors.

The associative model should be readily extensible to additional dimensions. We would need to add a fourth hypothesis formation principle, and add the additional dimensions to our attention shifting lists, but there would be no significant changes in the nature of the model.

We have not yet considered how to handle continuous values. We might need a pre-processing function to translate continuous values into a set of discrete values. For multiple, discrete values, the main change in the model would be in the use of complementary hypotheses, since the complement of one feature would become a set of features within that same dimension.

## 4.6 Inducing Dimensions

For the current task, the aircraft dimensions are readily discernible. The subject will presumably be aware of these dimensions because of previous knowledge (e.g., heavy and light are both values for weight), and because of the order of presentation of the input (e.g., the first value listed is either heavy or light, and the second value is either 40,000 or

20,000). We have included the dimensional information in our initial statement of the backcloth. So, for example, there is a node for "ac size" which links to both "heavy" and "light," as well as to the node "ac." Dimensions are used when shifting attention, as described in Section 4.3 above.

While we assume the aircraft dimensions in our implementation here, the DCOG-CL model is capable of inducing such dimensions based on input. We will give an informal explanation of the dimension induction process first, followed by the mathematical definitions, below.

Suppose that the aircraft dimension values were not heavy, light, 40,000, 27,000, moderate turbulence, and light turbulence, but rather {a, b, c, d, e, f}, and furthermore, that the values were not presented in any particular order. Some example input might be:

**Table 34: Aircraft exemplar input values**

| Initial Input | | Additional Input | |
|---|---|---|---|
| ac1 | a,c,e | ac4 | f,c,a |
| ac2 | c,e,b | | |
| ac3 | a,f,d | | |

Our task now is to discover the dimensions. In a well-formed dimension, the set of all possible values forms a partition on the set of exemplars. That is, if dimension D has values {d1, ... dn}, any given exemplar should take on exactly one value within that set. Given the initial input above, we can see that the potential dimensions are {a, b}, {c, d}, {c, f}, {e, d}, and {e,f}.

**Table 35: Co-occurrence of aircraft exemplars and input values for three aircraft**

| | ac1 | ac2 | ac3 |
|---|---|---|---|
| a | x | | x |
| b | | x | |
| c | x | x | |
| d | | | x |
| e | x | x | |
| f | | | x |

Upon receiving the additional input, ac4: {f,c,a}, we can then rule out f and c occurring in the same dimension, and the potential dimensions are reduced to {a,b}, {c,d}, and {e,f}:

**Table 36: Co-occurrence of aircraft exemplars and input values for four aircraft**

| | ac1 | ac2 | ac3 | ac4 |
|---|---|---|---|---|
| a | x | | x | x |
| b | | x | | |
| c | x | x | | x |
| d | | | x | |
| e | x | x | | |
| f | | | x | x |

Here are the specific mechanisms used to induce the dimensions: First, we create an adjacency matrix for the types, Ti, identifying co-occurring types. Any co-occurring types cannot be co-dimensional.

**Table 37: Adjacency matrix for input values**

| | a | b | c | d | e | f |
|---|---|---|---|---|---|---|
| a | | | \|\| | \| | \| | \|\| |
| b | | | \| | | \| | |
| c | \|\| | \| | | | \|\| | \| |
| d | \| | | | | | \| |
| e | \| | \| | \|\| | | | |
| f | \|\| | | \| | \| | | |

The adjacency matrix rules out these combinations: {a,c}, {a,d}, {a,e}, {a,f}, {b,c}, {b,e}, {c,e}, {c,f}, and {d,f}.

Next, we consider the remaining combinations, and identify the set of exemplars connected to the set of types in each combination. A well-formed dimension will assign a type to each exemplar.

**Table 38: Exemplars represented by various input value combinations**

| Combination | Exemplars Represented | | Complete? |
|---|---|---|---|
| a,b | 1,3,4 | 2 | yes |
| b,d | 2 | 3 | no |
| b,f | 2 | 3,4 | no |
| c,d | 1,2,4 | 3 | yes |
| d,e | 3 | 1,2 | no |
| e,f | 1,2 | 3,4 | yes |

Only these dimensions provide a partitioning of the dimensions: {a,b}, {c,d}, and {e,f}.

## 4.7 Other Issues

### 4.7.1 Reasoning with Dimensions

In Type I, The T-accept/T-reject dimension is equivalent to one of the other dimensions. That is, one of the other values dominates the same ac examples as the T-accept type, and the complementary value dominates the same ac examples as the T-reject type.

We assume that people have a way of noticing such equivalent types (i.e., types which subsume exactly the same set of examples).

Knowing you can ignore a dimension: for Type I, we are looking for one dimension, with types T-i and T-$i^C$, which are equivalent to T-accept and T-reject. If T-accept dominates both Xp E T-f and Xq E T-$f^c$, then we know that T-f is not equivalent to T-accept. That is, if both ac with and without feature f are accepted, then feature f cannot be the feature equivalent with accept.

### 4.7.2 Natural Categories

Do subjects go into the task expecting a certain sort of categorization? For example, subjects might look first for Type I explanations of the data. We could represent such pre-conceptions as the initial spreading of activation to certain nodes.

Note that Shepard, Hovland, and Jenkins's classification only contains categories that create equal numbers of category A (here, accept) and category B (here, reject). There are always four good possible exemplars and four bad possible exemplars. In this way, we avoid base rate issues (see Section 2.4.4).

