Soft Constraints at Interfaces

Nick Pendar

Iowa State University

Proceedings of the GEAF 2007 Workshop

Tracy Holloway King and Emily M. Bender (Editors)

CSLI Studies in Computational Linguistics ONLINE

Ann Copestake (Series Editor)

2007

CSLI Publications

http://csli-publications.stanford.edu/

Abstract

Assuming that grammar is best modeled as a set of modules each representing a constraint system, like other human cognitive faculties, these constraint systems are bound to disagree and soft constraints are needed for conflict resolution among modules. This paper outlines an approach to incorporate soft constraints among grammar modules. The paper first goes over the issue of modularity and conflicting requirements. It then summarizes a generalized theory of soft constraint satisfaction (SCSP). It then outlines a linguistic constraint system based on SCSP theory. Finally, it shows how the type antecedent constraints of HPSG can be extended in an SCSP-based framework to allow for constraint violations and gradient constraints.

1 Introduction

Inquiry in theoretical linguistics, cognitive science, and AI has led many researchers to believe that constraint-based approaches in modeling human behavior capture our understanding of the phenomena in question better than procedural approaches. The advantage of these approaches is that expressing what we know about the data in the form of constraints can capture generalizations more accurately and intuitively. A procedural formalization has the disadvantage of mixing *knowledge* with the *processing* of that knowledge. By keeping the two separate, we as researchers give ourselves the opportunity to improve each separately. Constraint-based linguistic theories have been making headway in better understanding of language. Two noteworthy examples are Head-Driven Phrase Structure Grammar (HPSG, Pollard and Sag, 1987, 1994), and Lexical Functional Grammar (LFG, Bresnan, 1982, 2001). Another remarkable example is Optimality Theory (OT, Prince and Smolensky, 1993); the underlying assumption in this theory is that constraints are not absolute (*crisp*), but instead they are violable (*soft*). Constraints are thus ranked according to their importance, and the form that violates the fewest high-ranking constraints is considered the optimal form. Constraint-based systems are also widely used to solve real-life problems in computer science. Network management, scheduling, and transportation problems are most easily solved in a constraint-based approach.

In cognitive science and AI, problems are envisioned as constituting discrete objects with constraints imposed on their interactions. A keyword in the preceding statement is *objects* as objects are more or less independent entities with certain properties and they perform a set of predefined functions. Several theories in cognitive science have found modular approaches beneficial. For example, Newell's (1990) unified theory of cognition paints a modular picture of the mind in which each cognitive faculty takes the form of a discrete entity that communicates with

[†]I thank Elizabeth Cowper, Elan Dresher, Frank Keller, Jean-Pierre Koenig and Gerald Penn for their insightful comments on my work. This work was partially funded by the Social Sciences and Humanities Research Council of Canada and Ontario Graduate Scholarships.

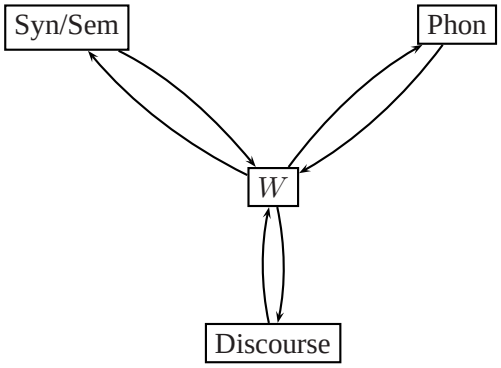

Figure 1: Modular grammar architecture proposed by Haji-Abdolhosseini (2003, 2005)

other modules. Oatley and Johnson-Laird (1987) and Oatley (1992) develop a theory of emotions within a larger context of cognitive science. This theory also relies on the fundamental assumption that human cognitive processes are modular and need to communicate with one another. In linguistics, Jackendoff (1992) also argues that human cognitive faculties form modules that need to communicate with one another. He also believes that each module (e.g., language, vision, or musical perception) is itself made up of its own sub-modules which in turn communicate with one another. Jackendoff (1997, 2002) argues for a tripartite architecture of grammar where phonological, morpho-syntactic and semantic components work in parallel and communicate at interface levels. Haji-Abdolhosseini (2003, 2005) takes a modular approach to HPSG where different linguistic modules interact through a shared list of domain objects.

However, disagreement among modules in any given intelligent system is a fact of life. Having crisp constraints, therefore, is not considered a desirable feature because as we gradually grow out of toy models and move towards approximating real-life problems, a system with crisp constraints quickly turns into what is known as an *over-constrained* system; i.e., one that yields no solution; or it becomes so complicated that it takes the system an inordinate amount of time to find an answer. In the AI community, several approaches have been proposed to remedy such problems. *Partial constraint satisfaction* (Freuder and Wallace, 1992), *constraint hierarchies* (Borning et al., 1992), *probabilistic soft constraint satisfaction* (Fargier and Lang, 1993), *valued constraint satisfaction* (Schiex et al., 1995), and *fuzzy soft constraint satisfaction* (Rosenfeld et al., 1976; Dubois et al., 1993; Ruttkay, 1994) are most notable.[1]

In computational linguistics, probabilistic approaches are dominant, and have led to some theoretical contributions (Abney, 1996, 1997; Bod, 1998; Bod, Hay and Jannedy, 2003a; Bod, Scha and Sima'an, 2003b; Foth, Menzel and Schröder,

[1]Some useful literature reviews can be found in Bistarelli (2001) and at `http://kti.ms.mff.cuni.cz/~bartak/constraints/` (accessed 9/11/2007).

2005; Schröder, 2002, among others). One notable approach within linguistics proper that relies on non-crisp constraints is OT.

Jackendoff (1997; 2002), who is a proponent of a modular approach, introduces *correspondence rules* that apply at the interfaces among major linguistic modules. Haji-Abdolhosseini (2005) observes that soft constraints tend to be interface constraints and hard constraints tend to be intramodular. This work advocates the use of soft constraints at interfaces.

From a practical point of view, there are additional reasons why it is important to do research in grammatical interfaces in constraint-based and multi-partite frameworks: A modular theory is easier for the researcher to work with. A grammar written in this approach is certainly more readable and more convenient to maintain. Furthermore, with the emergence of large-scale grammars a modular approach becomes even more significant to promote code readability and reuse.

