Bounding the Maximal Parsing Performance of Non-Terminally Separated Grammars

Franco M. Luque and Gabriel Infante-Lopez

Grupo de Procesamiento de Lenguaje Natural
Universidad Nacional de Córdoba & Conicet
Córdoba, Argentina
{francolq,gabriel}@famaf.unc.edu.ar

Abstract. Unambiguous Non-Terminally Separated (UNTS) grammars have good learnability properties but are too restrictive to be used for natural language parsing. We present a generalization of UNTS grammars called Unambiguous Weakly NTS (UWNTS) grammars that preserve the learnability properties. Then, we study the problem of using them to parse natural language and evaluating against a gold treebank. If the target language is not UWNTS, there will be an upper bound in the parsing performance. In this paper we develop methods to find upper bounds for the unlabeled F_1 performance that any UWNTS grammar can achieve over a given treebank. We define a new metric, show that its optimization is NP-Hard but solvable with specialized software, and show a translation of the result to a bound for the F_1. We do experiments with the WSJ10 corpus, finding an F_1 bound of 76.1% for the UWNTS grammars over the POS tags alphabet.

1 Introduction

Unsupervised parsing of natural language has received an increasing attention in the last years [3,4,8,13,14]. The standard way of evaluating the developed parsers is comparing their output against gold treebanks, such as the Penn Treebank [11], using the precision, recall and F_1 measures [15].

Related to the unsupervised parsing problem is the grammatical inference problem, that deals with theoretical learnability of formal languages. One of the goals of this area is to find classes of languages that are suitable to model natural language and that are learnable in some sense. In [5,6], it is argued that natural language is close to be in the class of Non-Terminally Separated (NTS) languages, and in [5] it is proved that Unambiguous NTS (UNTS) languages are PAC-learnable in polynomial time. However, we will see that UNTS grammars are much less expressive than NTS grammars and that they can not admit elemental sets of natural language sentences.

In order to shorten the gap between UNTS and NTS, we present a slight generalization of UNTS grammars called the Unambiguous Weakly NTS (UWNTS) grammars. In contrast with UNTS, UWNTS grammars are general enough to admit any finite language, but at the same time preserving every other aspect of UNTS grammars, including PAC-learnability in polynomial time.

J.M. Sempere and P. García (Eds.): ICGI 2010, LNAI 6339, pp. 135–147, 2010.

In this work, we study the potential of UWNTS grammars to model natural language syntax. Our approach to study UWNTS grammars is in the context of the unsupervised parsing problem, using the standard quantitative evaluation over gold treebanks. We do not aim at developing a learning algorithm that returns a UWNTS grammar, because the resulting evaluation will depend on the algorithm. Instead, our aim is to find upper bounds for the achievable F_1 performance of all the UWNTS grammars over a given gold treebank, regardless of the learning algorithm and the training material used to induce the grammars. Our bounds should only depend on the gold treebank that is going to be used for evaluation.

In general, knowing an upper bound for a model is useful in at least two senses. First, if it is lower than the performance we want to achieve, we should consider not using that model at all. Second, if it is not low, it might be useful for the development of a learning algorithm, given that it can be assessed how close the performance of the algorithm is to the upper bound.

In this paper we present methods to obtain such upper bounds for the class of UWNTS grammars. Our methods allow us to compute the bounds without explicitly working with grammars, because we are only interested in the parsings of the gold sentences that the grammars can return.

In principle, the function to be optimized is F_1, but instead we define a new metric W, related to F_1, but whose optimization is feasible. Our methods are based in the optimization of W and the recall, and in the translation of these optimal values to an upper bound of the F_1.

The optimization problems are solved by reducing them to the well known Integer Linear Programming (ILP) with binary variables problem, that is known to be NP-Hard, but for which there exists software to solve it [2]. Moreover, we show that the optimization of W is NP-Hard, by reducing the Maximum Independent Set problem for 2-subdivision graphs to it.

Finally, we solve the optimization problems for the WSJ10 subset of the Penn Treebank [11], compute the upper bounds, and compare them to the performances of state-of-the-art unsupervised parsers. Our results show that UWNTS grammars over the POS tags alphabet can not improve state-of-the-art unsupervised parsing performance.

2 Notation and Definitions

We skip conventional definitions and notation for formal languages and context-free grammars, but define concepts that are more specific to our paper.

Given a language L, $Sub(L)$ is the set of non-empty substrings of the elements of L, and $\overline{Sub}(L)$ is the set of non-empty proper substrings. We say that two strings r, s *overlap* if there exist non-empty strings r', s', t such that $r = r't$ and $s = ts'$; $r'ts'$ is called an *overlapping* of r and s. We say that two strings r, s *occur overlapped* in a language L if they overlap and if an overlapping of them is in $Sub(L)$. Given a grammar G and $s \in L(G)$, a substring r of $s = urv$ is called a *constituent in* (u, v) if and only if there is $X \in N$ such that $S \stackrel{*}{\Rightarrow} uXv \stackrel{*}{\Rightarrow} s$.

