Optimization in Coreference Resolution Is Not Needed: A Nearly-Optimal Algorithm with Intensional Constraints

Manfred Klenner & Étienne Ailloud
Computational Linguistics
Zurich University, Switzerland
{klenner, ailloud}@cl.uzh.ch

Abstract

We show how global constraints such as transitivity can be treated intensionally in a Zero-One Integer Linear Programming (ILP) framework which is geared to find the optimal and coherent partition of coreference sets given a number of candidate pairs and their weights delivered by a pairwise classifier (used as reliable clustering seed pairs). In order to find out whether ILP optimization, which is NP-complete, actually is the best we can do, we compared the first consistent solution generated by our adaptation of an efficient Zero-One algorithm with the optimal solution. The first consistent solution, which often can be found very fast, is already as good as the optimal solution; optimization is thus not needed.

1 Introduction

One of the main advantages of Integer Linear Programming (ILP) applied to NLP problems is that prescriptive linguistic knowledge can be used to pose global restrictions on the set of desirable solutions. ILP tries to find an optimal solution while adhering to the global constraints. One of the central global constraints in the field of coreference resolution evolves from the interplay of intra-sentential binding constraints and the transitivity of the anaphoric relation. Consider the following sentence taken from the Internet: 'He told him that he deeply admired him'. 'He' and 'him' are exclusive (i.e. they could never be coreferent) within their clauses (the main and the subordinate clause, respectively). A pairwise classifier could learn this given appropriate features or, alternatively, binding constraints could act as a hard filter preventing such pairs from being generated at all. But in either case, since pairwise classification is trapped in its local perspective, nothing can prevent the classifier to resolve the 'he' and 'him' from the subordinate clause in two independently carried out steps to the same antecedent from the main clause. It is transitivity that prohibits such an assignment: if two elements are both coreferent to a common third element, then the two are (transitively given) coreferent as well. If they are known to be exclusive, such an assignment is disallowed. But transitivity is beyond the scope of pairwise classification—it is a global phenomena. The solution is to take ILP as a clustering device, where the probabilities of the pairwise classifier are interpreted as weights and transitivity and other restrictions are acting as global constraints.

Unfortunately, in an ILP program every constraint has to be extensionalized (i.e. all instantiations of the constraint are to be generated). Capturing transitivity for e.g. 150 noun phrases (about 30 sentences) already produces 1,500,000 equations (cf. Section 4). Solving such ILP programs is far too slow for real applications (let alone its brute force character).

A closer look at existing ILP approaches to NLP reveals that they are of a special kind, namely Zero-One ILP with unweighted constraints. Although still NP-complete there exist a number of algorithms such as the Balas algorithm (Balas, 1965) that efficiently explore the search space and reduce thereby run time complexity in the mean. We have adapted Balas' algorithm to the special needs of coreference resolution. First and foremost, this results in an optimization algorithm that treats global constraints intensionally, i.e. that generates instantiations of a constraint only on demand. Thus, transitivity can be captured for even the longest texts. But more important, we found out empirically that 'full optimization' is not really needed. The first consistent solution, which often can be found very fast, is already as good—in terms of F-measure values—as the optimal solution. This is good news, since it reduces runtime and at same time maintains the empirical results.

We first introduce Zero-One ILP, discuss our baseline model and give an ILP formalization of coreference resolution. Then we go into the details of our Balas adaptation and provide empirical evidence for our central claim—that optimization search can already be stopped (without quality loss) when the first consistent solution has been found.

2 Zero-One Integer Linear Programming (ILP)

The algorithm in (Balas, 1965) solves Zero-One Integer Linear Programming (ILP), where a weighted linear function (the *objective function*) of binary variables $F(x_1,\ldots,x_n) = w_1x_1 + \ldots + w_nx_n$ is to be minimized under the regiment of linear inequalities $a_1x_1 + \ldots + a_nx_n \geq A$.[1] Unlike its real-valued counterpart, Zero-One ILP is NP-complete (cf., say, (Papadimitriou and Steiglitz, 1998)), but branch-and-bound algorithms with efficient heuristics exist, as the Balas Algorithm: Balas (1965) proposes an approach where the objective function's addends are sorted according to the magnitude of the weights: $0 \leq w_1 \leq \ldots \leq w_n$. This preliminary ordering induces the following functioning principles for the algorithm (see (Chinneck, 2004, Chap. 13) for more details):

1. It seeks to minimize F, so that a solution with as few 1s as possible is preferred.

2. If, during exploration of solutions, constraints force an x_i to be set to 1, then it should bear as small an index as possible.