2 A Generalized Theory of Constraint Satisfaction

Constraint programming has been a very exciting area of research in artificial intelligence in the past decade. The holy grail of constraint programming is to find ways of describing a problem in terms of constraints without having to worry about how those constraints are processed in finding a solution. This will allow one to concentrate on the problem as opposed to the details of algorithms and processing (for an excellent introduction, see Marriott and Stuckey, 1998). This constraint-based view of characterizing problems has also found its way into linguistics. Head-Driven Phrase Structure Grammar (HPSG, Pollard and Sag, 1987, 1994), Lexical Functional Grammar (LFG, Bresnan, 1982, 2001), and Optimality Theory (OT, Prince and Smolensky, 1993) are all constraint-based theories of language, and their claim is that by expressing linguistic generalizations in terms of constraints, we are better able to see the phenomena involved without getting entangled in procedural details.

The problem of over-constrained systems has led researchers to seek ways of relaxing or weighting constraints so that the less important ones can be violated in favor of the more important ones. This is also a route that OT has taken.

In the following subsection, we review a generalized theory of soft constraint satisfaction introduced by Bistarelli (2001). This theory is based on a certain algebraic structure called the *semiring*. Based on a solid mathematical foundation, Bistarelli's theory of Semiring-based Constraint Satisfaction Problems (SCSP) illustrates that several of the previous models of soft constraint satisfaction are instances of SCSP. The next section provides a formal introduction to constraint satisfaction problems in general, and section 2.2 introduces Bistarelli's semiring-based account. We will also outline an SCSP-based extension to HPSG type antecedent constraints in section 3.1.

2.1 Constraint Satisfaction Problems

This subsection is based on section 1.1 of Bistarelli (2001).

DEFINITION 2.1 ***Constraint Satisfaction Problem*** *A Constraint Satisfaction Problem is a sextuple* $\langle V, D, C, con, def, a\rangle$ *where*

- V *is a finite set of variables, i.e.,* $V = \{v_1, \ldots, v_n\}$*;*
- D *is a set of values, called* the domain*;*
- C *is a finite set of constraints, i.e.,* $C = \{c_1, \ldots, c_m\}$*.* C *is ranked, i.e.,* $C = \bigcup_k C_k$ *such that* $c \in C_k$ *if* c *involves* k *variables;*
- *con is called the* connection function *and it is such that*

$$con : \bigcup_k (C_k \rightarrow V^k),$$

  where $con(c) = \langle v_1, \ldots, v_k\rangle$ *is the tuple of variables involved in* $c \in C_k$*;*

- def *is called the* definition function *and it is such that*

$$def : \bigcup_k (C_k \rightarrow \wp(D^k)),$$

  where $\wp(D^k)$ *is the power set of* D^k*, that is, all the possible subsets of* k*-tuples in* D^k*;*

- $a \subseteq V$*, and represent the* distinguished *variables of the problem.*

con describes which variables are involved in which constraint; def specifies which are the domain tuples permitted by the constraint. The set a is used to point out the variables of interest in the given Constraint Satisfaction Problem (CSP), i.e., the variables for which we want to know the possible assignments, compatible with all the constraints. This set is equal to V if all the variables are of interest. This does not have to be the case however. In fact, it is reasonable to think that the CSP representation of a problem contains many details (in terms of constraints and/or variables) which are needed for a correct specification of the problem but are not important as far as the solution of the problem is concerned.

The solution $Sol(P)$ of a CSP $P = \langle V, D, C, con, def, a\rangle$ is defined as the set of all instantiations of the variables in a which can be extended to instantiations of all the variables which are consistent with all the constraints in C.

DEFINITION 2.2 ***Tuple Projection and CSP Solution*** *Given a tuple of domain values* $\langle v_1, \ldots, v_n\rangle$*, consider a tuple of variables* $\langle x_{i1}, \ldots, x_{im}\rangle$ *such that for all* $j = 1, \ldots, m$*, there exists a* $k_j \in \{1, \ldots, n\}$ *such that* $x_{ij} = x_{kj}$*. Then the projection of* $\langle v_1, \ldots, v_n\rangle$ *over* $\langle x_{i1}, \ldots, x_{im}\rangle$*, written* $\langle v_1, \ldots, v_n\rangle_{|\langle x_{i1}, \ldots, x_{im}\rangle}$*, is the*

tuple of values $\langle v_{i1}, \ldots, v_{im} \rangle$*. The solution* $Sol(P)$ *of a CSP* $P = \langle V, D, C, con, def, a \rangle$ *is defined as*

$$\left\{ \langle v_1, \ldots, v_n \rangle_{|a} \text{ such that } \left\{ \begin{array}{l} v_i \in D \text{ for all } i; \\ \text{for all } c \in C, \langle v_1, \ldots, v_n \rangle_{|con(c)} \in def(c). \end{array} \right. \right\}$$

The solution to a CSP is therefore an assignment of a value from its domain to every variable, in such a way that every constraint is satisfied. We may want to find just one solution, with no preference as to which one, or all solutions.

2.2 A Semiring-Based Theory of Constraint Satisfaction Problems

The constraint satisfaction approach defined above clearly employs crisp constraints, which can lead to over-constrained problems. To solve the problem of over-constrained CSPs, researchers have proposed several alternative approaches which enable one to relax some constraints in order to find a solution to the problem. As mentioned earlier, Bistarelli (2001) shows that some of these approaches (e.g., probabilistic, fuzzy, and weighted CSPs) can be thought of as special instances of a more general soft-constraint satisfaction framework, which he calls the Semiring-based Constraint Satisfaction Problems (SCSP). The present and the following sections, which are based on Chapter 2 of Bistarelli (2001), briefly introduce this theory.

Bistarelli's main idea is that

> ... a semiring (that is, a domain plus two operations satisfying certain properties) is all that is needed to describe many constraint satisfaction schemes. In fact, the domain of the semiring provides the levels of consistency (which can be interpreted as cost, or degree of preference, or probabilities, or others), and the two operations define a way to combine constraints together. More precisely, we define the notion of constraint solving over any semiring. Specific choices of the semiring will then give rise to different instances of the framework, which may correspond to known or new constraint solving schemes.

