PROCEEDINGS OF THE INTERNATIONAL CONFERENCE
ON FIFTH GENERATION COMPUTER SYSTEMS 1992,
edited by ICOT. © ICOT, 1992

Dynamics of Symbol Systems — An Integrated Architecture of Cognition —

HASIDA, Kôiti
Institute for New Generation Computer Technology (ICOT)*

Abstract

To account for the diversity and partiality of information processing in the cognitive process, we need a design method for cognitive system without explicit stipulation of domain/task dependent information flow, together with a control scheme for partial information processing which does not commit us to global and crisp consistency or completeness.

A computational architecture is proposed which consists of a first-order logic program with a dynamics. Information flow is controlled not by any domain/task dependent procedures but by a control scheme emergent from the dynamics. The declarative semantics of the logic program is defined by formulating the degree of violation in terms of *potential energy*, and a control scheme for both analog and symbolic inferences is derived from an energy minimization principle. This inborn integration of the control scheme with the declarative semantics guarantees a natural reflection of semantic relevance in inferences. Ideas underlying inference mechanisms developed so far, such as weighted abduction and marker passing, are captured in terms of such a dynamics.

1 Introduction

It is practically impossible to delimit the information of the world potentially relevant to the benefit (typically, survival) of a cognitive agent, whereas the information-processing capacity of the cognitive agent is severely restricted. Here arises *partiality of information*: the information potentially relevant to the determination of a cognitive agent's action (including information processing) is only partially reflected in its actual behavior.

Only very relevant information must hence be selectively reflected in the behavior of the cognitive agent. However, the distribution of relevant information, together with the degree of relevance, drastically changes depending on the context. Since only a very small part of the potentially relevant information is exploited at each context, dramatically different parts of the information must be exploited at different contexts, in order for the whole information the cognitive agent uses in various contexts to encompass as much of the relevant information as possible.

This causes very diverse patterns of information flow, underlying the complex behavior of a cognitive agent. So cognition is complex, not entirely because the design of the cognitive agent itself is complex, but rather because it is situated in a complex world, which provides the diverse contexts of the cognitive agent's behavior. The cognitive agent is complex indeed, but still is far simpler than the behavior of the agent reflecting also the vastness of the world.

To capture this situatedness and relative simplicity of a cognitive agent, the design of the cognitive system should largely abstract away the directions of information flow (the temporal order of actions, among others). The models which stipulate the directions of information flow (that is, *procedural* programming) quickly become untractably complex, attributing too much of the complexity of cognitive process to the complexity of the cognitive system itself, and thus failing to capture the situatedness of cognition. For instance, production systems (Anderson 1983) fail to serve as the functional architecture of cognition. This is where *constraint* paradigm comes in. Constraint abstracts the direction of information flow away from the design of a cognitive model, keeping the model within tractable complexity, attributing most of the complexity to the world, and thus capturing the situatedness of cognition.

So the domain-dependent aspects of cognition (language, vision, etc.) should be designed basically in terms of declarative semantics rather than operational semantics. Symbolic logic is a typical formalism for declarative design. Some sort of logic at least as powerful as first-order predicate calculus is considered necessary to design a cognitive system capable of combinatorial behaviors such as language use.

However, such a powerful formalism commits us to untractable computation for maintaining global consistency, exhaustive examination of the possible hypotheses, and so on. This applies to whatever logics have ever been fabricated, including non-monotonic logic, probabilistic logic, fuzzy logic, paraconsistent logic, and so forth. There has been no formalism of logic which could support useful inferences under arbitrary sort of violation of the constraint in question. The problem here is

*From April of 1992, the author is at Natural Language Section, Electrotechnical Laboratory, 1-1-4 Umezono, Tukuba, Ibaraki 305 JAPAN.

essentially that symbolic logics provide no control over inferences other than closure operation (exhaustive inference).

We need a declarative formalism which inherently supports partial and hence tractable computation, while approximately preserving the first-order expressive power and supporting diverse flow of information. To be useful at all, that computation must be about only very relevant information, which will lead to a diverse information flow sensitive to the context.

To implement all this, the present paper considers a system of constraint represented as a first-order logic program, and postulates a *dynamics* of this constraint. The degree of violation is captured in terms of *potential energy*, which is a real-valued function of the state of the constraint. The constraint is thus provided with a fuzzy declarative semantics which is finer-grained than the usual crisp semantics. An operational semantics is also derived from the dynamics. That is, control schemes for analog and symbolic inferences are obtained on the basis of energy minimization principle. Such an inborn integration of declarative semantics and inference method not only supports concise design but also guarantees natural reflection of semantic relevance in inferences.

