International Journal of Neural Systems, Vol. 7, No. 3 (July, 1996) 305–319
© World Scientific Publishing Company

CONCEPT HIERARCHY MEMORY MODEL: A NEURAL ARCHITECTURE FOR CONCEPTUAL KNOWLEDGE REPRESENTATION, LEARNING, AND COMMONSENSE REASONING

AH-HWEE TAN
RWCP Neuro ISS Laboratory*[†]
Heng Mui Keng Terrace, Kent Ridge, Singapore 119597
E-mail: ahhwee@iss.nus.sg

HUI-SHIN VIVIEN SOON
Center for Computer Studies,
Ngee Ann Polytechnic Institute,
535 Clementi Rd, Singapore 599489
E-mail: shs@titan.np.ac.sg

Received 7 September 1995
Revised 8 March 1996
Accepted 2 April 1996

This article introduces a neural network based cognitive architecture termed Concept Hierarchy Memory Model (CHMM) for conceptual knowledge representation and commonsense reasoning. CHMM is composed of two subnetworks: a Concept Formation Network (CFN), that acquires concepts based on their sensory representations; and a Concept Hierarchy Network (CHN), that encodes hierarchical relationships between concepts. Based on Adaptive Resonance Associative Map (ARAM), a supervised Adaptive Resonance Theory (ART) model, CHMM provides a systematic treatment for concept formation and organization of a concept hierarchy. Specifically, a concept can be learned by sampling activities across multiple sensory fields. By chunking relations between concepts as cognitive codes, a concept hierarchy can be learned/modified through experience. Also, fuzzy relations between concepts can now be represented in terms of the weights on the links connecting them. Using a unified inferencing mechanism based on code firing, CHMM performs an important class of commonsense reasoning, including concept recognition and property inheritance.

1. Introduction

Commonsense reasoning is used in the daily life of a human being. Some examples of commonsense knowledge are "snow is white", "bird flies", "elephant is gray", and "glass can break". Commonsense reasoning plays an important role in most types of intelligent activities such as language understanding, planning, pattern recognition, and expert reasoning. It is thus essential for an autonomous intelligent agent to function in a real life environment.

Modeling commonsense reasoning is a difficult task, as it involves many subtle modes of reasoning and a vast body of knowledge of different domains with complex interactions. The problem is to derive powerful representation schemes that support a sufficiently rich class of commonsense reasoning. A key element of human commonsense knowledge is *object concept.* In fact, a significant

*Real World Computing Partnership
[†]Institute of Systems Science, National University of Singapore

portion of commonsense knowledge involves simple relationships between object concepts. For example, "snow is white" is part of the definition of "snow". A useful representation scheme for organizing concepts is *concept hierarchy.* In a concept hierarchy, the meaning of a concept is built on a small number of simpler concepts, each in turn is defined at a lower level using other concepts. Concept hierarchy facilitates an important class of commonsense reasoning, including *inheritance* and *recognition.* Inheritance is the form of reasoning whereby the properties of a class (concept) are inferred from the properties of its super-classes (defining concepts). For example, given that "robin is a bird" and "bird flies", it can be inferred by inheritance that "robin flies". Inheritance is particularly useful in knowledge abstraction as it eliminates the redundancy of storing the same properties along the inheritance path. Recognition is the dual process whereby a concept is identified based on its property values. For example, given that "x flies" and "x lays eggs", x can be recognized as a bird.

Concept hierarchy has been typically represented in the form of *property inheritance graphs*, or in a more general form, *semantic networks.*[1,2] One major limitation of semantic networks is that crisp (non-fuzzy) relationships are used to represent relations between concepts, as well as between a concept and its properties. This results in rigid reasoning that is not suitable for processing commonsense knowledge. Moreover, besides simply adding/deleting links of networks, most semantic networks and inheritance systems, even in connectionist versions,[3,4] do not include any effective learning mechanism. One of the main difficulties in learning as noted by Feldman[5] is to create new concepts and new memory structures dynamically. The problem is aggravated by the slow learning nature of most neural network algorithms.

The Concept Hierarchy Memory Model (CHMM), presented in this paper, addresses the discrete representation and the learning problems neglected in other conceptual knowledge-based systems. The CHMM architecture is composed of two sub-networks: a Concept Formation Network (CFN), that acquires concepts based on their sensory representations; and a Concept Hierarchy Network (CHN), that encodes hierarchical relationships between concepts.[6–8] Using a unified inferencing mechanism, CHN performs an important class of commonsense reasoning including concept recognition, whereby a concept is identified based on its property values; and property inheritance, whereby the properties of a concept are implied by its lower level defining concepts. As suggested by the term *concept* hierarchy, the approach adopted here is an intensional one rather than extensional. Specifically, while most commonsense reasoning systems perform inheritance on classes of objects that fit into various concepts, CHN organizes a concept hierarchy and performs inheritance based on the meanings or semantics of concepts.

The remaining sections of this paper are organized as follows: Section 2 presents the Concept Hierarchy Memory Model architecture. Section 3 describes the concept learning algorithm of the Concept Formation Network. Section 4 introduces the concept hierarchy representation and presents the learning and inferencing algorithms of the Concept Hierarchy Network. Section 5 shows how commonsense reasoning, including recognition, property inheritance, and cancellation of inheritance, can be achieved in a concept hierarchy network through a unified inference mechanism. The final section provides some concluding remarks and discusses possible extensions.

2. Concept Hierarchy Memory Model

The Concept Hierarchy Memory Model (CHMM) is built upon Adaptive Resonance Associative Map (ARAM), a supervised Adaptive Resonance Theory (ART) neural network that performs rapid yet stable hetero-associative learning in a real-time environment.[9–11] ARAM performs two slightly different memory tasks, namely pattern classification and hetero-associative recall. While ARAM is simpler in architecture than another class of supervised ART systems — ARTMAP,[12,13] it is functionally equivalent to ARTMAP in pattern classification under certain parameter settings. Besides the fast and incremental advantages, empirical experiments have shown that ARAM pattern recognition performance is comparable, if not superior, to alternative methods, including counterpropagation and backpropagation neural networks.[11] As ARAM network structure and operations are symmetrical, associative recall can be performed in both directions. By encoding pattern pairs explicitly as cognitive chunks, ARAM guarantees perfect storage and recall of an *arbitrary* number of *arbitrary* pattern pairs. More importantly, it also exhibits a much higher recall accuracy than Bidirectional Associative Memory (BAM) models using the original[14] and improved learning rules.[15–17] By its symmetrical structure, ARAM generalizes readily to multi-channel ARAM that learns associations across multiple pattern channels. Whereas ARAM consists of two input representation fields sharing a category field, multi-channel ARAM comprises multiple input representation fields and a category field. Based on multi-channel ARAM that supports fast and stable associative learning, CHMM is designed to provide a systematic way for creating new concepts and organizing a concept hierarchy.

