Grammar Induction as Automated Transformation between Constraint Solving Models of Language *

Ife Adebara and Veronica Dahl
Computing Sciences Department, Simon Fraser University
iadebara@sfu.ca,veronica@cs.sfu.ca

Abstract

We specialize an efficient while linguistically savvy constraint solving model of grammar induction-Womb Grammars (WG) -, into an interesting specific application: that of inferring the grammar of an under-resourced language -Yorùbá- from that of English, through grammatical model transformation. The model represents both the known grammar and the grammar to be inferred in terms of constraints, or properties, between pairs of constituents. This allows us to view parsing as constraint solving, and to use the parser's output (namely, the information of which constraints failed and which were satisfied) as a guideline on how to transform the known grammar model of English into the (unknown to the system) grammar model of Yorùbá. Interesting extensions to the original Womb Grammar model are presented, motivated from the specific needs of Yorùbá and similar tonality infused languages. Our methodology is implemented in Constraint Handling Rule Grammars (CHRG) and has been used so far for inducing the grammar model of a useful subset of Yorùbá noun phrases.

1 Introduction

Grammar induction has met with reasonable success using different models of grammar: a) as a parametrized, generative process explaining the data [Fernando C. N. Pereira and Yves Schabes, 1992; Dan Klein and Christopher D. Manning, 2004], b) as a probability model, so that learning a grammar amounts to selecting a model from a pre-specified model family [Eugene Charniak and Mark Johnson, 2005; CMengqiu Wang and Noah A. Smith and Teruko Mitamura, 2007; Shay B. Cohen and Noah A. Smith, 2010], and c) as a Bayesian model of machine learning [William P. Headden and Mark Johnson and David McClosky, 2009].

Using linguistic information from one language for the task of describing another language has also yielded good results, albeit for specific tasks—such as disambiguating the other language [David Burkett and Dan Klein, 2008], or fixing morphological or syntactic differences by modifying tree-based rules [Lionel Nicolas and Miguel A. Molinero and Benoît Sagot and Elena Sánchez Trigo and de La Clergerie, Éric and and Jacques Farré and Joan Miquel Vergés, 2009]—rather than for syntax induction.

This usually requires parallel corpora, an interesting exception being [Shay B. Cohen and Noah A. Smith, 2010], where information from the models of two languages is shared to train parsers for two languages at a time, jointly. This is accomplished by tying grammar weights in the two hidden grammars, and is useful for learning dependency structure in an unsupervised empirical Bayesian framework.

Our novel approach, in contrast, works for more than just specific tasks such as disambiguation, and needs neither a pre-specified model family, nor parallel corpora, nor any of the typical models of machine learning. It proceeds instead through automatically transforming a given (task-independent) grammar description of a source language, from just the lexicon of the target language plus a representative input set of correct phrases in the target language.

Both descriptions (the syntax of the source and of the target language subset addressed) are stated in terms of linguistic constraints (also called "properties" in the linguistic literature) between pairs of constituents, although for the target language constraints we depart from the classic formalism [Blache, 2005] in view of Yorùbá motivated extensions. This choice allows us a great degree of modularity, in which constraints can be checked efficiently through constraint solving.

From the perspective of problem solving, our modelling framework thus compares favourably with others: it combines linguistic formality with efficient problem solving, and can transfer into other languages, in particular languages that use tones to differentiate meaning.

2 Motivation

Close to seven thousand languages are currently spoken in the world, the majority of which are understudied. Linguists cannot keep up with their study even for educational purposes, and there is a growing need for their automatic processing as well, since the amount of text sources grows much faster than humans can process them. To make matters worse, most linguistic resources are poured into English and a handful of

*This research was supported by NSERC Discovery grant 31611024

other first world languages, leaving the vast majority of languages and dialects under-explored. Clearly, automating the discovery of an arbitrary language's grammar model would render phenomenal service to the study and preservation of linguistic diversity.

Scientifically, we wanted to explore to what extent the parsing-as-constraint-solving paradigm of Natural Language Processing (NLP) problem solving could buy us a great degree of linguistic descriptive formality without sacrificing efficiency, in the realm of grammar induction and in particular for inducing Yorùbá, which is severely under-resourced and which one of the authors has expertise in.

