A Parser for Harmonic Context-Free Grammars

John Hale (hale@cogsci.jhu.edu)
Department of Cognitive Science
The Johns Hopkins University
3400 North Charles Street; Baltimore MD 21218-2685
Paul Smolensky (smolensky@cogsci.jhu.edu)
Department of Cognitive Science
The Johns Hopkins University
3400 North Charles Street; Baltimore MD 21218-2685

Abstract

Harmonic Grammar is a connectionist-derived grammar formalism, of which Optimality Theory is a kind of limiting case. Harmonic Grammar is expressive enough to specify the trees that are correct parses on a given context-free grammar. Here, we show how to construct a connectionist parsing network which finds correct parses given a sentence, of if none exist, signals a rejection. Finally, a brief comparison to other connectionist parsing work is provided.

Harmonic grammar is a grammar formalism which uses only soft rules of the following form:

> If condition X is violated, then the well-formedness (Harmony) of the structure is diminished by C_X.
>
> [Legendre et al., 1990, 388]

A linguistic theory in Harmonic grammar is a set of soft rules and a set of representational possibilities. Applying the soft rules to any representation yields the Harmony value for that representation. A representation with maximal Harmony from some class defined by common substructure — the inputs — is said to be the winning candidate. In Optimality Theory [Prince and Smolensky, 1993] the soft rules are ranked and the C_X values are arranged so that no number of violations of lower-ranked soft rules could ever outweigh a violation by a higher ranked one.

Harmonic grammar is expressive enough to specify the context-free languages [Smolensky, 1993]. Here Harmony maximization is made to serve as a processing algorithm for a parser. The design for the parser is a way of systematically arranging networks of threshold logic units so that they implement exactly the Harmonic grammar rules for context-free grammars. They maximize the Harmony of representations that share the same terminal entries. The general idea is that these networks do this by de-activating pieces of bad analyses until only correct analyses remain.

How Harmonic Grammar specifies context-free languages

Harmonic context-free grammar rules (as first presented in [Smolensky, 1993]) are integer occurrence and co-occurrence penalties that are defined on trees. Trees

Figure 1: A legal derivation?

whose Harmony value is zero are successful derivations which prove that their yield is generated on the grammar. All others have negative Harmony which indicates that the yield is not in the language generated by the grammar.

Derived from connectionist principles, Harmonic grammar assumes the following form for the Harmony function.

$$H(\mathbf{a}) = \sum_{\alpha<\beta} a_\alpha W_{\alpha\beta} a_\beta + \sum_{\alpha} a_\alpha b_\alpha \qquad (1)$$

In this case, **a** is the representation of a parse tree as a vector. W is a symmetric weight matrix, and H is a sum of terms containing pairs of elements from the vector **a**. As a consequence of this form of the Harmony function, the conditions X in the soft rules are restricted to referring to at most two structures; Harmony maximization is quadratic optimization.

This is an apparent problem for phrase-structure grammar, since no pairwise check of any two symbols from the tree depicted in figure 1 on the grammar $\begin{array}{lcl} X & \rightarrow & AP \\ X & \rightarrow & QB \end{array}$ could reveal that the tree is not a valid derivation of AB from X. It would seem that to check a rule with two children, rules that refer to three pieces of the representation at once are needed, implying a cubic Harmony function. But if this is so, then surely to check a rule with three children would require a quartic Harmony function. Rather than adopting Harmony functions of higher and higher degree, Harmonic context-grammars are defined for context-free grammars[1] in a special normal form where pairwise evaluation is sufficient to check global wellformedness: Harmonic Normal Form.

[1] Throughout, V is the set of all grammar symbols, Σ is the subset of V which are terminals, R is the set of rules or productions represented as (symbol,string) pairs, and S is the distinguished start symbol. See [Lewis and Papadimitriou, 1981] for notation, definitions and fundamental results on context-free grammars.

Definition 1 (branchingrhs) *Let $G = (V, \Sigma, R, S)$ be a context-free grammar in Chomsky Normal Form, $A \in V - \Sigma$ a nonterminal from G and γ a string in V^*. Then* $\text{branchingrhs}(A) = \{A \rightarrow \gamma \in R : |\gamma| > 1\}$

Definition 2 (Unique Branching) *A context-free grammar $G = (V, \Sigma, R, S)$ satisfies the Unique Branching condition if, for all nonterminals $A \in V - \Sigma$,* $|\text{branchingrhs}(A)| \leq 1$.

