A Model of Schema Selection Using Marker Passing and Connectionist Spreading Activation

Hon Wai Chun
Honeywell Bull
300 Concord Road, MA30-895A
Billerica, MA 01821-4186
U.S.A.

Alejandro Mimo
Gold Hill Computers
163 Harvard Street
Cambridge, MA 02139
U.S.A.

ABSTRACT

Schema selection involves determining which pre-stored schema best matches the current input. Traditional serial approaches utilize a match/predict cycle which is heavily dependent upon backtracking. This paper presents a parallel interactive model of schema selection called SAMPAN which is more flexible and adaptive. SAMPAN is a hybrid system that combines marker passing with connectionist spreading activation to provide a highly malleable and general representation for schema selection. This work is motivated by recent success in connectionist schema representations and in natural language marker passing systems. A connectionist schema representation provides many attractive features over traditional schema representations. However, a pure connectionist representation lacks generality; new propositions cannot easily be represented. SAMPAN gets around this problem by using marker passing to perform variable binding on generalized concepts. The SAMPAN system is a constraint satisfaction network with nodes that perform simple pattern matching and input summation. This approach is directly applicable to current schema-based systems.

I. INTRODUCTION

The most difficult task in schema-driven systems[1] is to select a schema that can accurately represent the current input. A typical serial approach to schema selection first extracts cues (in the form of settings, participants, events, etc.) from the input and then uses these cues to help select an appropriate schema. If the selected schema is later shown to be incorrect, the system must backtrack and select another schema. Once a schema is selected, top-down expectation or prediction can be used to assist recognition or natural language understanding. For example, schema or script selection in SAM [SCHA75] involves matching the input with a list of conceptualizations stored in each script. If the input matches, this script will be activated. SAM uses a search list that is dynamically updated to indicate which scripts are most likely to occur based on the conceptualizations or objects seen so far. FRUMP [DEJO79], on the other hand, processes the first paragraph to select a sketchy script. For efficiency, sketchy scripts are matched using a n-way branching discrimination tree.

There are several problems with current serial schema selection techniques [HAVE83]. When a system selects an incorrect schema, it must backtrack, retry another schema, and possibly reinterpret the inputs. For a system with a large number of schemata, this computational overhead may greatly reduce its effectiveness. In addition, there is a "chicken and egg problem." The system cannot make use of schema knowledge while interpreting the input to perform

1. Throughout this paper, the terms schemata, frames and scripts are used interchangeably to represent knowledge structures that describe typical situations or objects. The terms massively parallel networks and connectionist networks are also used interchangeably to represent the notion of large parallel networks with many simple processing elements.

schema selection. On the other hand, the system must hypothesize a correct schema before the material it is trying to explain is fully given. There is also a "match or near miss" problem, in which the system must decide whether an adequate match has been made or to continue searching. Although various index and search strategies have been used to improve the selection process, there is still no elegant approach that performs schema selection efficiently. This overall process of backtracking and reinterpreting inputs happens rarely in human perception and interpretation, suggesting that AI's heavy dependence on heuristic search might be inappropriate.

Recently, several connectionist schema representations have been introduced [RUME86, SHAR86] offering solutions to these problems. The parallel nature of connectionist networks avoids the "match or near miss" problem since all schemata are "matched" simultaneously. In addition, in [CHUN85, RUME86], bottom-up schema activation and top-down prediction generation are considered as continuous processes rather than discrete match, predict and backtrack stages. Several schemata may be active at any one time to provide predictions with varying certainty level. Hence, the "chicken and egg problem" disappears since schema knowledge is continuously available during input interpretation. The connectionist spreading activation also provides a default mechanism that is *multivariate distributed* [RUME86] in which defaults are determined by values filling other slots. Charniak [CHAR86a] indicated that the connectionist default mechanism avoided redundancy found in the *square formation* problem where redundant information is generated due to many levels of generality. Connectionist network also provides a "resonance effect" in which activated slots of the same schema reinforce themselves and strengthen the selection of that schema.

However, a pure connectionist representation for schemata lacks generality. For example, a system may have a node that represents "a person went home." However, in order to understand a particular instance, such as "Mary went home," the system must also have this node encoded beforehand. In other words, any possible combination of schema structures with their instantiated bindings must be encoded into the network structure before the network can understand them. Several approaches [MCCL86, TOUR85] have been proposed to solve this problem of representing new propositions or *variable binding*. Although these approaches provide connectionist networks with more flexibility, the set of all possible inputs must still be known beforehand. In addition, they tend to lose the simple structure of connectionist networks which makes them attractive.