In contrast, r is called a non-constituent or *distituent in* (u, v) if it is not a constituent in (u, v).

More than in the grammars, we are in fact interested in all the possible ways a given finite set of sentences $S = \{s_1, \ldots, s_n\}$ can be parsed by a particular class of grammars. As we are modeling unlabeled parsing, the parse of a sentence is an unlabeled tree, or equivalently, a bracketing of the sentence. Formally, a *bracketing* of a sentence $s = a_0 \ldots a_{n-1}$ is a set b of pairs of indexes that marks the starting and ending positions of the constituents, consistently representing a tree. A bracketing always contains the full-span bracket $(0, n)$, never has duplicated brackets (it is a set), and does not have brackets of span 1, i.e., of the form $(i, i+1)$. Usually, we will represent the bracketings together with their corresponding sentences. For instance, we will jointly represent the sentence *abcde* and the bracketing $\{(0,5),(2,5),(2,4)\}$ as $ab((cd)e)$. Observe that the full-span bracket is omitted.

Given an unambiguous grammar G and $s \in L(G)$, the *bracketing of* s *with* G, $br_G(s)$, is the bracketing corresponding to the parse tree of s with G. Given $S \subseteq L(G)$, the *bracketing of* S *with* G, $Br_G(S)$, is the set $\{br_G(s) | s \in S\}$.

2.1 UWNTS Grammars

A grammar G is said to be *Non-Terminally Separated (NTS)* if and only if, for all $X, Y \in N$ and $\alpha, \beta, \gamma \in (\Sigma \cup N)^*$, $X \stackrel{*}{\Rightarrow} \alpha\beta\gamma$ and $Y \stackrel{*}{\Rightarrow} \beta$ implies $X \stackrel{*}{\Rightarrow} \alpha Y \gamma$ [5]. A grammar is *Unambiguous NTS (UNTS)* if it is unambiguous and NTS.

Unambiguous NTS grammars are much less expressive than NTS grammars: It can be proved that a UNTS language having two overlapping sentences can not have the overlapping of them as a sentence. For instance, the set $\{ab, bc, abc\}$ can not be a subset of a UNTS language because *abc* is an overlapping of *ab* and *bc*. This situation is very common in the WSJ10 sentences of POS tags, and consequently there is no UNTS grammar that accepts this set. For instance, it has the sentences "NN NN NN" and "NN NN", but "NN NN NN" is an overlapping of "NN NN" with itself.

The limitations of UNTS grammars lead us to define a more expressive class, UWNTS, that is able to parse any finite set of sentences preserving, at the same time, every other aspect of UNTS grammars. The properties of UWNTS will also let us characterize the sets of bracketings of all the possible grammars that parse a given finite set of sentences. This characterization will be at the core of our methods for finding bounds.

Definition 1. *A grammar* $G = \langle \Sigma, N, S, P \rangle$ *is* Weakly Non-Terminally Separated (WNTS) *if* S *does not appear on the right side of any production and, for all* $X, Y \in N$ *and* $\alpha, \beta, \gamma \in (\Sigma \cup N)^*$ *such that* $Y \neq S$,

$$X \stackrel{*}{\Rightarrow} \alpha\beta\gamma \text{ and } Y \stackrel{*}{\Rightarrow} \beta \text{ implies } X \stackrel{*}{\Rightarrow} \alpha Y \gamma.$$

A grammar is Unambiguous WNTS (UWNTS) *if it is unambiguous and WNTS.*

Note that any finite set $\{s_1, \ldots, s_n\}$ is parsed by a UWNTS grammar with rules $\{S \to s_1, \ldots, S \to s_n\}$. It is easy to see that every NTS language is also WNTS.

A much more interesting result is that a WNTS language can be converted into an NTS language without loosing any information.

Lemma 1. *If L is WNTS, then $xLx = \{xsx | s \in L\}$ is NTS, where x is a new element of the alphabet.*

This conversion allows us to prove PAC-learnability of UWNTS languages using the PAC-learnability result for UNTS languages of [5]. If we want to learn a UWNTS grammar from a sample S, we can simply give the sample xSx to the learning algorithm for UNTS grammars, and remove the x's from the resulting grammar. A detailed proof of this result is beyond the scope of this work. We present a much more general result of PAC-learnability in [10].

In a UWNTS grammar G, every substring is either always a constituent in every proper occurrence or always a distituent in every proper occurrence. If two strings r, s occur overlapped in $L(G)$, then at least one of them must be always a distituent. In this case, we say that r and s are *incompatible* in $L(G)$, and we say that they are *compatible* in the opposite case. We say that a set of strings is compatible in $L(G)$ if every pair of strings of the set is compatible in $L(G)$.