Unique branching insists that for every parent, at most one ordered pair of children is licensed by the grammar. This is the condition that defines Harmonic Normal Form and makes pairwise evaluation sufficient to specify context-free grammar trees. For example, the apparent problem mentioned previously would be solved if only the following grammar, which satisfies the Unique Branching condition, could be used instead.

$$\begin{array}{rcl} X & \rightarrow & X[1] \\ X & \rightarrow & X[2] \\ X[1] & \rightarrow & AP \\ X[2] & \rightarrow & QB \end{array}$$

On this grammar, the tree in figure 1 is assigned negative Harmony. If X had been expanded by the first rule, the parent would be $X[1]$ and the tree would be penalized for lacking P. If the parent were $X[2]$ the tree would be penalized for lacking Q. The extra nonterminal encodes which original context-free rule was used, but this contextual information is not needed at higher levels of the parse tree, and the unary rules helpfully remove it.

When the grammar satisfies the Unique Branching condition, a natural interaction between Harmonic grammar rules becomes sufficient to evaluate local trees. Because H adds up harmony penalties, the grammar effectively computes an "AND" at the site of each bracketed parent. All that remains is to specially balance the soft rule weights so that pairs in local trees licensed by the grammar exactly balance out to 0 Harmony and those in ill-formed local trees receive some kind of harmony penalty, ultimately leading to $H < 0$ for the whole tree. A set of rules that does this, G_H, is given below.

G_{HNF}	G_H
a	R_a: If a is at a node, add -1 to H
A	R_A: If A is at a node, add -2 to H
$A[i]$	$R_{A[i]}$: If $A[i]$ is at a node, add -3 to H
start symbol S	R_{root}: If S is at the root, add $+1$ to H
$A \rightarrow \alpha$ ($\alpha = a$ or $A[i]$)	If α is a left child of A, then add $+2$ to H
$A[i] \rightarrow BC$	If B is a left child of $A[i]$, add $+2$ to H If C is a right child of $A[i]$, add $+2$ to H

[Smolensky and Legendre, 2001, chapter 10]

Grammar preprocessing

The penalties that figure into the Harmonic grammar rules are going to be connection weights and unit biases in a neural network that parses the grammar. The relation between the grammar and the neural network is established by two grammar transformations. The first ensures that the Unique Branching condition is upheld.

Definition 3 (HNF transform) *Let $G = (V, \Sigma, R, S)$ be a context-free grammar in Chomsky Normal Form and let $A, B, C, X \in V - \Sigma$. The HNF transform HNF of G is a new grammar $HNF(G) = (V', \Sigma, R', S)$ where for each nonterminal A that appears in i branching rules of the form $A \rightarrow BC$, each such rule is replaced by two new rules containing a new nonterminal not in $V - \Sigma$, having the forms $A \rightarrow A[i]$ and $A[i] \rightarrow BC$. Call the set of new nonterminals that appear in these additional rules* $\text{bracket}(V')$. *The transformed set V' is the union of* $\text{bracket}(V')$, *the old nonterminals $X \in V - \Sigma$ such that* $|\text{branchingrhs}(X)| = 0$ *and the old terminals Σ.*

If a symbol is an element of the set $\text{bracket}(V')$ it is called "bracketed" otherwise it is "unbracketed."

The second transformation adds information about string positions to every rule, and restricts the grammar to only describing sentences of a certain maximum length. Since this maximum can be arbitrary large, it seems reasonable to maintain that context-free grammars for infinite languages are described in the limit [Charniak and Santos, 1987].

The annotation of string positions enables grammar symbols to directly serve as parser items. An item B_{jm} is an assertion about the input string that means "there is a constituent of type B that spans sentence positions j to m."

Definition 4 (Itemification of a binary rule) *The itemification of a binary context-free rule $A \rightarrow BC$ to a sentence length ℓ is the set of rules given by the schema $A_{jkm} \rightarrow B_{jk} C_{km}$ for all $j, k, m = 0 \ldots \ell$ such that $j < k \leq m$.*

Definition 5 (Itemification of a unary rule) *The itemification of a unary context-free rule $A \rightarrow B$ to a sentence length ℓ is the set of rules given by the schema $A_{jm} \rightarrow B_{jkm}$ for all $j, k, m = 0 \ldots \ell$ such that $j < k \leq m$.*

Grammars resulting from both transformations include complex symbols of the form $A[i]_{jkm}$. These symbols express the assertion that there is an A-type constituent spanning sentence positions j to m which was derived via the i^{th} A-rule, and the left child's yield stops at position k. In this way, k plays the role of a back-pointer that addresses a bracketed parent's children.

