From: AAAI-86 Proceedings. Copyright ©1986, AAAI (www.aaai.org). All rights reserved.

A REPRESENTATION FOR TEMPORAL SEQUENCE AND DURATION IN MASSIVELY PARALLEL NETWORKS: Exploiting Link Interactions

Hon Wai Chun
Computer Science Department
Brandeis University
Waltham, MA 02254
U.S.A.

ABSTRACT

One of the major representational problems in massively parallel or connectionist models is the difficulty of representing temporal constraints. Temporal constraints are important and crucial sources of information for event perception in general. This paper describes a novel scheme which provides massively parallel models with the ability to represent and recognize temporal constraints such as sequence and duration by exploiting link to link interactions. This relatively unexplored yet powerful mechanism is used to represent rule-like constraints and behaviors. The temporal sequence of a set of nodes is defined as the constraints or the temporal context in which these nodes should be activated. This representation is quite robust in the sense that it captures subtleties in both the strength and scope (order) of temporal constraints. Duration is also represented using a similar mechanism. The duration of a concept is represented as a memory trace of the activation of this concept. The state of this trace can be used to generate a fuzzy set like classification of the duration.

I. INTRODUCTION

Massively parallel models of computation [1, 2] (also known as connectionist, parallel distributed processing, or interactive activation models) consist of large networks of simple processing elements with emergent collective abilities. The behavior of such networks has been shown to closely match human cognition in many tasks, such as natural language understanding and parsing [3], speech perception and recognition [4, 5, 6], speech generation, physical skill modeling, vision and many others. The use of such models provides cognitive scientists with experiment and simulation results in a finer level of detail than was previously possible. In addition, various learning algorithms [7, 8, 9] have been developed to enable these networks to acquire knowledge through gradual adaptation. The distributed nature of some of these models enables network structures to be less sensitive to structural damages.

One of the major representational problems in massively parallel models is the difficulty in representing temporal constraints. These are constraints which control network activation based on temporal knowledge. Temporal constraints are important and crucial sources of information for event perception in general. This is especially true for tasks such as speech perception and schema selection. This paper describes a novel scheme which provides connectionist models with the ability to represent and recognize temporal constraints such as sequence and duration by exploiting link to link interactions.

II. OVERVIEW

Sequential constraints on a set of events is one form of temporal constraints. There are basically two types of knowledge about temporal sequences. One is how to *generate* a sequence of activations. The second is how to *recognize* a given sequence of events. Figure 1 shows a simple network structure [1] which can activate a predetermined sequence of events (E1 → E2 → ...). This type of network structure is useful in modeling schema execution such as physical motor control.

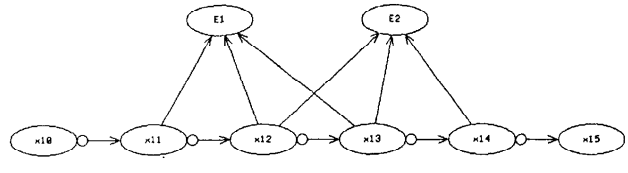


Figure 1. Activating a sequence of events.

The second type of temporal sequence knowledge (recognizing a sequence of events) is somewhat more difficult to represent. The main problem is constructing a network structure which can represent temporal contexts in which nodes should be activated. A mechanism is needed which could recognize particular sequences of activation patterns over time. This problem arises most notably in modeling speech perception. For example, a word should be recognized only if its constituent acoustic segments are heard in the correct sequence.

One approach to this problem is to construct time buffered network structures in which each time slot contains an identical substructure [1, 6]. Conceptually, this type of network contains a copy of the event to be recognized at each particular slot in time in which this event might occur. There are many problems associated with this type of duplicated structures (either pre-wired or dynamic). Various techniques of dynamic structures and connections [10, 11] have been suggested to partly handle these problems. However, there is still an overhead in both computation and memory when using this approach.

Another approach is to approximate sequence constraints through lateral priming [5]; priming events that follow those that have already been activated. Using the example of word perception, this approach is only useful if there are several words with similar acoustic segments. In this case, the priming will help activate the most plausible word. However, in general, this approach will not inhibit the perception of a word when segments are given out of sequence.

This paper presents an alternative network representation which can recognize particular temporal sequences without using duplicated or dynamic structures. This paradigm is further extended to form a representation of temporal duration.