The Balas algorithm follows a depth-first search while checking feasibility (i.e., through the constraints) of the branches partially explored: Upon branching, the algorithm *bounds* the cost of setting the current variable x_N to 1 by the costs accumulated so far: $w_1x_1 + \ldots + w_{N-1}x_{N-1} + w_N$ is now the lowest cost this branch may yield. If, on the contrary, x_N is set to 0, a violated $\geq$-constraint may only be satisfied via an x_i set to 1 ($i > N$), so the cheapest change to ameliorate the partial solution is to set the right-next variable to 1: $w_1x_1 + \ldots + w_{N-1}x_{N-1} + w_{N+1}$ would be the cheapest through this branch.

If setting all weights past the branching variable to 0 yields a cheaper solution than the so far minimal solution obtained, then it is worthwile exploring this branch, and the algorithms goes on to the next weighted variable, until it reaches a feasible solution; otherwise it backtracks to the last unexplored branching. The complexity thus remains exponential in the worst case, but the initial ordering of weights is a clever guide.

[1]Maximization and coping with $\leq$-constraints are also accessible via simple transformations.

3 Our Baseline Model

The memory-based learner TiMBL (Daelemans et al., 2004) is used as a (pairwise) classifier. TiMBL stores all training examples, learns feature weights and classifies test instances according to the majority class of the k-nearest (i.e. most similar) neighbors. We have experimented with various features; Table 1 lists the set we have finally used (Soon et al. (2001) and Ng and Cardie (2002) more thoroughly discuss different features and their benefits):

- distance in sentences and markables
- part of speech of the head of the markables
- the grammatical functions
- parallelism of grammatical functions
- do the heads match or not
- where is the pronoun (if any): left or right
- word form if POS is pronoun
- salience of the non-pronominal phrases
- semantic class of noun phrase heads

Table 1: Features for Pairwise Classification

As a gold standard the TüBa-D/Z (Telljohann et al., 2005; Naumann, 2006) coreference corpus is used. The TüBa is a treebank (1,100 German newspaper texts, 25,000 sentences) augmented with coreference annotations[2]. In total, there are 13,818 anaphoric, 1,031 cataphoric and 12,752 coreferential relations. There are 3,295 relative pronouns, 8,929 personal pronouns, 2,987 reflexive pronouns, and 3,921 possessive pronouns.

There are some rather long texts in the TüBa corpus. Which pair generation algorithm is reasonable? Should we pair every markable (even from the beginning of the text) with every other succeeding markable? This is linguistically implausible. Pronouns are acting as a kind of local variables. A 'he' at the beginning of a text and a second distant 'he' at the end of the text hardly tend to corefer, except if there is a long chain of coreference 'renewals' that lead somehow from the first 'he' to the second 'he'. But the plain 'he'-'he' pair does not reliably indicate coreference.

A smaller window seems to be appropriate. We have experimented with various window sizes and found that a size of 3 sentences worked best. Candidate pairs are generated only within that

[2]Recently, a new version of the TüBa was released with 35,000 sentences with coreference annotations.

window, which is moved sentence-wise over the whole text.

4 Our Constraint-Based Model

The output of the TiMBL classifier is the input to the optimization step, it provides the set of variables and their weights. In order to utilize TiMBL's classification results as weights in a minimization task, we have defined a measure called *classification costs* (see Fig. 1).

$$w_{ij} = \frac{\mid neg_{ij} \mid}{\mid neg_{ij} \cup \; pos_{ij} \mid}$$

Figure 1: Score for Classification Costs

$\mid neg_{ij} \mid$ ($\mid pos_{ij} \mid$) denotes the number of instances similar (according to TiMBL's metric) to $\langle i, j \rangle$ that are negative (positive) examples. If no negative instances are found, a safe positive classification decision is proposed at zero cost. Accordingly, the cost of a decision without any positive instances is high, namely one. If both sets are non-empty, the ratio of the negative instances to the total of all instances is taken. For example, if TiMBL finds 10 positive and 5 negative examples similar to the yet unclassified new example $\langle i, j \rangle$ the cost of a positive classification is 5/15 while a negative classification costs 10/15.

We introduce our model in an ILP style. In section 6 we discuss our Balas adaptation which allows us to define constraints intensionally.