Definition 6 (Itemification of a grammar) *Let $G = (V, \Sigma, R, S)$ be a context-free grammar in Chomsky Normal Form. The itemification of G carried out for a sentence length ℓ is a grammar $ITEM(G, \ell) = (V', \Sigma', R', S')$ in which*

1. *Σ' contains $\ell - 1$ symbols labeled v_{jj+1} (where $j = 0 \ldots \ell - 1$) for each terminal symbol v in Σ.*
2. *S' is a new start symbol labeled $S_{0\ell}$*
3. *R' contains the itemification of each rule r in R.*

and V' consists of all the symbols appearing in S', R', Σ'.

Itemification ensures that children are directly adjoining, and in the right order. For example, (neglecting k's for a moment) if the grammar contains $X_{13} \rightarrow Y_{12}\ Z_{23}$ it will definitely not contain $X_{13} \rightarrow Y_{13}\ Z_{23}$ where a part of Z's yield — the symbol from position 2 to position 3 — is enveloped by Y's yield.

It is also convenient to define the width of two-index itemified symbols X_{ij} from V' as $width(X) = j - i$. Further, we suggest (without going into the proofs here[2]) that there are cover homomorphisms between proper parse relations on each of $ITEM(G,\ell)$ and $HNF(G)$ and G. In the case of $ITEM(G)$ this homomorphism is only defined for parses of sentences of length ℓ. For these sentences, call these homomorphisms f_{ITEM} and f_{HNF}. These are the "inverse mappings" that, given a parse on a transformed grammar, supply a parse on the untransformed grammar — essentially undoing the work of their namesakes. The basic idea is that both transformations only add or rename rules, rather than deleteing them.

Finally, define the parent set of a grammar symbol to be the set of all nonterminals that appear on the left-hand side in rules that involve the symbol in question on the right-hand side.

Definition 7 (Parent set) *Let $G = (V,\Sigma,R,S)$ be a context-free grammar and $\gamma_0,\gamma_1 \in V^*$ Then the set of all possible parents of an element $X \in V - \Sigma$ is* $\text{parents}(X,G) = \{P : \exists \gamma_0,\gamma_1 \in V^* \text{ such that } P \rightarrow \gamma_0 X \gamma_1 \in R\}$

Every context-free grammar with a finite number of rules has a "maximal parent multiplicity" $p_{\max(G)}$, the highest number of possible parents for any symbol. All of these concepts and definitions will be used to completely specify the parsing network in the next section.

Hopfield network

The parsing network is a Hopfield network with units whose states take on just the values 0 and 1. The network shall be constructed to parse the grammar $ITEM(HNF(G),\ell) = (V',\Sigma',R',S')$ with maximal parent multiplicity $p_{\max} = p_{\max(ITEM(HNF(G),\ell))}$. There are $\alpha = 1 \ldots |V'|$ threshold logic units which update themselves according to the transition rule

$$\begin{aligned} a_\alpha &\rightarrow 0 \text{ if } \sum_{\alpha \neq \beta} W_{\alpha\beta} a_\beta + b_\alpha < 0 \\ &\rightarrow 1 \text{ if } \sum_{\alpha \neq \beta} W_{\alpha\beta} a_\beta + b_\alpha \geq 0 \end{aligned}$$

where $W_{\alpha\beta}$ are connection weights and b_α are biases. Let f_α denote the application of the transition rule to the α^{th} threshold logic unit. Then a network update f_{network} is defined by $f_{\rho(1)} \circ f_{\rho(2)} \circ \ldots \circ f_{\rho(|V'|)}$ where ρ indexes the entries in a random permutation of $1 \ldots |V'|$.

[2] See [Nijholt, 1980, chapter 2] for discussion of grammar covers.

Entries in the undirected $|V'| \times |V'|$ weight matrix W are indexed by grammar symbol and are only nonzero if one indexed symbol is bracketed and the other is unbracketed. Without loss of generality, identify the α^{th} grammar symbol as the bracketed one ($A[i]_{jkm}$) and the β^{th}, as the unbracketed one (A_{jm}).

