自 然 言 語 処 理 78－2
（1 9 9 0.　7.　1 9）

超並列制約伝播による自然言語処理の手法

苫米地 英人

ATR 自動翻訳電話研究所

梗概

記号的および非記号的な制約の超並列活性化ネットワーク上での伝播による自然言語処理の手法について述べる。この手法は既存の活性化マーカ伝播による方法と異なり、複雑な言語的制約を必要とする言語現象を扱うことを可能とする。また、軽量並列プロセスを利用することにより並列計算機上に超並列制約伝播処理を実現する手法を示す。この手法により、密結合共有メモリ型並列計算機上にて完全分散型の神経回路網と記号的制約伝播ネットを共存させることが可能となる。更に既存の解析法と異なり、文法知識の増大による複雑さの増大の問題が軽量並列プロセス数の増加で対処可能となる。

Massively-Parallel Constraint Propagation as a Paradigm for Natural Language Processing

Hideto TOMABECHI[1]

ATR Interpreting Telephony Research Laboratories
Seika-cho Sorakugun, Kyoto 619-02 JAPAN
tomabech%atr-la.atr.co.jp@uunet.UU.NET

Abstract

We propose a model of natural language processing based on a paradigm of massively parallel (symbolic/subsymbolic) constraint propagation. It differs from the traditional spreading-activation marker-passing schemes in its capacity to handle linguistic phenomena that require application of complex grammatical constraints. We also discuss a scheme of realizing massively parallel constraint propagation activity on a parallel machine hardware through the use of *light weight processes* while retaining the capacity to integrate fully-distributed contextual and acoustic recognitions. Unlike existing parsing schemes, in our model, the increase in the size of grammar can be directly countered through an increase in number of parallel light weight processes.

[1]Visiting Research Scientist from Center for Machine Translation, Carnegie Mellon University.

1 Motivation

Recent developments in marker-passing based theories have shown us some promising results especially in providing strong contextual inferences. There still remains a few questions that needs to be answered before these models replace traditional models of natural language processing. These questions are typically of 1) capacity to handle complex syntactic processing such as attained by unification-based framework 2) *ad hoc* nature of preparing top-down scriptal (and thematic) *a priori* contextual knowledge and retrieval of such a knowledge, 3) scalability, especialy to handle a realistic size of grammar and to actually implementing the massive parallelisms.

Thus, our motivation for proposing the model introduced in this paper:

- Processing of different levels of natural language phenomena in a uniform framework. Namely, we are interested in phonemic, syntactic, semantic and pragmatic processing in a uniform framework.
- Support of direct interactions between subsymbolic and symbolic level processing. We would like to maintain the constrained inferential capability of symbolic processing while minimizing the *ad hoc* nature of schemes designed for such a purpose through participation of subsymbolic recognitions.
- To propose a scalable theory of natural language processing and implementing it.

We have identified several criteria that needs to be met in order to propose a model of natural language processing that will address the above motivations. We would like to focus on two of such criteria in this paper:

1. The model needs to be massively parallel in nature. Constraints that participate at different levels of abstractions and processing at a given time slice may be arbitrarily large and their application may be fully nondeterministic[1].
2. The model needs to be able to handle explicit symbolic constraints. By meeting this criterion the model will be capable of imposing constraints that are identified within the existing linguistic and cognitive theories.

[1]This importance is particularly felt when dealing with noisy speech inputs and contextual inferences when candidate alternatives at a given point of sentential processing may be numerous ([Tomabechi and Tomita, 1988]).

2 Basic Processing Principle

In our past proposed models ([Tomabechi, 1987], [Tomabechi, *et al*, 1989], etc.); the architectures assumed data level fine-grain parallelism[2] such as supported by *Connection Machine.* The HMCP model proposed in this paper is massively parallel in nature essentially meeting our previously mentioned first criterion as well; however, as a parsing algorithm, direct support of a full massive parallelism will make the second criterion hard to meet. In other words, symbolic constraints such as formulated by HPSG ([Pollard and Sag, 1987]) are essentially functional and are not trivially *a priori* postulatable using fully distributed data-level fine grain parallel architecture (such as connectionist[3] and spreading activationist models).