The objective function is:

$$min: \sum_{\langle i,j \rangle \in \; O_{0.5}} w_{ij} \cdot c_{ij} + (1 - w_{ij}) \cdot c_{ji} \qquad (1)$$

$O_{0.5}$ is the set of pairs $\langle i, j \rangle$ that have received a weight ≤ 0.5 according to our weight function (see Fig. 1). Any binary variable c_{ij} combines the ith markable (of the text) with the jth markable ($i < j$) within a fixed window[3].

c_{ji} represents the (complementary) decision that i and j are not coreferent. The weight of this decision is $(1 - w_{ij})$. Please note that every optimization model of coreference resolution must include both variables[4]. Otherwise optimization would completely ignore the classification decisions of the pairwise classifier (i.e., that ≤ 0.5 suggests coreference). For example, the choice not to set $c_{ij} = 1$ at costs $w_{ij} \leq 0.5$ must be sanctioned by instantiating its inverse variable $c_{ji} = 1$ and adding $(1 - w_{ij})$ to the objective function's value. Otherwise minimization would turn—in the worst case—everything to be non-coreferent, while maximization would preferentially set everything to be actually coreferent (as long as no constraints are violated, of course).[5]

The first constraint then is:

$$c_{ij} + c_{ji} = 1, \;\; \forall \langle i, j \rangle \in O_{0.5} \qquad (2)$$

A pair $\langle i, j \rangle$ is either coreferent or not.

Transitivity is captured by (see (Finkel and Manning, 2008) for an alternative but equivalent formalization):

$$\begin{array}{l} c_{ij} + c_{jk} \leq c_{ik} + 1, \; \forall i,j,k \; (i < j < k) \\ c_{ik} + c_{jk} \leq c_{ij} + 1, \; \forall i,j,k \; (i < j < k) \\ c_{ij} + c_{ik} \leq c_{jk} + 1, \; \forall i,j,k \; (i < j < k) \end{array} \qquad (3)$$

In order to take full advantage of ILP's reasoning capacities, three equations are needed given three markables. The extensionalization of transitivity thus produces $\frac{n!}{3!(n-3)!} \cdot 3$ equations for n markables. Note that transitivity—as a global constraint—ought to spread over the whole candidate set, not just within in the window.

Transitivity without further constraints is pointless.[6] What we really can gain from transitivity is consistency at the linguistic level, namely (globally) adhering to exclusiveness constraints (cf. the example in the introduction). We have defined two predicates that replace the traditional c-command (which requires full syntactical analysis) and approximate it: *clause_bound* and *np_bound*.

Two mentions are *clause-bound* if they occur in the same subclause, none of them being a reflexive or a possessive pronoun, and they do not form an apposition. There are only 16 cases in our data set where this predicate produces false negatives (e.g. in clauses with predicative verbs: 'He$_i$ is still prime minister$_i$'). We currently regard this shortcoming as noise.

[3]As already discussed, the window is realized as part of the vector generation component, so $O_{0.5}$ automatically only captures pairs within the window.

[4]Even if an anaphoricity classifier is used.

[5]The need for optimization or other numerical preference mechanisms originates from the fact that coreference resolution is underconstrained—due to the lack of a deeper text understanding.

[6]Although it might lead to a reordering of coreference sets by better 'balancing the weights'.

Two markables that are clause-bound (in the sense defined above) are exclusive, i.e.

$$c_{ij} = 0,\ \forall i, j\ (clause\_bound(i, j)). \qquad (4)$$

A possessive pronoun is exclusive to all markables in the noun phrase it is contained in (e.g. $c_{ij} = 0$ given a noun phrase "[her$_i$ manager$_j$]"), but might get coindexed with markables outside of such a local context ("Anne$_i$ talks to her$_i$ manager"). We define a predicate *np_bound* that is true of two markables, if they occur in the same noun phrase. In general, two markables that *np-bind* each other are exclusive:

$$c_{ij} = 0,\ \forall i, j\ (np\_bound(i, j)) \qquad (5)$$

5 Representing ILP Constraints Intensionally

Existing ILP-based approaches to NLP (e.g. (Punyakanok et al., 2004; Althaus et al., 2004; Marciniak and Strube, 2005)) belong to the class of Zero-One ILP: only binary variables are needed. This has been seldom remarked (but see (Althaus et al., 2004)) and generic (out-of-the-box) ILP implementations are used. Moreover, these models form a very restricted variant of Zero-One ILP: the constraints come without any weights. The reason for this lies in the logical nature of NLP constraints. For example in the case of coreference, we have the following types of constraints:

1. exclusivity of two instantiations (e.g. either coreferent or not, equation 2)
2. dependencies among three instantiations (transitivity: if two are coreferent then so the third, equation 3)
3. the prohibition of pair instantiation (binding constraints, equations 4 and 5)
4. enforcement of at least one instantiation of a markable in some pair (equation 6 below).

