Syntactic Parsing and Compound Recognition via Dual Decomposition: Application to French

Joseph Le Roux[1] and **Matthieu Constant**[2] and **Antoine Rozenknop**[1]
(1) LIPN, Université Paris 13 – Sorbonne Paris Cité, CNRS
(2) LIGM, Université Paris Est, CNRS
leroux@univ-paris13.fr, mconstan@univ-mlv.fr, antoine.rozenknop@lipn.univ-paris13.fr

Abstract

In this paper we show how the task of syntactic parsing of non-segmented texts, including compound recognition, can be represented as constraints between phrase-structure parsers and CRF sequence labellers. In order to build a joint system we use dual decomposition, a way to combine several elementary systems which has proven successful in various NLP tasks. We evaluate this proposition on the French SPMRL corpus. This method compares favorably with pipeline architectures and improves state-of-the-art results.

1 Introduction

Dual decomposition (DD), which can be used as a method to combine several elementary systems, has already been successfully applied to many NLP tasks, in particular syntactic parsing, see (Rush et al., 2010; Koo et al., 2010) *inter alia*. Intuitively, the principle can be described quite simply: at decoding time, the combined systems seek for a consensus on common subtasks, in general the prediction of some parts of the overall structure, via an iterative process imposing penalties where the systems disagree. If the systems converge to a solution, it is formally guaranteed to be optimal. Besides, this approach is quite flexible and easy to implement. One can add or remove elementary systems without rebuilding the architecture from the ground up. Moreover, the statistical models for the subsystems can often be estimated independently at training time.

In this paper we show how syntactic parsing of unsegmented texts, integrating compound recognition, can be represented by constraints between phrase-structure parsers and sequence labellers, either for compound recognition or part-of-speech (POS) tagging, and solved using DD. We compare this approach experimentally with pipeline architectures: our system demonstrates state-of-the-art performance. While this paper focuses on French, the approach is generic and can be applied to any treebank with compound information, and more generally to tasks combining segmentation and parsing.

This paper is structured as follows. First, we describe the data we use to build our elementary systems. Second, we present related work in compound recognition, in particular for French, and the type of information one is able to incorporate in tag sets. Third, we show how CRF-based sequence labellers with these different tag sets can be combined using DD to obtain an efficient decoding algorithm. Fourth, we extend our method to add phrase-structure parsers in the combination. Finally, we empirically evaluate these systems and compare them with pipeline architectures.

2 Data

We use the phrase-structure treebank released for the SPMRL 2013 shared task (Seddah et al., 2013). This corresponds to a new version of the French Treebank (Abeillé et al., 2003). One of the key differences between French data and other treebanks of the shared task is the annotation of compounds. Compounds are sequences of words with a certain degree of semantic non-compositionality. They form a single lexical unit to which one can assign a single POS. In the SPRML corpus 15% of the tokens belong to a compound, or 12.7% if we omit numerals: the training, development and test sets respectively comprise 23658, 2120 and 4049 compounds.

In the treebank, compounds are annotated as subtrees whose roots are labelled with the POS of the compounds with a + suffix. Each leaf under a compound is the daughter of its own POS tag, which is in turn the daughter of the root of the compound. For example, the tree in Figure 1 contains a subtree with the compound adverb *pour l'instant* (so far) whose category `ADV+` dominates the preposition `pour`, the determiner *l'*, and the noun *instant*.

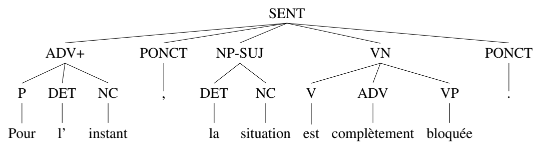

Figure 1: Syntatic annotation in the SPRML FTB: *So far, the situation has been completely blocked.*

The sequence labellers used in the experiments are able to exploit external lexical resources that will help coping with data missing from the training corpus. These resources are dictionaries, consisting of triplets `(flexed form, lemma, POS tag)`, where form and lemma may be compound or simple. Several such dictionaries exist for French. We use:

- DELA (Courtois et al., 1997) contains a million entries, among which 110,000 are compounds;
- Lefff (Sagot, 2010) contains 500,000 entries, among which 25,000 are compounds;
- Prolex (Piton et al., 1999) is a toponym dictionary with approximately 100,000 entries.

The described resources are additional to the SPMRL shared task data (Seddah et al., 2013), but were also used in (Constant et al., 2013a) for the shared task.

