Title
Concept Hierarchy Networks for Inheritance Systems: Concept Formations, Property Inhertitance and Conflict Resolution

Journal
Proceedings of the Annual Meeting of the Cognitive Science Society, 15(0)

Authors
Vivien Soon, Hui-Shin
Tan, Ah-Hwee

Publication Date
1993

Peer reviewed

Concept Hierarchy Networks for Inheritance Systems:

Concept Formation, Property Inheritance and Conflict Resolution

Hui-Shin Vivien Soon
College of Computer Science
Northeastern University, 161 Cullinane Hall
Boston, MA 02115, USA
vsoon@damon.ccs.northeastern.edu

Ah-Hwee Tan
Dept. of Cognitive & Neural Systems
Boston University, 111 Cummington St
Boston, MA 02215, USA
atan@park.bu.edu

Abstract

Most inheritance systems which use hierarchical representation of knowledge, do not consider learning. In this paper, a concept hierarchy network model based on adaptive resonance theory is proposed for inheritance systems, which explicitly includes learning as one of its major design goals. By chunking relations between concepts as cognitive codes, concept hierarchy can be learned/modified through experience. Furthermore, fuzzy relations between concepts can be represented by weights on links connecting them. It is shown that by a spreading activation process based on code firing, and competition between conflicting concepts, the model is able to exhibit property inheritance and to resolve such conflicting situations as exceptions and conflicting multiple inheritance.

Introduction

Formulating concepts about objects is the basis for human commonsense reasoning. In representing objects, inheritance systems (Fahlman, 1979; Touretzky, 1986) which use hierarchical organization of knowledge, is able to abstract low level information by property inheritance. However, besides simply adding/deleting links of networks, most inheritance systems are generally hard-wired, and even in their connectionist implementation (Shastri, 1988), often do not include learning mechanism. One of the main difficulties in learning as mentioned by Feldman (1989), is to create new concepts and new memory structure dynamically. The problem is aggravated by the slow learning nature of most neural network algorithms. Another limitation of most inheritance systems is that strict, non-fuzzy relationships are used to represent the relations between classes, as well as between a class and its properties. This results in rigid reasoning and is not suitable for processing commonsense knowledge.

In this paper, a concept hierarchy network (CHN) model is proposed for implementing inheritance systems. The model is based on supervised learning adaptive resonance theory (ART) networks (Carpenter *et al.*, 1992; Tan, 1992) which are able to self-organize knowledge structure using fast stable learning. As suggested by the name of concept hierarchy, the approach adopted here is an intensional one rather than extensional. Roughly speaking, while most other inheritance systems perform inheritance on classes of objects that fit into various concepts, this paper builds inheritance systems based on the meanings or semantics of concepts. The knowledge system that we are concerned with, is a common knowledge pool which interacts with sensory memory of various types including ver-

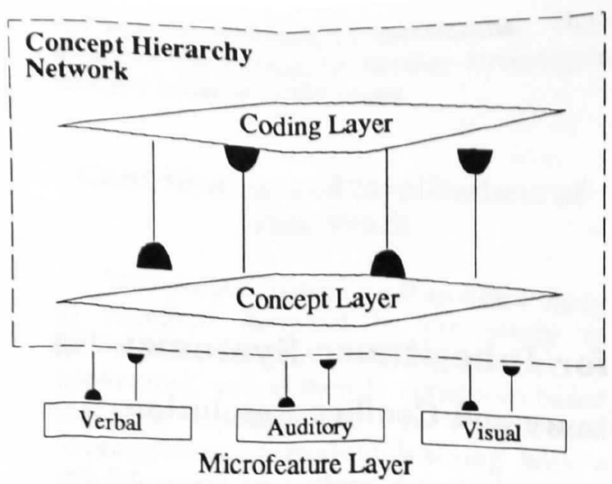

Figure 1. Schematic Diagram of Knowledge Hierarchy.

bal, auditory and visual. A schematic diagram of such a system is shown in Figure 1.