Yorùbá belongs to the Yoruboid group of the Kwa branch of the Niger-Congo language family, which cuts across most of sub-Saharan Africa. It is a tonal dialect-continnum comprising about 20 distinctive dialects and spoken by over 30 million people in the western part of Nigeria [Fagborun, 1994].

3 Background

Among the linguistic theories that lend themselves the most to constraint-based implementation are those that split the information previously packed in one rewriting rule into several constraints or properties. These *constraint based* or *property-based* theories, such as Property Grammars (PG) [Blache, 2005] evolved from Immediate Dominance / Linear Precedence (IDLP), which unfolds a rewrite rule into the two constraints of immediate dominance (expressing which categories are allowable daughters of a phrasal category) and linear precedence (expressing which of the daughters must precede which others).

For example in the PG framework, English noun phrases can be described through a few constraints such as precedence (a determiner must precede a noun, an adjective must precede a noun), uniqueness (there must be at most one determiner), exclusion (an adjective phrase must not coexist with a superlative), obligation (a noun phrase must contain the head noun), and so on. Instead of resulting in either a parse tree or in failure as traditional parsing schemes do, such frameworks characterize a sentence through the list of the constraints a phrase satisfies and the list of constraints it violates, so that even incorrect or incomplete phrases will be parsed. Moreover, it is possible to relax some of the constraints by declaring relaxation conditions in modular fashion. [Dahl and Blache, 2004] encodes the input PG into a set of CHRG rules that directly interpret the grammar in terms of satisfied or relaxed constraints, which are then propagated while a syntactic tree is built as a side effect. Womb Grammars- a recent adaptation of this framework into grammar transformation [Dahl and Miralles, 2012]- automates as well the induction of a language's syntax from that of another, focusing on failed constraints.

In such theories, the modularity obtained by splitting grammatical information apart into constraints leads naturally to more *robust parsers*, since it allows us to clearly identify from the parser's output which constraints are satisfied and which fail, which allows us to accept even incomplete or incorrect sentences, instead of silently failing to parse them. We can also produce some indication of the sentence's degree of acceptability by analyzing the failed properties.

The PG formalism presently comprises the following seven categories (we adopt the handy notation of [Duchier, Dao, and Parmentier, 2013], and the same example):

Constituency $A : S$, children must have categories in the set S

Obligation $A : \triangle B$, at least one B child

Uniqueness $A : B!$, at most one B child

Precedence $A : B \prec C$, B children precede C children

Requirement $A : B \Rightarrow C$, if B is a child, then also C is a child

Exclusion $A : B \nLeftrightarrow C$, B and C children are mutually exclusive

Dependency $A : B \sim C$, the features of B and C are the same

Example 1. *For example, if we denote determiners by D, nouns by N, personal nouns by PN, verbs by V, noun phrases by NP, and verb phrases by VP , the context free rules* $NP \rightarrow D\ N$ *and* $NP \rightarrow N$*, which determine what a noun phrase is, can be translated into the following equivalent constraints:* $NP : \{D, N\}$, $NP : D!$, $NP : \triangle N$, $NP : N!$, $NP : D \prec N$, $D : \{\}$, $N : \{\}$.

4 The Intuitive Idea

By giving up syntactic trees as a focus and focusing instead on *grammar constraints*, we can arrive at more modular, more flexible and less costly models. To see this, consider that if we were to work with tree-oriented rules such as:

```
noun_phrase --> determiner, noun.
```

such rules would fail for languages such as Yorùbá, where, for instance, nouns precede determiners, and more than one determiner can accompany a noun. For instance, *ilé yìí nàà wó* (house this the collapse) means some particular house in close proximity to the speaker collapsed, whereas *ilé kan nàà wó* (house a the collapse) means a house the speaker may not be familiar with collapsed.

Transforming a tree-oriented model into a language model for Yorùbá would require changing every rule where these two constituents are involved, as well as creating a grammar symbol that covers sequences of determiners rather than just a single one.