III. METHODOLOGY

The crux of this paper centers around the notion of representing interactions among the links to define temporal constraints. This idea is remotely analogous to neural networks where synapses may be made between different parts of the neuron; for example between axon and dendrite, axon and axon, dendrite and dendrite, and axon and cell body [12]. In the model presented here, links may be made between node and node, as well as between node and link.

The approach uses two types of link interaction. Figure 2 (a) shows one type in which the activation-flow from one node to another (node A to B) is *preconditioned* by a third node (node C). The association between nodes A and B can be formed only if node C is also activated. Figure 2 (b) shows how this interaction is implemented. The intersection of the two links is represented by a CONSTRAINT node. This node implements the computation (described in detail in the next section) required to represent the link interactions. Letter "P", in figure 2(b), indicates the *precondition* input link.

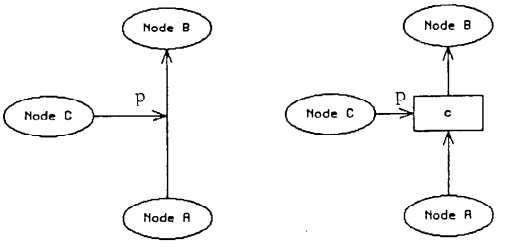


Figure 2. Precondition link.

The second type of interaction (figure 3) has just the reverse effect. Activation flows from node A to B, unless node C is already activated. In other words, node C inhibits the association between nodes A and B to occur. The inhibiting link is called the *exception* input and is labeled with an "E" at the CONSTRAINT node.

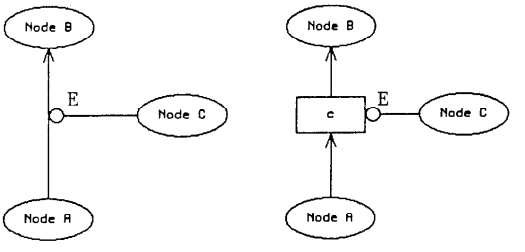


Figure 3. Exception link.

These two relatively unexplored yet powerful mechanisms can be used to represent rule-like constraints and behaviors. Link interactions represented by the CONSTRAINT node are, by no means, a simple binary enable/disable type mechanism. The CONSTRAINT node is robust enough to a capture a continuum of interactions. For instance, the CONSTRAINT node scales the activation flow between node A and B according to the activation level of the node C. In the *precondition* case, to account for noisy environments, a CONSTRAINT node can be programmed (by setting a positive *bias level*) to allow energy to trickle through, from node A to B, even when the node C is not activated. This *bias level* can be thought of as defining the fuzziness or looseness of the precondition. The CONSTRAINT node reduces to a BINDER node [2] when the *bias level* is zero and there are no *exception* inputs.

A. The Computation of the CONSTRAINT Node

Although the CONSTRAINT node was originally designed to represent link interactions, the computation is general enough to be used as a normal node in the network simply by not having *precondition* or *exception*

inputs. Thus allowing a uniform node unit to be used throughout the network to represent nodes as well as link interactions. The following describes the computation of the CONSTRAINT node used in our simulations:

The main energy parameters of this node are:

p – the *potential*, and
v – the *output value*.

These depend on the following internal parameters:

t – the *threshold*
b – the *bias level*
d – the *decay*
$\mathbf{i}$ – vector of *normal inputs* $i_1,\ldots,i_n$
$\mathbf{P}$ – vector of *precondition inputs* $p_1,\ldots,p_n$
$\mathbf{E}$ – vector of *exception inputs* $e_1,\ldots,e_n$

The functions to compute the new values are:

$$p \leftarrow f(\mathbf{i},\mathbf{P},\mathbf{E},p,b,d),$$
$$v \leftarrow g(p,t).$$

In the case where the CONSTRAINT node is used to represent a normal node in the network (i.e. when **P** and **E** is absent), the computation involved is defined as:

$$p \leftarrow p(1-d) + \Sigma(w_k \times i_k) \quad [0 \le w_k \le 1]$$
$$v \leftarrow \text{if } p > t \text{ then p else } 0,$$

where w_k is the link weight on the link i_k and p and v are continuous values in [0, 1]. The *potential* is simply the previous *potential* scaled by a *decay* factor, plus the summation of weighted inputs. The *output* is equal to p with a *threshold*.

When the CONSTRAINT node is used to represent link interactions (i.e. with **P** or **E**), p becomes:

$$p \leftarrow [\Sigma(w_k \times p_k) - \Sigma(w_k \times e_k) + b] \times p',$$

where p' is the *potential* without **P** and **E**. The new *potential* is equal to p' scaled by the difference in the weighted inputs of **P** and **E** plus the *bias level*.