The Concept Hierarchy Memory Model (CHMM) is composed of a Concept Formation Network (CFN) and a Concept Hierarchy Network (CHN) [Fig. 1]. CFN is a 3-channel ARAM consisting of a concept field F_2^c and three sensory fields, identified as visual memory field F_1^{c1}, auditory memory field F_1^{c2}, and verbal memory field F_1^{c3}. The concept field F_2^c uses one node to represent a concept. The activity value of a node indicates the degree of activation of the corresponding concept. The F_1^{c1}, F_1^{c2}, and F_1^{c3} activities form the distributed representations of concepts in the sensory fields. These activities function on a short time scale in the sense that they respond spontaneously to external inputs. The F_2^c activities function on a medium time scale in that they remain active for a brief period after activation and that more than one concept can be active at one time. Each sensory field is connected to the concept field by bidirectional conditionable links that allow learning and inexact match inferencing. Using the multi-channel ARAM learning algorithm, distinct activity patterns across F_1^{c1}, F_1^{c2}, and F_1^{c3} are self-organized into meaningful categories (concepts) in F_2^c.

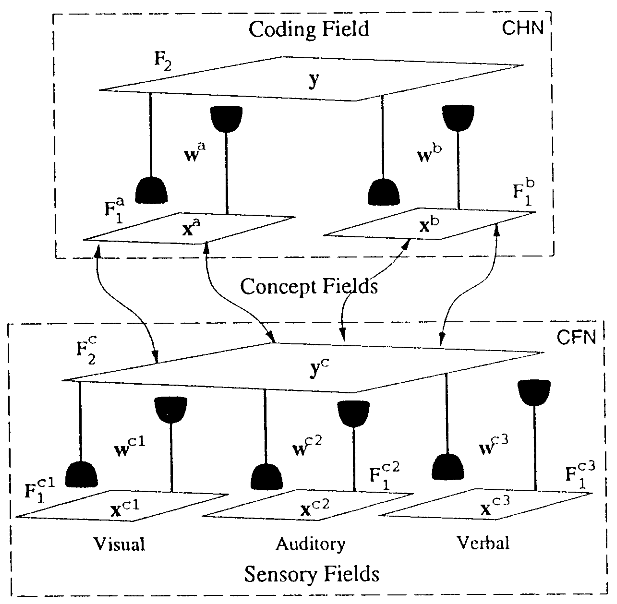


Fig. 1. The Concept Hierarchy Memory Model architecture.

The Concept Hierarchy Network (CHN) is an (2-channel) ARAM consisting of a coding field F_2 and two copies of the concept field F_1^a and F_1^b. Each coding node in F_2 learns a relation between a concept in F_1^b and its lower level defining concepts in F_1^a. Both F_1^a and F_1^b are connected to F_2 by bidirectional conditionable links, and are used for matching condition of code firing and for readout of code activation. Using a code firing procedure that propagates activities from concepts to concepts, CHN performs concept recognition and basic forms of property inheritance including top-down and bottom-up inheritance. By organizing sets of conflicting concepts into competitive fields, CHN resolves conflicting situations including *exceptions* and *conflicting multiple inheritance.*

3. Concept Formation Network

A concept is usually learned by sampling the objects that fit the concept and applying a certain kind of template-based learning to acquire the concept's meanings or semantics. For example, the concept *elephant* can be acquired by seeing many instances of "elephant", hearing many spoken words of "elephant", and/or seeing many written words of "elephant". Ultimately, a concept learning system should be able to form compact sensory representations of concepts across different modalities.

In the Concept Formation Network (CFN), a concept node in the concept field F_2^c can be recruited or activated by the signals from one or more of the sensory fields F_1^{c1}, F_1^{c2}, and F_1^{c3} [Fig. 2]. For example, the concept *elephant* can be activated by seeing an elephant, by hearing a pronounced word of "elephant", and/or by reading a word "elephant". Initially, all nodes in F_2^c are uncommitted. The nodes become committed one at a time when novel patterns occur across F_1^{c1}, F_1^{c2}, and F_1^{c3}.

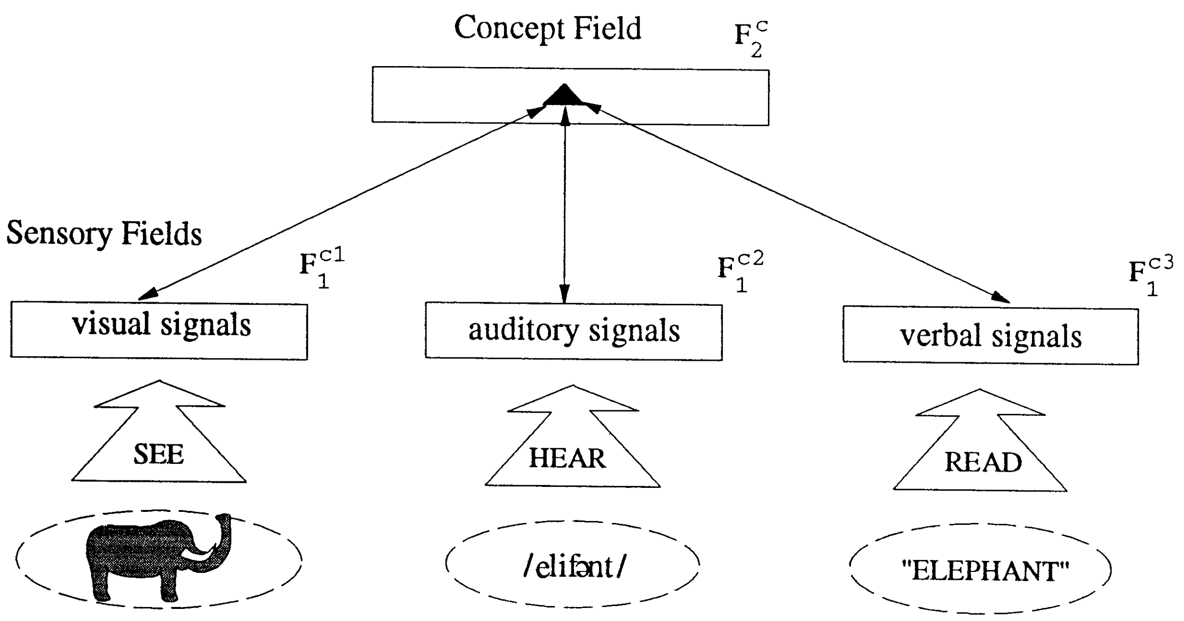


Fig. 2. Concept Formation Network: Distributed activities in the sensory fields F_1^{c1}, F_1^{c2}, and F_1^{c3} are self-organized into categories (concepts) in the concept field F_2^c.

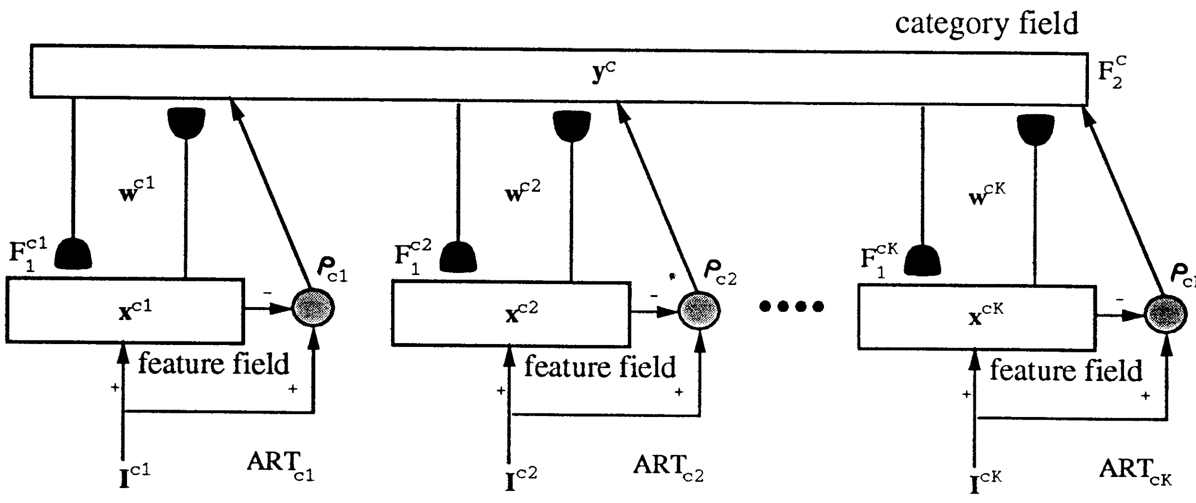


Fig. 3. The multi-channel ARAM architecture.