There are other categorizations, some of them more "natural", which are not balanced in this way and hence not included in this task. For example, Type II is exclusive

OR: an ac is good if it has feature f or feature g, but not if it has both feature f and feature g. We do not have a category for inclusive OR, in which the only rejected ac would be those that have neither f nor g. We also do not have a category for which each dimension is specified (e.g., the only accepted ac would be those with high turbulence, low altitude, and small ac size). We do not have a category for the intersection of two features (ac must be both f and g); this would yield two good possible ac, {f, g, z1} and {f, g, z2}.

If preconceptions are important, then there may be some interaction between "natural" categorizations and the Shepard, Hovland, and Jenkins types.

## 5.0 Implementation Design Plan

We plan to use the new memory model in the next implementation of the DCOG framework. Recall the four-level hierarchy depicting different views of agent behavior that emphasized cognitive activity. The Operational Level is the source for the agent implementation plan for our new DCOG model that will perform concept learning in the context of managing an air traffic control sector. The new model, tentatively labeled DCOG-CL (concept learning), is composed of five software agents. The original DCOG-1 actor model was composed of four agents (see Eggleston et al., 2005 for details). A comparison of the agent structure of these related models is shown in Table 39.

**Table 39: Correspondence of agents between DCOG-CL and DCOG-1**

| DCOG-CL | DCOG-1 |
|---|---|
| Radar Agent | Radar Agent |
| Activation Agent | |
| Structure Agent | Schema Agent |
| Recognition Agent | Exec Agent |
| Action Agent | Action Agent |

Three of these agents, Activation, Structure, and Recognition, take the place of DCOG-1's Schema agent. This change reflects the introduction of an associative memory component in DCOG-CL. The recognition agent takes over the functions of the executive agent. While some of the Recognition Agent's decision-making functions are still

implemented procedurally, it is our hope that the associative memory mechanisms will be able to handle these functions in later versions of DCOG.

The following diagram, repeated from Section 3.1, shows the operations that relate the Activation, Structure, and Recognition Agents.

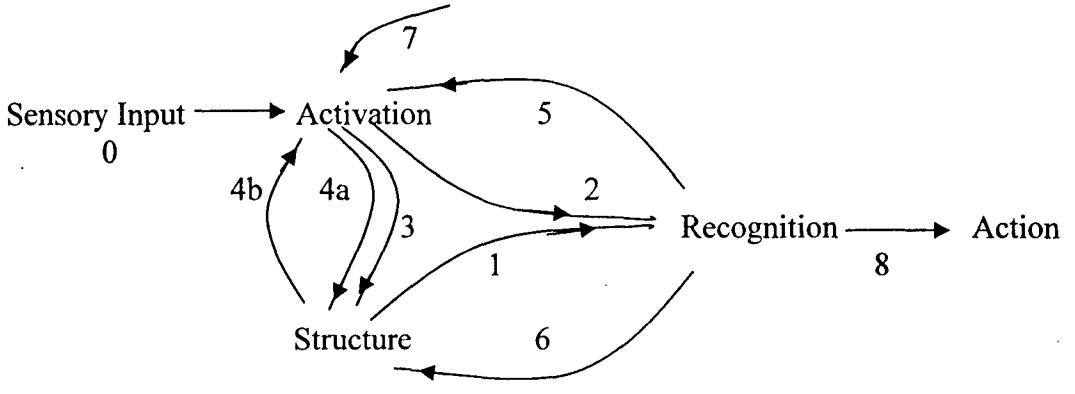



**Figure 31: Operations and their relationship to the Activation, Structure, and Recognition Agents**

The operations are:

**Table 40: Operations**

0. perceptual activation

1. concept identification

2. activation slicing

3. association

4a. firing

4b. spreading activation

5. attention shifting

6. symbol creation

7. decay of activation

8. recognized concept

The basic idea of the system is this: sensory input activates nodes which represent perceptual features (0). (We assume that the input is sufficient to push those nodes over firing thresholds.) Nodes that occur together become associated by links of varying strength (3). Groups of associated nodes represent concepts. Concept recognition follows an algorithm that considers the strength of connections (1) among currently active nodes (2). Recognition of a concept may lead to motor action (8).

Other procedures serve to change activation or structure. Activation is modified by a decay function that lowers activation over time (7). Activation also spreads from node to node along association links (4). A shift in attention can also spread activation (5). Associations are strengthened among co-active nodes (3), as mentioned above. The structure can also be modified by the creation of new nodes that serve as symbols (6). Data structures include a node array, a link array, a concept array, and a hypothesis array. Hypothesis formation is discussed in Section 4.3 above. Synchronization could be implemented by adding a value, firing rate, to each node (see Section 2.8).

**Table 41: Data structures for nodes, links, concepts, and hypotheses**

Node

numerical index
label
domain (symbolic, perceptual, motor)
current activation value (a number)
firing threshold (a number)
delta = current activation – firing threshold

Link

numerical index

origin node

destination node

connection strength (a number)

Concept

numerical index

label

set of nodes

subset of core nodes

Hypothesis

numerical index

symbol (a node in the symbolic domain)

set of source nodes

set of target nodes

list of association strengths for the target nodes,

where association strength = strength of link from symbol to target

The state change model represents activation, association, and conceptual structure within one backcloth. Our implementation, however, distributes this information across three arrays. We must therefore add some messages among the agents to communicate changes in the arrays.