Now let us consider a finite set of sentences S, a UWNTS grammar G, and the bracketing of S with G. The given properties imply that in the bracketing there are no substrings marked some times as constituents and some times as distituents. Consequently, the information in $Br_G(S)$ can be simply represented as the set of substrings in $\overline{Sub}(S)$ that are always constituents. We call this the set $Const_G(S)$. When considering all the possible UWNTS grammars G with $S \subseteq L(G)$, there is a 1-to-1 mapping between all the possible bracketings $Br_G(S)$ and all the possible sets $Const_G(S)$. Using this information, we can define our search space $\mathscr{C}(S)$ as

Definition 2. $\mathscr{C}(S) \doteq \{Const_G(S) : G \text{ UWNTS and } S \subseteq L(G)\}$.

With UWNTS grammars we can characterize the search space in a way that there is no need to explicitly refer to the grammars. We can see that the search space is equal to all the possible subsets of $\overline{Sub}(S)$ that are compatible in S:

Theorem 1. $\mathscr{C}(S) = \{C : C \subseteq \overline{Sub}(S), C \text{ compatible in } S\}$.

The proof of $\subseteq$ follows immediately from the given properties. $\supseteq$ is proved by constructing a UWNTS grammar mainly using the fact that S is finite and C is compatible in S.

2.2 UWNTS-SC Grammars

In the parser induction problem, it is usually expected that the induced parsers will be able to parse any sentence, and not only the finite set of sentences S that is used to evaluate them. However, the defined search space for UWNTS grammars does not guarantee this. In Theorem 1, it is evident that the sets of constituents are required to be compatible in S, but may be incompatible with other sets of sentences. For instance, if $S = \{abd, bcd\}$ and $C = \{ab, bc\}$, C

is compatible in S but not in $\{abc\}$. We define UWNTS-SC grammars as the subclass of UWNTS that guarantee that the constituents are compatible with any set of sentences.

Definition 3. *A grammar G is* UWNTS Strongly Compatible (UWNTS-SC) *if it is UWNTS and, for every $r, s \in Const_G(L(G))$, r and s do not overlap. When r and s do not overlap, we say that r and s are* strongly compatible. *And when every pair of a set of strings C do not overlap, we say that C is* strongly compatible.

As UWNTS-SC is a subclass of UWNTS, all the properties of UWNTS grammars still hold. The search space for UWNTS-SC grammars and its characterization are:

Definition 4. $\mathscr{C}_{SC}(S) \doteq \{Const_G(S) : G \text{ UWNTS-SC and } S \subseteq L(G)\}$.

Theorem 2. $\mathscr{C}_{SC}(S) = \{C : C \subseteq \overline{Sub}(S), C \text{ strongly compatible}\}$.

The proof of this theorem is analog to the proof of Theorem 1.

Theorem 2 shows a more natural way of understanding the motivation under UWNTS-SC grammars. While the compatibility property depends on the sample S, the strong compatibility do not. It can be checked for a given set of constituents without looking at the sample, and if it holds, we know that the set if compatible for any sample.

3 The W Measure and Its Relationship to the F_1

In this section we will study the evaluation measures in the frame of UWNTS grammars. In general, a measure is a function of similarity between a given gold bracketing $B = \{b_1, \ldots, b_n\}$ of a set of gold sentences and a proposed bracketing $\hat{B} = \{\hat{b}_1, \ldots, \hat{b}_n\}$ over the same set of sentences. The standard measures used in unsupervised constituent parsing are the micro-averaged precision, recall and F_1 as defined in [8]:

$$P(\hat{B}) \doteq \frac{\sum_{k=1}^{n} |\hat{b}_k \cap b_k|}{\sum_{k=1}^{n} |\hat{b}_k|} \quad R(\hat{B}) \doteq \frac{\sum_{k=1}^{n} |\hat{b}_k \cap b_k|}{\sum_{k=1}^{n} |b_k|} \quad F_1(\hat{B}) \doteq \frac{2P(\hat{B})R(\hat{B})}{P(\hat{B}) + R(\hat{B})}.$$

Note that these measures differ from the standard PARSEVAL measures [1], because from the definition of bracketing it follows that the syntactic categories are ignored and that unary branches are ignored.

Another way to define precision and recall is in term of two measures that we will call hits and misses.