DEFINITION 2.3 **Semiring** *A semiring is a quintuple* $\langle A, sum, \times, 0, 1 \rangle$ *such that*

- A *is a set and* $0, 1 \in A$*;*
- sum*, called the* additive operator*, is a commutative, i.e.,* $sum(a, b) = sum(b, a)$*, and associative, i.e.,* $sum(a, sum(b, c)) = sum(sum(a, b), c)$*, operation with* 0 *as its identity element, i.e.,* $sum(a, 0) = a = sum(0, a)$*;*
- $\times$*, called the* multiplicative operator*, is an associative operation such that* 1 *is its* identity element *and* 0 *is its* absorbing element*, i.e.,* $a \times 0 = 0 = 0 \times a$*;*
- $\times$*, distributes over* sum*, i.e., for any* $a, b, c \in A, a \times sum(b, c) = sum((a \times b), (a \times c))$.

The reader may have noted that the set of real numbers between 0 and 1 (inclusive) together with arithmetic $+$ and $\times$ form a semiring, for example.

Bistarelli introduces semirings with additional properties for the two operations. He calls this algebra a *c-semiring* (c for "constraint"), and defines it as follows:

DEFINITION 2.4 **C-Semiring** *A c-semiring is a quintuple* $\langle A, \oplus, \otimes, 0, 1\rangle$ *such that*

- A *is a set and* $0, 1 \in A$*;*
- $\oplus$ *is defined over (possibly infinite) sets of elements of* A *as follows:*[2]
  - *for all* $a \in A, \sum(\{a\}) = a$*;*
  - $\sum(\emptyset) = 0$ *and* $\sum(A) = 1$*;*
  - $\sum(\bigcup A_i, i \in I) = \sum(\{\sum(A_i), i \in I\})$ *for all sets of indices* I *(flattening property);*
- $\otimes$ *is a binary associative and commutative operation such that* 1 *is its identity element and* 0 *is its absorbing element;*
- $\otimes$ *distributes over* $\oplus$*, i.e., for any* $a \in A$ *and* $b \subseteq A, a \otimes \sum(B) = \sum(\{a \otimes b, b \in B\})$.

The fact that $\oplus$ is defined over *sets* of elements, and not *pairs* or *tuples*, automatically makes such an operation commutative, associative, and idempotent. It is also possible to show that 0 is the identity element of $\oplus$. By using the flattening property, we get $\sum(\{a, 0\}) = \sum(\{a\}\emptyset) = \sum(\{a\}) = a$. This means that a c-semiring is a semiring (where the sum operation is $\oplus$) with some additional properties. It is also possible to prove that 1 is the absorbing element of $\oplus$. By flattening and by the fact that we set $\sum(A) = 1$, we get $\sum(\{a, 1\}) = \sum(\{a\} \cup A) = \sum(A) = 1$.

According to Bistarelli, the advantage of using c-semirings instead of semirings is as follows: The idempotency of the $\oplus$ operation is needed in order to define a partial ordering $\leq_s$ over the set A, which will enable us to compare different elements of the semiring. Such a partial order is defined as: $a \leq_s b$ iff $a \oplus b = b$. Intuitively, $a \leq_s b$ means that b is "better" than a, or, from another point of view, that between a and b, the $\oplus$ operation chooses b. This ordering is used to choose the "best" solution in constraint problems.

Given any c-semiring $S = \langle A, \oplus, \otimes, 0, 1\rangle$, consider the relation $\leq_s$ over A such that $a \leq_s b$ iff $a \oplus b = b$. Then Bistarelli proves that $\leq_s$ is a partial order. He also proves that $\oplus$ and $\otimes$ are monotones over $\leq_s$. That is, given any c-semiring $S = \langle A, \oplus, \otimes, 0, 1\rangle$, consider the relation $\leq_s$ over A. Then that $\oplus$ and $\otimes$ are monotones over $\leq_s$ means that $a \leq_s a'$ implies $a \oplus b \leq_s a' \oplus b$ and $a \otimes b \leq_s a' \otimes b$.

[2] We use $\oplus$ in infix notation for a two-element set, and the symbol $\sum$ in prefix notation for more elements.

Since 1 is also the absorbing element of the additive operation, then $a \leq_s 1$ for all a. Thus 1 is the maximum element of the partial ordering. This implies that the $\otimes$ operation is *intensive*, that is, $a \otimes b \leq_s a$. This is important since it means that combining more constraints leads to a "worse" result in terms of the $\leq_s$ ordering.

Sometimes we need the $\otimes$ operation to be closed on a certain finite subset of the c-semiring.

DEFINITION 2.5 **AD-closed** *Given any c-semiring $S = \langle A, \oplus, \otimes, 0, 1\rangle$, consider a finite set $AD \subseteq A$. Then $\otimes$ is AD-closed if for any $a, b \in AD$, $(a \otimes b) \in AD$.*

It is shown that c-semirings can be assimilated to complete lattices. We also sometimes need to consider c-semirings where $\otimes$ is idempotent, which makes the c-semiring equivalent to distributive lattices.[3]

DEFINITION 2.6 **LUB, GLB, (Complete Lattice)** *Consider a partially ordered set S and any subset I of S. Then we define the following:*

- *an* upper bound *(resp.* lower bound*) of I is any element x such that for all $y \in I, y \leq x$ (resp., $x \leq y$);*
- *the* least upper bound (LUB) *(resp.* greatest lower bound (GLB) *of I is an upper bound (resp. lower bound) x of I such that for any other upper bound (resp. lower bound) x' of I, we have that $x \leq x'$ (resp., $x' \leq x$).*

A lattice *is a partially ordered set where every subset of two elements has a LUB and a GLB. A* complete lattice *is a partially ordered set where every subset has a LUB and GLB.*

Bistarelli proves that $\langle A, \leq_s\rangle$ is a complete lattice, which entails $\sum(I) = LUB(I)$ for any set $I \subseteq A$. Thus every subset I of A has a least upper bound (which coincides with $\sum(I)$). This means that $\langle A, \leq_s\rangle$ is a LUB-complete partial order. Note that the $\oplus$ operator coincides with the LUB of the lattice $\langle A, \leq_s\rangle$.

Bistarelli also proves that given a c-semiring $S = \langle A, \oplus, \otimes, 0, 1\rangle$ and a corresponding complete lattice $\langle A, \leq_s\rangle$, $\otimes$ is also idempotent. Furthermore, in the particular case in which $\otimes$ is idempotent and $\leq_s$ is total, we have that $a \oplus b = \max(a, b)$ and $a \otimes b = \min(a, b)$.