In a constraint-based model, in contrast, the needed transformation can be expressed in terms of separate constraints: we need to replace the English precedence constraint with its converse (in a noun phrase, a noun must precede a determiner) [1] , and delete the constraint that in a noun phrase, determiners must be unique. These modifications carry over to the entire grammar without further ado.

This intuition leads directly to that of our WG transformation methodology: first we implement an executable version (i.e., a parser) of our constraint-based grammar model of English, then instead of running English phrases by it, we feed it a *Yorùbá* input phrase. Given that we have a Yorùbá lexicon,

[1] In fact a subtler algorithm is needed- this will be covered later.

this parses into a pre-lexical string (e.g. "noun, adjective, determiner"). Obviously, many of the English constraints (in particular, for example, the English precedence constraints) will fail to hold for it. Examining such failures with respect to a representative set of input phrases in Yorùbá, our system will be able to determine whether for Yorùbá we must e.g. induce the converse of an English precedence constraint (if it were the case that the converse ordering is found in every single Yorùbá phrase), or simply delete the English constraint (if it were the case that both orderings are allowed), or make it conditional to some linguistic feature being there (if in some cases one ordering routinely held, whereas in other cases it didn't), etc.

5 Our methodology

5.1 Constraint modification- The WG Paradigm

Just as its ancestor, Property Grammar, the WG paradigm describes a language's phrases in terms of constraints between pairs of direct daughters of a phrasal category. The classic constraints are constituency (which constituents are allowable as immediate daughters of a phrasal category), precedence (in a given phrase, a certain daughter must precede a certain other daughter), obligation (a certain daughter must occur in a given phrase), uniqueness (if a daughter can occur only once in a given phrase), requirement (a certain daughter is required in a given phrase), dependency (same features must be shared between two daughters of a given phrase), and exclusion (if a certain daughter occurs in a given phrase, another certain daughter may not co-occur).

WG extends the parsing capabilities implicit in these properties into a model of grammatical induction, in addition to parsing. In their first incarnation [Dahl and Miralles, 2012] they allowed greater efficiency by focusing on failed constraints only, rather than explicitly calculating all successful and unsuccessful constraints. However in adapting the model into Yorùbá , we have found it more efficient to explicitly calculate satisfied constraints as well, as we go along, This is partly because in order to arrive at as nuanced a description as needed for Yorùbá , we had to extend the classical Property Grammar model to accommodate what we call conditional constraints: those whose satisfaction depends on the run-time values of combinations of features (more on this later,)

The general WG model can be described as follows: Let L^S be the source language. Its syntactic component will be noted L^S_{syntax}. Likewise, we call the target language L^T, its lexicon (L^T_{lex}) and its syntax L^T_{syntax}. If we can get hold of a sufficiently representative set of phrases in L^T that are known to be correct (a set where our desired subset of language is represented), we can feed these to a hybrid parser consisting of L^S_{syntax} and L^T_{lex}. This will result in some of the sentences being marked as incorrect by the parser. An analysis of the constraints these "incorrect" sentences violate can subsequently reveal how to transform L^S_{syntax} so it accepts as correct the sentences in the corpus of L_T—i.e., how to transform it into L^T_{syntax} by modifying the constraints that were violated into constraints that accept the input. Figures 1 and 2 respectively show the problem and our solution in schematic form.