Definition 5. *The* hits *is the number of proposed brackets that are correct, and the* misses *is the number of proposed brackets that are not correct. Formally,*

$$H(\hat{B}) \doteq \sum_{k=1}^{n} |\hat{b}_k \cap b_k| \qquad M(\hat{B}) \doteq \sum_{k=1}^{n} |\hat{b}_k - b_k|.$$

Using these two measures, and defining $K \doteq \sum_{k=1}^{n} |b_k|$ to be the number of brackets in the gold bracketing, we have

$$P(\hat{B}) = \frac{H(\hat{B})}{H(\hat{B}) + M(\hat{B})} \qquad R(\hat{B}) = \frac{H(\hat{B})}{K}.$$

As we saw in the previous section, in the case of UWNTS grammars the bracketings can be represented as sets of constituents $\hat{C}$. So, we will rewrite the definitions of the measures in terms of $\hat{C}$ instead of $\hat{B}$. Observe that, if $s \in \hat{C}$, $\hat{B}$ contains all the occurrences of s marked as a constituent. But in the gold B, s may be marked some times as a constituent and some times as a distituent. Let $c(s)$ and $d(s)$ be number of times s appears in B as a constituent and as a distituent respectively:

Definition 6.

$$c(s) \doteq |\{(u, s, v) : uv \neq \lambda, usv \in S \text{ and } s \text{ is a constituent in } (u, v)\}|$$
$$d(s) \doteq |\{(u, s, v) : uv \neq \lambda, usv \in S \text{ and } s \text{ is a distituent in } (u, v)\}|$$

Then, for every $s \in \hat{C}$, $\hat{B}$ will have $c(s)$ hits and $d(s)$ misses. This is,

$$H(\hat{C}) = \sum_{s \in \hat{C}} c(s) \qquad M(\hat{C}) = \sum_{s \in \hat{C}} d(s).$$

Using this, we can see that

$$F_1(\hat{C}) = \frac{2 \sum_{s \in \hat{C}} c(s)}{K + \sum_{s \in \hat{C}} c(s) + d(s)}.$$

Now that we have written the F_1 in terms of $\hat{C}$, we would like to define an algorithm to find the optimal $F_1(\hat{C})$ for every $\hat{C} \in \mathscr{C}(S)$. As the search spaces is finite, a simple algorithm to find $\max_{\hat{C}} F_1(\hat{C})$ is to compute the F_1 for every $\hat{C}$. The problem is that the order of this algorithm is $O(2^{|\overline{Sub(S)}|})$.

Given that the optimization of the F_1 doesn't seem to be feasible, we will define another measure W which optimization will result to be more tractable. This measure has its own intuition, and is a natural way to combine hits and misses. It is very different to the F_1, but we will see that an upper bound of W can be translated to an upper bound of the F_1 measure. We will also see that an upper bound of the recall R can be used to find a better bound for the F_1.

We want W to be such that a high value for it reflects a big number of hits and a low number of misses, so we simply say:

Definition 7. $W(\hat{B}) \doteq H(\hat{B}) - M(\hat{B})$.

Note that, when dealing with UWNTS grammars, as H and M are linear expressions over $\hat{C}$, W will also be linear over $\hat{C}$, unlike the F_1 measure. This is what will make it more tractable.

Table 1. Comparison of the F_1 and W measures: The scores of two bracketings with respect to the gold bracketing $(S, B) = \{(ab)c, a(bd), (ef)g, efh, efi\}$.

i	$(S, \hat{B}_i)$	P	R	F_1	H	M	W
1	$\{(ab)c, (ab)d, efg, efh, efi\}$	50%	33%	40%	1	1	0
2	$\{(ab)c, (ab)d, (ef)g, (ef)h, (ef)i\}$	40%	67%	50%	2	3	-1

W and F_1 are different measures that do not define the same ordering over the candidate bracketings $\hat{B}$. For instance, Table 1 shows the scores for two different bracketings $\hat{B}_1, \hat{B}_2$ with respect to the same gold bracketing. Here, F_1 is higher for $\hat{B}_2$ but W is higher for $\hat{B}_1$.

Anyway, it can be proved that the W and F_1 measures are related through the following formula:

$$f_1(r, w) = \frac{2r}{1 + 2r - \frac{w}{K}}. \tag{1}$$

From this formula it can be seen that the F_1 is monotonically increasing in both w and r. Then, if r and w are upper bounds for their respective measures, $f_1(r, w)$ is an upper bound for the F_1 measure. If we do not know an upper bound for the recall, we can simply use the bound $r = 1$.

4 The Optimization of W and R

In this section we first formalize the optimization problems of W and R over the UWNTS and UWNTS-SC grammars. We define them in terms of a more general problem of optimization of a score over a class of grammars. Then, we show how to reduce the problems to Integer Linear Programming problems, using the fact that W and R are computed as linear expressions in terms of $\hat{C}$. Finally, we show that the problems are NP-Hard.