3.1. *The multi-channel ARAM algorithm*

Although only three types of sensory memory are identified here, the Concept Formation Network (CFN) is built upon multi-channel ARAM that is capable of learning pattern associations across an *arbitrary* number of input channels. A multi-channel ARAM model consists of a number of input representation fields $F_1^{c1}, F_1^{c2}, \ldots, F_1^{cK}$ and a category field F_2^c [Fig. 3]. 2-channel ARAM is equivalent to ARAM. If only one channel is present, multi-channel ARAM reduces to ART. The dynamics of fuzzy multi-channel ARAM, that employs fuzzy ART operations,[18] is described below.

Activity vectors: Let $\mathbf{I}^{ck}$ denote the F_1^{ck} input vector. Let $\mathbf{x}^{ck}$ denote the F_1^{ck} activity vector. Let $\mathbf{y}^{c}$ denote the F_2^{c} activity vector. Upon input presentation, $\mathbf{x}^{ck} = \mathbf{I}^{ck}$.

Weight vectors: Let $\mathbf{w}_j^{ck}$ denote the weight vector associated with the jth node in F_2^c for learning the input representation in F_1^{ck}. Initially, all F_2^c nodes are uncommitted and the weight vectors contain all 1's.

Parameters: Fuzzy multi-channel ARAM dynamics is determined by choice parameters $\alpha_{ck} > 0$ for $k = 1, \ldots, K$; learning rate parameters $\beta_{ck} \in [0,\, 1]$ for $k = 1, \ldots, K$; contribution parameters $\gamma_{ck} \in [0,\, 1]$ for $k = 1, \ldots, K$ where $\sum_{k=1}^{K} \gamma_{ck} = 1$; and vigilance parameters $\rho_{ck} \in [0,\, 1]$ for $k = 1, \ldots, K$.

Code activation: Given activity vectors $\mathbf{x}^{c1}, \mathbf{x}^{c2}, \ldots, \mathbf{x}^{cK}$, for each F_2^c node j, the choice function T_j^c is computed as follows:

$$T_j^c = \sum_{k=1}^{K} \gamma_{ck} \frac{|\mathbf{x}^{ck} \wedge \mathbf{w}_j^{ck}|}{\alpha_{ck} + |\mathbf{w}_j^{ck}|}\,, \tag{1}$$

where the fuzzy AND operation $\wedge$ is defined by

$$(\mathbf{p} \wedge \mathbf{q})_i \equiv \min(p_i,\, q_i)\,, \tag{2}$$

and the norm $|.|$ is defined by

$$|\mathbf{p}| \equiv \sum_i p_i \tag{3}$$

for vectors $\mathbf{p}$ and $\mathbf{q}$.

Code competition: All F_2^c nodes undergo a code competition process. The winner is indexed at J where

$$T_J^c = \max\{T_j^c : \text{ for all } F_2^c \text{ node } j\}\,. \tag{4}$$

When a category choice is made at node J, $y_J^c = 1$; and $y_j^c = 0$ for all $j \neq J$.

Template matching: Resonance occurs if for each channel k, the *match function* m_J^{ck} meets its vigilance criterion:

$$m_J^{ck} = \frac{|\mathbf{x}^{ck} \wedge \mathbf{w}_J^{ck}|}{|\mathbf{x}^{ck}|} \geq \rho_{ck}\,. \tag{5}$$

Learning then ensues, as defined below. If any of the vigilance constraints is violated, mismatch reset occurs in which the value of the choice function T_J^c is set to 0 for the duration of the input presentation. The search process repeats to select another F_2^c node J until resonance is achieved.

Template learning: Once a node J is selected for firing, for each channel k, the weight vector $\mathbf{w}_J^{ck}$ is modified by the following learning rule:

$$\mathbf{w}_J^{ck(\text{new})} = (1 - \beta_{ck})\mathbf{w}_J^{ck(\text{old})} + \beta_{ck}(\mathbf{x}^{ck} \wedge \mathbf{w}_J^{ck(\text{old})})\,. \tag{6}$$

3.2. *Learning in CFN*

As in an ART system, an object signal in a sensory field does not always recruit an uncommitted node. If the weight vector of any committed concept node is *close* enough to the signal pattern (as determined by the choice functions) and satisfies the vigilance criteria (as determined by the match functions), the concept node will be used to encode the current object signal. Only when no such committed node can be identified, the object signal recruits an uncommitted node.

Concept nodes can be recruited or activated by object signals from one or more sensory fields. When novel signals of two or more channels are presented simultaneously, the signal activities are used in conjunction to recruit an uncommitted concept node during the code activation stage. The selected concept node is then used to learn the signals across multiple channels. If an uncommitted concept node is first recruited by a signal from one field, its weight vector will be adjusted to encode the signal pattern. When the encoded signal is presented later together with a novel signal from another field, the same concept node can be identified by the previously learned signal to encode the novel signal.

A typical scenario in which CFN learns an object concept from its sensory cues is as follows. Suppose CFN is presented with a few visual instances of an object. Based on the visual memory field activities, an uncommitted concept node is recruited so that the system is able to recognize any identical object seen later. If CFN is then presented with both the auditory and visual signals of the object, the same concept node activated by the visual signals can be used to learn the object's auditory representation and associate it to the learned visual representation. When CFN is later presented with the object's verbal signals together with other sensory signals, it can again use the same concept node identified by other learned sensory representations to acquire the verbal representation of the object. After learning sensory representations across multiple modalities, presentation of any one of the sensory inputs allows recall in all sensory channels.