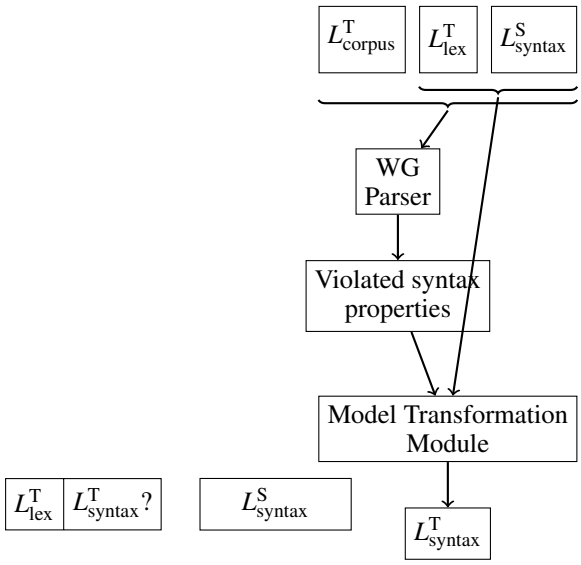

Figure 1: The Problem Figure 2: The Solution

An Example

Let $L^S = English$ and L^T = Yorùbá, and let us assume that English determiers always precede the noun they modify, while in Yorùbá they always post-cede it (an oversimplification, just for illustration purposes). Thus "a red book" is correct English, whereas in Yorùbá we would more readily say "iwe pupa kan" (book, red, a).

If we plug the Yorùbá lexicon and the English syntax constraints into our WG parser, and run a representative corpus of (correct) Yorùbá noun phrases by the resulting hybrid parser, the said precedence property will be declared unsatisfied when hitting phrases such as "ìwé pupa kan". The model transformation module can then look at the entire list of unsatisfied constraints, and produce the missing syntactic component of L^T's parser by modifying the constraints in L^S_{syntax} so that none are violated by the corpus sentences.

Some of the necessary modifications are easy to identify and to perform, e.g. for accepting "ìwé pupa kan" we only need to delete the (English) precedence requirement of determiner over noun (noted $det < n$). However, subtler modifications may be in order, after some statistical analysis in a second round of parsing: if in our L^T corpus, which we have assumed representative, *all* determiner appear after the noun they modify, Yorùbá is sure to include the reverse precedence property as in English: $n < det$. So in this case, not only do we need to delete $det < n$, but we also need to add $n < det$.

5.2 Constraint Solving as the implementation means

Constraint Satisfaction has yielded powerful results in many AI areas, including human language processing. To a large extent, this powerful problem-solving paradigm has allowed us to overcome the disjoin that prevailed until relatively recently between languages for problem description and those for problem solving. Since the advent of CHR [Frühwirth, 1998] and of its grammatical counterpart CHRG [Chris-

tiansen, 2005], constraint-based linguistic formalisms can materialize through fairly direct methodologies.

In a nutshell, CHRG rules rewrite constraints into other constraints, subject to possible checks described in a rule's guard and stated as Prolog calls. Their general format is:

$$\alpha \ -\backslash \ \beta \ /- \ \gamma \ ::> \ G \ | \ \delta.$$

This rule is called a propagation rule. The part of the rule preceding the arrow ::> is called the head, *G* the guard, and δ the body; $\alpha, \beta, \gamma, \delta$ are sequences of grammar symbols and constraints so that β contains at least one grammar symbol, and δ contains exactly one grammar symbol which is a non-terminal (and perhaps constraints); α (γ) is called *left (right) context* and β the *core* of the head; *G* is a conjunction of built-in constraints as in Constraint handling rules (CHR) and no variable in *G* can occur in δ. If left or right context is empty, the corresponding marker is left out and if *G* is empty (interpreted as **true**), the vertical bar is left out. The convention from Definite clause grammars (DCG) is adopted that constraints (i.e., non-grammatical stuff) in head and body of a rule are enclosed by curly brackets. Gaps and parallel match are not allowed in rule bodies. A gap in the rule heads is noted "...". Gaps are used to establish references between two long distant elements.

A *simplification (grammar) rule* is similar to a propagation rule except that the arrow is replaced by <:>. A *simpagation (grammar) rule* is similar to a simplification rule, however one or more grammar symbols is prefixed with *!* which means that the grammar symbols should not be removed from the constraint store. Details of the CHRG encoding, how to download the CHRG system and how to use the encoding can be found in the CHRG manual [Christiansen, 2010].

Implementing the WG paradigm through constraint solving allow us to both express and test linguistic constraints in very modular fashion, and results in succinct while executable code.