3 Compound Recognition

3.1 Related Work

The compound recognition traditionally relies on 2 types of information: lexical and syntactic clues. A strong lexical association between the tokens of a compound can be detected using a compound dictionary or by measuring a degree of *relatedness*, which can be learnt on a corpus. Some recent approaches use sequence labellers. The linear chain CRF model (Lafferty et al., 2001) is widely used, see for example (Vincze et al., 2011; Constant and Tellier, 2012). It has proven to be a very adequate model: it is flexible enough to incorporate information from labelled data and external resources (POS taggers, compound lexicons or named entity recognisers).

The compound recognition may also be directly performed by syntactic parsers learnt from corpora where compounds are marked, such as the one we use in this paper[1] (Arun and Keller, 2005; Green et al., 2011; Green et al., 2013; Constant et al., 2013b), but these results are contradictory. Green et al. (2011) experimentally show that a lexicalised model is better than an unlexicalised one. On the other hand, Constant et al. (2013b) show that, using products of PCFG-LA (Petrov, 2010), unlexicalised models can be more accurate. They obtain performance on a par with a linear chain CRF system without external information. But such information is difficult to incorporate directly in a PCFG-LA model. Constant et al. (2012) resort to a reranker to add arbitrary features in the parse selection process, but their system showed inferior performance compared with a CRF model with access to the same external information.

[1]Such an approach has been used already for joint named entity recognition and parsing based on CRF (Finkel and Manning, 2009).

3.2 Annotation schemes

Compound recognition can be seen as a segmentation task which consists in assigning to each token a label with segmentation information. We use label `B` if the token is the beginning of a word (single or compound), and label `I` if the token is inside a compound, but not in initial position. This lexical segmentation can be enriched with additional information, for example POS tags of compounds or tokens in compounds, and gives a variety of tag sets. This leads us to define 5 simple tag sets for our problem, each with very simple information, that will be combined in the next section. These tag sets are exemplified on a sentence with the compound *vin rouge* (red wine).

1. (basic) recognition with two labels (`B` and `I`)
   Luc/`B` *aime/*`B` *le/*`B` *vin/*`B` *rouge/*`I` (Luc likes red wine)
2. (partial) recognition with compound POS tags: `[BI]-POS` for tokens in compounds; `B` for others
   Luc/`B` *aime/*`B` *le/*`B` *vin/*`B-NC+` *rouge/*`I-NC+`
3. (partial-internal) recognition with token POS tags in compounds
   Luc/`B` *aime/*`B` *le/*`B` *vin/*`B-NC` *rouge/*`I-ADJ`
4. (complete) recognition with POS tags for all tokens; in compounds use compound POS tags
   Luc/`B-NPP` *aime/*`B-V` *le/*`B-DET` *vin/*`B-NC+` *rouge/*`I-NC+`
5. (complete-internal) recognition with POS tags for all tokens; in compounds use token POS tags
   Luc/`B-NPP` *aime/*`B-V` *le/*`B-DET` *vin/*`B-NC` *rouge/*`I-ADJ`

4 Dual decomposition for compound recognition using CRFs

4.1 A maximisation problem

4.1.1 CRF

A conditional random field (Lafferty et al., 2001), or CRF, is a tuple $c = (\Sigma, \mathcal{L}_c, w_c, \{f_p^c\}_p)$ which defines the conditional probability of a sequence of labels $y \in \mathcal{L}_c^*$ given a sequence of words of the same length $x \in \Sigma^*$ as a logistic regression of the form:

$$P_c(y|x) = \frac{\exp\left(\sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p)\right)}{Z(w_c, x)}, \text{ where} \tag{1}$$

- $w_c \in \mathbb{R}^d$ is a d-dimensional weight vector, where d is the number of features of the system,
- Z is the partition function
- $\mathcal{P}(x)$ is a set of *places*, in our case the set of unigram and bigram decompositions of sequences of words. A place p is of the form $[i]_x$ for unigrams and $[i, i+1]_x$ for bigrams. We omit x when context is unambiguous.
- y_p is the restriction of y to the place p, and we will write y_i for $y_{[i]}$ and $y_i y_{i+1}$ for $y_{[i,i+1]}$
- f_p^c is the feature function for the place p that projects (x, y_p) on $\mathbb{R}^d$.

