

Understanding Spoken Natural Language with Omni-Directional Information Flow

Katashi Nagao

Sony Computer Science Lab. Inc.
3-14-13 Higashi-gotanda,
Shinagawa-ku, Tokyo 141, Japan
nagao@csl.sony.co.jp

Kôiti Hasida

Electrotechnical Laboratory
1-1-4 Umezono,
Tukuba 305, Japan
hasida@etl.go.jp

Takashi Miyata

University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo 113, Japan
mya-u@is.s.u-tokyo.ac.jp

Abstract

A totally constraint-based computational architecture is employed to deal with problems in speech and natural language processing. The semantics of constraints is defined as a type of dynamics, with potential energy measuring the degree of violation and field of force towards minimal energy, so that information processing emerges as dynamical interaction. This establishes tight feedback loops among heterogeneous sorts of information, and integrates various types of computation under dynamical control. In speech understanding, the omni-directional information flow arising in this architecture leads to natural solutions to tough problems such as recognition of ill-formed or ambiguous speech, prediction of subsequent utterances, revision of former recognition, treatment of insertion/deletion of phonemes, detection of word boundaries, and so forth. Since the architecture is constraint-based, there is no underlying procedure specific to domains or tasks.

1 Introduction

Flexible intelligent behavior must involve very diverse information flows, which vary drastically from one context to another and encompass heterogeneous sorts of information. The methods proposed so far in AI are generally based on procedures concerning specific sorts of information such as syntax and semantics, and specific tasks such as speech recognition and parsing. An attempt to coordinate these methods faces fatal difficulties concerning scalability, because procedures stipulate encapsulated ways of information flow, which require a great deal of revision and annotation when they are put them together.

To overcome this difficulty, we consider an architecture consisting of modules of constraint and a general method for controlling context-dependent information flow among heterogeneous types of constraint. By *constraint* we mean a design orthogonal to information flow, so that there should be no special prescription of information flow for pattern recognition, planning, language

comprehension, and so on. Coordination among modules is, therefore, a far simpler issue with constraints than with procedures.

Dynamical semantics [Hasida, 1992] enables us to design context-sensitive complex behaviors under declarative semantics within tractable computational complexity. Constraints are represented in a clausal-form logic program, and a fine-grained declarative semantics is defined over the constraints by measuring the degree of violation in terms of real-valued *potential energy*. A field of force arises along the gradient of this energy, and inferences are controlled on the basis of the dynamics. As the potential energy reflects the context, so too does the control of information processing. The computational resources are concentrated on dynamic focal points in the huge hypothesis space. Also, the system is robust in the sense that it is free from hard failure of constraint satisfaction.

In the rest of the paper, we consider spoken language understanding. Since the constraint-based architecture implements omni-directional information flow between speech and natural language processing, we are able to resolve several tough problems in understanding spoken language, such as treatment of ambiguous or ill-formed speech, prediction of subsequent utterances, revision of previous interpretation, and some speech-specific issues including phoneme insertion/deletion and word boundary recognition. While most conventional spoken language systems first construct phonological hypotheses, then apply higher level knowledge and generate semantic interpretations, our system simultaneously applies all types of constraint, from acoustic to pragmatic. We discuss some constraints concerning spoken language understanding, but, of course, there is nothing in principle which prevents integration of any other sort of constraint.

Some research has managed to integrate linguistic information into speech understanding, such as that by Erman et al. [1980], Hayes et al. [1986], Young et al. [1989], Kitano [1991], and Chung et al. [1992], the methods they propose, however, are specific combinations of specific types of knowledge sources with rather fixed information flows, and with flexible interactions disallowed among them. Our method can flexibly control omni-directional information flow in a very context-sensitive

fashion. The details of the relationship to these work are discussed in Section 4.

2 Interaction across Heterogeneous Information

Section 2 of Hasida et al. [1993] accounts for the computational architecture employed in our study. The rest of this paper employs the notation and terminology discussed there.

We consider the computational processes along with an example in which the following Japanese sentence is understood from an acoustic input.

tegami kaita
letter wrote

‘(I) wrote a/the letter.’

Since phoneme /k/ and /h/ are similar to each other, *kaita* might sound like *haita* (vomited). In addition, *kaita* might also sound like *kaite* (write-IMPERATIVE) due to the similarity between /a/ and /e/. The former confusion is resolved by selectional restrictions, and the latter by the semantic and pragmatic context. As shown below, these processes involve omni-directional information flow among acoustic, phonetic, morphological, syntactic, semantic, and pragmatic constraints.