It is important to note that the above scenario is only one of the many possible ways in which CFN acquires a concept. A concept can have only one or two types of sensory representation. For example, the concept *air* has only auditory and verbal representation. It could also happen that two or more concept nodes are assigned to encode different sensory representations of a concept. In any case, CFN is robust enough to handle an arbitrary sequence of learning events in a dynamic environment. If more than one concept nodes happen to encode the different sensory representations of a concept, when the sensory representations are presented together, one of the concept nodes will be selected arbitrarily by the code competition mechanism to encode all sensory representations of the concept. The other nodes left unused can eventually become uncommitted or be removed. One way to remove unused concept nodes automatically (forgetting) is to associate to each node a confidence factor computed in real-time. A node with little or no reinforcement will have low confidence and can be removed once the confidence falls below a chosen threshold. The use of confidence factors has been found to be effective in pruning category nodes of fuzzy ARTMAP systems.[19–21]

4. Concept Hierarchy Network

4.1. *The concept hierarchy representation*

In Ref. 2, *is-a* and *is-not-a* links are used to connect various object classes as follows:

elephant *is-a* gray-thing
royal-elephant *is-not-a* gray-thing
royal-elephant *is-an* elephant

The above relations are normally represented in the form of an *is-a* hierarchy [Fig. 4]. By noting that the use of classes such as *gray-thing* seems unnatural, Sun[4] suggested to move from extensions (object classes) to intensions (object concepts), and interpret the concepts on the right-hand sides of the *is-a* statements to be the defining features of the concepts on the left-hand sides. The concept hierarchy representation proposed here also adopts

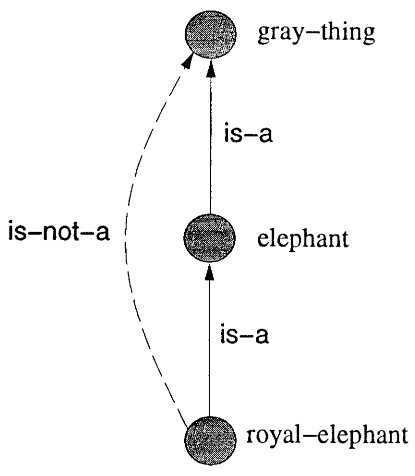


Fig. 4. The elephant relations represented in an *is-a* hierarchy.

the intensional approach. Formally, each of the above statements can be expressed in the form of relation [C:D], which denotes that a concept C is defined in terms of a list of other concepts D. Using this notation, the elephant *is-a* statements can be translated into the following relations:

[elephant : gray]
[royal-elephant : white, elephant]

Note that an additional concept representing the color of royal-elephant is introduced to facilitate discussions. To signify that the concept on the left-hand side of a relation is at a higher level than the concepts on the right-hand side, the above relations can be represented in a concept hierarchy network [Fig. 5]. Note that the ordering of concepts in the concept hierarchy network is just the opposite of the ordering of object classes in a property inheritance graph [Fig. 6]. The concept hierarchy representation is more natural for the intensional approach in which the meaning of a concept is built on a set of lower-level concepts. Another important difference is that, in place of the *is-a* links and *property* links in the property inheritance graph, a single type of bidirectional conditionable links is used in the concept hierarchy representation. These links are conditionable in the sense that there are connection strengths (or weights) attached that can be modified during learning.

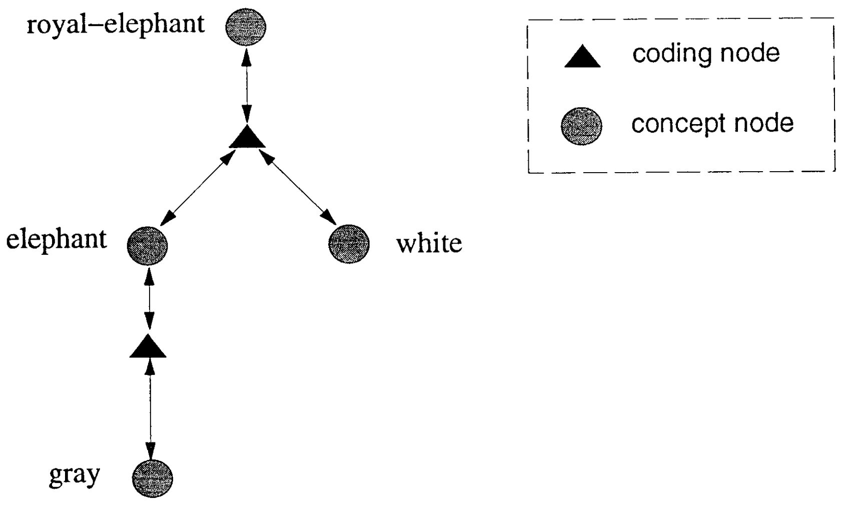


Fig. 5. A concept hierarchy network representing the elephant relations.

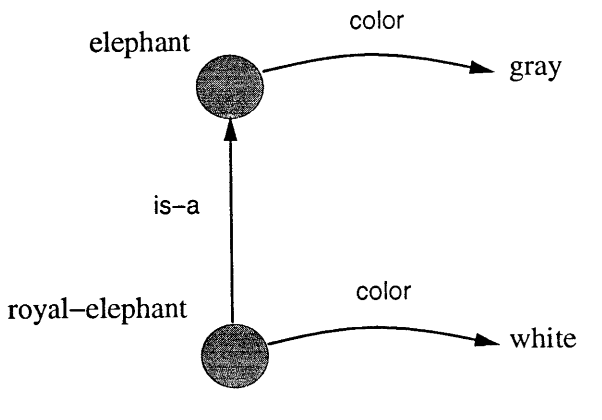


Fig. 6. A property inheritance graph representing the elephant relations.

4.2. *Learning in CHN*

A concept hierarchy is composed of a set of relations, each associating a concept to its defining concepts. The Concept Hierarchy Network (CHN) encodes each such relation using a coding node in the coding field F_2. Figure 7 shows how CHN encodes a sample elephant concept hierarchy given below:

[elephant: big-ear, long-nose, gray]
[royal-elephant: white, wear-clothes, elephant]

Two coding nodes are used to encode the relations. *elephant* is encoded by coding node 1, while *royal-elephant* is defined by coding node 2. Note that *elephant* defined in the first relation, is used in the second relation in defining *royal-elephant.* By unifying *elephant* in the two relations, the two cognitive nodes combine to represent the desired concept hierarchy.

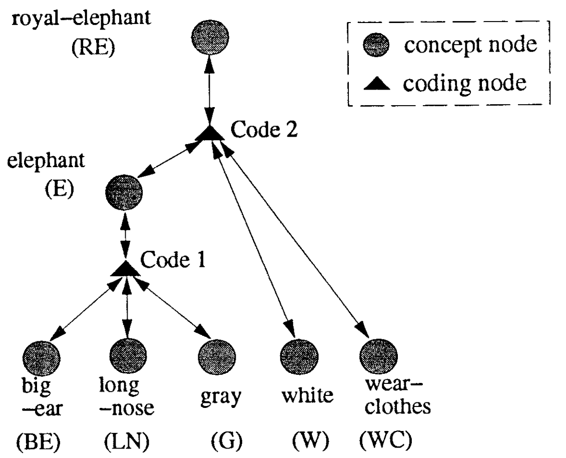
CHN learns one relation at a time. A learning cycle involves code activation, code competition, and template learning. It is important to note that the network does not merely remember each and every relation given (or else it will be of little interest). Competitive learning and template matching provide code compression (generalization) and abstraction of concept relations. The CHN dynamics realized by the fuzzy ARAM algorithm is described below.

Input vectors: Given a relation [C:D], let $\mathbf{I}^a$ be the F_1^a input vector representing D and $\mathbf{I}^b$ be the F_1^b input vector representing C.

Activity vectors: Let $\mathbf{x}^a$ and $\mathbf{x}^b$ denote the F_1^a and F_1^b activity vectors respectively. Let $\mathbf{y}$ denote the F_2 activity vector. Upon input presentation, $\mathbf{x}^a = \mathbf{I}^a$ and $\mathbf{x}^b = \mathbf{I}^b$.