2.3 Constraint Systems and Problems

The notions of constraint system, constraint, and constraint problem in this theory are parametric with respect to the notion of c-semiring discussed in the previous section. Intuitively, a constraint system specifies the c-semiring $\langle A, \oplus, \otimes, 0, 1\rangle$ to be used along with the set of all variables and their domain D.

[3] For an introduction to lattices and ordered sets, see Davey and Priestley (1990).

DEFINITION 2.7 **Constraint System** *A constraint system is defined as a triple $CS = \langle S, D, V \rangle$, where S is a c-semiring, D is a finite set, and V is an ordered set of variables.*

A constraint over a given constraint system specifies the involved variables and the "allowed" values for them. More precisely, for each tuple of values (of D) for the involved variables, a corresponding element of A is given. This element can be interpreted as the tuple's weight, or cost, or level of confidence, etc.

DEFINITION 2.8 **Constraint** *Given a constraint system $CS = \langle S, D, V \rangle$, where $S = \langle A, \oplus, \otimes, 0, 1 \rangle$, a constraint over CS is a pair $\langle def, con \rangle$, where*

- *$con \subseteq V$, it is called the* type *of the constraint;*
- *$def : D^k \rightarrow A$ (where k is the cardinality of con) is called the* value *of the constraint.*

A constraint problem is then just a set of constraints over a given constraint system, plus a selected set of variables (thus a *type*). These are the variables of interest in the problem, i.e., the variables for which we want to know the possible assignments compatibly with all the constraints.

2.4 Instances of the SCSP Framework

Having laid out the c-semiring-based theory of constraint satisfaction, Bistarelli shows that some of the previous constraint satisfaction approaches can be seen as instances of this theory differing only in the choice of the semiring. Below I list the different CSPs and the semirings used in them as discussed by Bistarelli.

- **Classical CSPs:** A classical CSP is just a set of variables and constraints, where each constraint specifies the tuples that are allowed for the involved variables. Since the constraints in a CSP are crisp, they can be modeled with a semiring containing only 0 and 1 in A. Also we can model constraint combination with logical *and*, and the projection over some of the variables (to obtain the value of the tuples of the variables in the type of the problem) with logical *or*. Thus, a CSP can be seen as just an SCSP with the following c-semiring:

$$\mathrm{S_{CSP}} = \langle \{0, 1\}, \vee, \wedge, 0, 1 \rangle$$

- **Fuzzy CSPs:** Fuzzy CSPs allow for non-crisp constraints, which associate a preference level with each tuple of values. This level of preference is always between 0 and 1. The solution to a fuzzy CSP is defined as the set of tuples of values for all the variables which have the maximal value. Fuzzy CSPs can be modeled in the SCSP framework by choosing the following c-semiring:

$$\mathrm{S_{FCSP}} = \langle \{x | x \in [0, 1]\}, \max, \min, 0, 1 \rangle$$

- **Probabilistic CSPs:** In probabilistic CSPs, each constraint c has an associated probability $p(c)$. Saying that c has probability p, means that the situation corresponding to c has probability p of occurring in the real-life problem. The c-semiring corresponding to the probabilistic CSPs is as follows:

$$\mathrm{S}_{\mathrm{prob}} = \langle \{x | x \in [0,1]\}, \max, \times, 0, 1 \rangle$$

- **Weighted CSPs:** Contrary to fuzzy CSPs whose constraints come with preferences, in weighted CSPs, constraints have associated costs. The solution to a problem in such models is the one with minimum cost (e.g., time, space, number of resources, etc.). Therefore, the associated c-semiring for a weighted CSP is the following:

$$\mathrm{S}_{\mathrm{WCSP}} = \langle \mathbb{R}^*, \min, +, +\infty, 0 \rangle$$

- **Set-Based CSPs:** The SCSP framework gives rise to an interesting class of its instances that are based on set operations such as union and intersection. The corresponding c-semiring for this class of CSPs is this:

$$\mathrm{S}_{\mathrm{set}} = \langle \wp(A), \bigcup, \bigcap, \emptyset, A \rangle$$

This section presented a brief overview of Bistarelli's c-semiring-based generalized theory of soft constraint satisfaction systems. As Bistarelli shows, many previous CLP approaches to soft constraints are in fact instances of this generalized framework, which is parametric with respect to the semiring used. In the next section, we will show that an instance of this theory, the *weighted soft constraint satisfaction* approach is suitable for modeling linguistic constraints.

3 Soft Linguistic Constraints

This section outlines a theory of linguistic soft constraint satisfaction based on the SCSP framework (the c-semiring-based theory of Soft Constraint Satisfaction Problems).

Section 3.1 briefly talks about how a c-semiring-based approach might be incorporated into a unification-based theory of grammar. We can think of a grammar in SCSP terms as a constraint system, $CS = \langle S, D, V \rangle$, where $S = \langle A, \oplus, \otimes, 0, 1 \rangle$ is a semiring, and V is a set of variables characterizing the candidate. D is a finite set of values that the variables in V can take. Therefore, a constraint over this CS is a tuple $\langle def, con \rangle$ such that $con \subseteq V$ and $def : D^k \rightarrow A$. Values in the carrier set A correspond to the overall compliance of a candidate linguistic structure, can, with the whole constraint system.

The function def in SCSP, takes the vector representation D^k of the candidate and maps it to a global valuation. Figure 2 shows how Can is mapped to A. Embedding applies to $can \in Can$ returning a vector of features, which is passed to

$\bar{c}_i$, for $1 \leq i \leq m$ perhaps, which returns a vector of valuations. These valuations are then combined (in this case, weighted and summed up) by the global valuation function g returning a value in A.