- If there is a binary rule $\alpha \rightarrow \beta\gamma$ or $\alpha \rightarrow \gamma\beta$ the weight between units α and β is 1.
- If there is a unary rule $\alpha \rightarrow \beta$ then the weight between units α and β is the same as the maximal parent multiplicity, $p_{\max}$.
- Otherwise the weight is zero.

The vector of $|V'|$ biases, **b** is all negative. Component b_α is set to one of three possible values.

- If α is an unbracketed start symbol of $width = \ell$ then $b_\alpha = -1$, or, if α is not a start symbol,
- if α is bracketed then $b_\alpha = -(p_{\max} + 2)$, or else
- α is unbracketed and $b_\alpha = -(p_{\max} + 1)$

These weights and biases reflect the kinds of input that bracketed and unbracketed units need to be correctly supported by units representing parents above them and units representing children below them.

Bracketed units By the Unique Branching condition, these units have indegree three. They receive input from exactly one (unbracketed) parent and exactly two (unbracketed) children. If all three of these neighbors are in the 1 state, then the net input is $p_{\max} + 1 + 1$ representing, respectively, the contributions from the parent and each child. This exactly balances the bias $-(p_{\max} + 1)$ and keeps the bracketed unit in that state, always updating via the ≥ 0 transition. Bracketed units that are on are guaranteed to have correct parents and children on.

Unbracketed units These units can be connected to as many as $p_{\max}$ (bracketed) parents. Even in the worst case, in which all possible parents are in the 1 state, an unbracketed units' $-(p_{\max} + 1)$ bias makes sure that it can only be in the 1 state itself when supported by at least one unbracketed child. Unbracketed units that are on are guaranteed to have at least one correct child on.

By construction, W is symmetric (if two symbols are in a parent-child relationship they are also in a child-parent relationship) and has zeros along the diagonal (no two symbols are in dominance relationships with themselves), making the results on convergence of Hopfield networks [Hopfield, 1982] applicable.

During the network's operation, only the states of units associated with symbols of width > 1 may change. Width-1 units, specifying the input to be parsed, are

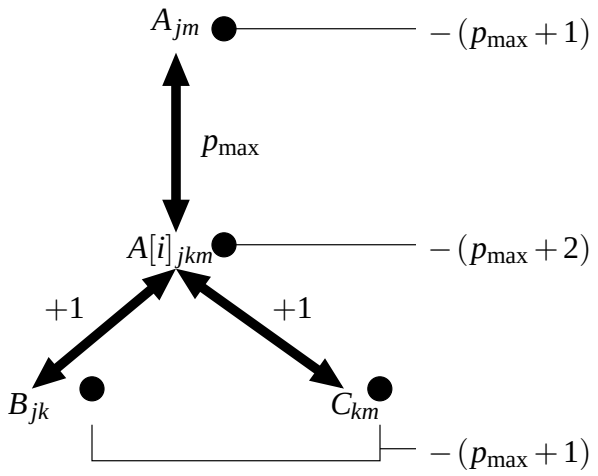

Figure 2: Fundamental parsing network block

clamped. For simple asynchronous updating, the change in a unit a_α's activation is

$$\Delta a_\alpha = \begin{cases} +1 & \text{if the old state was 0 and} \sum_\beta W_{\alpha\beta}a_\beta + b_\alpha \geq 0 \\ 0 & \text{if the old state was 1 and } \sum_\beta W_{\alpha\beta}a_\beta + b_\alpha \geq 0 \\ 0 & \text{if the old state was 0 and } \sum_\beta W_{\alpha\beta}a_\beta + b_\alpha < 0 \\ -1 & \text{if the old state was 1 and} \sum_\beta W_{\alpha\beta}a_\beta + b_\alpha < 0 \end{cases}$$

The corresponding change in Harmony is

$$\Delta H = \Delta a_\alpha \left(\sum_{\alpha \neq \beta} W_{\alpha\beta}a_\beta + b_\alpha \right)$$

Since Δa_α is positive when $\left(\sum_\beta W_{\alpha\beta}a_\beta + b_\alpha\right)$ is, and Δa_α is negative when $\left(\sum_\beta W_{\alpha\beta}a_\beta + b_\alpha\right)$ is, H is increasing whenever $\Delta a_\alpha \neq 0$. But H is also clearly bounded from above, at least by $\sum_\alpha |b_\alpha| + \sum_\alpha \sum_{\beta>\alpha} |W_{\alpha\beta}|$, and so cannot increase indefinitely. Therefore the dynamics reaches a maximum, at which point $\Delta H = 0$. At this point $f_{\text{network}}(\mathbf{a}^{\text{stable}}) = \mathbf{a}^{\text{stable}}$.