Weight vectors: Let $\mathbf{w}_j^a$ and $\mathbf{w}_j^b$ denote the weight vectors associated with the jth F_2 node for encoding concepts in F_1^a and F_1^b respectively. Initially, all F_2 nodes are uncommitted and the weight vectors contain all 1's.

Parameters: The CHN dynamics is determined by choice parameters $\alpha_a > 0$ and $\alpha_b > 0$; learning rate parameters $\beta_a \in [0, 1]$ and $\beta_b \in [0, 1]$; vigilance



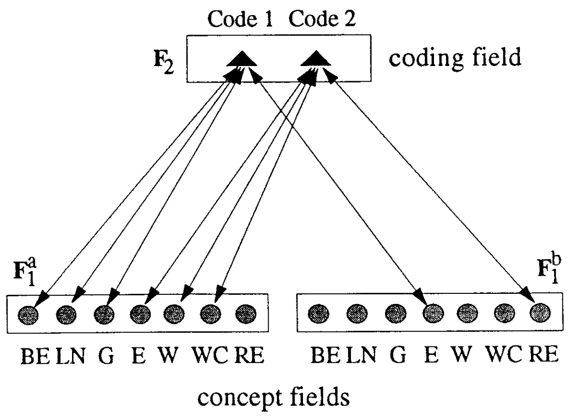


Fig. 7. Coding of a concept hierarchy. (Top) A concept hierarchy encoded; (Bottom) Actual encoding of the relations in CHN. Only non-zero connections are shown.

parameters $\rho_a \in [0, 1]$ and $\rho_b \in [0, 1]$; and a control parameter $\gamma \in [0, 1]$.

Code activation: Given activity vectors $\mathbf{x}^a$ and $\mathbf{x}^b$, for each F_2 node j, the choice function T_j is computed by

$$T_j = \gamma \frac{|\mathbf{x}^a \wedge \mathbf{w}_j^a|}{\alpha_a + |\mathbf{w}_j^a|} + (1-\gamma) \frac{|\mathbf{x}^b \wedge \mathbf{w}_j^b|}{\alpha_b + |\mathbf{w}_j^b|}, \quad (7)$$

where the fuzzy AND operation $\wedge$ is defined in Eq. (2) and the norm $|.|$ is defined in Eq. (3).

Code competition: All F_2 nodes undergo a code competition process. The winner is indexed at J where

$$T_J = \max\{T_j : \text{ for all } F_2 \text{ node } j\}. \quad (8)$$

When a category choice is made at node J, $y_J = T_J$; and $y_j = 0$ for all $j \neq J$.

Template matching: Resonance occurs if the *match functions* m_J^a and m_J^b meet the vigilance criteria in their respective fields:

$$m_J^a = \frac{|\mathbf{x}^a \wedge \mathbf{w}_J^a|}{|\mathbf{x}^a|} \geq \rho_a \quad \text{and} \quad m_J^b = \frac{|\mathbf{x}^b \wedge \mathbf{w}_J^b|}{|\mathbf{x}^b|} \geq \rho_b\,. \quad (9)$$

Learning then ensues, as defined below. If any of the vigilance constraints is violated, mismatch reset occurs in which the value of the choice function T_J is set to 0 for the duration of the input presentation. The search process repeats to select another F_2 node J until resonance is achieved.

Template learning: Once a node J is selected, the weight vectors $\mathbf{w}_J^a$ and $\mathbf{w}_J^b$ are modified by the learning rule

$$\mathbf{w}_J^{k(\text{new})} = (1-\beta_k)\mathbf{w}_J^{k(\text{old})} + \beta_k(\mathbf{x}^k \wedge \mathbf{w}_J^{k(\text{old})}) \quad \text{for } k = a \text{ and } b\,. \quad (10)$$

4.3. *Inferencing in CHN*

Inferencing in the Concept Hierarchy Network (CHN) is through the activation of F_2 nodes based on the F_1^a and F_1^b memory states [Fig. 8]. The concept fields F_1^a and F_1^b are identified as the working memory. They maintain the current memory state, provide premises for condition matching, and store the next memory state derived through a code firing.

A single inferencing cycle consists of code activation, code competition, template matching, activity readout, and memory update. During code activation, a choice function T_j is computed for each F_2 coding node based on the activity vectors $\mathbf{x}^a$ and $\mathbf{x}^b$. Code competition is realized by a winner-take-all interaction among all F_2 nodes in which the F_2 node with the largest choice function T_j is identified. During template matching, if the selected node does not satisfy a vigilance constraint, the system goes through another round of memory search to select another F_2 node that satisfies the vigilance criterion. If no such node exists, the system halts. Otherwise,

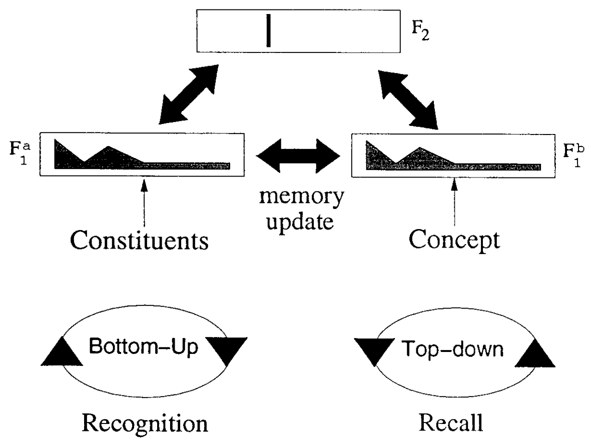


Fig. 8. Inferencing in a concept hierarchy. (Left) Bottom-up activation leads to recognition; (Right) Top-down activation gives rise to recall.

activity readout occurs in which the template values of the selected code are read out into F_1^a and F_1^b. Note that exact match is not required for a code to fire as long as it satisfies the vigilance criterion. At the end of the cycle, the activity vectors $\mathbf{x}^a$ and $\mathbf{x}^b$ update each other to prepare for the next inferencing cycle. One unique feature of the above inference procedure is that both $\mathbf{x}^a$ in F_1 and $\mathbf{x}^b$ in F_2 can be the premise of a code firing. This gives rise to bidirectional inferencing. By the code activation and competition process, a code can fire as long as the vigilance criterion is satisfied. This allows inexact match or approximate reasoning.

The equations for code activation and competition during inferencing are exactly the same as those during learning. However, for template matching, resonance is considered achieved even if only one of the vigilance criteria is satisfied. The dynamics of template matching, activity readout, and memory update is described below.