Our goal is to find the best labelling, i.e. the one that maximises the conditional probability given a sequence of tokens. One can observe that this labelling also maximises the numerator of Equation 1, as $Z(w_c, x)$ does not depend on y. We therefore write:

$$\hat{y}^c = \arg\max_y \sigma_c(x, y) = \arg\max_y \sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p) \tag{2}$$

4.1.2 Viterbi Algorithm for CRFs

Since our combination of CRF systems relies on the Viterbi algorithm, we review it briefly. For a given input sentence $x = x_1 \dots x_n$, we represent the problem of finding the best labelling with a CRF c as a best path algorithm in a directed acyclic graph $\mathcal{G}_c = (\mathcal{V}, \mathcal{E})$ built from a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. Nodes are pairs (x_i, l) where x_i is an input token and l is an admissible label for x_i.[2] Edges connect nodes of the form (x_i, l) to nodes of the form (x_{i+1}, l') and the weights of these arcs are given by c. In order to find the weight of the best path in this graph, that corresponds to the score of the best labelling, we use Algorithm 1.[3] One can remark that the score of a node decomposes as a score s_1, computed from a vector of unigram features, written $f^c_{[i]}(x, l)$, and a score s_2 computed from a vector of bigram features, written $f^c_{[i-1,i]}(x, l', l)$.[4] The Viterbi algorithm has a time complexity linear in the length of the sentence and quadratic in the number of labels of the CRF.

Algorithm 1 Viterbi Algorithm for CRFs with unigram and bigram features

```
1:  **Viterbi**($\mathcal{G}_c, w_c, \{f^c_p\}_p, \Lambda^{BI}, \Lambda^{IB}$):
2:  **for all** node $v$ **do**
3:      $\pi[v] = -\infty$
4:  **end for**
5:  $\pi[$ (<s>,START) $] = w \cdot f_{p_0}(x,$ START$)$
6:  **for all** non initial node $v = (x_i, l)$ in topological order **do**
7:      $s_1 \leftarrow w_c \cdot f^c_{[i]}(x, l)$
8:      $s_2 \leftarrow -\infty$
9:      **for all** incoming edge $v' = (x_{i-1}, l') \rightarrow v$ **do**
10:         $t \leftarrow \pi[v'] + w_c \cdot f^c_{[i-1,i]}(x, l', l)$
11:         $t \leftarrow t - b(l')i(l) \cdot \Lambda^{BI}[i] - i(l')b(l) \cdot \Lambda^{IB}[i]$     ▷ only for DD: we ignore this line otherwise
12:         **if** $t > s_2$ **then**
13:             $s_2 = t$
14:         **end if**
15:     **end for**
16:     $\pi[v] \leftarrow s_1 + s_2$
17: **end for**
18: **return** the best scoring path $\pi[$ (</s>,STOP) $]$
```

4.2 Dual decomposition for CRF combinations

In Section 3.2 we described several annotation schemes that lead to different CRF models. These schemes give the same lexical segmentation information but they use more or less rich part-of-speech tag sets. It is not clear *a priori* if the richness of the tag set has a beneficial effect over segmentation prediction: there is a compromise between linguistic informativeness and data sparseness. Instead of trying to find the best annotation scheme, we propose a consensus-based joint system between several CRF-based sequence labellers for the task of text segmentation relying on dual decomposition (Rush et al., 2010). This system maximises the sum of the scores of combined CRFs, while enforcing global consistency between systems in terms of constraints over the admissible solutions. These constraints are specifically realised as reparametrisations of the elementary CRFs until a consensus is reached. Since we deal with several annotation schemes, we will use predicates to abstract from them:

- $b(l)$ is true if l starts with `B`;
- $i(l)$ is true if l starts with `I`;
- $bi(i, y)$ is true if $b(y_{i-1})$ and $i(y_i)$ are true;
- $ib(i, y)$ is true if $i(y_{i-1})$ and $b(y_i)$ are true.

For a labelling y, we define 2 boolean vectors that indicate where the compounds begin and end:

[2] We also include two additional nodes: an initial state `(<s>,START)` and a final state `(</s>, STOP)`.

[3] Algorithm 1 calculates the score and backpointers must be added to retrieve the corresponding path.

[4] This algorithm takes as input 2 vectors that will be used for DD and will be explained in § 4.2. They can be ignored now.

- $D(y)$, such that $D(y)[i] = 1$ if $bi(i,y)$, and 0 otherwise;
- $F(y)$, such that $F(y)[i] = 1$ if $ib(i,y)$, and 0 otherwise.