As demonstrated in [Tomabechi and Levin, 1989], the advantage of the constraint propagation model we are proposing in this paper over sources of activation passing models lies in the capacity to handle symbolic linguistic constraints. This is not to claim that the model's all parallel processings need to stay at medium grain. The most node level firing activities stay essentially fine grain in nature and we would like to have different levels of granularity of parallelism to coexist in a constraint propagation architecture.

Our solution to this issue is the separation of algorithmic massive-parallelism from the fine-grain data-level massive-parallelism assumed in

[2]By way of definition, we will be using the following notion of levels of parallelism in this paper: Fine grain – a level where basic operations of the system is parallelized. For example, firing of each nodes, basic arithmetic operations on each input, etc.. Medium grain – the level where functional units are parallelized. By this definition, concurrent applications of various constraints at various locations of memory would be medium grain. Coarse grain – parallelism at the level of sub-modules of the whole system. Parallel processing of input at different modules of system may be coarse.

[3]Although some recent reports (such as [Elman, 1988]) imply these networks may *a posteriori* capture some of such constraints.

the massively parallel spreading activation architecture through the introduction of the notion of *light weight processes*[4]. A *lwp* is a process that is spawned explicitly by other *lwps* (or by an initial process)[5]. When a *lwp* completes its evaluations, it simply goes way (i.e., need not be killed by an external process). By making numerous *lwps* to work at the same time (with little or no synchronization between them) on different nodes, massively parallel processing can be attained without a hardware massive parallelism. Also, any number of *lwps* can work on one node, and therefore, if necessary the parallelism can be even finer than the node level parallelism. Since a *lwp* may work on a functional constraint, a mixture of a fine-grain and a medium-grain parallelisms can be supported. Thus our model has three distinct levels of processing:

- Node level: this is the level where phonemic and conceptual nodes receive and fire activations, ie, the representational level of memory nodes. In the past models of massively parallel natural language processing, this was the level of processing as well.
- Light weight process (lwp) level: this is the level at which actual massively parallel processing is performed. Any number of *lwps* may be created during processing, independent of number of nodes and number of processing units.
- Processing unit level: this is the level of actual processing hardware. Any number of processors may be configured depending on the hardware architecture. One (or more) processing unit may be dedicated to the scheduling of *lwps*.

Thus in our model, the representational level, the process level and the hardware level are explicitly separated.

[4]As supported in Mach through 'thread'. In our implementation, *lwps* are explicitly provided by CLiP Parallel Commonlisp.

[5]Each *lwp* may run on any available processing unit (processor) and is scheduled for by a separate scheduling process. Each lwp is capable of accessing the entire shared memory and may lock or unlock and read/write any parts of shared memory.

3 Our Paradigm

Let us now turn to the theoretical points of our model which is based on our proposed paradigm called "*Head-driven Massively-parallel Constraint Propagation*" (HMCP). HMCP, as a model of massively-parallel spreading activation, can be viewed as a strongly constrained spreading activation model. In the HMCP model, we propagate *constraints* instead of *markers*. From the algorithmic point of view, the constraints can be stored in the "marker objects" that are propagated in the network, and therefore, HMCP may be viewed as an extension of the "marker-passing" algorithm, except for our strong theoretical distinction between the underlying notions of **propagating constraints** and **spreading pointers to origins**. From a non-algorithmic point-of-view, the HMCP paradigm proposes a model in which input (natural language or from other sensory input) imposes constraints that are propagated from the lower classes in the abstraction hierarchy to the higher classes in the abstraction hierarchy. In other words, the new features or **constraints are reversely inherited** from the lower class to the upper class in order to determine the *meaning* (or identity) of the input that is captured by the network of concepts that received (reverse inherited) the new features from the nodes that are below them in the abstraction hierarchy. To be more precise, we view natural language "understanding" as a recognition process, in which already known concepts (ideas, episodes, memory about things, etc.) collectively receive new features that are screened through the *grammar*[6] of the language. By such an activity, the input language is recognized and identified with the existing concepts in memory, while the existing network itself is modified by ac-