The sensory memory can be viewed as a *microfeature layer* which forms the distributed representation of concepts. The concept hierarchy network comprises a *concept layer* and a *coding layer*. In the concept layer, a node is used to represent a concept. By organizing the microfeature layer and the concept layer in an ART network, new concept node can be created whenever a novel activity pattern is activated across the microfeature layer. In coding of concept hierarchy, no direct connection is used between concept nodes. Instead, relations between concepts are learned as cognitive codes represented in the coding layer. The details of the learning procedure is given in the next section. The operations in the model are strictly local interactions between nodes across layers. Using a spreading activation procedure which propagates activations from concepts to concepts through code firing, the system is able to perform an important subclass of commonsense reasoning including recognition and inheritance. In this paper, we shall focus on the latter, showing that the model is able to exhibit basic form of property inheritance including top-down and bottom-up inheritance. By pre-organizing conflicting concepts in competitive fields, the system provides a solution for cancellation of inheritance and conflicting multiple inheritance.

Concept Hierarchy Formation

A concept hierarchy is composed of a set of relations, each associates the meaning of a concept to its defining lower level concepts. The approach taken here is to learn each such relation using a cognitive code represented in the coding layer. Given a relation [C:D] denoting that a concept C is defined in terms of a list of other concepts D, we say that C is a hyper-concept of concepts in D, and concepts in D are element-concepts of C. To enable real-time encoding of relations, a fuzzy Adaptive Resonance Associative Map (Tan, 1992) architecture which performs fast, stable associative learning, is used to implement concept hierarchy (Figure 2). The F_2 coding layer allocates a node to learn a novel relation between a concept and its definitive lower level concepts. Two identical copies of working memory fields F_1^a and F_1^b which form the concept layer, are connected to the coding layer by bidirectional conditionable links. They are used for matching the conditions for code firing and for readout of code activation. When coding a relation, the concept to be learned is represented in F_1^b and its lower level concepts are represented in F_1^a.

Relations can be directly encoded in the system by fast learning. Alternatively, given many sample cases, salient relations can be extracted and represented by the model. Let us consider learning the concept of elephant by seeing many elephants. Depending on the particular instance of elephant, a slightly different activity pattern of concepts can be obtained across the F_1^a field. For example, seeing a big elephant will activate **big** but not **small**, and vice versa. However, those characteristic features of **elephant** such as **big-ear**, **long-nose** etc are bound to appear on

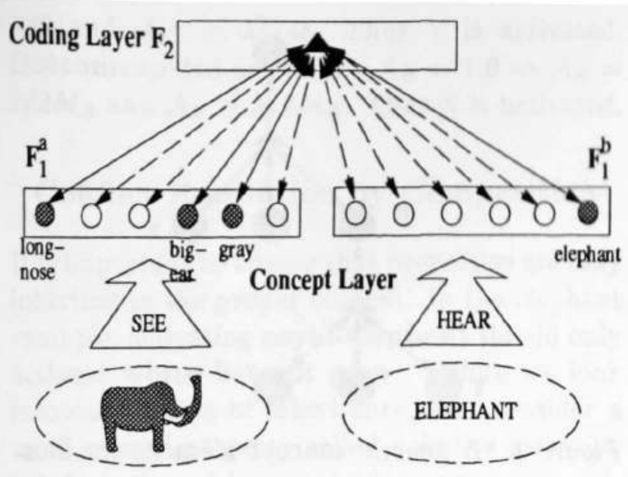

Figure 2. Learning of concept **elephant.** Solid/Empty circles indicate activated/inactive concepts. Solid/Dotted lines indicate pathways with non-zero/zero connection strengths.