As we want to *combine* CRFs, the solution of our system will be a *tuple* of label sequences with the same compound segmentation. For an input sequence x, this new maximisation problem is:

$$(P): \qquad \textit{find} \max_{(y^1,\dots,y^n)} \sum_{c=1}^{n} \sigma_c(y^c) = \sum_{c=1}^{n} \sum_{p\in\mathcal{P}(x)} w_c \cdot f_p^c(x, y_p^c) \tag{3}$$

$$\textit{s.t.} \ \exists u_1, u_2 \, \forall c \in [\![1,n]\!], \ D(y^c) = u_1, F(y^c) = u_2 \tag{4}$$

Objective (3) indicates that we seek for a tuple for which the sum of the scores of its elements is maximal. Constraints (4) imply that the compound frontiers – transitions `B` to `I` and `I` to `B` – must be the same for each element of the tuple. There are several ways to tackle this problem. The first one is by swapping the sum signs in (3) and noticing that the problem could then be represented by a joint system relying on dynamic programming – a CRF for which labels would be elements of the product $\mathcal{L} = \mathcal{L}_1 \times \cdots \times \mathcal{L}_n$ – and for which it is straightforward to define a weight vector and feature functions. We can therefore reuse the Viterbi algorithm but the complexity is quadratic in the size of $\mathcal{L}$, which is impractical[5].

In any case, this approach would be inadequate for inclusion of parsers, and we therefore rely on lagrangian relaxation. We modify the objective by introducing Lagrange multipliers, two real vectors Λ_c^{BI} and Λ_c^{IB} indexed by bigram places[6] for each CRF c of the combination. We obtain a new problem with the same solution as the previous one, since constraints (4) are garanteed to be satisfied at optimality:

$$\max_{(y^1,\dots,y^n,u_1,u_2)} \min_{(\Lambda^{BI},\Lambda^{IB})} \sum_{c=1}^{n} \sigma_c(y^c) - \sum_{c=1}^{n} \left[(D(y^c) - u_1) \cdot \Lambda_c^{BI} + (F(y^c) - u_2) \cdot \Lambda_c^{IB} \right] \tag{5}$$

The next step is dualisation, which gives an upper bound of our problem:

$$\min_{(\Lambda^{BI},\Lambda^{IB})} \max_{(y^1,\dots,y^n,u_1,u_2)} \sum_{c=1}^{n} \sigma_c(y^c) - \sum_{c=1}^{n} D(y^c) \cdot \Lambda_c^{BI} + u_1 \sum_{c=1}^{n} \Lambda_c^{BI} - \sum_{c=1}^{n} F(y^c) \cdot \Lambda_c^{IB} + u_2 \sum_{c=1}^{n} \Lambda_c^{IB} \tag{6}$$

We then remark that $\sum_{c=1}^{n} \Lambda_c^{BI}$ and $\sum_{c=1}^{n} \Lambda_c^{IB}$ must be zeros at optimum (if the problem is feasible).[7] It is convenient to convert this remark to hard constraints in order to remove any reference to vectors u_i – and therefore to the coupling contraints – from the objective. We obtain the constrained problem with the same optimal solution :

$$(Du): \qquad \textit{find} \min_{(\Lambda^{BI},\Lambda^{IB})} \sum_{c=1}^{n} \max_{y^c} \left[\sigma_c(y^c) - D(y^c) \cdot \Lambda_c^{BI} - F(y^c) \cdot \Lambda_c^{IB} \right] \tag{7}$$

$$\textit{s.t.} \ \sum_{c=1}^{n} \Lambda_c^{BI} = 0 \text{ and } \sum_{c=1}^{n} \Lambda_c^{IB} = 0 \tag{8}$$

In order to solve (Du) we use the projected subgradient descent method that has already been used in many problems, for example MRF decoding (Komodakis et al., 2007). For the problem at hand, this method gives Algorithm 2. This iterative algorithm consists in reparametrising the elementary CRFs of the system, by modifying the weights associated with the bigram features in places that correspond to compound frontiers, penalising them on CRFs that diverge from the average solution. This is performed

[5]One could object that some combinations are forbidden. It remains that the number of labels still grows exponentially.
[6]Bigram places are identified by their second position.
[7]Otherwise the sum expressions would be unbounded and their maximum is $+\infty$ for an appropriate value of u_i.

by amending the vectors Λ_c^{BI} and Λ_c^{IB} that are updated at each iteration proportionally to the difference between the feature vectors for c and the average values of these vectors. Hence the farther a solution is from the consensus, the more penalised it gets at the next iteration. This algorithm stops when the updates are null for all CRFs: in this case the consensus is reached.