[6]If we can view *the grammar of language* to be the information that maps the input language to the collection of concepts that are recognized and organized in the manner that is consistent with the already existent knowledge (memory) about the world, then the *grammar of language* for the conceptual inheritance network is the constraints that are imposed in order to guide (map) the recognition and re-organization of the conceptual inheritance (i.e., memory) network in order to accept the input language. "Meaning" representation in such a network is a time-sliced state of the network after the application of the constraints imposed by the input language itself, which is not extractable by isolating certain feature value pairs from the whole network.

cepting the constraints that are imposed by the input activations.

3.1 Constraint Propagation Parsing

Under the HMCP model, conceptual nodes representing argument-taking predicates carry subcategorization features which specify syntactic properties (such as case) of constituents which can fill their argument positions. Syntactic information such as case, number, and person is propagated up from noun phrases in a package of 'head features' which eventually collides with the constraints in subcategorization frames.

The following three things are propagated from lexically activated nodes: 1) head-features attached to the node, 2) identity of the instance node that is associated with the current lexical activation (i.e., which specific instance should be associated or created with the current lexical activation) and 3) the specific cost (weight) associated with the given lexical activation[7].

3.1.1 A Walk through a Parse

We will describe the HMCP parsing model by walking through the parse of *John persuaded Sandy to give Mary the book.* The verb *persuade* specifies that the entity associated with its object be shared with that of the unexpressed subject of its VP complement. In other words, *persuade* specifies that it subcategorizes for a complement which is itself unsaturated[8]. Thus, there is a dependency between the embedding object and the embedded subject. This phenomenon is known as *object control*[9].

We have three types of nodes in the constraint propagation network: lexical nodes, inheritance nodes, and memory-instance nodes. Lexical nodes are the nodes that have phonological entries attached to them. Two kinds of lexical nodes exist: head-node and complement-node. Head-nodes have subcategorization feature attached to them (i.e., package complement nodes). Complement-nodes do not. Inheritance nodes are the nodes that are organized as a hierarchy and are superclasses of lexical nodes. Memory-instance nodes are the specific instances of lexical and inheritance nodes that are recorded in the network as experiential memory.

We have four kinds of layers in the network: 1) Static Layer (SL); 2) Potential-activation Layer (PL); 3) Activation Layer (AL); and 4) Decaying Layer (DL). SL is where nodes by default belong to. PL is where head nodes and nodes that are packaged by head nodes initially belong to. AL is where node that received constraint propagation belong to. DL is where nodes in AL move to after a given period. Nodes in DL eventually move to PL.

Prior to the parse, all complement nodes (i.e. all nodes that potentially satisfy an element of a subcategorization list) are put into the Potential-activation Layer (PL). In this example, nodes corresponding to *persuade* and *give* contain subcategorization lists as the value of the subcategorization feature. In the node corresponding to *persuade*, the constraint NP[NOM] in the subcategorization list is provided with *PERSON in the *persuader (actor)* role, so *PERSON is added to the PL. *ACTION and all other concepts coindexed with subcategorized positions are concurrently added to the PL. All other nodes in the network are in the Static Layer (SL) (– Or in a DL if a previous utterance exists).

For example, the lexical concepts representing the verbs *persuade* and *give* are encoded[10] in the network as below:

```
(def-lex *PERSUADE
 (inherits-from *ACTION)
 (phonology |p| |r| |s| |w| |e| |i| |d|)
 (spelling persuade)
 (head-feature v-inf-plus)
 (control object)
 (subcat (n-nom n-acc v-inf))
 (roles (actor
        arg1
        arg2))
 (holders (*person
          *person
          *action)))


(def-lex *GIVE
  (inherits-from *ACTION)
  (phonology   |g| |i| |v|)
```

[7]The cost based ambiguity resolution schemes are discussed in detail in [Tomabechi, *et al*, 1989] and [Kitano, *et al*, 1988] and are not discussed in this paper.

[8]The vocabulary in this paper describing linguistic phenomena is based on HPSG framework ([Pollard and Sag, 1987]).