We call the last type of constraints 'boundness enforcement constraints'. Only two classes of pronouns strictly belong to this class: relative (POS label 'PRELS') and possessive pronouns (POS label 'PPOSAT')[7]. The corresponding ILP constraint is, e.g. for possessive pronouns:

$$\sum_i c_{ij} \geq 1,\ \forall j\ s.t.\ pos(j) = '\text{PPOSAT}' \qquad (6)$$

[7]In rare cases, even reflexive pronouns are (correctly) used non-anaphorically, and, more surprisingly, 15% of the personal pronouns in the TüBa are used non-anaphorically.

Note that boundness enforcement constraints lead to exponential time in the worst case. Given that such a constraint holds on a pair with the highest costs of all pairs (thus being the last element of the Balas ordered list with *n* elements): in order to prove whether it can be bound (set to one), 2^n (binary) variable flips need to be checked in the worst case. All other constraints can be satisfied by setting some $c_{ij} = 0$ (i.e. non-coreferent) which does not affect already taken or (any) yet to be taken assignments. Although exponential in the worst case, the integration of constraint (6) has slowed down CPU time only slightly in our experiments.

A closer look at these constraints reveals that most of them can be treated intensionally in an efficient manner. This is a big advantage, since now transitivity can be captured even for long texts (which is infeasible for most generic ILP models).

To intensionally capture transitivity, we only need to explicitly maintain the evolving coreference sets. If a new markable is about to enter a set (e.g. if it is related to another markable that is already member of the set) it is verified that it is compatible with all members of the set.

A markable *i* is *compatible* with a coreference set if, for all members *j* of the set, $\langle i, j\rangle$ does not violate binding constraints, agrees morphologically and semantically. Morphological agreement depends on the POS tags of a pair. Two personal pronouns must agree in person, number and gender. In German, a possessive pronoun must only agree in person with its antecedent. Two nouns might even have different grammatical gender, so no morphological agreement is checked here.

Checking binding for the *clause_bound* constraint is simple: each markable has a subclause ID attached (extracted from the TüBa). If two markables (except reflexive or possessive pronouns) share an ID they are exclusive. Possessive pronouns must not be np-bound. All members of the noun phrase containing the possessive pronoun are exclusive to it.

Note that such a representation of constraints is intensional since we need not enumerate all exclusive pairs as an ILP approach would have to. We simply check (on demand) the identity of IDs.

There is also no need to explicitly maintain constraint (2), either, which states that a pair is either coreferent or not. In the case that a pair cannot be set to 1 (representing coreference), it is set to 0; i.e. c_{ij} and c_{ji} are represented by the same index position p of a Balas solution v (cf. Section 6); no extensional modelling is necessary.

Although our special-purpose Balas adaptation no longer constitutes a general framework that can be fed with each and every Zero-One ILP formalization around, the algorithm is simple enough to justify this. Even if one uses an ILP translator such as Zimpl[8], writing a program for a concrete ILP problem quickly becomes comparably complex.

6 A Variant of the Balas Algorithm

Our algorithm proceeds as follows: we generate the first consistent solution according to the Balas algorithm (Balas-First, henceforth). The result is a vector v of dimension n, where n is the size of $O_{0.5}$. The dimensions take binary values: a value 1 at position p represents the decision that the pth pair c_{ij} from the (Balas-ordered) objective function is coreferent (0 indicates non-coreference). One minor difference to the original Balas algorithm is that the primary choice of our algorithm is to set a variable to 1, not to 0—thus favoring coreference. However, in our case, 1 is the cheapest solution (with cost $w_{ij} \leq 0.5$). Setting a variable to zero has cost $1 - w_{ij}$ which is more expensive in any case. But aside from this assignment convention, the principal idea is preserved, namely that the assignment is guided by lowest cost decisions.

The search for less expensive solutions is done a bit differently from the original. The Balas algorithm takes profit from weighted constraints. As discussed in Section 5, constraints in existing ILP models for NLP are unweighted. Another difference is that in the case of coreference resolution both decisions have costs: setting a variable to 1 (w_{ij}) and setting it to 0 ($1 - w_{ij}$). This is the key to our cost function that guides the search.