In this example, we consider the following clauses.

- (1) `true -sentence(S,X,-) -speech0(X,Y) -sounds(Y).`
- (2) `np(S,X0,X6) -letter(S) -t(X0,X1) -e(X1,X2)`
`-g(X2,X3) -a(X3,X4) -m(X4,X5) -i(X5,X6).`
- (3) `vp(S,X0,X5) -write(S) -past(S) -k(X0,X1)`
`-a(X1,X2) -i(X2,X3) -t(X3,X4) -a(X4,X5).`
- (4) `sentence(Event,X,Z) -np(Index,X,Y) -vp(Event,Y,Z)`
`-sbjobj(Event,Index).`
- (5) `sbjobj(Event,Index) -agent(Event,Index).`
- (6) `-agent(-,X) +animate(X).`
- (7) `-animate(X) -letter(X).`
- (8) `-letter(X) +document(X).`
- (9) `sbjobj(Event,Index) -patient(Event,Index).`
- (10) `-patient(Event,Index) -write(Event)`
`+document(Index).`

The details will be accounted for in the following discussion.

2.1 Acoustics and Phonetics

It is theoretically possible for our computational architecture to incorporate traditional methods of speech recognition, such as Hidden Markov Models (HMM) [Itou et al., 1992], DP matching, and Time-Delay Neural Networks [Waibel et al., 1989]. For instance, HMM consists of a finite state automaton and stochastic information, which are encoded straightforwardly in logic program and dynamics, respectively. In the current implementation, however, phoneme recognition is an external procedure call, because this involves sequential simulation and, hence, cannot accommodate the massive computation required for speech recognition in our computation-demanding architecture.

Although this excludes acoustic information from dynamical interaction, the phonological information can still interact dynamically with syntactic, semantic, and other types of information within the uniform computational architecture. That is, the scores for the phonemes are not determined once and for all, but vary over time depending upon the global context, including various other types of information, as discussed later.

Figure 1 shows an example of a phoneme lattice that is

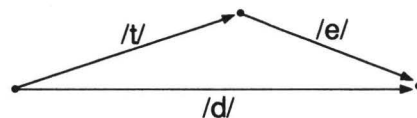


Figure 1: Recognized Phoneme Lattice

constructed in the traditional speech recognition phase. In this figure, /t/, /e/, and /d/ represent phonemes. This phoneme lattice is encoded in the constraint network in Figure 2. The output of the HMM procedure

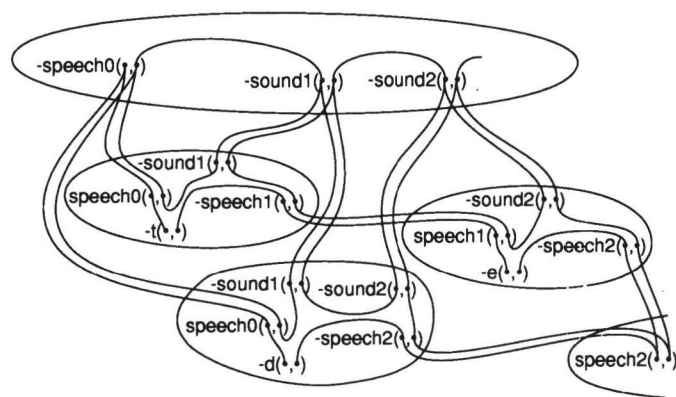


Figure 2: Constraint Encoding Phoneme Lattice

augments such a network. In this figure, `sound1(•,•)` and `sound2(•,•)` represent acoustic signals introduced over time. They may be regarded as sequences of vector quantization codes, for example. `speech0(•,•)`, `speech1(•,•)`, and `speech2(•,•)` represent phoneme boundaries, which correspond to nodes in the phoneme lattice shown in Figure 1. Also `t(•,•)`, `e(•,•)`, and `d(•,•)` correspond to the recognized phonemes /t/, /e/, and /d/. Currently, when a new phoneme (say /t/) comes in, a new literal `-soundi(•,•)` is introduced into the top clause (1). The scores of speech recognition are reflected in the relevant parameters of energy functions.

As mentioned above, such structures are made by an external procedure in the current implementation, but this process is designed to fit into our main architecture in the following two senses. First, these procedure calls are eventually subject to dynamical control. That is, the procedure is called when a phoneme on the right edge of the lattice is linked to another through a lexical entry. The creation of such a link is controlled on a dynamical basis, as described in the next subsection. Second, we

can make the structure in Figure 2 without recourse to external procedures, if a speech recognition method such as HMM is really encoded in constraints as mentioned before.