[9]For detail of handling control verbs and word order (obliqueness) constraints, please refer to [Tomabechi and Levin, 1989].

[10]The representation here is taken from our implementation using the 'HyperFrame' ([Nyberg, 1989]) frame-based knowledge representation tool.

```
(spelling    give)
(head-feature v-bse-minus)
(subcat (n n-acc1 n-acc2))
(roles  (actor
         arg1
         arg2))
(holders (*person
          *person
          *object)))
```

The head-features such as *v-inf-plus* and *v-bse-minus* are themselves part of a subsumption relation subnetwork for grammatical categories. They are representing the features [**maj: v, vform: inf, aux: plus**] and [**maj: v, vform: base, aux: minus**] respectively[11]. Each element in the *subcat*, *roles* and *holders* lists at one particular position represents the constraint for another element in the same position in other two lists. Therefore, *n* in SUBCAT represents the subcategorization constraint for the *actor* to be noun and the semantic (relational) constraint for this position is **person* as represented in the HOLDER list. Since a word order is captured through a separate application of obliqueness order constraints ([Tomabechi and Levin, 1989]), each set of lists is order independent. The CONTROL feature specifies the constraints for complement control relations. For example, if the value is *object*, it is postulating that its object (*arg1*) gets unified[12] with the first role in the complement.

A lexical entry for *to* may be encoded as below (following HPSG analysis) as well:

```
(def-lex *TO
 (inherits-from *AUX-ACTION)
 (phonology |t| |u|)
 (spelling to)
 (head-feature v-inf-plus)
 (control subject)
 (subcat (n v-bse))
 (roles (actor
         arg1))
 (holders (*generic-object*
           *action)))
```

The speech recognition subnetwork[13] that receives the acoustic input activates the phonemic nodes in the constraint propagation network. Once the phonemic sequence < /j//o//n/ > is recognized lexical-node *JOHN is activated. The head-features, phonemic and other cost information, and memory-instance of *JOHN (i.e., *JOHN001, etc.) are propagated upward in the abstraction hierarchy. Phonemic cost information is used for phonemic confusion disambiguation not discussed in this paper ([Tomabechi, *et al*, 1989])[14]. The memory-instance represents the discourse entity that *John* is referring to in the current utterance[15] for the input *John*. When an upward propagation reaches a node in the PL, in this case *PERSON, the constraints propagated (such as head-features) are left on that node in the PL. In this example, the upward constraint propagation triggered by *John* carries the head feature NP[ALL-CASE], and this is left (along with phonemic cost and memory-instance information) on the node *PERSON.

When an activation reaches the top of the inheritance network, *lwps* are spawned for all head-nodes in the network. If any head-node is already activated, the spawned *lwps* in turn spawn their children *lwps* for each role of the head-node to check their constraints. Grand children *lwps* may be further spawned[16] for different types of linguistic constraints that may be applied nondeterministically (including subcategorization, linear-precedence, obliqueness-order constraints.). Since at the input of the first word *John*, no head-node is already activated, nothing happens and all spawned *lwps* disappear.

The next word, *persuaded*, activates[17] the

[11]In this particular implementation, head-features and subcategorization constraints are checked by traversing the subsumption relation network. This could be performed by a unification operation as well.

[12]In our model, the notion of unification in the context of control relation is specified by the constraint that the memory instance for the *arg1* position is the same node as the memory instance for the first role position in the complement.

[13]The efforts to integrate Time-Delay Neural Network phoneme recognition with an HMCP parsing is described in [Tomabechi, 1990].

[14]Other cost information includes reverse cost that is given by the subsymbolic recurrent network as contextual priming, which is not discussed in this paper ([Tomabechi, 1990], [Tomabechi, ms]).

[15][Kitano, *et al*, 1988] and [Tomabechi, ms] discuss the schemes for identity resolution when multiple candidate discourse entities exist for a noun phrase using top-down ([Kitano, *et al*, 1988]) and subsymbolic ([Tomabechi, ms]) contextual priming.