Template matching: Resonance during inferencing occurs if either the *match function* m_J^a or m_J^b meets its vigilance criterion:

$$m_J^a = \frac{|\mathbf{x}^a \wedge \mathbf{w}_J^a|}{|\mathbf{x}^a|} \geq \rho_a \quad \text{or} \quad m_J^b = \frac{|\mathbf{x}^b \wedge \mathbf{w}_J^b|}{|\mathbf{x}^b|} \geq \rho_b\,. \tag{11}$$

Activity readout: When an F_2 node J is selected for firing, the activity vectors $\mathbf{x}^a$ and $\mathbf{x}^b$ are updated as follows:

$$\mathbf{x}^{a(\text{new})} = \mathbf{x}^{a(\text{old})} \vee \lambda y_J \mathbf{w}_J^a$$

and

$$\mathbf{x}^{b(\text{new})} = \mathbf{x}^{b(\text{old})} \vee y_J \mathbf{w}_J^b\,, \tag{12}$$

where the fuzzy OR operation $\vee$ is defined by

$$(\mathbf{p} \vee \mathbf{q})_i \equiv \max(p_i, q_i)\,. \tag{13}$$

The readout equations write the inference results, denoted by the weight templates $\mathbf{w}_J^a$ and $\mathbf{w}_J^b$ of the activated code J, into the activity vectors $\mathbf{x}^a$ and $\mathbf{x}^b$ respectively. The fuzzy OR operation is used to merge the newly derived results with the existing memory state. $\lambda \in (0, 1)$ is an attenuation parameter to prevent the infinite propagation of activities down the concept hierarchy (explained below). λ is also essential for achieving cancellation of inheritance in conflicting situations [Sec. 5.2].

Memory update: After activity readout, $\mathbf{x}^a$ and $\mathbf{x}^b$ update each other using the equation

$$\mathbf{x}^{a(\text{new})} = \mathbf{x}^{b(\text{new})} = \mathbf{x}^{a(\text{old})} \vee \mathbf{x}^{b(\text{old})}\,. \tag{14}$$

This ensures that the two activity vectors contain the same accumulated inference results. The next inferencing cycle can then be performed using either $\mathbf{x}^a$ or $\mathbf{x}^b$ as the premise.

To prevent persevering firing, a fired F_2 node is forbidden from getting chosen again in the same inferencing task. The inferencing process continues until no coding node selected satisfies the vigilance condition. Thereafter, no change in the $\mathbf{x}^a$ and $\mathbf{x}^b$ values occurs and the system is said to be stabilized.

Note that the direction of inferencing is determined by the control parameter γ used in Eq. (7). With $0 < \gamma < 1$, the system exhibits a general form of spreading activation in which inferencing can be performed in both $F_1^a \to F_1^b$ and $F_1^b \to F_1^a$ directions. With $\gamma = 1$, F_2 receives activities solely from F_1^a and thus the activities can only flow from F_1^a to F_1^b. This corresponds to an upward flow of activations in the concept hierarchy, which effectively implements a recognition process. With $\gamma = 0$, F_2 receives activities solely from F_1^b and thus the activities can only flow from F_1^b to F_1^a. This corresponds to a downward flow of activations, which results in a recall or inheritance process. Inheritance is a complicated process in which several properties or constraints need be observed. A formal treatment of property inheritance and conflict resolution is given in the next section.

5. Commonsense Reasoning

Before we begin, it would be appropriate to fix some terminologies to facilitate discussions that follow.

Definition 1 [Hyper-concept]

Let A and B be two concepts. A is said to be a hyper-concept of B if A uses B as one of its defining features.

Definition 2 [Element-concept]

Let A and B be two concepts. B is said to be an element-concept of A if A is a hyper-concept of B.

When we say "A is a hyper-concept of B", we mean "A is a more complex concept built on B". In the elephant example, *royal-elephant* is a hyper-concept of *elephant*; and *elephant* is an element-concept of *royal-elephant.*

5.1. *Inheritance*

The most common type of property inheritance is super-class to sub-class inheritance or *top-down* inheritance, in which the properties of a sub-class (hyper-concept) are inherited from its super-classes (element-concepts). For example, the *long-nose* property of *royal-elephant* can be inherited from *elephant.* In a concept hierarchy, top-down inheritance can be visualized as the top-down propagation of activities from a hyper-concept to its element-concepts and so on down the hierarchy. More complex situations require *cancellation of inheritance*, also known as *exception handling*, in which inheritance of a property can be overridden by other conflicting information. For example, *royal-elephant* cannot inherit *gray* from *elephant* as it is stated *white.* Cancellation of inheritance and conflicting multiple inheritance are discussed in more details in Secs. 5.2 and 5.3 respectively.

Shastri[3] and Sun[4] also described a type of *bottom-up* inheritance (percolation of inheritance) in which the properties of a super-class (element-concept) can be, to a certain extent, inferred from the properties

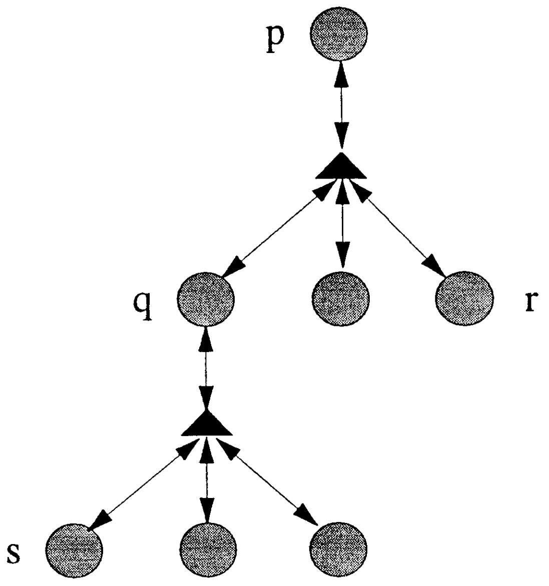


Fig. 9. A generic concept hierarchy for illustrating property inheritance and conflict resolution.

of its sub-classes (hyper-concepts). For example, the *wear-clothes* property of *elephant* may be somewhat inferred from that of *royal-elephant.* Similarly, in a concept hierarchy, bottom-up inheritance can be visualized as the bottom-up propagation of activities from an element-concept to its hyper-concepts and so on up the hierarchy. The functional behaviors of the above two types of inheritance can be translated into the concept hierarchy formalism to serve as the design constraints of Concept Hierarchy Network (CHN). For a generic concept hierarchy [Fig. 9], the following properties must hold.

Property 1 [Top-down Inheritance]

Let p be a concept and q be an element-concept of p. Suppose q has an element-concept s that is not an element-concept of p. If p is activated, s should also be activated.

Property 2 [Bottom-up Inheritance]

Let p be a concept and q be an element-concept of p. Suppose p has an element concept r that is not an element-concept of q. If q is activated, r should also be somewhat activated.

Let x_c denote the activity of a concept c and M_c denote the number of element-concepts of c. It can be verified mathematically that a spreading activation process in CHN obtained by $\gamma = \frac{1}{2}$ satisfies the properties delineated above. For simplicity, all computations assume $\alpha_a = \alpha_b = 0$, $\rho_a = \rho_b = 0$, and unit weights for all non-zero connections.

Proof [Top-down Inheritance]

Given $x_p = 1.0$, we have $x_q = \frac{\lambda}{2}$ and $x_s = \frac{\lambda^2}{4}$. Thus s is activated.

Proof [Bottom-up Inheritance]

Given $x_q = 1.0$, we have $x_r = \frac{\lambda}{2M_p}$. Thus r is activated.

5.2. *Cancellation of inheritance*

It is important to ensure that properties are only inherited in proper contexts. In the elephant example, when *royal-elephant* is activated, by the spreading activation procedure, both attributes *gray* and *white* will be activated. Cancellation of inheritance is thus required, so that only *white* is activated. Before discussing cancellation of inheritance, first consider a more general property called *selective attention* by which more relevant concepts are more activated than the others. For example, when *elephant* is activated, *long-nose* and *big-ear* should be more activated than *wear-clothes.* The property of selective attention can be stated as follows:

Property 3 [Selective Attention]

Let p be a concept and q and r be element-concepts of p. Suppose q has an element-concept s that is not an element-concept of p.