SAMPAN maintains the elegance and simplicity of distributed connectionist networks yet solves the variable binding problem by augmenting the system with marker passing capabilities and nodes that represent generalized concepts. The SAMPAN model itself, is influenced by the PDP schema representation [RUME86], the NEXUS dictionary structure [ALTE85], and Wimp's marker passing algorithm [CHAR86]. SAMPAN integrates features found in these recent representational paradigms to construct a model of schema selection that is highly flexible, general and dynamic:

— Connectionist networks are very good at completing patterns from partial information. SAMPAN uses this representation to encode general schema knowledge as well as perform schema activation and prediction generation.

— Marker passing algorithms have proven successful as a method of sharing and locating information. SAMPAN uses a marker passing algorithm similar to that of Wimp [CHAR86] to bind variables.

— In order to represent general episodic events and concepts, SAMPAN uses nodes which are conceptual templates similar to those found in NEXUS's dictionary network.

SAMPAN is relevant to many AI applications, such as natural language understanding (expectation-driven processing), recognition (fill partial patterns with defaults), learning (recognize unexpected events), and run-time plan generation (through expectation). This paper concentrates mainly on the problem of schema selection.

II. OVERVIEW OF SAMPAN

SAMPAN networks are massively parallel constraint satisfaction networks. Nodes represent concept templates and links represent associations between concepts. Communication is in the form of marker passing combined with connectionist value passing. Information passed consists of purely local information. The processing within a SAMPAN network is totally parallel without global procedures.

To demonstrate SAMPAN, a script-driven natural language understanding system is used as an example. In this paper, SAMPAN performs schema selection and generates expectations which can be used by a natural language understanding system. In other words, the SAMPAN network acts as an active schema knowledge base. Input to SAMPAN consists of pre-parsed internal representations. The output of SAMPAN is a stream of expected conceptualizations. The state of the SAMPAN network at any one time represents an emergent schema structure. Currently, SAMPAN is not interfaced to an actual parser. However, the example can still adequately illustrate SAMPAN's performance, since only the pre-parsed input is needed. In the future, we are interested in exploring a tighter integration between the SAMPAN network and a massively parallel parser.

The next section introduces the connectionist schema representation in detail. This is followed by a description of the marker passing used and computations involved. Explanation of the test domain is then given.

III. DISTRIBUTED SCHEMA REPRESENTATION

The SAMPAN network, in terms of its connectionist structure, is adapted from the PDP distributed schema representation [RUME86]. The structure is a constraint satisfaction network of nodes that represents schema slots. A coalition of active nodes represents a schema. In [RUME86], schemata were used to represent different room types (e.g. kitchen, living room, etc.) and slots to represent room descriptors (e.g. ceiling, coffee cup, refrigerator, etc.). In SAMPAN, schemata represent episodic events and slots represent generalized concepts (or event/state concept [ALTE85] such as "a person went to a restaurant"). The representation is distributed since each schema is represented not as a single node but as the activation of a collection of nodes.

Through this connectionist representation, concepts that usually occur together will tend to support each other, while suppressing those that do not. In other words, nodes that are usually active or inactive at the same time will have an activating link between them. In the opposite case, an inhibitory link will be found. In SAMPAN, nodes can be activated either by the parser (analogous to "clamping") or by other nodes through the input links. Links are bi-directional and the weights are symmetrical. Unlike marker passing networks, the links are not labeled and

are used solely to modify the information passed through them. For simplicity, the weights are set according to a Bayesian probability measurement [RUME86] (although learning rules can be used as well):

$$w_{ij} = -\ln \frac{P(x_i{=}0 \ \& \ x_j{=}1)\ P(x_i{=}1 \ \& \ x_j{=}0)}{P(x_i{=}1 \ \& \ x_j{=}1)\ P(x_i{=}0 \ \& \ x_j{=}0)}$$

where w_{ij} is the weight of the link between nodes i and j, $P(x_i{=}0 \ \& \ x_j{=}1)$ is the probability that the activation level of node i is 0 and the activation level of node j is 1, and so on.

The result of input activation to this network is a gradual emergence of a coalition of active nodes. This coalition represents a schema structure. This approach is reminiscent of *event concept coherence* presented by Alterman [ALTE85] where concepts are coherent if their positions in the net are *proximal.* In the SAMPAN network, concepts are coherent if they are connected by activating links (links with a positive weight).