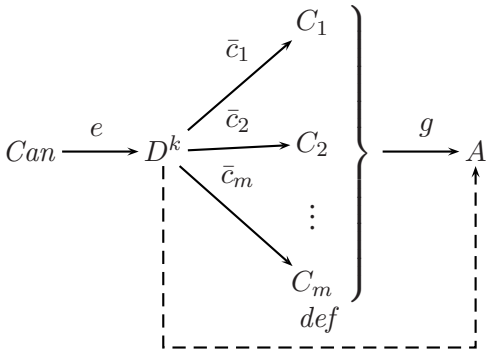

Figure 2: Global valuation calculation of candidate structures

The optimal candidate, to use optimality theoretic terminology, will be the one that has the smallest value for its global valuation; that is, the candidate with a global valuation closest to zero is optimal and the ones with larger values are increasingly suboptimal (in other words, $\mathcal{V}(can)$ represents a cost). In this context, the semiring used will be the one shown below in (1), where $\mathbb{R}^*$ is the set of non-negative real numbers, min chooses the solution, and $+$ combines the values in the carrier set A, (which in this semiring is $\mathbb{R}^*$). Absolute consistency is denoted by 0 and absolute inconsistency by $+\infty$.

(1) $$S_{\mathrm{SCSP}} = \langle \mathbb{R}^*, \min, +, +\infty, 0 \rangle$$

A semiring with $\min$ as the additive operation and $+$ as the multiplication operation with $+\infty$ and 0 corresponding to 0 and 1, respectively, is also known as the *tropical semiring*. S_{V} is a c-semiring since the additive operation, $\min$, is idempotent (i.e., $\min(a, a) = a$ for all a), and the multiplicative operation, $+$, is commutative. Also, $+\infty$ is the identity element for $\min$ (i.e., $\min(a, +\infty) = a$ for all a), and 0 is the identity element for $+$, (i.e., $a + 0 = a$ for all a). $\mathbb{Z}^*$ is the set of non-negative integers (including 0). The associated ordering $\leq_s$ corresponds to $\geq$ over non-negative integers, which means that smaller numbers correspond to better candidates.

A linguistic constraint system is then defined as follows:

DEFINITION 3.1 *A linguistic constraint system based on SCSP, $CS = \langle S_{\mathrm{SCSP}}, D, V \rangle$, will then have the following components:*

- **C-Semiring:** $S_{\mathrm{SCSP}} = \langle \mathbb{R}^*, min, +, +\infty, 0 \rangle$.
- **Variables:** *An ordered set V representing the candidate structure.*

- **Domain:** *D a finite set of values that members of V can take.*

DEFINITION 3.2 **Connection:** *$con \subseteq V$, is called the type of the constraint.*

The set con tells us which variables are involved in each constraint.

DEFINITION 3.3 **Domain Function:** $dom : V \rightarrow \mathsf{D}$, *where* $\mathsf{D} \subseteq D$.

The domain function dom tells us which members of D can be assigned to each member of V.

DEFINITION 3.4 **Embedding:** *$e : Can \hookrightarrow D^k$ is a bijective function that maps the set of linguistic structures onto vector representations; in particular, for all $\vec{d} \in D^k$, $d_i \in dom(v_i)$, where $1 \leq i \leq k$.*

Embedding ensures a representation of linguistic structures that is suitable for the constraint solver. The embedding function must be bijective because once the solver returns a vector as the solution, we want to be able to identify a candidate with that vector.

Bistarelli (2001) also defines a function def as follows:

DEFINITION 3.5 **Definition:** $def_k : D^k \rightarrow \mathbb{R}^*$

This function is actually the composition of $\bar{c}_i$ with g (see Figure 2 above and definition 3.9 below), that is, $def = g \circ \bar{c}_i$. Based on this definition, a constraint in SCSP is defined as follows:

DEFINITION 3.6 **Constraint:** *$\langle \bar{c}_i, con^n \rangle$, for some $1 \leq n \leq k$ where n is the cardinality of con.*

DEFINITION 3.7 **Constraint Weight:** *w_i is a numerical weight associated with each constraint $\langle \bar{c}_i, con_i^n \rangle$. In OT literature, this is known as the* rank *of the constraint.*

DEFINITION 3.8 **Valuation:** *$\bar{c}_i : D^k \rightarrow C_i$, a function that evaluates the value d of each variable v.*

DEFINITION 3.9 **Global Valuation:** *$g : C_1 \times C_2 \times \ldots \times C_m \rightarrow \mathbb{R}^*$ is the combination function that calculates the global valuation of $\vec{d}$ based on a vector of valuations returned by all of the $\bar{c}_i$. In this model, $g(c_1, c_2, \ldots, c_m) = \Sigma_{i=1}^m w_i c_i$ for all $c_i \in C_i$.*

3.1 Toward Graded Unification-Based Grammars

In section 1, we talked about the benefits of a parallel modular grammar architecture saying that such an architecture leads to simpler modules and captures generalizations better. One important advantage of implementing such an architecture in a unification-based framework is that unification naturally allows for the modules to constrain one another. Through structure-sharing, even though the modules may not care about nor see the details inside other modules, they cannot generate structures that are unacceptable to other modules. It would then be natural to try to implement the proposed soft-constraint satisfaction system in a unification-based framework such as HPSG. In order to do this, we need to change how type antecedent constraints are enforced without modifying the unification mechanism. Standard HPSG type antecedent constraints are crisp; their violation causes the generated structure to be rejected. Multiple constraints on a type are explicitly connected by logical AND, which also means the violation of any one constraint results in the rejection of the generated structure. This constraint system roughly corresponds to Bistarelli's $\mathrm{S}_{\mathrm{CSP}}$ mentioned in section 2.4.

Malouf (2003) argues that the fact that an analysis naturally falls out of OT's notion of ranked violable constraints does not necessarily mean that it *has* to be analyzed that way. He states that OT suffers from a "procedural metaphor;" that is, the theory relies on some cognitively unreal and intractable operations to account for acceptable structures. The most notable part of this metaphor is the generate-and-test procedure of the theory where a partial representation (such as a logical form) is fed to a component called *Gen* that generates a potentially infinite number of candidate output structures to be evaluated against a set of constraints by *HEval*. This is a common concern. As a solution, Malouf discards the procedural metaphor along with the violability of the constraints, and accounts for his data using an HPSG type hierarchy. His analysis, albeit elegant, does not leave any room for graded grammaticality judgments, accounting for multiple violations of the same constraint, and ganging up effects, not to mention the graded constraints discussed by Haji-Abdolhosseini (2005). This section shows that we can incorporate soft constraints within constraint-based grammars such as HPSG without resorting to any procedural metaphors.