Association (3) is listed as an operation relating activation to structure, changing the structure of the backcloth by forming or strengthening connections between co-active nodes. Our implementation, however, represents structure redundantly, both in the link arrav and in the concept array, which draws upon the information in the link array. After the association prc

ıcedure (3a) has changed the structure of the backcloth, the Structure

Agent must send a changed structure message (3b) to the Recognition Agent, telling it to update the concept array.

Generate Symbol (6) is listed as an operation which proceeds from recognition to structure, reflecting the stable nature of this change. However, when implementing the generate symbol procedure (6), the Recognition Agent must update both the node array and the link array. The message new node (6a) tells the Activation Agent to create a node in the node array, with a certain initial level of activation, while the message new link (6b) tells the Structure Agent to link the new symbol to the relevant nodes.

The revised operation diagram, then, looks like this:

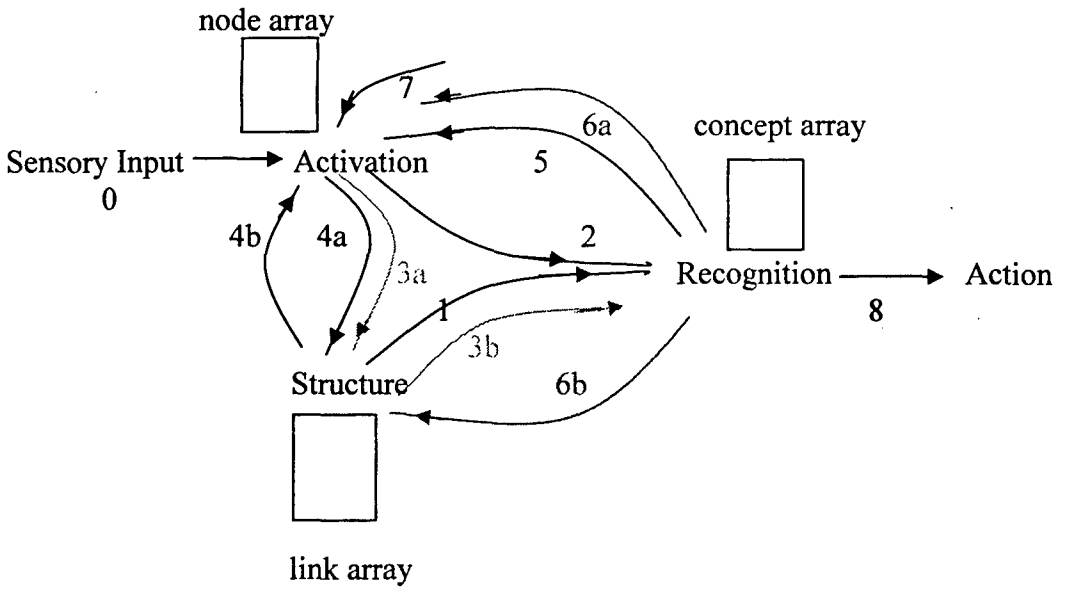



**Figure 32: Operation diagram, revised for implementation**

Here are the responsibilities of the various agents (Table 42). Procedures (listed in bold type) are described below, in Table 43.

**Table 42: Distribution of procedures and messages among agents**

Activation Agent
(0) receives **activate** messages from radar agent (perceptual input)
(4) receives **activate** messages from structure agent (spreading activation)
(5) receives **activate** messages from recognition agent (attention shifting)
(6) receives **new node** message from Recognition Agent
(see note about (6) in Recognition Agent section, below)
**activate** procedure updates node array
**activate** procedure may call **fire** procedure
(4a) **fire** procedure sends **spread activation** message to structure agent
(7) **decay** procedure updates node array [on time cycle t1]
(6) **new node** procedure updates node array, and calls **activate** procedure to activate the new node
Structure Agent
(4a) receives **spread activation** message from Activation Agent
(6) receives **new link** message from Recognition Agent
(3) sends **changed structure** message to Recognition Agent
(4b) **spread activation** procedure uses node array and link array
(4b) **spread activation** procedure sends message to Activation Agent to **activate** a set of nodes
(6) **new link** procedure updates link array
(3) **association** procedure [on time cycle t2]
uses node array and link array, updates link array
sends **changed structure** message to Recognition agent
Recognition Agent
(3) receives **changed structure** message from Structure Agent
**latent concept identification** procedure [called after (3) association, via **changed structure** message]
1. use concept id algorithm to identify latent concepts, based on the structure of associations
2. update concept array
    a. if the concept is new (see below), then create a new concept in the concept array.
    b. if the concept is not new, then update the values for the set of nodes and/or the set of core nodes for that concept in the concept array.

A latent concept is considered new if more than z percent of the nodes in the identified concept are not found together in any existing concept core in the array. (z is some constant, perhaps variable from subject to subject. A high level of z would indicate a tendency to lump new information in with old; a low level of z would suggest a tendency to proliferate categories.)
**concept recognition** procedure [on time cycle t3]
1. **slicing** for active nodes (2)
2. compare the set of active nodes to the core nodes in each concept in the concept array;
3. send message **recognized concept** to Action Agent, as appropriate (8)
4. run **executive** procedure

**executive** procedure uses concept array and node array
**executive** procedure determines when to call **generate symbol** and **shift attention**
(5) **shift attention** sends **activate** message to Activation Agent

Symbol creation (6) is indicated on the diagram as affecting the Structure agent, reflecting the stable nature of this operation. However, symbol creation involves both creating a new node and creating new links, so this procedure actually involves both the Activation agent and the Structure agent.