Algorithm 2 Best segmentation with combined CRF system via subgradient descent

```
Require: $n$ CRF $c = (\Sigma, \mathcal{L}_c, w_c, \{f_p^c\}_p)$, an input sentence $x$, a maximum number of iterations $\tau$, stepsizes $\{\alpha_t\}_{0\leq t\leq\tau}$
 1: for all CRF $c$, bigram end position $i$, bigram label pair $(l, m)$ do
 2:     $\Lambda_c^{BI}[i,l,m]^{(0)} = 0$;     $\Lambda_c^{IB}[i,l,m]^{(0)} = 0$
 3: end for
 4: for $t = 0 \to \tau$ do
 5:     for all CRF $c$ do
 6:         $y^{c^{(t)}} = Viterbi(\mathcal{G}_c, w_c, f_c, \Lambda_c^{BI^{(t)}}, \Lambda_c^{IB^{(t)}})$
 7:     end for
 8:     for all CRF $c$ do
 9:         $\Delta_c^{BI^{(t)}} \leftarrow \alpha_t \left( D\left(y^{c^{(t)}}\right) - \frac{\sum_{1\leq d\leq n} D\left(y^{d^{(t)}}\right)}{n} \right)$;     $\Delta_c^{IB^{(t)}} \leftarrow \alpha_t \left( F\left(y^{c^{(t)}}\right) - \frac{\sum_{1\leq d\leq n} F\left(y^{d^{(t)}}\right)}{n} \right)$
10:         $\Lambda_c^{BI^{(t+1)}} \leftarrow \Lambda_c^{BI^{(t)}} + \Delta_c^{BI^{(t)}}$;     $\Lambda_c^{IB^{(t+1)}} \leftarrow \Lambda_c^{IB^{(t)}} + \Delta_c^{IB^{(t)}}$
11:     end for
12:     if $\Delta_c^{BI^{(t)}} = 0$ and $\Delta_c^{IB^{(t)}} = 0$ for all $c$ then
13:         Exit loop
14:     end if
15: end for
16: return $(y^{1^{(t)}}, \cdots, y^{n^{(t)}})$
```

We set the maximum number of iteration τ to 1000. For the step size, we use a common heuristic: $\alpha_t = \frac{1}{1+k}$ where k is the number of times that (Du) has increased between two successive iterations.

4.3 Experimental results for CRF combinations

We modified the `wapiti` software (Lavergne et al., 2010) with Algorithm 2. Table 1 reports segmentation results on the development set with the different tag sets, the best DD combination, and the best voting system.[8]

Tag Set CRF / combination	Recall	Precision	F-score
partial-internal	79.59	85.49	82.44
partial	78.98	85.57	82.14
basic	79.74	84.65	82.12
complete	79.69	83.10	81.36
complete-internal	79.03	82.66	80.80
`MWE` basic complete partial-internal	80.82	86.07	**83.36**
vote (basic complete partial-internal)	80.49	85.46	82.90

Table 1: Segmentation scores of CRF systems (dev)

System	F-score (all)	F-score (compounds)
complete	94.29	78.32
`MWE`	94.59	**80.00**

Table 2: Segmentation + POS tagging (dev)

We see that the best system is a combination of 3 CRFs (tag sets *basic*, *complete* and *partial-internal*) with DD, that we call `MWE` in the remaining of the paper. The subgradient descent converges on all instances in 2.14 iterations on average. The DD combination is better than the voting system.

We can also evaluate the POS tagging accuracy of the system for systems including the *complete* tag set. We compare the results of the *complete* CRF with the `MWE` combination on Table 2. The second column gives the F-score of the complete task, segmentation and POS tagging. The third column restricts the evaluation to compounds. Again, the `MWE` combination outperforms the single system.

[8]Each system has one vote and in case of a draw, we pick the best system's decision.

In some preliminary experiments, the weights of the CRF systems were based on unigram features mainly – i.e. those described in (Constant et al., 2012). As our CRFs are constrained on transitions from `B` to `I` and `I` to `B`, penalising systems resulted in modifying (low) bigram weights and had only a minor effect on the predictions and consequently the projected gradient algorithm was slow to converge. We therefore added bigram templates for some selected unigram templates, so that our system can converge in a reasonable time. Adding these bigram features resulted in slower elementary CRFs. On average the enriched CRFs were 1.8 times slower that their preliminary counterparts.