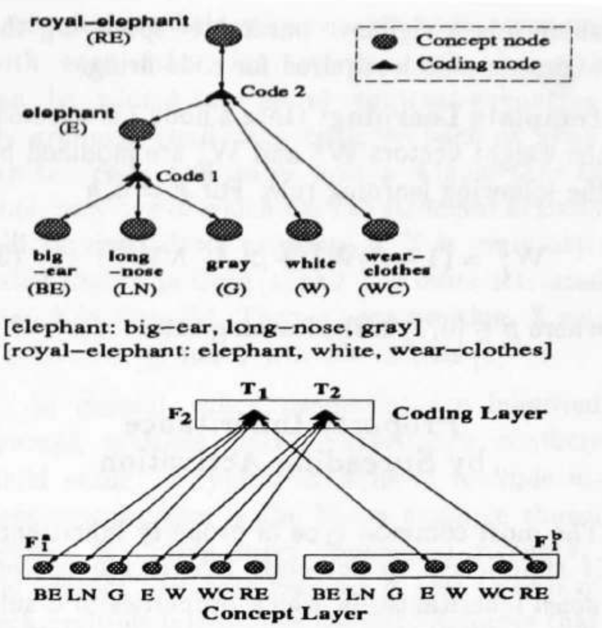

Figure 3. Coding of Concept Hierarchy. (Top) Concept hierarchy encoded; (Center) The elephant hierarchical relationships; (Bottom) Actual coding of relations in the model. Only non-zero connections are shown.

each and every elephant, and will activate the corresponding concepts consistently. The goal of the learning procedure is thus to form a template vector containing salient element-concepts in F_1^a and associate it to **elephant** represented in F_1^b. Other relations can be learned in a similar fashion. Concept hierarchy is obtained by the chaining effect of relations represented. For example, given a new relation: [royal-elephant: elephant, white, wear-clothes], another cognitive code can be created to learn it, which together with the first cognitive code, form a 3-level concept hierarchy. Figure 3 shows the relations, the concept hierarchy encoded and its actual coding in the model.

The mathematics of the system dynamics is described below. The learning procedure takes in a relation or a sample case at a time. A learning cycle involves code activation, code competition and template learning. Competitive learning and template matching achieve code compression and abstraction of concept relations.

Code Activation: Let $\mathcal{A}^a$ and $\mathcal{A}^b$ be the activity vectors in the concept layer fields F_1^a and F_1^b respectively. Let W_j^a and W_j^b be the weight vectors associated with a node j in the coding layer for coding concepts in F_1^a and F_1^b respectively. The activity of node j is computed as follows:

$$T_j = \gamma \frac{|\mathcal{A}^a \wedge \mathbf{W}_j^a|}{\alpha_a + |\mathbf{W}_j^a|} + (1-\gamma)\frac{|\mathcal{A}^b \wedge \mathbf{W}_j^b|}{\alpha_b + |\mathbf{W}_j^b|} \qquad (1)$$

where α_a and α_b are small constants, γ is a control parameter (0 for recall, 1 for recognition, typically set to 1/2 for learning and spreading activation), the fuzzy AND operation $\wedge$ is defined by $(x \wedge y) \equiv min(x, y)$ and the norm $|.|$ is defined by $|\mathbf{x}| \equiv \sum_{i=1}^{M} x_i$.

Code Competition: To ensure that only one code can be fired at a time, all F_2 nodes have to undergo a code competition process in which the eligibility for activation, E_j of a node j is evaluated as follows:

$$E_j = \begin{cases} 1 & \text{if } T_j = max\{T_J\text{: for all node } J\} \\ & \text{and } T_j > \rho \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

where ρ is a vigilance parameter specifying the minimum match required for code firing.

Template Learning: Once a node j is selected, the weight vectors $\mathbf{W}_j^a$ and $\mathbf{W}_j^b$ are modified by the following learning rule: For $F = a, b$

$$\mathbf{W}_j^F = (1-\beta)\mathbf{W}_j^F + \beta(\mathcal{A}^F \wedge \mathbf{W}_j^F) \tag{3}$$

where $\beta \in [0,1]$ is the learning rate.

Property Inheritance by Spreading Activation

The most common type of property inheritance is super-class to sub-class inheritance or *top-down* inheritance, in which properties of a subclass are inherited from its superclasses. Shastri (1988) and Sun (In press) also described a type of *bottom-up* inheritance (percolation of inheritance) in which some properties of a superclass can be, to a certain extent, inferred from the properties of its sub-classes. The functional behaviors of these two types of inheritance are translated into the concept hierarchy formalism to serve as the design constraints for our model. For a generic concept hierarchy as depicted in Figure 4, the following properties must hold.