There are several representational advantages to this distributed model. Traditional schema structures are often rigid and static, lacking in flexibility and generativity. Although, scripts may have tracks to represent variations or have embedded scripts to represent hierarchical structure, this knowledge is still rigidly defined. In a distributed connectionist representation, a schema is not an explicit representational entity, but an emergent structure based on the events seen so far. The result is a very malleable representation that is highly dynamic and adaptive.

SAMPAN networks are used differently from that of the PDP schema representation. Instead of "clamping" all the inputs at the same time, inputs to SAMPAN networks are fed sequentially. This permits a natural language system which processes sentences sequentially to interpret each sentence in the context of what has been processed before. In addition, instead of representing a schema as the stable coalition of nodes (where all the nodes of a particular schema will eventually be fully activated), SAMPAN allows nodes to decay in the process of schema selection. In this case, a SAMPAN schema is a "dynamic" coalition of nodes that reflects recent inputs and the expectations generated from them. This permits SAMPAN to continuously process input stories without having to first clear network activity after each schema. Otherwise, the network might exhibit a "heat death" phenomenon where too many concepts would be activated or expected.

IV. COMPUTATION OF SAMPAN

This section describes the massively parallel computation involved in schema selection and contextual expectation generation. First, the internal structure of SAMPAN nodes is presented. Next, the pattern matching procedure that is invoked by inputs is introduced. Finally, the relaxation function that is evaluated at each relaxation cycle is described.

Node Structure

In contrast to a connectionist node that contains a memory of its activation level and its internal parameters, a SAMPAN node contains a memory of its local variable bindings augmented with activation levels and a conceptual template. Each node in SAMPAN represents a general concept. Concept templates are in the form:

(*Primitive* (*Role1 Filler1*) (*Role2 Filler2*) ...)

where *Primitive* is a semantic primitive, *Role* is a case structure for *Primitive* and *Filler* is its default value. After each filler, there may be property value pairs to describe the filler. In this paper, the internal representation used for concepts is based upon Schank's [SCHA77] Conceptual Dependency (CD) theory. The choice of internal representation is not crucial to the overall SAMPAN model; other similar representations will also suffice.

For example, a node that represents "a person went to a store" will have the following conceptual template stored:

(PTRANS (ACTOR ?PERSON) (OBJECT ?PERSON) (TO ?STORE))

This will match the input "John went to K-MART":

(PTRANS (ACTOR JOHN-1) (OBJECT JOHN-1) (TO K-MART-1))

Pattern Matching

Pattern matching occurs whenever a node receives direct input from the natural language system (as opposed to inputs from other nodes). When the concept template matches an input, the node creates a mark in the form of a binding list. A SAMPAN mark contains a list of variable bindings augmented with an activation level:

[(*Variable1 Value1 Activation1*) (*Variable2 Value2 Activation2*) ...]

In the above example, the resulting mark will be:

((PERSON JOHN-1 1.0) (STORE K-MART-1 1.0))

Activation is a real number ranging from 0 to 1. This can be considered as a measurement of how certain a particular binding is, based upon the concepts seen so far. The activation level is set to 1.0 if the mark is generated as a result of a direct match with the natural language input. Link will attenuate this activation level as the mark is passed from node to node. This indicates that nodes which are further away from the matched concept will be expected less (or less coherent) than those which are directly connected. Charniak [CHAR86] uses a "zorch" as a rough measure of the "strength" of a mark.

Type constraint checking is also performed during pattern matching. For example, the above template is actually a shorthand notation for:

```
(PTRANS     (ACTOR ?X TYPE PERSON)
            (OBJECT ?X TYPE PERSON)
            (TO ?Y TYPE STORE))
```

where PERSON and STORE are object types. An object type hierarchy is needed so that a subclass will match a type constraint requiring its parent class. For example, an input of type PERSON should match an ANIMATE constraint. To avoid storing a type hierarchy in each node, a bit representation of sets [HILL85] is used. Each type is represented as a string of bits. For example, all types (including PERSON) under the set of ANIMATE have their first bit set. When the system tries to match ANIMATE it will only need to check for the first bit. Hierarchical information, in the form of a bit string, is thus passed with the type description. Throughout this paper, only the simplified shorthand notation will be used.