The rest of the paper proceeds as follows. In the next section we outline the combinatorial structure of the constraint. Section 3 provides a declarative semantics for this constraint. The components of the declarative semantics are each formulated in terms of potential energy. Section 4 discusses the field of force induced from the potential energy, and analog information processing driven by this field of force. It will be shown that associative inferences naturally emerge out of the dynamics. Section 5 defines a method of symbolic inference which is a sort of program transformation, and derives a control scheme for it on the basis of energy minimization principle. The proposed framework is pointed out to capture the ideas underlying some inference mechanisms tailored so far, such as weighted abduction (Hobbs et al. 1990, Stickel 1989) and marker passing (Charniak 1986, Norvig 1989). Section 6 concludes the paper.

2 Constraint Network

A constraint consists of *clauses*. A clause is a set of *literals*, and roughly means their disjunction, which is inclusive or exclusive to various degrees depending of their dynamical properties as discussed later. A *literal* is an *atomic constraint* preceded by a sign. An atomic constraint is an *atomic formula* such as p(X,Y,Z) or an *equation* such as X=Y. Signs are '+' and '−' and stand for affirmation and negation, respectively. '+' is omitted in cases discussed below. Names beginning with capital letters represent variables, and the other names predicates.[1] A clause is written as a sequence of the included literals followed by a period. The order among literals is not significant. So (1) and (2) represent the same clause, which means (3) in a rough, crisp approximation.

(1) −p(U,Y) +q(Z) −U=f(X) −X=Z.

(2) +q(X) −p(f(X),Y).

(3) $\forall U,X,Y\{\neg p(U,Y) \vee q(X) \vee U \neq f(X)\}$

A clause containing a literal with empty sign is called a *definition clause* of the predicate of that literal. The meaning of such a predicate is defined in terms of completion based on its definition clauses. For instance, if the definition clauses of predicate p are those in (4), then p is defined as in (5).

(4) p(X) −q(X,a).   p(f(X)) −r(X).

(5) $\forall A\{p(A) \Leftrightarrow \{\exists Y(q(A,Y) \wedge Y = a) \vee$
$\exists X(A = f(X) \wedge r(X))\}\}$

A definition clause of a zero-ary predicate **true** is called a *top clause*. A top clause corresponds to the query in Prolog. That is, top clause (6) represents top-level hypothesis (7).[2]

(6) true −p(X) +q(X,Y).

(7) $\exists X,Y\{p(X) \;\wedge\; \neg q(X,Y)\}$

We postulate clause +**true**. to give rise to such a top-level hypothesis. The computation is to tailor the best hypothesis to explain a top-level one.

A constraint is regarded as a network. For instance, the following constraint may be graphically shown as in Figure 1.

(i) +true −p(A) −q(B).

(ii) +p(X) −r(X,Y) −p(Y).

(iii) +r(X,Y) −q(X).

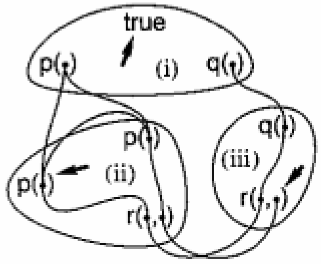

Figure 1: Constraint Network.

[1] A binding is also regarded as an atomic formula. For example, X=f(Y) is an atomic formula with binary predicate =f.

[2] Theoretically, Prolog uses false instead of true here so that the negation of the top clause amounts to the top-level hypothesis. In our formulation, a top clause itself directly means a top-level hypothesis.

In such a graphical representation, a clause is a closed domain containing the atomic constraints constituting that clause. Short thick arrows indicate references to the atomic constraints as positive literals in clauses. Atomic constraints without such indication are negative literals. An argument of an atomic formula is shown either as a '•' or as an identifier. Equations between arguments are links. Equations in clauses are called *intraclausal equations*, and those outside of clauses are called *extraclausal equations*.

We will write $\alpha \circ \beta$ to mean that atomic formulas α and β are unifiable. We regard each part of constraint network as a set of instances, and $\alpha \circ \beta$ as meaning that whether $I(\alpha) \cap I(\beta) = \emptyset$ or not is unknown. I is an interpretation function which maps those instances to objects (state of affairs, in the case of atomic formulas) in the world. So unifiability is not transitive. We assume two atomic formulas are unifiable if and only if their corresponding arguments are directly connected through an extraclausal equation, and that every extraclausal equation connects two corresponding arguments of two unifiable atomic formulas.[3] For each zero-ary predicate, the constraint network contains only one atomic formula with it.

3 Declarative Semantics

Now we move on to dynamics to define a declarative semantics for the constraint network described above.

Each atomic constraint α has an *activation value* x_α, which is a real number such that $0 < x_\alpha < 1$ and may be regarded as the truth value (or a subjective probability of the truth) of α. The *potential energy* of a constraint network is a function of the activation values, and represents the degree of violation of the constraint. The potential energy U of the entire constraint is the sum of the potential energy of the parts of the constraint.

The declarative semantics of the entire constraint is decomposed into several aspects. U is a sum of terms each representing one such aspect, so that U captures the whole declarative semantics. Each term of U is the degree of violation of the aspect of declarative semantics in question. These aspects are enumerated below and each formulated by a term of potential energy.