In order to account for violable constraints as well as degrees of constraint violation, multiple constraint violations, and ganging up effects discussed in Keller (2000), we can use the weighted CSP paradigm defined in the previous section ($\mathrm{S}_{\mathrm{WCSP}} = \langle \mathbb{R}^*, min, +, +\infty, 0 \rangle$).

We now go over some illustrative examples. It should be mentioned that the goal of the following examples is not to derive the "correct" analysis but to show how a system of type antecedent constraints based on the tropical semiring would calculate costs for different analyses.

In the case of sentences (2) and (3), we can formulate a constraint on head-complement phrases (*hd-comp-ph*) as in Figure 3. For simplicity of exposition, this constraint only employs two non-head daughters. The extension of the constraint

to accommodate more daughters is straightforward.

(2) a. He wanted to demonstrate it to us.
b. He wanted to demonstrate that life to us.
c. He wanted to demonstrate the consequences to us.
d. ? He wanted to demonstrate the consequences of such an unholy life to us.
e. ?? He wanted to demonstrate the consequences of such a horribly filthy and unholy life to us.

(3) a. * He wanted to demonstrate to us it.
b. ?? He wanted to demonstrate to us that life.
c. He wanted to demonstrate to us the consequences of such an unholy life.

$$\textit{hd-comp-ph} \Rightarrow \begin{bmatrix} \text{PHON} & \langle \boxed{1}, \boxed{3} \rangle \\ \text{NON-HD-DTRS} & \left\langle \begin{bmatrix} \text{PHON} & \boxed{1} \end{bmatrix}, \begin{bmatrix} \text{PHON} & \boxed{2} \end{bmatrix} \right\rangle \end{bmatrix} \wedge length\left(\boxed{1}\right) \leq length\left(\boxed{2}\right)$$

Figure 3: An HPSG formulation of the LH constraint on verb complements

The valuation function for the LH constraint as formulated above can be calculated according to the following function:

(4) **Valuation Function for LH:**

Given the description $\begin{bmatrix} \textit{hd-comp-ph} & \\ \text{PHON} & \boxed{3} \oplus \langle \boxed{1}, \boxed{2} \rangle \\ \text{NON-HD-DTRS} & \left\langle \begin{bmatrix} \text{PHON} & \boxed{1} \end{bmatrix}, \begin{bmatrix} \text{PHON} & \boxed{2} \end{bmatrix} \right\rangle \end{bmatrix}$,

$$val(\text{LH}) = \frac{length(\boxed{1})}{length(\boxed{1}) + length(\boxed{2})}$$

Let us assume, for now, that $length(x)$ is the number of words in x, and that the weight of the constraint LH is 1. The formula in (4) returns a number between 0 and 1. If the two complements are of equal size, the valuation returned will be 0.5; the valuation approaches 1 as the first complement gets longer than the second, and it approaches 0 as the second complement gets longer than the first. Of course, one can think of many ways to formulate LH. The definition presented here is just one of them. This formulation is flexible because it reflects the magnitude of the difference between the two complements. We can now see how the sentences in (2) are evaluated in terms of LH. The valuations of LH calculated for (2a–e) are shown in (5) below.

(5) a. For (2a): $val(\text{LH}) = \frac{1}{1+2} \approx .33$
b. For (2b): $val(\text{LH}) = \frac{2}{2+2} = .5$
c. For (2c): $val(\text{LH}) = \frac{2}{2+2} = .5$
d. For (2d): $val(\text{LH}) = \frac{7}{7+2} \approx .78$
e. For (2e): $val(\text{LH}) = \frac{10}{10+2} \approx .83$

As can be seen, this accounts for the declining acceptability of the examples in (2).

The examples in (3) demonstrate the interaction of two constraints: (i) LH, and (ii) the constraint that requires verbal complements to appear in descending order of obliqueness (call it COMPORD). If obliqueness is a total order defined over verbal complements represented with $>_o$, then we can formulate COMPORD as in Figure 4. The valuation function for COMPORD is defined in (6).

$$\textit{hd-comp-ph} \Rightarrow \left[\text{NON-HD-DTRS} \left\langle \boxed{1},\boxed{2} \right\rangle\right] \wedge \boxed{1} >_o \boxed{2}$$

Figure 4: An HPSG formulation of COMPORD

(6) **Valuation Function for** COMPORD**:**

Given the description $\begin{bmatrix} \textit{hd-comp-ph} \\ \text{NON-HD-DTRS} \quad \left\langle \boxed{1},\boxed{2} \right\rangle \end{bmatrix}$,

$$val(\text{COMPORD}) = \begin{cases} 0 & \text{iff } \boxed{1} >_o \boxed{2} \\ 1 & \text{otherwise} \end{cases}$$

$val(\text{COMPORD})$ is defined as a *characteristic* or *selector* function returning either 0 or 1, but notice that since we are using the tropical semiring these values do not have their traditional *true* or *false* meaning. In this constraint system, 0 corresponds to no violation and 1 corresponds a violation of degree 1. Also notice, since we are adding costs, multiple instances of such constraint violations will incrementally increase global evaluation as it does in Linear Optimality Theory (Keller, 2000). Let us assume that the two constraints LH and COMPORD have equal weights. Then the valuation of the sentences in (3a–c) with respect to LH and COMPORD is calculated as in (7).

(7) a. For (3a): $val(\text{LH}) + val(\text{COMORD}) \approx .66 + 1 = 1.66$
b. For (3b): $val(\text{LH}) + val(\text{COMORD}) = .50 + 1 = 1.50$
c. For (3c): $val(\text{LH}) + val(\text{COMORD}) \approx .22 + 1 = 1.22$

This analysis quantitatively captures the increasing acceptability of the sentences in (3) as the sentence-final direct object gets longer than the indirect object.