B. Characteristics of the CONSTRAINT Node

To give a flavor of the type of constraints that can be represented using the CONSTRAINT node, a simple example from the monkey-and-banana problem is shown. In this problem, one of the rules might be expressed as:

if [goal = possess-banana] **then** [action = grasp-banana].
precondition: [location = at-banana].
exception: [state = grasped-banana].

Figure 4 shows how this can be represented in network form. The CONSTRAINT node permits the association between the goal to possess banana and the action to grasp banana only if the monkey is at the banana location and have not already grasped the banana.

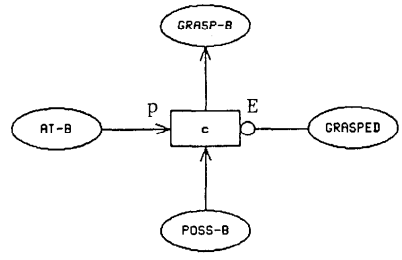


Figure 4. Monkey and banana.

Imagine that, initially, the monkey wants to possess the banana but is not at the banana location. Although, the POSS-B node is activated, there is no energy flow to the GRASP-B node (since the precondition is not present). The *bias level* of the CONSTRAINT node in this case is zero since we want a strict precondition (i.e. the monkey cannot start to grasp the banana unless it is positively at the banana location). As the monkey moves towards the banana, AT-B node gets activated (perhaps through visual perception) and energy gradually flows to the GRASP-B node which triggers the grasping action. As the monkey performs this action, the GRASPED node is activated and inhibits the grasping action to continue. The GRASP-B node is activated only after the precondition AT-B is activated and is deactivated after the exception GRASPED. The CONSTRAINT node in this example behaves like a feedback mechanism to control physical motion.

IV. TEMPORAL SEQUENCE

The network structure used to represent/recognize a temporal sequence is conceptually an extension to the structure used in the monkey-and-banana problem. Figure 5 shows how a schema (S1), which consists of a sequence of three events (E1 → E2 → E3), can be recognized using CONSTRAINT nodes to represent link interactions.

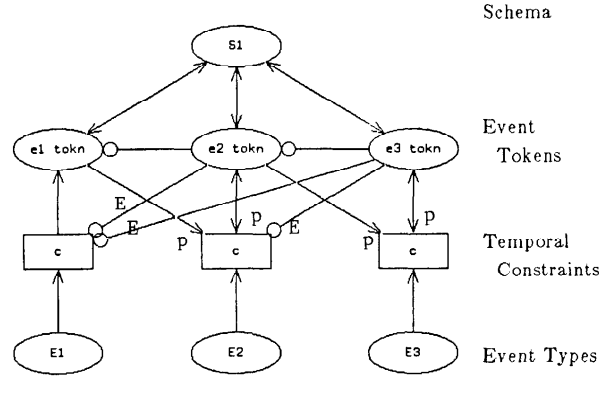


Figure 5. Representation for temporal sequence.

The lowest layer of this network structure consists of the *type* nodes for the various events. The *type* is accessible by other schema structures as well. Above this is the *temporal constraint* layer which restricts a particular temporal ordering of the *types* to occur before the event *tokens* can be activated. The schema structure is represented as a set of event *tokens*. The schema node remains activated, with a varying *potential* level, as long as the input events follow the sequence defined in the *temporal constraint* layer. For example, after E2 is activated, the token for E2 will be activated only if the token of the preceding event (E1) is already activated and the token of the next event (E3) has not yet been activated. In other words, the token for E2 will only be activated after E1 and before E3.

Not only can this schema structure recognize a strict sequence of events, it is also flexible enough to accommodate variations such as missing events or events which are only weakly present. For example, in the case of word recognition, a particular instance of a phoneme might be present only in a very weak form or even absent, since speech input is often incomplete or only partially specified. There are three parameters which can adjust the overall behavior of the temporal constraints to accommodate for these variations. First, there is the *bias level* in the CONSTRAINT node. A non-zero *bias level* allows energy to trickle through even in the absence of the precondition input. This is useful for cases where the precondition events may not always be present. Second, there is the *strength* of the precondition inputs (i.e. the link weight on the precondition input links). This weight reflects the probability that the precondition events would lead to the current event. The third parameter is the *scope* of the temporal constraints. It can be adjusted by having higher order dependencies for precondition and exception inputs (i.e. not only dependencies on the preceding and next node but also on a larger temporal scope). In our experiments, these parameters were adjusted manually. However, it is conceivable that a learning algorithm can be used to fine-tune the parameters based on experience.