Normalization of activation value. In order to normalize the activation value of an atomic constraint α so that $0 < x_\alpha < 1$, let us employ a standard sigmoid function $\mathrm{sg}(x) = \frac{1}{1+\exp(x)}$ and postulate $x_\alpha = \mathrm{sg}(-F_\alpha/T)$ holds at equilibria of force, where F_α stands for the total force to α from outside of α, and T is a positive constant called the *temperature*. This amounts to assuming the following energy inherent in α.

$$T\{x_\alpha \log x_\alpha + \overline{x_\alpha} \log \overline{x_\alpha}\} \tag{8}$$

Let us call this the *normalization energy* of α. For any v, $\overline{v}$ stands for $1 - v$. Here and henceforth, mathematical details are not very important; they are quite tentative indeed. The formulas are mainly motivated by convenience. In fact, the fancy outlook of (8) is for the computational ease of analog inference, though we do not go into details here.

Disjunction of literals in a clause. The *disjunction energy* of a clause implements the ordinary disjunctive meaning of the clause. For instance, consider the following clause.

(9) −p +q.

The ordinary disjunctive meaning of this clause is that ¬p or q is true. The disjunction energy of this clause as follows captures this meaning.

$$D x_{\mathsf{p}} \overline{x_{\mathsf{q}}} \tag{10}$$

D is a positive constant associated with clause (9). (10) is small iff either x_{p} or $\overline{x_{\mathsf{q}}}$ is small; keep in mind that the activation values are between 0 and 1 due to the normalization energy. The semantics of (9) may be depicted like Figure 2. D in (10) represents how large the area b is in comparison with a in this figure.

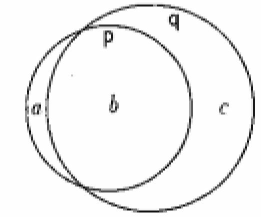

Figure 2: Venn Diagram for (9).

Mutual exclusion of literals in a clause. By 'mutual exclusion' we mean that at most one literal may be true in a clause. In the case of (9), the mutual exclusion will allow us to abductively assume p when given q, and assume ¬q when given ¬p. The following term, called the *exclusion energy* of (9), will take care of such inferences.

$$E \overline{x_{\mathsf{p}}} x_{\mathsf{q}} \tag{11}$$

E is a constant associated with clause (9). In Figure 2, E represents how large the area b is in comparison with c. If q means that you are in Japan, for instance, E is larger when p means that you are in Tokyo than when it means that you are in Imabari, a small city in the island of Sikoku.

In the general form, the disjunction energy and the exclusion energy of clause Φ consisting of literals $l_1, \cdots, l_m$ are (12) and (13), respectively.

[3] There can hence be $O(N^2)$ extraclausal equations, for N different atomic formulas sharing the same predicate. So an efficient encoding schema would be necessary to avoid that space complexity. We skip further details of this issue.

$$(12)\quad D_\Phi \prod_i \overline{r_i y_i} \qquad\qquad (13)\quad E_\Phi \sum_{i\neq j} r_i y_i r_j y_j$$

y_i is the activation value of l_i. For any atomic constraint α, the activation value of literal $+\alpha$ is defined to be x_α and that of $-\alpha$ is defined to be $\overline{x_\alpha}$. r_i is a constant such that $0 < r_i \leq 1$, and is called the *relevance coefficient* of l_i. In the digital approximation, (12) means that at least one literal should be true, whereas (13) means that at most one literal may be true. Incidentally, it is due to exclusion energy that top clause (6) means (7). In (9), (10) and (11), l_1 =−p, l_2 =+q, $y_1 = \overline{x_\mathsf{p}}$, $y_2 = x_\mathsf{q}$, and $r_1 = r_2 = 1$.

Completion of an atomic formula. We somewhat extend the notion of completion so that to complete atomic formula (not predicate) α positively (negatively) means that α ($\neg\alpha$) should be inferred either deductively or abductively[4] on the basis other than the one on which α ($\neg\alpha$) was first postulated. For example, if we have postulated q(X) (say, based on clause +p(X) −q(X)., abductively) and it is positively completed, then it must be inferred from another reason; typically, another atomic formula q(Y) could be closely related with q(X) (in the sense of assimilation to be discussed later) and is inferred on the basis of a clause such as +q(Y) −r(Y). deductively or −q(Y) +s(Y). abductively. As discussed later, completion implements assumability cost (Hobbs et al. 1990).