Note that unbracketed units will only turn off if all units representing their (bracketed) child-options are off. Bracketed units will switch off if any of their neighbors switch off. The basic arrangement repeated throughout the network is depicted in figure 2.

Theorem 1 (Correctness) *Let* $\mathbf{a}^{\text{stable}}$ *be a stable state of a Hopfield network constructed as above to parse* $ITEM(HNF(G),\ell) = (V',\Sigma',R',S')$ *whose initial state* $\mathbf{a}^0$ *is determined by the input string* $v = v_{01}v_{12}\ldots v_{\ell-1\ell}$ *in the following way:*

- *If* v_{ij} *is contained in the input and the* m^{th} *grammar symbol is* v_{ij}*, then the* m^{th} *component of the initial state is 1.*
- *If* v_{ij} *is not contained in the input and the* m^{th} *grammar symbol is* v_{ij}*, then the* m^{th} *component of the initial state is 0.*
- *Otherwise the* m^{th} *component of the initial state is 1.*

Then, if the final state is 1 for a unit associated with a start symbol of $\text{width}(\alpha_m) = \ell$*, then the set* $\{\alpha_m : a_m = 1\}$ *determine a shared packed forest [Tomita, 1986] of v-parses on G. Otherwise the parser has rejected v.*

Proof: We must show that if a unit representing a start symbol spanning the entire input is in the 1 state at $\mathbf{a}^{\text{stable}}$, then all trees determined by sequences of choices about which activated bracketed child-units to move to from activated unbracketed parent-units, going from the root to the leaves, are correct parses of *v*.

If a unit representing a start symbol spanning the entire input is in the 1 state, it must be because its -1 bias has been counterbalanced by activation from at least one child, since by definition there are no parents for start symbols.

Select one of these bracketed children that are also in the 1 state. As bracketed units, being on implies a full and correct set of neighbors in the 1 state. Two of these neighbors are bracketed children.

Continue the proof by selecting arbitrarily from among the activated bracketed children at each successive unbracketed unit. This selected unit must be part of a correct parse in virtue of a grammar rule, or it would not be activated. Eventually because the network is finite this selecting and traversing must end at clamped, unbracketed units of width 1.

Each selection of a bracketed unit from the perspective of an unbracketed parent is an unpacking of one choice that has been packed in the shared-packed parse forest. The representation is shared because no symbol is represented more than once.

Since all of the units that are on are part of some correct parse corresponding to some sequence of bracketed-rule selections, for each such correct parse there must be a sequence $\pi = \pi_0, \pi_1, \ldots, \pi_n$ of rules which each describe one piece of local tree structure. Since the above argument did not depend on which bracketed-rule unit was selected at each point, all sequences of selections result in correct parses and all the resulting π stand in a proper parse relation with *v* on $ITEM(HNF(G),\ell)$. The proper parse relation on *G* is $f_{HNF} \circ f_{ITEM}(v,\pi)$.

Corollary 1 (Completeness & the initial state) *If the initial state* $\mathbf{a}^0$ *includes enough* $a_j = 1$ *to describe a parse of w then that parse will be represented in the final state.*

Proof: The parser's operation can only switch bracketed units $\alpha \in \text{bracket}(V')$ in the 1 state into the 0 state, and not the other way around, because *W* is constructed so that α's row, W_α has exactly three nonzero entries, and their sum is $(p_{\max}+2)$. By construction these nonzero entries are at exactly the columns for the two unique children and unique parent. α's bias has also been constructed to be exactly $-(p_{\max}+2)$. So given that α is on, it must be that all of α's neighbors are on and that they are licensed by the grammar. But bracketed units in the

1 state with *correct* parents and children do not change their state. So if all correct parents and children from a parse are present in the initial state, and no bracketed units can switch off, then all correct parents and children must still be on in the stable state.