(a) If p is activated, r should be more activated than s.
(b) If q is activated, s should be more activated than r.

Proof

By the same spreading activation process ($\gamma = \frac{1}{2}$), selective attention is readily achieved as the attenuation parameter λ is less than 1 and M_c is usually greater than 1:

(a) Given $x_p = 1.0$, we have $x_q = \frac{\lambda}{2}$, $x_r = \frac{\lambda}{2}$, and $x_s = \frac{\lambda^2}{4}$. With $0 < \lambda < 1$, we have $x_r > x_s$.
(b) Given $x_q = 1.0$, we have $x_r = \frac{\lambda}{2M_p}$ and $x_s = \frac{\lambda}{2}$. With $M_p > 1$, we have $x_s > x_r$.

Property 4 [Cancellation of Inheritance]

Let p be a concept and q and r be element-concepts of p. Suppose q has an element-concept s that is contradictory to r.

(a) If p is activated, the activity of r should quash that of s.
(b) If q is activated, the activity of s should quash that of r.

Proof

Cancellation of Inheritance can be achieved by organizing sets of conflicting concepts into shunting competitive fields.[22] In an on-center off-surround competitive field, every node has an excitatory connection to itself and inhibitory connections to other nodes. By grouping conflicting concepts such as r and s in the generic concept hierarchy into a winner-take-all field, only one of which has the strongest activity will survive. In case (a) above, r which is more activated than s will quash the activity of s and become the winner. In case (b), s being more activated than r will quash the activity of r and become the winner.

In the elephant example, the conflicting concepts of colors such as *gray* and *white* can be grouped into a competitive pool. When *royal-elephant* is activated, by Property 3, *white* is more activated than *gray.* By Property 4, the stronger activity of *white* will quash that of *gray,* so that *white* is inferred as the color of *royal-elephant.*

5.3. *Conflicting multiple inheritance*

In general, when properties are inherited through multiple inheritance paths, conflicts could occur. This is known as conflicting multiple inheritance of which a typical example is the Nixon multiple inheritance problem [Table 1]. By using fuzzy connection strengths, CHN resolves such a multiple inheritance conflict. Suppose that Nixon is known to be a 90%

Quaker and a 100% Republican. These fuzzy relationships can be captured in their respective template weights [Fig. 10]. Assuming non-fuzziness for other relations, it can be verified in the recall (inheritance) mode with $\gamma = 0$ that the activation of *pacifist* is $0.9\lambda^2$ and that of *non-pacifist* is λ^2. This implies that Nixon is more likely a *non-pacifist*. In this problem, the use of fuzzy connection strengths provides an extra degree of freedom in resolving subtle conflicting situations.

Table 1. A multiple inheritance problem: Is Nixon a pacifist or non-pacifist?

Nixon	:-	Quaker
Nixon	:-	Republican
Quaker	:-	Acifist
Republican	:-	Non-Pacifist

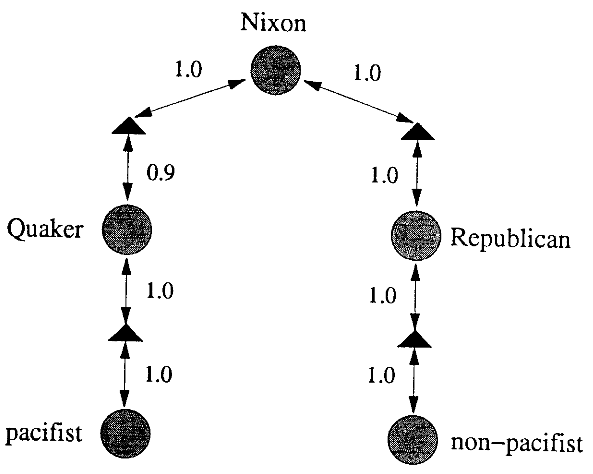


Fig. 10. Concept hierarchy representation of the Nixon multiple inheritance problem.

5.4. *Illustrations*

In this section, some examples of CHN inferencing are given to illustrate how the network performs recognition and inheritance. The elephant example [Fig. 7] is used as the sample concept hierarchy. Again for simplicity, all computations assume $\alpha_a = \alpha_b = 0$, $\rho_a = \rho_b = 0$, and unit weights for all non-zero connections.

5.4.1. *Recognition*

In a recognition task, higher level concepts are identified based on a set of lower level concepts. With $\gamma = 1$, CHN is in a recognition mode, in which the coding field F_2 receives inputs solely from F_1^a. Table 2 shows the transition of memory states in F_1^a, F_1^b, and F_2 after the three input cues: *long-nose*, *big-ear*, and *white*, are presented. After the first code firing, *long-nose* and *big-ear* activate *elephant* with an activity of $\frac{2}{3}$. In the second inferencing cycle, *elephant* and *white* activate *royal-elephant* with an activity of $\frac{5}{9}$. Thus the concept *royal-elephant* is recognized.

5.4.2. *Inheritance*

Two recall tasks are given below to illustrate property inheritance and exception handling. With $\gamma = 0$, CHN is in a recall mode. Table 3 shows the transition of memory states after *royal-elephant* is activated. The first code firing activates *elephant*, *white*, and *wear-clothes*, each with an activation of λ. The second code firing activates *long-nose*, *big-ear*, and

Table 2. Transition of CHN memory states in recognizing *royal-elephant*.

	F_1^a and F_1^b Activities ($\mathrm{x}^a = \mathrm{x}^b$)							F_2 Activities (y)	
Time	Long-Nose	Big-Ear	Gray	Elephant	Wear-Clothes	White	Royal-Elephant	y_1	y_2
0	1	1	0	0	0	1	0	0	0
1	1	1	$\frac{2\lambda}{3}$	$\frac{2}{3}$	0	1	0	$\frac{2}{3}$	0
2	1	1	0	$\frac{2}{3}$	$\frac{5\lambda}{9}$	1	$\frac{5}{9}$	0	$\frac{5}{9}$

Table 3. Transition of CHN memory states after activating *royal-elephant*.

	F_1^a and F_1^b Activities ($\mathbf{x}^a = \mathbf{x}^b$)							F_2 Activities ($\mathbf{y}$)	
Time	Long-Nose	Big-Ear	Gray	Elephant	Wear-Clothes	White	Royal-Elephant	y_1	y_2
0	0	0	0	0	0	0	1	0	0
1	0	0	0	λ	λ	λ	1	0	1
2	λ^2	λ^2	λ^2	λ	λ	λ	1	λ	0
3	λ^2	λ^2	0	λ	λ	λ	1	0	0

Table 4. Transition of CHN memory states after activating *elephant*.