The positive and negative *completion energy* of an atomic formula α are defined by (14) and (15), respectively.

$$(14)\quad C^+_\alpha x_\alpha \prod_{\alpha\circ\beta} \overline{s_{\alpha\beta} x_\beta} \qquad\qquad (15)\quad C^-_\alpha \overline{x_\alpha} \prod_{\alpha\circ\beta} \overline{s_{\alpha\beta}\overline{x_\beta}}$$

C^+_α and C^-_α are positive constants, and are called the *positive completion coefficient* and the *negative completion coefficient* of α, respectively. $s_{\alpha\beta}$ is a constant called the *subsumption coefficient* of α as to β. $s_{\alpha\beta}$ represents how close α is related to β, as seen also in the formulation of assimilation below. We say α *subsumes* β to mean $I(\alpha) \supseteq I(\beta)$. When $\alpha \circ \beta$, $s_{\alpha\beta} = 1$ if α subsumes β, and otherwise $s_{\alpha\beta} = s_0$ for a small positive constant s_0. In the digital approximation, the positive (negative) completion energy means that some β ($\neg\beta$ for some β) satisfying $\beta \circ \alpha$ should be true in order for α ($\neg\alpha$) to be true.[5] Since a subsumption coefficient usually equals to s_0, which is close to 0, completion energy and accordingly other types of energy often decrease if subsumption coefficients increase, which is caused by symbolic operation discussed in the next section.

The dynamics for definition clauses may be defined on the basis of exclusion energy and completion energy, but we do not go further into details here.

Assimilation between atomic formulas. Two unifiable atomic formulas are the same if they have the same arguments for the corresponding argument places. By relaxing this, we obtain the notion of assimilation: two unifiable atomic formulas should have similar truth values to the extent that they share the same assignment of the arguments. So for instance p(X) and p(Y) tend to have similar activation values if X and Y are linked with a strongly activated equation.

To capture this, we postulate *assimilation energy*. Suppose $\alpha \circ \beta$ for two atomic formulas α and β, and let δ be the extraclausal equation connecting their i-th arguments. Then the assimilation energy of δ is defined as follows.

$$(16)\qquad -A_{\pi i}\,(s_{\alpha\beta} + s_{\beta\alpha})\,x_\delta\,(x_\alpha - \frac{1}{2})(x_\beta - \frac{1}{2})$$

$A_{\pi i}$ is a positive constant called the *assimilation coefficient* of the i-th argument place of the predicate π shared by α and β. The assimilation energy roughly means that x_α and x_β should be similar (both close to 0 or 1) if x_δ is close to 1, and vice versa.

Transitivity of equality. A *transitive cycle* is a cycle of equations $\Delta = \delta_0\delta_1\cdots\delta_{k-1}$ where either $\delta_{(i-1)\bmod k}$ or $\delta_{i \bmod k}$ is an intraclausal equation for every i. Note that no cycle of extraclausal equations is a transitive cycle. Transitivity of equality as to Δ is regarded as excluding the cases where just one equation in Δ is false. To capture this, we define the *transitivity energy* U_Δ of Δ as below.

$$(17)\ U_\Delta = \begin{cases} -t\prod_i(e_i - \theta) & (e_i < \theta \text{ for at most one } i) \\ 0 & (\text{otherwise}) \end{cases}$$

e_i is the activation value of δ_i, and θ is a constant such that $0 < \theta < 1$. t is a positive constant called the *transitivity coefficient*. Note that the transitivity energy is large when just one equation in Δ has a small activation value.

Since detection of cycles is a very costly computation, we will have to consider some approximate method for efficient processing of transitivity energy instead of guaranteeing perfect detection of transitive cycles. We do not go further into such implementation details.

4 Analog Inference

Potential energy gives rise to a field of force to change the state of the system so as to decrease the total potential energy. Suppose there are n distinct atomic constraints in the given constraint, and hence n activation values, x_1 through x_n. Then the current analog state of the system is regarded as a point (18) in the n-dimensional Euclidean space, and the global potential energy U defines a *field of force* (19).

$$(18)\quad \vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \qquad\qquad (19)\quad \vec{F} = \begin{pmatrix} -\frac{\partial U}{\partial x_1} \\ \vdots \\ -\frac{\partial U}{\partial x_n} \end{pmatrix}$$

[4] In this respect, only deduction is considered in Prolog.

[5] The completion in Prolog corresponds to our positive completion. In Prolog β must be deduced only.

$\vec{F}$ causes *spreading activation*: when $\vec{F} \neq \vec{0}$, a change of x_i so as to reduce U influences the neighboring parts of the constraint network, which causes further changes of activation values there, and thus state transition propagates across the network. In the long run, the assignment of the activation values will settle upon a stable equilibrium satisfying $\vec{F} = \vec{0}$, under an appropriate scheme of spreading activation. The resulting state gives a minimal value of U.[6] That is, it satisfies the constraint best in some neighborhood.

Let us look at some typical patterns of analog inferences emerging from the dynamics through spreading activation. First, the dynamics gives rise to associative inference based on syntactic similarity. Suppose for instance that, as in Figure 3, the extraclausal equation

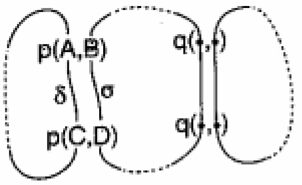

Figure 3: Association due to Syntactic Similarity.