Relaxation Computation

Connectionist relaxation involves a weighted summation of the input activations. In SAMPAN, the relaxation is similar, except the inputs are in the form of activations tagged with variable bindings. When SAMPAN performs weighted summation, only activations with the same variable binding will be summed together. We call this process *marker merging*, i.e. marks with consistent bindings are merged together. In this way, inputs are merged with the binding list stored locally. The net result may be one or more binding lists if inconsistencies are found. Being able to differentiate inconsistent bindings is a feature of SAMPAN that permits multiple instantiation of the same concept template. The following example illustrates this relaxation computation:

NODE #0018:

CONCEPT TEMPLATE:
(POSS (ACTOR ?PERSON) (OBJECT ?POBJ))
LOCAL BINDINGS:
((PERSON MARY-2 0.2) (POBJ BOOK-2 0.1))
DECAY: 0.2
INPUT MARKS:
((PERSON MARY-2 0.5)) INPUT WEIGHT: 0.1
((PERSON MARY-2 0.4) (POBJ BOOK-2 0.4) (FROM JOHN-1 0.2))
INPUT WEIGHT: 0.2
((PERSON JOE-3 0.5) (POBJ CUP-3 0.5)) INPUT WEIGHT: 0.5

LOCAL BINDINGS (AFTER RELAXATION):
((PERSON MARY-2 0.29) (POBJ BOOK-2 0.16))
((PERSON JOE-3 0.25) (POBJ CUP-3 0.25))

In this example, the local binding of NODE #0018 originally contained only one binding list. After relaxation or *marker merging* with the three input marks, two local binding lists results. This indicates two instantiation of the concept template, namely:

(POSS (PERSON MARY-2) (POBJ BOOK-2)) and
(POSS (PERSON JOE-3) (POBJ CUP-3))

In addition, any variable bindings in the input marks that are not needed for the local concept template will be removed. For example the binding (FROM JOHN-1 0.2) in the above example was removed. This keeps the size of the local binding lists manageable and reduces communication overhead. Hence, only bindings required for the local concept template will be passed to other nodes. Even though only a partial binding list is passed, this should not be problematic, since each node receives inputs from many other nodes, the net bindings received collectively fill in the required local variables. Nodes will not have any outputs and hence will not spread marks to other nodes, unless all the required local variables are bound.

In SAMPAN, activated nodes provide contextual expectation. For example, whenever the activation level within a local binding list reaches a particular threshold value, the instantiated concept template can be sent to the natural language system. These expected concepts assist the natural language system in interpreting the inputs. In addition, the predictions provide default values to missing concepts.

V. EXAMPLE APPLICATION

To investigate the use of SAMPAN, a network was constructed to represent five different schemata:

restaurant schema	— general eating out scenario
theater schema	— movie, concert, opera, etc.
subway schema	— subway transportation
shopping schema	— purchasing of merchandise
sport schema	— sport related physical activity

These schemata were chosen to demonstrate the generality and flexibility of SAMPAN schema selection, since they share many common concepts (such as going out, travelling, paying for something, etc.). In addition, these schemata can exhibit many types of interactions. For example, these schemata may occur by themselves, embed with another (e.g. eating in a shopping mall), overlap with another (e.g. buy hotdog while leaving subway), or occur in parallel with another (e.g. going to a restaurant play house).

The network currently contains 70 concept templates and 4830 links. The templates encode general concepts which may be shared by several schemata. The following is a sample of the concept templates encoded into the SAMPAN network:

A person leaves his home.
A person sits on a chair.
A business establishment gives a person an object.
An employee takes a person's money.
A person picks up exercise equipment.
A person is hungry.
A person listens to music.
A person orders food.

The SAMPAN network was constructed by gathering five sets of data from each of ten different human subjects. Each subject was asked to visualize a particular scenario within each of the five schemata. The subjects then checked the list of concept templates and indicated those that were relevant to the scenario. Finally, a program constructed the SAMPAN network based upon the collected statistics.