Let us first make some properties of the search space explicit. First of all, given no constraints were violated, the optimal solution would be the one with all pairs from $O_{0.5}$ set to 1 (since any 0 would add a suboptimal weight, namely $1 - w_{ij}$). Now we can see that any less expensive solution than Balas-First must be longer than Balas-First, where the length (1-length, henceforth) of a Balas solution is defined as the number of dimensions with value 1. A shorter solution would turn at least a single 1 into 0, which leads to a higher objective function value.

[8]`http://zimpl.zib.de/`

Any solution with the same 1-length is more expensive since it requires swapping a 1 to 0 at one position and a 0 to 1 at a farther position. The permutation of 1/0s from Balas-First is induced by the weights and the constraints. A 0 at position q is forced by (a constraint together with) some (or more) 1 at position p ($p < q$). Thus, we can only swap a 0 to 1 if we swap at least one preceding 1 to 0. The costs of swapping a preceding 1 to 0 are higher than the gain from swapping the 0 to 1 (as a consequence of the Balas ordering). So no solution with the same 1-length can be less expensive than Balas-First.

We then have to search for solutions with higher 1-length. In Section 7 we will argue that this actually goes in the wrong direction.

Any longer solution must swap—for every 1 swapped to 0—at least two 0s to 1. Otherwise the costs are higher than the gain. We can utilize this for a reduction of the search space.

Let p be a position index of Balas-First (v), where the value of the dimension at p is 1 and there exist at least two 0s with position indices $q > p$.

Consider $v = \langle 1,0,1,1,0,0 \rangle$. Positions 1, 3 and 4 are such positions (identifying the following parts of v resp.: $\langle 1,0,1,1,0,0 \rangle, \langle 1,1,0,0 \rangle$ and $\langle 1,0,0 \rangle$).

We define a projection $c(p)$ that returns the weight w_{ij} of the pth pair c_{ij} from the Balas ordering. $v(p)$ is the value of dimension p in v (0 or 1). The cost of swapping 1 at position p to 0 is the difference between the cost of c_{ji} $(1 - c(p))$ and c_{ij} $(c(p))$: $costs(p) = 1 - 2 \cdot c(p)$.

We define the potential gain $pg(p)$ of swapping a 1 at position p to 0 and every succeeding 0 to 1 by:

$$pg(p) = costs(p) - \sum_{q>p \text{ s.t. } v(q)=0} 1 - 2 \cdot c(q) \quad (7)$$

For example, let $v = \langle 1,0,1,1,0,0 \rangle$, $p = 4$, $c(4) = 0.2$ and (the two 0s) $c(5) = 0.3$, $c(6) = 0.35$. $costs(4) = 1 - 0.4 = 0.6$ and $pg(4) = 0.6 - (0.4 + 0.3) = -0.1$. Even if all 0s (after position 4) can be swapped to 1, the objective function value is lower that before, namely by 0.1. Thus, we need not consider this branch.

In general, each time a 0 is turned into 1, the potential gain is preserved, but if we have to turn another 1 to 0 (due to a constraint), or if a 0 cannot be swapped to 1, the potential gain is decremented by a certain cost factor. If the potential gain is exhausted that way, we can stop searching.

7 Is Optimization Really Needed? Empirical Evidence

The first observation we made when running our algorithm was that in more than 90% of all cases, Balas-First already constitutes the optimal solution. That is, the time-consuming search for a less expensive solution ended without further success.

As discussed in Section 6, any less expensive solution must be longer (1-length) than Balas-First. But can longer solutions be better (in terms of F-measure scores) than shorter ones? They might: if the 1-length re-assignment of variables removes as much false positives as possible and raises instead as much of the true positives as can be found in $O_{0.5}$. Such a solution might have a better F-measure score. But what about its objective function value? Is it less expensive than Balas-First?

We have designed an experiment with all (true) coreferent pairs from $O_{0.5}$ (as indicated by the gold standard) set to 1. Note that this is just another kind of constraints: the enforcement of coreference (this time extensionally given).

The result was surprising: The objective function values that our algorithm finds under these constraints were in any case higher than Balas-First without that constraint.