δ connecting argument A of p(A,B) and argument C of p(C,D) is included in a transitive cycle as shown in the figure, and that the activation value of every equation in this cycle is greater than θ. Then δ is excited due to the transitivity energy. This raises the tendency (due to the assimilation energy of δ) for p(A,B) and p(C,D) to have similar activation values. Thus the assimilation energy of the extraclausal equation σ between B and D makes σ to have a high activation value, provided that the equations in the transitive cycle involving σ as shown in the figure are all highly activated. So each equation in a transitive cycle including σ could be excited even stronger due to the transitivity energy. This might make other pairs (such as the two q(•,•)s in Figure 3) of atomic formula with corresponding arguments on that transitive cycle have similar activation values, and so on. In general, two syntactically similar combinations of atomic constraints thus tend to have similar activation patterns, corresponding parts exciting each other or inhibiting each other.

Transitivity energy also enhances semantic association. Consider the following discourse.

(20) Tom took a telescope. He saw a man with it.

We assume that *he* and *it* in the second sentence are anaphoric with *Tom* and *the telescope*, respectively, in the first sentence. There is an attachment ambiguity in the second sentence. about whether the prepositional phrase *with it* modifies *saw* or *a man*. Let us assume that the structure of the constraint generated by processing this discourse looks like Figure 4. Each region

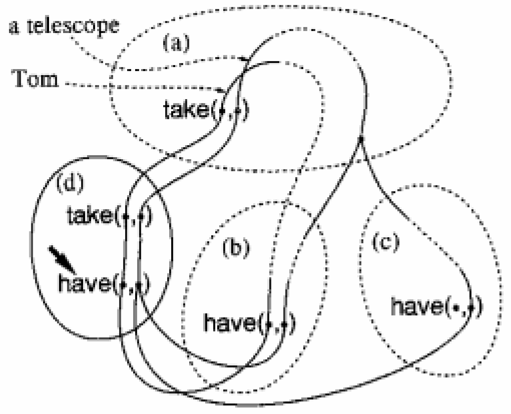

Figure 4: Semantic Association Concerning (20).

in a dashed closed curve represents a cluster of clauses. These clauses have been created by symbolic inference as described in the next section. (a) is a set of clauses including the top clause. (b) and (c) represent two alternative readings of the second sentence of (20), each derived by backward (abductive) inferences. The take(•,•) in (a) is a part of the hypothesis obtained by interpreting the first sentence. Its first argument stands for Tom and the second argument the telescope, so that the whole thing means that Tom takes the telescope at some time. Thus, reading (b) means that Tom has the telescope when he sees the man, and (c) that the man has it when Tom sees him. Clause (d) is an inference rule to the effect that if A takes B then A will have B.[7] Due to this inference rule, the take(•,•) in (a) can imply the have(•,•) in (b) but not that in (c), so (b) is more plausible than (c).

Note that there are two transitive cycles both going through the take(•,•)s in (a) and (d). So these two atomic formulas tend to strongly excite each other due to assimilation energy, provided that every relevant equation is excited. These two cycles also both go through the have(•,•)s in (b) and (d), making them tend to strongly excite each other. too. On the other hand. there is only one transitive cycle which goes through both the have(•,•)s in (c) and (d). Hence the associative inference based on the take(•,•) in (a) through (d) supports the have(•,•) in (b) more strongly than it supports the have(•,•) in (c).

[6] When $\vec{F}$ is not entirely attributed to potential energy, spreading activation is not guaranteed to converge into a stable equilibrium but may exhibit chaotic behaviors. Such a less restricted system may be more powerful and useful, but that is beyond the scope of the present discussion.

[7] We ignore the temporal relation between the taking and the having here.

5 Symbolic Inference

We consider just one type of symbolic operation called *subsumption*. It is a sort of program transformation to create a new subsumption *relation*. A subsumption *operation* concerns a pair of unifiable atomic formulas. As shown in Figure 5, subsumption operation from atomic formula α to β divides β into β' and β''. β' is the maximum subset of β subsumed by α, and β'' is the rest of β: $beta'' = \beta - \beta'$. Neither α nor β' is hence unifiable with β'', as indicated in the figure. If it is somehow known that α subsumes β from the beginning, then no copy (division) need to happen. When the division of β actually takes place, then it causes a duplication of the clause containing β, atomic formula ξ accordingly dividing into ξ' and ξ''.[8] Unlike in the division of β, ξ' and ξ'' are unifiable both with each other and with all the atomic formulas unifiable with ξ, because there is no reason to believe $I(\xi') \cap I(\xi'') = \emptyset$.and so on.