[16]Since at this grand-children level, *lwps* need to be coordinated through 'and' parallelism, depending upon implementations, parallel spawning at this level may not be advantageous over sequential constraint satisfactions. Such a trade-off between making grain size finer and increase in overhead varies depending on the specific machine architectures.

[17]Nodes for past tense morphology inherit all lexical

(lexical) head-node *PERSUADE, which is subcategorized for NP[NOM] coindexed with *PERSON. The head-nodes that are activated perform local activities to find their complement role fillers (this is performed by spawned *lwps* for each of the activated head-node). In performing role filling activities, memory-instances of the head-nodes are created and the constraint checking is performed on the memory-instances for the current utterance. The constraint application activities for subcategorization, obliqueness-order, complement-order, etc. are performed and if all constraints are met, (memory-instances of) complement-nodes fill the relational roles of their heads. In our example, *PERSUADE001 tries to find (memory-instances of subclasses of) *PERSON to fill its *persuader (actor)* role. The head-feature NP[ALL-CASE] constraints successfully meet the subcategorization constraint NP[NOM] and therefore, *JOHN001 fills the *persuader (actor)* role. NP[NOM] is removed from the subcategorization list and the parse continues looking for the other subcategorized argument of *persuade*. Other activated head-node (memory-instance) continues its role filler constraint application activity (performed by their *lwps*) concurrently[18]

Recognition of *to give Mary the book* continues in a similar manner. *Mary* (*MARY001) fills the *receiver (arg1)* role and *the book* (*BOOK001) fills the *given (arg2)* role. When the whole subcategorization of a head-node (memory-instance) is accepted (i.e., all roles are filled), the head-node propagates its own constraints upward (i.e., its head-features and its memory-instance that packages the accepted memory-instances of the complement nodes). In this case, *GIVE propagates the head features ((MAJ V) (VFORM fin) (AUX minus))[19] along with the identity of the memory-instance that packages the accepted complement instances. The concept *ACTION in the PL receives this activation, which satisfies the constraints on the *circumstance (arg2)* role of *PERSUADE. *PERSUADE specifies that the NP[ACC] which fills the *persuadee (arg1)* role is coindexed with the NP that was unsaturated inside the VP which fills the *circumstance (arg2)* role. This now indicates that same *John* fills the *giver (actor)* role in *GIVE. This way, the phenomenon known as control is correctly handled in the HMCP recognition model. Thus if we inspect the content of the created instance of the root lexical node *PERSUADE, we get the following output[20]:

```
(*PERSUADED-10
  (INHERITS-FROM *PERSUADED)
  (TYPE
    (VALUE
      (COMMON MEMORY-INSTANCE)))
  (ACTOR
    (VALUE
      (COMMON *JOHN-10)))
  (ARG1
    (VALUE
      (COMMON *MARY-10)))
  (ARG2
    (VALUE
      (COMMON *TO-10))))
```

Note that we follow the HPSG analysis in our syntactic constraint handling that we treat *to* as a VP-head of the infinitival phrase which itself subcategorized for a VP. Thus, when we print the contents of *TO-10 we get:

```
(*TO-10
  (INHERITS-FROM *TO)
  (TYPE
    (VALUE
      (COMMON MEMORY-INSTANCE)))
  (ARG2-OF
    (VALUE
      (COMMON *PERSUADED-10)))
  (ACTOR
    (VALUE
      (COMMON *MARY-10)))
  (ARG1
    (VALUE
      (COMMON *GIVE-10))))
```

which subject controls the embedded VP headed by *give*:

```
(*GIVE-10
  (INHERITS-FROM *GIVE)
  (TYPE
    (VALUE
      (COMMON MEMORY-INSTANCE)))
  (ARG1-OF
    (VALUE
      (COMMON *TO-10)))
  (ACTOR
    (VALUE
      (COMMON *MARY-10)))
```

node information from the default finite verb forms except (TENSE PAST) is added to the verb form features.

[18]The system's recognition is massively-parallel in nature and multiple number of subcategorization can be active at a given time, as well as different hypotheses for discourse entity reference and phonemic, lexical and conceptual ambiguity.