In our CHRG implementation the appropriate WG constraints are entered in terms of a constraint g/1, whose argument stores each possible grammar property. For instance, our English grammar hybrid parser for noun phrases includes the constraints: g(obligatority(noun, pronoun, proper noun)), g(constituency(determiner)), g(precedence(determiner,adjective)), g(unicity(determiner)), g(requirement(noun, determiner)), g(dependence(determiner,noun)), and so on. These properties are weeded out upon detection of a violation by CHRG rules that look for them, e.g. an input noun phrase where an adjective precedes a noun may provoke deletion of the constraint g(precedence(noun,adjective)) when the following CHRG rule applies:

```
!word(C2,F1,_),...,!word(C1,F2,_),
{g(precedence(C1,C2))}<:>
{fail(precedence(C1,C2), F1, F2)}.
```

Each word is stored in a CHRG symbol word/3, along with its category and features (i.e. word(noun,[neutral, n/a, common, inanimate],aso)). Since the CHRG parse predicate stores and abstracts the position of each word in the sentence, this simpagation rule is triggered when a word of category C2 comes before one of category C1, given the existence of the grammar constraint that C1 must precede C2 [2]. Each of the properties dealt with has similar rules associated with it.

5.3 Main Modifications to the Model resulting from our Application to Yorùbá

Conditional Properties

The original constraint based model, as we have seen, succeeded in detecting and signalling mistakes in the sentence, without blocking the analysis. In this first model, which is parsing-oriented, incorrect sentences could be "accepted" through declaring some constraints as relaxable. For instance, while from the context-free grammar rules shown in Section 3 we wouldn't be able to parse "the the book" (a common mistake from cutting and pasting in word processors), in the constraint-based formulation we can if we relax the uniqueness of determiner constraint.

Relaxation can be made conditional (e. g. a head noun's requirement for a determiner can be made relaxable in case the head noun is generic and in plural form, as in "Lions sleep tonight"). The failure of relaxable constraints is signalled in the output, but does not block the entire sentence's analysis. Implementations not including constraint relaxation capabilities implicitly consider all properties as relaxable. And in fact, when exploiting constraint-based parsing for grammar transformation rather than for parsing, this is exactly what we need, since in an unknown language any constraint may need to be "relaxed" and even of course, corrected.

For that reason, we have considered it more appropriate, in the context of grammar induction, to state "exceptions" as part of a property itself, rather than separately in a relaxation definition. Thus we have created conditional properties, e.g. "precedence(pronoun,noun)" which occurs if a pronoun preceeds a noun, is plural, is a personal pronoun and can be used in place of both animate and inanimate nouns.

We have also used conditional properties to express choice, e.g. whereas before we could state that one element was obligatory in a phrase, we can now state that a property needs to have one of a list of constituents. E.g., the g(obligatority(noun, pronoun, proper noun)) constraint only needs to find one of noun, pronoun or proper noun to succeed. If none of these is found, then a failure occurs which indicates that an obligatory head is absent.

From Failure Driven to Success-and-Failure driven

In view of unprecedented efficiency where property grammar related models were concerned, the original WG model calculated only failed properties, on the assumption that these could be trusted to be a complement of the successful properties, thus obviating the need to calculate both explicitly. However our more in depth analysis in the context of Yorùbá has, as we have seen, uncovered the need for more nuances than simply failing or succeeding, as in the case of conditional properties. We therefore now use a success driven *and*

[2] Recall that in CHRG syntax the symbols prefixed with exclamation points are kept, while the ones without are replaced by the body of the rule, in this case an update constraint that invokes some housekeeping procedures

failure driven approach for inducing the grammar of our target language, Yorùbá. Each input phrase of the target language is tested with all relevant constraints for both failure and success. This makes the model slightly less efficient than if we only were to calculate failed properties, but of course the gain is in accuracy. Efficiency is still guaranteed by the normal CHRG way of operating: rules will only trigger when relevant, e.g. if a phrase is comprised of only a noun and an adjective, it will not be tested with for instance precedence(pronoun, determiner) or any other constraint whose categories are different from those of the input phrase. We keep a list of all properties that fail and another for those that succeed together with the features of the categories of each input phrase and their counts. It is important to state that constituency constraints are tested only for success. This is because we are interested in checking that our target grammar shares similar constituents with our source language and testing for failure will be irrelevant for these constraints.