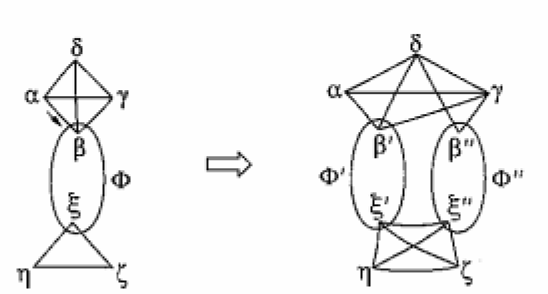

Figure 5: Subsumption Operation From Atomic Formula α to β.

We omit further details of combinatorial aspects of symbolic inference, due to the space limitation, and go on to the dynamical aspect. Subsumption generates new atomic constraints and thus redefines U. $s_{\alpha\beta'}$ is set to 1. because α subsumes β'. $s_{\xi'\xi''}$ and $s_{\xi''\xi'}$ are both set to s_0. because we are not sure about the subsumption relation between these atomic formulas. The other coefficients are simply inherited along with the copy of the part of the constraint network.

Since subsumption is a local operation, it may take place in parallel at many different places. Now we consider how to guide such computation based on the dynamics, without recourse to any centralized control. As the preference score for a subsumption, we could use the expected contribution of that subsumption to reduction of U at the equilibrium of spreading activation. As mentioned above, a subsumption from atomic formula α to β divides β into β' and β'', setting $s_{\alpha\beta'}$ to 1. The expected influence of this to reduction of the total energy could be estimated by $-\frac{\partial P}{\partial s_{\alpha\beta}}$, where P is defined to be the minimal U_0 (U_0 under the condition $\vec{F} = \vec{0}$) in a neighborhood of the current $\vec{x}$. U_0 is a representative part of energy whose definition is not changed due to symbolic operations. For instance, it could be the disjunction energy of clause +true.. At any rate, the symbolic computation is controlled so as to minimize some part of energy, whereas the analog computation to minimize the whole energy. By employing generalized backpropagation (Pineda 1988), $\frac{\partial P}{\partial s_{\alpha\beta}}$ can be efficiently computed for all $s_{\alpha\beta}$. The space complexity of that computation is linear with regard to the size of the constraint network, and its parallel time complexity practically constant. See APPENDIX for mathematical details.

Our method implements some important features of other inference mechanisms proposed elsewhere. First, weighted abduction (Hobbs et al. 1990. Stickel 1989) emerges from our method. In weighted abduction, just as in the current framework, one attempts to tailor a best hypothesis to explain the observed fact. A hypothesis is a conjunction of (negated) atomic formulas. Each conjunct in a hypothesis is assigned an assumability cost, which is a cost of assuming the conjunct. A hypothesis is better when the total assumability cost is smaller. Assumability cost may be reduced by unifying the conjuncts. For instance, if the current hypothesis contains p(A) and p(B) one of which has a large cost, then this cost will be reduced by unifying them. Assumability cost is inherited through abduction. For example, a cost of p(A) in the current hypothesis is inherited down to q(A) and r(A) when p(A) is resolved by clause +p(X) −q(X) −r(X)..

Assumability cost is basically captured by our completion energy: the conjunct in question must be inferred otherwise than the way it was first postulated, or it would be inhibited due to its completion energy. So an *inherent* cost is encoded by a completion coefficient of atomic formula α. This gives rise to a high preference score of subsumption from α, because if α comes to subsume another atomic formula β then perhaps the completion energy of α is reduced due to $s_{\alpha\beta} = 1$, which will be indicated by a large value of $-\frac{\partial P}{\partial s_{\alpha\beta}}$. An *inherited* cost is captured along the same line. For example, when p(A) with a large cost subsumes p(X) in clause +p(X) −q(X) −r(X)., the completion energy of p(A) is probably still large, but it will decrease if q(X) and r(X) get more excited. So the preference score of subsumptions from q(X) and r(X) tend to be large, corresponding to the inherited cost in weighted abduction.

Our framework is more flexible and dynamic than weighted abduction. That is, we allow inferences concerning a hypothesis to influence the state of other hypotheses, whereas in weighted abduction assumability costs change only due to unification involving the atomic formulas carrying those costs. So our method is more appropriate to account for such phenomena as belief revision. In this connection, our dynamical semantics is much more general than the probabilistic semantics of

[8] If α and β belonged to the same clause, then α is also divided into α' and α''. If α' and β' belong to one clause and hence α'' and β'' belong to another, then α' and β'' subsume each other and α'' and β'' are not unifiable.

Charniak and Shimony (1990), which is restricted to propositional Horn logic.

Second, marker passing (Charniak 1986, Norvig 1989) may be also understood as an emergent property of the dynamics, along the same line as above. Consider the following discourse for example.

(21) Taro got a book. He paid one thousand yen.

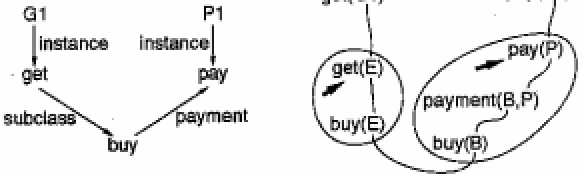

Figure 6: Marker-Passing for (21).