Fig. 2 illustrates this schematically (Fig. 4 below justifies the curve's shape). The curve rep-

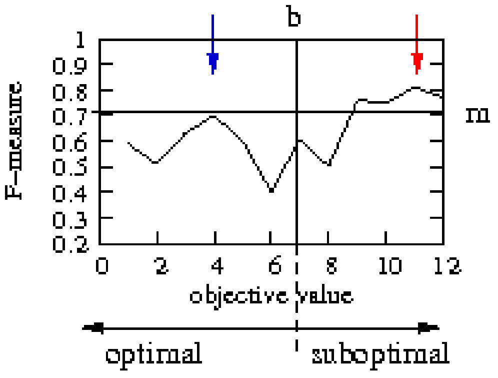

Figure 2: The best solution is 'less optimal'

resents a function mapping objective values to F-measure scores. Note that it is not monotonically decreasing (from lower objective values to higher ones)—as one would expect (less expensive = higher F-measure). The vertical line labelled *b* identifies Balas-First. Starting with Balas-First, optimization searches to the left, i.e. searching for smaller objective function values. The horizontal line labelled *m* shows the local maximum of that search region (the arrow from left points to it). But unfortunately, the global maximum (the arrow from right), i.e. the 1-length solution with all (true) coreferent pairs set to 1, lies to the right-hand side of Balas-First.

This indicates that, in our experimental conditions, optimization efforts can never reach the global maximum, but it also indicates that searching for less expensive solutions nevertheless might lead (at least) to a local maximum. However, if it is true that the goal function is not monotonic, there is no guarantee that the optimal solution actually constitutes the local maximum, i.e. the best solution in terms of F-measure scores.

Unfortunately, we cannot prove mathematically any hypotheses about the optimal values and their behavior. However, we can compare the optimal value's F-measure scores to the Balas-First F-measure scores empirically. Two experiments were designed to explore this. In the first experiment, we computed for each text the difference between the F-measure value of the optimal solution and the F-measure value of Balas-First. It is positive if the optimal solution has higher F-measure score than Balas-First and negative otherwise. This was done for each text (99) that has more than one objective function value (remember that in more than 90% of texts Balas-First was already the optimal solution).

Fig. 3 shows the results. The horizontal line is

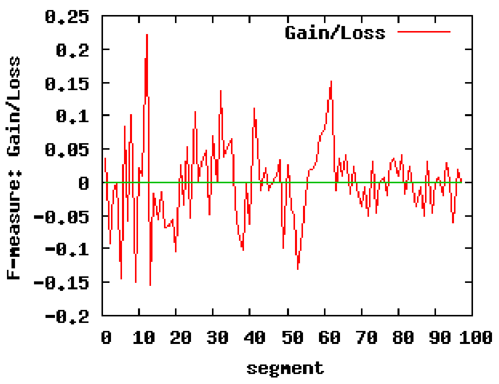

Figure 3: Balas-First or Optimal Solution

separating gain from loss. Points above it indicate that the optimal solution has a better F-measure score, points below indicate a loss in percentage (for readability, we have drawn a curve). Taking the mean of loss and gain across all texts, we found that the optimal solution shows no significant F-measure difference with the Balas-First solution: the optimal solution even slightly worsens the F-measure compared to Balas-First by -0.086%.

The second experiment was meant to explore the curve shape of the goal function that maps an objective function value to a F-measure value. This is shown in Fig. 4. The values of that function are empirically given, i.e. they are produced by our algorithm. The x-axis shows the mean of the nth objective function value better than Balas-First. The y-value of the nth x-value thus marks the effect (positive or negative) in F-measure scores while proceeding to find the optimal solution. As can be seen from the figure, the function (at least empirically) is rather erratic. In other words, searching for the optimal solution beyond Balas-First does not seem to lead reliably (and monotonically) to better F-measure values.

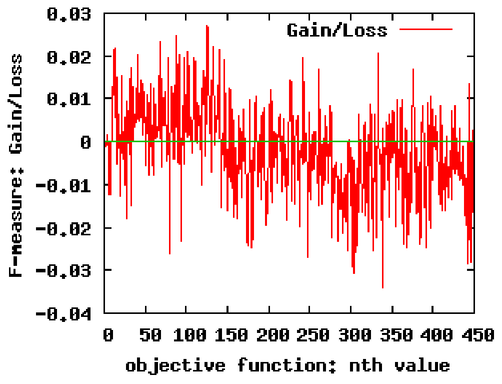

Figure 4: 1st Compared to Balas-nth Value

In the next section, we show that Balas-First as the first optimization step actually is a significant improvement over the classifier output. So we are not saying that we should dispense with optimization efforts completely.