Definition 8. *Given a class of grammars $\mathcal{G}$, a score function s, a set of gold sentences S and a set of gold bracketings B, the* Maximum Score Grammar (MSG) *problem is such that it computes*

$$MSG(\mathcal{G}, s, S, B) = \max_{G \in \mathcal{G}, S \subseteq L(G)} s(B, Br_G(S)).$$

Definition 9. *The* Maximum W UWNTS (MW-UNTS)*,* Maximum W UWNTS-SC (MW-UNTS-SC)*,* Maximum R UWNTS (MW-UNTS) *and* Maximum R UWNTS-SC (MW-UNTS-SC) *problems are such that they compute*

$$\begin{aligned} MW\text{-}UWNTS(S, B) &= MSG(UWNTS, W, S, B) \\ MW\text{-}UWNTS\text{-}SC(S, B) &= MSG(UWNTS\text{-}SC, W, S, B) \\ MR\text{-}UWNTS(S, B) &= MSG(UWNTS, R, S, B) \\ MR\text{-}UWNTS\text{-}SC(S, B) &= MSG(UWNTS\text{-}SC, R, S, B). \end{aligned}$$

4.1 Solving the Problems for UWNTS Grammars

Let us first consider the problem of W maximization, MW-UWNTS. By the characterization of the search space of Theorem 1, we have that

$$\text{MW-UWNTS}(S,B) = \max_{C \subseteq \overline{Sub}(S), C \text{ compatible in } S} \sum_{s \in C} c(s) - d(s).$$

Now, let $H(S,B) = \langle V, E, w\rangle$ be an undirected weighted graph such that $V = \overline{Sub}(S)$, $w(s) = c(s) - d(s)$ and $E = \{(s,t) : s,t \text{ incompatible in } S\}$. An independent set of a graph is a set of nodes such that every pair of nodes in it is not connected by an edge. So, C is an independent set of $H(S,B)$ if and only if C is compatible in S. Now, consider the Maximum Weight Independent Set problem, MWIS(G), that given G returns the weight of the independent set with maximum weight of G [7]. Clearly,

$$\text{MW-UWNTS}(S,B) = \text{MWIS}(H(S,B)).$$

This is a reduction of our problem to the MWIS problem, that is known to be NP-Hard [7]. At its turn, MWIS is reducible to the Integer Linear Programming (ILP) problem with binary variables. This problem is also NP-Hard, but there exists software that implements efficient strategies for solving some of its instances [2].

An ILP problem with binary variables is defined by three parameters $\langle \mathbf{x}, f, \mathbf{c}\rangle$: A set of variables $\mathbf{x}$ that can take values in $\{0,1\}$, an objective linear function $f(\mathbf{x})$ that has to be maximized, and a set of linear constraints $\mathbf{c}(\mathbf{x})$ that must be satisfied. The result of the ILP problem is a valuation of the variables that satisfies the constraints and maximizes the objective function.

In the case of MWIS, given a weighted graph $G = \langle V, E, w\rangle$, we define a binary variable x_v for every node v. A valuation in $\{0,1\}$ of the variables will define a subset of nodes $\{v | x_v = 1\}$. To ensure that the possible subsets are independent sets, we define the constraints $\mathbf{c}(\mathbf{x}) = \{x_v + x_w \leq 1 | (v,w) \in E\}$. Finally, the objective function is $f(\mathbf{x}) = \sum_{v \in V} x_v w(v)$. Using this instance $I(G) = \langle \mathbf{x}, f, \mathbf{c}\rangle$, we have that in general MWIS(G) = ILP($I(G)$), and in particular that

$$\text{MW-UWNTS}(S,B) = \text{ILP}(I(H(S,B))).$$

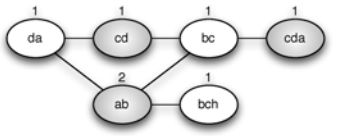
We illustrate the reduction of MW-UWNTS to MWIS and then to ILP with the example instance of Fig. 1 (a). The sets of substrings such that $w(c) \geq 0$ is $\{da, cd, bc, cda, ab, bch\}$. The input graph for the MWIS problem and the instance of the ILP problem are given in Fig. 1 (b) and (c).

Now we consider the solution of the problem of recall maximization, MR-UWNTS. We can do a similar reduction to MWIS changing the graph weights from $c(s) - d(s)$ to $c(s)$, and then reduce MWIS to ILP. If $H'(S,B)$ is the MWIS instance, then

$$\text{MR-UWNTS}(S,B) = \frac{1}{K}\text{ILP}(I(H'(S,B))).$$

(a) $(S, B) = \{(da)((bc)h), b((cd)a), (ab)e, (ab)f, (ab)g, (bc)i, (da)j\}$

(b)



(c) $\mathbf{x} = \{x_{da}, x_{cd}, x_{bc}, x_{cda}, x_{ab}, x_{bch}\}$
$f(\mathbf{x}) = x_{da} + x_{cd} + x_{bc} + x_{cda} + 2x_{ab} + x_{bch}$
$\mathbf{c}(\mathbf{x}) = \{x_{da} + x_{cd} \leq 1, x_{cd} + x_{bc} \leq 1,$
$x_{bc} + x_{cda} \leq 1, x_{da} + x_{ab} \leq 1,$
$x_{ab} + x_{bc} \leq 1, x_{ab} + x_{bch} \leq 1\}$

Fig. 1. (a) A gold bracketing. (b) Graph for the MWIS problem. The shadowed nodes form the solution. (c) Instance for the ILP problem.