5 Dual Decomposition to combine parsers and sequence labellers

We now present an extension of the previous method to incorporate phrase-structure parsers in the combination. Our approach relies on the following requirement for the systems to agree: if the parser predicts a compound between positions i and j, then the CRFs must predict compound frontiers at the same positions. In this definition, like in previous CRF combinations, only the positions are taken into account, not the grammatical categories. From a parse tree a, we define two feature vectors:

- $D(a)$, such that $D(a)[i] = 1$ if a contains a subtree for a compound starting at position $i-1$
- $F(a)$, such that $F(a)[i] = 1$ if a contains a subtree for a compound ending at position $i-1$

In other words, $D(a)[i]$ indicates whether the CRFs should label position $i-1$ with `B` and position i with `I`, while $F(a)[i]$ indicates whether the CRFs should label position $i-1$ with `I` and position i with `B`. See Figure 2 for an example.

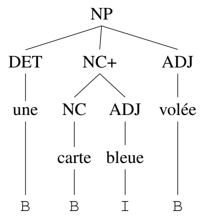

Figure 2: Parser and CRF alignments (A stolen credit card)

5.1 Parsing with probabilistic context-free grammars

We follow the type of reasoning we used in § 4.2. With a PCFG g, we can define the score of a parse for an input sentence x as the logarithm of the probability assigned to this parse by g. Finding the best parse takes a form analogous to the one in Equation 2, and we can write the CKY algorithm as a best path algorithm with penalties on nodes, as we did for the Viterbi algorithm previously. This is closely related to the PCFG combinations of (Le Roux et al., 2013). We introduce penalties through two real vectors Λ^{BI} and Λ^{IB} indexed by compound positions. The modified CKY is presented in Algorithm 3[9] where the parse forest $\mathcal{F}$ is assumed to be already available and we note w the vector of rule log-probabilities.

5.2 System combination

As in § 4.2, our problem amounts to finding a tuple that now consists of a parse tree and several labellings. All systems must agree on compound frontiers. Our objective is:

$$(P') : \qquad \textit{find} \max_{(a,y^1,\dots,y^n)} \sigma_p(a) + \delta \sum_{c=1}^{n} \sigma_c(y^c) \tag{9}$$

$$\textit{s.t.}\ \exists u_1, u_2\, \forall c \in [\![1,n]\!],\ D(y^c) = u_1, F(y^c) = u_2, D(a) = u_1, F(a) = u_2 \tag{10}$$

[9]Without loss of generality, only binary rules are taken into account.

Algorithm 3 CKY with node penalties for compound start/end positions

```
1: CKY($\mathcal{F}, w, \Lambda^{BI}, \Lambda^{IB}$):
2: for all node $v$ in the forest $\mathcal{F}$ do
3:     $\pi[v] = -\infty$
4: end for
5: for all leaf node $x$ do
6:     $\pi[x] = 0$
7: end for
8: for all non-terminal node $(A, i, j)$ in topological order do
9:     for all incoming hyperedge $u = (B, i, k)(C, k+1, j) \to (A, i, j)$ do
10:        $s \leftarrow \pi[(B, i, k)] + \pi[(C, k+1, j)] + w_{A \to BC}$        ▷ $w_{A \to BC}$ is the score for rule $A \to BC$
11:        if $A$ is a compound label then
12:            $s \leftarrow s - \Lambda^{BI}[i] - \Lambda^{IB}[j+1]$
13:        end if
14:        if $s > \pi[(A, i, j)]$ then
15:            $\pi[(A, i, j)] \leftarrow s$
16:        end if
17:    end for
18: end for
19: return hyperpath with score $\pi[(\text{ROOT}, 1, n)]$
```

We use δ to set the relative weights of the CRFs and the PCFG. It will be tuned on the development set. We then reuse the same procedure as before: lagrangian relaxation, dualisation, and projected subgradient descent. Algorithm 4 presents the function we derive from these operations.

Algorithm 4 Find the best segmentation with a PCFG and CRFs

```
Require: a PCFG parser $p$, $n$ CRFs, an input sentence $x$, a bound $\tau$
1: set Lagrange multipliers (penalty vectors) to zero
2: for $t = 0 \to \tau$ do
3:     for all CRF $c$ do
4:         $y^{c^{(t)}} \leftarrow Viterbi(\mathcal{G}_c, w_c, f_c, \Lambda_c^{BI^{(t)}}, \Lambda_c^{IB^{(t)}})$
5:     end for
6:     $a^{(t)} \leftarrow CKY(\mathcal{F}, w, \Lambda_p^{BI^{(t)}}, \Lambda_p^{IB^{(t)}})$
7:     for all CRF $c$ and parser $p$ do
8:         Update penalty vectors proportionally to the difference between corresponding solution and average solution
9:     end for
10:    if update is null for all $c$ and $p$ then
11:        Exit loop
12:    end if
13: end for
14: return $(a^{(t)}, y^{1^{(t)}}, \cdots, y^{n^{(t)}})$
```

Algorithm 4 follows the method used in § 4 and simply adds the PCFG parser as another subsystem. This method can then be extended further: for instance, we can add a POS tagger (Rush et al., 2010) or multiple PCFG parsers (Le Roux et al., 2013). Due to lack of space, we omit the presentation of these systems, but we experiment with them in the following section.