For experimentation, new stories were constructed and feed to the network. SAMPAN correctly matches concepts it knows about in the stories and generates the appropriate expectations. The following is an example story processed by SAMPAN:

```
; John was hungry.
(IS (ACTOR JOHN-1) (STATE *HUNGER*) (VAL 0))
; John left his home to go to McDonald's.
(PTRANS (ACTOR JOHN-1) (OBJECT JOHN-1) (FROM HOME-1) (TO MCDONALDS-1))
; John told the cashier he wanted a hamburger.
(MTRANS (ACTOR JOHN-1) (OBJECT HAMBURGER-1) (TO CASHIER-1))
; John paid the cashier.
(ATRANS (ACTOR JOHN-1) (OBJECT *MONEY*) (FROM JOHN-1) (TO CASHIER-1))
; The cashier gave John the hamburger.
(ATRANS (ACTOR CASHIER-1) (OBJECT HAMBURGER-1) (FROM CASHIER-1) (TO JOHN-1))
; John sat down.
```

(PTRANS (ACTOR JOHN-1) (OBJECT JOHN-1) (TO *CHAIR*))
; John ate his hamburger.
(INGEST (ACTOR JOHN-1) (OBJECT HAMBURGER-1))
; John was happy.
(IS (ACTOR JOHN-1) (STATE *MENTAL*) (VAL 10))
; John threw away some garbage into the garbage can.
(PROPEL (ACTOR JOHN-1) (OBJECT *GARBAGE*) (TO GARBAGE-CAN-1))
; John returned home.
(PTRANS (ACTOR JOHN-1) (OBJECT JOHN-1) (FROM MCDONALDS-1) (TO HOME-1))

For each concept in the story, SAMPAN sends the concept to all the seventy nodes in the network and relaxes the network for one cycle. During the relaxation, pattern matching and marker merging was performed. Figure 1 is a screen printout from a personal computer showing the state of the SAMPAN network when the above story was processed. Each row represents the network activation at a particular cycle. Each column represents a particular node in the network. The darkness of the shading is proportional to the activation level in a node.

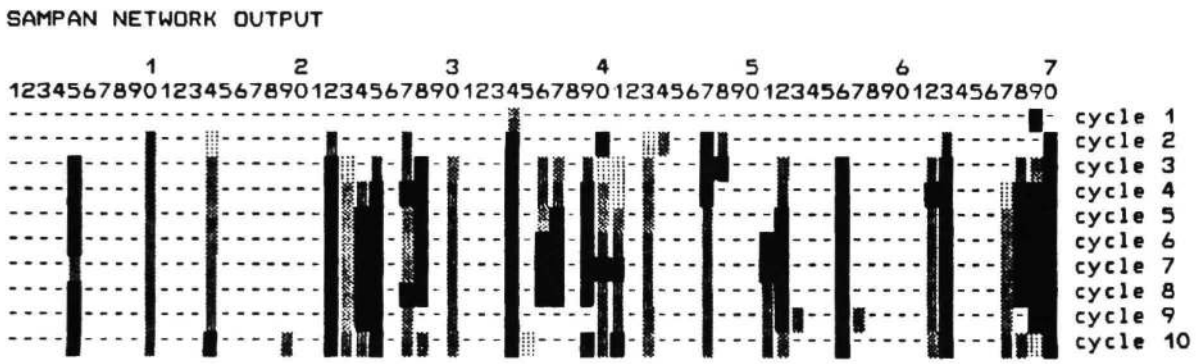

Figure 1. The activation level of the seventy nodes in the example network.

The figure shows that the network formed coalitions of concepts as it processed the story. (Due to the decay factor, some of the nodes gradually died out.) These coalitions represent the emergent schema structure. For example, after SAMPAN processed the third input:

; John told the cashier he wanted a hamburger.
(MTRANS (ACTOR JOHN-1) (OBJECT HAMBURGER-1) (TO CASHIER-1))

activation and bindings were spread to the nodes representing: John having to wait, a cashier giving John a hamburger, John holding a hamburger, John eating a hamburger, John leaving McDonald's to go home, etc. These expected concepts can be used by an expectation-driven natural language system or can help fill in missing concepts not provided by the input.

The example network also handles simple pronoun references. For example, if the concept:

(MTRANS (OBJECT HAMBURGER-1) (TO CASHIER-1))

was used instead of the original third input, the network fills in the ACTOR role to be JOHN-1 through relaxation. However, in complicated stories involving many actors, inference rules need to be used to resolve pronoun references. These inference rules can be encoded in the links as constraints on how variables should be mapped as they are passed from one node to another.

For example, the ACTOR of one concept template might be restricted to be the OBJECT of another.

Summarizing, SAMPAN is able to form coalitions of related concepts that represented the schema structures. The network is flexible enough to accommodate many variations of the same schema as well as novel event sequences. Stories with multiple schemata (embedded or overlapped) are also processed without any difficulty.