Figure 6 shows the network involved in the abductive inference to assume that Taro bought the book. In the left is the marker-passing network encoded[9] by the constraint network, which is in the right. A node in marker passing network corresponds to an argument or a predicate in our constraint. An edge between an argument node and a predicate node represents that the argument satisfies the predicate, and an edge between two predicate nodes represent a clause referring to the two predicates. The directions of the arrows are static, and irrelevant to the direction of marker passing. get(G1) and pay(P1) are created upon reading/hearing (21), where G1 and P1 stand for the event of Taro's getting a book and that of his paying money, respectively.

In marker passing, the abductive inference of Taro's buying the book will be suggested by a collision of markers passed down from G1 and P1 along the path between them. In our framework, the same abductive inference consists of three subsumption operations along the (copy of) extraclausal equations in the right of Figure 6. The preference scores of these subsumptions are probably all high, because of the path of clauses between get(G1) and pay(P1). If the activation value of get(G1) is larger than $\frac{1}{2}$, then it excites get(E) due to assimilation energy, get(E) excites buy(E) due to exclusion energy, buy(E) excites buy(B) due to assimilation energy, and buy(B) excites pay(P) due to disjunction energy. get(E) is similarly excited indirectly by pay(P1). So get(E), buy(E), buy(B) and pay(P) are excited stronger than when there were no such path. The subsumptions along the extraclausal equations in the right of Figure 6 are therefore very promising for reduction of positive completion energy, so that the abductive inference mentioned above is suggested.

A path of clauses between two very informative atomic formulas (ones with activation values close to 0 or 1) thus tends to raise the preference for subsumptions along it. This is what marker passing is designed to capture in general. Of course how much the preference for subsumption increases depends on the dynamical properties of the path. For instance, the path in Figure 6 would not indicate the above abductive inference if the exclusion coefficients of the two clauses are small.[10] Suggestion of inference also depends on the length of the path. Obviously, shorter paths more readily suggest inferences.

Subsumptions can also be promoted by associative inferences discussed in the previous section, because a subsumption between two atomic formulas will strongly affect P when some of the extraclausal equations between them are strongly excited owing to transitive cycles involving them. See Hasida (1991) for how generation of natural language sentence is controlled by heuristics regarded as approximating our control scheme taking such associations into account.

6 Concluding Remarks

We have discussed a framework of constraint for designing a cognitive system. To capture the partiality and the corresponding situatedness of cognition, the constraint is situated in a field of force derived from potential energy representing the degree of violation. This field of force gives rise to analog inference as spreading activation, and also controls symbolic computation to transform the constraint. Not only nearly logical inferences and abductive inferences but also associative inferences emerge out of such a dynamics.

A distinguished feature of our framework is that the control scheme for inference is derived from a dynamics which also provides the declarative semantics. In comparison, the other frameworks such as marker passing stipulate the inference control apart from the declarative semantics. The inborn integration of declarative semantics and inference control as in our method will not only provide a clear perspective of the design, but also guarantee emergent reflection of semantic relevance in information processing. In this connection, our method is integrated also in another sense that it controls analog and symbolic inferences based on the same dynamics. This is a strong advantage over the methods such as in Waltz and Pollack (1985) which separate the two inference schemes.

The current framework should be extended with respect to several points. First, some partial processing method is necessary for dealing with transitive cycles, although at any rate a massively parallel computational system is essential to implement our theory. Second, *deletion* should be incorporated in addition to subsumption, in order to prevent the constraint network from unlimited growth. Probably deletion is regarded as a reverse of subsumption, and hence the control of deletion

[9] Charniak (1986) employs a similar encoding scheme.

[10] What Charniak (1986) calls isa-plateau can be understood along the same line.

may be formulated along the same line as that of subsumption. Third, the control method should take into account consistency checking as well. Consistency checking pertaining to binding is discussed in Hasida (1991).[11] In order to handle consistency checking in general, we will have to give preferences not only to subsumptions which seem to decrease P but also to those which seem to increase P. Finally, learning is vitally necessary for both the coefficients (Suttner and Ertel 1990) and the symbolic structure of constraint. Further scrutiny is open with regard to the role of the dynamics in learning.