2.2 Phonetics and Morphology

Next, we explain interaction between recognized input phonemes and phonemes in the lexicon. Clauses (2) and (3) are lexical entries.

For the sake of simplicity, we first show exact matches between inputs and the lexicon. Figure 3 illustrates how an input phoneme sequence /t e/ ($t(\bullet, \bullet)$ and $e(\bullet, \bullet)$ in the figure) and phonemes in the lexicon *tegami* (letter) are connected to each other by unification links.

There is a local potential energy which captures the transitivity of equality. Due to this energy, an equation tends to be highly activated when it belongs to a cycle of equations. Symbolic operations to create cycles of equations are hence preferred in most cases. In Figure 3, consider the initial situation where the links between the two $t(\bullet, \bullet)$ s and those between the two $e(\bullet, \bullet)$ s are not yet created. The first argument of $\text{speech0}(\bullet, \bullet)$ and the sec-

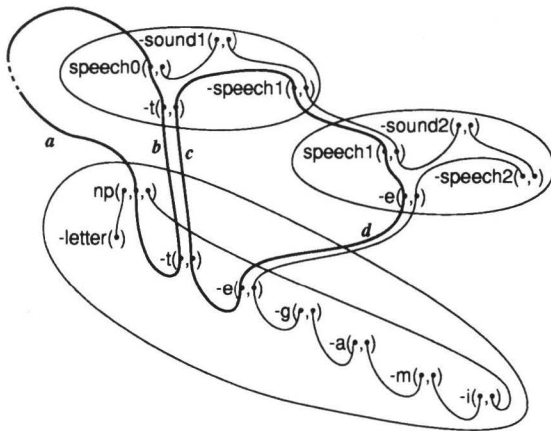


Figure 3: Exact Match between Phonemes and a Word

ond argument of $\text{np}(\bullet, \bullet, \bullet)$ are connected through equation path *a*. (*a* runs across the syntactic constraint, as shown in Figure 7.) If the two $t(\bullet, \bullet)$ s are linked, then equation *b* together with equation path *a* will constitute a cycle of equations as indicated by the thick curve. So, the operation to create the links between the two $t(\bullet, \bullet)$ s is preferred here. Namely, $|\frac{\partial P}{\partial s}|$ is large now, where *s* is parameter of the energy function associated with links *b* and *c*. Similarly, if the two $e(\bullet, \bullet)$ s are linked as well, then equation *c* together with equation *d* will constitute a cycle of equations. Thus, the operation to create links between the two $e(\bullet, \bullet)$ s is preferred once the two $t(\bullet, \bullet)$ s are linked. Consequently, the recognized phoneme sequence /t e/ is likely to be matched against the initial part of the phonological form of the word *tegami* (letter).

Figure 4 illustrates a deviation from exact matching, including one insertion/deletion and one substitution. The leftmost pair of the figure exhibits a substitution of /t/ with /k/ or vice versa. The atomic formulas and equations in the thick cycle have lower activation

values than when that matching was exact. This is because mutual excitation between different phonemes ($t(\bullet, \bullet)$ and $k(\bullet, \bullet)$ in this case) is weaker than that between the same phonemes. Here, we omit the details of how to encode this in the potential energy. Insertion and deletion also reduce the activation values of the neighboring part of the constraint network because they break cycles of equations. In Figure 4, no cycle of equations involves /g/. Note that, in general, a cycle of equations is created only by consecutive matching of phonemes. Approximate matching is an emergent phenomenon of

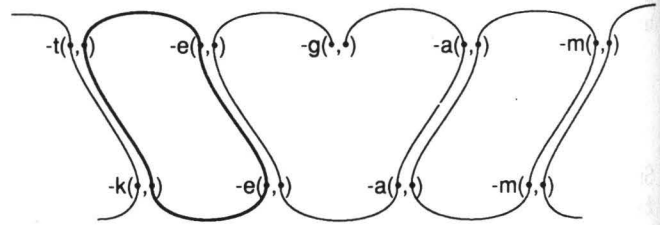


Figure 4: Insertion, Deletion, and Substitution of Phonemes

this computational architecture, because the dynamics allows us to compare the adequacy of inexact matchings.