Model Transformation Module

After the first round of parsing, we do a statistical analysis in a second round of parsing where we determine which constraints should be induced as a property or a conditional property of our target language and those which need to be weeded out. First we determine which constraints should be added to the grammar by finding all constraints that succeed in all relevant phrases. We are correct in doing so since, our input phrase is correct and representative of the target language. We also add the converse of any precedence rule that fails in all relevant phrases so that for instance, if precedence(pronoun, determiner) fails in all relevant phrases, precedence(determiner, pronoun) is induced as a property of our target language. Next, we decide which constraints should be conditional properties. All constraints that fail and succeed are tested in this phase. We find these properties by searching for features which are unique to the failure and or success of these constraints. Any constraint where such features are found are added to the grammar as a conditional property, with its conditions i.e features clearly stated. All other constraints which are not induced by the above rules are not added to the grammar of Yorùbá.

Inducing Constraints not present in the Original Language

We also added an additional function that finds all precedence constraints that are absent in our source but present in our target language. We are equally able to test these constraints as above by checking if such a constraint succeeds in all relevant phrases and if the converse succeeds in any input phrase. If we find that the converse does not succeed in any of the input phrases, this constraint is induced as a property of the target language, else we search for a unique feature as above.

Using Tonality to Induce Conditional Properties

Yorùbá is a tone language, therefore it was imperative to ensure that tones are properly represented. We adopt a full tone marking approach so that each word is marked with their tones in order to avoid ambiguities. The tones are also stored as features, so that for instance the word tútù(cold) that has two syllables with a high and low tone on each syllable respectively is stored as "high, mid" in the features of the word. For instance:

[tútù]::> $word(adjective, [neutral, n/a, descriptive, both]$, tútù).

These tones are used to infer conditional properties in the second phase of parsing, so that if a property is found to occur around words with certain tones, this property will be conditioned on the presence of such tones. It is important to note that the tones are not used in isolation of other features and a property will be said to be conditioned on tones if, tones are the only unique features present. The addition of tones makes our approach extensible to many other languages including tone languages.

In addition to the afore mentioned, the system provides a record of the English translation of every word in the input phrases and this record is consulted during parsing to produce the English equivalent for each word in the input phrase.

5.4 Sample Output

Here is a subset of our induced Yorùbá grammar:

```
-precedence(pronoun,noun):-
pronoun preceeds noun if  the pronoun is
plural, personal and can be used in place of
both animate and inanimate nouns

-precedence(adjective,noun):-
adjective preceeds noun if the adjective is
plural, quantity-uncountable and can be used with
both animate and inanimate nouns

-obligatority((noun;proper-noun;pronoun)):-
noun;proper-noun;pronoun are obligatority.
It succeeds in all 46 input phrases.
noun succeeds in 37 relevant phrases;
pronoun succeeds in 5 relevant phrases;
proper-noun succeeds in 4 relevant phrases;

-precedence(pronoun,adjective):-
pronoun preceeds adjective in all 4 relevant
phrases

-precedence(quantifier,adjective):-
quantifier preceeds adjective in all 4 relevant
phrases

-precedence(quantifier,pronoun):-
quantifier preceeds pronoun in all 4 relevant
phrases

-precedence(pronoun,determiner):-
pronoun preceeds determiner in all 7 relevant
phrases

-precedence(proper-noun,adjective):-
proper-noun preceeds adjective in all 3 relevant
phrases
```

```
-precedence(proper-noun,determiner):-
proper-noun preceeds determiner in all 3 relevant
phrases.

-pronoun is a constituent of noun phrases. It
occurs in 12 phrases.

-adjective is a constituent of noun phrases. It
occurs in 26 phrases.
```

The first two properties are conditional properties and the features of these properties are explictly stated. The obligatority constraint which represents the head of phrases consists of nouns, proper-nouns and pronouns. We provide further information on the number of times each of these categories occur as the head of the phrase. We do not output the features for other properties except the conditional properties because they already succeed in all relevant phrases.