[19]One thing we have omitted here for the sake of simplicity is that there is an intermediate subject control head *to* whose head-feature is (VFORM inf).

[20]Taken from our actual sample output on a Sequent /Symmetry running a parallel Commonlisp.

```
  (ARG1
    (VALUE
      (COMMON *SANDY-10)))
  (ARG2
    (VALUE
      (COMMON *SUSHI-10))))
```

Thus, correct assignments of controlled and embedded complement subject positions are attained through the constraint propagation parsing.

4 Discussion:

The HMCP model shares with the spreading activation marker passing models its difference from localist connectionist models that HMCP and marker passing models pass sources of activations around. This is an algorithmic solution to the problem of connectionism commonly known as 'binding problem' in that variable binding is a hard problem in a connectionist network. By actually storing a pointer to the origin of an activation, the HMCP and the traditional marker passing models make it possible to keep track of an instance that a variable (activated node) needs to be bound to. The theoretical difference between the HMCP and the traditional models of spreading activation marker passing models such as DMTRANS([Tomabechi, 1987]) is paradigmatic and philosophical in nature as well as functional. In traditional models, the markers were typically passed in order to: 1) activate relevant nodes, 2) record sources of activations, and 3) predict the next input activations. Given the networks were purely conceptual, these models were schemes of semantic (conceptual) recognition and inferences. While the inference provided from these networks may be valuable for external linguistic modules ([Norvig, 1989] [Tomabechi and Tomita, 1988]), these models were insufficient as models of parsing because knowledge such as phonology, syntax and discourse parameters were not formulatable within these frameworks (for example, there is no way to capture the obligatory *object control* in traditional marker passing framework). HMCP is a model of constraint propagation and in HMCP, markers act as media for propagating constraints. Constraint satisfaction activities of a network constitute a parsing process.

We view the HMCP model to be scalable in that constraints are stored distributedly in each lexical-nodes. In other words, increase of number of constraints, such as corresponding to the increase in number of grammatical rules, will take the form of increase in number of nodes horizontally in the network. Given the activations are essentially local (i.e. only upward in the abstraction hierarchy, never horizontally in the network), increase in the grammatical complexity can be countered by the increase in number of *lwps* (and actual processors to counter the increase in number of processes). Each node changes its state through a constraint satisfaction activity based on its own local environment and sole communication between nodes is a passing of head-feature markers vertically upward in the abstraction hierarchy. An aggregation of numerous such local activities will constitute a result of a parse.

In the past models of massively parallel spreading activation natural language processing, algorithmic massive parallelism as activities of nodes were either explicitly designed to be hardware activities of fully distributed massively parallel processing units ([Tomabechi, *et al*, 1989]), or implicitly assumed [Riesbeck and Martin, 1985] [Norvig, 1989]). However, when the amount of information passed between nodes increases (such as in our constraint propagation network), it is easy to predict that communication speed of loosely coupled processing units, typically found in hardware massive parallel architecture, will create an intolerable bottleneck. We have learned that by introducing the notion of *light weight processes*, tightly coupled shared memory architecture is one viable architecture for implementing a massively parallel constraint propagation network.

5 Conclusion

We have seen that HMCP model makes the coexistence of massively-parallel memory-based recognition activities and strict linguistic constraint applications such as postulated by HPSG possible through a propagation of both linguistic and non-linguistic constraints in the inheritance memory network. We have also proposed a scheme of realizing such a massively parallel constraint propagation activity on a parallel machine hardware through the use of *light weight processes*. By separating the data-level massive-parallelism of the localist connectionist network from the hardware level parallelism by intermediate *lwps*, the mixture of fine-grain node level activities and medium-grain constraint application activities were attained in a uniform massive parallelism. Also, HMCP network is compatible with a connectionist network in its architecture and we have already succeeded in integrating various types of connectionist network as subnetwork of the HMCP network to enhance contextual and other processing of the system[21]. We are especially interested in combining the explicit *a priori* provided deductive symbolic constraints with a *a posteriori* learned inductive constraints from the connectionist subnetwork[22]. Currently, the HMCP architecture seems to be a sole viable model for such a symbolic and subsymbolic interaction.