Example

As an example, consider the ambiguous grammar $\begin{array}{rcl} S & \rightarrow & AB \\ A & \rightarrow & AA \end{array}$, where S is the start symbol. We follow [Nijholt, 1990] in assuming that preterminal rules such as $A \rightarrow$ v don't play a role and parsing may begin at the nonterminals. The input sequence *AAAB* is ambiguous on this grammar between an analysis where the left most pair of *A*'s form a constituent $(((AA)A)B)$ and one where the middle two *A*'s form one $((A(AA))B)$. Either analysis ultimately will be compatible with a correct parse.

To build a parser for this grammar for sentences of length $\ell = 4$, the grammar is first transformed by HNF. Even though the grammar was already in Harmonic Normal Form, HNF here serves to explicitly encode the unary/binary status of each rule.

$$\begin{array}{rcl} A[1] & \rightarrow & AA \\ A & \rightarrow & A[1] \\ S[1] & \rightarrow & AB \\ S & \rightarrow & S[1] \end{array}$$

Itemification is then performed, resulting in a larger grammar in which all possible rule applications have been annotated with string position indices for every possible location at which they could be applied.

$$\begin{array}{rcl} A[1]012 & \rightarrow & A01A12 \\ A[1]013 & \rightarrow & A01A13 \\ A[1]023 & \rightarrow & A02A23 \\ \vdots & \vdots & \vdots \\ A01 & \rightarrow & A[1]011 \\ A02 & \rightarrow & A[1]012 \\ \vdots & \vdots & \vdots \\ S[1]012 & \rightarrow & A01B12 \\ S[1]013 & \rightarrow & A01B13 \\ \vdots & \vdots & \vdots \\ S01 & \rightarrow & S[1]011 \\ S02 & \rightarrow & S[1]012 \\ S03 & \rightarrow & S[1]013 \\ S03 & \rightarrow & S[1]023 \\ \vdots & \vdots & \vdots \end{array}$$

There are 54 units, each associated with symbol in this transformed grammar. The network runs until it reaches a stable state, at every transition increasing Harmony. The Harmony values for one simulation run are shown in figure 3.

In the final state, units representing the symbols *S*04, $S[1]034$, *A*03, $A[1]013$, $A[1]023$, *A*02, $A[1]012$, $A[1]123$, *A*13, *A*01, *A*12, *A*23 and *B*34 are all in the one state, and all others are in the zero state. Since the start symbol

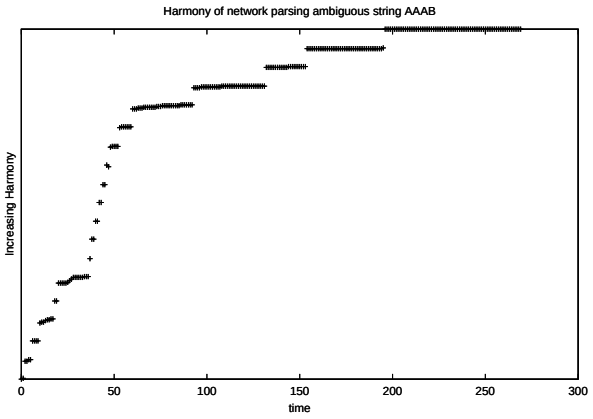

Figure 3: Operation of parsing network on example grammar

*S*04 is activated, we can interpret the parse as having accepted *AAAB*. To determine the parses, we conceptually traverse downward from *S*04 to $S[1]034$, and then to *A*03 and *B*34. From *A*03, there is a choice of which of the two ambiguous parses to be taken. Both are represented by activated units all of which are part of correct parses that figure into a shared packed parse forest. Selecting $A[1]013$ determines one parse, and selecting $A[1]012$ determines the other, just as in chart parsing.

Comparison

One difference between the architecture of this Harmonic grammar parser and various other deterministic connectionist parsers ([Fanty, 1985], [Nijholt, 1990], [Sikkel, 1997]) resides in the lack of central control over evaluation order. The formulation here is in terms of fixed points for randomly-ordered, Harmony-increasing updates. Despite this apparent freedom, the de-activation of unsupported units proceeds bottom-up, an order effect which follows from the connectivity of the network.

As in treatments that avoid Harmony minima through simulated annealing ([Selman, 1985], [Howells, 1988]) the parser's progress can be tracked by examining the current value of H, although here the parser state is not probabilistic.