PROPERTY 1 (Top-down Inheritance): Let A be a concept, B be an element-concept of A, and that B has an element-concept Y which is not in A. If A is activated, Y should be activated.

PROPERTY 2 (Bottom-up Inheritance): Let A be a concept, B be an element-concept of A, and that A has an element-concept X which is not in B. If B is activated, X should be somewhat activated.

In the concept hierarchy network, property inheritance is performed by a spreading activation process in which code firing in the F_2 layer modifies the memory contents in F_1^a and F_1^b. A single spreading activation cycle involves code activation, code competition (as in learning) and readout of activities. Readout into F_1^a corresponds to propagation of activities down a concept hierarchy and readout into F_1^b denotes upward flow of activities.

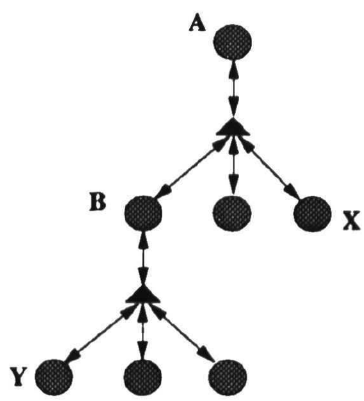

Figure 4. A generic concept hierachy for illustrating property inheritance and conflict resolution.

Activity Readout: After each code firing, the activities in F_1^a and F_1^b are updated as follows:

$$\mathcal{A}^a = \mathcal{A}^a \vee \mathcal{F}(\lambda \sum_j \mathbf{W}_j^a T_j E_j) \tag{4}$$

$$\mathcal{A}^b = \mathcal{A}^b \vee \mathcal{F}(\sum_j \mathbf{W}_j^b T_j E_j)) \tag{5}$$

where the fuzzy OR operation $\vee$ is defined by $(x \vee y) \equiv max(x, y)$, $\lambda \in (0,1)$ is an attenuation parameter to prevent infinite propagation of activities down the concept hierarchy, and $\mathcal{F}$ is a threshold-linear function with an identity range in $[0,1]$. The memory contents in F_1^a and F_1^b then update each other as follows: $\mathcal{A}^a = \mathcal{A}^b = \mathcal{A}^a \vee \mathcal{A}^b$. To prevent perservative firing of a code, a fired node is forbidden from getting fired again in a single inferencing task.

It can be verified that the spreading activation process satisfies the constraints delineated above. For simplicity, all computations assume $\alpha_a = \alpha_b = 0$, $\rho = 0$, $\gamma = 1/2$ and unit weights for all non-zero connections. $\mathcal{A}_c$ denotes the activity of a concept c. M_c denotes the number of element-concepts of a concept c.

Top-down Inheritance: $\mathcal{A}_A = 1.0 \Rightarrow \mathcal{A}_B =$

$\lambda/2$ and $\mathcal{A}_Y = \lambda^2/4$. Thus Y is activated. **Bottom-up Inheritance:** $\mathcal{A}_B = 1.0 \Rightarrow \mathcal{A}_A = 1/2M_A$ and $\mathcal{A}_X = \lambda/2M_A$. Thus X is activated.

Conflict Resolution by Competition

It is important to ensure that properties are only inherited in the proper context. In the elephant example, activating **royal-elephant** should only activate **white** but not **gray**. Before we look into cancellation of inheritance, first consider a more general property called *selective attention* in which more relevant concepts are more activated than others. For example, when **elephant** is activated, **long-nose** and **big-ear** should be more activated than **wear-clothes**. In general, the property of selective attention can be stated as follows:

PROPERTY 3 (Selective Attention): Let A be a concept, B be an element-concept of A, A has a element-concept X and B has a element-concept Y. [a] If A is activated, X should be more activated than Y. [b] If B is activated, Y should be more activated than X.