6 Experiments

For this series of experiments, we used `wapiti` as in § 4.3 and the `LORG` PCFG-LA parser in the configuration presented in (Le Roux et al., 2013) that we modified by implementing Algorithm 4. This parser already implements a combination of parsers based on DD, a very competitive baseline.

For parse evaluation, we used the EVALB tool, modified by the SPMRL 2013 organisers, in order to compare our results with the shared task participants. We evaluated several configurations: (i) the `LORG` parser alone, a combination of 4 PCFG-LA parsers as in (Le Roux et al., 2013), (ii) a pipeline of `POS`, a CRF-based POS tagger, and `LORG`, (iii) joint `LORG` and `POS`, using DD as in (Rush et al., 2010), (iv) joint `LORG` and `MWE` (our best CRF combination for compound segmentation) using DD, and (v) joint `LORG`, `POS` et `MWE` using DD. We also compare these architectures with 2 additional pipelines, in which we first run `MWE` and then merge compounds as single tokens. The converted sentences are then sent to a version of `LORG` learnt on this type of corpus. After parsing, compounds are *unmerged*, replaced with the corresponding subtree. In one of these two architectures, we add a POS tagger.

The evaluations for the parsing task of all these configurations are summarised in Table 3. The best system is the DD joint system combining the POS tagger, the parser and the compound recognisers.

System	Recall	Precision	Fscore	EX	Tag
`LORG`	82.01	82.37	82.19	18.06	97.35
pipeline `POS` → `LORG`	82.36	82.59	82.47	19.22	97.73
DD `POS` + `LORG`	82.48	82.73	82.61	19.19	97.84
DD `MWE` + `LORG`	82.91	83.07	82.99	19.19	97.41
DD `POS` + `MWE` + `LORG`	83.38	83.42	**83.40**	20.73	97.85
pipeline `MWE MERGE` → `LORG` → `UNMERGE`	82.56	82.63	82.59	18.79	97.39
pipeline `MWE MERGE/POS` → `LORG` → `UNMERGE`	82.73	82.64	82.69	20.02	97.57

Table 3: Parse evaluation on *dev* set (recall, precision and F-score, exactness and POS tagging).

Table 4 shows evaluation results of our best system and comparisons with baseline or alternative configurations on the SPMRL 2013 *test* set.

Parsing The DD method performs better than our baseline, and better than the best system in the SPMRL 2013 shared task (Björkelund et al., 2013). This system is a pipeline consisting of a morpho-syntactic tagger with a very rich and informative tag set, a product of PCFG-LAs, and a parse reranker. Although this approach is quite different from ours, we believe our system is more accurate overall because our method is more resilient to an error from one of its components.

Compound recognition and labelling For the task of recognition alone, where only the frontiers are evaluated, the DD combinations of CRFs performs better than the best single CRF which itself performs better than the parser alone, but the complete architecture is again the best system. If we also evaluate compound POS tags, we get similar results. The DD combination is always beneficial.

System	Recall	Precision	Fscore	EX	Tag
`LORG`	82.79	83.06	82.92	22.00	97.39
(Björkelund et al., 2013)	–	–	82.86	–	–
DD `POS` + `MWE` + `LORG`	83.74	83.85	**83.80**	23.81	97.87
compound recognition `LORG`	78.03	78.63	78.49	–	–
compound recognition best single CRF (partial-internal)	78.27	82.84	80.49	–	–
compound recognition `MWE`	79.68	83.50	81.54	–	–
compound recognition DD `POS` + `MWE` + `LORG`	80.76	84.19	**82.44**	–	–
compound recognition + POS tagging `LORG`	75.43	75.71	75.57	–	–
compound recognition + POS tagging `MWE`	76.49	80.10	78.28	–	–
compound recognition + POS tagging DD `POS` + `MWE` + `LORG`	77.92	81.23	**79.54**	–	–

Table 4: Evaluation on SPMRL 2013 *test* set: parsing (first 3 lines), and compound recognition.

7 Conclusion

We have presented an original architecture for the joint task of syntactic parsing and compound recognition. We first introduced a combination of recognisers based on linear chain CRFs, and a second system that adds in a phrase-structure parser. Our experimental prototype improves state-of-the-art on the French SPMRL corpus.