This model suits word recognition quite appropriately. In this case, the schema is a word in the lexicon and the events are phonemes. The model has been used successfully in constructing a word recognition system which recognizes alphabets and digits spoken by a single speaker [13]. The massively parallel computation was simulated on a Symbolics Lisp machines using the AINET-2 simulator [14]. This system represents one of the first to successfully use real speech data on a speech system based on a massively parallel model. One of the main reasons for its accomplishments is the robustness in temporal representation which this model can capture.

The behavior of this model is similarly, in certain aspects, to a discrete Markov chain model. Both models traverse a discrete number of states over a discrete number of time intervals. The link weights on the *precondition* inputs in the temporal sequence model reflects the state transition probabilities in the Markov chain model. However, the state transition in the temporal sequence model is gradual over time, unlike the Markov chain. This permits smooth graceful transitions between states based on the gradual accumulation of evidence to support the transition. The temporal sequence model also allows for higher order dependencies other than just the previous state.

V. TEMPORAL DURATION

The network structure which represents/recognizes temporal duration uses a similar temporal constraint mechanism based on link interactions. The duration of a concept is represented as a memory trace of the activation of this concept.

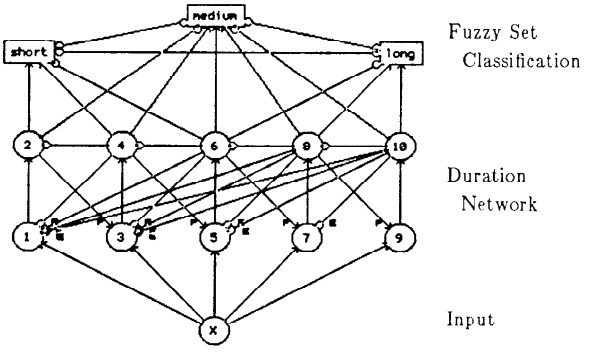
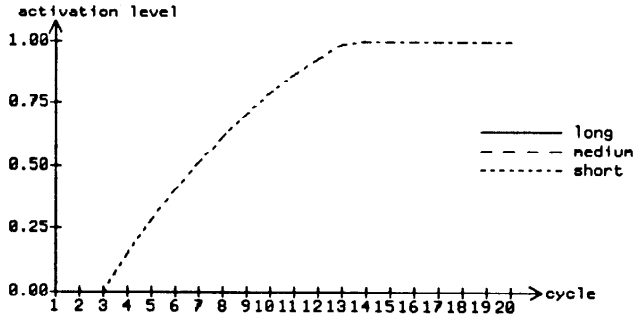


Figure 6. Representation for temporal duration.

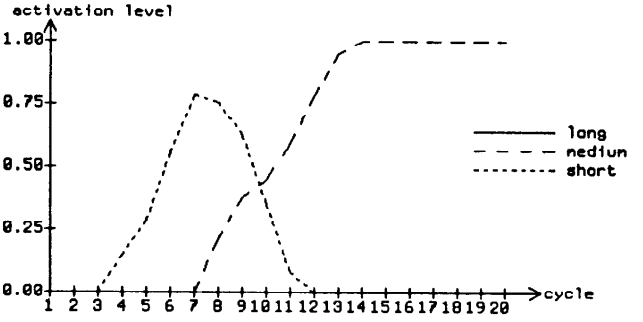
Figure 6 shows the network structure which can measure the duration of the activation of the node "x". Activation energy in this example network gradually spreads from the node labeled "1" (short duration) to the node labeled "10" (long duration) as node "x" remains activated. This is similar to a memory trace of how long "x" was active. The state of this trace can be used to generate a fuzzy-set like classification of the duration. Figure 7 shows the different activation plots for the different durations of "x".

Although knowledge of temporal duration is extremely important in event perception, this source of information has been seriously lacking before in massively parallel models. One example, in speech recognition, in which duration information is useful is in the distinction between the letters "b" and "v". If a labial stop is long, it is more likely to be a labial fricative that has been misidentified as a labial stop.