6 How can our model be acquired?

Our model for grammar induction is in a sense, already acquired by many other possible source-target language pairs (those that are amenable to description through the linguistic system of constraints that we accept), because rather than developing a language-dependent implementation of our model, we have implemented a generator of grammar inducers which, to work with a different pair of languages than the one we've chosen, requires only: a) a representative sample of target language phrases, b) that the source grammar be expressed in terms of the linguistic constraints accepted in our model (namely, obligatority, precedence, constituency, requirement, dependency, uniqueness and exclusion), and c) that the target grammar's lexicon be described in our user-friendly provided plain text format, for the subset of words used in the input set of target language phrases.

7 Verification and validation

Verifying and validating that our results are correct comports two tasks:

- checking that the input and the output syntax descriptions are correct. Since these descriptions are done in terms of the linguistic Property grammar formalism, it may be challenging for users unfamiliar with it to be sure that it covers exactly the subset of language addressed. However there is a mechanical procedure to transform context-free grammars into property grammars [Philippe Blache, Stephane Rauzy, 2012], so if the user is more familiar with context-free grammars, and the subset of language allows it, we can at least start from a known, correct Context Free (CF) version and apply the said procedure, either by hand or mechanically, in order to arrive at an equivalent formulation in terms of our linguistic constraints. Alternatively, we can consult a linguist, if one is available who specializes in the involved languages.
- checking that the set of input phrases from the target language is correct and representative. This requires at the very least a native speaker, and if possible also a linguist, given that some users' intuitions about what is correct in their language (and indeed, about even whether they use it "correctly" or not) may not be accurate.

8 Concluding Remarks

We have presented an implemented model for integrating grammatical knowledge representation and reasoning around the CHRG constraint paradigm of problem solving. Our model extends the original grammar induction formalism, WG, in ways motivated by the needs of our particular application: the automatic transformation of an English model of syntax into that of Yorùbá.

We have also presented a proof-of-concept system that allows users to input a source language's grammar's description in modular and declarative fashion, in terms of the linguistic constraints that are supported (constituency, precedence, dependency, obligatority, requirement, exclusion, unicity). Since such descriptions also stand alone as a purely linguistic model of a language's syntax, our system can cater even for users who are not conversant with coding, such as pure linguists wanting to perform syntactic model transformation experiments aided by computers. Their purely (constraint-based) linguistic descriptions of syntax automatically turn into executable code when appended to our own code.

As we have seen, our system automatically transforms a user's syntactic description of a source language into that of a target language, of which only the lexicon and a set of representative sample phrases are known. While demonstrated specifically for English as source language and Yorùbá as target language, our implementation can accept any other pair of languages for inducing the syntactic constraints of one from that of the other, as long as their description can be done in terms of the supported constraints.

We have thus contributed a flexible modelling framework solidly grounded in linguistic knowledge representation which, through the magic of constraint solving, turns into an efficiently executable problem solving tool. Through its inherent modularity, our model can support revision and adaptation quite straightforwardly: new constraints between immediate daughters of any phrase can be modularly added and implemented, without touching the rest of the system.

Also, our model can accommodate solution revisions at execution time by interacting with a linguist whenever constraints can not be induced consistently. Visual representations of the output that would be helpful for linguists in this respect have been proposed in [Adebara I., Dahl V., 2015] and is yet to be implemented.

Other than integrating knowledge-based and efficiency-based paradigms of problem solving, our approach demonstrates that constraint programming is not just about solving classic OR problems. It is also a powerful tool for addressing interdisciplinary applications [Dahl, Miralles, and Becerra, 2012; Becerra, Dahl, and Miralles, 2013; Becerra, Dahl, and Jiménez-López, 2014; Adebara, Dahl, and Tessaris, 2015; Adebara I., Dahl V., 2015].

Further interesting ramifications of our work would include: testing our system for a larger coverage of syntax (we have only addressed noun phrases so far), testing it for other

tonal-sensitive languages, studying how assimilation influences language structure especially in tone languages, studying whether any further constraints or approach extensions would be needed to accommodate families of languages not easily describable within our approach (e.g. languages which have different categories from the source language, those who have different inflectional paradigms from the source language, those that exhibit constraints not among our allowable set of constraints).

Acknowledgments

This research was supported by V. Dahl's NSERC Discovery grant 31611024.