8 Does Balas-First help? Empirical Evidence

Besides the empirical fact that Balas-First slightly outperforms the optimal solution, we must demonstrate that Balas-First actually improves the baseline. Our experiments are based on a five-fold cross-validation setting (1100 texts from the TüBa coreference corpus). Each experiment was carried out in two variants. One where all markables have been taken as input—an application-oriented setting, and one where only markables that represent true mentions have been taken (cf. (Luo et al., 2004; Ponzetto and Strube, 2006) for other approaches with an evaluation based on true mentions only). The assumption is that if only true mentions are considered, the effects of a model can be better measured.

We have used the Entity-Constrained Measure (ECM), introduced in (Luo et al., 2004; Luo, 2005). As argued in (Klenner and Ailloud, 2008), it is more appropriate to evaluate the quality of coreference sets than the MUC score.[9]

To obtain the baseline, we merged all pairs that TiMBL classified as coreferent into coreference sets. Table 2 shows the results.

	all mentions		true mentions	
	Timbl	B-First	Timbl	B-First
F	61.83	64.27	71.47	78.90
P	66.52	72.05	73.81	84.10
R	57.76	58.00	69.28	74.31

Table 2: Balas-First (B-First) vs. Baseline

In the 'all mentions setting', 2.4% F-measure improvement was achieved, with 'true mentions' it is 7.43%. These improvements clearly demonstrate that Balas-First is superior to the results based on the classifier output.

But is the specific order proposed by the Balas algorithm itself useful? Since we have dispensed with 'full optimization', why not dispense with the Balas ordering as well? Since the ordering of the pairs does not affect the rest of our algorithm we have been able to compare the Balas order to the more natural linear order. Note that all constraints are applied in the linear variant as well, so the only difference is the ordering. Linear ordering over pairs is established by sorting according to the index of the first pair element (the i from c_{ij}).

	all mentions		true mentions	
	linear	B-First	linear	B-First
F	62.83	64.27	76.08	78.90
P	70.39	72.05	81.40	84.10
R	56.73	58.00	71.41	74.31

Table 3: Balas Order vs. Linear Order

Our experiments (cf. Table 3) indicate that the

[9]Various authors have remarked on the shortcomings of the MUC evaluation scheme (Bagga and Baldwin, 1998; Luo, 2005; Nicolae and Nicolae, 2006).

Balas ordering does affect the empirical results. The F-measure improvement is 1.44% ('all mentions') and 2.82% ('true mentions').

The search for Balas-First remains, in general, NP-complete. However, constraint models without boundness enforcement constraints (cf. Section 5) pose no computational burden, they can be solved in quadratic time. In the presence of boundness enforcement constraints, exponential time is required in the worst case. In our experiments, boundness enforcement constraints have proved to be unproblematic. Most of the time, the classifier has assigned low costs to candidate pairs containing a relative or a possessive pronoun, which means that they get instantiated rather soon (although this is not guaranteed).

9 Related Work

The focus of our paper lies on the evaluation of the benefits optimization could have for coreference resolution. Accordingly, we restrict our discussion to methodologically related approaches (i.e. ILP approaches). Readers interested in other work on anaphora resolution for German on the basis of the TüBa coreference corpus should consider (Hinrichs et al., 2005) (pronominal anaphora) and (Versley, 2006) (nominal anaphora).

Common to all ILP approaches (incl. ours) is that they apply ILP on the output of pairwise machine-learning. Denis and Baldridge (2007; 2008) have an ILP model to jointly determine anaphoricity and coreference, but take neither transitivity nor exclusivity into account. So no complexity problems arise in their approach. The model from (Finkel and Manning, 2008) utilizes transitivity, but not exclusivity. The benefits of transitivity are thus restricted to an optimal balancing of the weights (e.g. given two positively classified pairs, the transitively given third pair in some cases is negative, ILP globally resolves these cases to the optimal solution). The authors do not mention complexity problems with extensionalizing transitivity. Klenner (2007) utilizes both transitivity and exclusivity. To overcome the overhead of transitivity extensionalization, he proposes a fixed transitivity window. This, however, is bound to produce transitivity gaps, so the benefits of complete transitivity propagation are lost.

Another attempt to overcome the problem of complexity with ILP models is described in (Riedel and Clarke, 2006) (dependency parsing). Here an incremental—or better, cascaded—ILP model is proposed, where at each cascade only those constraints are added that have been violated in the preceding one. The search stops with the first consistent solution (as we suggest in the present paper). However, it is difficult to quantify the number of cascades needed to come to it and moreover, the full ILP machinery is being used (so again, constraints need to be extensionalized).