(a) H: u x y v (2-subdivision of G: u v)
(b) $\Sigma = \{a_u, b_u, a_v, b_v, c_{uv}, d, e\}$
$Substrings = \{a_u b_u, a_v b_v, b_u c_{uv}, c_{uv} a_v\}$
$(S, B) = \{(a_u b_u)c_{uv}, (b_u c_{uv})a_v, (c_{uv} a_v)b_v, (b_u c_{uv})d, (c_{uv} a_v)d, (a_v b_v)d, (a_v b_v)e\}$
(c) $H(S, B)$: $\overset{1}{a_u b_u}$ $\overset{1}{b_u c_{uv}}$ $\overset{1}{c_{uv} a_v}$ $\overset{1}{a_v b_v}$

Fig. 2. Example of conversion of a 2-subdivision graph H (a), instance of the MIS problem, to a pair (S, B) (b), instance of the MW-UWNTS problem. In (c) it is shown that H is equivalent to $H(S, B)$.

4.2 Solving the Problems for UWNTS-SC Grammars

The problems for UWNTS-SC grammars can also be reduced to MWIS as in the previous section just changing in the graph construction the definition of the edge set. Instead of saying that $(s, t) \in E$ iff s and t are incompatible in S, we say that $(s, t) \in E$ iff s and t are strongly incompatible, this is, s and t overlap. But this reduction leads to a much bigger number of edges. In our experiments with the WSJ10, this resulted in a number of ILP constraints that was unmanageable by our working version of the ILP software we used.

For instance, if there are n substrings that end with the symbol a and m substrings that start with a, there will be nm edges/constraints. These constraints express the fact that "every string starting with an a is strongly incompatible with every string ending with an a", or equivalently "every string starting with an a have value 0 or every string ending with an a have value 0". But this can be expressed with less constraints using a new ILP variable y_a that has value 0 if every substring starting with a has value 0, and value 1 if every substring ending with a has value 0. To do this we use, for each substring of the form as, the constraint $x_{as} \leq y_a$ and for each substring of the form ta, the constraint $x_{ta} \leq 1 - y_a$ leading to $n + m$ constraints instead of nm.

In the general form, the ILP instance is $I = \langle \mathbf{x}, f, \mathbf{c} \rangle$, where $\mathbf{c}(\mathbf{x}) = \{x_s \leq y_t | s \in V, t$ proper prefix of $s\} \cup \{x_s \leq 1 - y_u | s \in V, u$ proper suffix of $s\}$.

4.3 NP-Hardness of the Problems

It can be shown that the presented problems, MW-UWNTS, MR-UWNTS, MW-UWNTS-SC, and MR-UWNTS-SC, are all NP-Hard. In this section, we prove only that the MW-UWNTS problem is NP-Hard. The proofs for the rest of the problems are analogous.

Theorem 3. *MW-UWNTS is NP-Hard.*

Proof. We will prove this by reducing the NP-Hard Maximum Independent Set (MIS) graph problem to the MW-UWNTS problem. MIS is the optimization problem over unweighted graphs that is equivalent to the MWIS problem over weighted graphs where all the weights are 1. We have to provide a way to convert instances of MIS to instances of MW-UWNTS. To simplify this process, we will restrict the domain of the MIS problem to the class of 2-subdivision graphs, where it is still known to be NP-Hard [12]. This is the key idea of the proof. The 2-subdivision graphs are those graphs that are obtained from another graph by replacing every edge $\{u, v\}$ by a path $uxyv$, where x and y are new nodes. The path $uxyv$ is called the *2-subdivision path* of the edge $\{u, v\}$ and x and y are called the *2-subdivision nodes* of $\{u, v\}$.

Let H be a 2-subdivision graph for which we want to solve the MIS problem, and let G be the graph from which H is obtained by 2-subdividing it. We will construct an instance of the MW-UWNTS problem (S, B) in terms of H, this is, a set of gold sentences and its corresponding gold bracketing. The instance (S, B) will be such that, when reduced to the graph $G^w_{S,B}$ as explained in section 4.1, this graph will have all weights equal to 1 and will be structurally equal to the original graph H. As a solution of the MW-UWNTS problem gives a solution of the MWIS problem for $G^w_{S,B}$, this solution is also a solution of the MIS problem for H. This way, we can successfully solve the MIS problem in terms of a MW-UWNTS problem.