	F_1^a and F_1^b Activities ($\mathbf{x}^a = \mathbf{x}^b$)							F_2 Activities ($\mathbf{y}$)	
Time	Long-Nose	Big-Ear	Gray	Elephant	Wear-Clothes	White	Royal-Elephant	y_1	y_2
0	0	0	0	1	0	0	0	0	0
1	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	1	0	0	0	$\frac{1}{2}$	0
2	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	1	$\frac{\lambda}{6}$	$\frac{\lambda}{6}$	$\frac{1}{6}$	0	$\frac{1}{6}$
3	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	$\frac{\lambda}{2}$	1	$\frac{\lambda}{6}$	0	$\frac{1}{6}$	0	0

gray, each with a smaller activation λ^2. By organizing conflicting concepts like *gray* and *white* into a winner-take-all competitive field, cancellation of inheritance can be achieved by which the activity of *white* quashes that of *gray*, as illustrated in the third time step.

With $\gamma = \frac{1}{2}$, the model exhibits a more general form of spreading activation. Table 4 shows the transition of memory states after *elephant* is activated. The coding node of *elephant* is first fired which activates *long-nose*, *big-ear*, and *gray*, each with an activation of $\frac{\lambda}{2}$. The next code firing activates *white* and *wear-clothes*, each with a smaller activation $\frac{\lambda}{6}$. In this case, the activity of *gray*, being greater than the activity of *white*, quashes that of *white*.

6. Conclusion

Based on Adaptive Resonance Theory (ART), a cognitive architecture termed Concept Hierarchy Memory Model (CHMM) has been proposed for concept hierarchy representation and commonsense reasoning. Composed of a Concept Formation Network (CFN) and a Concept Hierarchy Network (CHN), CHMM provides a systematic treatment for dynamic concept formation and concept hierarchy organization. Using a unified inferencing mechanism, CHN performs an important class of commonsense reasoning, including recognition and inheritance.

This paper however only marks a starting point of modeling conceptual knowledge. There are many more desirable features that could be incorporated into the Concept Hierarchy Memory Model. Some of the more interesting ones are highlighted below.

1. The inferencing outcomes of the Concept Hierarchy Network (CHN) are indicated by the activities of the concept nodes and are not readily available to an external user. It is thus useful to develop a query mechanism that based on a user's query, activates the appropriate concepts in CHN, and based on the stabilized memory state, produces applicable

answer(s). Additionally, the query system could help to direct the inferencing processing. CHMM integrated with the query mechanism should form an important component of a knowledge-based language understanding system.

2. The article considers only inferencing in the Concept Hierarchy Network (CHN). The potential subsymbolic inferencing process in the Concept Formation Network (CFN) is not investigated. Interaction between the symbolic inferencing of CHN and the subsymbolic processing of CFN might provide an even more powerful reasoning mechanism.
3. Finally, CHMM is currently restricted to representing object concepts that can be defined in terms of a number of other concepts. It is thus not as expressive as semantic network. The model faces an immediate and more challenging problem of representing structural concepts that involves handling of role/filler relationships.

Despite the above limitations, CHMM has offered a novel approach to learning conceptual knowledge, a problem that is generally not tackled by other commonsense reasoning systems. It is the author's contention that by building from a model that is simple, intuitive, and has captured certain important aspects of conceptual knowledge learning, one has a better chance of modeling human intelligence.

Acknowledgments

This manuscript is based on a chapter of a doctoral dissertation submitted to Boston University. The first author wishes to acknowledge his dissertation committee members: Gail A. Carpenter, Stephen Grossberg, and Michael Cohen for their guidance and support. The authors also thank the two anonymous referees for providing useful comments and suggestions to a previous version of this manuscript.

References

1. S. Fahlman 1979, *NETL: A System for Representing and Using Real-World Knowledge* (MIT Press, Cambridge, MA).
2. D. Touretzky 1986, *The Mathematics of Inheritance Systems* (Morgan Kaufmann, San Mateo, CA).
3. L. Shastri 1988, "A connectionist approach to knowledge representation and limited inference," *Cognitive Science* **12**, 331–392.
4. R. Sun 1993, "An efficient feature-based connectionist inheritance scheme," *IEEE Trans. Syst., Man, and Cybernetics* **23**, 512-522.
5. J. A. Feldman 1989, "Connectionist representation of concepts," *Connectionism in Perspective* (Elsevier/North Holland, Amsterdam), pp. 25–45.
6. H. S. Soon and A. H. Tan 1993, "Concept hierarchy networks for inheritance systems: Concept formation, property inheritance and conflict resolution," *Proc. 15th Conf. Cognitive Sci. Soc.* 941–946.
7. H. S. Soon and A. H. Tan 1993, "A memory model for concept hierarchy representation and commonsense reasoning," *Proc. World Congress Neural Networks* **II** (Lawrence Erlbaum Associates, Hillsdale, NJ), pp. 206–209.
8. A. H. Tan 1994, *Synthesizing Neural Network and Symbolic Knowledge Processing*, Ph.D. thesis, Department of Cognitive and Neural Systems. Boston University, Boston, MA.
9. A. H. Tan 1992, "Adaptive Resonance Associative Map: A hierarchical ART system for fast stable associative learning," *Proc. IJCNN-92* **I** (IEEE Service Center, Piscataway, NJ), pp. 860–865.
10. A. H. Tan 1994, "Adaptive Resonance Associative Map: A neural model for heteroassociative learning and recall," *Proc. ICONIP-94* **II**, 731–736.
11. A. H. Tan 1995, "Adaptive Resonance Associative Map," *Neural Networks*, **8**, 437–446.
12. G. A. Carpenter, S. Grossberg and J. H. Reynolds 1992, "ARTMAP: Supervised real time learning and classification by a self-organizing neural network," *Neural Networks* **4**, 565–588.
13. G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds and D. B. Rosen 1992, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Networks* **3**, 698–713.
14. B. Kosko 1992, *Neural Networks and Fuzzy Systems* (Prentice-Hall, Englewood Cliffs, NJ).
15. Y. F. Wang, J. B. J. Cruz, and J. H. J. Mulligan 1990, "Two coding strategies for bidirectional associative memory,"*IEEE Trans. Neural Networks* **1**, 81–92.
16. Y. F. Wang, J. B. J. Cruz, and J. H. J. Mulligan 1991, "Guaranteed recall of all training pairs for bidirectional associative memory," *IEEE Trans. Neural Networks* **2**, 559–567.
17. X. H. Zhuang, Y. Huang, and S. S. Chen 1993 "Better learning for bidirectional associative memory," *Neural Networks* **6**, 1131–1146.
18. G. A. Carpenter, S. Grossberg, and D. B. Rosen 1991, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks* **4**, 759–771.
19. G. A. Carpenter and A. H. Tan 1993, "Fuzzy ARTMAP, rule extraction and medical databases,"

Proc. World Congress on Neural Networks **I** (Lawrence Erlbaum Associates, Hillsdale, NJ), pp. 501–506.

20. G. A. Carpenter and A. H. Tan 1995, "Rule extraction: From neural architecture to symbolic representation," *Connection Science* **7**, 3–27.
21. A. H. Tan 1994, "Rule learning and extraction with self-organizing neural networks," *Proc. 1993 Connectionist Models Summer School*, eds. M. C. Mozer, P. Smolensky, D. S. Touretzky, J. L. Elman and A. S. Weigend, (Lawrence Erlbaum Associates, Hillsdale, NJ), pp. 192–199.
22. S. Grossberg 1973, "Contour enhancement, short term memory, and constancies in reverberating neural networks," *Studies in Applied Mathematics* **52**, 217–257.