2.3 Morphology and Syntax

Syntax is also considered entirely as a constraint consisting of clauses such as (4). Phoneme sequences across word boundaries are encoded as a part of the syntactic constraint. So, consecutive matching of phonemes is preferred due to equation cycles, just as in the matching within a single word discussed above.

In Figure 5, the word *tegami* has been totally covered

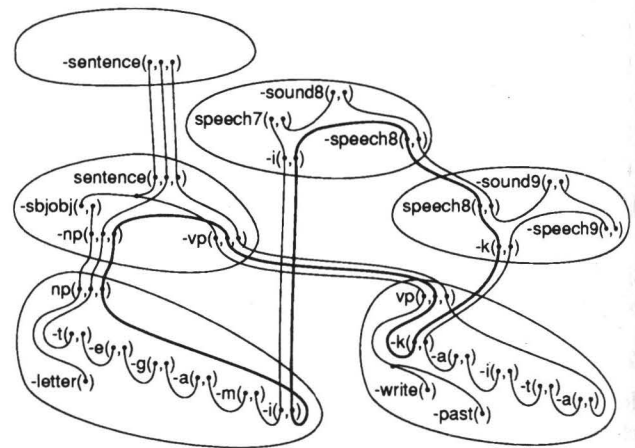


Figure 5: Interaction among Phonemes, Words, and Syntax

by the input phoneme lattice. Now, the most preferred operation is to establish a link from the input phoneme /k/ to its counterpart in the word *kaita*, because that will create a cycle of equations as shown by the thick curve.

As noted earlier, we do not employ any domain/task-specific prescription on information flow. Algorithmic

behavior is just an epiphenomenon of the dynamics. For instance, an efficient parsing process similar to chart parsing has been shown to emerge out of general heuristics similar to our dynamical control [Hasida, 1991]. In contrast, the disadvantage of procedures like Earley's algorithm [Earley, 1970] and generalized LR parser [Tomita, 1987] is that they are less sensitive to context because they stipulate a specific information flow. It is difficult to extend them to incorporate semantic information in the course of syntactic parsing, for instance, because that requires you to further stipulate when to use which semantic information. Generally speaking, procedure-based systems tend to become complex, thus reflecting the diversity of the information flow to be implemented. The information flow in the human cognitive process appears diverse enough to render such systems intractable.

2.4 Plus Semantics, Pragmatics, and Whatever

Now, we explain a semantic process using clauses (5, 6, 7, 8, 9, 10). The process is roughly logic-based, and uses deduction and abduction, whose applications are also controlled by the dynamics. (5) means that if **Index** is **Event's agent**, then **Index** is **Event's subject** or **object**. Similarly, (9) means that if **Index** is **Event's patient**, then **Index** is **Event's subject** or **object**. Due to completion, to make **sbjobj(V,N)** true in (4) requires that **Index** be an **agent** or **patient** of **Event**. From (6), if **Index** is **agent**, then **Index** should be **animate**. From (7), however, if **Index** is **letter**, then **Index** cannot be **animate**. Thus, link *g* in Figure 6 causes a contradiction. Then, the cycle

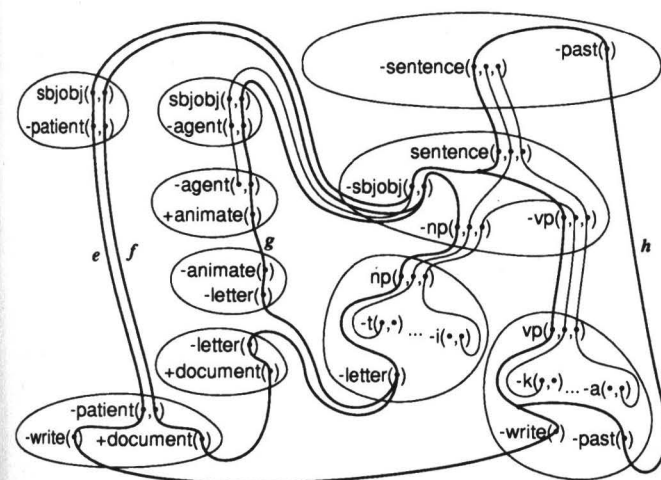


Figure 6: Interaction among Syntax, Semantics, and Pragmatics

involving *g* becomes less activated than the cycle involving *f*, which does not contain contradictions. In general, an interpretation with more contradiction-free cycles is preferable. This heuristics emerges from the dynamics of constraints. In this example, the interpretation *tegami kaita* ('(I) wrote a/the letter') is the most preferable, because it involves more contradiction-free cycles than other interpretations such as *tegami haita* ('(I) vomited a/the letter'), which has no cycle, and *tegami kaite*

('please write a letter'), which involves two cycles corresponding to the ones involving *e* and *f*.