APPENDIX

The equilibrium condition of the spreading activation concerning $\vec{x}$ is regarded in general as $\vec{x} = \vec{y}$, where $\vec{y}$ is a vector function of $\vec{x}$ and $\vec{S}$.[12] Let s be a parameter in $\vec{S}$. When $\vec{y}$ is differentiable, we get (22) where $\frac{\partial \vec{y}}{\partial \vec{x}}$ is defined by (23).

$$(22) \qquad \frac{\partial \vec{x}}{\partial s} = \begin{pmatrix} \frac{\partial x_1}{\partial s} \\ \vdots \\ \frac{\partial x_n}{\partial s} \end{pmatrix} = \frac{\partial \vec{y}}{\partial \vec{x}} \frac{\partial \vec{x}}{\partial s} + \frac{\partial \vec{y}}{\partial s}$$

$$(23) \qquad \frac{\partial \vec{y}}{\partial \vec{x}} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_n}{\partial x_1} & \cdots & \frac{\partial y_n}{\partial x_n} \end{pmatrix}$$

(24) follows from (22), where I is the n-dimensional unit matrix.

$$(24) \qquad \frac{\partial \vec{x}}{\partial s} = \left(I - \frac{\partial \vec{y}}{\partial \vec{x}}\right)^{-1} \frac{\partial \vec{y}}{\partial s}$$

Let H be a scaler function of $\vec{x}$ and $\vec{S}$, and P be a scalar function of $\vec{S}$ such that $P = H$ when $\vec{x} = \vec{y}$. Where H is differentiable, we get the following.

$$(25) \qquad \begin{aligned} \frac{\partial P}{\partial s} = \frac{\partial H}{\partial \vec{x}} \frac{\partial \vec{x}}{\partial s} + \frac{\partial H}{\partial s} &= \frac{\partial H}{\partial \vec{z}} \left(I - \frac{\partial \vec{y}}{\partial \vec{x}}\right)^{-1} \frac{\partial \vec{y}}{\partial s} + \frac{\partial H}{\partial s} \\ &= \vec{z} \frac{\partial \vec{y}}{\partial s} + \frac{\partial H}{\partial s} \end{aligned}$$

Here $\vec{z}$ is defined by (26), from which we obtain (27).

$$(26) \quad \vec{z} = \frac{\partial H}{\partial \vec{x}} \left(I - \frac{\partial \vec{y}}{\partial \vec{x}}\right)^{-1} \qquad (27) \quad \vec{z} = \vec{z} \frac{\partial \vec{y}}{\partial \vec{x}} + \frac{\partial H}{\partial \vec{x}}$$

Thus, $\vec{z}$ is computed via spreading activation based on (27). So as a whole we are to do double-layered spreading activation, the first layer for $\vec{x}$ and the next for $\vec{z}$. We omit mathematical discussions on the convergence of spreading activation. Finally, $\frac{\partial P}{\partial s}$ can be obtained from (25). We have avoided calculating $\frac{\partial \vec{x}}{\partial s}$, which would be a very complex computation. Note that $\frac{\partial x_i}{\partial s}$ is not zero for most x_i, whereas $\frac{\partial \vec{y}}{\partial \vec{x}}$ and $\frac{\partial \vec{y}}{\partial s}$ are sparse.

Acknowledgments

The author would like to thank DEN Yasuharu, Mark Gawron, Jerry Hobbs, NAGAO Katashi, NAKASHIMA Hideyuki, Manny Rayner, Ivan Sag, TUTIYA Syun, and Rémi Zajac for valuable discussions and comments. He is also indebted to ASOH Hideki for drawing his attention to generalized backpropagation. Special thanks go to NISHIKAWA Noriko, NAGATA Kazumi, and MIYATA Takasi for implementation works.

References

Anderson, J. R. (1983). *The Architecture of Cognition*. Harvard University Press, Cambridge.

Charniak, E. and Shimony, S. E. (1990). Probabilistic semantics for cost based abduction. In *Proceedings of AAAI '90*. pp. 106–111.

Charniak, E. (1986). A neat theory of marker passing. In *Proceedings of AAAI '86*, pp. 584–588.

Hasida, K. (1991). Common heuristics for parsing, generation, and whatever ... In *Proceedings of the Workshop on Reversible Grammar in Natural Language Processing*, Berkeley.

Hobbs, J., Stickel, M., Appelt, D., and Martin, P. (1990). Interpretation as abduction. Tech. rep. Technical Note 499, SRI International.

Norvig, P. (1989). Marker passing as a weak method for text inferencing. *Cognitive Science*, *13*, 569–620.

Pineda, F. J. (1988). *Generalization of Backpropagation to Recurrent and Higher Order Neural Networks*, pp. 602–611.

Stickel, M. E. (1989). Rationale and methods for abductive reasoning in natural-language interpretation. In Studer, R. (Ed.), *Proceedings, Natural Language and Logic, International Scientific Symposium*, No. 459 in Lecture Notes in Artificial Intelligence, pp. 233–252. Springer Verlag.

Suttner, C. B. and Ertel, W. (1990). Automatic acquisition of search guiding heuristics. In *Proceedings of the 10th International Conference on Automated Deduction (CADE)*, pp. 470–484.

Waltz, D. and Pollack, J. (1985). Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, *9*, 51–74.

[11] Treatment of binding could probably be ascribed to the general case of consistency checking plus transitivity energy.

[12] In the current formulation, $y_i = \frac{1}{1+\exp(Y_i)}$, where Y_i is a polynomial not involving x_i.