An interesting outcome of this analysis is that it naturally captures speakers' intuitions about the relative acceptability of forms like the ones in (8) without the need to posit an arbitrary constraint prohibiting ending a dative construction with

$$\textit{clause} \Rightarrow \begin{bmatrix} \text{DOM} & \boxed{1} \oplus \boxed{2} \\ \text{INFO} & \left\langle \begin{bmatrix} \textit{theme} \\ \text{I-DOM} \quad \boxed{1} \end{bmatrix}, \begin{bmatrix} \textit{rheme} \\ \text{I-DOM} \quad \boxed{2} \end{bmatrix} \right\rangle \end{bmatrix}$$

Figure 5: An HPSG formulation of THRH

a pronoun (which would completely rule out (8b), incorrectly). According to this analysis, we not only capture the graded grammaticality of each example, we can also show how much each example is worse than the other.[4]

(8) a. Give it to me.
$val(\text{LH}) + val(\text{COMPORD}) \approx .33 + 0 = .33$

b. ?? Give me it.
$val(\text{LH}) + val(\text{COMPORD}) = .5 + 1 = 1.5$

c. * Give to me it.
$val(\text{LH}) + val(\text{COMPORD}) \approx .66 + 1 = 1.66$

To incorporate more modules, let us now consider how information structure can be integrated into this model. Let THRH stand for the violable constraint that requires the theme to appear before the rheme. A version of this constraint for just one theme and one rheme is shown in Figure 5;[5] an extension to the constraint for multiple themes and rhemes is also straightforward. The valuation function for THRH is formulated in (9).

(9) **Valuation Function for** THRH**:**

Given the description $\begin{bmatrix} \textit{clause} \\ \text{INFO} \quad \left\langle \boxed{1}, \boxed{2} \right\rangle \end{bmatrix}$,

$$val(\text{THRH}) = \begin{cases} 0 & \text{iff } type(\boxed{1}) = \textit{theme} \wedge type(\boxed{2}) = \textit{rheme} \\ 1 & \text{otherwise} \end{cases}$$

Let us assume that the preferred response to the question "What did John give to the man?" is (10a) as opposed to (10b).[6]

(10) a. [He gave]$_{\text{theme}_1}$ [money]$_{\text{rheme}}$ [to the man]$_{\text{theme}_2}$.

b. [He gave the man]$_{\text{theme}}$ [money]$_{\text{rheme}}$.

[4] Note that we are assuming equal weights for these constraints. Estimating the exact weights of the constraints requires having access to training data obtained through corpus analysis or experimental work. I shall leave this for future research.

[5] I am following the analysis in Haji-Abdolhosseini (2003, 2005) where *sign* has the features DOM and INFO taking a list of lexical items and *info* objects, respectively. The types *theme* and *rheme* are subtypes of *info*.

[6] Again note that we are not making any strong claims as to what sentence is actually the preferred response. This should be determined through separate studies. The point here is to illustrate how valuation calculations work.

Again assuming equal weights for LH, COMPORD, and THRH, we can calculate the global valuations of these sentences with respect to these three constraints as in (11). It can be seen that (10a) gets a lower global valuation (i.e., is preferred by the model).

(11) a. For (10a): $val(\text{LH}) + val(\text{COMPORD}) + val(\text{THRH}) = .25 + 0 + 1 = 1.25$

b. For (10b): $val(\text{LH}) + val(\text{COMPORD}) + val(\text{THRH}) \approx .66 + 1 + 0 = 1.66$

In this example, THRH has been violated in favor of COMPORD and LH. Note that since the difference in the lengths of the two verb complements is small the two sentences show a small difference in their global valuation (0.41).

Let us look at another example in which the difference in the lengths of the verb complements is larger.

(12) a. [He gave]$_{\text{theme}_1}$ [a lot of his hard earned money]$_{\text{rheme}}$ [to the man]$_{\text{theme}_2}$.

b. [He gave the man]$_{\text{theme}}$ [a lot of his hard earned money]$_{\text{rheme}}$.

The global valuations of these sentences are given below:

(13) a. For (12a): $val(\text{LH}) + val(\text{COMPORD}) + val(\text{THRH}) = .70 + 0 + 1 = 1.70$

b. For (12b): $val(\text{LH}) + val(\text{COMPORD}) + val(\text{THRH}) \approx .22 + 1 + 0 = 1.22$

Here we see that (12b) is preferred. We also see that the difference between the global valuations of (12a) and (12b) is larger than before (0.48), which means that in this example the alternative is costlier than in the previous example. In other words, the gradient characterization of LH captures the fact that larger differences in the lengths of the complements result in higher degrees of constraint violation if the heavier constituent appears before the lighter one, an observation made by Arnold et al. (2000) as well as Haji-Abdolhosseini (2005). In addition, the c-semiring-based implementation of type antecedent constraints in HPSG allows for capturing the ganging up effects of constraint violation as well as multiple violations of the same constraint (since valuations are summed up).

The cost of a feature structure, f, of type τ is the weighted sum of the valuations of all the constraints imposed on τ with respect to f plus the sum of the costs of all the feature values of f. This is formalized in (14).

(14) $cost(f_\tau) = \sum_i w_i \cdot val(c_i^\tau) + \sum_j cost(g_j)$
where f_τ is the feature structure of type τ to which we want to assign a cost; c_i^τ is a constraint on the type τ; $val(c_i^\tau)$ is the valuation of c_i^τ; and g_j is a feature value of f.

The formula in (14) implies that the cost of a feature structure is never less than the sum of the costs of its substructures (provided that there are no negative

weights, which is what we have been assuming). If there are any cases where the same description gets different valuations in different contexts (i.e., in different feature structures), then we can replace our original constraint with other more specific constraints.

The reason for this desideratum is twofold: (i) We want to make sure that every part of the feature structure is contributing information about constraint violations in substructures; and (ii) we want the constraints to be local; that is, every constraint has to be sensitive only to the description on its consequent and should not be affected by the context in which it is applied. For instance, consider the following feature structures:

(15) a. $\begin{bmatrix} t & \\ \text{F} & a \end{bmatrix}$

b. $\begin{bmatrix} u & \\ \text{G} & a \end{bmatrix}$

If the cost of a feature structure f of type a depends on whether f is the value of F or G, then we cannot formulate a single constraint on type a. This is because such a constraint on a would be non-local as it would have to have information about whether f occurs in a feature structure of type t or u. In order to keep our constraint local in this case, we must formulate two constraints on t and u making reference to the value of the F and G features, respectively.

4 Conclusion

This paper incorporated a generalized c-semiring-based theory of soft constraint satisfaction within a constraint-based grammar architecture. The stipulation that soft constraints apply at interfaces while hard constraints apply inside modules is advantageous from a grammar engineering point of view. Linguistic modules can be developed separately and then put together allowing soft constraints to resolve conflicts among modules.