ACKNOWLEDGMENTS

The author would like to thank Akira Kurematsu, Tsuyoshi Morimoto, Hitoshi Iida and other members of ATR, and Koichi Imai of Pana-Sequent Inc.. Thanks are also due to Jaime Carbonell, David Evans, Carl Pollard, Alex Waibel, Lori Levin, Sergei Nirenburg, Masaru Tomita and other members of the Carnegie Mellon community. I would also like to thank Jun-ichi Tsujii, Hiroaki Kitano, Margalit Zabludowski, and Kiyoaki Hara for their continued support.

Appendix: Implementation

The HMCP parsing system is implemented using Allegro CLiP version 3.0.3 which is a parallel Commonlisp from Franz Inc. The system is running on a Sequent Symmetry which is a tightly coupled multiprocessor shared memory machine running DYNIX 3.0 parallel UNIX. Light weight processes and their scheduling are directly supported by CLiP. Parallel implementation of HMCP was originally done on Multilisp running on Mach at CMU. A serial lazy evaluation version is also running on CMU-COMMONLISP.

[21]Please refer to [Tomabechi, 1990] for the schemes to integrate a time-delay neural network and a recurrent neural network.

[22]Since a learned subsymbolic knowledge in the connectionist subnetwork is captured in a smooth activation space (as opposed to frame-based scriptal and thematic knowledge encoded originally in the network), constraints captured in a subsymbolic network may be valuable to a symbolic network.

References

[Elman, 1988] Elman, J. *Finding structure in time.* CRL TR-8801. Center for Research in Language, University of California, San Diego, 1988.

[Kitano, *et al*, 1988] Kitano, H., Tomabechi, H., and Levin, L. "Ambiguity Resolution in the DmTrans Plus". In *Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics*, 1988,

[Norvig, 1989] Norvig, P. "Marker Passing as a Weak Method for Text Inferencing". In *Cognitive Science*, Vol. 13, Num. 4, 1989.

[Nyberg, 1989] Nyberg, E. *The HyperFrame User's Guide Version 1.0.* Technical Memo. Cognitive Research Laboratories. 1989.

[Pollard and Sag, 1987] Pollard, C. and Sag, A. *Information-based Syntax and Semantics.* Vol 1, CSLI, 1987.

[Riesbeck and Martin, 1985] Riesbeck, C. and Martin, C. *Direct Memory Access Parsing.* Yale University Report 35, 1985.

[Tomabechi, 1987] Tomabechi, H. "Direct Memory Access Translation". In *Proceedings of the IJCAI-87*, 1987.

[Tomabechi and Tomita, 1988] Tomabechi, H. and Tomita, M. "The Integration of Unification-based Syntax/Semantics and Memory-based Pragmatics for Real-Time Understanding of Noisy Continuous Speech Input". In *Proceedings of the AAAI-88*, 1988.

[Tomabechi and Levin, 1989] Tomabechi, H. and Levin, L. "Head-driven Massively-parallel Constraint Propagation: Head-features and subcategorization as interacting constraints in associative memory", In *Proceedings of The Eleventh Annual Conference of the Cognitive Science Society*, 1989.

[Tomabechi, *et al*, 1989] Tomabechi, H., Kitano, H., Mitamura, T., Levin, L., Tomita, M. *Direct Memory Access Speech-to-Speech Translation: A Theory of Simultaneous Interpretation.* CMU-CMT-89-111, Carnegie Mellon University, 1989.

[Tomabechi, ms] Tomabechi, H. 'Feature-based Dual-Recurrent Neural Network for Symbolic/Subsymbolic Constraint Interactions', Manuscript. Carnegie Mellon University 1990.

[Tomabechi, 1990] Tomabechi, H. 'Symbolic and Subsymbolic Massive-Parallelism for Speech-to-Speech Translation: Hybrid Time-Delay, Recurrent, and Constraint Propagation Connectionist Architecture'. In *Proceedings of The International Conference on Information Technology (InfoJapan'90).*