(6) **generate symbol** sends **new node** message to Activation Agent
**(6)** generate symbol **sends** new link **message(s) to Structure Agent**
Action Agent
(8) receives **recognized concept** message from Recognition Agent
(8) **recognized concept** message triggers motor action, if appropriate

**Table 43: Procedure Specifications**

**0.0 Activate (node n by amount m) Procedure**
look up node n
current activation of node n = current activation + m
delta = current activation of n – firing threshold of n
if delta ≥ 0, call fire procedure for node n

**1.0 Concept ID Algorithm Procedure**
This is similar to slicing, only instead of setting a parameter for activation energy; we set a parameter for association strength. This allows us to pick out sets of highly connected nodes. (In general, the higher the parameter, the greater the number of disconnected pieces.)
Here we specify Concept ID for immediate links (i.e., chains of length 1). It is also possible to specify this algorithm with a second parameter, d, allowing chains of length d.
**Concept ID** with association strength parameter p
**Slice** (2) to find the set of active nodes. Call this set A.

Take a set of nodes, A. For each node $n_i \in A$, find the set of nodes which share a link with $n_i$, with the connection strength of the link ≥ p. Call this set $Rp(n_i)$, the p-range of $n_i$.

Start with the $n_i \in A$ which has the largest number of elements in its p-range. Call it $n_1$.

(If there are two $n_i$ with the same number of elements in their p-ranges, arbitrarily choose one to consider first.)
Let $T_1 = Rp(n_1) \cup n_1$ (this is the start of the first concept)
Take the next smallest p-range, call its source node $n_2$.

If $T_1 \cap Rp(n_2) = \varnothing$,
else let $T_1 = T_1 \cup Rp(n_2) \cup n_2$. (this adds the new nodes to the first concept)

Continue checking the p-range of each $n_i \in A$. The $T_i$ are the distinct concepts recognized at strength p.

**2.0 Slicing for Active Nodes Procedure**

There are two sub-procedures, slicing for active nodes and absolute slicing.

*2.1 Slicing for Active Nodes Sub procedure:*

using node array, find the set of nodes for which delta $\geq 0$.

*2.2 Absolute Slicing Sub procedure:*

with slicing parameter s

using node array, find the set of nodes for which current activation $\geq$ s.

**3.0 Association Procedure**

This is an automatic, time-governed procedure. To be determined: What time cycle should we use? Does the same time cycle govern association, decay, and concept recognition?

*3.1 Association (link strengthening) Sub procedure:*

**Slice** for active nodes (2)

For each active node $n_1$, find the set of co-active nodes.

For each co-active node $n_2$, find the link $n_1$-$n_2$ in the link array, and increment its connection strength by z, where z is some constant.

(If we set z to ½, then, since each link will be updated twice, from each end, we will end up strengthening each link by 1 total. If we set z to 1, then, to strengthen each link by 1, we will have to write this procedure to avoid redundant access to the links which have already been processed.)

*3.2 Association (change structure) Sub procedure:*

send message **changed structure** to Recognition Agent

3.2.1 **Fire** node n:

send message to Structure Agent to **spread activation** from node n.

(Eventually, we may include here as well the instruction to reduce the activation of a node after it fires: activate node n by amount –f, where f is some constant. We are not implementing this at this time.)

3.2.2 **Spreading Activation** from source node n:

Activation spreads from a source node to its neighbor nodes, proportionally to the strength of connection between the neighbor and the source.

Spreading Activation

Look up the set of all links which contain node n

For each link in this set,

**activate** the second node in the link by amount m*connection strength of link, where m = some constant (probably 1).

3.2.3 **Shift Attention** to node n:

**Activate** node n by amount x, where x is some constant

The amount of x is determined by the **Executive** procedure, and could be varied to reflect a subject's strategy.

**4.0 Generate Symbol Procedure**

*4.1 New Node, label "L":*

Create a new node, with label L

**activate** node L by amount y, where y is some constant.

*4.2 New Link, origin node, destination node, strength p:*

Create a link of strength p between the origin node and the destination node. This procedure is used by generate symbol as follows:

Given a symbol node, s, and a set of concept nodes, {c1, ... cn}, create links between s and each ci, at association strength m, where m is some constant (probably 1).

**5.0 Decay Procedure**

Decay is an automatic, time-governed procedure, time cycle to be determined.

Decay:

For each node in the node array, **activate** n by –d, where d is some constant.

**6.0 New Concept Procedure**

This procedure might eventually be implemented by an array of concepts and associated actions, to be maintained and used by the Action Agent.

Ultimately, the triggering of motor action should be done by coupling symbolic and perceptual concepts with concepts in the motor domain. We may implement motor action procedurally for now (e.g., if the concept Accept is recognized, then the Action Agent pushes the accept button).