Other comparisons invite exciting extensions. The work of Hopfield [Hopfield, 1984] suggests that the results for linear threshold units should extend straightforwardly to more realistic neural models, while that of Stolcke [Stolcke, 1989] points the way to more linguistically realistic unification-based grammars.

Perhaps the most intriguing comparison is to Optimality Theory itself. As in Optimality Theory, where all representational possibilities are said to come from *Gen*, the Hopfield network parser described here starts from a state in which all possible constituents are represented. As processing progresses, units representing constituents that lack support given the input string deactivate themselves. In this way the parser acts as a filter

that removes ungrammatical analyses from a initial universe of conceivable analyses. The parser is implementing constraints from a *Con* that contains the soft rules G_H. However, because constraint interaction is numerical, strict domination does not necessarily hold: two or more violations of R_a can be just as bad, or worse, than a single violation of R_A even though $R_A \gg R_a$.

Conclusion

In fact, the ultimate goal of the larger research program of which this work forms a part is the integration of insights from three different sources: formal grammar, constraint-based processing, and linguistic theory. Harmonic grammar is a competence theory that can declaratively specify context-free and other formal languages. Here we have shown that a simple performance theory can be constructed that incorporates this competence theory in a relatively straightforward way into a procedural specification for parsing using abstract neural computing units. In the overall program, however, the role of formal languages is to benchmark theories of human parsing. The analogies to OT in this simple performance theory suggest that such an architecture may be flexible enough to accommodate insights into human language processing from OT syntax and constraint-based approaches to psycholinguistics.

References

[Charniak and Santos, 1987] Charniak, E. and Santos, E. (1987). A connectionist context-free parser which is not context-free, but then it is not really connectionist either. In *Proceedings of the 9th Annual Conference of the Cognitive Science Society*, pages 70–77, Hillsdale, NJ. Erlbaum.

[Fanty, 1985] Fanty, M. (1985). Context-free parsing in connectionist networks. Technical Report TR147, Rochester Computer Science Department.

[Hopfield, 1982] Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States*, 79:2554–2558.

[Hopfield, 1984] Hopfield, J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences of the United States*, 81:3088–3092.

[Howells, 1988] Howells, T. (1988). VITAL: a connectionist parser. In *Proceedings of 10th Annual Meeting of the Cognitive Science Society*, pages 18–25.

[Legendre et al., 1990] Legendre, G., Miyata, Y., and Smolensky, P. (1990). Harmonic grammar – a formal multi-level connectionist theory of linguistic well-formedness: Theoretical foundations. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 388–395, Cambridge MA. Erlbaum.

[Lewis and Papadimitriou, 1981] Lewis, H. R. and Papadimitriou, C. H. (1981). *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, NJ.

[Nijholt, 1980] Nijholt, A. (1980). *Context-Free Grammars: Covers, Normal Forms and Parsing*. Number 93 in Lecture Notes in Computer Science. Springer-Verlag, Berlin.

[Nijholt, 1990] Nijholt, A. (1990). Meta-parsing in neural networks. In Trappl, R., editor, *Proceedings of the 10th European Meeting on Cybernetics and Systems Research*, pages 969–971, Teaneck NJ. World Scientific.

[Prince and Smolensky, 1993] Prince, A. and Smolensky, P. (1993). *Optimality theory: constraint interaction in generative grammar*. MIT Press. Forthcoming.

[Selman, 1985] Selman, B. (1985). Rule-based processing in a connectionist system for natural language understanding. Technical Report CSRI-168, University of Toronto Computer Science Department.

[Sikkel, 1997] Sikkel, K. (1997). *Parsing Schemata: a framework for specification and analysis of parsing algorithms*. EATCS Texts in Theoretical Computer Science. Springer.

[Smolensky, 1993] Smolensky, P. (1993). Harmonic grammars for formal languages. *Advances in neural information processing systems*, 5:847–854.

[Smolensky and Legendre, 2001] Smolensky, P. and Legendre, G. (2001). *Architecture of the Mind/Brain: neural computation, optimality and universal grammar in cognitive science*. Forthcoming.

[Stolcke, 1989] Stolcke, A. (1989). Unification as constraint satisfaction in structured connectionist networks. *Neural Computation*, 1:559–567.

[Tomita, 1986] Tomita, M. (1986). *Efficient parsing for natural language: a fast algorithm for practical systems*. Kluwer Academic Publishers, Boston.