This work can be pursued in many different directions. One obvious way to follow up this work would be to work out the formal details of incorporating SCSP in a theory like HPSG. Another way to pursue would be to implement an SCSP-based constraint solver in a logic-programming language like ALE (Carpenter and Penn, 1999). One can also investigate different learning algorithms for an SCSP-based grammar.

References

Abney, Steven. 1996. Statistical Methods and Linguistics. In Judith Klavans and Philip Resnik (eds.), *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, Cambridge, MA: The MIT Press.

Abney, Steven. 1997. Stochastic Attribute-Value Grammars. *Computational Linguistics* 23(4), 597–618.

Arnold, Jeniffer E., Wasow, Thomas, Losongco, Anthony and Ginstrom, Ryan. 2000. Heaviness vs. Newness: The effects of structural complexity and discourse status on constituent ordering. *Language* 76(1), 28–55.

Bistarelli, Stefano. 2001. *Soft Constraint Solving and Programming: A General Framework*. Ph.D. thesis, Università di Pisa.

Bod, Rens. 1998. *Beyond Grammar: An Experience-Based Theory of Language*. CSLI Publications, Cambridge University Press.

Bod, Rens, Hay, Jennifer and Jannedy, Stefanie (eds.). 2003a. *Probabilistic Linguistics*, Cambridge, MA, MIT Press.

Bod, Rens, Scha, Remko and Sima'an, Khalil (eds.). 2003b. *Data-Oriented Parsing*. CSLI Publications.

Borning, Alan, Freeman-Benson, Bjorn and Wilson, Molly. 1992. Constraint Hierarchies. *Lisp and Symbolic Computation* 5(3), 223–270.

Bresnan, Joan (ed.). 1982. *The Mental Representation of Grammatical Relations*. Cambridge, MA: The MIT Press.

Bresnan, Joan. 2001. *Lexical-Functional Syntax*. Blackwell Textbooks in Linguistics, Malden, MA: Blackwell.

Carpenter, Bob and Penn, Gerald. 1999. ALE The Attribute Logic Engine: User's Guide, available online at `www.cs.toronto.edu/~gpenn/ale/files/aleguide.ps.gz`.

Davey, Brian A. and Priestley, Hilary A. 1990. *Introduction to Lattices and Order*. Cambridge Mathematical Textbooks, Cambridge: Cambridge University Press.

Dubois, Didier, Fargier, Hélèn and Prade, Henri. 1993. The Calculus of Fuzzy Restrictions as a Basis for Flexible Constraint Satisfaction. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 1131–1136, IEEE.

Fargier, Hélèn and Lang, Jérôme. 1993. Uncertainty in Constraint Satisfaction Problems: A Probabilistic Approach. In *Proceedings of the European Conference on Symbolic and Qualitative Approaches to Reasoning and Uncertainty (ECSQARU), number 747 in LNCS*, pages 97–104, Springer-Verlag.

Foth, Kilian, Menzel, Wolfgang and Schröder, Ingo. 2005. Robust Parsing with Weighted Constraints. *Natural Language Engineering* 11(1), 1–25.

Freuder, Eugene C. and Wallace, Richard J. 1992. Partial Constraint Satisfaction. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, IJCAI-89*, volume 58, Detroit, MI.

Haji-Abdolhosseini, Mohammad. 2003. A Constraint-Based Approach to Information Structure and Prosody Correspondence. In Stefan Müller (ed.), *Proceedings of The HPSG-2003 Conference*, pages 143–162, `http://cslipublications.stanford.edu/HPSG/4/`.

Haji-Abdolhosseini, Mohammad. 2005. *Modularity and Soft Constraints: A Study of Conflict Resolution in Grammar*. Ph.D. Thesis, University of Toronto.

Jackendoff, Ray. 1992. *Languages of the Mind: Essays on Mental Representation*. The MIT Press.

Jackendoff, Ray. 1997. *The Architecture of the Language Faculty*. Linguistic Inquiry: Monograph Twenty-Eight, Cambridge, Mass.: The MIT Press.

Jackendoff, Ray. 2002. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. New York, NY: Oxford.

Keller, Frank. 2000. *Gradience in Grammar: Experimental and Computational Aspects of Degrees of Grammaticality*. Ph.D. thesis, University of Edinburgh.

Malouf, Robert P. 2003. Cooperating Constructions. In E. Francis and L. Michaelis (eds.), *Mismatch: Form-function Incongruity and the Architecture of Grammar*, pages 403–424, Stanford: CSLI Publications.

Marriott, Kim and Stuckey, Peter J. 1998. *Programming with Constraints: An Introduction*. Cambridge, MA: The MIT Press.

Newell, Allan. 1990. *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Oatley, Keith. 1992. *Best Laid Schemes: The Psychology of Emotions*. Cambridge: Cambridge.

Oatley, Keith and Johnson-Laird, Philip N. 1987. Towards a Cognitive Theory of Emotions. *Cognition and Emotions* 1, 29–50.

Pollard, Carl and Sag, Ivan A. 1987. *Information-Based Syntax and Semantics, Volume I: Fundamentals*. CSLI Lecture Notes, No. 13, Stanford: CSLI Publications.

Pollard, Carl and Sag, Ivan A. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics, Chicago: CSLI Publications.

Prince, Alan and Smolensky, Paul. 1993. Optimality Theory: Constraint Interaction in Generative Grammar. Technical Report 2, Rutgers University Center for Cognitive Science, Piscataway, NJ.

Rosenfeld, Azriel, Hummel, Robert A. and Zucker, Steven W. 1976. Scene Labelling by Relaxation Operations. *IEEE Transactions on Systems, Man, and Cybernetics* 6(6), 420–433.

Ruttkay, Zsofi. 1994. Fuzzy Constraint Satisfaction. In *Proceedings of the 3rd IEEE International Conference on Fuzzy Systems*, pages 1263–1268.

Schiex, Thomas, Fargier, Hélèn and Verfaille, Gérard. 1995. Valued Constraint Satisfaction Problems: Hard and Easy Problems. In *Proceedings of IJCAI95*, pages 631–637, Morgan Kaufman.

Schröder, Ingo. 2002. *Natural Language Parsing with Graded Constraints*. Ph. D.thesis, Fachbereich Informatik der Universität Hamburg.