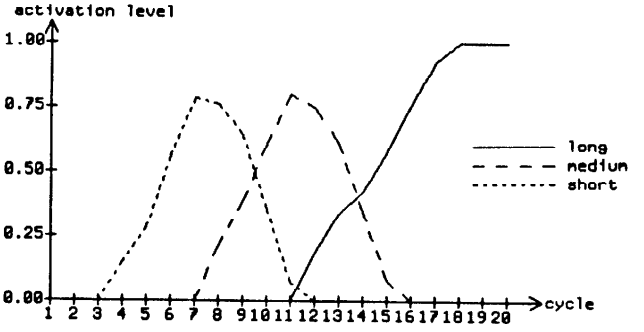
The example network structure shown above only measures the absolute duration according to a fixed scale. For relative duration, one needs an indication of the context to shift the attention of the fuzzy classification network.



(a) The activation plot for a short "x" input.



(b) The activation plot for a medium "x" input.



(c) The activation plot for a long "x" input.

Figure 7. Activation plot showing fuzzy classification generated by the duration network for the node "x'.

VI. SUMMARY

This paper presented a novel approach towards representing temporal sequence and duration based on capturing interactions among links. This model fills a much needed void in present massively parallel models. In addition, the mechanism seems capable of representing a wide variety of constraints other than temporal constraints.

Future research may include developing learning algorithms which could gradually improve or learn the link weights in highly structured networks such as those described in this paper. At present, this lack of suitable learning algorithms, is the main limitation of this model. Another interesting topic is to explore how temporal constraints might fall out from learning in random or semi-random distributed networks. One recent approach uses the "error propagation" learning algorithm [8] on a recurrent distributed network to learn to complete sequences, i.e. given the initial part of a sequence, the network generates the rest of this sequence. However, network structures that result from this process seem to be less flexible in accepting distorted inputs or inputs with varying durations.

ACKNOWLEDGEMENTS

I would like to sincerely thank Dave Waltz and the members of the Brandeis AI Group for their comments on this research, Tangqiu Li for his refreshing Eastern point of view, and Maurice Wong for cooperating with me in using real speech data to test the concepts described in this paper.

REFERENCES

[1] Feldman, J.A. and D.H. Ballard, "Connectionist Models and Their Properties," *Cognitive Science*, 6, 1982.

[2] Shastri, L. and J.A. Feldman, "Semantic Networks and Neural Nets," CS Dept., The University of Rochester, TR131, 1984.

[3] Waltz, D.L. and J.B. Pollack, "Massively Parallel Parsing," *Cognitive Science*, Vol. 9, No. 1, 1985.

[4] Chun, H.W., T. Li, J. Peng and X. Zhang "Massively Parallel Approach to Chinese Speech Processing Problems," *IEEE - Academica Sinica Workshop on Acoustics, Speech, and Signal Processing*, Beijing, China, April 1986.

[5] Elman, J.L. and J.L. McClelland, "Speech Perception as a Cognitive Process," In N. Lass (ed.), *Speech and Language: Vol. X.*, Orlando, Florida, Academic, 1984.

[6] McClelland, J.L. and J.L. Elman, "The TRACE Model of Speech Perception," *Cognitive Psychology*, 18, 1986.

[7] Hinton, G.E., T.J. Sejnowski and D.H. Ackley, "Boltzmann Machines: Constraint Satisfaction Networks that Learn," Technical Report CMU-CS-84-119, Department of Computer Science, Carnegie-Mellon University, May, 1984.

[8] Rumelhart, D.E., G.E. Hinton and R.J. Williams, "Learning Internal Representations by Error Propagation," in D.E. Rumelhart and J.L. McClelland (eds.), *Parallel Distributed Processing. Vol. 1*, MIT Press, 1986.

[9] Kirkpatrick, S., C.D. Gelatt,Jr., and M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol.220, No.4598, May, 1983.

[10] Feldman, J.A., "Dynamic Connections in Neural Networks," *Biological Cybernetics*, 46, 1982.

[11] McClelland, J.L., "Putting Knowledge in Its Place," *Cognitive Science*, Volume 9, Number 1, January-March, 1985.

[12] Stevens, C.F., "The Neuron," *Scientific American*, Volume 241, Number 3, September, 1979.

[13] Wong, K.M. and H.W. Chun, "Toward a Massively Parallel System for Word Recognition," *1986 IEEE - International Conference on Acoustics, Speech, and Signal Processing*, 1986.

[14] Chun, H.W., "AINET-2 User's Manual," CS Department, Brandeis University, Technical Report, CS-86-126, 1986.