If **past(•)** does not exist in the top clause and, instead, we have some pragmatic information indicating that the speaker is requesting something, then the first interpretation will have only two cycles while the third interpretation will have three or more cycles. Thus, the third interpretation (i.e., the imperative interpretation) will be preferred. The semantic/pragmatic constraints can, thus, contribute to prediction/activation of the interpretation of utterances.

A more global view of the constraint involved in the current example is given in Figure 7, which shows a snap-

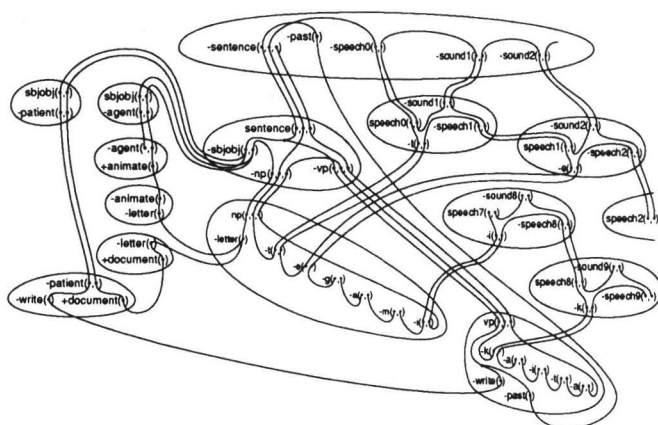


Figure 7: Interaction among All Types of Information

shot of recognition of phoneme sequence /t e g a m i k/. As suggested by this figure, all the different types of constraints dynamically interact with each other, due to omni-directional information flow across this picture.

3 Experiment

In this section, we show some snapshots of execution of the program implementing the above theory.

Figure 8 represents the state of the constraint network where the first phoneme of the *tegami kaita* example has just been accepted. The buttons at the top of the window are used to execute actions such as spreading activation and backpropagation. In the right near the bottom of Figure 8 is a small window entitled 'DP.info' to communicate some information about parameters and activation values. The other squares are clauses. The top clause is near the upper right corner, and the clauses to its left correspond to those in Figure 6. In the lower half of the left hand side are lexical entries for *tegami* and *kaita*. The clause to the right of them contains the first phoneme /t/. The four links connected to the phoneme literal indicate tentative matching of that phoneme with ones in lexical entries.

Figure 9 shows how a phoneme lattice is created after some phonemes are accepted. The pyramid of clauses in the center of the window encodes the lattice shown in Figure 10. An insertion of a phoneme takes place in Figure 11. In this case, the input phoneme sequence is