References

Adebara I., Dahl V. 2015. Domes as a Prodigal Shape in Synthesis-Enhanced Parsers. In *SHAPES 3.0*, Constraints and Logic Programming (CSLP'04).

Adebara I., Dahl V. 2015. Shape Analysis as an Aid for Grammar Inductions. In *SHAPES 3.0*, Constraints and Logic Programming (CSLP'04).

Adebara, I.; Dahl, V.; and Tessaris, S. 2015. Completing mixed language grammars through womb grammars plus ontologies. Henning Christiansen, M. D. J. L., and Loukanova, R., eds., *Proceedings of the International Workshop on Partiality, Underspecification and Natural Language Processing*, 32–40.

Becerra, L.; Dahl, V.; and Miralles, E. 2013. On second language tutoring through womb grammars. 12th International Work-Conference on Artificial Neural Networks (IWANN) 2013, June 12-14, Tenerife, Spain.

Becerra, L.; Dahl, V.; and Jiménez-López, M. D. 2014. Womb grammars as a bio-inspired model for grammar induction. In *Trends in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*. Springer International Publishing. 79–86.

Blache, P. 2005. Property grammars: A fully constraint-based theory. In *Proceedings of the First International Conference on Constraint Solving and Language Processing*, CSLP'04, 1–16. Berlin, Heidelberg: Springer-Verlag.

Blache P., Rauzy S. 2012. Hybridization and Treebank Enrichment with Constraint-Based Representations. *Proceedings of International Conference on Language Resources and Evaluation (LREC)*.

Burkett D. and Klein D. 2008. Two Languages are Better than One (for Syntactic Parsing). In *Empirical Methods for Natural Language Processing (EMNLP)*. 877–886.

Charniak E. and Johnson M. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Association for Computational Linguistics (ACL)*.

Christiansen, H. 2005. CHR grammars. *Theory and Practice of Logic Programming (TPLP)* 5(4-5):467–501.

Christiansen, H. 2010. CHRG manual. http://akira.ruc.dk/ henning/chrg/CHRGusersGuide.html.

Cohen S.B. and Smith N.A. 2010. Covariance in Unsupervised Learning of Probabilistic Grammars. *Journal of Machine Learning Research*11:3017–3051

Dahl, V., Blache, P. 2004. Directly Executable Constraint Based Grammars. In *Proc. Journees Francophones de Programmation en Logique avec Contraintes*.

Dahl, V., and Miralles, J. E. 2012. Womb grammars: Constraint solving for grammar induction. Sneyers, J., and Frühwirth, T., eds., In *Proceedings of the 9th Workshop on Constraint Handling Rules*, volume Technical Report CW 624, 32–40.

Dahl, V.; Miralles, E.; and Becerra, L. 2012. On language acquisition through womb grammars. In *7th International Workshop on Constraint Solving and Language Processing*, 99–105.

Duchier, D.; Dao, T.-B.-H.; and Parmentier, Y. 2013. Model-Theory and Implementation of Property Grammars with Features. *Journal of Logic and Computation* 19.

Fagborun, O. 1994 The Yor'uba Koine - its history and linguistic innovations. Munchen : LINCOM EUROPA.

Frühwirth, T. W. 1998. Theory and practice of constraint handling rules. *Journal. Logic. Programming.* 37(1-3):95–138.

Klein D. and Manning C.D. 2004. Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency. In *Association of Computational Linguistics (ACL)*. 478–485.

Headden W.P. and Johnson M. and McClosky D. 2009. Improving Unsupervised Dependency Parsing with Richer Contexts and Smoothing. *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*. 101–109.

Nicolas L. and Molinero M.A., Sagot B., Trigo E., de La Clergerie, Éric, Farré J. and Vergés J.M. 2009. Towards efficient production of linguistic resources: the Victoria Project.

Pereira F.C.N and Schabes Y. 1992. Inside-Outside Reestimation from Partially Bracketed Corpora. *Association of Computational Linguistics (ACL)*. 128–135

Wang M. and Smith N. A. and Mitamura T. 2007. What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA. *Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. 22–32.