## 6.0 SUMMARY

This report addresses the second phase in the development of our DCOG framework for use in the design and implementation of intelligent software actors that can perform complex work. It has concentrated on the theoretical foundations for an associative memory model that can be used in a distributed control and knowledge storage software architecture, like the one we developed for the DCOG-1 actor. This memory model is patterned after, in broad strokes, known knowledge about the neurophysiology of the human brain. It emulates neural level communication (feature activation and associational connection strength) and cell assembly formation, functionality, and communication (synchronization, emergent type space, emergent concept space) derived from a lower-order feature space. We have provided a careful and detailed description of how type and concept spaces can emerge from a feature set using the concept of co-occurrence. This provides an elegant way by which knowledge can be grown by a cognitive actor while being constrained by fixed size limits of a feature space.

To add further insight into this type of memory system, we examined it in the context of a challenging cognitive work problem: the classic Shepard, Hovland, and Jenkins concept learning task. This examination illustrated how feature-based emergent knowledge can potentially co-exist and interoperate with symbol (concept) level knowledge to provide an intelligent actor flexible and adaptable methods for achieving concept learning. Thus, it illustrates how a common structural and process core can be the foundation for both a symbolic and nonsymbolic memory system. A cognitive actor may have a single memory system of this form or a network of local memory systems following this pattern.

We have also demonstrated how the DCOG memory model can be implemented in software. The discussion included a description of important theoretical constructs, how they could be represented algorithmically, and how they could be formed into an operational architecture using conventional features of a modern software language. The implementation plan developed here serves as the departure point for the design and construction of our next DCOG software actor. This actor will be tasked to perform an enroute air traffic management task while simultaneously being engaged in solving an embedded concept learning task. This is a challenging work environment for a software actor. An empirical investigation of the performance of the DCOG-CL actor will be documented in a future report.

# 7.0 REFERENCES


Anderson, J.R. and Labiere, C. (1998). *The Atomic Components of Thought.* Mahwah, NJ: Lawrence Erlbaum Associates.

Atkin, R. H. (1974). *Mathematical Structure in Human Affairs*. London: Heinemann Educational Books.

Barsalou, L.W. (1989). Intra-concept similarity and its implications for inter-concept similarity. In S. Vosniadou & A. Ortony (Eds.), *Similarity and Analogical Reasoning* (pp. 76-121). New York: Cambridge University Press.

Eggleston, R.G., McCreight, K.L. and Young, M. (2005). Distributed Cognition and Situated Behavior. To appear in Pew, R.W. and Gluck, K.A. (Eds.) *Modeling Human Behavior with Integrated Cognitive Architectures: Comparison, Evaluation, and Validation.* Mahwah, NJ: Lawrence Erlbaum Associates.

Eggleston, R.G., Young, M.J. and McCreight, K.L. (2001). Modeling Human Work through Distributed Cognition. *Proceedings of the Computer Generated Forces Conference*, Orlando, FL, Summer, 2001.

Eggleston, R.G., Young, M.J. and McCreight, K.L. (2000). Distributed Cognition: A New Type of Human Performance Model. *Proceedings of the AAAI Fall Symposium on Simulating Human Agents*, 3-5 Nov 2000.

Kruschke, John K. (1992). ALCOVE: An exemplar-based connectionist model of category learning. *Psychological Review*, Vol. 99, no.1, 22-44.

Rosch, E. (1978). Principles of Categorization. In E. Rosch & B. Lloyd (Eds.), *Cognition and Categorization* (pp. 27-48). Hillsdale, NJ: Erlbaum.

Schank, R. C. (1982/1990). Dynamic Memory: A theory of reminding and learning in computers and people. Cambridge: Cambridge University Press.

Shepard, R. N., Hovland, C.I. and Jenkins, H. M. (1961). Learning and memorization of classifications. *Psychological Monographs,* 75 (13, Whole No. 517).

Tenney, Y.J. and Spector, S.L. (2001). Comparisons of HBR models with human-in-the-loop performance in a simplified air traffic control simulation with and without HLA protocols: Task simulation, human data and results. *Proceedings of the Tenth Conference on Computer Generated Forces and Behavioral Representation*, 10, 15-26.

Young, M. J. (1998). Computational modeling of memory: The role of long-term potentiation and Hebbian synaptic modification in implicit and schema memory. *Dissertation Abstracts International*, Vol. 59/03-B.

## Appendix A: Additional Mathematics

The latent concept identification algorithm can be generalized to include chains (links of distance greater than one). Here is the general statement:

Take two nodes, ei and ej. (A1)

s(ei, ej) = strength of connection between ei and ej.

s(ei, ej) = 0 if ei and ej are not connected.

s(ei, ej) > 0 if ei and ej are connected.

Assume a set of points, E, with various connections of different strengths. Take ei as a starting point. We define a set of points, Lp,d(ei) = all points connected to ei by a link of strength $\geq$ p and distance $\leq$d. Start with d=1.

$$Lp,1(ei) = \{ej \mid s(ei,ej) \geq p\} \quad \text{(A2)}$$

$$Lp,2(ei) = Lp,1(ei) \cup Lp,1(ek) \text{ for each } ek \in Lp,1(ei)$$

This is the set of immediate links of ei (d=1), plus the immediate links of each of those neighbors (d=2). Note that this will include the original node, ei, by going out and back; we may want to explicitly exclude the original node, ei. In general,

$$Lp,n(ei) = Lp,n-1(ei) \cup Lp,1(ek) \text{ for each } ek \in Lp,n-1(ei) \quad \text{(A3)}$$

Distance Ds(ei, ej) is defined as the number of iterations required to link ei and ej at some strength s, $s \geq 1$.