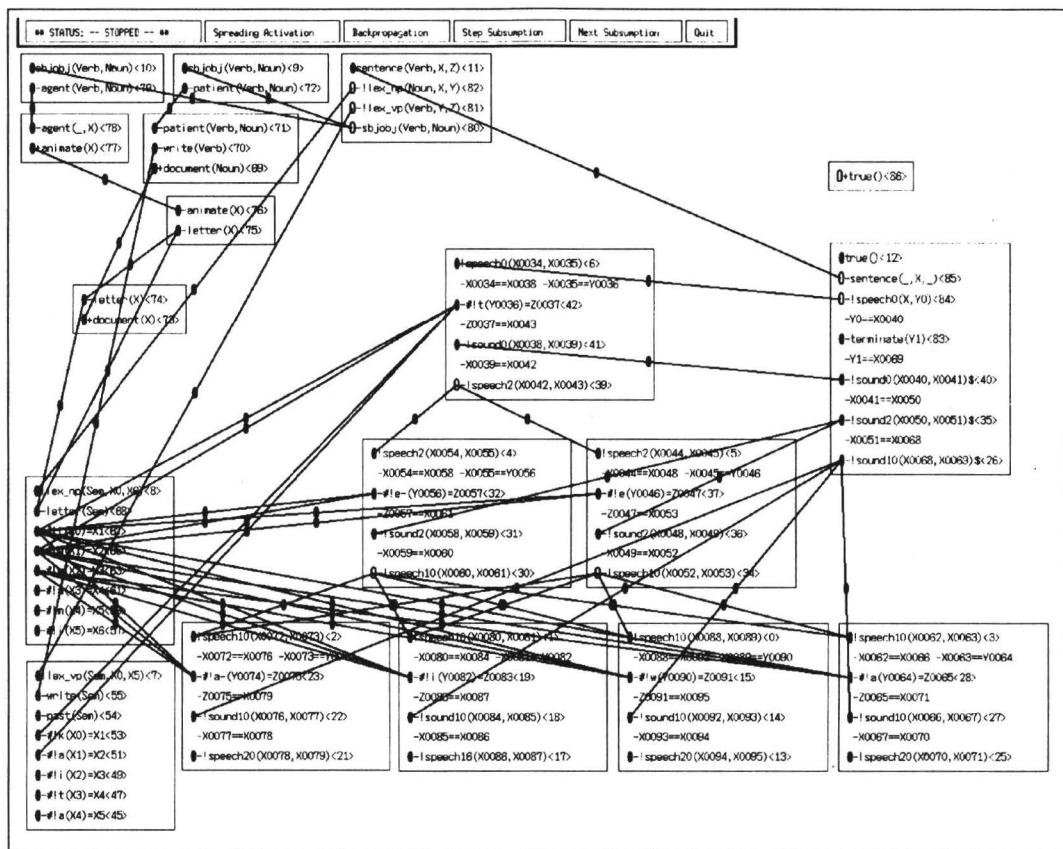


Figure 9: Emergence of Phoneme Lattice

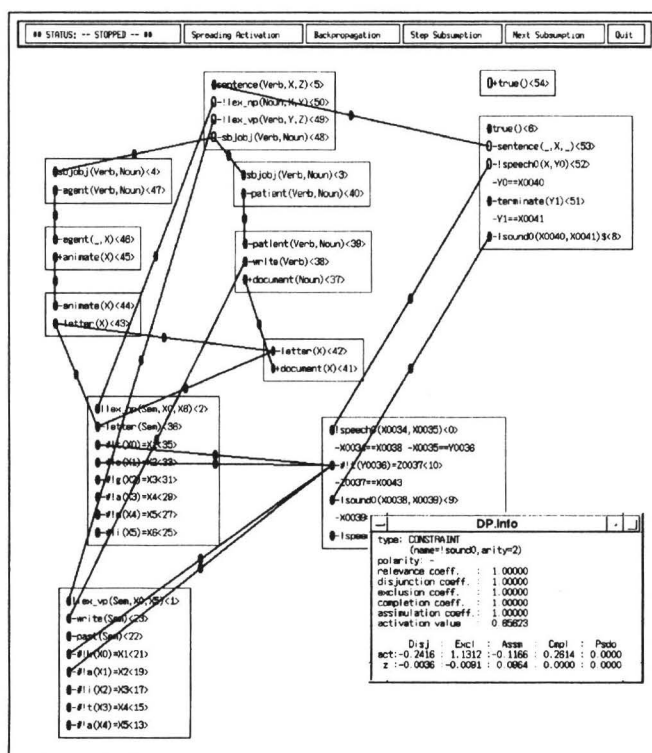


Figure 8: Incorporation of the First Phoneme

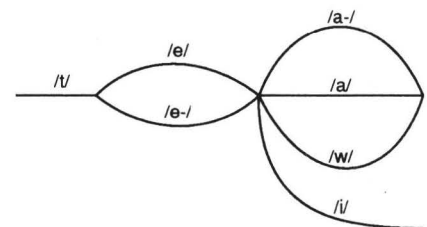


Figure 10: Phoneme Lattice

/t e i g a m i/, where the first /i/ is regarded as noise. In the figure, this /i/ is matched against /e/ and /g/ in the lexical entry of *teqami*.

The current implementation is in 14,000 lines of C code, and runs on standard UNIX workstations. This program is registered as *ICOT Free Software*, and the source is publicly available free of charge for noncommercial purposes. In this sequential implementation, the computational complexity in average cases appears to be linear to the size of the constraint network, which means that the average time complexity is expected to be constant on an appropriate massively parallel machine.