To describe the instance (S, B), we will first define the alphabet it uses, then the set of substrings that occur in S, and finally the gold sentences S and the brackets that conform the gold bracketing B. The alphabet will have, for each node v in H that is also in G, two symbols a_v and b_v, and for each 2-subdivision path $uxyv$ in H, a symbol c_{uv}. The set of substrings will have one substring for each node in H. The substrings must overlap in a way that the overlappings encode the edges of H. For every node v in G we will use the substring $a_v b_v$ in the treebank, and for every 2-subdivision nodes x and y of $\{u, v\}$ we will use the substrings $b_u c_{uv}$ and $c_{uv} a_v$ respectively. Note that, for every 2-subdivision path $uxyv$ of H, the string corresponding to the node u overlaps with the string of the node x, the string of x overlaps with the string of y, and the string of y overlaps with the string of v. Also, it can be seen that there is no overlapping that doesn't come from an edge of a 2-subdivision path of H.

The sentences must contain the mentioned substrings in a way that all the pairs of substrings that overlap, effectively appear overlapped. This way, the edges of H will be rebuilt from the treebank. To do this, for each 2-subdivision path $uxyv$ we define the sentences $\{a_u b_u c_{uv}, b_u c_{uv} a_v, c_{uv} a_v b_v\}$.

We must define also the brackets for these sentences. These have to be such that every substring s that correspond to a node in H has weight $w(s) = c(s) - d(s) = 1$. This will not be possible unless we use some extra sentences. For instance, if we use the bracketings $\{(a_u b_u)c_{uv}, (b_u c_{uv})a_v, (c_{uv}a_v)b_v\}$ we will have $w(a_u b_u) = 1$, $w(b_u c_{uv}) = w(b_u c_{uv}) = 0$, and $w(a_v b_v) = -1$. All the weights can be set to 1 using new symbols and sentences like $\{(b_u c_{uv})d$, $(c_{uv}a_v)d$, $(a_v b_v)d$, $(a_v b_v)e\}$. The new substrings, such as $c_{uv}d$, will all have weight < 0 so they will not appear in the graph. In general, it will always be possible to fix the weights of the substrings by adding new sentences.

The described conversion of H to (S, B) can be carried in $O(|V(H)|)$, or equivalently $O(|V(G)| + |E(G)|)$, time and space.

A very simple example of this process is shown in Fig. 2. □

5 Upper Bounds for the WSJ10 Treebank

This section shows the results of computing concrete upper bounds for the classes of UWNTS and UWNTS-SC grammars over the WSJ10 treebank. The WSJ10 consists of the sentences of the WSJ Penn Treebank whose length is of at most 10 words after removing punctuation marks [8].

There is a total of 63742 different non-empty substrings of POS tags in $\overline{Sub}(S)$. To solve the optimization problems, the strings with $w(s) \leq 0$ can be ignored because they do not improve the scores. In the case of the MW problems, where $w(s) = c(s) - d(s)$, the number of strings with $w(s) > 0$ is 7029. In the case of the MR problems, where $w(s) = c(s)$, the number of strings with $w(s) > 0$ is 9112. The sizes of the resulting instances are summarized in Table 2. Observe that the number of edges of the MWIS instances for UWNTS-SC grammars have a higher order of magnitude than the size of the improved ILP instances.

Using the obtained results of Maximal W and Maximal R we proceeded to compute the upper bounds for the F_1 using (1) from Sect. 3. Table 3 shows the results, together with the performance of actual state-of-the-art unsupervised parsers. We show the computed maximal W and maximal recall for UWNTS and UWNTS-SC grammars, and also the values of all the other measures associated to these solutions. We also show the upper bounds for the F_1 in the rows labeled UBoundF_1(UWNTS) and UBoundF_1(UWNTS-SC), together with their corresponding precision and recall. RBranch is a baseline parser that parses every sentence with a right-branching bracketing. DMV+CCM is the parser from

Table 2. Sizes of the MWIS and ILP instances generated by the MW and MR problems for the WSJ10 treebank

		MW-UWNTS	MR-UWNTS	MW-UWNTS-SC	MR-UWNTS-SC
MWIS	Nodes	7029	9112	7029	9112
	Edges	1204	45984	1467257	3166833
ILP	Variables	7029	9112	26434	28815
	Constraints	1204	45984	67916	79986

Table 3. Summary of the results of our experiments with the WSJ10, in contrast with state-of-the-art unsupervised parsers. The numbers in bold mark the upper bounds or maximal values obtained in each row.