If ei = ej, then Ds (ei, ej) = 0, for any value of s (A4)

If $ej \in Ls,1(ei)$, then Ds(ei, ej) = 1

if $ej \in Ls,2(ei)$, then Ds(ei, ej) = 2

if $ej \in Ls,n(ei)$, then Ds(ei, ej) = n

If we further define

Lp,0(ei) = ei (A5)

then we can state the following generalization:

Ds(ei,ej) = min d such that $ej \in Lp,d(ei)$ (A6)

Based on this, we could have distance varying with strength. Two nodes might be connected by a direct but weak connection when p=1, but also connected by a stronger but roundabout connection going through several nodes when p>1.

For example, consider the network of associations discussed in Section 2.3., Figure 1, repeated here as Figure A1.

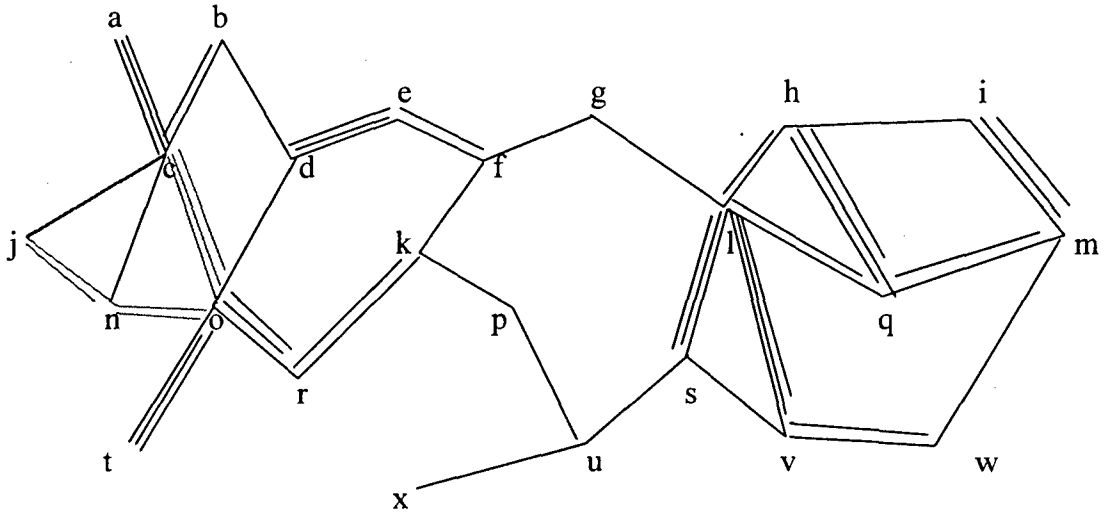



**Figure A1: Distance between nodes depends on required level of association strength. The red line represents the short path between j and c at s=1. The blue line represents the longer path required to link j and c at s ≥ 2**

The distance between the nodes c and j varies, depending on the level of connection strength required. At connection strength s=1, the distance between c and j is 1, as illustrated in red, above.

$$j \in L1,1\,(c) \qquad (A7)$$

$$D1(c, j) = 1$$

At a stronger connection strength, the distance between c and j is greater. To form a path with links of strength ≥ 2, we must travel through two other nodes, n and o. This path is illustrated in blue, above.

$$L2,1(c) = \{a,b,o\} \tag{A8}$$

$$L2,2(c) = L2,1(a) \cup L2,1(b) \cup L2,1(o) \tag{A9}$$

$$= \{c\} \cup \{c\} \cup \{c,r,t,n\} = \{c,r,t,n\}$$

$$L2,3(c) = L2,1(c) \cup L2,1(r) \cup L2,1(t) \cup L2,1(n) \tag{A10}$$

$$= \{a,b\} \cup \{k,o\} \cup \{o\} \cup \{j,o\} = \{a,b,j,k,o\}$$

$$j \in L2,3(c) \tag{A11}$$

$$D2(c,j)=3$$

The **range** R1(ei) is the set of all elements ej, such that there is some connection, with strength $\geq 1$, between ei and ej, at distance d=1. The **p-range** of ei, Rp(ei), is the set of all elements ej, such that there is a connection between ei and ej, at distance d=1, and strength $\geq p$.

$$R1(ei) = L1,1(ei) = \{ ej \mid s(ei, ej) \geq 1\} \text{ "the range of ei"} \tag{A12}$$

$$Rp(ei) = Lp,1(ei) = \{ej \mid s(ei, ej) \geq p\} \quad \text{"the p-range of ei"} \tag{A13}$$

Take some local set of nodes, T={e1, e2, ... en}. Let A(T) represent the currently

ei $\in$ T, yields the connections for ei within T above strength p. We call this the p-range of ei in A(T).

$T = \{e1, e2, \ldots en\}$ (A14)

$A(T) = \{ei \in T \mid ei \text{ is active}\}$

$Rp(ei) \cap T$ "The p-range of ei in T"

The main idea is to set p to some slicing value to identify the core of T. But, we have to sum across ei $\in$ T, not just find connections for one ei, because there may be more than one group of nodes in T that are connected at strength p. The following method provides a way to slice for such "islands" across a set of nodes. We calculate the p-range for each node in T; then we eliminate any nodes with an empty p-range. These nodes are disconnected at strength p.