By using the same spreading activation procedure, selective attention can be achieved by noting that the attenuation parameter λ is less than 1 and M_c is usually greater than 1:
[a] $\mathcal{A}_A = 1.0 \Rightarrow \mathcal{A}_B = \lambda/2$, $\mathcal{A}_X = \lambda/2$ and $\mathcal{A}_Y = \lambda^2/4$. With $0 < \lambda < 1$, we have $\mathcal{A}_X > \mathcal{A}_Y$.
[b] $\mathcal{A}_B = 1.0 \Rightarrow \mathcal{A}_A = 1/2M_A$, $\mathcal{A}_X = \lambda/2M_A$ and $\mathcal{A}_Y = \lambda/2$. With $M_A > 1$, we have $\mathcal{A}_Y > \mathcal{A}_X$.

PROPERTY 4 (Cancellation of Inheritance): Let A be a concept, B be an element-concept of A, A and B have element-concepts X and Y respectively, and that X and Y are contradictory. [a] If A is activated, X should squash the activity of Y. [b] If B is activated, Y should squash that of X.

Cancellation of Inheritance can be achieved by pre-organizing sets of conflicting concepts into competitive fields (Grossberg, 1973). In a competitive field where every node competes with each other for activities, the behavior can be winner-take-all or contrast-enhancing. By grouping conflicting concepts such as **gray**, **white**, **red**, and **blue** into a winner-take-all pool, only one of which has the strongest activity will survive. From property 3, X is more activated than Y in Case [a] and Y is more activated than X in Case [b]. Through competition, X will win in case [a] and Y will win in case [b].

In general, when properties are inherited through multiple paths, inheritance conflicts could occur. A typical example of *multiple inheritance* problem is the Nixon example shown below. By using fuzzy connection strengths, the concept hierarchy network is able to resolve such multiple inheritance conflict. Suppose that Nixon is a 90% Quaker and a 100% Republican, these fuzzy relationships can be captured in their template weights. Assuming non-fuzziness for other relations, it can be verified that the activation of **pacifist** is $0.9\lambda^2/4$ and that of **non-pacifist** is $\lambda^2/4$, which means that Nixon is more likely to be a **non-pacifist**. In this problem, the use of fuzzy connection strengths provides an extra degree of freedom in resolving subtle situations.

Box 1. The Nixon Multiple Inheritance Problem: Is Nixon a pacifist or non-pacifist ?

Nixon : Quaker	Quaker : pacifist
Nixon : Republican	Republican : non-pacifist

Concluding Remarks

We have shown how an adaptive resonance theory (ART) based model can be used to represent and learn object concept hierarchy. By a spreading activation procedure and by organizing conflicting concepts into competitive fields, the model is capable of performing property inheritance and resolving inheritance conflicts. The work presented here, of course, only serves as a starting point in understanding and modeling semantic knowledge from an adaptive coding approach. The model faces an immediate and more challenging question on how to represent structural concepts which involves handling of role/filler relationships. It is the authors' intention that by extending from a more intuitive model which has captured some flavors of human knowledge learning, one has a better chance of modeling human intelligence.

Acknowledgement

The authors would like to thank Professor Carole Hafner of the College of Computer Science, Northeastern University, for providing valuable suggestions and ideas behind the writing of this paper.

References

Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., and Rosen, D.B. 1992. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions in Neural Networks*, **3**, 698-713.

Fahlman, S. 1979. *NETL: A System for Representing and Using Real-World Knowledge*, MIT Press.

Feldman, J.A. 1989. Connectionist representation of concepts. *Connectionism in perspective*, North-Holland, 25-45.

Grossberg, S. 1973. Contour Enhancement, Short Term Memory, and Constancies in Reverberating Neural Networks. *Studies of Mind and Brain*, Chapter 8, D.Reidel Publishing Company.

Shastri, L. 1988. A Connectionist Approach to Knowledge Representation and Limited Inference. *Cognitive Science*, **12**, 331-392.

Sun, R. In press. An efficient feature-based connectionist Inheritance Scheme. *IEEE Transaction on System, Man and Cybernetics.*

Tan, A.H. 1992. Adaptive Resonance Associative Map: A hierarchical ART system for fast stable associative learning. In *proceedings, IJCNN*, Baltimore, Maryland, June 17-21, Vol I, 860-865.

Touretzky, D. 1986. *The Mathematics of Inheritance Systems*, San Mateo, CA: Morgan Kaufman.