To the best of our knowledge, our work is the first that studies the proper utility of ILP optimization for NLP, while offering an intensional alternative to ILP constraints.

10 Conclusion and Future Work

In this paper, we have argued that ILP for NLP reduces to Zero-One ILP with unweighted constraints. We have proposed such a Zero-One ILP model that combines exclusivity, transitivity and boundness enforcement constraints in an intensional model driven by best-first inference.

We furthermore claim and empirically demonstrate for the domain of coreference resolution that NLP approaches can take advantage from that new perspective. The pitfall of ILP, namely the need to extensionalize each and every constraint, can be avoided. The solution is an easy to carry out reimplementation of a Zero-One algorithm such as Balas', where (most) constraints can be treated intensionally. Moreover, we have found empirical evidence that 'full optimization' is not needed. The first found consistent solution is as good as the optimal one. Depending on the constraint model this can reduce the costs from exponential time to polynomial time.

Optimization efforts, however, are not superfluous, as we have showed. The first consistent solution found with our Balas reimplementation improves the baseline significantly. Also, the Balas ordering itself has proven superior over other orders, e.g. linear order.

In the future, we will experiment with more complex constraint models in the area of coreference resolution. But we will also consider other domains in order to find out whether our results actually are widely applicable.

Acknowledgement The work described herein is partly funded by the Swiss National Science Foundation (grant 105211-118108). We would like to thank the anonymous reviewers for their helpful comments.

References

E. Althaus, N. Karamanis, and A. Koller. 2004. Computing locally coherent discourses. In *Proc. of the ACL*.

A. Bagga and B. Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at The First International Conference on Language Resources and Evaluation (LREC98)*, pages 563–566.

E. Balas. 1965. An additive algorithm for solving linear programs with zero-one variables. *Operations Research*, 13(4):517–546.

J.W. Chinneck. 2004. Practical optimization: a gentle introduction. Electronic document: `http://www.sce.carleton.ca/faculty/chinneck/po.html`.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2004. TiMBL: Tilburg Memory-Based Learner.

P. Denis and J. Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. *Proceedings of NAACL HLT*, pages 236–243.

P. Denis and J. Baldridge. 2008. Specialized models and ranking for coreference resolution. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP 2008)*, Hawaii, USA. To appear.

J.R. Finkel and C.D. Manning. 2008. Enforcing transitivity in coreference resolution. Association for Computational Linguistics.

E. Hinrichs, K. Filippova, and H. Wunsch. 2005. A data-driven approach to pronominal anaphora resolution in German. In Proc. of RANLP '05.

M. Klenner and É. Ailloud. 2008. Enhancing coreference clustering. In C. Johansson, editor, *Proc. of the Second Workshop on Anaphora Resolution (WAR II)*, volume 2 of *NEALT Proceedings Series*, pages 31–40, Bergen, Norway.

M. Klenner. 2007. Enforcing consistency on coreference sets. In *Recent Advances in Natural Language Processing (RANLP)*, pages 323–328, September.

X. Luo, A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the Bell tree. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*.

X. Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32. Association for Computational Linguistics Morristown, NJ, USA.

T. Marciniak and M. Strube. 2005. Beyond the pipeline: Discrete optimization in NLP. In *Proc. of the CoNLL*.

K. Naumann. 2006. Manual for the annotation of indocument referential relations. Electronic document: `http://www.sfs.uni-tuebingen.de/de_tuebadz.shtml`.

V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In Proc. of the ACL.

C. Nicolae and G. Nicolae. 2006. Best Cut: A graph algorithm for coreference resolution. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 275–283. Association for Computational Linguistics.

C.H. Papadimitriou and K. Steiglitz. 1998. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications.

S.P. Ponzetto and M. Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proc. of HLT-NAACL*, volume 6, pages 192–199.

V. Punyakanok, D. Roth, W. Yih, and D. Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proc. of the COLING*.

S. Riedel and J. Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proc. of the EMNLP*.

W.M. Soon, H.T. Ng, and D.C.Y. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

H. Telljohann, E.W. Hinrichs, S. Kübler, and H. Zinsmeister. 2005. Stylebook for the Tübingen treebank of written German (TüBa-D/Z). *Seminar fur Sprachwissenschaft, Universität Tübingen, Tübingen, Germany*.

Y. Versley. 2006. A constraint-based approach to noun phrase coreference resolution in German newspaper text. In Konferenz zur Verarbeitung Natürlicher Sprache (KONVENS).