For p = some slicing value, take

$Cp(T) = \{ei \in T \mid Rp(ej) \cap T \neq \varnothing\}$ (A15)

This is the "p-core of T." That is, all ei in T such that ei is connected to some other ej in T with strength greater than or equal to p, at a distance less than or equal to 1.

The algorithm for identifying the separate cores is the same as that for latent concept identification (Section 2.3 (1)), repeated here as (A16).

Concept Identification Algorithm (A16)

Take a set of nodes, A. Select a strength parameter, p.

For each node $n_i \in A$, find the set of nodes which share a link with $n_i$, with the connection strength of the link $\geq p$. Call this set $Rp(n_i)$, the p-range of $n_i$.

Start with the $n_i \in A$ which has the largest number of elements in its p-range. Call it $n_1$.

(If there are two $n_i$ with the same number of elements in their p-ranges, arbitrarily choose one to consider first.)

Let $T_1 = Rp(n_1) \cup n_1$ (this is the start of the first concept)

Take the next smallest p-range, call its source node $n_2$.

If $T_1 \cap Rp(n_2) = \varnothing$, then let $T_2 = Rp(n2) \cup n_2$ (this creates a second concept)

else let $T_1 = T_1 \cup Rp(n_2) \cup n_2$. (this adds the new nodes to the first concept)

Continue checking the p-range of each $n_i \in A$. The $T_i$ are the distinct concepts identified at strength p.

# GLOSSARY

access node A node which is useful in activating the core of a concept.

activation One of the three main components of the Operational Level of the abstraction hierarchy. Activation deals with temporary changes in activation energy.

activation slicing An operation providing information about activation to the Recognition agent; activation slicing selects a set of nodes which have activation energy above a slicing parameter.

adjacency matrix Following Atkin (1974), an adjacency matrix is used to represent co-occurrence of features. The same feature set is used on each axis; the number in the cell (a,b) represents the rate of co-occurrence for the features a and b.

analytic mode At the Conscious Level of the abstraction hierarchy, the analytic mode is used when subjects use additional processing in addition to perceiving, recognizing, and acting. Additional processes include creating and manipulating symbols, and shifting attention.

association The creation or strengthening of links among co-active nodes. An operation that modifies structure based on activation.

attention shifting Directing activation energy to a particular node or set of nodes. An operation in which the Recognition agent tells the Activation agent to change activation levels.

backcloth A set of nodes connected by association links of varying strengths. See Atkin

base-rate neglect Different target concepts may occur at different base rates within a sample of exemplars. This can lead to confusion in human subjects, who may neglect to take base rates into account in making classifications.

basic level A level of abstraction at which objects are most usefully categorized, with exemplars having many shared features within a category, and many distinguishing features across categories. See Rosch (1978).

category This term relates to both concepts and types. A category is a type: one member of a dimension which partitions the set of exemplars under consideration. When we categorize, we recognize an exemplar as an example of a type. When we recognize a concept, we may also be categorizing, selecting one concept out of a set of possible concepts that form a dimension.

central exemplar An exemplar from one of Types I-VI which follows the main generalization for the category. See Kruschke (1992).

co-active nodes Two nodes are considered co-active if they have both exceeded their activation thresholds and they are firing in synchrony. When two nodes are co-active, the connection between them is strengthened by a process of association.

complementary hypothesis Given a hypothesis T(F), where T is Accept or Reject, and F is a set of exemplars in x, y, and z, the complementary hypothesis associates $T^c$ with the complementary set of exemplars, $F^c$.

concept A group of tightly connected nodes that reflect the common elements of a set of exemplars, together with the less frequent nodes found in those exemplars. See also category.

concept drift The structure of a concept may change over time, as new exemplars create stronger associations to nodes not previously found in the core of the concept.

concept identification An operation providing information about structure to the Recognition agent; groups of strongly connected nodes are identified as latent concepts.

concept recognition A concept is recognized when the core nodes of the concept are activated, either directly or indirectly, and firing in synchrony.

confirmation, disconfirmation The information about an aircraft's Accept or Reject status provided by the system after the subject has classified the aircraft as Accept or Reject.

conflict resolution If more than one hypothesis is applicable to the input, the conflict will be resolved in favor of the most specific hypothesis.

connection strength The strength of an association link between two nodes. Each co-occurrence of two nodes increases the connection strength between them.

conscious level Second level in the abstraction hierarchy describing the DCOG-CL model; the thoughts of which a subject is consciously aware.

context, contextual activation Activation energy in nodes other than the node or set of nodes under consideration.

core The core of a concept is a tightly connected set of nodes, mathematically identified by selecting a parameter for connection strength. The core represents the area of greatest overlap among the exemplars of the concept.

coupling Connecting two concepts through synchronization of firing. Coupling may be caused by linking symbols from two concepts.

decay of activation The gradual lowering of activation energy. (1.2.) An operation in which the Activation agent reduces activation energy for all active nodes according to a time cycle.

differential association The fact that, given different contextual information, we find apparent changes in association strengths among nodes. See layers.

dimension A set of types which partitions the set of examples of a concept.
Each pair of aircraft values constitutes a dimension (e.g., moderate turbulence and no turbulence are the two values or types of the turbulence dimension). The categories Accept and Reject also form a dimension, sometimes represented as types of the target dimension, T and $T^c$.

direct activation Activation of a node by perceptual input.

disconfirmation See confirmation.