SAMPAN was implemented using the MicroAINET language [CHUN86b] written in Golden Common Lisp. MicroAINET is a language to define, construct, and simulate a general class of massively parallel networks. The experiments were performed using 386-based hardware.

VI. RESEARCH DIRECTIONS

This paper represents the first step in providing a massively parallel schema representation. Only the problem of schema selection has been discussed in detail in this paper. We are interested in exploring other issues such as the learning of new structures, the storage of new schema instances, and the integration with existing memory structures [SCHA82]. Another issue that needs to be addressed is the representation of temporal constraints and relationships (we have already done some preliminary work in this area [CHUN86a]). We also hope to integrate SAMPAN with a massively parallel natural language understanding system [LI87]. The SAMPAN marker passing has great potential of providing an inference mechanism and explanation facility within the network (possibly similar to that of Wimp [CHAR86]).

VII. SUMMARY

This paper presented a massively parallel model of schema selection which solved many problems inherent in serial approaches. The selection process is more flexible and adaptive than present methods used. In addition, the resulting schema structure is more malleable and general. SAMPAN has been tested successfully in the domain of natural language understanding.

ACKNOWLEDGEMENTS

The authors would like to thank the members of the Knowledge Engineering Center and Gold Hill Computers for their interest and comments on this research.

REFERENCES

[ALTE85] Alterman, R., "A Dictionary Based on Concept Coherence," *Artificial Intelligence*, Volume 25 Number 2, February 1985, pp.153-186.

[CHAR86a] Charniak, E., "A Neat theory of Marker Passing," In *Proceedings of AAAI-86*, 1986, pp.584-588.

[CHAR86b] Charniak, E., "Connectionism and Explanation," In *Proceedings of TINLAP 3*, 1986.

[CHUN86a] Chun, H.W., "A Representation for Temporal Sequence and Duration in Massively Parallel Networks: Exploiting Link Interactions," In *Proceedings of AAAI-86*, August 1986.

[CHUN86b] Chun, H.W., "The MicroAINET Language Manual," unpublished draft, 1986.

[CHUN85] Chun, H.W., "Schemata in a Massively Parallel Model of Computation: A Non-Committing Approach to Schema Selection," unpublished Ph.D. thesis proposal, University of Illinois at Urbana-Champaign, 1985.

[DEJO79] DeJong, G., *Skimming Stories in Real Time*, Ph.D. Thesis, Yale University, New Haven, CT, 1979.

[HAVE83] Havens, W.S., "Recognition Mechanisms for Schema-based Knowledge Representations," in N. Cercone (ed.), *Computational Linguistics*, Pergamon Press Limited, 1983.

[HILL85] Hillis, W.D., *The Connection Machine*, MIT Press, Cambridge, 1985.

[LI87] Li, Tangqiu and H.W. Chun, "Massively Parallel Network-based Natural Language Parsing System," In *IEEE-Proceedings of the Second International Conference on Computers and Applications*, Beijing, China, June 1987.

[MCCL86] McClelland, J.L. and A.H. Kawamoto, "Mechanisms of Sentence Processing: Assigning Roles to Constituents," in J.L. McClelland, D.E. Rumelhart (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2: Psychological and Biological Models.*, Cambrdige, MA: Bradford Books/ MIT Press, 1986.

[RUME86] Rumelhart, D.E., P. Smolensky, J.L. McClelland, and G.E. Hinton, "Schemata and Sequential Thought Processes in PDP Models," in J.L. McClelland, D.E. Rumelhart (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2: Psychological and Biological Models.*, Cambrdige, MA: Bradford Books/ MIT Press, 1986.

[SCHA82] Schank, R., *Dynamic Memory*, Cambridge University Press, 1982.

[SCHA77] Schank, R. and R. Abelson, *Scripts, Plans, Goals and Understanding*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

[SCHA75] Schank, R.C. and Yale A.I. Project, "SAM: a story understander," Yale Computer Science Department Research Report 43, New Haven CT., 1975.

[SHAR86] Sharkey, N.E., R.F.E. Sutcliffe and W.R. Wobcke, "Mixing Binary and Continuous Connection Schemes for Knowledge Access," In *Proceedings of AAAI-86*, 1986, pp.262-266.

[TOUR85] Touretzky, D.S. and G.E. Hinton, "Symbols Among the Neurons: Details of a Connectionist Inference Architecture," In *Proceedings of IJCAI-85*, 1985, pp.238-242.