4 Discussion

Some previous studies have addressed how to combine different knowledge sources for spoken language understanding. HEARSAY-II [Erman *et al.*, 1980] employs

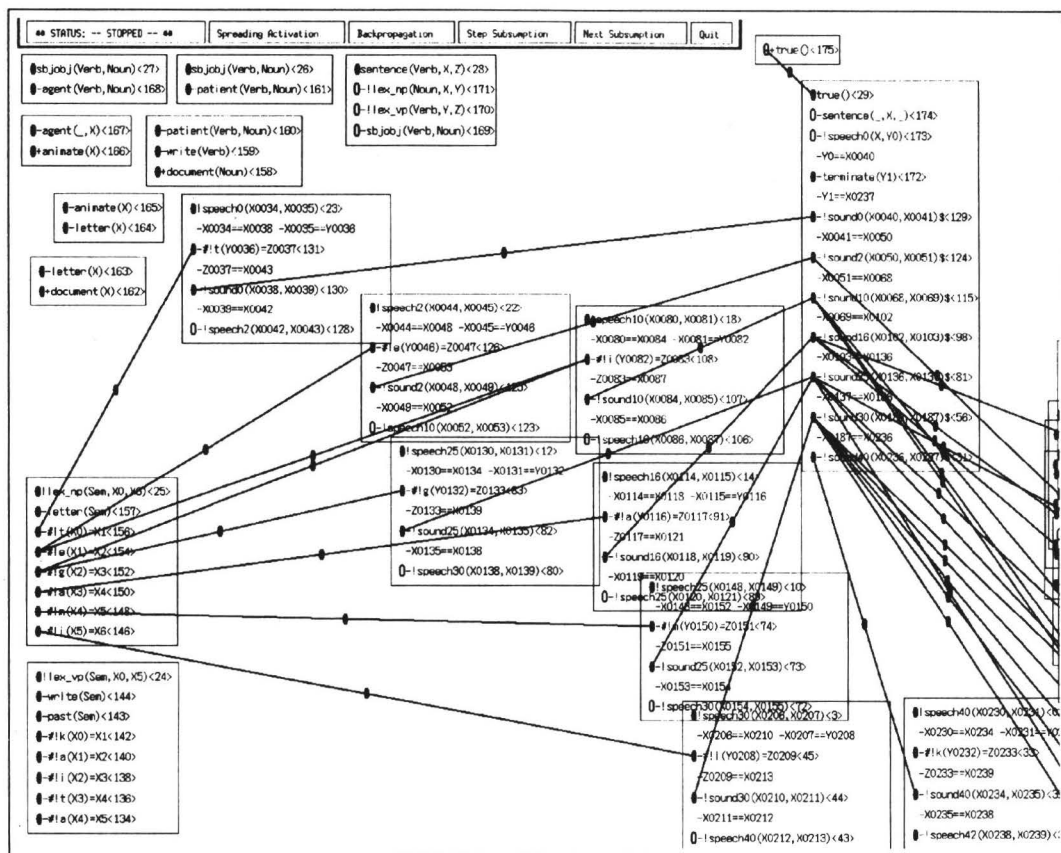


Figure 11: Insertion of Phoneme

blackboard architecture to coordinate several types of information relevant to spoken language. Hayes et al. [1986] employ semantic case frames to analyze spoken language. Young et al. [1989] use dialogue knowledge to predict subsequent utterance. Since they use specific procedures to incorporate specific sorts of knowledge, however, extension to include other sorts of knowledge into the tight interaction is hopelessly cumbersome, because procedures are basically encapsulated prescription of information flow. Although blackboard architecture is an attempt to reduce such difficulty, the attempt is not successful as far as it is domain/task-specific procedures that operate on the blackboard. Generally speaking, prescriptions of information flow give rise to encapsulated patterns of information processing, and hence domain/task-specific procedures prevent tight interactions among diverse sorts of information.

Kitano [1991] and Chung et al. [1992] have developed a method of parallel marker passing to support both top-down prediction and bottom-up activation. Their approach is more flexible than the traditional ones, because marker passing is a weak method and thus does not pose so large chunks of encapsulated information flow as in the traditional approaches. However, they prescribe the routes of marker passing, restricting the diversity of information flow so as to exclude revision of previous phonological interpretation, among others. Also, their approach does not naturally extend to incorporate the acoustic process into the tight feedback loops involving

also linguistic and other information.

After all, a major advantage of our method over the previous ones is that it implements more fine-grained and diverse feedback loops integrating heterogeneous sorts of information, rather than constructing phoneme-level hypotheses in advance and then going on to the linguistic level as in most of the other approaches. So our system can for instance pick up an alleged phoneme depending upon linguistic or extralinguistic context, even if that phoneme looks so implausible on the acoustic ground alone that traditional bottom-up architecture would filter it out before entering the linguistic mode. Note that such diverse information flow does not cause computational intractability, because information processing at each context is centered around dynamically determined focal points or the focus of attention in the constraint network. This control mechanism gives rise to spoken language understanding without any specific procedures for specific tasks, such as matching between phonemes, parsing, and semantic and pragmatic inferences. The computation emergent from this control is highly context-sensitive, since there is no prescription of encapsulated information flow specific to any individual domain or task.