domain A group of nodes; each node belongs to only one domain. We assume that the domains include a symbolic domain, a perceptual domain, and a general cognitive domain. Concepts may form across domains. When discussing words, we may use the terms verbal and non-verbal to distinguish the symbolic and general cognitive domains.

embedded level Highest level in the abstraction hierarchy describing the DCOG-CL model; the level embedded in the environment.

emergent concept formation Over time, the perception of similar exemplars builds up a structure of strongly connected nodes. This structure represents a concept.

example Perceptual input activates a set of nodes; this input is recognized as an example of a concept if the input directly or indirectly activates the core nodes of the concept (i.e., the concept is recognized).

exceptional exemplar An exemplar from one of Types I-VI which violates the main generalization for the category. See Kruschke (1992).

execution level Lowest level in the abstraction hierarchy describing the DCOG-CL model; the level at which we specify the execution of the operations.

explicit concept formation The creation or modification of a concept structure through association with a symbol, including the coupling of two concepts through the association of their respective symbolic nodes.

failure-driven memory Term used to express the observation of Schank (1982/1990) that we remember things which fail to meet our expectations.

firing When a node fires, it spreads activation to its neighboring nodes, in proportion to the strength of connection between them. An operation which sends information to the Structure agent, based on activation.

firing threshold The level of activation energy needed for a node to become active, or fire.

full specification The claim that, in calculating workload, enough hypotheses must be present so that each potential exemplar in x, y, and z can be assigned to a category.

generation The creation and possible subsequent manipulation of symbols. Generation is used to form stable links between separate concepts, which cannot be directly associated because they have no unitary representation.

hypothesis formation The creation of a symbolic node, linked to source nodes in the x, y, and z dimensions, and also linked to a target node, Accept or Reject. The hypothesis symbol serves to retain information about the co-occurrence of certain x, y, or z values with Accept or Reject.

hypothesis rejection A hypothesis which fails to make accurate predictions will be rejected: The original hypothesis becomes associated with both Accept and Reject, and attention shifts to a new dimension, facilitating formation of a better hypothesis.

indirect activation Activation of a node by spreading activation, either from perceptual input or from previously active contextual nodes.

induction of dimensions The process of determining which types can be combined into well-formed dimensions.

input-association chart A chart with feature values for the x, y, and z dimensions across the top, and rows for the number of exemplars associated with Accept and Reject for each of those columns. Summing across the input-association chart gives a ratio that shows the information about Accept and Reject that is provided by emergent concept formation for a given set of input features.

intuitive mode At the Conscious Level of the abstraction hierarchy, the intuitive mode is used when subjects simply perceive, recognize, and act.

latent concept identification Looking at structure in isolation from current activation energy, we can identify latent concepts, or structures with strongly connected cores.

layers A representation of different sets of association strengths for one set of nodes. Layers can be simulated through contextual activation.

octagonal association diagram A diagram with eight nodes, representing the values of x, $x^c$, y, $y^c$, z, $z^c$, Accept, and Reject. Each exemplar aircraft in the air traffic control task can be represented as a set of associations on these eight nodes.

OODA Loop The sequence of a subject's actions: Observe, Orient, Decide, Act.

operational level Third level in the abstraction hierarchy describing the DCOG-CL model; the level at which we describe the relationships among the operations.

perceptual activation An operation providing information to the Activation agent; features perceived in the environment activate nodes representing those features.

peripheral exemplar An exemplar from one of Types I-VI which shares some values with the central exemplar(s). See Kruschke (1992).

recognition One of the three main components of the Operational Level of the abstraction hierarchy. Recognition deals with the recognition of concepts.

recognized concept An operation in which the Recognition agent passes concept information to the Action agent.

sign Sensory input which triggers further processing within the sensory domain.

signal A strong link between sensory input and the motor domain.

source node See hypothesis formation.

specificity A feature is specific to a concept if activating the feature tends to activate that concept more than any other concept.

spreading activation When a node exceeds its firing threshold and becomes active, it spreads activation energy to other nodes, in proportion to the connection strength of the links to those nodes. An operation which tells the Activation agent to change activation levels of certain nodes, based on structure.

structure One of the three main components of the Operational Level of the abstraction hierarchy. Structure deals with stable or long-term changes in the associations among nodes.

symbol (first sense) A link between sensory input and verbal/symbolic processing.

symbol (second sense) A node in the verbal/symbolic domain. A symbol may be associated with a concept, but does not have a privileged relationship with respect to that concept.

symbol creation An operation in which the Recognition agent tells the Structure agent to create a new node, representing a symbol, and associate it with certain other nodes.

symbolic concept A structure of strongly connected nodes, all occurring within the symbolic domain. A symbolic concept can represent relationships among concepts which are associated with the various symbolic nodes.

synchronization Synchronized nodes fire at the same rate. Two concepts which are synchronized form a thread.

target node See hypothesis formation.

thread A group of two or more concepts, taken as a single entity.

type A type of a concept is an example of that concept, such that the examples of the type are all examples of the concept.

Type I, Type II, etc. Categories defined on four two-valued dimensions, as defined by Shepard, Hovland, and Jenkins (1961), and represented by a cube with x, y, and z axes.

typicality A feature is typical of a concept if it occurs in most exemplars of that concept; activating the concept tends to activate the typical features.

work layer A description of the backcloth when there is strong contextual activation related to some task; the Work Layer enables recognition within a particular task.

world layer A description of the backcloth when the activation is diffuse; the World Layer enables flexible thought.