In order to derive decoding algorithms for these joint systems, we used dual decomposition. This approach, leading to simple and efficient algorithms, can be extended further to incorporate additional components. As opposed to pipeline approaches, a component prediction can be *corrected* if its solution is too far from the general consensus. As opposed to joint systems relying on pure dynamic programming to build a complex single system, the search space does not grow exponentially, so we can avoid using pruning heuristics such as beam search. The price to pay is an iterative algorithm.

Finally, this work paves the way towards component-based NLP software systems that perform complex processing based on consensus between components, as opposed to previous pipelined approaches.

Acknowledgements

We would like to thank Nadi Tomeh and Davide Buscaldi for their feedback on an early draft of this paper, as well as the three anonymous reviewers for their helpful comments. This work is supported by a public grant overseen by the French National Research Agency (ANR) as part of the *Investissements d'Avenir* program (ANR-10-LABX-0083).

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In Anne Abeillé, editor, *Treebanks*. Kluwer, Dordrecht.

Abhishek Arun and Frank Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of French. In *Proceedings of the Annual Meeting of the Association For Computational Linguistics (ACL'05)*, pages 306–313.

Anders Björkelund, Richárd Farkas, Thomas Müller, and Wolfgang Seeker. 2013. (re) ranking meets morphosyntax: State-of-the-art results from the spmrl 2013 shared task. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages: Shared Task*.

Matthieu Constant and Isabelle Tellier. 2012. Evaluating the impact of external lexical resources into a CRF-based multiword segmenter and part-of-speech tagger. In *Proceedings of the 8th conference on Language Resources and Evaluation*.

Matthieu Constant, Anthony Sigogne, and Patrick Watrin. 2012. Discriminative strategies to integrate multiword expression recognition and parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL'12)*, pages 204–212.

Matthieu Constant, Marie Candito, and Djamé Seddah. 2013a. The LIGM-Alpage Architecture for the SPMRL 2013 Shared Task: Multiword Expression Analysis and Dependency Parsing. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages: Shared Task*, Seattle, WA.

Matthieu Constant, Joseph Le Roux, and Anthony Sigogne. 2013b. Combining compound recognition and PCFG-LA parsing with word lattices and conditional random fields. *ACM Transaction in Speech and Language Processing*, 10(3).

Blandine Courtois, Mylène Garrigues, Gaston Gross, Maurice Gross, René Jung, Michel Mathieu-Colas, Anne Monceaux, Anne Poncet-Montange, Max Silberztein, and Robert Vivés. 1997. Dictionnaire électronique DELAC : les mots composés binaires. Technical Report 56, University Paris 7, LADL.

Jenny Rose Finkel and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 326–334, Stroudsburg, PA, USA. Association for Computational Linguistics.

Spence Green, Marie-Catherine de Marneffe, John Bauer, and Christopher D. Manning. 2011. Multiword expression identification with tree substitution grammars: A parsing tour de force with french. In *Proceedings of the conference on Empirical Method for Natural Language Processing (EMNLP'11)*, pages 725–735.

Spence Green, Marie-Catherine de Marneffe, and Christopher D Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics*, 39(1):195–227.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. 2007. MRF optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, pages 504–513.

Joseph Le Roux, Antoine Rozenknop, and Jennifer Foster. 2013. Combining PCFG-LA models with dual decomposition: A case study with function labels and binarization. In Association for Computational Linguistics, editor, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, October.

Slav Petrov. 2010. Products of random latent variable grammars. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, pages 19–27.

Odile Piton, Denis Maurel, and Claude Belleil. 1999. The Prolex Data Base : Toponyms and gentiles for NLP. In *Proceedings of the Third International Workshop on Applications of Natural Language to Data Bases (NLDB'99)*, pages 233–237.

Alexander Rush, David Sontag, Michael Collins, and Tommi Jaakola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In ACL, editor, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Benoît Sagot. 2010. The lefff, a freely available, accurate and large-coverage lexicon for french. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*.

Djamé Seddah, Reut Tsarfaty, Sandra K´ubler, Marie Candito, Jinho Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiorkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clérgerie. 2013. Overview of the spmrl 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages*, Seattle, WA.

Veronica Vincze, István Nagy, and Gábor Berend. 2011. Multiword expressions and named entities in the wiki50 corpus. In *Proceedings of the conference on Recent Advances in Natural Language Processing (RANLP'11)*, pages 289–295.