5 Concluding Remarks

We have proposed a computational architecture which supports omni-directional information flow among het-

ergeneous knowledge sources, from acoustics to pragmatics, and demonstrated that problems in spoken language understanding are naturally resolved in this architecture without any domain/task dependent prescription of information flow. Although the current implementation handles the acoustic information by an external procedure of the traditional type, it is possible to absorb it into the general architecture, provided with appropriate massively parallel, communication-intensive hardware.

In fact, we are planning to implement the present theory on a massively parallel machine and include the acoustic computation into the constraint-based architecture, so as to directly incorporate it into the tight feedback loops realized by the constraints. This will necessitate a learning algorithm to acquire the dynamical properties of the constraint on the linguistic sounds. Our conjecture is that such parameter learning is possible by using the generalized backpropagation method we have already implemented to control computation.

Acknowledgements

The authors would like to thank HAYAMIZU Satoru and ITOU Katunobu at Electrotechnical Laboratory for their help in preparation of the reported experiment.

References

- [Chung *et al.*, 1992] S. Chung, D. Moldovan, and R. F. DeMara, "A Parallel Computational Model for Integrated Speech and Natural Language Understanding," *Technical Report*, IMPACT-92-008-USC, International Consortium for Massively Parallel Advanced Computing Technologies, 1992.
- [Earley, 1970] J. Earley, "An Efficient Context-Free Parsing Algorithm," *Communications of the ACM*, Vol.13, No.2, pp.94-102, 1970.
- [Erman *et al.*, 1980] L. Erman, F. Hayes-Roth, V. Lesser, and R. Reddy, "The Hearsay II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty," *ACM Computing Surveys*, Vol.12, No.2, pp.213-254, 1980.
- [Hasida, 1991] K. Hasida, "Common Heuristics for Parsing, Generation, and Whatever ...," In *Proceedings of Workshop on Reversible Grammar in Natural Language Processing*, pp.81-90, 1991. Held in connection with the 29th Annual Meeting of the Association for Computational Linguistics.
- [Hasida, 1992] K. Hasida, "Dynamics of Symbol Systems: An Integrated Architecture of Cognition," In *Proceedings of the International Conference on Fifth Generation Computer Systems (FGCS'92)*, pp.1141-1148, 1992.
- [Hasida *et al.*, 1993] K. Hasida, K. Nagao, and T. Miyata, "Joint Utterance: Intrasentential Speaker/Hearer Switch as an Emergent Phenomenon," In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93)*, this volume, 1993.
- [Hayes *et al.*, 1986] P. J. Hayes, A. G. Hauptmann, J. G. Carbonell, and M. Tomita, "Parsing Spoken Language: A Semantic Caseframe Approach," In *Proceedings of the 11th International Conference on Computational Linguistics (COLING-86)*, pp.787-792, 1986.
- [Itou *et al.*, 1992] K. Itou, S. Hayamizu, and H. Tanaka, "Continuous Speech Recognition by Context-Dependent Phonetic HMM and an Efficient Algorithm for Finding N-best Sentence Hypotheses," In *Proceedings of the International Conference on Acoustic, Speech, and Signal Processing (ICASSP'92)*, pp.I.21-I.24, 1992.
- [Kitano, 1991] H. Kitano, "ΦDM-DIALOG: An Experimental Speech-to-Speech Dialog Translation System," *IEEE Computer*, Vol.24, No.6, pp.36-50, 1991.
- [Tomita, 1987] M. Tomita, "An Efficient Augmented-Context-Free Parsing Algorithm," *Computational Linguistics*, Vol.13, pp.31-46, 1987.
- [Waibel *et al.*, 1989] A. Waibel, H. Sawai, and K. Shikano, "Modularity and Scaling in Large Phonemic Neural Networks," *IEEE Transactions on Acoustic, Speech, and Signal Processing*, Vol.37, pp.1888-1898, 1989.
- [Young *et al.*, 1989] S. R. Young, A. G. Hauptmann, W. H. Ward, E. T. Smith, and P. Werner, "High Level Knowledge Sources in Usable Speech Recognition Systems," *Communications of the ACM*, Vol.32, No.2, pp.183-193, 1989.