Model	P	R	F_1	H	M	W
RBranch	55.1	70.0	61.7			
DMV+CCM	69.3	88.0	77.6			
U-DOP	70.8	88.2	78.5			
Incremental	75.6	76.2	75.9			
MW-UWNTS	91.2	62.8	74.4	22169	2127	**20042**
MR-UWNTS	73.8	**69.0**	71.3	24345	8643	15702
UBoundF_1(UWNTS)	85.0	69.0	**76.1**			
MW-UWNTS-SC	89.1	52.2	65.8	18410	2263	**16147**
MR-UWNTS-SC	65.3	**61.1**	63.1	21562	11483	10079
UBoundF_1(UWNTS-SC)	79.9	61.1	**69.2**			

Klein and Manning [8], U-DOP is the parser from Bod [4] and Incremental is the parser from Seginer [13].

6 Discussion

Our bounding methods are specific to the evaluation approach of comparing against gold treebanks, but this is the most accepted and supported approach, and it is scalable and unbiased with respect to the evaluator [15]. Our methods are also specific to the unlabeled F_1 measure as we defined it, with micro-averaging and removal of unary brackets. Also in [15], micro-averaging and removal of trivial structure are proposed as the preferred evaluation method.

Our experiments show that UWNTS grammars over short English sentences of POS tags have a low performance compared to state-of-the-art unsupervised parsers. We believe that these experiments provide enough evidence to conclude that unsupervised parsing over POS tags requires more expressive grammars. However, our results should not be interpreted as a theoretical conclusion about the relationship between natural languages and UWNTS languages in general.

This work is an extended and improved version of [9]. Here, we explicitly define the UWNTS grammars, improve the bounding method using the recall, formalize the optimization problems, and show that they are NP-Hard. We also define the subclass of UWNTS-SC grammars and provide an alternative reduction to ILP for them.

Acknowledgments

This work was supported in part by grant PICT 2006-00969, ANPCyT, Argentina. We would like to thank Pablo Rey (UDP, Chile) for his help with ILP, and Demetrio Martín Vilela (UNC, Argentina) for his comments.

References

1. Abney, S., Flickenger, S., Gdaniec, C., Grishman, C., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M., Roukos, S., Santorini, B., Strzalkowski, T.: A procedure for quantitatively comparing the syntactic coverage of English grammars. In: Black, E. (ed.) Proceedings of a workshop on Speech and natural language, pp. 306–311 (1991)
2. Achterberg, T.: SCIP - a framework to integrate Constraint and Mixed Integer Programming. Tech. rep. (2004)
3. Adriaans, P.W., Vervoort, M.: The EMILE 4.1 grammar induction toolbox. In: Adriaans, P.W., Fernau, H., van Zaanen, M. (eds.) ICGI 2002. LNCS (LNAI), vol. 2484, pp. 293–295. Springer, Heidelberg (2002)
4. Bod, R.: Unsupervised parsing with U-DOP. In: Proceedings of the 10th CoNLL (CoNLL-X), pp. 85–92 (2006)
5. Clark, A.: PAC-learning unambiguous NTS languages. In: Sakakibara, Y., Kobayashi, S., Sato, K., Nishino, T., Tomita, E. (eds.) ICGI 2006. LNCS (LNAI), vol. 4201, pp. 59–71. Springer, Heidelberg (2006)
6. Clark, A.: Learning deterministic context free grammars: The Omphalos competition. Machine Learning 66(1), 93–110 (2007)
7. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103 (1972)
8. Klein, D., Manning, C.D.: Corpus-based induction of syntactic structure: Models of dependency and constituency. In: Proceedings of the 42nd ACL, pp. 478–485 (2004)
9. Luque, F., Infante-Lopez, G.: Upper bounds for unsupervised parsing with Unambiguous Non-Terminally Separated grammars. In: Proceedings of CLAGI, 12th EACL, pp. 58–65 (2009)
10. Luque, F., Infante-Lopez, G.: PAC-learning unambiguous k, l-NTS$^{\leq}$ languages. In: J.M. Sempere and P. García (Eds.): ICGI 2010. LNCS(LNAI), vol. 6339, pp. 122–134. Springer, Heidelberg (2010)
11. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The Penn treebank. Computational Linguistics 19(2), 313–330 (1994)
12. Poljak, S.: A note on stable sets and coloring of graphs. Commentationes Mathematicae Universitatis Carolinae 15(2), 307–309 (1974)
13. Seginer, Y.: Fast unsupervised incremental parsing. In: Proceedings of the 45th ACL, pp. 384–391 (2007)
14. van Zaanen, M.: ABL: alignment-based learning. In: Proceedings of the 18th conference on Computational linguistics, pp. 961–967 (2000)
15. van Zaanen, M., Geertzen, J.: Problems with evaluation of unsupervised empirical grammatical inference systems. In: Clark, A., Coste, F., Miclet, L. (eds.) ICGI 2008. LNCS (LNAI), vol. 5278, pp. 301–303. Springer, Heidelberg (2008)